Lab 9: Sequence Assembly

Using SPAdes, you will assemble a bacterial genome de novousing a combination of long PacBioreads and short Illumina reads. Next week, you will analyze your genome to determine the species and obtain an overview of its metabolism.You are expected to keep a thorough record of everything you did in your notebook. Copy and paste any terminal commands you used into a Markdown section and explain what the input was, what the tool did, and what the output was. Plot any results in-line and explain them.

References

- Sequence assembly
- SPAdes
- PacBio SMRT Sequencing
- Illumina sequencing

Background

Genome sequencing and assembly arecommon techniquesin biology. To obtain the sequence of a long genome, DNA must be chopped into small pieces that can be read by a sequencer. These short reads must then be stitched back together to form acomplete genome.Often, the genome cannot be fully assembled because there are multiple equally plausible ways of stitching the reads together. Ideally, each chromosome is assembled into a single, long sequence. In practice, chromosomes are often assembled into multiple “contigs,” or contiguous sequences. A genome assembly is generally considered complete only when all (or nearly all) the sequences are accounted for. Otherwise, it is considered a draft genome.

In this lab, you will assemble and analysis a bacterial genome, using Illumina and PacBio reads. This week, you will take the reads and assemble them into a complete genome. Next week, you will analyze the contents of your genome.

Locating the data

DNA from an unknown bacterium was sequenced using PacBio and Illumina technologies. The resulting reads are uploaded onto bCourses. You will need to download these files. You will need to upload the files if you are using DataHub.

`illumina_reads_R1.fastq` – first paired-end read \ `illumina_reads_R2.fastq` – second paired-end read \ `pacbio_reads.fastq` – long PacBio reads

In [7]:
```
!bzip2 -dk reads/*.fastq.bz2
```

```
bzip2: Output file reads/illumina_reads_R1.fastq already exists.
bzip2: Output file reads/illumina_reads_R2.fastq already exists.
```

Running SPAdes

SPAdes is a hybrid genome assembler, meaning that it takes multiple sources of information as input and combines them to produce an optimal assembly. Assemblies using only short reads tend to be highly fragmented (i.e., many contigs). Assemblies using a high-quality short read set and a higher error rate long-read set (like PacBio) tend to be the best.

Why do we expect short reads to produce a more fragmented assembly than long reads?

> We would expect short reads to result in more fragmented assemblies because the shortness of the input leads to a higher amount of contigs versus long reads, which would increase the likelihood of there being gaps between reads, thus splitting up the assembly.

Why does a single-molecule sequencing like PacBio have a higher error rate than Illumina?

> Illuma uses many small reads versus the individual long strands used in SMRT. Thus, in the case of inevitable misreads, Illumina can use multiple overlaping contigs for the same reading frame thus having a greater coverage than the equivalent PacBio sequencing and reducing error.

We need to come up with a SPAdes command. At a minimum, you will need to specify the output directory with -o, the path to the first Illumina read with -1, the path to the second Illumina read with -2, and the path to your PacBio reads with --pacbio. Note: SPAdes must be run from the command line, and can take a while.

Genome assembly requires a relatively large amount of computer memory. Sometimes up to 1TB. Datahub instances have 64GB of memory. We have significantly subsampled the reads in order to run the analysis on DataHub.

SPAdes typically uses multi-threading to speed up assembly. Each thread requires memory. You may need to add -t 4 to your command so that it uses only 4 threads on the system rather than 16. If the program crashes, reduce the number of threads.

```
```

=== KDE started.

```
```

=== KDE finished.

```
```

=== KDE finished.

```
```