Reilly Becker

3 February 2020

BINF 6211 Genomics

Genome Assembly

Intro

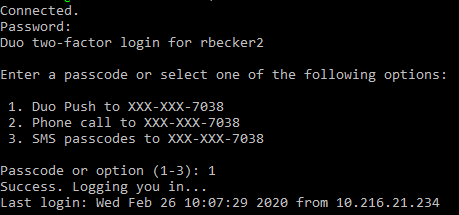
Sequencing genomes is a primary step to isolating diseases, characteristics, or disorders, and improving those conditions. Genomes are made up of strings of nucleotide (A, T, C, G). The assembly mapping software maps segments of nucleotides called contigs. Ideally the user would prefer fewer contigs per genome. This would ensure longer read length and confidence in the data. The first part of this lab was to assemble this data using Ion Torrent to break down a chloroplast genome with SPAdes.py, and use different alterations on the documentation to reap different output results. Providedfor the second part in this lab exercise were 2 sets of paired end Illumina reads. The first was ERR008613, which contained 200bp E. coli fragments, while the ERR022075 contained 600bp E. coli fragments. This data would be assembled using SPAdes.py, in conjunction to the provided Pacbio CCS (500 bp) and CLR (10k bp) refence genomes. These SPAdes.py script will be wrapped in a batch script. Later the resulting contigs would be interpreted by Quast. Quast 2.27.1, can be downloaded via cmd line or manually, from the host site. This lab will be broken down to the most basic step of the assembly process and its interpretation for users unfamiliar with either computer science or biology.

Methods

The files ERR008613sample\_1.fasta, ERR008613sample\_2.fasta, ERR022075sample\_1.fasta, ERR022075sample\_2.fasta, BC30\_BINF650\_Summer2014\_13pm.fasta, NC\_000913\_chloroplast.gff , NC\_007898.gff -r NC\_007898.fasta were uploaded to the UNCC cluster Mamba, from the provided drop box. The file NC\_000913\_chloroplast.fasta was taken from an online GitHub resource [Paudano,2015]. The requirements for this was to first sign up as a root user to mamba and to activate a Duo authenticate account with the university. Both of these could be found with a simple google search, or a search of “mamba” or “duo” on the university’s page. Ubuntu version 1804.2019.521.0 sub system was used as the interface connection to mamba. In order to access mamba enter the syntax seen in the image below.



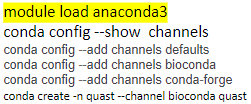
Once the user presses enter, they should be asked for their password. Note when using clusters, it is often required to be on the network. UNCC provides a home use access to that network by connecting to the VPN Cisco. **Only after connecting to this or the school wifi can the user access the mamba cluster.** Depending on how the user set up their Duo account, the user will be prompted, after entering their mamba password, to verify it. In this example the user’s cell phone was linked to the Duo account through the iPhone app. Option 1 was entered for a prompt verification to be sent to the phone.



Once in the Mamba cluster the domain should be changed to the mamba user ([rbecker2@mba-i1 ~]). To transfer files from the local computer to the cluster one must enter the following command from the home directory, not the cluster directory:



File paths can be changed around depending on where one wants to take and put the given files from the local computer to the cluster. The next step was to load the correct modules. This is so that the SPAdes script will run. SPAdes scripts or any script requiring data processing can not run independently in the cluster. This is because the University expects each processing job to be requested, so that the cluster can anticipate computational load. This is what batch scripts are for, which contain parts of code specifying the amount of nodes and time the process will demand for completion. **These commands must be entered before the rest of the lab is approached.**



To write a batch script it is recommended to use Notepad or any text editor. The part in green text, in the image below containing the labeling info. (Line 1): script type id (Line 3): cluster name (Line 4): Name of job to be run (Line 5): the amount of nodes and ppn to be requested (Line 6): the amount of time it will take to run this script (Line 9): allows jobs to be run by user (Line 11): specifies the file pathway of where to place the file in your cluster space (Line 15 &17) running the actual spades command. The rest of the line are optional an are only user friendly to someone unfamiliar with what they ran. The -k numbers in line 17 can be changed around to assemble the contigs differently. In the spades command it was important to use this exact spacing and syntax. The .fasta file starting with BC30\_... is the file that is being assembled. The part with -o is the output name of the finished job. This text file **must** be saved as a .qsub file. This can be done by selecting “all files” and putting .qsub at the end of the file path before saving. Once it has been uploaded to the cluster using the scp command, The user must type “$ qsub <file\_name>.qsub”. This job took about 5 minutes. Job status can be checked on using the command “qstat -l”.



Figure 1:batch script containing SPAdes job request

Users will get a few files back from each job the output file name, the job name with and .o and .e file (ex. MyJob.o42856). These o and e show all information on what may have gone wrong in the script.

The first part of this lab was to show the effects of changing around assembling criteria in spades.

Figure 1: --careful

Figure 2: --only assembler + –careful

Figure 3: --only assembler (without --careful)

Figure 4: --only-error-correction

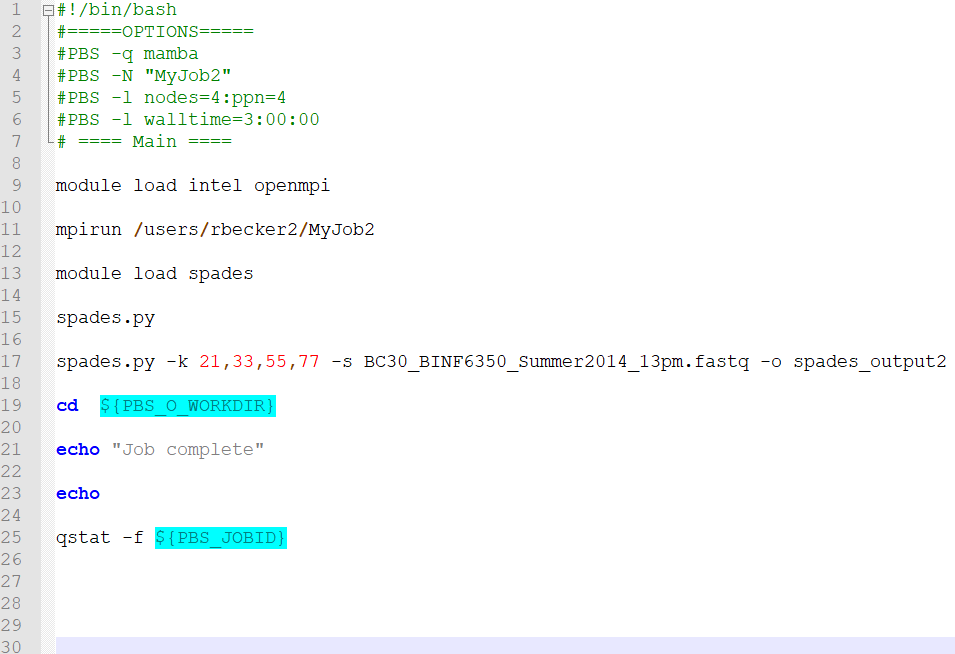


Figure 2



Figure 3

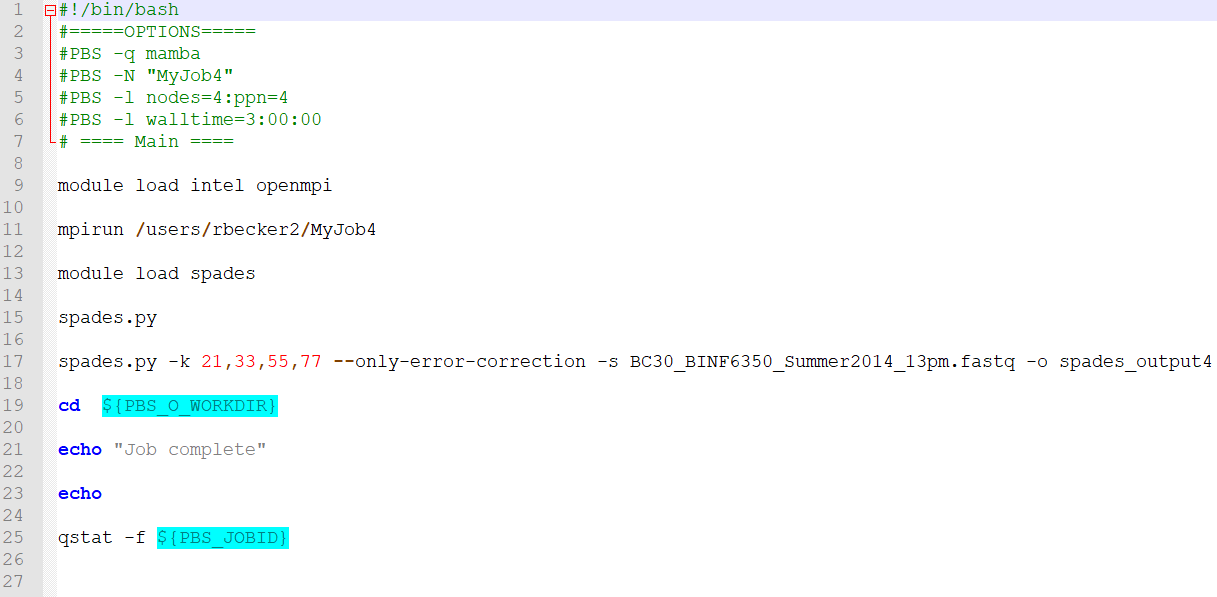


Figure 4

The resulting contig file seen in the output folder, was compared and contrasted. Quast 2.4.0 was downloaded from its home site for windows 10. Each of the spades outputs contained contig reads for each kmer value. These could be accessed by simply using the command “cd <folder>”. Once inside the “finished\_contig.fasta” could be seen. These were extracted and transferred into the same directory that held the Quast program directory manually using WinSCP tool. All files were put together with the program to make it easy to run the script without having to worry about proper file path calling.

The command to activate the quast.py script, to compare the spades alteration of “careful” and “only-assembly” on the chloroplast genome was run:

$ python3 quast.py -g NC\_000913\_chloroplast.gff -r NC\_000913\_chloroplast.fasta -m 100 final\_contigs\_K21\_careful.fasta final\_contigs\_K21\_only\_assem.fasta -o QUAST\_chloroplast\_ouput

The command to compare to compare the CCS\_2k\_500bp with the CLR\_10k was run using:

$ quast.py -g NC\_007898\_Ecoli.gff -r NC\_007898\_Ecoli.fasta -m 100 final\_contigs\_K21\_CCS.fasta final\_contigs\_K21\_CLR.fasta -o QUAST\_ecoli\_ouput

Both of these create an output file containing a .html report.

**Results**

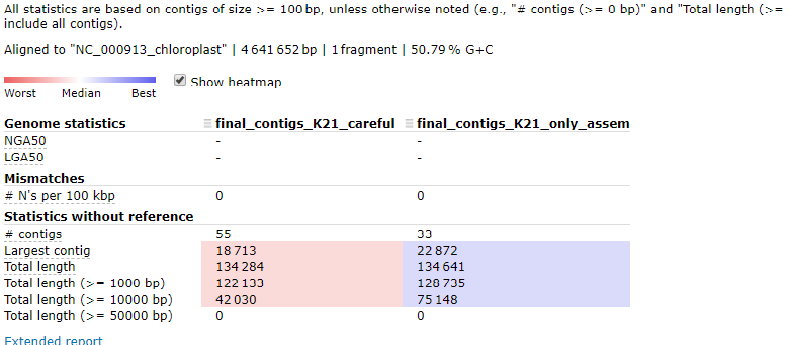


Figure 5:Chloroplast K21 parameter comparison

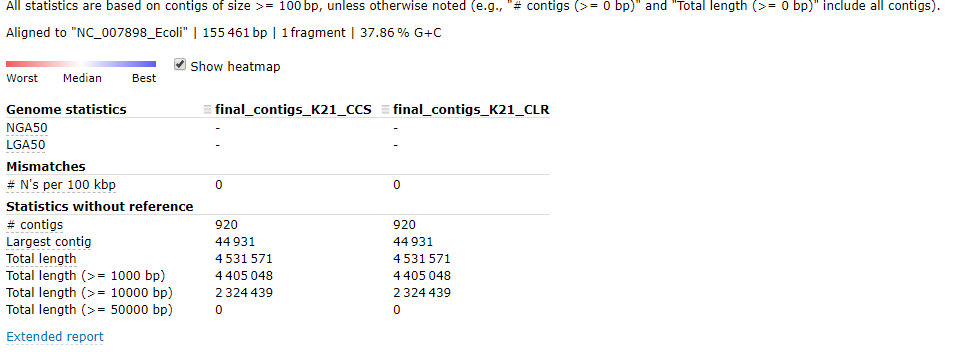


Figure 6: E. coli CCS\_2k\_500bp and CLR\_10k comparison

**Discussion**

The sought goal of any genome assembly is to obtain only a few long contigs. The number 7 is ideal. In Figure 5 the Chloroplast genome contained 55 contigs using “careful” as oppose to 33 with using “only-assemble”. For The E. Coli genomes the output comparison was identical. After confirming with fellow students who received the same kind of out put my conclusion is that the CSS and CLR BioPac files may in fact be similar for K21. For either of the 2 resulting outputs the contig number could have been lowered if one were to first trim each data set. This would get rid of many unpaired illumine reads. In conclusion it is easy to run these assemblies but takes much domain knowledge and judgment to know when to apply the proper parameters to each sequence.

**Citations**

Bankevich, Anton, et al. “SPAdes: A New Genome Assembly Algorithm and Its Applications to Single-Cell Sequencing.” *Journal of Computational Biology*, vol. 19, no. 5, 2012, pp. 455–477., doi:10.1089/cmb.2012.0021.

Paudano. (n.d.). paudano/kescases. Retrieved from https://github.com/paudano/kescases/blob/master/data/ecoli/NC\_000913.fasta.gz