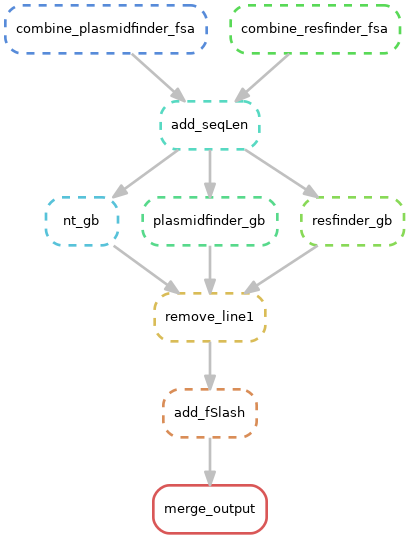
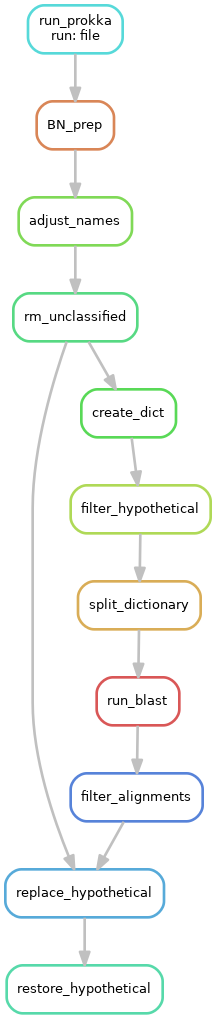
**Annotation pipeline**V1, 08/10/2020   
  
  
**Introduction.**  
Because of the increasing usage of the MinIon sequencer within IDS-BSR-RIVM, and the generation of circular sequenced plasmids and chromosomes in bulk, the need for fast annotation arose.  
  
At first the visualization tools BLAST Ring Image Generator (BRIG), SnapGene and New Plasmid Viewer were considered because of their automatic annotation functionality. But after testing it was found that far too few regions were annotated and no resistance genes were found.  
  
Because it became clear we were in need of dedicated annotation tools NCBI Prokaryotic Genome Annotation Pipeline (PGAP), Prokka, PlasMapper, DDBJ Fast Annotation and Submission Tool & RASTtk were researched and Prokka was chosen as the most fitting annotation tool for this project. Mainly because Prokka and PGAP were documented as having the most accurate results but having one main difference between eachother. While Prokka was very fast it only used a small subset of curated Uniprot data, PGAP was very slow but used the NCBI database as a whole.  
  
In early 2019 the annotation was set-up by using Prokka.  
  
**Requirements.**  
Conda, creating and managing environments  
Snakemake, execution of the pipeline  
Prokka, annotation  
Blast, local pairwise alignment  
These are the main packages, note that a lot of programs like Python will also be installed because it is an dependency of Snakemake.  
  
The set-up for this environment in which the program will run would be:  
*conda create -n Prokka -c bioconda -c conda-forge snakemake Prokka blast* **Databases.**  
Uniprot (curated subset within Prokka)  
ResFinder  
PlasmidFinder  
Custom database  
The custom database is as of this moment still a work in progress, the data within this database is comprised of annotated chromosomes and plasmids downloaded from NCBI.  
  
*ResFinder & PlasmidFinder*  
ResFinder and PlasmidFinder data is retrieved from their bitbucket website. The multiple files containing sequences will go through a small pipeline consisting of python scripts.   
  
First the smaller files will be combined into two FASTA files, a python script will calculate the sequence length for each individual sequence and add it in the header. Sequence length is needed to generate a Genbank format semi-specific for Prokka.   
  
The Genbank files are then combined into one Genbank file. The first line would be removed because it would turn up as a white space, and at the end of the file two forwards slashes would be added to define the end of the record.   
  
The then merged output is used with the ‘--protein’ parameter of Prokka.  
  
As of now the custom database is not included in this process. This database would also be incorporated in the future as such it is also included in the flowchart of the code.  
  
 **Program.**Input for this pipeline is assembled sequence data in FASTA format, provided by the IDS-BSR-AMR team.  
  
As of this moment a script makeConfig.py needs to run beforehand in order to generate a yaml file for the new input files. In the future the code could be structured to monitor a specific input folder and run the program when new files arrive.  
  
Prokka is the first tool to handle the input files. As stated above it uses its standard databases and also the ResFinder and the PlasmidFinder data with the --protein parameter.  
  
After Prokka is finished a custom script adjusts some genenames, originated from the ResFinder and PlasmidFinder data, this is needed because in their original database the genes are also noted with a numeric value, this value is not needed for annotation. Unclassified is also removed from the records because it holds no additional information. rm\_unclassified marks the end of the initial annotation.  
  
Because a high number of hypothetical proteins is found within our data a set of python scripts creates dictionaries of all the annotation records, extracts the hypothetical proteins and their sequence from the dictionary and runs them through BLAST.  
The output is then filtered based on identity percentage and the best blast-hit will replace the original hypothetical protein in the annotation file.  
  
restore\_hypothetical checks all remaining hypothetical proteins on underscores and numerical values, deleting the characters if found, marks the end of the annotation.