Improving the Draft Assembly of the Horse Genome: Midyear Report

Megan Smedinghoff, smeds@umd.edu

Advisor: James A. Yorke, yorke@umd.edu

December 20, 2007

Abstract

I aim to improve the draft genome of the horse that was produced by Broad Institute in early 2007. My strategy is to begin by producing a second assembly of the horse using the Celera Assembler. After generating a new assembly, I intend to use existing University of Maryland software to reconcile the two assemblies and produce a third that is more accurate than either of the two drafts.

Background

In February 2007, the Broad Institute of MIT announced that it had completed a draft assembly of the horse genome (*Equus Caballus*). The announcement was the culmination of a $15 million project funded by the National Institute of Human Genome Research and the National Institute of Health. The draft genome will allow the equine research community to better understand diseases that affect horses. Additionally, the release of the horse genome has caused some excitement in the human genome research community. There are over 80 known conditions in horses that are analogous to disorders in humans (including arthritis and allergies). Many believe that comparative genomics methods will lead to better understanding of these disorders and therefore better treatments for both animals.

Project Goals

My original proposal was to reassemble the horse genome using the Celera assembler. The resulting assembly would be different from the draft released by Broad Institute since they used a different assembler. I also proposed merging the two assemblies into one hybrid assembly that would be better than either of the individuals. Provided that the process went smoothly, I intended to submit the improved assembly to GenBank so that anyone could use it. The timeline for my project was as follows:

Fall 2007: Produce a Celera assembly of the horse

January 2008: Compare the Celera and the Arachne Assemblies

Spring 2008: Produce a reconciled assembly and write up my results

The Celera Pipeline

Most of my work to date involved becoming familiar with the programs needed to create an assembly. In some cases I used Celera products and in others I used special University of Maryland software. Since the process was somewhat unusual, I will devote some time to describing how large mammals are assembled at Maryland.

The first step to creating an assembly is getting the data from the organism being sequenced. The actual sequencing is a complex and expensive process that need not concern us. All we need to know is that the sequence is represented in overlapping strings of 800-1000 nucleotides (these strings are known as reads). The whole process of assembly is devoted to finding overlapping reads and stringing them together into one long stretch of sequence. The process is complicated due to two problems. First, there are many repeat regions in a genome, and it is often difficult to tell if two reads truly overlap or if they are from different copies of a repeat. Second, the sequence of the reads is not perfect. There are sequencing errors and both ends of a read tend to be of low quality.

The next step in the assembly process is to trim the reads so that we have only the high quality sequence from the middle section of the read. We are able to accomplish this since each nucleotide has a quality score associated with it that represents the probability that the base is correct. We can simply request that every base in our new read be above a certain quality score. There are a number of trimming programs in existence. We prefer to use the University of Maryland trimmer that was written by Mike Roberts.

Once the data has been trimmed, we can start calculating overlaps. Again, we do not use the built in Celera overlapper but instead use the University of Maryland overlapper. UMD Overlapper has proven to be a very good product and frequently yields much better results than the Celera overlapper. UMD Overlapper is an iterative process that begins by calculating all the overlaps. It then uses the list of overlaps to perform error correction on the reads. The newly corrected reads are then subjected to a second round of trimming. Once the reads have been retrimmed, the new overlaps are calculated and outputted. UMD Overlapper outputs two sets of overlaps. The first set is a list of all the overlaps and the second is a list of those it believes to be "reliable".

The last thing we must do before running the Celera assembler is generate a .frg file. This file is easily generated by running a built in Celera program that combines sequence, quality, and trimming information into one big file. Once we have the .frg file, we feed it and the overlaps list to Celera. The assembler usually takes about a week to run on a large mammal and outputs roughly one terabyte of data.

Producing a Celera Assembly for the Horse

The process described above is not necessarily straightforward. One difficulty I faced was that the UMD Overlapper was not designed to run on large genomes. Usually the overlapper can be run in one invocation from the command line. However, since the horse is so large, I had to break up the data into chunks and run pieces of the overlapper line by line. The process ended up being somewhat convoluted, but I will do my best to present a clear picture of what I did.

I began by downloading the horse data from the NCBI website (http://www.ncbi.nlm.nih.gov/). There were 31,240,954 reads. I started by making sure the data made sense and modifying it when it did not. I first determined that the reads were in the same order in the sequence files and the quality files. I also checked to make sure the reads were the same length in both sets of files. After assuring myself that the sequence files and the quality files represented the same data, I checked for duplicate reads. I ended up eliminating 535,739 duplicate reads. Finally, I checked to see that the quality scores corresponded to the sequence. I discovered that some of my reads had base N (representing A, C, G, or T) but that the quality score was positive. I decided that it did not make sense to assign a positive probability to a base that could be any of the four possibilities. I changed all the quality values corresponding to N to be zero.

After looking at the data and fixing some of the obvious problems, I was ready to start the trimming process. Before I began the trimming, though, I eliminated reads of types “PCR”, “EST”, and “FINISHING”. These types of reads were either not consecutive sequence or were sequenced by an unusual method which would confuse the trimmer. After this final elimination of reads, I ran the trimmer with both a 5% accuracy level and a 10% accuracy level. The accuracy level refers to the probability that a nucleotide was reported incorrectly. Thus, in the 5% case, we took the longest possible part of the read where every nucleotide had less than a 5% chance of being incorrect. I ended up using the 5% trim values, which was the more conservative option. One side affect of the trimming process was that some of the reads become too short to be used in the later stages of assembly and had to be eliminated. By the time I had trimmed all my reads at 5% and thrown away the ones that were too short, I was down to 29,318,901 reads and 21,095,993,829 total nucleotides.

After trimming the reads, I was finally ready to run the overlapper. Since I had too much data to run the overlapper all at once, I split the reads into five groups. I then computed the overlaps between each pair of groups and each group with itself. As previously mentioned, the overlapper does a second round of trimming. As a result, more reads became too short and were eliminated. After the completion of the overlapper, I had 29,078,173 reads and 232,162,427 overlaps.

The last step before running the assembler was to make a .frg file. This process was somewhat complicated by the fact that the newest version of Celera requires the back ends of the reads to remain untrimmed. I was able to append all of the previously trimmed ends, though, and the .frg file was created correctly. I started running the assembler on December 12th and am still waiting for it to finish.

Project Status and Future Work

My goal for the first semester of this project was to create an assembly of the horse. I am currently in the final stage of running Celera and will have an assembly as soon as the program terminates. I anticipate that it will be done in the next day or two.

While working on the project, I noticed several places where the assembly pipeline could be improved. The most noticeable place was during the running of the overlapper. Instead of splitting the data into groups and running commands by hand, it would be much better to have a version that could be run on large genomes with only one invocation. I also noticed that there were several portions of the code that could be easily parallelizable.

I have three goals for next semester. The first is to compare my assembly to the Broad assembly and make sure they are similar. Secondly, I want to produce a reconciled assembly. The software to do this already exists, and I do not anticipate that this will require a huge amount of time. Finally, I want to modify the UMD Overlapper so that it will run easily on a large genome and take advantage of parallel processing. I also aim to make the overlapper process more compatible with the Celera pipeline so that people outside of Maryland can easily use the UMD Overlapper to improve their assemblies. I will be testing my new version of the overlapper on bacteria and hope to have a version that will run on large genomes before the end of the spring 2008 semester.

Acknowledgements and References

I would like to thank James Yorke, Aleksey Zimin, and Mike Roberts for advising me on the assembly process.

I also made use of the following sources:

1. Roberts, M, Hunt, BR, Yorke JA, Bolanos R, and Delcher A. *A preprocessor for shotgun assembly of large genomes*. Journal of Computational Biology (2004). **11**(4), 734-52.

2. Interpreting Celera Assembler Output (http://www.cbcb.umd.edu/research/castats.shtml)