CS224 Final Project Report

Jeannie Huang
Christina De Cesaris

ABSTRACT

This report describes the procedure for determining haplotype phasing for genotypes with some masked data.

1 INTRODUCTION

In this project, we used the Expectation-Maximization algorithm to determine the most likely haplotype phasings for a given dataset of genotypes. Because sequencing technology is imperfect, it will generate values that are missing or masked at some places in the genome, represented by a *, adding additional complexity. A file with an array size number of SNPs by number of people is given as input, and file with array size number of SNPs by number of people*2 is returned. Both the example 1 data and test data have about 39,000 SNPs and 50 samples, while the example 2 data has about 75,000 SNPs and 50 samples.

2 IMPLEMENTATION

2.1 The broad picture

The natural solution is to use the EM algorithm, an iterative method to find maximum likelihood of parameters. We decided to deal with missing data by simply treating a * as either a 0 or 1 and adding those acceptable haplotypes to the possible compatible haplotype pairings. This gives us an approximate rather than perfect solution by "bloating" the list of possible haplotype phasings for a given genotype.

Due to the high combinatorial complexity of phasing a 39,000+ long genotype that could contain up to 300 masked SNPs and many 1's, we decided to run the EM algorithm on chunks of the data instead. Each usage of the EM algorithm determines the best haplotype phasings for the given genotypes of a fixed SNP window size and running the algorithm for n iterations. Both window size and number of iterations are passed in as parameters. Once the best haplotype phasing for that window is determined, it is appended to the horizontally growing list of haplotype phasings. This gradual construction of full haplotypes occurs in the wrapper EM function.

2.2 Setup for the EM algorithm

The setup for the EM algorithm function works as follows.

For every genotype, a list of compatible haplotypes, ignoring masks, are created recursively. For instance, for the genotype *2201, the list *1101 and *1100 are created. From this haplotype list, haplotype pairings are computed by finding the complement to a given haplotype and recursively generating combinations by turning * into a 0 or 1. The compatible pairs would be (01100, 01101), (01100, 11101), (11100, 01101), (11100, 11101). Each haplotype is recorded in a unique haplotypes dictionary.

Once all the genotypes have been read, each haplotype in the unique haplotypes dictionary will be paired to a frequency value p. Another dictionary maps the haplotype pair to an expected value. The goal of the EM algorithm is to run until the expected value for a haplotype type pair converges and the pair with the largest probability is selected for each genotype.

2.3 The EM algorithm

The EM is an iterative algorithm that alternates between the Expectation Step and the Maximization step. The algorithm runs as follows: A guess is made for the initial haplotype frequencies. In this case, each haplotype is initialized with an observation frequency equal to one over the total number of haplotypes.

$P^0 = \{p^0_{h1}, ...p^0_{hk}\}$

2.3.1 The E-Step. Given the current genotypes,all possible haplotype pairs, and the observation frequencies for each haplotype, the joint probability of observing a unique haplotype and its complement pair is calculated by multiplying together their associated frequencies. The expectation of that pair is then calculated by dividing that pair's joint probability by the sum of all other pairs' joint probabilities conditioned on the current genotype.

$$Pr((h_i, h_j)|g) = \frac{p^t_{h_i} p^t_{h_j}}{\sum_{h,h' \in C(g)} p^t_h p^t_{h'}}$$

2.3.2 The M-Step. Given the expectation of each pair, each haplotype frequency is then updated by summing all expectation values which contain that haplotype and then dividing that value by double the number of genotypes. The process then repeats by going back to the E step.

$$p^{t+1}_h = \frac{E_{pt}[h|g_1,...,g_n]}{2n}$$

3 EXPERIMENTS

The project requirements were to complete the program under 4 hours on a modern single-core machine and produce an accuracy of over 75 percent. We ran the haplotype inference program with varying window size and EM iterations on a

Table 1: Experiment results on a 4-core, 16GB RAM machine

File	SNPs in window	Num EM iterations	Accuracy	Time (min)
Example1	10	10	0.811	30
Example1	10	20	0.810	56
Example1	12	10	0.828	170
Test	10	10	0.829	35
Test	11	15	0.829	182*

*Run on the Hoffman cluster.

personal machine to ensure that we met the requirements. The runtime is expected to be slower on a single core on the Hoffman cluster. We focused on Example 1 rather than Example 2 because Example 1 and test data both have about 39,000 SNPs. The results are shown in Table 1.

4 ANALYSIS AND CONCLUSIONS

Increasing the number of EM iterations from 10 to 20 for Example 1 does not improve accuracy, so we can infer that the algorithm takes less than or equal to 10 iterations to converge. A window size of 12 for Example 1 improves the accuracy slightly, but would likely perform over 240 min on the Hoffman cluster. The middle window size of 11 on the Hoffman cluster curiously performs just as accurately as a window size of 10, but takes significantly longer.

We therefore decided to submit the solution for the test dataset with 10 SNPs for window size and 10 EM iterations, as the most accurate and fastest solution.

5 THANK YOU

Thank you to Dat Duong, Nathan LaPierre, and Professor Eran Halperin for their support.