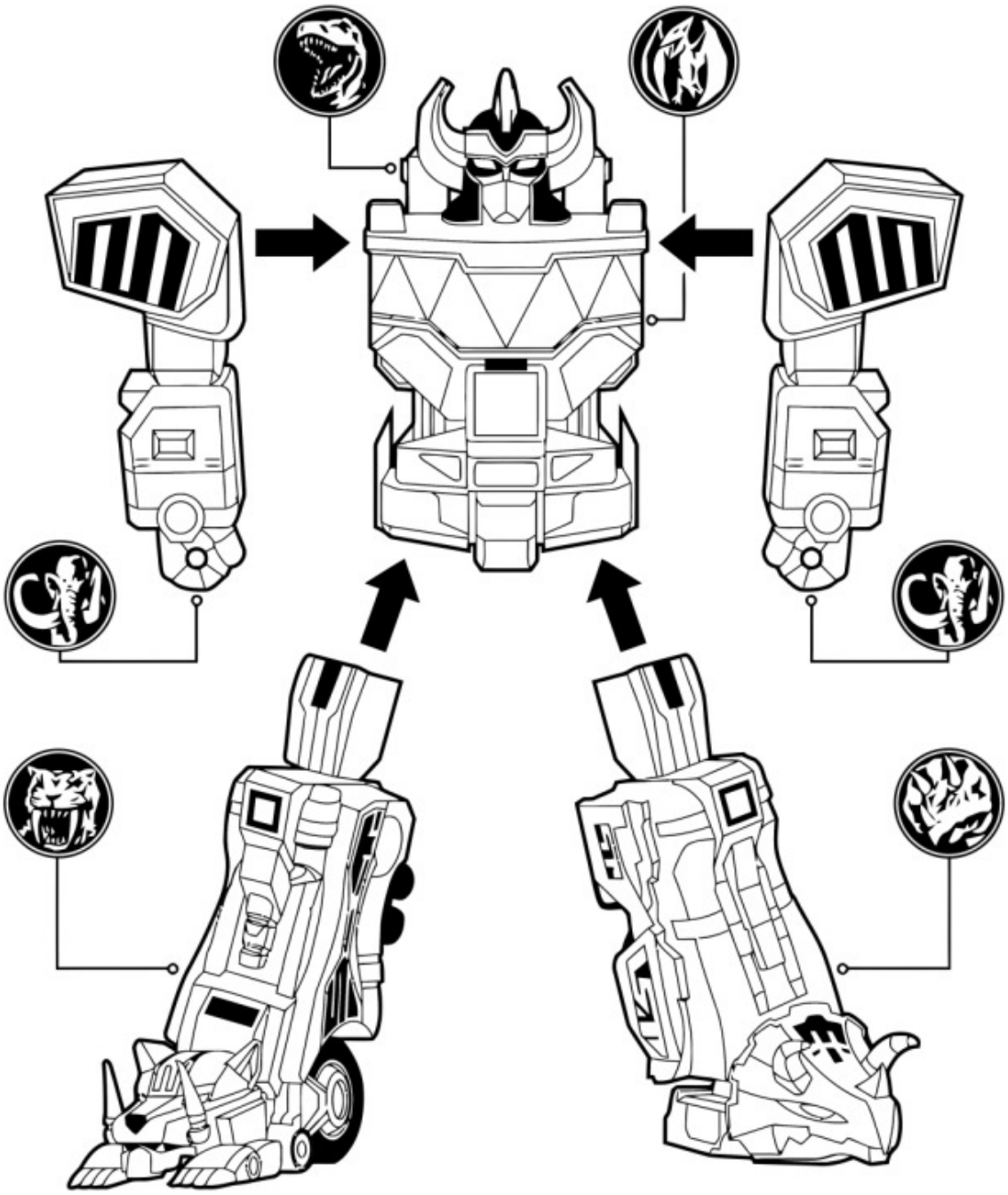
Assembly Project

26 mai 2022

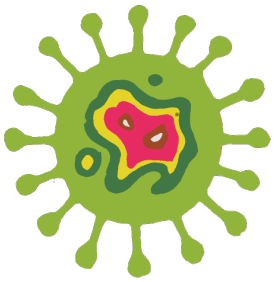


1 Main steps

1. Pre-processing
2. Assembly
3. Evaluation
4. Comparison
5. Repeat

2 Context

During this session we will suppose, as an unlucky but not so surprising scenario, that there is a viral outbreak in Krumlov Luckily, we were able to produce datasets and successfully extracted viral sequences using the nearby sequencers available in every Czech city ! This virus seems unexpectedly hard to assemble. Fortunately, we have dozens of talented bioinformaticians that will gather their skills to disclose this virus' secrets.



3 Available Data

All sequencing data and reference sequences are available in the *assembly* folder.
For the first part of the practical, you can access three available datasets in the folder *synthetic_datasets* :

- Illumina paired-end short-reads (Paired_Illumina)
- Pacific Biosciences High Fidelity reads (HiFi_PB)
- Ultra-long Oxford Nanopore reads (Ultra_long_ONT)

Using one or several of these datasets, you should be able to get a good assembly of the virus, hopefully…

4 Take a look at your data

Suggested duration : 30min
In this part, you will use **seqkit** tools (or your exceptional scripting skills) to investigate the properties of the different datasets. This page https://bioinf.shenwei.me/seqkit/usage/ will help you as you get to know `seqkit`. Using this library, you should be able to answer the following questions for each dataset :

- How many bases are there ?
- What is the mean read length (the N50)?
- How long is the longest (shortest) read ?
- Can you estimate a coverage knowing that we gauge the virus genome size to be around 40kb ?
- For high coverage datasets, you can build smaller datasets via sub-sampling for quick tests
- If the sub-sampled datasets can contain the longest reads it is even better !

5 Your first assembly !

Suggested duration : 15min
This part aims at obtaining a quick, initial assembly of the virus genome using any of the three datasets. You could use any of the assemblers that have been pre-installed on your instance, listed below. "But, which one", you ask ? The point of this practical session is to let you take the initiative and pick one ! So, for this first attempt, if the assembler asks for any parameter, try to guess a reasonable value but don't over-think it. At this point, it does not matter if the assembly is of poor quality. During this step, you can try to work with a sub-sampling of your datasets to accelerate the process.

6 Available Assemblers

Here we list and provide a brief description of the available tools you can use (feel free to install your favorite assembler if it is not listed here). Most assemblers will display a list of arguments if you run them without parameters. Otherwise take a look at their GitHub pages.

6.1 Short reads assembler :

- SPAdes `github.com/ablab/spades`
- Minia `https://github.com/GATB/gatb-minia-pipeline`
- Megahit `github.com/voutcn/megahit`

6.2 Long reads assemblers :

- Raven `github.com/lbcb-sci/raven`
- Flye `github.com/fenderglass/Flye`
- Canu `github.com/marbl/canu`

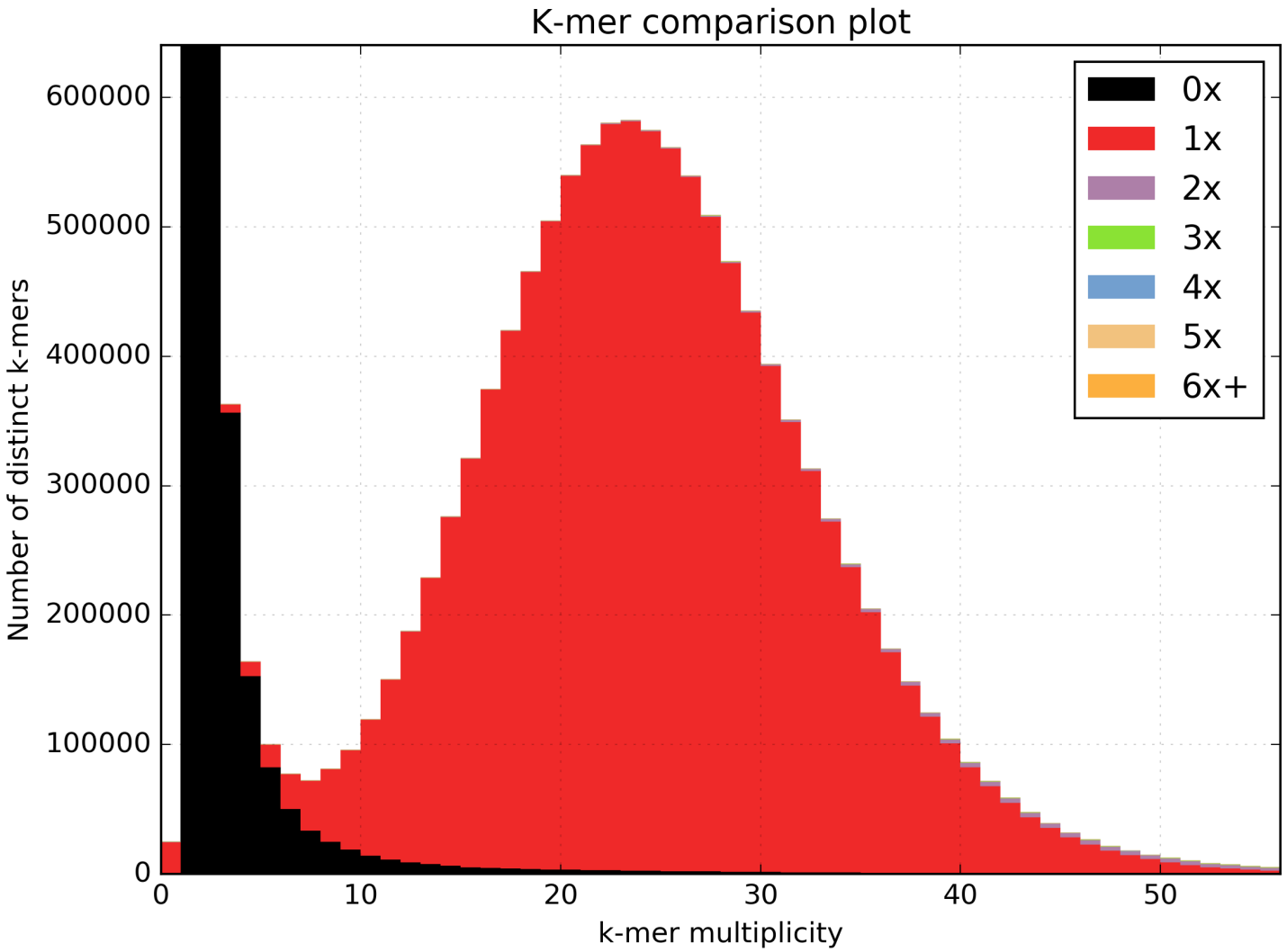
Spades is an Illumina assembler designed for prokaryotic and small eukaryotic genomes. It does an excellent job at assembling bacteria with short-read sequencing, either multi-cell or single-cell data, and small metagenomes. It uses multiple sizes of k (k-mer size in the de Bruijn graph) to construct the best possible contigs. It generally takes longer time and memory than other assemblers. It can also improve its contigs using long reads.
TIP : Spades runtime can be long. You can use a single k size to get results faster with the `-k` option.
TIP : You may skip the correction module to get results faster with the `--only-assembler` option since you have no FASTQ file.
Minia is an Illumina assembler designed to be resource-efficient and able to assemble very large genomes. You can run the GATB pipeline that tries to remove sequencing errors from the reads (Bloocoo), generates contigs (Minia) and scaffolds them (BESST).
TIP : You can also use GATB-pipeline without correction nor scaffolding, it will produce a better-quality assembly than Minia alone due to multi-k.
MEGAHIT is an Illumina assembler designed to assemble large metagenomic experiments. It is very conservative and can assemble even low coverage regions. It is very fast and memory-efficient even though it uses several k-mer sizes.
TIP : Like Spades, you can specify your own k-mer sizes to use with `--k-list` parameter
Raven is an overlap graph assembler for long-reads, based on existing components : Minimap (overlap detection) and Racon (polishing), and use them to produce a clean and contiguous assembly.
TIP : The polishing step can be costly, you have control on the amount of rounds using `-p, --polishing-rounds <int>` option.
Flye is a long-read assembler based on an original paradigm. It builds a repeat graph (conceptually similar to de Bruijn graphs) but uses approximate matches between reads. Flye is also able to assemble metagenomes.
TIP : You can obtain a `.gfa` file with the `--graphical-fragment-assembly` option.
Canu is one of the reference assemblers for long-reads data. It gives good results, however, it might take a while to run IT in the context of this workshop.
TIP : Canu has a specific mode for the different long read data type, use the right mode according to your input.

7 Assembly evaluation

Suggested duration : 15min
Once you get an assembly, you will want to assess your contigs quality. For this, you have multiple options.

7.1 *De novo* evaluation

You can use the **KAT** tool to evaluate your contigs *without* using a reference genome. The idea is to compare the k-mer spectrum of a sequencing experiment (Illumina or HiFi preferably) of your individual with the k-mers present in your contigs. You expect the frequent k-mers to be present in your contigs (red in the figure below) and the rare k-mers to be absent (black).
The k-mer spectrum analysis can be performed following this guide kat.readthedocs.io/en/latest/walkthrough.html#genome-assembly-analysis-using-k-mer-spectra. This page will also give you pointers on what kind of problems this plot can highlight.



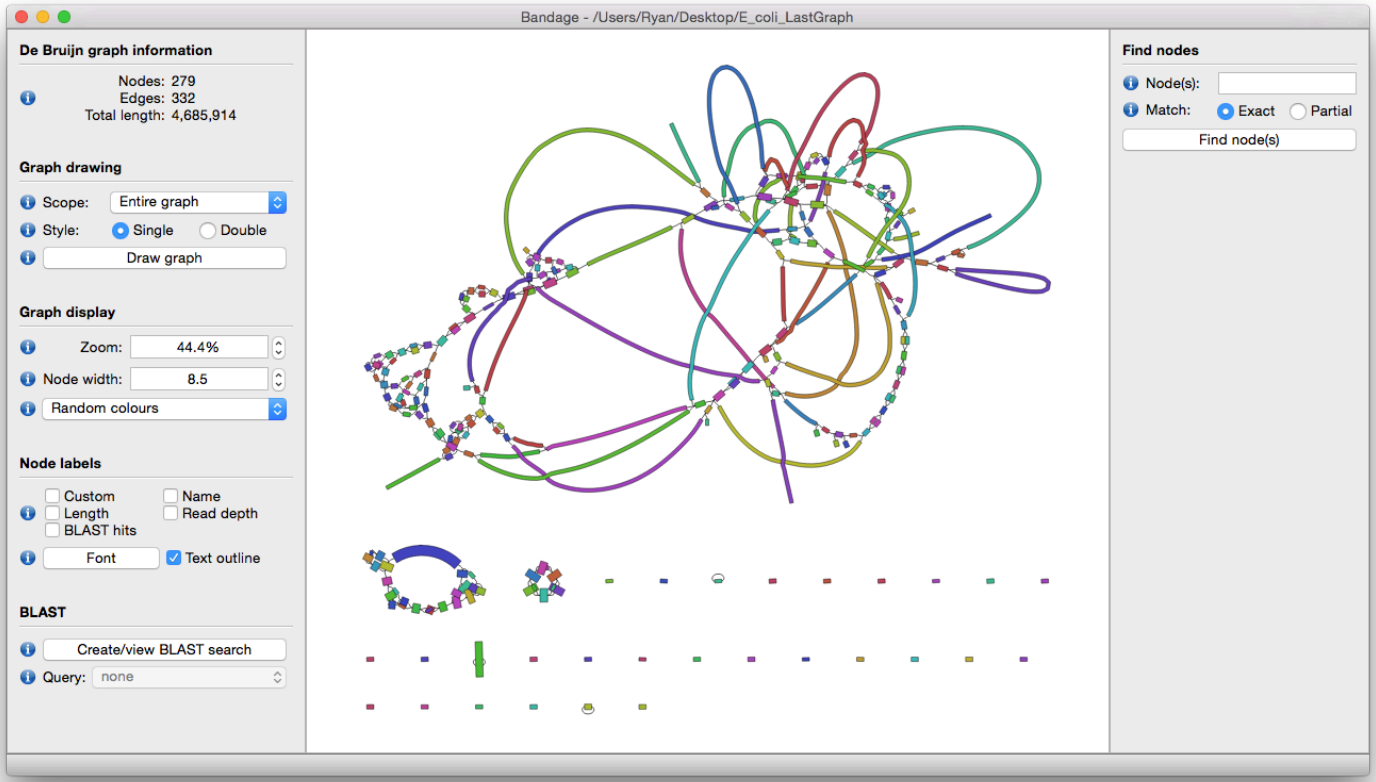
7.2 Reference-based evaluation

You can compare your assembly to a reference using the **QUAST** tool to assess its quality. A nice manual can be found here quast.sourceforge.net/docs/manual.html. The idea is to align your contigs on your reference genome and compute stats by analyzing their alignment.
The main questions that Quast will answer for you are :
How many large/small misassemblies were made by the assembler ?
What part of your genome is covered by your contigs ?
How contiguous is your assembly (N50/N75/NGA50/NGA75 metrics)

But for this, you need a reference...
In real life you often do not have an "actual" reference genome. For the propose of the practical we give you one, feel free to use it, but do not look at it too closely, it will spoil the fun!
Note that you can give Quast multiple assemblies simultaneously for comparison purposes.

7.3 Visualize your assembly graph

Use the **Bandage** software to visualize the assembly graph. Generally, it is the file that ends with `.gfa`. You may skip this step if you do not have a gfa file.



Once you can look at your graph, you can make the following observations :
Is your graph well connected ?
How many disjoint components are there ?
This indicates whether you had sufficient coverage to perform the assembly. It could also mean that the sequencer produced some sequences that did not overlap with one another.
Does the graph has many branching nodes ?
This indicates variants or repetitions that the assembler could not resolved.

8 Let the fun begin !

Suggested duration : ∞
Now is the time for your creative side to shine ! Try different datasets or combinations of datasets with the available tools and various parameters. Test your contigs with KAT, Quast, or Bandage to see if they have good properties. When you are satisfied, please keep your assembly safe in a folder, as it will ~~save Krumlov under the virus attack~~ be evaluated and compared with the other participant assemblies. To do so please fill the following online form https://docs.google.com/spreadsheets/d/1ALvyLAzDbqy70qrjWxu3tI3teJRUPbJOrJauI7D7ti4/edit?usp=sharing
Once you get a "good" assembly, you should be able to answer the following questions :

1. **Why is this genome so hard to assemble ?**
2. **Why the genome size is hard to determine here ?**
3. **From which continent the patient zero of this outbreak may be from ?**

9 Real world assembly

As you may have guessed, this practical uses synthetic data (ie a handcrafted genome made for the purpose of this practical). We chose to build a hard to assemble virus for the computing steps to be fast (and avoid long and boring waiting time). In the real world, even a bacteria is an order of magnitude larger than this virus. To get insight into real data assembly, you can assemble a mycobacterium, one of the smallest known bacteria (around 500kb). Keep in mind that most bacteria are way larger (around five megabases), and you may understand why we chose to assemble a small virus.
To perform this "real" assembly you have two datasets :

- Illumina Miseq paired-ended short-reads (SRR13195562)
- Oxford Nanopore reads (SRR13195563)

You also have a mycobacterium reference genome to evaluate this assembly (you can use KAT too).
You can do the whole practical once again using this data !
TIP : Take a look at your data before doing anything (the coverage in particular).

10 Treasure hunt

If you are brave enough, you can try to assemble this set of "reads" :
CAGATGTCCT
GACCTTTTCT
TAAGATCTTT
TCTTTAGCCG
TTTCTTCTTT
GAGCTTTACT
TTACTTCATT
TCATTCACAT
CACATGAGCT
GTCCTGAACA
GAGCTGATCA
TCTTTCAGAT
AGCCGGACCT
TATGATCATT
TCATTAGGCG

AGGCGGAGCT
HINT : There are NO sequencing errors
HINT : Overlaps are at least of length 5