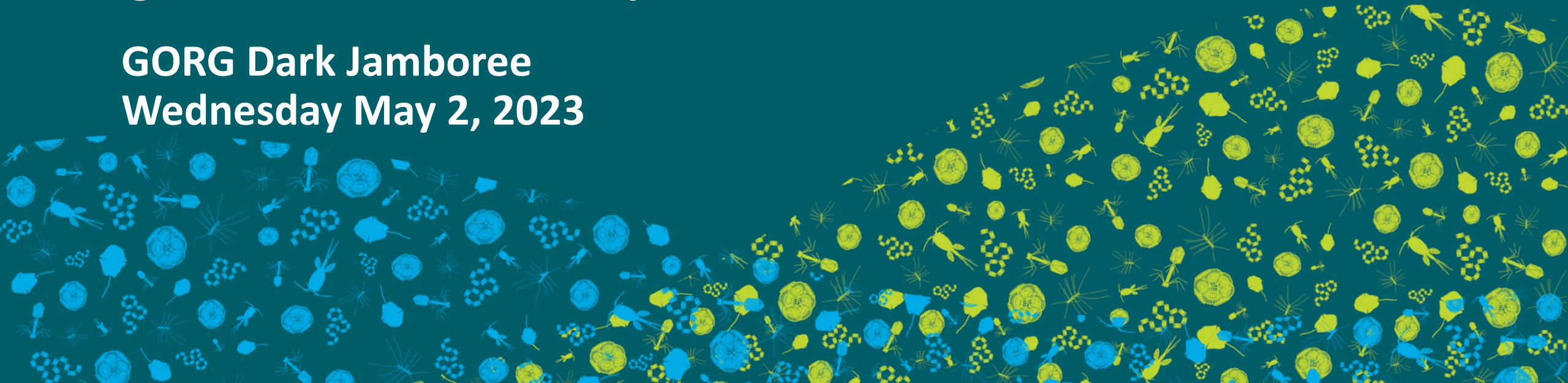
Bigelow | Laboratory for Ocean Sciences

GORG Classifier

Greg Gavelis, Ramunas Stepanauskas, Joe Brown

GORG Dark Jamboree
Wednesday May 2, 2023



Goal of Kaiju & Kraken

"One of the major biological questions in metagenomics is the inference of the composition of a microbial community, that is, the relative abundances of the sampled organisms."

i.e. Who are the organisms in this sample?

How Kraken & Kaiju work (Broadly)

Maps:

Reads to reference proteins (A -> B)

Proteins to the reference organism's NCBI taxon (B -> C)

Achieves *very fast mapping* or reads to taxa. (A -> C)

How Kraken & Kaiju work (Broadly)

Maps:

Reads to ref proteins (A -> B)

Ref proteins to ref organism's NCBI taxon (B -> C)

Achieves *very fast mapping* or reads to taxa.

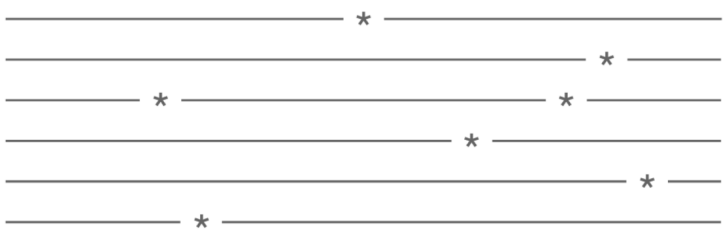
How? Doesn't rely on local alignment. Uses *exact* alignment to kmers from the reference proteins.

Kaiiju read mapping algorithm

For each read…

1. Translation

Translate nucleotide sequence into amino acid sequences by the six possible reading frames and split into *fragments* at stop codons.

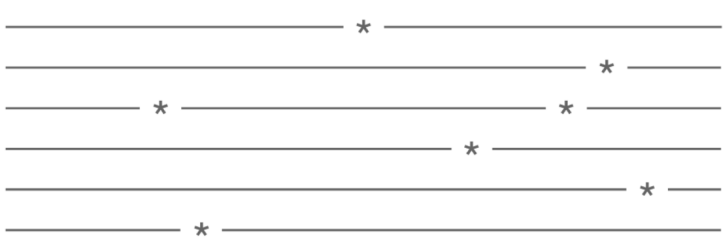


Kaiiju read mapping algorithm

For each read…

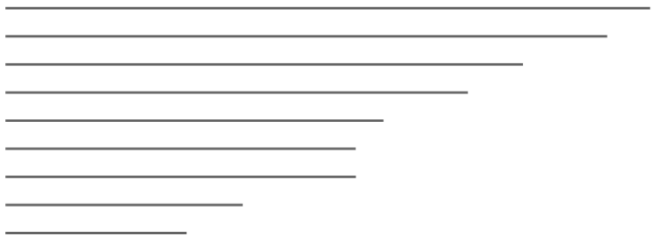
1. Translation

Translate nucleotide sequence into amino acid sequences by the six possible reading frames and split into *fragments* at stop codons.



2. Sorting

Sort fragments by length



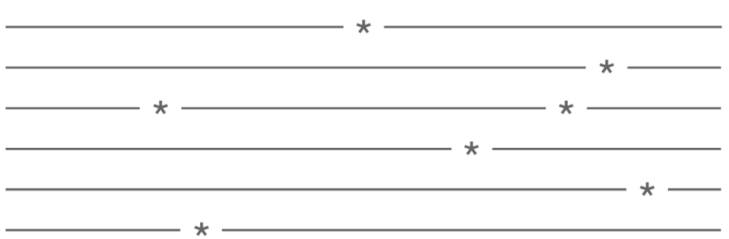
Kaiiju read mapping algorithm

For each read…

looks for *exact* AA match of length k

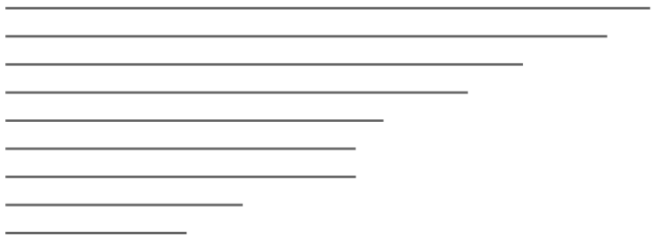
1. Translation

Translate nucleotide sequence into amino acid sequences by the six possible reading frames and split into *fragments* at stop codons.



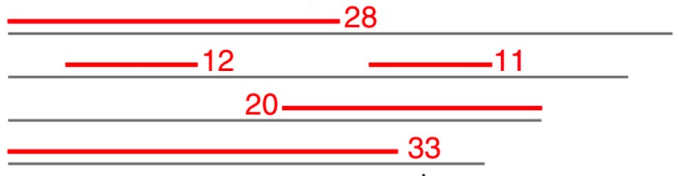
2. Sorting

Sort fragments by length



3. Database search

Aligns kmers with length > m



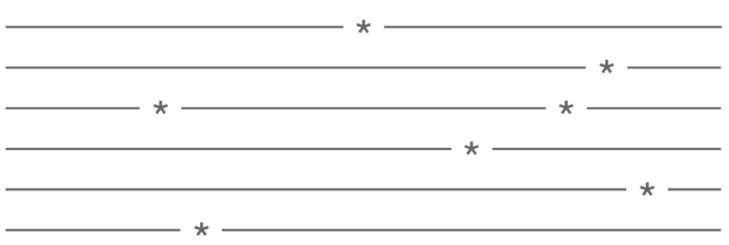
Kaiiju read mapping algorithm

For each read…

looks for *exact* AA match of length k

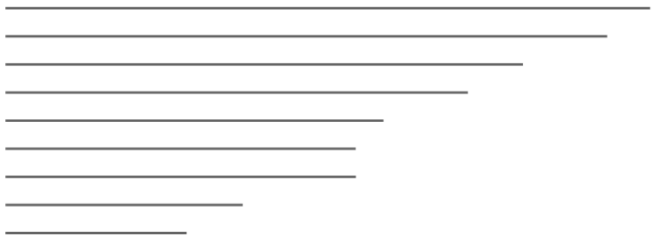
1. Translation

Translate nucleotide sequence into amino acid sequences by the six possible reading frames and split into *fragments* at stop codons.



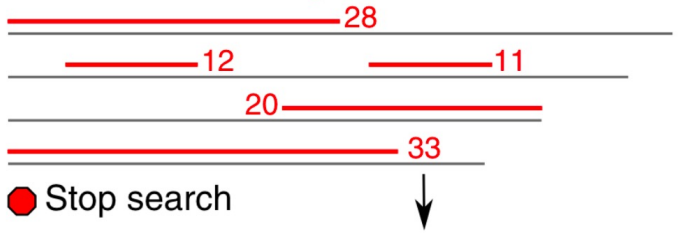
2. Sorting

Sort fragments by length



3. Database search

Aligns kmers with length > m



Stop search

Assign read to the taxon with longest match

Kaiiju read mapping algorithm

For each read…

looks for *exact* AA match of length k

1. Translation

Translate nucleotide sequence into amino acid sequences by the six possible reading frames and split into *fragments* at stop codons.

2. Sorting

Sort fragments by length

3. Database search

Aligns kmers with length > m

28

12

11

20

33

Stop search

Assign read to the taxon with longest match

What happens if two reference genomes both have an *equally good kmer match*?

Kaiiju read mapping algorithm

For each read…

looks for *exact* AA match of length k

1. Translation

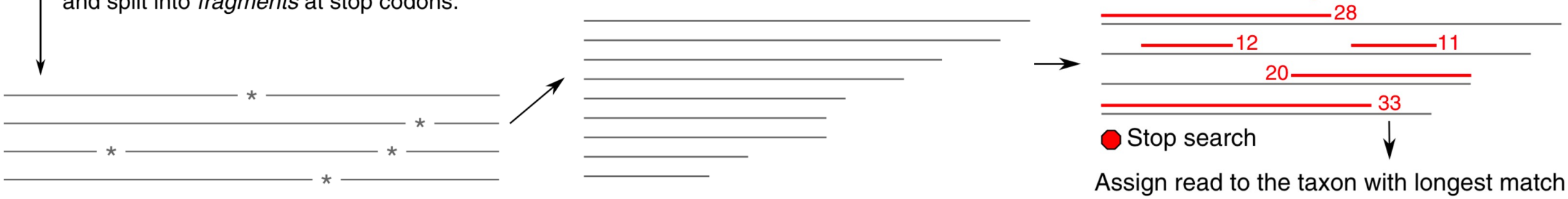
Translate nucleotide sequence into amino acid sequences by the six possible reading frames and split into *fragments* at stop codons.

2. Sorting

Sort fragments by length

3. Database search

Aligns kmers with length > m



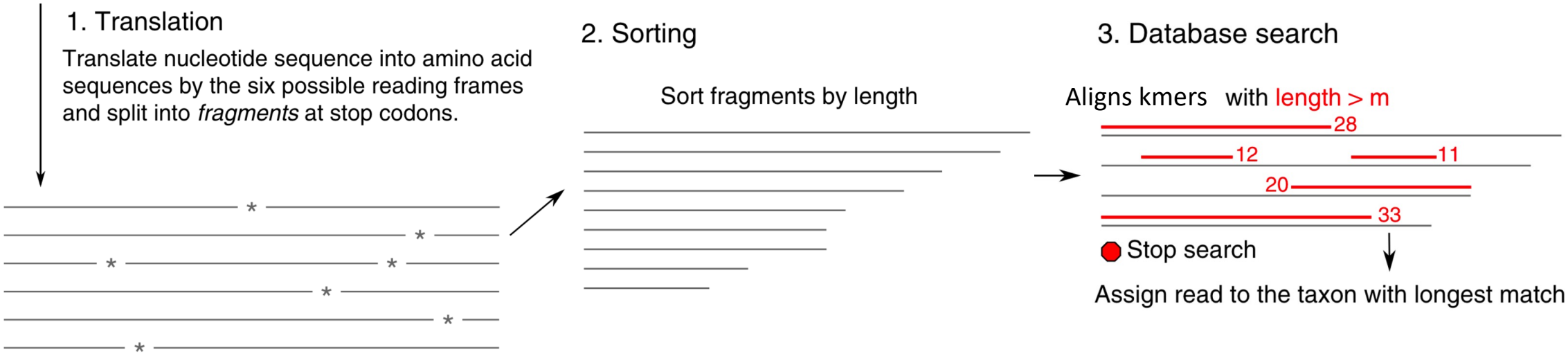
What happens if two reference genomes both have an *equally good kmer match*?

Kaiju goes higher in the tree to the last common ancestor (LCA) with that kmer. Assigns the read to that ancestor.

Kaiiju read mapping algorithm

For each read…

looks for *exact* AA match of length k



What happens if two reference genomes both have an *equally good kmer match*?

Kaiju goes higher in the tree to the last common ancestor (LCA) with that kmer. Assigns the read to that ancestor.

Note: Kaiju's innovation is using variable kmer lengths (Kraken uses fixed length). Kaiju first attempts 63AA exact matches, but retries down to 7AA if necessary

Kaiiju also allows *mismatches* at *end* of kmers

"greedy mode" (what we use)

1. Translation

Translate nucleotide sequence into amino acid sequences by the six possible reading frames and split into *fragments* at stop codons.

2. Sorting

MEM Sort fragments by length > m

Greedy Sort fragments by score > s

3. Database search

Find MEMs with length > m

28
12
11
20
33

Stop search

Assign read to the taxon with longest match

Find matches with score > s

75
140
198

Stop search

Assign read to the taxon with highest scoring match

So Kraken & Kaiju are built on…

1. Protein alignment software:

   Burrough's Wheeler Aligner

2. A reference protein databases:

   NCBI's nr (just the microbes)

   OR

   Refseq

3. A taxonomic framework for those reference taxa

   NCBI taxonomy

So Kraken & Kaiju are built on…

1. Protein alignment software:

   Burrough's Wheeler Aligner

2. A reference protein databases:

   NCBI's nr (just the microbes)

   OR

   Refseq

3. A taxonomic framework for those reference taxa

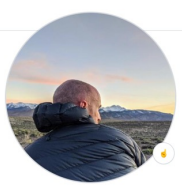
   NCBI taxonomy

Do you see any limitations here?

Advantages of GORG classifier, pt. I

BigelowLab / gorg-classifier

Coded by Joe Brown



1. Recruits to more taxonomically diverse references:

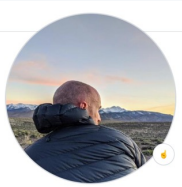
GORG-Tropics (~8000 classified SAGs)
--integrated by Joe ~2019

GORG-Dark (> 9000 classified SAGs)
--integrated by me a week ago. ago (in **Beta**)

Advantages of GORG classifier, pt. I

BigelowLab / **gorg-classifier**

Coded by Joe Brown



1. Recruits to more taxonomically diverse references:

   GORG-Tropics (~8000 classified SAGs)
   --integrated by Joe ~2019

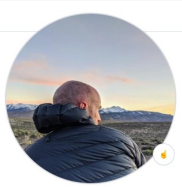
   GORG-Dark (> 9000 classified SAGs)
   --integrated by me a week ago. ago (in **Beta**)

2. Within an improved taxonomic framework
   NCBI taxonomy has massive blind spots and biases.
   So GORG-Dark classifier uses GTDB taxonomy. (<- **names.dmp** & **nodes.dmp**)

Advantages of GORG classifier, pt. I

BigelowLab / **gorg-classifier**

Coded by Joe Brown



1. Recruits to more taxonomically diverse references:

   GORG-Tropics (~8000 classified SAGs)
   --integrated by Joe ~2019

   GORG-Dark (> 9000 classified SAGs)
   --integrated by me a week ago. ago (in **Beta**)

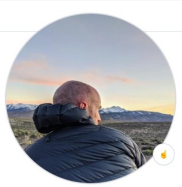
2. Within an improved taxonomic framework
   NCBI taxonomy has massive blind spots and biases.
   So GORG-Dark classifier uses GTDB taxonomy. (<- **names.dmp** & **nodes.dmp**)

   *Currently GORG-**tropics** still uses ncbi taxonomy*
   Eventually we should re-release it w/ GTDB

Advantages of GORG classifier, pt. II

BigelowLab / gorg-classifier

Coded by Joe Brown



Joe Brown
brwnj

3. Biggest innovation (by Joe).

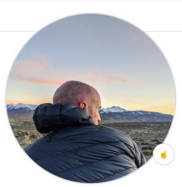
Also maps reads ***to functions***!

(annotations by KEGG, Swissprot, CAZy, etc.)

Advantages of GORG classifier, pt. II

BigelowLab / gorg-classifier

Coded by Joe Brown



Joe Brown
brwnj

3. Biggest innovation (by Joe).

Also maps reads ***to functions***!

(annotations by KEGG, Swissprot, CAZy, etc.)

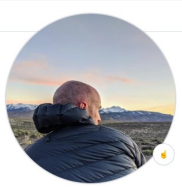
So we can ask:

Who are the organisms in this sample *<u>and what are they doing?</u>*

Advantages of GORG classifier, pt. II

BigelowLab / gorg-classifier

Coded by Joe Brown



3. Biggest innovation (by Joe).

Also maps reads ***to functions***!

(annotations by KEGG, Swissprot, CAZy, etc.)

So we can ask:

Who are the organisms in this sample *and what are they doing?*

For metagenomics: Which taxa are in the sample and what is their metabolic potential?

For metatranscriptomes: Which taxa are in the sample and what are they expressing?

BigelowLab / gorg-classifier

How does GORG classifier work?

- It's a pipeline built on Kaiju 

- Runs Kaiju with:
  - new reference DB (proteins from GORG Dark or Tropics)
  - modified taxonomy (GTDB for Dark)

- After Kaiju, runs extra steps to incorporate annotations

- Output: A recruitment table -> {ID}_annotations.txt.gz
  - Massive, 1 row per read.

BigelowLab / gorg-classifier

How does GORG classifier work?

- It's a pipeline built on Kaiju 

- Runs Kaiju with:
  - new reference DB (proteins from GORG Dark or Tropics) <- **index.fmi**
  - modified taxonomy (GTDB for Dark) <- **names.dmp** & **nodes.dmp**

- After Kaiju, runs extra steps to incorporate annotations. <- **annotations.tsv**

- Output: A recruitment table -> {**ID**}**_annotations.txt.gz**
  - Massive, 1 row per read.

To run GORG classifier, pt. I

BigelowLab / gorg-classifier

Program files

Index.fmi, names.dmp, nodes.dmp, annotations.tsv

To run GORG classifier, pt. I

BigelowLab / **gorg-classifier**

<u>Program files</u>

Index.fmi, names.dmp, nodes.dmp, annotations.tsv

<u>Your read files to analyze</u>

E.g. **{ID}_R1.fq.gz** & **{ID}_R2.fq.gz**

Metagenomic or metatranscriptomic. Paired or unpaired. Zipped or unzipped.

To run GORG classifier, pt. I

BigelowLab / **gorg-classifier**

Program files

index.fmi, names.dmp, nodes.dmp, annotations.tsv

Your read files to analyze

E.g. **{ID}_R1.fq.gz** & **{ID}_R2.fq.gz**

Metagenomic or metatranscriptomic. Paired or unpaired. Zipped or unzipped.

Software dependencies

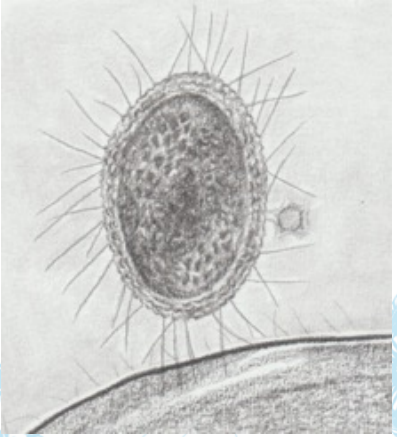
1. Nextflow (including **nextflow.config** file) = Runs bioinf pipelines
2. Singularity (on cluster) or Docker. = Manages containerized programs

Today's Example

- Let's say we did some metagenomic sampling in the anoxic depths of the Baltic Sea (100-200 meters)

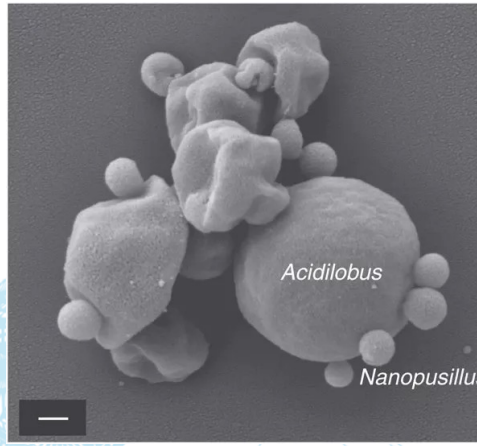
Baltic Metagenome

Baltic Sea photo by Sascha Kilmer



Patescibacteria

wikipedia



Nanoarchaeota

wikipedia

Today's Example

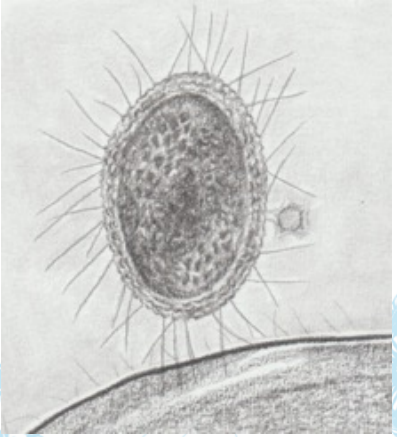
- Let's say we did some metagenomic sampling in the anoxic depths of the Baltic Sea (100-200 meters)

- Have 1 million reads in file:
  - **`SRR11600247.fq`**

- Located in /mnt/storage/data/daily-data/day3/

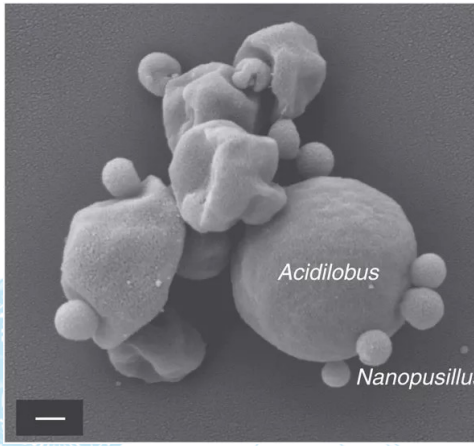
Baltic Metagenome

Baltic Sea photo by Sascha Kilmer



Patescibacterium

wikipedia



Nanoarchaeota

wikipedia

BigelowLab / gorg-classifier

To run GORG classifier, pt. II

Example command:

```
nextflow run BigelowLab/gorg-classifier \
-profile charlie \
--seqs './SRR11600247.fq' \
--mode local \
--names names.dmp \
--nodes nodes.dmp \
--fmi index.fmi \
--annotations annotations.tsv
```

BigelowLab / gorg-classifier

To run GORG classifier, pt. II

Example command:

```
nextflow run BigelowLab/gorg-classifier \
-profile charlie \
--seqs './SRR11600247.fq' \
--mode local \
--names names.dmp \
--nodes nodes.dmp \
--fmi index.fmi \
--annotations annotations.tsv
```

Runs the code hosted on Bigelow's github

BigelowLab / gorg-classifier

To run GORG classifier, pt. II

Example command:

```
nextflow run BigelowLab/gorg-classifier \
-profile charlie \
--seqs './SRR11600247.fq' \
--mode local \
--names names.dmp \
--nodes nodes.dmp \
--fmi index.fmi \
--annotations annotations.tsv
```

Runs the code hosted on Bigelow's github

On our reads

BigelowLab / gorg-classifier

To run GORG classifier, pt. II

Example command:

```
nextflow run BigelowLab/gorg-classifier \
-profile charlie \
--seqs './SRR11600247.fq' \
--mode local \
--names names.dmp \
--nodes nodes.dmp \
--fmi index.fmi \
--annotations annotations.tsv
```

Runs the code hosted on Bigelow's github

On our reads

'Local' modes looks fo local files (i.e. names.dmp etc. in your current dir)

Now it’s running ....

BigelowLab / gorg-classifier

```
N E X T F L O W  ~  version 21.10.6
Launching `BigelowLab/gorg-classifier` [peaceful_fermat] - revision: 23b3f7092d [master]
NOTE: Your local project version looks outdated - a different revision is available in the r
emote repository [5ae3a4beb7]


================================================================
GORG Classifier, Single Cell Genome Center, Bigelow Laboratory
================================================================

#### Homepage / Documentation
https://github.com/BigelowLab/gorg-classifier
#### Citation
https://doi.org/10.1016/j.cell.2019.11.017
#### Reference data
URL: https://osf.io/pcwj9
License: Attribution-NonCommercial 4.0 International.
#### Authors
Joe Brown <brwnjm@gmail.com>
----------------------------------------------------------------
```

- … And it’s done

BigelowLab / gorg-classifier

```
Output directory                    : ./results
Kaiju mismatches                    : 3
Kaiju minimum alignment length      : 11
Kaiju CPUs                          : 8
-----------------------------------------------------------------------------


executor >  pbspro (4)
[9b/69211d] process > run_kaiju (SRR11600247)              [100%] 1 of 1 ✔
[d3/e8df20] process > add_taxonomy (SRR11600247)           [100%] 1 of 1 ✔
[0e/3ad0b5] process > add_functions (SRR11600247)          [100%] 1 of 1 ✔
[72/09e68f] process > summarize_annotations (SRR11600247) [100%] 1 of 1 ✔
Completed at: 02-May-2023 18:02:51
Duration    : 7m 32s
CPU hours   : 0.6
Succeeded   : 4
```

Let’s explore the results!

Copy these to your working directory…

Classifier’s output table:
/mnt/storage/data/daily-data/day3/SRR11600247_annotated.txt

This morning’s notebook:
/mnt/storage/data/daily-data/day3/Exploring_classifier_results.ipynb

- &

Notes about Nextflow

- **Learning curve**: Yes, Nextflow is more involved than writing a pipeline in bash. Takes some time to learn, but less so to run other people's pipelines.

- **Nice tutorial:**
- https://carpentries-incubator.github.io/workflows-nextflow/aio/index.html

- **Singularity to run on a cluster**: Consult with your system administrator about installing Singularity if you need to use it on a cluster. (Or just use Docker if you're running on a personal machine).

Advantages of nextflow

- **Safe**r than running pipelines in bash (E.g. can't delete your own file system).
- More **flexible** than Snakemake or other bioinf' pipeline languages.
- **Portability**: it installs & runs containerized programs for you.
- **Efficiency**: It saves time/energy by never rerunning steps that were successful.