Comparison of SPAdes and MEGAHIT for Assembly of Honeybee and Carpenter Bee Genomes

Will Mayer, Alan Kusparmakov

Background

About the Datasets

The initial phase of this project was to find a whole genome sequencing (WGS) read for two similar species. These reads typically come in the form of a FASTQ file. For this project we have used two species' datasets - *Apis mellifera ligustica* (honeybee), and *Xylocopa dejeanii* (carpenter bee). The honeybee data was originally created for an experiment where the grooming behavior of a 12 day worker bee was investigated for relevant single nucleotide polymorphisms (SNPs). The carpenter bee data was gathered in a study that aimed to create the first chromosome level genome for its species. All specimens were gathered in Motuo County, Linzhi, Xizang, China. It's worth noting that if one were to attempt complete de novo genome assembly, the carpenter bee dataset is more suited for such purposes. It has much higher coverage than the honeybee dataset, which was intended for studying specific areas of the genome.

Whole Genome Sequencing

The datasets for this project used Illumina technologies for whole genome sequencing. In whole genome sequencing, DNA is extracted and isolated from the sample, and then fragmented into smaller pieces. Adapters are attached to the ends of the fragments in order for them to be amplified and read by the sequencer. The presence of adapters affects the results of the assembly step, which is why we trim the genome before processing it. Illumina is a type of short-read sequencing which uses fluorescence to identify bases, and produces fragments of 100-300 bp.

Trimming

We used Trimmomatic as our preprocessing software to trim the FASTQ files. Besides trimming the adapters and other Illumina-specific helper sequences, Trimmomatic runs a sliding window algorithm to trim out sections of the datasets below a threshold of quality.

Subsampling

The datasets used in this project were both substantially sized. In addition, the assembly algorithms used take up a lot of processing power, and Google Collab is limited with the amount of computing they allow for each account. Collab was by far the biggest bottleneck, even with Pro accounts we were unable to run the software on either full genome for a reasonable amount of time. The result of this meant we were forced to use a subsample of the input data for our assembly. Because of this we knew that our QUAST and BUSCO scores would be subpar when compared to that of actual full genome assembly, but by keeping the amount of reads we took consistent we are still able to draw conclusions between the differences in software and genomes. We used SeqTK to generate randomized subsamples of the FASTQ files. FASTQ files often come in related pairs, but if SeqTK is given the same randomization seed for both subsamples, the parity is preserved.

SPAdes Assembler

SPAdes stands for St. Petersburg genome assembler. It is primarily developed for Illumina sequencing data. SPAdes uses k-mers for building an initial de Brujin graph. It then uses various graph-theory operations graph topology analysis in order to identify incorrect reads and constructing contigs. For Illumina data, SPAdes uses an error correction tool called the BayesHammer, is a computationally complex statistical clustering algorithm and can take up to 10 GB of additional RAM. Due to the heavy requirements of the BayesHammer beyond Google Colab's resources, we had to disable it for our run.

Megahit Assembler

Unlike SPAdes, megahit aims to be ultra-fast and memory efficient, forgoing SPAdes' heavy data structures and error correction. Megahit is optimized for complex metagenomics data, but it can also be used for single genome assembly of small or mammalian size, as well as single-cell assembly. This assembler is designed for single-node operation, so that it can run efficiently on a single machine. It uses a modified version of the de Brujin graph called a Succinct De Brujin

graph (SdBG), which uses various compression techniques to condense the data structure to a more manageable size.

Quast

QUAST stands for QUality Assessment Tool for genome assemblers. It evaluates the structural quality of your genome. It can tell you how complete / contiguous your genome is, as well as where any errors might be. It does not actually measure any metrics of your gene content. Most information is gleaned from a few key metrics. N50 is the length at which 50% of your genome is contained in contigs of that size or larger. It also provides the total genome length, longest contig, as well as the number of contig files (sections of genome). It's also possible to provide a reference genome to QUAST to gain more data, such as missassemblies, genome fraction, and mismatches.

BUSCO

BUSCO stands for Benchmarking Universal Single-Copy Orthologs. Unlike QUAST it does check for the content of your genome. The idea is that there are certain genes that are present in every living organism on the planet, as well as genes that are precinct in certain broad taxon. By scanning our genome for these genes and finding them we can be confident that our genome was assembled correctly. If you find 90% of BUSCOs it gives you a rough estimate that your genome is 90% accurate. After reading the input genome BUSCOs fall into 3 categories: complete, fragmented, and missing.

Methods

Google Colab Pipeline

Colab was used to run and annotate code, store intermediate files, and connect to Google Drive for long-term storage. All operations were done within Colab’s virtual machine filesystem, and results were then sent to a Drive directory. Bash was the primary language for running our toolkits.

Fasterq-Dump

Datasets were pulled from the database by giving an accession number to Fasterq-Dump, which deposits the large FASTQ files in the given output directory. Fasterq-Dump generates a pair of files containing paired reads for each DNA fragment. Up to 10x of the final output's worth of memory should be available in order to house the temporary files.

Micromamba

All subsequent operations were called from a virtual environment which was set up using micromamba. This setup made it convenient to install packages and call commands across different Colab cells.

Pre-Processing

The paired FASTQ files were ran through Trimmomatic, which results in two paired files and two unpaired files. The unpaired reads were discarded as noise. To keep runtimes reasonable both genomes were reduced to 500,000 reads using SeqTK random subsampling. In both cases this shrunk the input to about 3.5% of its original size. In addition, runs were done directly in Colab and then moved to google drive after. Running an operation on files across Colab's virtual environment and Google Drive's filesystem is a simple mistake to make that reduces the operation speed significantly.

Assembly

The trimmed and subsampled FASTQ files were then fed into SPAdes and Megahit. The results were saved to Google Drive, and the contigs.fasta files were analyzed.

BUSCO

Both species were compared against the hymenoptera_odb10 dataset to determine what BUSCO's to look for. Hymenoptera contain all sawflies, wasps, bees, and ants. Contig files were given as input and a text log containing three classifications of BUSCO genes was returned.

QUAST

The resulting FASTA files were compared against standard reference genomes for honeybees and carpenter bees. QUAST then outputs the results in various formats.

Results

Final Files

Contig files for four partial genomes:

Honeybee (SPAdes): 24.5MB

Honeybee(MegaHit): 6.7MB

Carpenter Bee(SPAdes): 3.4MB

Carpenter Bee(MegaHit): 332KB

BUSCO

Honeybee (SPAdes):

```
  --------------------------------------------------
  |Results from dataset hymenoptera_odb10            |
  --------------------------------------------------
  |C:0.1%[S:0.1%,D:0.0%],F:1.9%,M:98.0%,n:5991       |
  |7    Complete BUSCOs (C)                          |
  |7    Complete and single-copy BUSCOs (S)          |
  |0    Complete and duplicated BUSCOs (D)           |
  |114    Fragmented BUSCOs (F)                      |
  |5870    Missing BUSCOs (M)                        |
  |5991    Total BUSCO groups searched               |
  --------------------------------------------------
```

Honeybee (MegaHit):

```
  --------------------------------------------------
  |Results from dataset hymenoptera_odb10            |
  --------------------------------------------------
  |C:0.0%[S:0.0%,D:0.0%],F:0.9%,M:99.1%,n:5991       |
  |0    Complete BUSCOs (C)                          |
  |0    Complete and single-copy BUSCOs (S)          |
  |0    Complete and duplicated BUSCOs (D)           |
  |52    Fragmented BUSCOs (F)                       |
  |5939    Missing BUSCOs (M)                        |
  |5991    Total BUSCO groups searched               |
  --------------------------------------------------
```

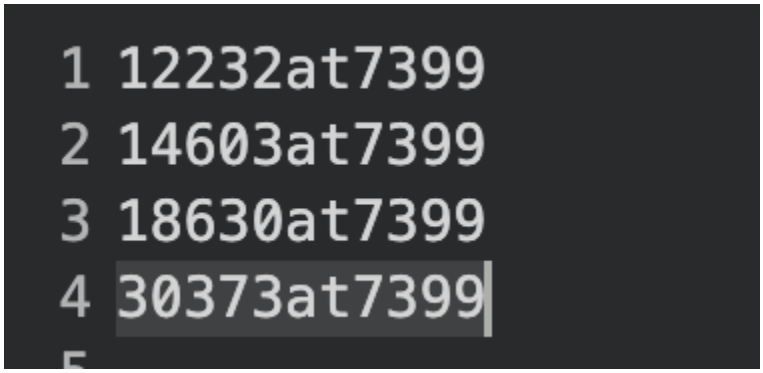
Carpenter Bee (SPAdes):

```
   --------------------------------------------------
   |Results from dataset hymenoptera_odb10            |
   --------------------------------------------------
   |C:0.0%[S:0.0%,D:0.0%],F:0.4%,M:99.6%,n:5991       |
   |0    Complete BUSCOs (C)                          |
   |0    Complete and single-copy BUSCOs (S)          |
   |0    Complete and duplicated BUSCOs (D)           |
   |23    Fragmented BUSCOs (F)                       |
   |5968    Missing BUSCOs (M)                        |
   |5991    Total BUSCO groups searched               |
   --------------------------------------------------
```

Carpenter Bee (MegaHit):

```
    ---------------------------------------------------
    |Results from dataset hymenoptera_odb10             |
    ---------------------------------------------------
    |C:0.0%[S:0.0%,D:0.0%],F:0.0%,M:100.0%,n:5991       |
    |0    Complete BUSCOs (C)                           |
    |0    Complete and single-copy BUSCOs (S)           |
    |0    Complete and duplicated BUSCOs (D)            |
    |1    Fragmented BUSCOs (F)                         |
    |5990    Missing BUSCOs (M)                         |
    |5991    Total BUSCO groups searched                |
    ---------------------------------------------------
```

Genome	Complete	Fragmented	Missing
Honeybee (SPAdes)	7	114	5870
Honeybee (MegaHit)	0	52	5939
Carpenter Bee (SPAdes)	0	23	5968
Carpenter Bee (Megahit)	0	1	5990

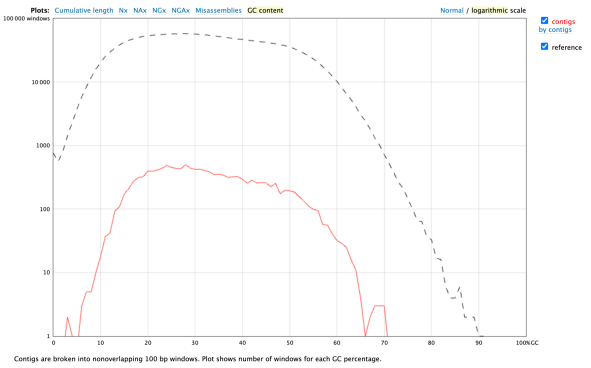


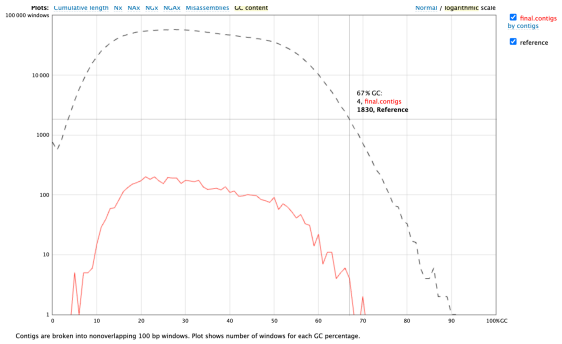
<u>QUAST</u>

GC content:

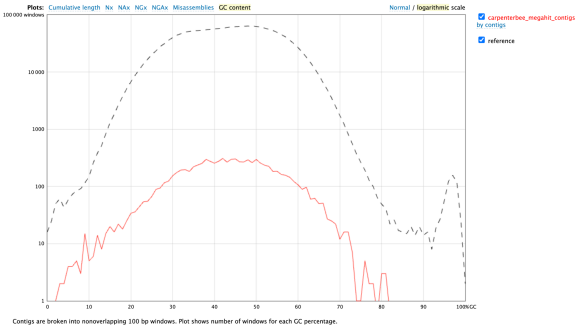
One graphic metric taken from QUAST is the GC content. GC content is a metric used to determine how accurate your data is based on the ratio of G's to C's found throughout the genome. You expect it to be a roughly normal distribution with a peak at 50%. All four of our genomes show this trend, meaning the assembly is likely of good quality. The graphs all fall far below the expected quantity number, but this is due to the small input data we provided.

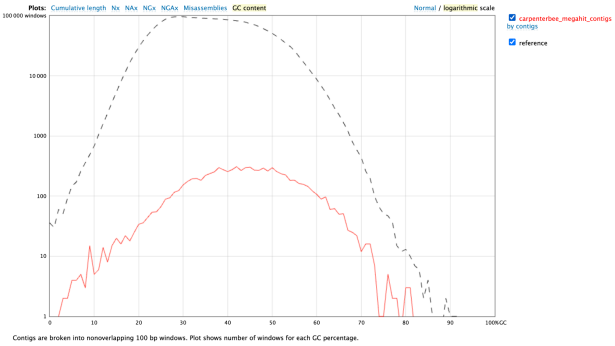
Honeybee





Carpenter Bee





Genome	Genome Fraction	No. Contigs	Duplication Ratio	No. Misassemblies	Largest allignment
Honeybee (SPAdes)	.53	2038	0.999	1127	10,836
Honeybee (MegaHit)	.218	817	1.005	47	9,718
Carpenter Bee (SPAdes)	.326	1025	1.008	101	11,604
Carpenter Bee (Megahit)	.002	1025	1.010	0	1,930

Discussion

Two different assembly softwares were used for the partial assembly of two bee genomes. SPAdes and Megahit were the two assemblers used. Carpenter Bees and Honeybees were our target species. It is crucial to note that of the two inputs, the size of the reads of the

carpenter bee was much larger than that of the honey bee. Since subsampling was used and we took the first 500,000 reads from both, the input provided to the assembler was much more complete for the honey bee. We believe this to be the cause for much of the results we see in the data. The honeybee genome assembled using SPAdes outperformed all other genomes in every testing metric we used.

First let's consider BUSCO scores. The major taxon hymenoptera was used to classify what genes would be searched for as BUSCOs. The previously mentioned honeybee spades genome was the only one to contain complete BUSCOS, finding 7 in total. It also had the highest number of fragmented genes with 114. The second best BUSCO was the honey bees MegaHit assembly. This makes sense when you consider how honey bee reads had a larger percentage of the total genome. When classifying by species however you'll notice that the SPAdes read outperforms the MegaHit read both times, by a very large degree. It seems BUSCOs are more dependent on the size of the input data, and not the assembly process.

The same however can not be said for QUAST data. Once again the Honey Bee SPAdes run shines above the rest with the highest genome fraction (.53) and number of contigs. It also has the best duplication ratio at .999 (DR is better the closer it is to 1). Interestingly the second best performer here is not the other honey bee but rather the SPAdes run for the carpenter bee. It had a genome percent of .326 and even edged out the honey bee SPAdes run for the longest single alignment. Despite being over half the size of the honey bee run it also managed to get 1/10th the number of disassemblies. Taking second with the disadvantage of less input data leads us to believe that QUAST data is much more dependent on the method of assembly used. This makes sense when you consider what's actually being measured - gaps in data and strength of assembly.

On the species side of things, although only one genome was able to produce any full BUSCOs, we were able to map some of the fragmented BUSCOS across carpenter and honey bees. In total there were 4 overlapping BUSCO's which is reasonable considering that the best carpenter bee assembly only had 23 fragmented BUSCOs, meaning around 18% of the BUSCOs in one appeared in the other. Carpenter Bees and Honey Bees are both apart of the Apidae family, but belong to different subfamilies - meaning this genetic similarity is reasonable especially considering the small size of our data. It does show a clear relationship between the two species.

In terms of choosing one assembler over the other in almost all cases it makes more sense to use SPAdes. This is especially true if you're trying to make any de novo gene assembly and want your work to be used in actual applications. This is because what SPAdes lacks in compute and time efficiency it well makes up for in the quality of genome assembled. In all metrics attempted it outperformed its megahit counterpart. That isn't to say that megahit has no use cases. There are times when time is low or there is minimal access to processing power, making megahit an amazing and still decently accurate alternative.