A Mathematically Rigorous Algorithm for Long-Range Haplotype Phasing

Samuel Angelo Crisanto
Adviser: Sorin Istrail

Introduction

Humans are diploid organisms: we have two copies of DNA. We inherit one variation of our genes from our father and one from our mother: these two sequences are called haplotypes. A genotype consists of both of an individual's haplotypes.

To sequence DNA, we cut it into small fragments our machines can read, but by doing this we lose information about which strands any differences we observe come from.

We can use the EM algorithm to phase short sequences, which finds the most likely haplotypes that can be assigned to each strand. To do so, however, we must enumerate every possible potential answer, which quickly becomes intractable.

Current long-range techniques rely on the Markov Assumption: only local information is used in order to simplify calculations. My research studied how current methods could be improved by capturing long-range dependencies.

Ambiguity in Genotypes

It is convenient to treat a sequence of DNA as a long string. Human DNA is mostly identical, so let us consider only those places where two haplotypes vary by one base (SNPs). Let us further assume that there are only two variants (the biallelic assumption).

We symbolically represent a haplotype as a string $\{0,1\}^n$. We combine two haplotypes to form a genotype: where they agree, we place that symbol, and where they disagree we use the symbol 2. Given a set of such genotypes, can we infer the haplotypes over the binary alphabet that most likely gave rise to the genotypes?

True Haplotype	Genotype	Possible Haplotypes	
1100101	1202101	1101101	1100101
1001101		1000101	1001101

Avoiding Combinatorial Explosion

We observe that if there are n ambiguous sites in the genotype, there are potentially 2^n explanations. We avoid enumerating all of them by making the assumption that we only need local information to make a good prediction about an ambiguous symbol.

This makes intuitive sense if we phrase it like this: some genes only have a small number of variations. For us to predict which variation you have, we only need to know which gene we are looking at - we don't need information from several genes ago.

It becomes important, then, to recognize the pattern you are in, but we should only remember as much as we need to make a good prediction. We turn to a specific type of data structure that can help us perform this recognition: The APFA.

Acyclic Probabilistic Finite Automata (APFA)

An APFA M is defined as a 7-tuple $(Q, q_0, q_f, \Sigma, \zeta, \tau, \gamma)$ where

- Q is a finite set of states
- $q_0 \in Q$ is the starting state
- $q_f \notin Q$ is the final state
- Σ is a finite alphabet
- $\zeta \notin \Sigma$ is the final symbol
- $\tau : Q \times \{\Sigma \cup \{\zeta\}\} \rightarrow Q \cup \{q_f\}$ is the transition function
- $\gamma : Q \times \{\Sigma \cup \{\zeta\}\} \rightarrow [0,1]$ is the next symbol probability function

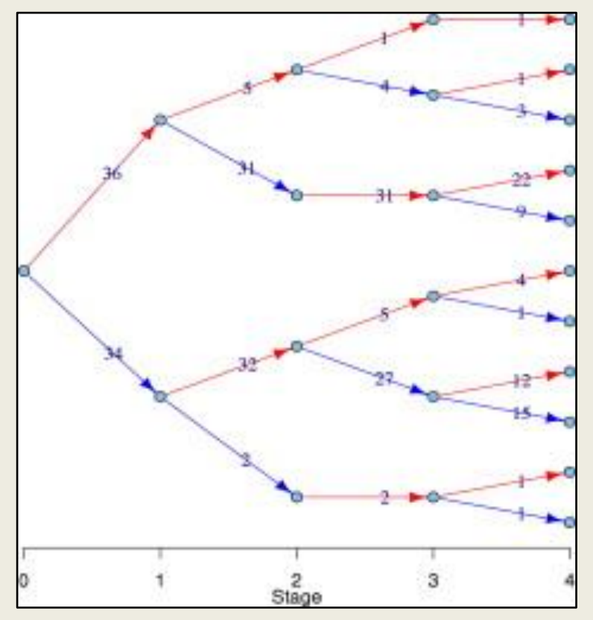
Storing the strings in this way allows us to infer the most likely haplotypes given ambiguous genotypes and a training set. The more strings we train on, the better the estimate.

A Customizable Implementation

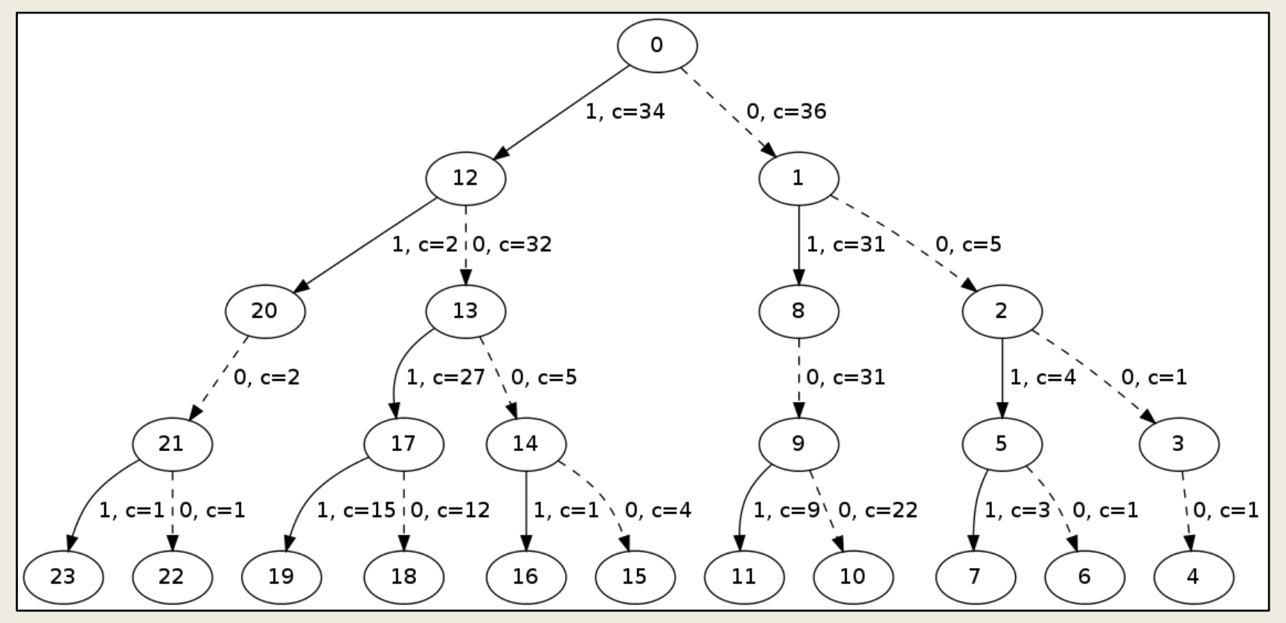
My work has revolved around designing an in-house implementation of this data structure (currently the foundation of the gold-standard BEAGLE phasing software). This allows us to experiment with different parameters and metrics, as well as provides us with the ability to extend and augment its base functionality.

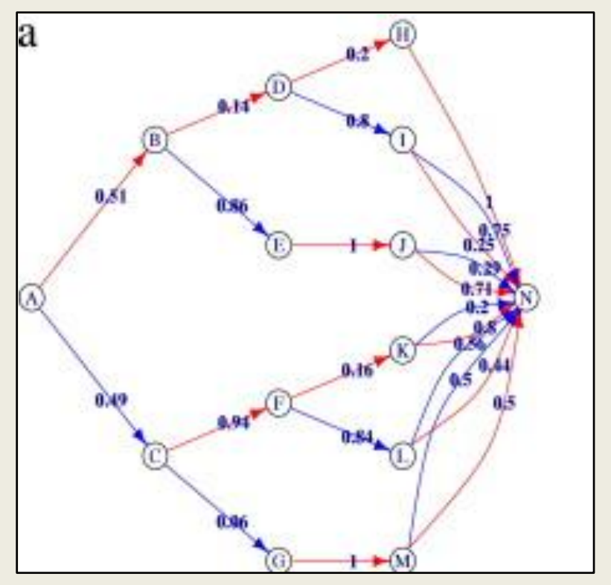
OUTPUT COMPARISON WITH THE INDUSTRY STANDARD

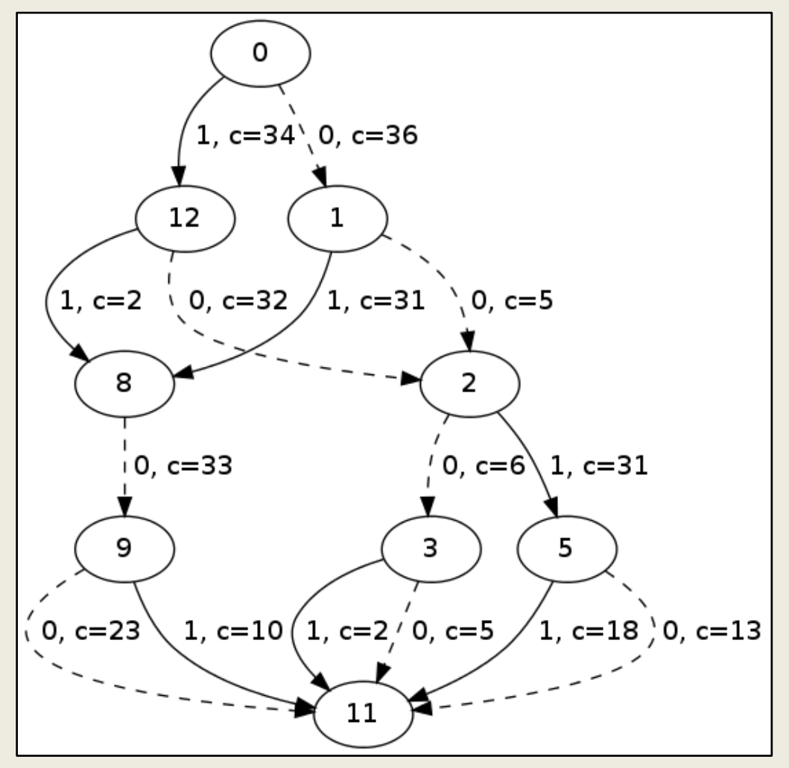
BEAGLE



Custom Implementation







The data in the left column is the output of the BEAGLE algorithm when applied to a short segment of Mildew sequence data.
The output on the right is the result of the implementation for the Istrail Lab.

Figures on the left reproduced from Ankinakatte and Edwards, 2015[1]

Merging Criterion

A primary goal of my research was to examine the merging criteria used to create the final APFA from the initial tree. Whenever two nodes merge back into one node, this constitutes a "loss of memory" for the model: although the sequences are different , future events depend only on this one collapsed node.

It is worrisome, then, that there is no universally agreed-upon "best" metric for determining when nodes are similar enough to merge them. In particular, I was interested in the functional difference between the Ron et. al. merging criterion and the Browning and Browning implementation, which modifies it to account for a higher variance in nodes with lower observed counts.

Ron et. al.

For some constant threshold μ, two nodes *u* and *v* are similar if for every string *s:*

$$| (p_u(s) / p_u) - (p_v(s) / p_v) | < \mu$$

Browning and Browning

The threshold α is allowed to vary as a function of the node counts.

$$| (p_u(s) / p_u) - (p_v(s) / p_v)| < 0.5(n_u^{-1}+ n_v^{-1})^{1/2}$$

Capturing Long-Range Dependencies

While the Markov assumption is especially convenient for this sort of data, there are long-range dependencies that are not being leveraged in order to make more accurate decisions. The way DNA folds and interacts, sequences that are very far apart can interact, which indicates that knowledge of one of those far away sequences will inform our phasing of the other. The Markov Assumption ignores these sorts of correlations.

It is therefore worthwhile to examining ways to use information from biology to enhance the algorithm – sequence data from long reads and the powerful haplotype assembly software HapCompass to help inform decisions that are currently being made purely statistically. This research is ongoing.

References

1. Smitha Ankinakatte, David Edwards, Modelling discrete longitudinal data using acyclic probabilistic finite automata, Computational Statistics & Data Analysis, Volume 88, August 2015, Pages 40-52, ISSN 0167-9473, http://dx.doi.org/10.1016/j.csda.2015.02.009. (http://www.sciencedirect.com/science/article/pii/S016794731500050X) Keywords: Context-specific graphical model; Acyclic probabilistic finite automata; State merging; Discrete longitudinal data
2. Dana Ron, Yoram Singer, Naftali Tishby, On the Learnability and Usage of Acyclic Probabilistic Finite Automata, Journal of Computer and System Sciences, Volume 56, Issue 2, April 1998, Pages 133-152, ISSN 0022-0000, http://dx.doi.org/10.1006/jcss.1997.1555. (http://www.sciencedirect.com/science/article/pii/S0022000097915555)
3. D. Ron, Y. Singer, N. Tishby, Learning probabilistic automata with variable memory length, Proceedings of the Seventh Annual Workshop on Computational Learning Theory (1994)
4. Sharon R. Browning, Multilocus Association Mapping Using Variable-Length Markov Chains, The American Journal of Human Genetics, Volume 78, Issue 6, June 2006, Pages 903-913, ISSN 0002-9297, http://dx.doi.org/10.1086/503876. (http://www.sciencedirect.com/science/article/pii/S0002929707639135)
5. Brian L. Browning, Sharon R. Browning, A Unified Approach to Genotype Imputation and Haplotype-Phase Inference for Large Data Sets of Trios and Unrelated Individuals, The American Journal of Human Genetics, Volume 84, Issue 2, 13 February 2009, Pages 210-223, ISSN 0002-9297, http://dx.doi.org/10.1016/j.ajhg.2009.01.005. (http://www.sciencedirect.com/science/article/pii/S0002929709000123)
6. Sharon R. Browning, Brian L. Browning, Rapid and Accurate Haplotype Phasing and Missing-Data Inference for Whole-Genome Association Studies By Use of Localized Haplotype Clustering, The American Journal of Human Genetics, Volume 81, Issue 5, November 2007, Pages 1084-1097, ISSN 0002-9297, http://dx.doi.org/10.1086/521987. (http://www.sciencedirect.com/science/article/pii/S0002929707638828)