Final Assignment Report

Beatriz Gascón Lourenço

In order to acquire the “assembly.fasta file”, I first created a nano script called “config” in order to setup my parameters for the assembly command. I set maximum read length as 90, maximum read length in this library as 90, sequence reverse as 0, average insert size as 200, Asm flags as 3 and minimum aligned length to contigs for a reliable read location as 32. In the script, I also added the absolute paths to the files I wanted to assemble (7079_1.fq and 7079_2.fq). Following this, I used the script as a reference to run SOAPdenovo using kmer values from 35 to 75 while checking the scaffStatistics file after every output in order to find the optimal N50 length value for the annotation. The kmer value chosen was 65 with an N_{50} length of 298 and number of 1243.

Having found the ideal kmer, I used the “. scafSeq” file from the kmer 65 output and renamed the file as “proto_assembly.fasta”. This was done in order to obtain the sequences that were going to be included in the required fasta file for the assignment. I then wrote a python script named “len”, in which I imported the SeqIO and sys packages and looped through the “proto_assembly.fasta” file; the script contained an if statement that outputted to standard output only the sequences with a length bigger or equal than 300 base pairs. The SeqIO package allowed for the standard output to also print out the sequences ID’s preceded by a “>” symbol, which manually formatted the output into a fasta file. The output of this script on “proto_assembly.fasta” was pipped into a file called “assembly.fasta”, concluding the first part of the assignment.

For the second part of the assignment, I first made a blast database I called “mydb” using the file Ath11.fa, this file contained an annotated genome from the plant *Arabidopsis thaliana* that was downloaded into my final directory from “michdeyh/bd/” along with Ath11.txt. Then I ran BLASTP on the “assembly.fasta” file using “mydb” as a reference genome. The BLAST results were saved in a file called “a_1” and restricted to only output sequence ID (qseqid), sequence length (slen) and sequence title (stitle), this last parameter was used in order to fulfill the description requirement of this assignment. Since there were no instructions for e-value restriction, I did not use e-value as a treshold for the BLAST output.

The obtained BLASTP output now needed to be made into a clean, tab-delimited file. For this purpose, I built a nano script called “parseblp” that looped through the “a_1” file, divided it into lists based on tabs (“\t”) and printed to standard output only the data that would be going into the blast MySQL table for annotation, which was comprised only of Scaffold number (ID correspondent in the fasta file), length of nucleotide sequence, and pacId number (information I chose as the primary key). This was the only information I found necessary for the Blast MySQL table once the other data present in the BLAST output, such as locus name and transcript name, was already present in the “Ath11.txt” file and, therefore, would go into the connecting table. The “parseblp” script also included an if statement that checked whether the outputs from the list index 2 had the “title =” pattern and stripped that specific part of the output in order to obtain a clean, data-only file. For this to be done, however, I first had to divide list index 2 into lists based on spaces.

Lastly, I built yet another nano script called “parseid” that looped through the Ath11.txt file and outputted only the pacId data; the output from this script was pipped into a file called “pac_id.txt” and was done so once this was the information chosen as the primary key to connect the MySQL output tables. Subsequently I logged into MySQL and made three tables: the first one was called pac_id and it stored the “pac_id.txt” output as a VARCHAR variable; The second one was called annotated and it held the entire output from Ath11.txt as TEXT variables plus the pacId information as a VARCHAR in a column called “pac_id”; The third table was called parseblp and it stored the entire output from “a_1” as TEXT variables, as well as the pacId information as a VARCHAR variable, put in a column called “gene_id”.

Having loaded all of the datasets into the respective tables, I created a query that selected the scaffold and length from table “parsedblp”, as well as riceDefline, pfam and GO from table “annotated”; in which the primary key used was the tables’ pacId information (pac_id for “annotated” and gene_id for “parsedblp”). The two tables were left joined in order to still obtain the sequences from “parsedblp” that did not match to any fields in the connecting table “annotated”. The output from this query was then saved to a file I called “annotation.txt” which was sorted with the command “sort unique” in order to eliminate redundant results.