# Running Haplotype Identification Software in a Grid Environment

Timothy Beissinger

August 31, 2010

## 1 Background and Problem

Recently, members of my lab have needed to run a software package called FastPHASE (http://stephenslab.uchicago.edu/software.html). FastPHASE is a genetic analysis program designed for haplotype reconstruction and estimation of missing genotypes from population data. In short, it has the ability to identify segments of DNA (haplotypes) that are shared by several individuals in the same or different populations. I work primarily with corn, and there has been an explosion in the amount of genetic data that is available for analysis in the species. With this being the case, it has become nearly impossible, both computationally and statistically, to do analysis at a mutation-by-mutation scale. This is why software such as FastPHASE is important. It has the ability to take a large set of information and condense it into a much smaller form. Unfortunately doing this takes time, which is why grid computing may be of use.

The present data that I would like to analyze contains over two million potential locations of genetic mutation on each of 5000 individual lines of maize. FastPHASE compares every one of these data points to find some sort of pattern. The software has several statistical shortcuts that make the job doable–it is already a much faster approximation of a very slow software package called PHASE–but it is still far from instant. I performed a practice run of fastPHASE using 28 of 5000 individuals and $9 * 10^4$ out of $2 * 10^6$ mutations on a relatively fast Dell workstation. The job took approximately one day to complete. According to the program documentation it should scale linearly when both more mutations and more individuals are considered, leaving the expected total computing time at somewhere around 2,000 days. This amount of time is not acceptable for our purposes, especially since we anticipate the amount of data there is to analyze growing rapidly. Fortunately, the project could be completed much faster using either the local Wisconsin Condor pool or, if necessary, the Open Science Grid.

## 2 Use of Grid Computing

There are several reasons that this may be an ideal project for grid computing, as well as one serious drawback. One convenient feature is that FastPHASE is a small and extremely portable program. It could easily be brought along with jobs to run on remote computers that do not have the program previously installed (I expect that few, if any, will). Further, it is available for several different architectures, so virtually any computer in a pool could run the program. Also, with a maximum runtime of around 2,000 days on a single machine, I expect it not to be overwhelming to try to make it run faster by using several. For instance, with just 100 machines the program may complete in just 20 days, and with 1,000 machines it should take only two. The major drawback of using a grid environment for FastPHASE is that it will be no easy task to break the job into smaller ones. There are 10 natural break points (each of the chromosomes in the corn genome), but beyond these finding ways to break the program down may pose a challenge; the software needs to know about the other individuals being analyzed as well as the other mutations being analyzed in order to identify patterns. In a worst-case scenario the corn genome will have to be split at arbitrary points for which haplotypes that cross them will not be identifiable. However, I propose what may prove to be a better technique.

It may be possible to use a two step method that does the following. First the genome would be arbitrarily split as described above and a set of haplotypes that are not affected by those splits could be identified. In the second step, the beginnings or endings of these haplotypes would be used to define a second set of splits, and using these FastPHASE would be run a second time. Because already identified haplotypic breaks are used to determine the splits of the second run, there would be no chance that a true haplotype is affected by one of the splits utilized. This should lead to a result that is identical to what would be obtained from running the software for the full amount of time on a single node. This task will pose a serious scripting challenge, but it may lead to a very effecient method for quickly identifying haplotypes in parallel.