Resolving metagenomes using single-molecule linked-read sequencing

JENNIFER THELAND
Abstract

The development of Massively Parallel Sequencing (MPS) has enabled more accurate and less time-consuming DNA sequencing. Although MPS technologies are theoretically applicable to all samples and species, the majority of studies on microorganisms have been conducted on those able to be isolated and cultivated in laboratories. In the field of metagenomics, DNA from uncultivated environmental samples is analyzed. Whole genome sequencing of such complex samples poses difficult computational challenges due to the characteristics of metagenomic data, where one major challenge lies in determining the true origin of high similarity reads. In addition, the short-range information acquired from MPS reveals little about how reads from DNA sequencing fit together. Consequently, producing genome drafts from reads generated by MPS remains difficult. Here, the linked-read sequencing technology DB-Seq has been applied to bacterial samples in order to assess its potential in metagenomics. Specifically, its performance in retaining long-range information in de novo whole genome assembly has been tested. The results obtained in this initial study show great potential of DB-Seq in genome assembly, with significantly more contiguous results than conventional methods generate.

Keywords: DNA sequencing, linked-read sequencing, DB-Seq, metagenomics, de novo assembly
Contents

1 Introduction
   1.1 The history of DNA, genetics and metagenomics ............................. 1
   1.2 The challenge of resolving metagenomes ..................................... 2
   1.3 Aim and purpose ................................................................. 3

2 Materials and Methods .............................................................. 4
   2.1 Single-molecule linked-read sequencing ...................................... 4
      2.1.1 DB-Seq - Method description ........................................... 4
      2.1.2 Sample preparation and DNA extraction ................................ 4
      2.1.3 Droplet Barcode Sequencing - Laboratory protocol .................. 5
   2.2 Whole genome assembly .......................................................... 7
      2.2.1 Algorithm descriptions ................................................... 7
      2.2.2 Whole genome assembly - Pipelines .................................... 8
      2.2.3 Analysis and statistics .................................................. 12

3 Results ......................................................................................... 14
   3.1 De novo whole genome assembly .............................................. 14
      3.1.1 One species - E. coli BL21 ............................................... 14
      3.1.2 Two species - E. coli BL21 and R. eutropha H16 ..................... 16
      3.1.3 Aquarium water sample .................................................... 18
   3.2 Sequencing depth .................................................................... 19
      3.2.1 Uneven sequencing depth .................................................. 20
      3.2.2 Minimum sequencing depth ............................................... 20

4 Discussion .................................................................................. 23

5 Future Perspectives ........................................................................ 25

6 Acknowledgements ......................................................................... 25

7 Appendix ..................................................................................... 29
   7.1 Genome references ................................................................. 29
   7.2 Open-source programs and customized scripts ............................ 29
   7.3 Customized scripts .................................................................. 30
Dictionary

Metagenomics The study of all genetic material in environmental samples.

Sequencing read The data obtained from DNA sequencing. In short-read sequencing, a read is usually 150-300 bases long.

Paired-end sequencing DNA sequencing when fragments are read from both ends, producing two sequencing reads.

Sequencing depth The number of times each base in a genome is sequenced.

Interleaved read file A merged file wherein paired reads appear directly after each other.

Genome assembly To join sequencing reads into a genome.

Contig The sequence data resulting from an assembly, i.e. reads are assembled into contigs.

De novo assembly To assemble a genome without a reference.

Scaffolding To arrange assembled contigs in their correct order (relative each other).

Scaffold The sequence data resulting from scaffolding, i.e. contigs are merged into scaffolds.

NG50 value The size of the smallest contig when 50 % of the bases in the reference genome are covered.

N50 value The size of the smallest contig when 50 % of the total assembly length is covered.

Tagmentation A fragmentation method where bead-linked transposomes are used to cut DNA.

Read pair duplicates Read pairs being identical due to amplification of the sequencing library. Since duplicate read pairs confer the same data, they are uninformative.

Scaffold graph A graph illustrating possible orientations of contigs relative each other. Each contig is represented as a node and edges link nodes having evidence supporting their connection.

Misassembly An event wherein a part of the assembly does not agree with the reference sequence. A misassembly is here referred to as local when the divergence from the reference is less than 1 kb long.
Introduction

1.1 The history of DNA, genetics and metagenomics

Despite that the inheritance of traits through generations has been suspected since the ancient Greece [1], DNA was not discovered until the mid-nineteenth century. Although it was clear that DNA differed greatly from other organic substances isolated from cells, its significance and function remained unclear for decades. When chromosomes were established as the carriers of hereditary information, many accredited proteins the heritable properties due to DNA still being relatively unexplored and proteins having displayed high specificity. Conclusive evidence for DNA as the genetic material was not presented until 1944, when Oswald Avery and colleagues performed experiments with strains of *Pneumococcus* bacteria. In the years to follow, major breakthroughs were made in the field of genetics, among the most important ones being the establishment of the DNA double helix structure, the stating of the central dogma and the development of DNA sequencing [2].

DNA sequencing has in the last decades been the subject of major technological development. The ability to determine the order of nucleotides within a DNA sequence has enabled important discoveries in virtually all areas of biological research [3]. Since its entrance into the field, it has been employed in numerous studies and much knowledge has been gathered about the characterization and cellular operations of microorganisms [4]. However, the majority of studies on microorganisms have been focused on those able to be isolated and cultivated in laboratories, despite present day sequencing technologies being theoretically applicable to all samples and species. Naturally, characterization of microorganisms is less complicated in pure cultures than in mixed samples, but given that 99% of all microorganisms are estimated to be uncultivable, a substantial part of earth’s microbiome remains undiscovered [5] [6].

In 1986, Norman Pace and colleagues presented one of the first approaches for sequencing nucleic acids from uncultivated samples of mixed species. In the approach, DNA was extracted from natural biomass before cloned into bacteriophages. The recombinant DNA libraries were enriched for 16S rRNA clones before sequencing. The results were compared to reference databases in order to determine the phylogeny and quantity of microorganisms within the sample. The decision to solely base the analysis on 16S rRNA was motivated by (1) the sequence homology across organisms and (2) the conservation of sequence throughout evolution [7]. However, the high similarity between 16S rRNA genes of closely related species as well as the occurrence of horizontal gene transfer contribute to uncertainty in the inferences from this type of targeted analysis [8]. Furthermore, this approach introduces limitations for identification and analysis of previously unannotated species [9].

The development of Massively Parallel Sequencing (MPS) technologies has enabled new strategies in research and provided opportunities for more comprehensive and less time-consuming DNA sequencing [10]. In the field of metagenomics, MPS enables high-throughput and parallel whole-genome analyses of species lacking homologous DNA, e.g. bacteria and viruses [11]. An analysis based on the whole genome rather than just a single gene generates more data and enhances detection of species while simultaneously providing more comprehensive knowledge and understanding [12]. For previously unstudied microorganisms, MPS can theoretically acquire information for characterization and
charting of an entire genome in a single experiment [13].

Although the development of MPS has resulted in accelerated and less laborious DNA sequencing, it has also raised new challenges. Results obtained from MPS not only introduce difficulties due to the large amount of generated data, but also holds limitations with regards to the type of information it confers. The short read length in MPS produces highly discontinuous and scattered data, with little information about how reads fit together [14]. Consequently, producing genome drafts from reads generated by MPS is difficult [15][16]. As a reaction to the limitations of MPS, several new sequencing technology platforms and approaches have emerged in the last decades [14].

1.2 The challenge of resolving metagenomes

Many approaches with potential of solving the challenges introduced by MPS have been established in recent years. Generally, these methods can be divided into two categories: single molecule long-read sequencing and linked-read sequencing methods [17]. The former includes Pacific Biosciences’ Single Molecule Real-Time (PacBio SMRT) sequencing platform [18] and Oxford Nanopore [19], both having the advantage of producing long blocks of phased information. However, the error rate of such methods remains high and a solution to this is yet to come [20]. Therefore, efforts have been devoted to developing linked-read sequencing technologies where highly accurate MPS platforms are combined with specialized library preparation protocols. Generally, these protocols include attaching a barcode sequence to all fragments originating from the same DNA molecule. The DNA molecules, usually a few kilobases in length, correspond to long blocks of genomic DNA. The reads generated by these methods can effectively be traced to their molecule of origin using the barcode sequence and thereby, long-range information able to facilitate e.g. genome assembly is acquired [21].

10x Genomics has developed one such linked-read sequencing technology that has been applied in in whole-genome haplotyping studies with great success [22]. However, this technology is associated with significant costs, both with regards to instrumentation as well as consumable kits [23]. For metagenomic purposes, further issues arise with the number of DNA molecules sharing the same barcode. Having DNA molecules from different species share the same barcode signals they have the same origin and consequently, whole genome assembly is aggravated [24]. Although this problem potentially could be solved by repeated experiments, the costs are a bottleneck. The shortcomings of the technology leaves more to be desired for metagenome assembly purposes, especially with regards to pricing.

Droplet Barcode Sequencing (DB-Seq) is a newly developed linked-read sequencing technology, previously applied to targeted phasing but now further developed for whole genome purposes. The method is based on separating DNA molecules into droplets wherein fragmentation and barcode coupling are performed. DB-Seq generates droplets by shaking and is thereby independent of specialized instrumentation. The reagents used are cheaper than similar technologies, consequently, the cost of analysis is significantly less than e.g. 10x Genomics. The low cost further allows for reduction of the number of DNA molecules sharing the same barcode to ~1, without introducing significant pricing bottlenecks [25]. For the purpose of metagenome assembly using linked-read sequencing, this feature greatly facilitates the task.

To date, the application of linked-read sequencing technologies in resolving
metagenomes is relatively unexplored, with only a few articles having been found reporting on the subject [24][26].

1.3 Aim and purpose

The development of DNA sequencing has enabled major biological discoveries, but the inability to culture and isolate all kinds of microorganisms leaves much to be discovered. Although technologies such as Nanopore, PacBio and 10x Genomics display great potential, bottlenecks like as error rate and pricing remain. The newly developed DB-Seq has the ability to uniquely label DNA molecules while keeping the price low.

In this master’s thesis project, the potential of DB-Seq in metagenomics will be investigated. Specifically, the ability of barcodes to retain information of read origins will be tested in de novo whole genome assembly. Furthermore, conventional de novo assembly without long-range information will be carried out in order to benchmark the method to existing platforms.
2 Materials and Methods

This project can generally be divided into two parts; single-molecule linked-read sequencing of bacterial samples (Section 2.1) and whole genome assembly of the generated sequencing data (Section 2.2). In Section 2.1.1, an overview of DB-Seq is presented. Sections 2.1.2 and 2.1.3 describe sample preparation and the laboratory protocol of DB-Seq, respectively. Descriptions of algorithms implementing barcode information are found in Section 2.2.1 and data analysis pipelines are presented in Section 2.2.2.

2.1 Single-molecule linked-read sequencing

2.1.1 DB-Seq - Method description

DB-Seq is a linked-read sequencing method where barcode oligonucleotides are coupled to sample DNA in order to acquire long-range genomic information. Specifically, the barcodes confer information about the DNA molecule of origin for sequencing reads, where reads harboring the same barcode are assumed to originate from the same DNA molecule.

In DB-Seq, genomic DNA is fragmented using on-bead tagmentation wherein DNA is cut by transposome enzymes linked to magnetic beads. Following tagmentation, DNA remains bound to the beads thereby hindering fragmented DNA with different origin to mix. The DNA-covered beads are emulsified in droplets by forcing an aqueous and an oil phase to blend. In addition to the bead complexes, barcode oligonucleotides together with reagents required for barcode-coupling are present in the compartments. In theory, each droplet will hold a single DNA molecule and one unique barcode. The barcode within each droplet is amplified before coupled to the genomic DNA in an emulsion PCR (emPCR). As a result, all DNA fragments originating from the same droplet are covalently attached to identical barcodes.

The droplets in DB-Seq are generated independently of specialized instrumentation. However, controlling the distribution of reagents within the droplets is difficult and as a consequence, some may hold only the barcode oligonucleotide while others simply harbour the DNA-covered bead. The products of such droplets are removed in purification and enrichment of the DNA sequencing library.

The emulsions are broken before DNA sequencing is performed using MPS technology from Illumina. An overview of DB-Seq is illustrated in Figure 1.

2.1.2 Sample preparation and DNA extraction

High Molecular Weight (HMW) genomic DNA was extracted by the MagAttract HMW DNA kit (Qiagen, MD, USA) and quantified using Qubit 3.0 (Life Technologies). Since this is the first application of DB-Seq in metagenomics, the complexity of all samples was reduced by either isolation or cultivation prior to DNA extraction.

E. coli - BL21

The DB-Seq protocol had been performed on 25 pg HMW DNA prior to the initiation of this project. However, no attempt had been made to de novo assemble the genome by implementing barcode information.

E. coli - BL21 & R. eutropha - H16

A cultivated sample of R. eutropha was provided by the group of assistant professor
Paul Hudson (KTH/SciLifeLab, Solna).

Equal amounts of extracted HMW DNA from *R. eutropha* and *E. coli* were pooled and a final input amount of 25 pg was used in the protocol.

**Figure 1:** Overview of Droplet Barcode Sequencing. Genomic DNA is fragmented using transposome-covered beads. The beads, together with barcode oligonucleotides, are emulsified by allowing an oil and an aqueous phase to mix. The barcodes are amplified before coupled to the genomic DNA in an emPCR. The emulsions are broken and the library is enriched for successfully created amplicons. The enriched library is subjected to short-read sequencing using technology from Illumina.

**Aquarium water**

Water from an aquarium was cultivated by the group of Gunaratna Kuttuva Rajarao (KTH, Stockholm). Two HMW DNA extractions were performed; one for gram positive and one for gram negative bacteria. Both samples were run in the Agilent 2100 bioanalyzer in order to determine whether the fragments were long enough for linked-read sequencing. The results showed fragments within a tolerable interval for the gram negative DNA sample, while the fragments from the gram positive were decided as too short to continue with. 1 ng from the gram negative sample was used as input.

### 2.1.3 Droplet Barcode Sequencing — Laboratory protocol

**On-bead tagmentation**

Genomic DNA was fragmented according to the Nextera DNA Flex Library Prep protocol (Illumina, San Diego, CA, USA). However, instead of amplifying the tagmented DNA as specified, the DNA-covered beads were incubated at 68°C for 10 minutes (Mastercycler Pro S, Eppendorf, Hamburg, Germany) without the addition of Nextera DNA Flex Indexes.

The beads were washed twice with Tagment Wash Buffer (Illumina) and resuspended in Elution Buffer (Qiagen).

**Emulsification and emPCR**

100 μl HFE7500 oil with 5%(w/v) 008-Fluorosurfactant (Ran Biotechnologies, MA, USA) was added to a Qubit tube (Life Technologies). The tube was vortexed thoroughly (upright and upside down) before 50 μl emPCR mix (Milli-Q, 1M Betaine (Sigma Aldrich, MO, USA), 3 %vol DMSO (Thermo Scientific, MA, USA), 400 nM Enrichment Oligo, 80 nM Coupling Oligo, 330 fM Barcoding Oligo (Integrated DNA Technologies, USA), 2%wt PEG-6000 (Sigma Aldrich), 2%vol Tween-20 (Sigma Aldrich), 1x...
Phusion Flex (New England Biolabs, UK), 5 µl beads with tagmented DNA) was added on top of the oil. The oil and aqueous phase were emulsified by shaking for 8 minutes at 15 Hz (Tissuelyser, Qiagen). After emulsification, the tube was placed upright for 15 minutes. The entire emulsion phase was transferred to the bottom of a thin-walled PCR tube holding 60 µl FC-40 oil with 5% (w/v) Fluorosurfactant (Ran Biotechnologies). 85 µl filtered mineral oil (Sigma Aldrich) was added on top of the emulsion phase before PCR according to 95°C 5 min, 30 cycles of 95°C 30s (50 % ramp) - 55°C 30s (40 % ramp) - 72°C 30s (20 % ramp), 7 cycles of 95°C 1 min (20 % ramp) - 40°C 2 min - 72°C 5 min (3 % ramp), 95°C 1 min (20 % ramp), 40°C 5 min, 72°C 15 min (3 % ramp), 20°C 1 min, finish at 12°C (Mastercycler Pro S).

Emulsion breakage

The mineral oil was removed and 4 µl EDTA (100 mM) (Invitrogen, CA, USA) was added before the entire reaction was transferred to a 0.5 ml DNA LoBind tube (Eppendorf). In order to break the emulsions, 100 µl 1H,1H,2H,2H-Perfluoro-1-octanol (Sigma Aldrich) and 5 µl Elution Buffer were added to the tube before vortexing at maximum speed for 10 seconds. To separate the phases, the tube was centrifuged for 4 minutes at 25 000 x g. The aqueous phase was separated from the oil phase before the tagmentation beads were removed using a magnet.

Purification and enrichment

A size selection was performed in order to remove barcode sequences lacking sample DNA and sample DNA not coupled to barcodes. The former was accomplished by polyethylene glycol precipitation on carboxylic acid beads according to Lundin et. al (CA-purification) [27]. Fragments without barcodes were removed in an enrichment step, where the presence of a biotin molecule on successfully created amplicons enabled the use of streptavidin-covered beads.

Dynabeads MyOne Streptavidin T1 beads (Life Technologies) were washed twice with 2x Bind&Wash¹ before resuspended in 3x Bind&Wash. 10 µl bead solution and 20 µl CA-purified product were added to a fresh tube before the mixture was incubated in room temperature under rotation for 40 minutes.

The supernatant was discarded and the beads washed twice with Elution Buffer, four times with 0.125 M NaOH (Sigma Aldrich) and twice in Elution Buffer with 0.05% Tween20.

PCR extension and amplification

In order to release the DNA from the streptavidin-coated beads, a PCR indexing reaction was performed. 1x Phusion Flex and 400 mM Nextera DNA Flex index (Illumina) were mixed with the streptavidin beads before PCR according to: 95°C 2 min, 65°C 10 s (20 % ramp), 55°C 3 min (3 % ramp), 72°C 10 min (3 % ramp), 95°C 1 min 15 s. At this step, the reaction was paused and the supernatant transferred to a fresh tube. Before the reaction was continued, an equimolar amount of i5 adapter oligonucleotide (Integrated DNA Technologies) was added to the supernatant. The reaction was continued with 4 cycles of 95°C 30 s (20 % ramp) - 55°C 30 s (10 % ramp) - 72°C 1 min (20 % ramp), 72°C, 2 min (Mastercycler Pro S).

The final product was subjected to two rounds of CA-purification before quantification.

¹1x Bind&Wash (1 M NaCl, 5 mM Tris-HCl, 500 µM EDTA)
2 MATERIALS AND METHODS

Sequencing

The DNA sequencing libraries were diluted before they were sequenced using paired-end technology (MiSeq, Illumina).

2.2 Whole genome assembly

The whole genome assembly pipeline developed in this project utilizes open-source software, parts of an in-house automation script\(^2\) and newly developed programs created specifically for this project. Since the implementation of linked-reads in whole genome assembly is in focus, only software specialized on this will be explained in detail in this section. However, the commands used to run all programs are specified for the entire pipeline and a brief description of all software can be found in Section 7.2.

2.2.1 Algorithm descriptions

Athena-meta assembler

*Athena-meta* is a whole genome assembly software, originally developed for linked-reads from 10x Genomics [24].

Sequencing reads in an interleaved file are assembled using the de Bruijn graph-based short read assembler *IDBA-UD* [28], without implementing barcode information. The resulting contigs, referred to as seed contigs, make up an initial draft of the metagenome. The seed contigs are further used as reference sequences in a *BWA-MEM* [29] read alignment, where the same reads used to create the seeds are mapped back to the contigs. The mapping confers information regarding what barcode clusters lie in proximity to each other. All seed contigs are represented in a scaffold graph where edges connect contigs having read pairs spanning over them. To avoid spurious connections, a minimum of three read pairs must be shared between any two seeds for an edge to be considered. Furthermore, reads laying within a 500 bp neighbourhood and span the same two seeds are grouped. If a single group contains at least 50% of all spanning read pairs of a seed, an edge is introduced in the scaffold graph. Branches are introduced when ambiguities arise.

In order to ultimately connect seed contigs linked in the scaffold graph, the barcode information is implemented. All barcode clusters with at least one mapped read within a 10kb region on either side of an edge are passed to a short-read assembler. That is, all reads belonging to these barcode clusters are assembled in a *de novo* fashion. This divides the whole-genome assembly problem into smaller subassemblies. The subassembled contigs and the seed contigs are treated as long reads in a final step using the long-read assembler *Flye* [30].

Arcs/LINKS scaffolding

*LINKS* is a scaffolding software originally developed for long-read sequencing data produced by platforms such as PacBio SMRT and Oxford Nanopore. When combined with *Arcs*, a program created by the same developers, *LINKS* is also applicable to linked-read sequencing data.

Linked-reads are mapped onto the assembled genome. The output file is processed in *Arcs* and read pairs within the same barcode cluster are used for orientation of contigs. Since the main objective is to link ends of contigs, only complete read pairs inside specified end-regions are considered. In order to avoid spurious links, a minimum number of read pairs are required for the support to be registered. To decide the most probable orientation of two contigs, the number of supporting barcodes are counted. That is, if

\(^2\)GitHub: https://github.com/FrickTobias/WGH_Analysis/blob/master/WGH_automation.sh
two contigs share many barcodes, the support will be high. The orientation with the most support is recorded to an output file [31].

The contigs in the output file are joined using the LINKS algorithm. For merging to occur, the links between two contigs must exceed a minimum value. In addition, the ratio of the number of links between the most supported and the second most supported orientation must be sufficiently large [32].

2.2.2 Whole genome assembly - Pipelines

Read processing

All adaptors and barcode sequences attached in the emPCR were removed from the sequencing reads with commands as specified in the in-house automation script WGH_automation.sh, wherein the software Cutadapt [33] and UMI-tools [34] are employed (Box 1).

Box 1. Adaptor removal and barcode extraction.

cutadapt \\
  -g/-a/-A adaptor_sequence \\
  -o R1_trimmed.fq \\
  -p R2_trimmed.fq" \\
R1_untrimmed.fq \\
R2_untrimmed.fq \\
--discard-untrimmed -e 0.2 \\
-m 65

Where -g/-a/-A is used depending on what end of a read an adapter is located, -e is the allowed error rate, -m is the minimum allowed length of a read and --discard-untrimmed signals to discard all untrimmed reads.

umi_tools extract \\

--stdin=R1_untrimmed.fq \\
--stdout=R1_trimmed.fq \\
--bc-pattern=NNNNNNNNNNNNNNNNNNNN \\
--bc-pattern2= \\
--read2-in=R2_untrimmed.fq \\
--read2-out=R2_trimmed.fq

Since the barcode sequences are unknown, the barcode pattern is specified as random. The raw barcode sequences are extracted from read 1 and stored in the read header of both reads in the pair.

Since a single sequencing error within a barcode will result in a similar but different sequence, a read pair holding a mis-sequenced barcode will be assumed belonging to a separate cluster. In such a case, the information of origin will be lost. To account for such sequencing errors, cd-hit-454 was used to cluster barcode sequences if the sequence identity exceeded 90 %. Note, before this step, all barcode sequences were placed in a separate file which was further split into many, smaller sub-files. This was performed due to the inability of cd-hit-454 to handle large files. The sub-files were categorized according to the first three bases of the barcode sequence. That is, all barcodes with the same first three bases were placed in the same sub-file. Both the preparation of sub-files and the barcode clustering were performed as specified in WGH_automation.sh (Box 2).

Box 2. Barcode clustering.

python3 cdhit_prep.py \\
R1_trimmed.fq \\
cdhit_prep_output_folder \\
-r 3

Where -r is the number of bases used as index for dividing barcode sequences into subfiles.

for file \

8
in cdhit_prep.output_folder/*.fa do
  cdhit-454 -i $file -o clustered_out -c 0.9 -gap 100 -g 1 -n 3 -M 0
done

Where \(-c\) is the required identity for clustering, \(-\text{gap}\) is the gap opening score, \(-g\) specifies which cluster a sequence will be assigned to, \(-n\) is the word length and \(-M\) is the memory limit.

All clustered sub-files were merged into a single file before continuing the analysis.

The raw barcode sequences in the read header were exchanged for the consensus sequence of its cluster by cross-referencing with the merged cluster file. The consensus barcodes were stored in the header according to the format:

\[
BC : Z : (NN)_{10} - 1
\]

This way of storing the barcode sequence is concordant with that of 10x Genomics and thereby enables use of software developed specifically for their data. In addition, it enables the barcode to be appended to the output in the subsequent read alignment steps. The consensus-tagged read files were sorted according to barcode sequence and thereafter merged into an interleaved read file using `reformat.sh` from `BBMap` (Box 3) [35]. The tagging and sorting operations were performed using customized scripts written in this project for these specific purposes\(^3\) (Section 7.3).

Box 3. Tagging, sorting and merging of read files.

```bash
python3 tag_fastq.py RX_trimmed.fq barcode_clusters.clstr RX_trimmed_tagged.fq
bash sort_file.sh RX_trimmed_tagged.fq RX_trimmed_tagged_sorted.fq
reformat.sh in=R1_trimmed_tagged_sorted.fq out=R1_R2.fq
reformat.sh in=R1_R2.fq out=R1_R2.fa
```

The read processing pipeline is illustrated in Figure 2.

\(^3\)Note: Parts of the customized scripts were already available in the regular analysis pipeline used for DB-Seq. These parts have been directly implemented in the ones created here but have been marked in the code for clarity.
Figure 2: All adaptor sequences are removed and discarded from both read files. The barcode sequence is extracted from the read 1 (R1.fq) file and stored with both reads in the pair. In addition, a separate file with all barcodes is created and divided into many, smaller subfiles according to the first three bases in the sequence (CAA.fa, CAT.fa, GAT.fa, etc.). The barcodes in each file are clustered if they display a similarity > 90 % and further stored in their corresponding NNN.clstr file. For each new cluster, one of the barcodes is randomly assigned as the consensus sequence. The NNN.clstr files are concatenated into one file (barcodes.clstr) before the trimmed read files are tagged with the cluster consensus barcodes. That is, the raw barcode sequence is replaced with the consensus sequence for its cluster and stored with the reads in a different format. The tagged files are merged into an interleaved read file and sorted according to barcode sequence.
Whole genome assembly

The interleaved read file was passed into *IDBA-UD* to generate the first assembled draft of the metagenome\(^4\) (Box 4).

**Box 4. Conventional short-read assembly.**

```
idba-ud -r R1_R2.fa -o seed_contigs
```

The reads were mapped back to the draft using *BWA-MEM* and the output file was sorted with *SAMtools* (Box 5) [36].

**Box 5. Mapping to genome draft.**

```
bwa mem -C -p seed_contigs.fa R1_R2.fq | SAMtools sort -o mapped_reads.bam -
```

Options `-p` and `-C` are used to signal the input file is interleaved and that the barcode should be appended to the output file, respectively.

An index to the mapped read file was created using *SAMtools* before *Athena-meta* was run (Box 6).

**Box 6. Barcode assembly.**

```
athena-meta config.json
```

Where config.json specifies the paths to (1) the seed_contigs.fa (2) mapped_reads.bam and (3) R1_R2.fq.

Whole genome scaffolding

The reads from the original interleaved read file were mapped to the assembled contigs from *Athena-meta* before the support for each orientation of the contigs was computed by *Arcs*. The output file was converted before the scaffolds were generated with *LINKS* (Box 7).

**Box 7. Barcode scaffolding.**

```
bwa mem -C -p athena_contigs.fa R1_R2.fq | SAMtools sort -n -o mapped_reads.bam -
```

Where `-n` signals the reads in mapped_reads.bam will be sorted according to read name, `-C` and `-p` options are used as previously described.

```
arcs -f athena_contigs.fa -a bamfiles.txt -e 5000
```

Where `-a` specifies a .txt file with the mapped_reads.bam file and `-e` the length of the head-tail region.

```
python makeTSVfile.py file.gv file.tsv athena_contigs.fa
touch empty.fof
LINKS -f athena_contigs.fa -s empty.fof -k 20 -b files_from_arcs -l 5
```

Where `-k` is the k-mer value, `-b` is the base name for the output files and `-l` is the minimum required links.

In order to enable full comparison between

\(^4\)Note: This corresponds to conventional *de novo* whole genome assembly.
whole genome assembly with and without barcode information, the seed contigs generated by *IDBA-UD* were further scaffolded by the conventional scaffolder *SSpace* (Box 8) [37].

**Box 8. Conventional scaffolding.**

```plaintext
perl SSPACE_Basic.pl \
-l read_files.txt \ 
-s seed_contigs.fa \ 
-b output_directory
```

Where `-l` specifies a .txt file with the trimmed read files.

To facilitate the data analysis, an automation script for the entire pipeline was created (Section 7.3, `athena_assembly_automation.sh`). The whole genome assembly and scaffolding pipeline is illustrated in Figure 3.

### 2.2.3 Analysis and statistics

Statistics on the whole genome assemblies were generated using the software *MetaQUAST* [38] together with reference genomes for relevant species (Section 7.1). Furthermore, *BEDTools* [39] was used to calculate the sequencing depth for all samples but the unknown one. Additional customized scripts have been used for operations such as calculation of N50 values and G/C-content in zero coverage regions.
2 MATERIALS AND METHODS

Conventional short read assembly / Mapping to genome draft

Barcode assembly

Barcode scaffolding

Figure 3: Overview of the whole genome assembly and scaffolding pipeline. **Conventional short read assembly.** Reads (R1_R2.fa) are assembled using a conventional short read assembler before barcoded reads (R1_R2.fq) are mapped onto the resulting contigs (seed_contigs.fa). **Barcode assembly.** The mapped reads (mapped_reads.bam) are used to generate edges between contigs, where an edge is formed if two contigs share a read pair. All reads belonging to a barcode cluster laying within a 10 kb region on either side of an edge are passed into a short read assembler. The contigs generated in the subassemblies (barcode_contigs.fa) and the initial seeds are passed into a long-read assembler to generate the final assembly. **Barcode scaffolding.** Barcoded reads (R1_R2.fq) are mapped onto the barcode assembled contigs (athena_contigs.fa), only read pairs on the edge (head (H)/tail (T)) of contigs are considered. The number of supporting read pairs for each orientation is registered and the most probable result is used.
3 Results

3.1 De novo whole genome assembly

De novo whole genome assembly was performed on samples with one, two and an unknown number of species. In Section 3.1.1, the results from using conventional and barcoded assembly strategies on a single species are presented and compared. In the following two sections (3.1.2, 3.1.3), results from the same pipeline on samples holding two and an unknown number of species are found.

3.1.1 One species - E. coli BL21

The E. coli sample, sequenced to a depth of 190X, was assembled in two steps using IDBA-UD and Athena-meta, respectively. In the first step, where the barcode information was not taken into account, the assembly resulted in 190 contigs covering approximately 99.3% of the reference genome. Applying the long-range information in Athena-meta reduced the number of contigs to 18 while increasing the percentage of the reference genome covered. Furthermore, the NG50 value increased from 75,053 to 708,772. Neither IDBA-UD nor Athena-meta generated any misassemblies, however, both software produced local misassemblies where smaller regions disagreed with the reference. Closer investigation of these revealed they were primarily located within repeat regions.

Scaffolding of the IDBA-UD contigs using SSpace produced 131 scaffolds. Furthermore, the genome coverage was slightly reduced compared to before the operation while both the NG50 value and the number of misassemblies increased. Using long-range information, scaffolding merged the 18 contigs generated by Athena-meta into 8. Due to one of the scaffolds covering ~ 85% of the reference genome, the NG50 value increased drastically. The genome coverage remained the same while the number of local misassemblies increased. It is worth noting that when two contigs are merged, a random string of N’s is inserted into the assembled sequence. If the length of this string exceeds a certain threshold, it will give rise to a local misassembly. All statistics regarding the sequencing and the whole genome assembly are summarized in Table 1 and 2.

Table 1: Overview of genome size and sequencing statistics.

<table>
<thead>
<tr>
<th>Sequencing coverage and genome information</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Paired reads</td>
<td>15,102,643</td>
</tr>
<tr>
<td>Read duplicates</td>
<td>71%</td>
</tr>
<tr>
<td>Genome size (bp)</td>
<td>4,528,118</td>
</tr>
<tr>
<td>Sequencing depth</td>
<td>190X</td>
</tr>
</tbody>
</table>

Note: the samples of mixed species were not subjected to conventional scaffolding, only scaffolding using linked-reads.
Table 2: Overview of assembly and scaffolding statistics.

<table>
<thead>
<tr>
<th>Contigs/scaffolds</th>
<th>IDBA-UD</th>
<th>Athena-meta</th>
<th>SSpace</th>
<th>Arcs/LINKS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contigs/scaffolds</td>
<td>190</td>
<td>18</td>
<td>131</td>
<td>8</td>
</tr>
<tr>
<td>NG50 (bp)</td>
<td>75,053</td>
<td>708,772</td>
<td>115,062</td>
<td>3,812,784</td>
</tr>
<tr>
<td>Genome coverage (%)</td>
<td>99.3</td>
<td>99.7</td>
<td>99.2</td>
<td>99.7</td>
</tr>
<tr>
<td>Misassemblies + local</td>
<td>0 + 2</td>
<td>0 + 10</td>
<td>1 + 57</td>
<td>0 + 18</td>
</tr>
<tr>
<td>misassemblies</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The whole genome assemblies with and without implementing barcodes are illustrated in Figure 4. Genomic positions are marked at the edge of the plot where the sequencing depth at each specific position is represented in a histogram. The conventional assembly and scaffolding are visualized by the two gray circles, where the innermost corresponds to the contigs assembled by IDBA-UD and the outer to the scaffolds from SSpace. The colored plots represent the assembly and the scaffolding wherein linked-reads were used.

Figure created with Circos [40].
3.1.2 Two species - *E. coli* BL21 and *R. eutropha* H16

The pooled sample with genomes from *E. coli* and *R. eutropha* held a total of four genomic molecules of varying sizes. Although the average depth of sequencing was 16X, separate calculations for each of the genomic molecules revealed uneven coverage between the two species, where the *E. coli* genome was sequenced approximately 4 times deeper than *R. eutropha* (Table 3).
The consequence of the uneven sequencing depth between the two species was observed in the whole genome assembly statistics. At 38X depth, the *E. coli* genome was assembled into 47 and scaffolded into 34 parts, both covering 99.2% of the reference genome. Without applying the long-range information, the same sequencing reads were assembled into 181 contigs while covering the same extent of the reference.

At 10X sequencing depth, the assemblies of *R. eutropha* chromosome 1 and 2 primarily generated small contigs with NG50-values < 2,000 bases. Furthermore, the number of misassemblies (both global and local), were higher than for the *E. coli* genome and the *R. eutropha* megaplasmid.

A summary of all results from the whole genome assembly can be observed in Table 4.
### Results

#### Table 4: Overview of sequencing and assembly statistics.

<table>
<thead>
<tr>
<th>Assembly and Scaffolding</th>
<th>IDBA-UD</th>
<th>Athena-meta</th>
<th>Arcs/LINKS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>E. coli - BL21</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Contigs/scaffolds</td>
<td>181</td>
<td>47</td>
<td>34</td>
</tr>
<tr>
<td>NG50 (bp)</td>
<td>81,584</td>
<td>223,716</td>
<td>431,360</td>
</tr>
<tr>
<td>Genome coverage (%)</td>
<td>98.6</td>
<td>99.2</td>
<td>99.2</td>
</tr>
<tr>
<td>Misassemblies + Local misassemblies</td>
<td>0 + 2</td>
<td>1 + 8</td>
<td>1 + 14</td>
</tr>
<tr>
<td><strong>R. eutropha - H16 Chromosome 1</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Contigs/scaffolds</td>
<td>2,479</td>
<td>947</td>
<td>946</td>
</tr>
<tr>
<td>NG50 (bp)</td>
<td>1,759</td>
<td>1,809</td>
<td>1,811</td>
</tr>
<tr>
<td>Genome coverage (%)</td>
<td>82.4</td>
<td>62.6</td>
<td>62.6</td>
</tr>
<tr>
<td>Misassemblies</td>
<td>84 + 25</td>
<td>48 + 1</td>
<td>49 + 1</td>
</tr>
<tr>
<td><strong>R. eutropha - H16 Chromosome 2</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Contigs/scaffolds</td>
<td>1,839</td>
<td>670</td>
<td>668</td>
</tr>
<tr>
<td>NG50 (bp)</td>
<td>1,704</td>
<td>1,764</td>
<td>1,764</td>
</tr>
<tr>
<td>Genome coverage (%)</td>
<td>82.6</td>
<td>61.6</td>
<td>61.6</td>
</tr>
<tr>
<td>Misassemblies</td>
<td>34 + 17</td>
<td>18 + 6</td>
<td>19 + 6</td>
</tr>
<tr>
<td><strong>R. eutropha - H16 Megaplasmid</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Contigs/scaffolds</td>
<td>127</td>
<td>54</td>
<td>53</td>
</tr>
<tr>
<td>NG50 (bp)</td>
<td>10,768</td>
<td>12,298</td>
<td>12,298</td>
</tr>
<tr>
<td>Genome coverage (%)</td>
<td>92.3</td>
<td>84.8</td>
<td>84.8</td>
</tr>
<tr>
<td>Misassemblies</td>
<td>0 + 1</td>
<td>1 + 0</td>
<td>1 + 1</td>
</tr>
</tbody>
</table>

#### 3.1.3 Aquarium water sample

Reads from the unknown sample were assembled into 1,347 and 79 contigs using **IDBA-UD** and **Athena-meta**, respectively. Summarizing the lengths of the assembled contigs revealed the barcoded assembly as approximately 500 kb smaller than that of the conventional one. The 4.5 Mb generated from the linked-read assembly held a N50 value of 170,377 bp. The same statistic for the conventional assembly was roughly three times smaller, 56,834 bp\(^7\).

As contigs shorter than 50kb confer little long-range information in an average sized bacterial genome, they were before further analysis filtered out from both assemblies. From the initial 1,347 contigs of the barcode-free assembly, only 36 contigs covering a total of 3,024,541 bases remained after this operation. That is, less than 3 % of the **IDBA-UD** assembled contigs were longer than 50kb. As a consequence, approximately 2Mb sequence data was removed from the assembly. Filtering of the linked-read assembly reduced the number of contigs to 23, giving that approximately 30 % of the original assembled contigs were longer than 50kb. The lengths of the new set were summarized to 3,863,428

\(^7\)Note: For this assembly, the N50 value is used instead of the NG50 since a reference genome cannot be used.
3 RESULTS

bases. All sequencing and assembly statistics are summarized in Table 5.

BLAST was used to investigate the species to which the filtered barcode-assembled contigs belonged. All contigs in the assembly were found aligning to the genus Aeromonas. Specifically, 91% of the contigs aligned unambiguously with the highest scores to A. veronii, while the remaining also mapped to A. hydrophila and A. salmonicida. The ambiguously aligned contigs aggravate the process of approximating the size of the metagenome as it may be comprised of one, two and three species. That is, the best estimation is 4,551,783 - 14,664,952 bp, where the lower value corresponds to the genome size of A. veronii and the higher to the sum of all three. Given the number of paired reads, the average sequencing depth was approximated to lie between 69X and 221X.

Table 5: Overview of sequencing and assembly statistics.

<table>
<thead>
<tr>
<th>Sequencing coverage and genome information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paired reads</td>
</tr>
<tr>
<td>Read duplicates</td>
</tr>
<tr>
<td>Metagenome size (bp)</td>
</tr>
<tr>
<td>Sequencing depth</td>
</tr>
</tbody>
</table>

Assembly and Scaffolding - Unfiltered

<table>
<thead>
<tr>
<th>Contigs/scaffolds</th>
<th>IDBA-UD</th>
<th>Athena-meta</th>
<th>Arcs/LINKS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assembly size (bp)</td>
<td>5,082,190</td>
<td>4,517,660</td>
<td>4,517,660</td>
</tr>
<tr>
<td>N50 (bp)</td>
<td>56,834</td>
<td>170,377</td>
<td>170,377</td>
</tr>
</tbody>
</table>

Assembly and Scaffolding - 50kb filter

<table>
<thead>
<tr>
<th>Contigs/scaffolds</th>
<th>IDBA-UD</th>
<th>Athena-meta</th>
<th>Arcs/LINKS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assembly size (bp)</td>
<td>3,024,541</td>
<td>3,863,428</td>
<td>3,863,428</td>
</tr>
<tr>
<td>N50 (bp)</td>
<td>86,033</td>
<td>191,640</td>
<td>191,640</td>
</tr>
</tbody>
</table>

3.2 Sequencing depth

As the R. eutropha genome displayed significantly lower sequencing depth than the genome of E. coli (despite equal input amounts), the data was further studied to uncover the reason behind this (Section 3.2.1). Furthermore, the difference in assembly contiguity between sequencing depths of 38X (E. coli) and 10X (R. eutropha) indicated a minimum sequencing depth threshold for contiguous assembly existed. The contiguity of genome assemblies at varying sequencing depths was therefore investigated (Section 3.2.2).
3 RESULTS

3.2.1 Uneven sequencing depth

The *R. eutropha* genome was sequenced to depth of less than 10X. Further studies revealed the depth to be uneven within the genomic molecules, with parts of the genome lacking coverage completely. These zero-coverage regions were dispersed evenly throughout the genome and were all shorter than 1,500 bases in length. A calculation of the base composition within these regions revealed the G/C percentage as 10 % higher than the genomic average. This indicates that regions with high G/C-content affect the analysis negatively. The results from the investigation are summarized in Table 6.

Table 6: G/C-content statistics for the *R. eutropha* whole genome assembly.

<table>
<thead>
<tr>
<th></th>
<th><em>R. eutropha</em> Chromosome 1</th>
<th><em>R. eutropha</em> Chromosome 2</th>
<th><em>R. eutropha</em> Megaplasmid</th>
</tr>
</thead>
<tbody>
<tr>
<td>Genomic G/C-content</td>
<td>66.5 %</td>
<td>66.8 %</td>
<td>62.3 %</td>
</tr>
<tr>
<td>Bases with zero coverage</td>
<td>309,491</td>
<td>225,517</td>
<td>11,418</td>
</tr>
<tr>
<td>G/C-content in zero-coverage regions</td>
<td>78.4 %</td>
<td>78.3 %</td>
<td>77.1 %</td>
</tr>
</tbody>
</table>

3.2.2 Minimum sequencing depth

To investigate the impact of shallow sequencing depth on whole genome assembly, the sequencing data from the analysis of a single species (*E. coli* BL21) was used to generate random subsets of data corresponding to 10X, 25X, 50X and 100X. All subsets were assembled and scaffolded using the pipeline outlined in Section 2.2.2.

For all data subsets, the whole genome assemblies covered > 98 % of the reference genome, regardless of implementation of barcode information. However, the assembly performed at 10X was significantly less contiguous than the remaining three, indicating that the minimum depth threshold for contiguous assembly lies between 10X and 25X sequencing depth. Furthermore, the increase in NG50 values between the conventional and barcoded assemblies was significantly smaller at 10X than what was observed for the remaining subsets.

Overall, *IDBA-UD* performed consistently with regards to all reported statistics given data of at least 25X depth, with no major improvements or declines when increasing the amount of input data. *Athena-meta* displayed an increase in contiguity with increasing sequencing depth, but at the cost of accuracy as the number of misassemblies grew when supplying more input data.

At 10X depth, scaffolding the contigs gave no effect and generated an output identical to the input. For all other subsets, the scaffolding improved the contiguity by merging assembled contigs and thereby increasing the NG50 values. The increase in NG50 was most significant at 50X and 100X where the value more than tripled for both. However, it should be noted that when mapped to the reference genome, contigs as far as 20 kb apart have been merged. These regions can essentially be observed as large assembly deletion events.

All results from the subset assemblies are summarized in Table 7.
### Table 7: Whole genome assembly statistics for subsets of data corresponding to 10X, 25X, 50X and 100X sequencing depth.

<table>
<thead>
<tr>
<th>Assembly and Scaffolding</th>
<th>IDBA-UD</th>
<th>Athena-meta</th>
<th>Arcs/LINKS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>10X depth</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Contigs/scaffolds</td>
<td>217</td>
<td>133</td>
<td>133</td>
</tr>
<tr>
<td>NG50 (bp)</td>
<td>52,180</td>
<td>66,119</td>
<td>66,119</td>
</tr>
<tr>
<td>Genome coverage (%)</td>
<td>99.2</td>
<td>98.6</td>
<td>98.6</td>
</tr>
<tr>
<td>Misassemblies + Local misassemblies</td>
<td>0 + 9</td>
<td>0 + 13</td>
<td>0 + 13</td>
</tr>
<tr>
<td><strong>25X depth</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Contigs/scaffolds</td>
<td>147</td>
<td>22</td>
<td>18</td>
</tr>
<tr>
<td>NG50 (bp)</td>
<td>89,664</td>
<td>691,273</td>
<td>810,133</td>
</tr>
<tr>
<td>Genome coverage (%)</td>
<td>99.3</td>
<td>99.7</td>
<td>99.7</td>
</tr>
<tr>
<td>Misassemblies + Local misassemblies</td>
<td>0 + 4</td>
<td>0 + 11</td>
<td>0 + 13</td>
</tr>
<tr>
<td><strong>50X depth</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Contigs/scaffolds</td>
<td>154</td>
<td>17</td>
<td>8</td>
</tr>
<tr>
<td>NG50 (bp)</td>
<td>89,669</td>
<td>626,882</td>
<td>4,150,954</td>
</tr>
<tr>
<td>Genome coverage (%)</td>
<td>99.4</td>
<td>99.8</td>
<td>99.8</td>
</tr>
<tr>
<td>Misassemblies + Local misassemblies</td>
<td>0 + 3</td>
<td>1 + 10</td>
<td>1 + 17</td>
</tr>
<tr>
<td><strong>100X depth</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Contigs/scaffolds</td>
<td>164</td>
<td>17</td>
<td>11</td>
</tr>
<tr>
<td>NG50 (bp)</td>
<td>81,584</td>
<td>1,162,758</td>
<td>3,747,374</td>
</tr>
<tr>
<td>Genome coverage (%)</td>
<td>99.4</td>
<td>99.8</td>
<td>99.8</td>
</tr>
<tr>
<td>Misassemblies + Local misassemblies</td>
<td>0 + 2</td>
<td>3 + 12</td>
<td>4 + 15</td>
</tr>
</tbody>
</table>

In Figure 5, scaffolds generated from linked-reads at different sequencing depth are illustrated. Scaffolds appear with increasing depth starting from the center. The outermost circle represents the reference genome with the genomic positions marked. The red marking at ∼ 2 Mb corresponds to a large deletion whose sequence can be found in a separate contig.\(^8\)

---

\(^8\)Figure created with Circos [40].
Figure 5: Scaffolds generated with linked-reads at varying sequencing depth. Starting from the center: 10X, 25X, 50X and 100X depth. The red marking at ~ 2 Mb corresponds to a large deletion, the sequence missing from the scaffold is found in a separate contig.
4 Discussion

In this master’s thesis project, DB-Seq was applied to bacterial samples in order to investigate its potential in metagenomics. Specifically, its ability in retaining information of sequencing read origins was investigated by performing de novo whole genome assemblies. Several software were utilized, two of which were developed to process linked-read sequencing data from 10x Genomics. In addition, customized scripts specifically created for data generated by DB-Seq were applied in the data analysis pipeline.

The whole genome assembly of \textit{E. coli}, performed with data of 190X sequencing depth, displayed great ability of DB-Seq to retain long-range information. The assembly and scaffolding performed using barcodes were significantly more contiguous than those performed in a conventional manner. When using linked-reads, an equally large part of the genome was contained in less than one tenth as many contigs and scaffolds as it was in the barcode-free assembly. The increase in NG50 when using long-range information further demonstrate the overall higher contiguity. It should be noted that the implementation of barcodes did not affect the overall quality of this assembly. The small parts not aligning perfectly to the reference genome were mainly in proximity to repeat regions. That is, the long-range information from DB-Seq reads was not sufficient for resolving these. However, this inability is common for most short-read sequencing methods since the read lengths are too short to span over such regions.

When sequencing \textit{E. coli} with \textit{R. eutropha}, the whole genome assembly was less contiguous and held more misassembly events than when sequenced independently. However, the genome coverage of the assembly remained high with over 99 % of the reference covered. The genome coverage for \textit{R. eutropha} was significantly lower, with the linked-read assembly covering as little as 61 % of the reference (chromosome 2). It is worth noting that the coverage was higher prior to applying the long-range information, indicating sequencing data being discarded in \textit{Athena-meta}. Studying the source code verifies this hypothesis as it specifies all contigs shorter than 500 bases should be discarded. The overall large differences between the assemblies of \textit{E. coli} and \textit{R. eutropha} were traced back to uneven sequencing depths. Two factors likely contributing to the unevenness are (1) the \textit{R. eutropha} genome consisting of three different molecules. Although the input amount was the same for both species, the majority ought to belong to \textit{E. coli}, (2) the \textit{R. eutropha} genome having an overall higher G/C-content. When mapping all sequencing reads to the reference genome of \textit{R. eutropha}, several regions were found lacking coverage completely. Further investigation of these zero-coverage regions revealed an average G/C-content of approximately 78 %, over 10 % higher than the genomic content. These results point to the assay having a G/C bias. For comparison, \textit{E. coli} has a G/C-content of 50.8 %. Although the scenario of uneven sequencing depth observed in this sample is highly probable in a real metagenomic study, explanations were sought in order to find requirements necessary for successful analysis.

The unsatisfactory assemblies at shallow sequencing depth (\textit{R. eutropha} chromosome 1 and 2) suggested that at 10X, a contiguous assembly was difficult to generate. Therefore, the minimum required sequencing depth was examined by extracting subsets from the 190X \textit{E. coli} data. When comparing the results, it was evident that a sequencing depth threshold existed between 10X and 25X. Overall, the 10X assembly held more contigs and scaffolds, covered less of the reference genome and had lower NG50 values. Furthermore, it had the smallest percental increase in NG50 when the long-range information was implemented.
DISCUSSION

This indicates that the long-range information was not conferred successfully when the sequencing depth was too shallow. This hypothesis is in concordance with the results from the *R. eutropha* assembly, where the NG50 values were little to nothing improved when using barcodes.

Furthermore, comparison of the 25X subset and the 38X *E. coli* (sequenced with *R. eutropha*) assemblies indicates that foreign DNA affects the results negatively. The 25X assembly displayed both higher NG50 and fewer scaffolds than what was observed for 38X, despite the depth being larger for the latter. A probable cause of this is that more than one DNA molecule was emulsified in the droplets when sequencing *E. coli* with *R. eutropha*, i.e., fragments were not uniquely barcoded. In the assembly software, these events can introduce ambiguities, and further result in branches in scaffolding graphs remaining unresolved. This would in turn give shorter contigs and thereby lower NG50.

The subset assembly at 50X displayed an equal number of scaffolds but a higher NG50 than the assembly at 190X, although using less than a third of the data. Since the data was selected randomly, ambiguous reads hindering contig expansion may have been removed in the smaller data set. This would further result in a higher NG50. The same explanation may be applied to why more misassemblies were observed for the subset data overall. That is, contigs and scaffolds have been expanded inappropriately due to the lack of complete data.

The assembly of the unknown aquarium sample showed large differences in contiguity between the conventional and linked-read approach. Counting contigs of all sizes, the barcode-free assembly consisted of 1,347 contigs covering a total of 5 Mb and holding a N50 of 56,834. Although covering less bases, the linked-read assembly displayed overall longer contigs with a N50 of 170,377. It should be noted that there is no reference genome available for this data set. Therefore, there is no way of knowing whether the smaller assembly size corresponds to the loss of information or simply to a more efficient way of representing the data.

Due to the scarce information conferred by small contigs, all shorter than 50kb were filtered from both assemblies before further analysis. The remaining contigs from the conventional assembly, 36 psc, covered 3 Mb and held a N50 of 86,033 while the linked-read approach covered 3.9 Mb and had 23 contigs with N50 of 191,640. Although the linked-read scaffolding did not improve the assemblies, the difference in contiguity between barcoded and conventional methods is still large enough to confirm superiority of DB-Seq.

In the cultivated aquarium sample, only bacteria from a single genus were identified. However, some contigs aligned to more than one species within this genus in BLAST. Considering the high input amount, its is likely homologous regions from different species of *Aeromonas* have been emulsified in the same droplet and thereby given the same barcode sequence. This would further lead to mixed contigs where reads originating from different species have been assembled. This would in turn explain the ambiguous alignment. It is difficult to say what the results would have been if the complexity was higher or if the input amount was lower.

It is safe to say that the application of DB-Seq in *de novo* whole genome assembly shows great potential. Compared to conventional sequencing and assembly, it has displayed abilities of producing more contiguous and equally complete genomes. Thereby, the conclusion that DB-Seq successfully confers information of read origins can be drawn. Experiments using multiple species revealed requirements contributing to successful analysis to be (1) minimum threshold of
sequencing depth between 10X and 25X and (2) low G/C-content due to G/C-bias in the assay. Although there is little to do about this in a metagenomic sample, DB-Seq can in the future be improved to reduce the impact.

5 Future Perspectives

Although having displayed great potential in its first application in metagenomics, future developments to improve the performance of DB-Seq in the field are many, and exiting. In the laboratory procedure, developments to reduce the read duplicates would not only generate more informative data, but also reduce the information bias of the assay. It should be noted that the process of solving this issue is already underway and improvements have been observed in the duration of this project. Furthermore, in order to fully resolve complex metagenomes, more reads must be generated for each sample. To achieve this, the HiSeq platform from Illumina ought to be used instead of Miseq.

The whole genome assembly pipeline developed in this project utilized programs specifically written for 10x Genomics data. This suggests that software developed for other linked-read platforms successfully can be applied to DB-Seq data. However, the strengths of DB-Seq are not fully exploited when using existing programs. Therefore, the assembly using linked-reads from DB-Seq may be improved if Athena-meta was modified or an assembly software was specifically developed for the method.

Finally, as the cost of sequencing DNA continues to decrease, the demands on cheap library preparation protocols will increase. In this aspect, DB-Seq shows great promise in the future.

6 Acknowledgements

First and foremost, I would like to thank professor Afshin Ahmadian for granting me the opportunity to work in his laboratory, and for the supervision he has provided me with. Furthermore, I would like to recognize the members of my research group for their assistance in this project. A special thank you to the group of assistant professor Paul Hudson for providing me with an isolated sample of \textit{R. eutropha}, and to Gunaratna Kuttuva Rajarao via Caroline Oskarsson for preparing cultivated bacteria from aquarium water.
References


REFERENCES


REFERENCES


7 Appendix

7.1 Genome references

<table>
<thead>
<tr>
<th>Species</th>
<th>Reference ID</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>E. coli</em> - BL21</td>
<td>CP010816.1 [41]</td>
</tr>
<tr>
<td><em>R. eutropha</em> - H16 Chromosome 1</td>
<td>AM260479.1 [42]</td>
</tr>
<tr>
<td><em>R. eutropha</em> - H16 Chromosome 2</td>
<td>AM260480.1 [42]</td>
</tr>
<tr>
<td><em>R. eutropha</em> - H16 Megaplasmid</td>
<td>AY305378.1 [43]</td>
</tr>
<tr>
<td><em>A. veronii</em> - B565</td>
<td>CP002607.1 [44]</td>
</tr>
<tr>
<td><em>A. hydrophila</em> - D4</td>
<td>CP013965.1 [45]</td>
</tr>
<tr>
<td><em>A. salmonicida</em> - pectinolytica</td>
<td>CP022426.1 [46]</td>
</tr>
</tbody>
</table>

7.2 Open-source programs and customized scripts

<table>
<thead>
<tr>
<th>Software</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>WGH_automation.sh</code></td>
<td>In-house automation script specifying commands for <em>Cutadapt</em>, <em>UMI-tools</em>, <em>cdhit_prep.py</em> and <em>cd-hit-454</em>. That is, the removal of adaptor/barcode sequences and the clustering of barcode sequences.</td>
</tr>
<tr>
<td><em>Cutadapt</em></td>
<td>Removes adaptors from sequencing reads.</td>
</tr>
<tr>
<td><em>UMI-tools</em></td>
<td>Extracts barcodes from sequencing reads.</td>
</tr>
<tr>
<td><em>cd-hit_prep.py</em></td>
<td>Splits file holding all barcode sequences into subfiles according to the first three bases in the sequence.</td>
</tr>
<tr>
<td><em>cd-hit-454</em></td>
<td>Clusters barcode sequences with a similarity of &gt; 90 %.</td>
</tr>
<tr>
<td><em>tag_fastq.py</em></td>
<td>Tags reads with the consensus barcode sequences generated by <em>cd-hit-454</em>. The format of tagging is the same as used by 10x Genomics.</td>
</tr>
<tr>
<td><em>sort_file.sh</em></td>
<td>Sorts read files according to barcode sequence by collapsing the .fastq read records into a single line and sorting on the barcode column.</td>
</tr>
<tr>
<td><em>sort_tagged_file.py</em></td>
<td>Reformats the sorted file to .fastq format.</td>
</tr>
<tr>
<td><em>IDBA-UD</em></td>
<td>Assembles short reads without barcode information, i.e. conventional short-read assembly. <em>IDBA-UD</em> is used in the first step of the linked-read assembly to generate a genome draft.</td>
</tr>
<tr>
<td><em>BWA-MEM</em></td>
<td>Aligns sequencing reads onto reference sequences.</td>
</tr>
<tr>
<td><em>Athena-meta</em></td>
<td>Performs de novo assembly of linked-reads.</td>
</tr>
<tr>
<td><em>Arcs/LINKS</em></td>
<td>Performs scaffolding of contigs using linked-reads.</td>
</tr>
<tr>
<td><em>SSpace</em></td>
<td>Performs scaffolding of contigs without barcode information, i.e. conventional scaffolding.</td>
</tr>
</tbody>
</table>
7.3 Customized scripts

tag_fastq.py

```python
#!/usr/bin/env python

def main():
    """ Takes a .fastq file with trimmed reads and a .fasta file with clustered barcodes
    Output is a tagged .fastq file. """

global args, summaryObject
from Bio import SeqIO
import logging

configureLogging('info')
argParser() # Argument parsing
logging.info('Arguments read successfully')
summaryObject = Sum()
readClusters(args.input_clstr)
logging.info('Cluster file processed successfully')

with open(args.input_trimmed_fastq, 'r') as openin, 
     open(args.output_tagged_fastq, 'w') as openout:
    for record in SeqIO.parse(openin,'fastq):
        name,bc_seq = record.id.split('_')
        try:
            consensus_seq = summaryObject.master_barcode_dict[bc_seq]
        except KeyError:
            continue
        record.id = name + '_' + consensus_seq
        record.description = record.id + ' BC:Z:' +consensus_seq + '-1'
        SeqIO.write(record,openout,'fastq')
logging.info('Tagging completed')

class configureLogging(object):
    global logging
    def __init__(self,logType):
        log_format = ' %(asctime)s %(levelname)s: < %(message)s >'
        if logType == 'info':
            configureLogging.INFO(self,log_format)
        INFO(self,log_format):
            logging.basicConfig(level=logging.INFO, format=log_format)

##### Code from WGH_automation.sh scripts, i.e. same way of processing clusters ######

class argParser(object):
    """Reads input arguments""

def __init__(self):
    argParser.parse(self)
```

def parse(self):
    import argparse
    global args
    parser = argparse.ArgumentParser(description=__doc__)
    parser.add_argument("input_trimmed_fastaq",
                        help=".fq file with trimmed reads to be tagged with bc id:s.")
    parser.add_argument("input_clstr",
                        help=".clstr file from cdhit clustering.")
    parser.add_argument("output_tagged_fastaq",
                        help=".fq file tagged with barcode consensus sequence")
    args = parser.parse_args()

def readClusters(clstrfile):
    
    """Reads clstr file and builds consensus_bc:bc dict in Sum ""
    new_cluster = False
    with open(clstrfile, 'r') as f:
        cluster = ClusterObject(clusterId=f.readline())
        for line in f:
            if line.startswith('>'):
                new_cluster = True
            elif new_cluster == True:
                new_cluster = False
                cluster = ClusterObject(clusterId=line)
                cluster.addBarcodeToDict(line)
                summaryObject.updateDict(cluster)
            else:
                cluster.addBarcodeToDict(line)
        summaryObject.updateDict(cluster)

class ClusterObject(object):
    """ Cluster object, one for each cluster"
    def __init__(self, clusterId):
        self.consensus_to_bc_dict = dict()
        self.consensus = clusterId.split()[-2].split(':')[1].rstrip('.

        def addBarcodeToDict(self, line):
            accession = line.split()[2].rstrip('.
            barcode = accession.split(':')[1][-1]
            self.consensus_to_bc_dict[barcode] = self.consensus

class Sum(object):
    """ Merges cluster-specific dictionaries to a master dictionary.""
    def __init__(self):
        self.master_barcode_dict = dict()
    def updateDict(self, input_object):
        input_dict = input_object.consensus_to_bc_dict
        for barcode in input_dict.keys():
            self.master_barcode_dict[barcode] = input_object.consensus

###########################################################################
if __name__=='__main__': main()

sort_file.sh

#!/bin/bash

############ Argument and error handling-code same as in WGH_automation.sh ############

while getopts "hp:" OPTION do
  case ${OPTION} in
    h)
      echo ''
      echo 'This script sorts a tagged .fastq file based on barcode sequence.'
      echo ''
      echo 'Usage: bash sort_file.sh <tagged.fastq> <tagged_sorted.fastq>'
      echo ''
      echo "Positional arguments (required)"
      echo "<tagged.fastq> Read file in .fastq format"
      echo "<tagged_sorted.fastq> Sorted .fastq file, according to barcode sequence."
      echo ""
      echo "Optional arguments"
      echo "-h help (this output)"
      echo ""
      exit 0
  esac
done

ARG1=${@:$OPTIND:1}
ARG2=${@:$OPTIND+1:1}

wgh_path=$(dirname "$0")

if [ -z "$ARG1" ] || [ -z "$ARG2" ] then
  echo ""
  echo "ARGUMENT ERROR"
  echo "Did not find all positional arguments, see -h for more information."
  echo "(got input:"$ARG1" and output:"$ARG2" instead)"
  echo ""
  exit 0
fi

#cat $ARG1 |
```python
while read first_line; read second_line; read third_line; read fourth_line
do
echo "$first_line" "$second_line" "$third_line" "$fourth_line" >> temp.fastq;
done
echo "Read records collapsed into one line - DONE"
cat temp.fastq | sort -d -k2,2 -k1,1 > temp2.fastq
echo "Reads sorted according to barcode sequence - DONE"
rm temp.fastq
echo "Initiating reformatting to .fastq format."
cat temp2.fastq |
python $wgh_path/python_scripts/sort_tagged_file.py temp2.fastq $ARG2
echo "Sorting - DONE"
rm temp2.fastq

sort_tagged_file.py

#!/usr/bin/env python
def main():
    """ Takes a fastq file and sorts it according to barcode sequence """
    global args
    readArgs()
    with open(args.one_line_file_records, 'r') as openin, 
        open(args.reformatted_fastq_sorted, 'w') as openout:
        for line in openin:
            arguments = line.split()
            openout.write(arguments[0] + ' ' + arguments[1] + '
')
')

class readArgs(object):
    def __init__(self):
        readArgs.parse(self)
    def parse(self):
        import argparse
```
global args
parser = argparse.ArgumentParser(description=__doc__)

# Arguments
parser.add_argument("one_line_file_records",
    help="temp file where a .fastq record is in a single line")
parser.add_argument("reformatted_fastq_sorted",
    help="outfile, sorted .fastq based on barcode sequence")

args = parser.parse_args()

if __name__ == '__main__':
    main()

athena_assembly_automation.sh

#!/usr/bin/env bash

############ Argument and error handling-code same as in WGH_automation.sh ############

while getopts "hp:" OPTION
do
    case ${OPTION} in
        h)
            echo ''
            echo 'This script processes paired read metagenomics data and performs
deo novo whole genome assembly'
            echo ''
            echo 'Usage: bash athena_assembly.sh \
                 <r1_trimmed.fq> <r2_trimmed.fq> <NNN.clstr> <output_dir>'
            echo ''
            echo "Positional arguments (required)"
            echo " <r1_trimmed.fq> R1 in .fq format. Trimmed with the bc in the header"
            echo " <r2_trimmed.fq> R2 in .fq format. Trimmed with the bc in the header"
            echo " <NNN.clstr> Concatenated .clstr file with all barcode clusters."
            echo " <output_dir> Output directory holding all assembly files."
            echo ''
            echo "Optional arguments"
            echo " -h help (this output)"
            echo ''
            exit 0
        ;;
    esac
done

ARG1=${@:$OPTIND:1} # r1 file
ARG2=${@:$OPTIND+1:1} # r2 file
ARG3=${@:$OPTIND+2:1} # .clstr file
ARG4=${@:$OPTIND+3:1} # output dir

file1=$ARG1
name_ext1=$(basename "$file1")
name1="${name_ext1%.*}" # name without extension

file2=$ARG2
name_ext2=$(basename "$file2")
name2="${name_ext2%.*}" # name without extension

if [ -z "$ARG1" ] || [ -z "$ARG2" ] || [ -z "$ARG3" ] || [ -z "$ARG4" ]
then
  echo "" # ARGUMENT ERROR
  echo "Did not find all four positional arguments, see -h for more information."
  echo "" # name without extension
  exit 0
fi

dir_of_scripts=$(dirname "$0")
dir_of_files=$PWD

grep -r ".tagged.fastq" $dir_of_files

source activate athena_assembly

#### STEP 1 ####
# TAG THE .FASTQ FILES WITH "BC:Z:BC_SEQ" USING tag_fastq.py
printf "### STEP 1 - Tagging initiated \n"
for file in $ARG1 $ARG2;
do
  name_ext=$(basename "$file")
  printf 'Tagging %s \n' "$name_ext"
  name="${name_ext%.*}"
  python3 $dir_of_scripts/python_scripts/tag_fastq.py 
  $file $ARG3 $name".tagged.fastq";
done;

printf "### STEP 1 complete. Read files tagged with BC:Z:BC_SEQ \n"

#### STEP 2 ####
# SORT THE FILE ACCORDING TO BC_SEQ
printf "### STEP2 - Sorting initiated \n"
for file in $dir_of_files/*.tagged.fastq
  do

35
name_ext=$(basename "$file")
printf "Sorting %s \n" "$name_ext"
name="${name_ext%.*}"
bash $dir_of_scripts/sort_file.sh $file $name".sorted.fastq"
done;

printf "### STEP 2 complete. Tagged read files sorted according to barcode sequence \n"

#### STEP 3 ####
# MERGE THE TAGGED AND SORTED .FASTQ FILES (CREATING AN INTERLEAVED FILE)
# AND CONVERT IT TO .FASTA FORMAT (bbmap reformat)
reformat.sh in1=$dir_of_files/$name1.tagged.sorted.fastq 
in2=$dir_of_files/$name2.tagged.sorted.fastq 
out=$dir_of_files/interleaved_R1_R2.fastq
reformat.sh in=$dir_of_files/interleaved_R1_R2.fastq 
oun=$dir_of_files/interleaved_R1_R2.fasta
rm $dir_of_files/*tagged*
printf "### STEP 3 complete. read1.fastq and read2.fastq merged to interleaved file 
and converted to .fasta format. \n"

#### STEP 4 ####
# RUN IDBA TO ASSEMBLE SEED CONTIGS
mkdir idba_seed_contigs
idba_ud -r $dir_of_files/interleaved_R1_R2.fasta -o idba_seed_contigs
cp $dir_of_files/idba_seed_contigs/contig.fa $dir_of_files
rm -rf $dir_of_files/idba_seed_contigs
printf "### STEP 4 complete. Seed contigs generated. \n"

#### STEP 5 ####
# RUN BWA MEM TO MAP THE READ CLOUDS TO THE ASSEMBLED CONTIGS
printf "### STEP 5 - Initiating BWA alignment\n" bwa index $dir_of_files/contig.fa samtools faidx $dir_of_files/contig.fa
printf "STEP 5.1 complete. BWA indexes generated.\n"
bwa mem -C -p $dir_of_files/contig.fa $dir_of_files/interleaved_R1_R2.fastq | 
samtools sort -o mapped_reads.idba_contigs.bam -
printf "STEP 5.2 complete. Reads mapped to seed contigs.\n" printf "### STEP 5 complete.\n"
#### STEP 6 ####

`must make index of the .bam file in order for it to work.`

```
samtools index mapped_reads.idba_contigs.bam mapped_reads.idba_contigs.bam.bai
```

printf "STEP 6 complete. .bai file generated \n"

#### STEP 7 ####

`GENERATE CONFIG.JSON FILE`

```
echo -e "{" >> config.json
echo -e "	 "ctgfasta_path": "$dir_of_files/contig.fa",
" >> config.json
echo -e "	 "reads_ctg_bam_path": "$dir_of_files/mapped_reads.idba_contigs.bam",
" >> config.json
echo -e "	 "input_fqs": "$dir_of_files/interleaved_R1_R2.fastq",
" >> config.json
echo -e "	 "cluster_settings": {" >> config.json
echo -e "		 "cluster_type": "multiprocessing",
" >> config.json
echo -e "		 "processes": 4 >> config.json
echo -e "}" >> config.json
```

printf "STEP 7 complete. \n"

#### STEP 8 ####

`RUN ATHENA`

```
athena-meta $dir_of_files/config.json
```

printf "STEP 8 complete. Genome assembled."