Progress Report #2

Jeremy Fisher, Simon Levine,Tomas Matteson and Siddharth Reed

(1) A description of your planned design for solving the problem. This should be more than an uninformed plan. By now, you should have tested software, sketched out algorithms, and have a solid start towards creating a solution.

We already have the beginnings of the first two steps: (1) filtering out non-host transcripts; and (2) assembling these into contiguous sequences.

We have outlined a rough solution to the entire pipeline implemented using a Snakemake workflow, where so far we have defined the underlying software dependencies and command line invocations.

This pipeline currently consists of the (single-ended) STAR alignment step, whereby host genomes are ideally identified and filtered from the inputted novel sequence, followed by an extraction step for the non-host genomes only (per the problem constraints). Currently, we are using the sample *STAR* reference genome (http://labshare.cshl.edu/shares/gingeraslab/www-data/dobin/STAR/STARgenomes) and index, as well as associated files, for pipeline development. We provide a bash script to download these files and will incorporate this download stage into our main snakemake workflow.

Next, we have written a step to isolate unmapped reads using *samtools*. We discovered a binary flag that indicates that such is the case for individual reads. Then, we used a simple bash script to convert the SAM file back into a FASTQ file ready for ingestion into the contig assembly step. This step has also been outlined, whereby we have decided on the SPADES package using a viral-specific command-line flag.

We are not certain, however, that these “rules” connect to each other correctly -- that is, that major components such as *STAR* and *samtools* save and export files in the expected location. We are waiting until we finalize suitable data, like our RNAseq dataset, to polish these and make sure they flow together. An example issue is that single-ended and pair-ended RNAseq datasets are processed differently by *STAR*. Note again that we are using a provided reference genome and index.

So far, we have described the steps we have already taken. What follows remains to be implemented.

After isolating and assembling the viral genomes present in our input files we want to BLAST them (using the blastn command line tool) against all known human viruses to see which, if any, viruses are present in the sample and report those. The report should contain information about which virus and gene the read was mapped to and the similarity of the read to the suspected viral genome.

Next, we incorporate a model to detect viral ORFs. Currently we plan to use *Prodigal* to identify to do so in the suspected viral sequences from the previous step. *Prodigal* is unsupervised and will learn the patterns of viral ORFs from the viral sequences themselves. This alleviates the need to label our own dataset or produce a complementary dataset and train a model in a supervised fashion.

Finally, we would like to use bioconductor packages (like ensembldb and bioMart) to query a protein database for structural element identification and other functional annotations of viral proteins. We have not determined the proper software package yet, but we are confident that such tools exist that fill this use case.

This should ultimately produce a JSON or JSON-lines file that relates a viral contig to its ORF(s) and functional/structural element properties.

(2) A description of which parts of the problem you've have decided are too hard to tackle during the semester

Although most features we have proposed are feasible to implement individually, we may not have time to implement them all. Here, we list several features we would like to include but fall under this category:

- Generating *STAR* index files from the input genome (we currently only support pre-computed indices).
- Supporting both paired-end and single-end reads for input.
    - We support both but have yet to test paired-end.
- Performing quality control on the input reads with Trimmomatic and *FASTQC*
- Visualizing the functional annotation data with an interactive dashboard such as with R Shiny

(3) A timeline for completing the work you've chosen to do.

We hope to:

- Finish the testing, download and alignment step using *STAR* by the end of this week (Friday Feb 19th)
- Implement the *SPAdes* step to assemble the viral RNA reads by next week (Friday Feb 24th)
- Set up the *BLAST* step to identify sequences from the input most similar to known viral sequences by the end of this week (Friday Feb 19th).
- Implement a baseline functional ORF search and/or classifier by February 25th.
    - Note that we have some confusion on ORF "prediction" from assembled viral sequences. We propose to use *Prodigal*.
- Finish structural element search from viral sequences (to protein residues) by February 25th.
- We hope to have a working version of the entire pipeline by March 4th.