SAHap: Solving the Weighted Minimum Error Correction Problem With Simulated Annealing
(not quite yet)

Zhaofeng Li

December 2019

1 Abstract

Haplotypes of individuals play key roles in numerous genomic analyses. A haplotype refers to a set of alleles found on the same chromosome, inherited from the same parent. Through the process of haplotype assembly, we align fragments of chromosomes to reveal the complete haplotypes for an individual. In this report, we present SAHap, a haplotype assembly program based on the simulated annealing algorithm that solves the Weighted Minimum Error Correction (wMEC) problem. We show that for smaller numbers of sites (#SNPs < 5000), it performs at the same level of accuracy as competing approaches. Problems and future plans for the implementation are also discussed.

2 Introduction

A haplotype refers to the a of alleles found on the same chromosome, inherited from the same parent. DNA sequencing machines produce lossy end reads, paired fragments of sequences on a chromosome, with occasional errors in the results. End reads may overlap with each other, with the average number of times a site is covered defined as the "coverage." Through the process of haplotype assembly, we align the individual end reads to reveal the complete haplotypes for an individual.

SAHap is a haplotype assembly program written in C++ that solves the Weighted Minimum Error Correction (wMEC) problem based on the simulated annealing algorithm. It does so by minimizing the number of conflicts across all sites in the alignment, resulted from incorrect alignment of reads (reads placed on the wrong chromosome) and sequencer error. Currently SAHap is able to accept input files in the WhatsHap Input Format (WIF), automatically determine a temperature schedule using naive heuristics, and complete the simulated annealing process constrained by a specified run time. For smaller numbers of sites (#SNPs < 5000), it performs at the same level of accuracy as HapCHAT, an existing haplotype assembler. The program currently cannot consistently

reconstruct the haplotypes from inputs with larger numbers of sites, performing reasonably in some while failing others.

Visual SAHap is a Jupyter notebook that enables interactive manipulation of program parameters, as well as visualization of the simulation annealing process. It relies on the xeus-cling Jupyter kernel to execute C++ code interactively, and the matplotlib-cpp library for plotting. Plots in this report are produced in Visual SAHap.

3 Haplotype Assembly And the Minimum Error Correction Problem

The process of haplotype assembly concerns assigning each read from the DNA sequencer to the correct chromosome. Humans are diploids with 2 sets of chromosomes, each coming from a parent. The result of the haplotype assembly process for humans is therefore two sequences. By convention, the two haplotype sequences are often binary strings, as the possible alleles at each locus is known. The most common variant is assigned 0 (reference), with the alternative assigned 1. There may be extremely uncommon variants, which are not considered by such implementations.

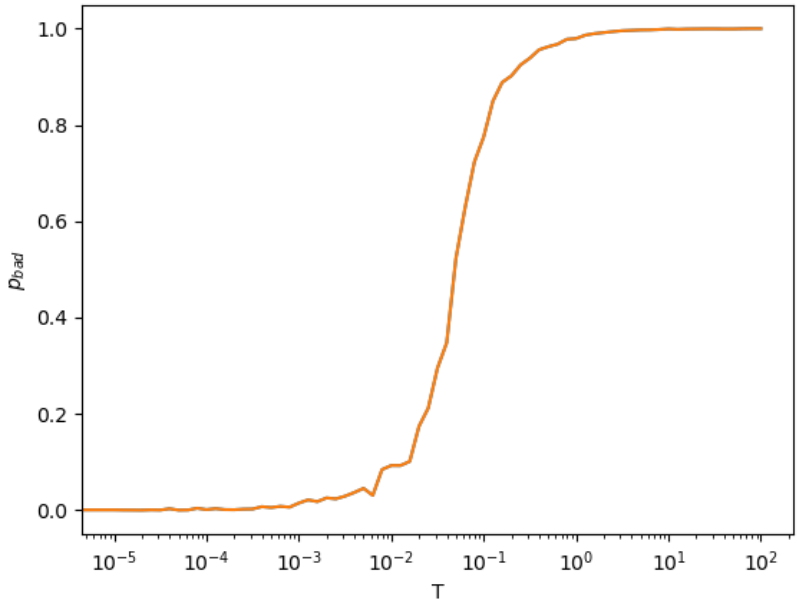
The Minimum Error Correction Problem, proposed by Lippert et al., is NP-hard. When different reads assigned to the same chromosome overlap, their letters on the same site may differ. The MEC value refers to the total number of letters different from the solution. In other words, the MEC value for a particular alignment is the minimum number of letter changes (corrections) one needs to make in order to achieve perfect alignment. Because the haplotype assembly process is often done on binary strings where individual digits may flip, the problem is also called Minimum Letter Flips (MLF).

4 Simulated Annealing

In simulated annealing, the state is perturbed by a random change (move) in each iteration, with each change accepted or rejected according to probabilistic function computed from the amount of loss (energy) the change decreases. The initial state of SAHap is created with each read assigned to a chromosome at random. In each iteration, SAHap randomly selects a read in the system and moves it to another chromosome, recomputing the MEC value in the process. The energy value is computed by dividing current MEC with the maximum possible value.

The system accepts all moves that cause a decrease in the overall energy, and has a higher chance of accepting moves that cause a smaller amount of increase in energy (bad moves). As time passes, the temperature decreases, and so does the chance of accepting bad moves. In a correct implementation of simulated annealing, the probability of accepting a bad across temperatures resembles the

sigmoid function ("hockey stick"). An example of a p_{bad} curve generated by SAHap is as follows:



The temperature schedule can be either specified manually, or determined using simple heuristics. SAHap takes uniform samples in the p_{bad} curve, and selects the endpoints to be the temperatures where a large change ($\geq 20\%$) in p_{bad} occurs by iterating from either direction.

5 Results

For smaller numbers of sites, SAHap performs at the same level of accuracy as HapCHAT, completing the assembly in a reasonable amount of time. The accuracy measure is *HE*, described in the HapCHAT paper, equal to the percentage of sites with the wrong value. The following table shows the average *HE* values across different numbers of SNPs, using the synthetic PacBio RS II datasets provided by the GenHap project:

# SNPs	SAHap HE	HapCHAT HE
500	0.14%	0.09%
1000	0.24%	0.09%
5000	23.58%	3.61%

Results on input files with large number of sites are unreliable, with extremely large variances in accuracy. This inconsistency starts to occur between #SNP$s = 1000$ and 5000, and the underlying reason is still being investigated.

6 Conclusion

There is still much to be done, and we have hit a roadblock regarding accuracy for datasets with a large number of SNPs. In particular, an algorithm for automatically determining the optimal temperature schedule was discussed, but it has not been implemented yet.