CRISPR Hunting

Today we'll be identifying CRISPR sequences in a set of genomes from strains isolated from Sippewissett salt marsh mats. They appeared as **w**hite **o**n the **g**reen background of the mats, and are referred to here as WOGs. They were sequenced with PacBio long-read technology, and are likely to have intact CRISPR arrays - if they have CRISPRs at all.

Step 0: Get ready

Our tool of choice today is `cctk`, which is installed in its own conda environment. Activate it with:

In [ ]:
```
conda activate cctk
```

Make a new working directory and copy over the WOG genomes.

In [ ]:
```
mkdir /YOUR_GROUP/YOUR_DIRECTORY/crispr_hunting
```

In [ ]:
```
cp /datahaus/crispr_hunting/crispr_data.zip .
```

In [ ]:
```
unzip crispr_data.zip
```

Step 1: Find CRISPRs with MinCED

You should have a folder called `WOG_assemblies`, containing .fasta files of contigs from the 4 WOG genomes. To search for CRISPRs in these genomes, ues `cctk minced`. This uses the program MinCED to identify CRISPR repeats in the contigs and extract repeats and spacer sequences. First, make a directory for the output files, then run the command using the WOG_assemblies directory as input. `-m -p` tells cctk to run MinCED and process the output.

In [ ]:
```
mkdir minced_crisprs
```

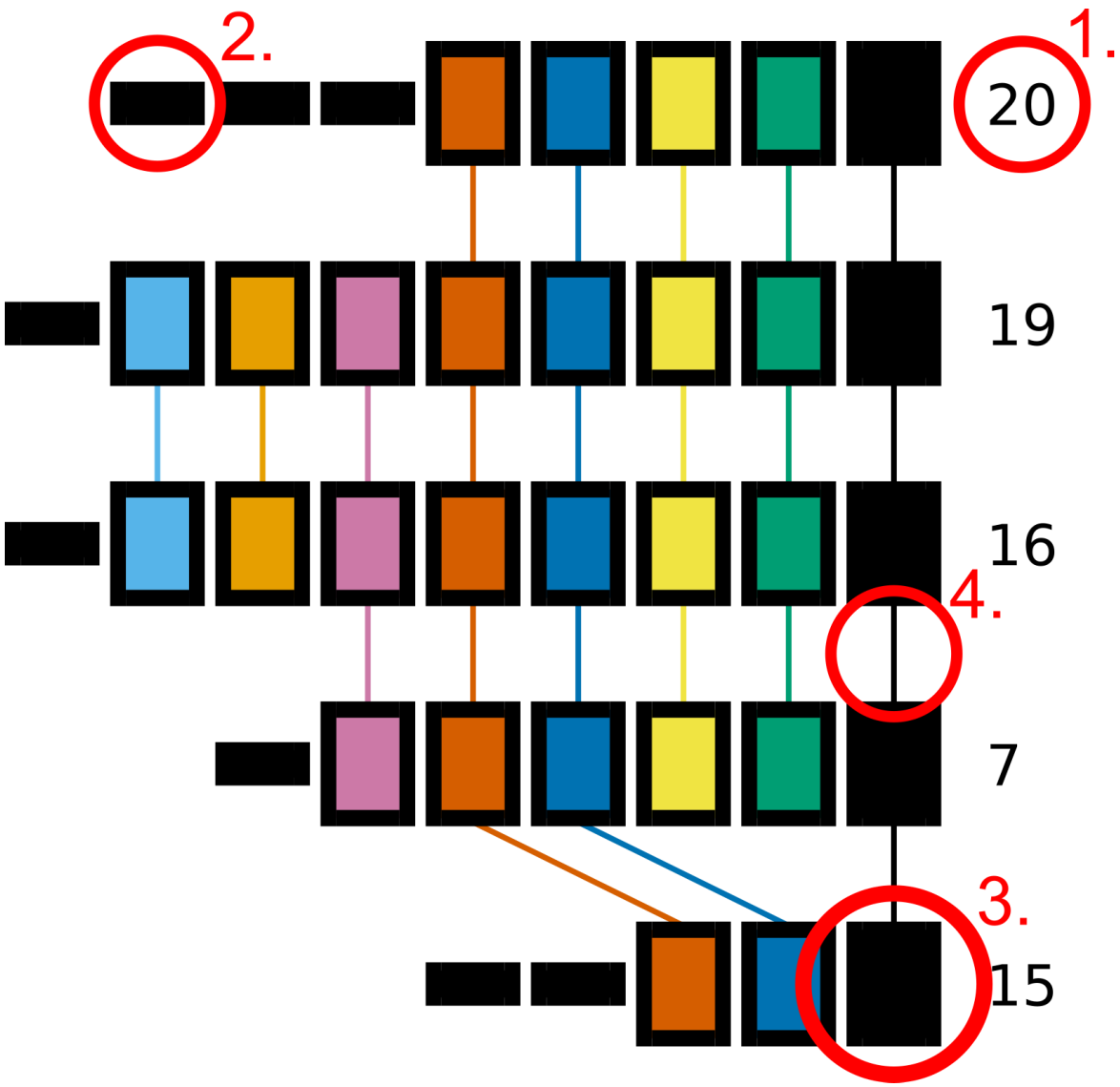
In [ ]:
```
cctk minced -i WOG_assemblies/ -o minced_crisprs/ -m -p
```

Check the output files in `minced_crisprs/PROCESSED`.

- *Do all strains have CRISPRs?*
- *Do the strains with CRISPRs have different spacer arrays?*

Step 2: Visualize CRISPR arrays

cctk's crisprDiff function can generate a plot showing relatedness among CRISPR arrays. We can use it to compare our two types of arrays. An example plot is shown below.



The key elements are:

1) Array ID

2) Spacers unique to a single array on the plot (black line)

3) Spacers found in more than one array (large filled rectangles)

4) Lines connecting shared spacers

In [ ]:
```
cd minced_crisprs/
```

In [ ]:
```
mkdir plots
```

In [ ]:
```
cctk crisprdiff -a PROCESSED/Array_IDs.txt -o plots/all_diff.png
```

In [ ]:
```
display plots/all_diff.png
```

- *Which spacers are shared between array types?*

Step 3: Find targets

Now that we know the spacer sequences of all our arrays, we can use them to find targeted sequences with BLAST. First, we'll check our set of genomes. Build a BLAST database of all 4 metagenomes.

In [ ]:
```
makeblastdb -in blastdb/all_WOG_assemblies.fasta \
-out blastdb/all_WOGs -dbtype nucl -parse_seqids
```

Now look for matches using `cctk spacerblast`. We can provide the array locations to avoid self-matches to the CRISPR array. `-p 100` sets the percent identity of the match, so we'll only be looking for perfect matches.

In [ ]:
```
cctk spacerblast -d blastdb/all_WOGs \
-s minced_crisprs/PROCESSED/CRISPR_spacers.fna \
-p 100 -r minced_crisprs/PROCESSED/Array_locations.bed
```

There are two additional blast databases you can search for viral matches:

- all_shep_viromes, which contains sequenced supernatant from six Sippewissett isolate cultures
- pinksand_meta, which contains a metagenome of Sippewissett pink sand

In [ ]:
```
cctk spacerblast -d blastdb/all_shep_viromes \
-s minced_crisprs/PROCESSED/CRISPR_spacers.fna \
-p 100
```

In [ ]:
```
cctk spacerblast -d blastdb/pinksand_meta \
-s minced_crisprs/PROCESSED/CRISPR_spacers.fna \
-p 100
```

- *Are there hits to any of the databases?*
- *How could you check if these are hits to viruses or other CRISPR arrays?*