מכון ויצמן למדע
WEIZMANN INSTITUTE OF SCIENCE

Thesis for the degree
Master of Science

חבור לשם קבלת התואר
מוסמך למדעים

By
Tamar Barzuza

מאת
תמר ברזוזה

הסקה ותיוג של הפלוטיפים בעלי פילוגנזה מושלמת

Computational Resolution and Tagging of Perfect Phylogeny Haplotypes

Advisors
Prof. Ron Shamir
Prof. Jacques S. Beckmann

מנחים
פרופ' רון שמיר
פרופ' ז'ק בקמן

February 2004

שבט ה'תשס"ד

Submitted to the Scientific Council of the
Weizmann Institute of Science
Rehovot, Israel

מוגש למועצה המדעית של
מכון ויצמן למדע
רחובות, ישראל

Acknowledgments

I wish to thank all those people who taught me, listened to me, accompanied me, inspired me and distracted me during these two years of Msc.

Thank you Itsik Pe'er
Itsik has guided me through this work, being the ultimate guide one can possibly hope for. He has the passion of 10 men, the creativity of 100 men, and the brilliance of 1000 men; but the goodness and kindness that only few men have, and I am proud to call him my teacher and my friend. A great deal of what I have learned during this thesis, Itsik was the one to teach me.

Thank you Ron Shamir
Ron is the Dr. Dolittle of sciences; speaks fluently Biology, Computer Science, Mathematics and Statistics, and I hear that he can even speak Physics. On top of that, he is a generous man with a strong drive to teach, and will be happy to share his knowledge and understanding in any occasion. Being Ron's student was therefore a fortunate privilege. He has been supportive, patient and reassuring.

Thank you Jacqui Beckmann
Jacqui has Genetics running through his veins, and I don't mean that in the sense that we all do. He gave the Biology spirit to this work; constantly asking the right questions and checking the right angles. He has been passionate, creative, encouraging and trusting.

Thank you Ron's group and Jacqui's group: Nili, Iris, Yudit, Daniela, Clara, Tzvika, Noga, Adi, Amos, Irit, Gadi, Chaim and Israel. Thank you for your teaching and for your listening. You were all very kind, and it was a pleasure spending this time with you. Special thanks to Orna who is my computer guru and my friend.

Thank you to the friends who accompanied me through my Msc: Michal, Libi and Lena. It was inspiring and fun.

Thank you to the friends who distracted me from my Msc (lists may overlap): Michal, Libi, Lena, Shiri, Tzvika, Tal, Orna, Noga, Adi, Sally, Bari and Lili. It was pure fun.

Thank you ima Dalya and aba Itsik for all the help and support. Thank you to the rest of my family, especially Michal, Inbar, Efrat, Edna, Mazal, Menash and Ezra, for constant encouragement.

Thank you my beloved Harel, for making everything better.

Table of Contents

Abstract

The perfect phylogeny model for haplotype evolution has been successfully applied to haplotype resolution from genotype data. In this study we explore the application of the perfect phylogeny model to other problems in the design and analysis of genetic studies. We consider a novel type of data, xor-genotypes, which distinguish heterozygote from homozygote sites without identifying the homozygote alleles. We show how to resolve xor-genotypes under a perfect phylogeny model, and study the degrees of freedom in such resolutions. Interestingly, given xor-genotypes that produce a single possible resolution, we show that the full genotypes of at most three individuals suffice in order to determine all haplotypes across the phylogeny. Our experiments with xor-genotyping data indicate that the approach requires a number of individuals only slightly larger than full genotyping, at potentially reduced typing costs.

We also consider selection of minimum-cost sets of tag SNPs, i.e., polymorphisms whose alleles suffice to recover the haplotype diversity. We show that this problem lends itself to divide-and-conquer linear-time solution. Finally, we study genotype tags, i.e., genotype calls that suffice to recover those of all other SNPs. Since most genetic studies are genotype-based, such tags are more relevant in such studies than the haplotype tags. We show that under the perfect phylogeny model a subset of SNPs is a set of haplotype tags, if and only if it also tags the genotypes.

Chapter 1. Introduction and Summary

In this chapter we briefly introduce the biological background of the problems addressed here, and outline the problems. Fuller details and exact formulations are deferred to the next chapter. We also summarize the thesis results.

1.1. Introduction

1.1.1. Basic concepts

Genetic information in nature is usually stored as a linear sequence, written in a molecular DNA alphabet of four letters (nucleotides), A, C, G and T. Higher organisms are *diploid*, i.e., have two near-identical copies of their genetic material arranged in paired molecules called *chromosomes*, one originating from each parent. Such chromosomes are *homologous*, that is, contain essentially the same genes or altered variants thereof. Differences between variants comprise mainly of *Single Nucleotide Polymorphisms* (SNPs), i.e., sequence sites where one of two letters may appear (Sachidanandam et al., 2001). These SNPs are numerous and it is estimated that any two homologous human chromosomes sampled at random from the population differ on average once in every thousand letters, accounting thus for a few million such differences along the entire genome. The variants of a SNP are called *alleles*. An individual is said to be *homozygous for a SNP* if both homologous chromosomes bear the same allele for this SNP and *heterozygous* otherwise. The sequence of alleles along a chromosome is called a *haplotype*. At first approximation a chromosome can be considered as a patchwork of haplotypes along its length. A *genotype* along homologous chromosomes lists the conflated (unordered pair of) alleles for each SNP (see Fig 1).

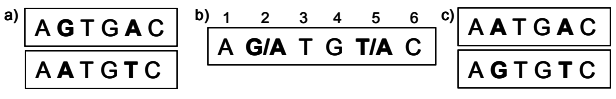


Figure 1 An example of 6 SNPs along two homologous chromosomes of an individual. (a) This individual's haplotypes. (b) This individual's genotype. Here the Xor-genotype (set of heterozygous SNPs) would be {2,5}. (c) Another potential haplotype pair giving rise to the same genotype.

Note that only SNPs are presented here. Every two SNPs can be separated by several hundred monomorphic base pairs.

Both genotype and haplotype data are used in genetic studies. Haplotypes are often more informative (Gabriel et al., 2002). Unfortunately, current experimental methods for haplotype determination are technically complicated and cost prohibitive. In contrast, a variety of current technologies offer practical tools for genotype determination (Kwok, 2001). Given genotype data, however, haplotypes can be inferred computationally, in a process called *resolving*, *phasing* or *haplotyping* (Clark, 1990; Excoffier and Slatkin,

1995; Gusfield, 2002; Bafna et al., 2002; Eskin et al., 2003; Pe'er and Beckmann, 2003). A single genotype may be resolved by different, equally-plausible haplotype pairs (see Fig 1), but the joint inference of a set of genotypes may favor one haplotype pair over the others for each individual. Such inference is usually based on a model for the data. Informally, most models rely on the observed phenomenon that over relatively short genomic regions, different human genotypes tend to share the same small set of haplotypes (Patil et al., 2001; Daly et al., 2001).

1.1.2. The Perfect Phylogeny Model

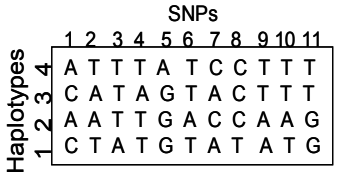
During sexual reproduction, only one homologous copy of each chromosome is transmitted to the offspring. Moreover, that copy has alternating segments from the two homologous chromosomes of the parent - the grandpaternal and grandmaternal copies, due to a segmental exchange process called (*meiotic*) *recombination*. Studies have shown that recombination occurs mainly in narrow regions called *hotspots*. The genomic segments between hotspots are called *blocks*. Such blocks show essentially no recombination (Jeffreys et al., 2001) and their haplotype diversity is limited (Patil et al., 2001; Daly et al., 2001). Within blocks, haplotypes evolve by mutations, i.e., replacement of one nucleotide by another at particular sites (other mutation types are not discussed here). Since mutations are relatively rare (Nachman and Crowell, 2000), it is often assumed, as we do here, that at most one mutation occurs in each site. The *perfect phylogeny model* for haplotype block evolution assumes that all haplotypes in the population descended from a common ancestor, with no recombination and no recurrent mutation events.

The *Perfect Phylogeny Haplotyping* problem (PPH) seeks to infer haplotypes that satisfy the perfect phylogeny model (we defer formal definitions to chapter 2). PPH was first introduced by Gusfield (2002), who presented an almost linear solution by reducing PPH to the classical *Graph Realization* problem. Simpler, direct solutions were later given (Bafna et al., 2002; Eskin et al., 2003), which take $O(nm^2)$ for n haplotypes and m SNPs.

1.1.3. Informative SNPs

Many medical genetics studies first determine the haplotypes for a set of individuals and then use these results to efficiently type a larger population. Having identified the restricted set of possible haplotypes for a region, the identity of a subset of the SNPs in the region may suffice to determine the complete haplotype of an individual. Such SNPs are called *tag SNPs*, and typing them alone would lose no information on the haplotypes. More generally, given a subset S of *interesting SNPs,* an *informative SNP set* is a subset disjoint from S that captures all the information on S (see Fig 2). Hence, the identification of few informative SNPs may lead to substantial saving in typing costs. For this reason, the computational problems of finding a minimal tag (or informative) set have been studied (Patil et al., 2001; Bafna et al., 2003; Zhang et al. 2002).

Figure 2 Tag SNPs and informative SNPs. The set {1, 2} is a tag SNP set. If {9,10,11} is the interesting SNP set, then {1,2} and {6,8} are both informative SNPs sets but {4,5} and {2,3} are not. Notice that the same genotype **A/C T/A** is obtained for the tag SNP set {1,2} from the two pairs of haplotypes {1,2} and {3,4}.



Finding the minimum set of tag SNPs within an unconstrained block is NP-hard (Garey and Johnson, 1979). When the perfect phylogeny model is assumed, in the special case of a single interesting SNP, a minimal set of informative SNPs was shown to be detectable in $O(nm)$ time, for n haplotypes and m SNPs (Bafna et al., 2003).

1.2. Summary of thesis results

We study in this thesis several problems arising under the perfect phylogeny model during genetic analysis of a region, along the process from haplotype determination toward their utilization in a genetic study. Our analysis focuses on a single block.

Experimental methods such as DHPLC (Xiao and Oefner, 2001) can determine whether an individual is homozygous or heterozygous for each SNP, but cannot distinguish between the two homozygous states. Typing SNPs in such manner will provide, for each individual, a list of the heterozygous sites, which we refer to as the individual's *xor-genotype*. Xor-genotypes are less informative than regular ("full") genotypes; but their generation may be less costly. Therefore, it is of interest to examine if it is possible to infer the haplotypes based primarily on xor-genotypes instead of full genotypes.

In chapter 3 we introduce the *Xor Perfect Phylogeny Haplotyping* problem (XPPH), study the limitations of using only xor-genotypes, and the additional genotype information required. Section 3.1 presents an efficient solution to XPPH based on the graph realization problem (Bixby and Wagner, 1988).

We implemented our solution and evaluated the XPPH strategy in chapter 4. Our tests show that the method compares favorably with standard genotyping.

Chapter 5 studies informative SNPs under the perfect phylogeny model. We generalize the minimum informative set (and tag set) problems by introducing a cost function for SNPs, and seek minimum cost sets. The cost is motivated by the facts that typing some SNPs may be technically harder (e.g., those in repetitive or high GC regions), and that some SNPs are more attractive for direct typing (e.g., protein-coding SNPs, due to prior assumptions on their functionality). In section 5.2 we find minimal cost informative SNP sets in $O(m)$ time for any number of interesting SNPs, when the perfect phylogeny tree is given. This generalizes the result of (Bafna et al., 2003).

Section 5.3 discusses a practical variant of the tag SNPs set, i.e., the phasing tag SNPs set: As we usually have only genotypic (conflated) information on the SNPs, a practical goal would be to find a set of SNPs that give the same information as tag SNPs, but instead of knowing their haplotype we only know their genotype. We prove that under the perfect phylogeny model, the set of tag SNPs formed based on genotype data is the same as the set of tag SNPs formed based on haplotype data.

Chapter 2. Preliminaries and Background

This chapter gives a detailed presentation of the problems that were addressed in this thesis. The Biological background that is required to understanding these problems is thoroughly described throughout this section. Each problem description is preceded by the biological motivation to it, and by a short description of the previous work that is related to this problem.

2.1. Basic Concepts in Genotyping

DNA is the component of living organisms that encodes hereditary information. It is organized in linear molecules, called *chromosomes*, along which trait-determining segments, called *genes*, are arranged, interspersed with other DNA segments.

A chromosome is composed of two winding polymer strands that form a double helix. Each strand is a chain of individual units called *bases*, and each base is one of four chemicals - Adenine, Cytosine, Guanine, or Thymine - linked to a phosphate molecule. The bases are abbreviated as A, C, G and T, respectively. The sequence of bases along the DNA molecule encodes the genetic information content of this molecule. The sequence of one of the strands is fully determined by its counterpart, due to a principle called sequence complementarity, and therefore we refer only to one of the strands hereafter.

Most cells of sexually reproducing organisms are *diploid*, i.e., have a duplicate set of genetic material consisting of paired chromosomes, one from each parent. Such paired chromosomes are *homologous*, that is, they are essentially identical, containing the same linear order of gene segments with minor differences in contents. In other words, homologous chromosomes contain the same genes in occasionally different variants. The alternative variants are referred to as the *alleles* of the gene or of a given DNA segment or *locus*.

The human genome contains a little more than 3.2 billion bases. Variation along this sequence comprises primarily of *Single Nucleotide Polymorphisms* (SNPs), i.e., two or more possible bases at a certain position in the sequence. SNPs occur once every several hundred base pairs, and they usually have two alleles only (Sachidanandam, 2001). While independent homologous chromosomes have almost identical sequence content, they differ from each other here and there, mainly at SNP sites. A site with more than one sequence alternative is considered a polymorphism if the alternative allele has an abundance of at least 1% in the general population.

An individual is said to be *homozygous for a SNP* if both homologous chromosomes bear the same allele for this SNP, and *heterozygous* otherwise.
The genetic information given by homologous segments can be presented in two levels of detail:

(a) The information in each SNP is described independently of the other SNPs. The description of a SNP with two possible alleles **A** and **T**, would be either **A**, **T** or **A/T**, for homozygote **A**, homozygote **T**, and heterozygote respectively. An independent description for a collection of SNPs is called a *genotype*.
(b) The sequence of alleles in each chromosome is presented separately. The allele combination along one chromosome is called a *haplotype*.

For an example of the connection between genotypes and haplotypes consider Fig 1.

2.2. Haplotyping

2.2.1. Biological Background

Experimental methods for haplotype determination often require extracting single chromosomes independently or large fragments thereof (Patil et al., 2001). Such approaches are technically complicated and cost prohibitive. Current technologies offer a variety of practical molecular techniques for genotype determination (Kwok, 2001). However, the identity of haplotypes is not readily available when diploid individuals are heterozygous at more than one locus. For example, the genotype in Fig 1b can be comprised of four distinct haplotypes (divided into two distinct pairs), shown in Figs 1a and 1c.

Both genotype and haplotype data are used in a variety of genetic studies. In certain situations, it is much more informative to rely on haplotype data than on genotype data (Gabriel, 2002). Building a haplotype map of the human genome has thus become a central goal of the genetics community. The goal of this project is to develop a genome-wide haplotype map by identifying the haplotype block boundaries and the common haplotypes along the entire human genome. Several institutes and centers have joined an international consortium for this purpose. The haplotype map is expected to be a key resource for finding genes affecting health, disease, and response to drugs and environmental factors, as well as for improving our understanding of the pattern of human genetic variation and evolution (NIH, 2002).

2.2.2. The haplotyping problem

The experimental difficulty in direct measurement of haplotypes motivated the study of computational haplotype inference from genotype data. The process of computationally determining the haplotypes from the genotypes is called *resolving*, *phasing* and also *haplotyping*. It is defined as follows:

Problem 1: The haplotyping problem:
Input: Genotypes of sampled population.
Goal: Infer the haplotypes.

While for a single genotype different haplotype configurations are equally plausible (Fig 1); a collection of genotypes together with additional modeling assumptions may give information that will lead to favoring one inference over the other. One such modeling assumption is that there are few distinct haplotypes in the population, and therefore resolving the genotypes so that many haplotypes will reoccur is desirable. This informal goal of haplotype inference may be formulated in several computational optimization criteria, for example: minimizing the number of distinct haplotypes in the inferred population, or requiring each haplotype to appear in at least α-fraction of the population.

2.2.3. Previous work

The first significant algorithm for inferring haplotypes from diploid population samples was introduced by Clark (1990). This algorithm is a heuristic effort to infer the genotypes using the fewest possible haplotypes. A different class of algorithms is based on the EM (expectation-maximization) paradigm, and seeks maximum-likelihood estimates of haplotypes and their frequencies. The first EM approach for haplotyping was introduced by Excoffier and Slatkin (1995). Both classes of algorithms may take exponential time, and are not guaranteed to solve the problem optimally. In fact, up until recently there has been no efficient solution to the haplotyping problem under any model. More recent approaches and improvements are described in (Gusfield, 2002; Bafna et al., 2002; Eskin et al., 2003).

2.3. Block partitioning

Several extensive studies have been conducted to map the structure of human haplotype diversity (Patil et al., 2001; Daly et al., 2001). These studies showed that the human genome can be divided into blocks of limited haplotype diversity. For example, Daly et al. (2001) examined the haplotypic structure of 500 kilobases on chromosome 5q31, where they found a partition into 11 blocks with 2-4 haplotypes each, in other words each block is represented by a limited number of haplotypes.

Due to the potential utility of block haplotypes for association studies, dividing the chromosome into blocks became an important goal. Some works such as Daly et al. (2001) aim at blocks of low diversity; while others such as Patil et al. (2001) aim at blocks where there is a relatively small subset of the SNPs that gives complete information on the block (namely *tag SNPs* that will be described in detail later). These two studies used heuristic algorithms which were computationally inefficient. Zhang et al. (2002) introduced an exact algorithm for dividing a chromosome into blocks, with the goals of minimizing the number of tag SNPs that gives complete information on all blocks.

2.4. The perfect phylogeny model

2.4.1. Biological background

A cell is called *haploid* if it has one chromosome out of each homologous pair. A special type of cells, *germ cells* (sperm and egg cells and their precursors), have a haploid set of chromosomes. These cells are obtained from diploid cells by two consecutive cell divisions (that follow a single replication), in a process called *meiosis*. Before diploid cells divide, the complete set of chromosomes is replicated, and each of the daughter cells has one identical copy of the duplicate set. During meiosis, homologous segments are exchanged between homologous chromosomes in a process called *recombination*. Thus, although the haploid set of chromosomes has one chromosome for each pair of homologous chromosomes in the diploid cell, this one chromosome is never one of the paired chromosomes, but rather a mosaic thereof. This way, each progeny receives a different combination of genes from that of either parent. Some parts of the genetic material, e.g., mitochondrial genome and most of the Y chromosome, do not undergo recombination (Cummins, 2001).

Studies have shown that recombination events are not uniformly distributed across the genome. Instead, there are recombination hotspots, leading to block structure of the genome where recombination within blocks is relatively rare (Jeffreys, 2001). Indeed, as mentioned above, it has been shown that the human genome can be divided into blocks of limited haplotype diversity (Patil et al., 2001; Daly et al., 2001). Such blocks may indicate that during the recent expansion of the human race from a small founder population, no significant recombination occurred within the block.

Within blocks, haplotypes evolve by mutations. A mutation occurs when the allele of a single nucleotide changes, leading to single nucleotide polymorphisms. The average mutation rate was estimated to be $\sim 2.5 \times 10^{-8}$ mutations per nucleotide site per generation (Nachman and Crowell, 2000); which is 1:1000 per nucleotide site along the lineages between two contemporary chromosomes since their common ancestor (an average of 40,000 generations). Therefore the chance of a recurrent mutation is negligible.

These insights promote *the perfect phylogeny model* for haplotype block evolution. This model suggests that all the haplotypes in the current population have a common ancestor, such that the following two conditions are satisfied:

- *No-Recombination*: Every haplotype in the population has originated from the common ancestor haplotype through a series of site-specific mutations; hence none of the observed haplotypes is a recombinant of two predecessor haplotypes.
- *The Infinite-Site assumption*: During this process of accumulating mutations in time, at most one mutation occurred at each site.

The no-recombination and infinite-site assumptions are only an approximation, and therefore should be used with care when handling real data. First, the limited diversity of haplotypes in a block does not preclude the possibility that the haplotypes of our founding ancestors were recombinants of one another. For the haplotypes of a population to fit the perfect phylogeny model, there must not be any recombinant haplotype in the population (Gusfield, 2000). This means that ever since our common ancestor (one ancestral haplotype) there has been no recombination event within a block. This

requirement is much stricter than the conditions required for limited diversity and does not always hold when it comes to real data, even for groups of very close markers. Both Patil et al. (2001) as well as Daly et al. (2001) report blocks of limited haplotype diversity that do not satisfy the perfect phylogeny model.

As for the infinite site assumption, it has been shown that human mutation rates vary among different sites and different types of mutation. In fact, there are mutation hotspots in the human genome (Nachman and Crowell, 2000) and recurrent mutations are occasionally observed.

2.4.2. Perfect Phylogeny Haplotyping

In principle, SNPs could have between two to four alleles. However, SNPs with three or four alleles are rare enough to be neglected (Sachidanandam, 2001). It is common to refer to SNPs as bi-allelic, arbitrarily denoting one allele by 0 and the other by 1. Hereafter, a haplotype will be denoted by a binary vector, and a genotype will be denoted by an {0,1,2}-vector, with heterozygote denoted by 2.

The perfect phylogeny model yields a tree of ancestry describing the relation between different haplotypes. This tree is the perfect phylogeny for these haplotypes, formally defined as follows:

Definition: Let $H_{n\times m}$ be a binary matrix of n distinct haplotypes over m SNPs. A *perfect phylogeny* for H is a pair (T,f) where $T=(V,E)$ is a tree with $\{1,\ldots,n\}\subseteq V$ and $f:\{1,\ldots,m\}\rightarrow E$ is an assignment of SNPs to edges such that:
(1) every edge of T is labeled at least once (i.e., the mapping f is onto E).
(2) for any two rows k, l, $H_{kj}\neq H_{lj}$ iff the edge $f(j)$ lies on the unique path from node k to node l in T.

Definition: We say that a collection of haplotypes is *coalescent* if there exists a perfect phylogeny for these haplotypes.

Non-recombinant, infinite-site haplotypes have a well known characterization: H has a perfect phylogeny iff there are no two columns of H that include all four possible row combinations 00, 01, 10 and 11. This forbidden submatrix is called a *four-gamete*. The fact that this condition is necessary is obvious. Sufficiency is proved in (Gusfield, 1997).

The problem of phasing genotypes under the perfect phylogeny model can be formulated as follows:

Problem 2: Perfect phylogeny haplotyping (PPH):
Input: Genotypes of sampled population, which is assumed to be generated under the perfect phylogeny model.

Goal: Infer the haplotypes and the perfect phylogeny, or determine that no solution exists[1].

2.4.3. Previous work

The definition of PPH and the first algorithm for the problem were introduced by Gusfield (2002). Gusfield showed that when the haplotypes are coalescent, inferring them could be done in $O(\alpha(m,n)mn)$, where $\alpha(m,n)$ is the inverse Ackerman function, by reducing PPH to a problem known as the graph realization problem (details will follow). Bafna et al. (2002) and Eskin et al. (2003) later solved PPH directly. Their solutions have worse running time of $O(m^2n)$; however, they are simpler to understand, modify and implement than the solutions to the graph realization problem.

2.4.4. The graph realization problem

The connection between *graph realization* and haplotyping was first introduced by Gusfield (2002). The graph realization problem turns out to be significant for this study as well.

Throughout this study we shall denote by (T,f) a tree $T=(V,E)$ with a labeling function $f:\{1,\ldots,m\}\rightarrow E$.

Definition: We say that a pair (T,f) *realizes* $P_i\subseteq\{1,\ldots,m\}$ if P_i is the union of edge labels that constitute a path in T. (T,f) is said to *realize a collection* $P=\{P_1,\ldots,P_n\}$ of subsets if each P_i is realized by (T,f).

The graph realization problem is defined as follows (Bixby and Wagner, 1988):

Problem 3: Graph realization (GR):
Input: A collection $P=\{P_1,\ldots,P_n\}$ of subsets, $P_i\subseteq S$.
Goal: Find a pair (T,f) that realizes P.

For an example see Fig 3.

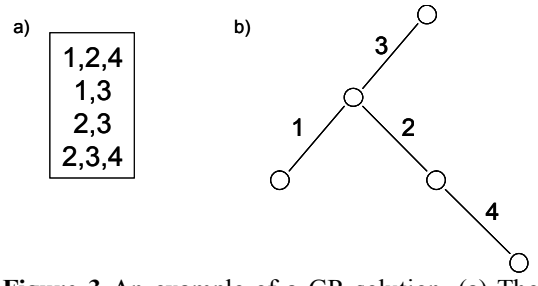


Figure 3 An example of a GR solution. (a) The input to GR. (b) A GR solution to the input in (a). Notice that indeed {1,2,4}, {1,3}, {2,3} and {2,3,4} label paths in (b).

[1] Throughout the study, we shall omit the term "or determine that no solution exists" from problem statements for brevity. This requirement is part of the goal of any algorithm.

Algorithms for the graph realization problem were studied beginning in the 1950's and until the 1980's (Tutte, 1960; Gavril and Tamari, 1983; Bixby and Wagner, 1988). The problem was first defined and solved in matroid theory by Tutte (1960), who gave $O(mn^2)$ time solution. It was then solved in time $O(m^2n)$ by Gavril and Tamari (1983). Later Bixby and Wagner (1988) introduced an $O(\alpha(m,n)mn)$ time solution. Here $\alpha(m,n)$ is the inverse Ackermann function, which for all practical purposes is never bigger than 4. All these algorithms require linear space.

2.5. Our contribution – XOR perfect phylogeny haplotyping

2.5.1. Biological Background

One can consider a hierarchy of genetic information: The ultimate and most complex to derive, is the complete haplotyping, i.e., one determines the haploid phase of each parental chromosome; this in essence is equivalent to having the entire linear sequence of each chromosome. At a lower resolution level one has complete knowledge of an individual's genotype, i.e., both alleles at each SNP are known, but not their parental origin. The next level only differentiates between homozygous and heterozygous states, and thus provides no information as to the exact allele content.

Suppose we cannot distinguish between the two homozygous alleles but can only determine whether an individual is homozygous or heterozygous for each SNP. This is the case in several biological experimental methods such as DHPLC (Xiao and Oefner, 2001). Examination of SNPs in such manner will provide a list of homozygous versus heterozygous sites. We call such information the *XOR-genotype*. Formally:

Definition: The *XOR-genotype* of an individual is the list of SNPs for which that individual is heterozygote[2].

This information is, of course, less than what we know if we have that individual's genotype. In this study we are interested in the following question: what can be inferred from knowing the xor-genotypes of a sampled population? As we shall see, when the perfect phylogeny model is assumed, information of this type is sufficient to infer the haplotypes and their perfect phylogeny.

2.5.2. Problem definition

We present a new variant of the haplotyping problem:

Problem 4: Xor perfect phylogeny haplotyping:

[2] The reason for the name is by analogy to the logic gate 'exclusive or' (abbreviated xor) which satisfies $1 \oplus 1 = 0 \oplus 0 = 0$ but $1 \oplus 0 = 1$, and therefore does not distinguish between the inputs 00 and 11, just like our model.

Input: xor-genotypes from a population, which is assumed to satisfy the perfect phylogeny model.
Goal: Infer the haplotypes and the perfect phylogeny.

Denote by H the actual haplotypes in the population. There are two levels of accuracy in the inference of H.
1. An inference of *H up to bit flipping* is inference of H' such that H' can be transformed to H by changing the roles of 0 and 1 in some of the SNPs of H'.
2. *A concrete inference of H* results in obtaining H itself.

Level 1 of accuracy leads us to the following reformulation of problem 4:

Problem 4': Xor perfect phylogeny haplotyping up to bit flipping (XPPH):
Input: xor-genotypes from a population, which is assumed to satisfy the perfect phylogeny model.
Goal: Infer the haplotypes up to bit flipping and the perfect phylogeny.

Concrete inference, however, requires additional data of full genotypes:

Problem 5: Concrete xor perfect phylogeny haplotyping (concrete XPPH):
Input: (a) xor-genotypes from a population, which is assumed to satisfy the perfect phylogeny model. (b) Several genotypes of the individuals tested in (a).
Goal: Concretely infer the haplotypes and the perfect phylogeny.

For examples of XPPH and concrete XPPH solutions see Fig 4.

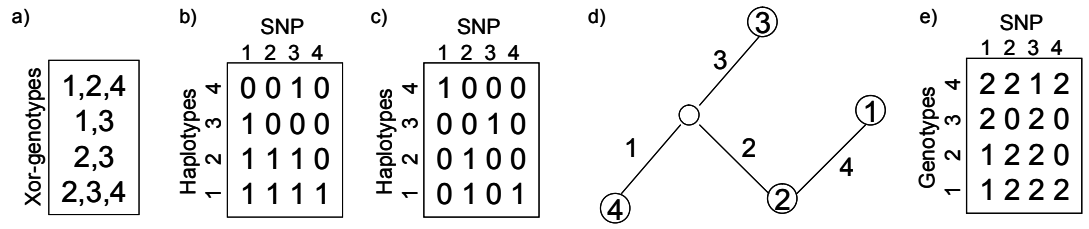


Figure 4 Examples of XPPH and concrete XPPH solutions. (a) The xor-genotypes that are given as input to XPPH and to concrete XPPH. (b) H – A solution to concrete XPPH. H are the (unknown) haplotypes that actually generated X. (c) H' – A solution to XPPH. Note that H can be obtained from H' by flipping SNPs 1 and 3. (d) (T,f) - A perfect phylogeny for H and also for H'. (e) The genotypes that are given as input to concrete XPPH. Note that these genotype can only be generated by H and not by H', hence they may be used to identify H from H'.

2.5.3. Results

In section 3.1 we prove that XPPH is equivalent to the graph realization problem.

In order to implement a solution to XPPH we implemented (in C++) Gavril and Tamari's algorithm for GR (1983). We tested our software on simulated data produced by a standard simulator (Hudson, 2002). We studied the quality of our solution by answering questions such as: How many individuals are required to get a single solution? How

many solutions are there otherwise? What characterizes a large population with multiple solutions? One of our conclusions is that although xor-genotypes carry less information than regular genotypes, they often require only a few more individuals to resolve than complete-information genotypes would.

2.6. Informative SNPs

2.6.1. Biological background

The collection of distinct haplotypes in a certain population, possibly along with their frequencies, is referred to as a *haplotype map* of the population. After such haplotype map is obtained by inferring the haplotypes of sampled individuals, we may wish to determine the haplotypes of other individuals from the same population. The knowledge of the haplotype map can be useful here, because it limits the possibilities of haplotypes for an individual. This is particularly true since, as mentioned earlier, it has been shown that the human genome contains blocks of limited haplotype diversity (Patil et al., 2001; Daly et al., 2001).

Limited diversity of a block indicates that there is redundancy of information in its SNPs. Any subset of the SNPs that captures the complete information of the block is called a *tag SNP set*, i.e., knowledge of the alleles in these SNPs suffices to uniquely determine the haplotype. More generally, an *informative SNP set* is a subset that captures the information of a distinct, previously defined, set of interesting SNPs. For an example see Fig 2. Typing the informative SNPs set may suffice instead of the more laborious typing of all the interesting SNPs, which motivates the computational problem of finding a minimal informative set (Patil et al., 2001; Bafna et al, 2003; Zhang et al, 2002).

2.6.2. Problem definition

Definition: Given a set of haplotypes $H=\{H_1,\ldots,H_n\}$ over a set of SNPs $S=\{s_1,\ldots,s_m\}$, a set $S'\subseteq S$ is *tag SNPs* set if for each $1\leq l\neq k\leq n$ there is $s'\in S'$ for which H_l and H_k have different alleles.

Definition: Let $S''\subseteq S$ be a subset of the SNPs. S'' is called the set of *interesting* SNPs. $S'\subseteq S\backslash S''$ is *informative* on H w.r.t. S'' if for each $1\leq l\neq k\leq n$, whenever there is a SNP $s''\in S''$ for which for which H_l and H_k have different alleles, there is a SNP $s'\in S'$ for which H_l and H_k have different alleles.

The problem of finding the minimal informative SNPs set is defined as follows (Bafna et al., 2003):

Problem: Block minimum informative SNP set (MIS):
Input: (a) A set of coalescent haplotypes $H=\{H_1,\ldots,H_n\}$ over a set of SNPs $S=\{s_1,\ldots,s_m\}$. (b) An interesting set $S''\subseteq S$.
Goal: Find a minimal set of informative SNPs.

2.6.3. Previous work

The problem of finding a minimum set of tag SNPs within a block with no special structure is equivalent to the MINIMUM TEST SET problem, which was proven to be NP-hard in (Garey and Johnson, 1979). However, since most blocks are relatively short (up to 30 SNPs), the problem can be solved to optimality in reasonable time for such blocks.

For the special case where there is one interesting SNP and the haplotypes are coalescent, Bafna et al. (2003) find informative SNPs in $O(mn)$ time. The problem of finding informative SNPs on coalescent haplotypes was posed as open, for more than one interesting SNP (Halldorsson et al., 2003).

2.6.4. Our contribution – Diallelic informative SNPs of minimal cost

We generalize the minimum informative set (and tag set) problems by introducing a cost function for SNPs, and looking for the minimal cost sets. This is motivated by the facts that (a) some SNPs are technically harder to type, e.g., those in repetitive or high GC regions, and (b) some SNPs, e.g. protein-coding SNPs, are more attractive for direct typing, as they are more likely to be causative. This gives rise to the following problem variant:

Problem 6: Minimum-Cost Informative SNPs (MCIS):
Input: (a) A set of coalescent haplotypes $H=\{H_1,\ldots,H_n\}$ over a of SNPs $S=\{s_1,\ldots,s_m\}$. (b) An interesting set $S''\subseteq S$. (c) A cost function $C:S\rightarrow R^+$.
Goal: Find an informative SNPs set of minimal cost.

In Chapter 5 we present an efficient solution for Problem 6. Since this is a generalization of the tag SNPs problem, this also implies an efficient solution to the tag SNPs problem on coalescent haplotypes.

Both the tag SNPs and the informative SNPs problems, as were defined by previous works, suggest that once we know the alleles for these SNPs we can determine the complete haplotype of an individual. However, as we usually have only genotypic (conflated) information on the SNPs, the alleles for this set of SNPs is not always readily available from the set's genotype. In particular phasing the genotypes that are obtained for informative SNPs is not always unique. In Fig 2 for example, the genotype **A/C T/A** for the tag SNPs has two possible resolutions. This motivates the search for a *phasing tag SNPs* set:

Definition: Given a set of haplotypes $H=\{H_1,\ldots,H_n\}$ over a set of SNPs $S=\{s_1,\ldots,s_m\}$, a set $S'\subseteq S$ is *phasing tag SNPs* set if each pair of haplotypes from H gives a distinct genotype on the set S'.

This will be described in detail later; nevertheless, for coalescent haplotypes, we proved that tag SNPs are in fact equivalent to phasing tag SNPs. This result justifies the common use of tag SNPs and informative SNPs problems in genomic studies based on genotypes.

Chapter 3. Xor-Haplotyping

Xor-haplotyping seeks to infer the haplotypes of a set of individuals given their xor-genotypes. Formally:

Problem 4: Xor perfect phylogeny haplotyping:
Input: xor-genotypes from a population, which is assumed to satisfy the perfect phylogeny model.
Goal: Infer the haplotypes and the perfect phylogeny.

This goal can be achieved on two levels of accuracy:
1. Infer the haplotypes up to bit flipping, i.e., infer the haplotypes such that the actual haplotypes can be obtained from the inferred haplotypes replacing the roles of 1 and 0 for some SNPs. This type of inference determines the structure of haplotypes in the population (namely their perfect phylogeny) and may sometime suffice.
2. Concretely infer the haplotypes, i.e., infer the exact haplotype sequences of the individuals. This type of inference gives the complete information on the inferred haplotypes.

Xor-genotype data can only resolve the haplotypes up to bit flipping, as we explain below. Then, in order to concretely infer the haplotypes, we augment the data by up to three genotypes from the population.

Notation: Hereafter, a set of haplotypes is represented by a binary matrix H, where each row is a haplotype vector and each column is the vector of SNP alleles. We denote the allele of haplotype i for SNP j by H_{ij} or by h_j for the haplotype $h=H_i$.

3.1. XPPH

In this section we resolve the haplotypes up to bit flipping.

Recall the definition of xor-genotype from section 2.5.1:
Definition: A *xor-genotype* of a haplotype pair $\{h,h'\}$ is the set of their heterozygote SNPs, i.e., $\{s|h_s \neq h'_s\}$ (see Fig 1). A set of haplotypes H *explains* a set of xor-genotypes X if for each xor-genotype in X there is a pair of haplotypes in H that gives rise to it.

We assume hereafter, w.l.o.g., that xor-genotypes are not empty.

Definition: Two haplotype matrices $H_{n\times m}$ and $H'_{n\times m}$ are *equivalent up to bit flipping* (denoted $H \leftrightarrow H'$) if for any two rows k, l, $H_{k,j} \neq H_{l,j} \Leftrightarrow H'_{k,j} \neq H'_{l,j}$. $H \leftrightarrow H'$ iff one can be obtained from the other by exchanging the roles of 1 and 0 for some columns. Notice that $\leftrightarrow$ is a set-theoretic equivalence relation.

Observation 1: If $H \leftrightarrow H'$ then X can be explained by H iff it can be explained by H'.

Observation 1 implies that XPPH can only be solved up to bit flipping based on the data given by X. This leads to the following definition of XPPH up to bit flipping:

Problem 4′: XPPH:
Input: A set $X=\{X_1,\ldots,X_n\}$ of xor-genotypes over SNPs $S=\{s_1,\ldots,s_m\}$, such that $X_1 \cup \ldots \cup X_n = S$.
Goal: Find a haplotype matrix H and a perfect phylogeny (T,f) for H, such that H explains X. (See Fig 4) .

In some cases, however, there may be several alternative sets of haplotypes that explain X and are not $\leftrightarrow$-equivalent. In that case, we will not be able to determine which of those sets is the one that really generated X. Our next goal is to identify when the solution obtained is guaranteed to be the correct one. We will next show that this is guaranteed by the uniqueness of the solution.

Key property: The definition of a perfect phylogeny implies the following: Let (T,f) be a perfect phylogeny for H. If $H_{ij}=0$ then for all k, $H_{kj}=0$ iff nodes i and k are in the same component of $T \backslash f(j)$.

Definition: (T,f) is a *perfect phylogeny for X* if (T,f) is a perfect phylogeny for some H that explains X.

Proposition 1: When X has a unique perfect phylogeny then there is a unique matrix H (up to $\leftrightarrow$-equivalence) that explains it.
Proof:
It suffices to prove that if (T,f) is a perfect phylogeny for H then there is no H' such that (T,f) is a perfect phylogeny for H' and $\neg(H \leftrightarrow H')$. Assume w.l.o.g. that $|H|>2$, otherwise $H \leftrightarrow H'$ is always true assuming each SNP has two alleles in H. Consider some leaf v of T which corresponds to some unknown $h \in H$. Let $s \in S$ be a label of the edge incident on v. h is the only haplotype with one of the alleles of s, while all the other haplotypes have the other allele. Therefore we can identify that allele and h. The key property now uniquely determines all the other haplotypes that correspond to nodes of T, and in particular all H. Consider now the same leaf v in T and determine the corresponding $h' \in H'$. As before, the haplotypes at all the other nodes are uniquely determined. By the key property, the possible differences between H and H' are bit flips of the bits on which h and h' differ. □

Proposition 1 motivates a new formulation of Problem 4':

Problem 4″: XPPH:
Input: The same input as in Problem 4′.
Goal: Find a unique perfect phylogeny (T,f) for X, or determine that there are multiple perfect phylogenies for X.

Proposition 1 implies that a solution to Problem 4″ (if unique) is guaranteed to be a perfect phylogeny for the correct set of haplotypes, i.e., the haplotypes that actually generated X.

We will next discuss the connection between Problem 4″ and the graph realization problem.

Recall the definition of realization from section 2.4.4:
Definition: We say that a pair (T,f) *realizes* $X_i \subseteq S$ if X_i is the union of edge labels that constitute a path in T. (T,f) is said to *realize a collection* X of subsets if each $X_i \in X$ is realized by (T,f). For example: in Fig 4, (T,f) realizes X. (T,f) also realizes the sets $\{2,4\}$ and $\{1,2\}$ which are not part of X.

Lemma 1: (T,f) is a perfect phylogeny for X iff X is realized by (T,f).
Proof:
Assume (T,f) is a perfect phylogeny for X. Let i_1,i_2 indicate the haplotypes of individual i. Then by the key property, the path between i_1 and i_2 in T is labeled by exactly those SNPs that are heterozygous for the pair of haplotypes i_1 and i_2, therefore it is labeled exactly by X_i. For the opposite direction, assume X is realized by (T,f). Obtain a matrix H by labeling an arbitrary node of T with an all-0 haplotype, then applying the key property to infer the remaining haplotypes that correspond to T. Then X_i can be obtained from H by taking the pair of haplotypes in H that correspond to the end nodes of the path labeled by X_i.□

The following formulation for XPPH is finally obtained:

Problem 4‴: XPPH:
Input: The same input as in Problem 4′.
Goal: Find a unique realization (T,f) for X, or determine that there are multiple realizations for X.

Observation 2: Problem 4‴ is reducible to the graph realization problem.

The existing theory for the classical graph realization problem (see introduction) provide efficient algorithms for determining the existence of a graph realization and also the uniqueness of such a solution. Hence, these algorithms can be applied to solve XPPH.

Occasionally however, X has multiple realizations even when it is highly informative. That is the case when there are xor-equivalent SNPs in the data:

Definition: We say that $s_1,s_2 \in S$ are *xor-equivalent w.r.t.* X and write $s_1 \approx^X s_2$ if for all i: $s_1 \in X_i \Leftrightarrow s_2 \in X_i$. Note that this is an equivalence relation.

Fortunately, xor-equivalent SNPs may be redundant. That is the case when they are haplotype-equivalent:

Definition: We say that $s_1,s_2 \in S$ are haplotype-equivalent w.r.t. H and write $s_1 \approx^H s_2$ if for all i,j: $H_{is_1} \neq H_{js_1} \Leftrightarrow H_{is_2} \neq H_{js_2}$ (See Fig 5). Notice that $s_1 \approx^H s_2$ iff (1) s_1 is equivalent to s_2 or (2) s_1 can be obtained from s_2 by changing the roles of 0 and 1. Note that $\approx^H$ is an equivalence relation.

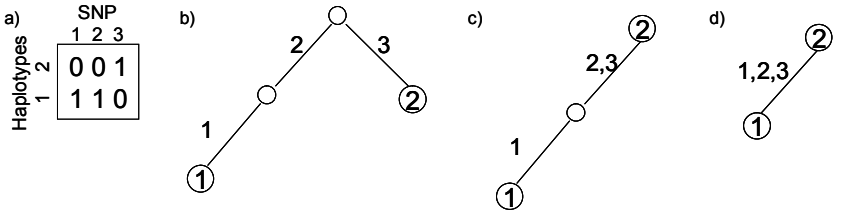


Figure 5 An example of haplotype-equivalent SNPs. (a) H - The haplotypes matrix. SNPs 1, 2 and 3 are haplotype-equivalent. (b-d) Three different perfect phylogenies H.

Observation 3: Haplotype-equivalence implies xor-equivalence but not vice versa. (See Fig 6).

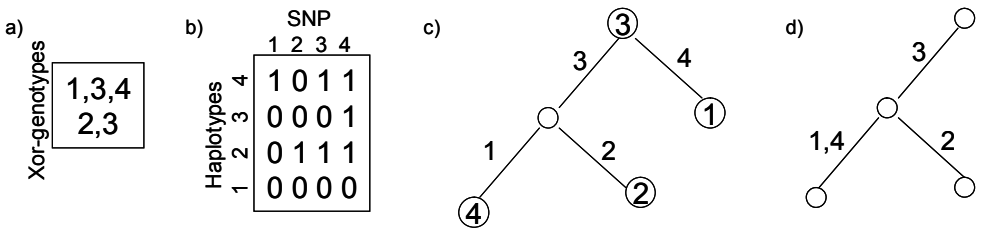


Figure 6 An example of SNPs that are xor-equivalent but not haplotype-equivalent. (a) X – The xor-genotypes. SNPs 1 and 4 are xor-equivalent. (b) H – The haplotypes matrix. There are no haplotype-equivalent SNPs. (c) (T,f) – A realization for X that is a perfect phylogeny for H. (d) (T',f') – Another realization for X that is not a perfect phylogeny for H.

We next show that haplotype-equivalent SNPs are redundant.

Notation: Denote by $S^H \subseteq S$ the set that is obtained by taking one representative from each haplotype-equivalence class. Denote by H^S the haplotype matrix that is obtained by restricting H to S^H.

Observation 4: To obtain a perfect phylogeny (T,f) for H, one can obtain a perfect phylogeny (T,f') for H^S and then for each $s \in S$ take $f(s)=f'(s^H)$ for the $s^H \in S^H$ that is haplotye-equivalent to s.

Observation 4 implies that haplotype-equivalent SNPs are redundant, hence may be merged to label a single edge in (T,f) (See Fig 5). By doing so, we discard the degrees of freedom that comes with having haploype-equivalent SNPs, as proven by Proposition 2:

Proposition 2: If H is coalescent and it contains no haplotype-equivalent SNPs then H has a unique perfect phylogeny.
Proof:

Arbitrarily choose one of the haplotypes, i, and bit flip the SNPs so that i will become an all-0 haplotype. Denote by $(s)_1$ the set of rows for which $s \in S$ has the 1 allele. Then for a pair of SNPs $s_1,s_2 \in S$, w.l.o.g., either $(s_1)_1 \subset (s_2)_1$ or $(s_1)_1 \cap (s_2)_1 = \varnothing$ (Gusfield, 1997). Since there are no SNPs s_1,s_2 such that $(s_1)_1=(s_2)_1$, every edge is labeled by exactly one SNP. Moreover, along the path from i to each node the edge labels correspond to SNPs with increasing number of 1-s (i.e., s_1 is closer to i than s_2 on the path iff $(s_1)_1 \subset (s_2)_1$) (Gusfield, 2002). Therefore, a BFS run that is originated at i is completely defined by the increasing order of SNPs. Hence the tree is unique.□

Identifying haplotype-equivalent SNPs is not trivial when we only have xor-genotype information, and as Observation 3 implies it may not suffice. In other words, in order to merge haploype-equivalent SNPs, we must merge the xor-equivalent SNPs, which by Observation 3 may lead to information loss (See Fig 6).

Definition: Denote by $S^X \subseteq S$ the set that is obtained by taking one representative from each xor-equivalence class. Denote by X^S the xor-genotypes that are obtained by restricting X to S^X. X^S is called the *canonic version* of X.

We show next that when the canonic version of X has a unique realization, then there was no information loss in merging xor-equivalent SNPs, since xor-equivalence implies haplotype-equivalence in this particular case.

Theorem 1: Let (T,f') be a realization for X^S. Extent the mapping f' to S by setting $f(s)=f'(s^X)$ for every s that is xor-equivalent to s^X. Then (T,f) is a perfect phylogeny for X

Proof:

Since (T,f') is a realization for X^S then it is a perfect phylogeny for X^S. We will next show that xor-equivalence implies haplotype-equivalence for the data set X of xor-genotypes. Therefore by observation 4, (T,f) is a perfect phylogeny for the correct haplotype matrix that generated X.

Assume to the contrary that SNPs $s_1,s_2 \in S$ are xor-equivalent but not haplotype equivalent. Consider the unique perfect phylogeny (T^S,f^S) of H^S. Since s_1 and s_2 are not haplotype-equivalent they label distinct edges, e_1 and e_2 respectively, in T^S. By definition, the SNPs in $f^{-1}(e_1) \cup f^{-1}(e_2)$ form a xor-equivalence class. Let (T^S_1,f^S_1) be obtained from (T^S,f^S) by contracting e_1 (identifying e_1's nodes), and by taking $f^S_1(s)=e_2$ for $s \in f^{-1}(e_1)$. (T^S_2,f^S_2) is similarly obtained from (T^S,f^S) by contracting e_2. Since $f^{-1}(e_1) \cup f^{-1}(e_2)$ are xor-equivalent, both (T^S_1,f^S_1) and (T^S_2,f^S_2) realize X. It remains to see that $(T^S_1,f^S_1) \neq (T^S_2,f^S_2)$. Assume to the contrary that $(T^S_1,f^S_1)=(T^S_2,f^S_2)$. Fig 7 shows all possible structures of (T^S,f^S). Consider 8a: Since s_1 and s_2 are not haplotype-equivalent then ν must correspond to a haplotype from H. However, each haplotype of H participates in at least one xor-genotype, in contradiction to the assumption that s_1 and s_2 are xor-equivalent. The proof for 8b is similar to 8a, since either υ or ω must correspond to a haplotype from H. Consider 8c: A path from η can go either through e_1 or through e_2 but not both. Since X^S has a unique realization, there must be a xor-genotype that realizes such a path, in contradiction to s_1 and s_2 being xor-equivalent. □

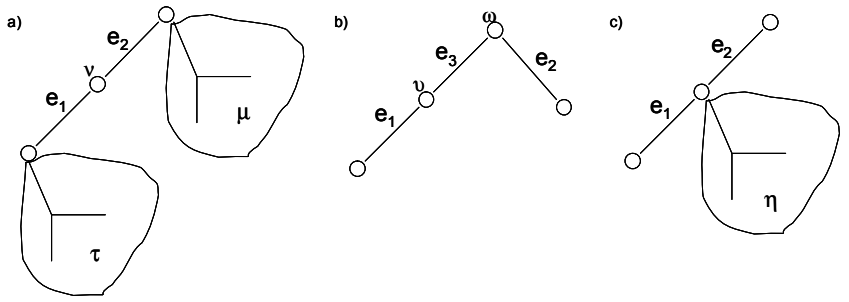


Figure 7 All possible tree structures where merging e_1 to e_2 yields the same result as merging e_2 to e_1. Notice that τ, μ and η may be empty.

Theorem 1 leads to the following algorithm for XPPH:

1. Identify xor-equivalent SNPs and generate a canonic version X^S of X.
2. Solve GR on X^S
    a. If the solution is unique report the corresponding perfect phylogeny of X (Theorem 1).
    b. If the solution is not unique – find all possible GR solutions on the original data X.

The computing of S^X is done as follows: Let $M_{n\times m}$ be the incidence matrix of X, i.e., $M_{ij}=1$ iff $s_j \in X_i$. S^X is computed in $O(nm)$ bitwise operations by radix sort of the columns of M. After sorting, equivalent SNPs are adjacent.

Since haplotype-equivalent SNPs are merged, a unique solution (2a) can often be obtained when the data of xor-genotypes are sufficiently enriched. Sometimes, however, one does not have the resources needed in order to enrich the data. In that case, Lemma 1 implies that the true perfect phylogeny of X is found among the realizations of X (2b). Hence, if possible, one can choose among the realizations of X one that according to other considerations is more likely to be a perfect phylogeny for X. A method for choosing a realization among all possible is described in (Gusfield, 2002).

3.2. Concrete XPPH

As implied by Observation 1, XPPH can only be solved up to bit flipping based on the data given by X. Here a concrete inference is pursued:

Definition: A set of haplotypes H *concretely explains* the sets $X=\{X_1,\ldots,X_n\}$ and $G=\{G_1,\ldots,G_n\}$ of xor-genotypes and genotypes respectively, if for each $\{X_i,G_i\}$ there is a pair of haplotypes in H that gives rise to them.

Problem 5: Concrete XPPH:

Input: (a) A set $X=\{X_1,\ldots,X_n\}$ of xor-genotypes over SNPs $S=\{s_1,\ldots,s_m\}$, such that $X_1\cup\ldots\cup X_n=S$. (b) The genotypes $G_{i_1},\ldots,G_{i_p}$ of individuals $\{i_1,\ldots,i_p\}\subseteq\{1,\ldots,n\}$.

Goal: Find a unique haplotype matrix H and a perfect phylogeny (T,f) for H, such that H explains $X\setminus\{G_{i_1},\ldots,G_{i_p}\}$ and concretely explains $X_{i_1},\ldots,X_{i_p}$ and $G_{i_1},\ldots,G_{i_p}$; or determine that H is not unique.

Problem 5 is solved in two steps. First we solve XPPH for the xor-genotypes of the input and obtain a perfect phylogeny. Then, for some (two or three) specifically selected individuals that were xor-genotyped, we obtain full genotypes. Using these genotypes and the tree obtained in the first step, we infer the full haplotypes of all individuals. By the reasoning of section 3.1 we assume here that there is a single realization to the canonic version of X. If not, the second step may be tried for each realization.

Let (T,f) be the perfect phylogeny for X as was computed by solving XPPH.

3.2.1. A solution to concrete XPPH

Proposition 3: Concrete XPPH can be solved iff $X_{i_1}\cap\ldots\cap X_{i_p}=\phi$.

Proof:

Let j_1,j_2 indicate the haplotypes of individual i_j. If G_{i_j} is homozygous for SNP k, then we know the allele of SNP k for the haplotypes j_1 and j_2, and the key property gives the identity of SNP k for all haplotypes. Consequently, for each SNP $s\in S\setminus X_{i_j}$, its value is resolved by G_{i_j} and the bit-flip degree of freedom is removed. Hence, all degrees of freedom are removed and the concrete XPPH is solved when $X_{i_1}\cap\ldots\cap X_{i_p}=\phi$.□

The proof of Proposition 3 is constructive. In order to solve concrete XPPH we do the following:

(a) Identify j_1 and j_2 for each genotyped individual i_j, and determine their alleles for $s\in S\setminus X_{i_j}$.

(b) Infer the haplotypes by applying the key property to all SNPs.

Step (a) is done by DFS on (T,f) for individuals $\{i_1,\ldots,i_p\}$, in $O(pm)$ time.
In step (b), first label arbitrarily one node of (T,f) by the all-0 haplotype and apply the key property to infer all haplotypes accordingly. Then for each individual i_j and the corresponding nodes j_1 and j_2, determine whether the inference that you obtained for each $s\in S\setminus X_{i_j}$ is correct or opposite. Finally, bit-flip the opposite arbitrary inferences. This is done in $O(m)$ time.
Since in the next section we show that $p\leq 3$, the total time complexity is $O(m)$.

3.2.2. Finding which individuals to genotype

3.2.2.1 Problem definition

As mentioned earlier, concrete XPPH is solved in two steps. First a perfect phylogeny (T,f) for X is found, and then the genotypes of individuals $\{i_1,\ldots,i_p\}$ are applied to (T,f) in order to infer the haplotypes. In light of that, the problem can be broken into three steps as well:

(1) The xor-genotypes are used in order to find (T,f).
(2) The set $\{i_1,\ldots,i_p\}$ is identified based on both X and (T,f), and their genotypes are exposed.
(3) The haplotypes are inferred.

This reasoning motivates the following set-selection problem:

Problem: Minimum tree intersection set (MTI):
Input: A collection of sets $X=\{X_1,\ldots,X_n\}$ and a perfect phylogeny (T,f) for X.
Goal: Find a minimum subset of X whose intersection is empty.

3.2.2.2 Three genotypes suffice

It remains to show how to choose a minimum set $\{i_1,\ldots,i_p\}$ such that $X_{i_1} \cap \ldots \cap X_{i_p} = \phi$.

Without the condition that each X_i is a path in the tree, the problem is equivalent to the NP-hard set-cover problem (Arkin and Hassin, 1992). In contrast, when each X_i is a path in the tree, one gets the interesting result that only three individuals suffice, and they can be efficiently identify as shown next.

Theorem 2: If $\{X_1,\ldots,X_n\}$ are tree paths such that $X_1\cap\ldots\cap X_n=\varnothing$, then there is a minimum tree intersection set of size at most 3.
Proof:
Consider the path X_1, and w.l.o.g. label the SNPs according to their order along that path as $(1,\ldots,k)$. For each i, the set $X_1\cap X_i$ defines an interval in that order. If $X_1\cap X_i=\varnothing$ for some i then $\{X_1,X_i\}$ are a solution. Otherwise all intervals overlap X_1. Denote these intervals by $[l_j,r_j]$ for $j=2,\ldots,n$. Take an interval that ends first and an interval that begins last, i.e., $L=\text{argmin}_j(r_j)$ and $R=\text{argmax}_j(l_j)$. Since $X_1\cap\ldots\cap X_n=\varnothing$ then $[l_2,r_2]\cap\ldots\cap[l_n,r_n]=\varnothing$, hence it follows that $[l_L,r_L]\cap[l_R,r_R]=\varnothing$. We get $(X_1\cap X_L)\cap(X_1\cap X_R)=\varnothing$ and thus $(X_1\cap X_L\cap X_R)=\varnothing$.□

Corollary 1: Let $Y=X_1\cap\ldots\cap X_n$. There are at most three individuals whose genotypes can resolve all the haplotypes on the SNP set $S\backslash Y$. In case $Y\neq\varnothing$, the SNPs in Y can be inferred up to bit flipping.

The proof for theorem 2 is constructive, and implies an $O(n^2m)$ algorithm for MTI. With slight modifications two individuals instead of three can be found more efficiently when possible.

3.2.2.3 A solution to MTI

Notation: For $T=(V,E)$ and $v\in V$ denote by $E(v)\subseteq E$ the set of edges that are incident on v. We assume hereafter, w.l.o.g., that every edge of $E(v)$ participates in a path of some $X_i\in X$.

We next present a linear algorithm for MTI. We begin with a concise outline of the algorithm:

If $j_R - j_L>1$ then an interval that ends at j_L and an interval that begins at j_R correspond to paths that do not intersect. The interesting cases are therefore when $j_R - j_L=1$. We consider the vertex v, which is incident to the two edges that are labeled by j_R and j_L. The interesting case now is when the degree of v is larger than three; otherwise we can easily determine a minimum set of paths that do not intersect. When the degree is larger than three, we prove that there is another edge e incident to v, such that: there is an interval $X_1\cap X_i$ that ends at j_L for which the path X_i goes through e, and there is an interval $X_1\cap X_j$ that begins at j_R for which the path X_j does not go through e; or vice versa. In both cases we get $X_j\cap X_i$.

The algorithm

1) Go over $2\le i\le n$; if $X_1\cap X_i=\emptyset$, return $\{X_1,X_i\}$.
2) Label the SNPs according to their order along X_1 as $(1,\ldots,k)$. Find j_L and j_R as in Theorem 2; take $X^L=\{X_i|\ X_1\cap X_i$ ends at $j_L\}$ and $X^R=\{X_i|\ X_1\cap X_i$ starts at $j_R\}$.
3) If $j_R\le j_L$ return FALSE.
4) If $j_R - j_L>1$ return any choice of $x\in X^L$ and $x'\in X^R$.
5) We remain with $j_R - j_L=1$.
    a) Let $e_L=f(j_L)$ and $e_R=f(j_R)$. Since $j_R>j_L$, $e_L\neq e_R$. Since $j_R - j_L=1$, e_L and e_R are connected by a vertex $v\in V$.
    b) For $j=2,\ldots,n$ take $E_j=\{e|f^{-1}(e)\subseteq X_j\}\cap E(v)$. Notice that $1\le|E_j|\le 2$.
    c) For each $X_i\in X^L\cup X^R$, if $|E_i|=1$ mark X_i as *completed*. If there is $x\in X^R$ that is completed take any $x'\in X^L$ and return $\{x,x'\}$. If there is $x'\in X^L$ that is completed take any $x\in X^R$ and return $\{x,x'\}$.
    d) By now $|E(v)|\ge 3$.
    e) If $|E(v)|=3$ there are no $x\in X^R$ and $x'\in X^L$ such that $x\cap x'=\emptyset$. Take any choice of $x\in X^R$ and $x'\in X^L$ and return $\{x,x',X_1\}$.
    f) If $|E(v)|>3$, take any $e\in E(v)\setminus\{e_R,e_L\}$. There are $x\in X^R$ and $x'\in X^L$ such that either (1) $f^{-1}(e)\subseteq x$ but $f^{-1}(e)\cap x'=\emptyset$ or (2) $f^{-1}(e)\subseteq x'$ but $f^{-1}(e)\cap x=\emptyset$. Find and return $\{x,x'\}$.

Proof of correctness

Most of the proof of correctness was given by the proof of Theorem 2. We complete it by the following.

If $x\in X^R$ is completed then it ends at e_R therefore $x\cap x'=\emptyset$ for any $x'\in X^L$, and vice versa.

If no path is completed then obviously $|E(v)|\ge 3$. If $|E(v)|=3$ then all paths but X_1 go through the edge $E(v)\setminus\{e_R,e_L\}$, therefore there are no two disjoint paths.

Assume $|E(v)|>3$ and take any $e\in E(v)\setminus\{e_R,e_L\}$. Since every edge of $E(v)$ participates in at least one path, $f^1(e)\subseteq x$ for some $x\in X^L\cup X^R$. Assume w.l.o.g. that $x\in X^R$. If there is $x'\in X^L$ such that $f^1(e)\cap x'=\varnothing$ we are done. Otherwise, $f^1(e)\subseteq x'$ for all $x'\in X^L$. If all paths in $X^L\cup X^R$ go through e then $|E(v)|=3$, therefore there exists $x\in X^R$ such that $f^1(e)\cap x=\varnothing$.■

Algorithm complexity

To execute this algorithm we go over the elements of X for a constant number of times. Therefore the time and space complexity are $O(mn)$.

Chapter 4. Implementation and Results

4.1. GREAL

We implemented Gavril and Tamari's (1983) algorithm for Graph Realization. We chose it since it is simpler to implement and modify than the asymptotically fastest algorithm (Bixby and Wagner, 1988). Moreover, as the block size is usually bounded in practice by $m<30$, the difference in the complexity between the best know algorithm (Bixby and Wagner, 1988) which requires $O(mn\alpha(m,n))$ and $O(nm^2)$ complexity of Gavril and Tamari's algorithm is not significant, and the quadratic dependence of the algorithm on m is not a handicap.

We call our software **GREAL**. For an example of a run of GREAL, input and output, see Fig 8. GREAL freely available for academic use and can be downloaded from http://www.cs.tau.ac.il/~rshamir/greal/. GREAL was written in C++, and executables are available for Windows, Linux and SunOS. It was written with great care on time and space performance, and the executable is rather small to download (~400KB for Windows). The code contains about ~4500 lines. We believe the software to be precise and stable after performing massive testing. The running time of GREAL on a standard 1.6GHz P4, 256MB RAM machine is about one second, for a problem of 40 SNPs and 200 individuals[3].

[3] When we embarled on this implementation, no other public implementation was available. Gusfield (2002) discussed such an implementation but it was not available to us. A short while before making GREAL available on the web, another implementation due to Chung and Gusfield has been announced (2003), but we have been unable to operate it. Recently, a new implementation of Bixby and Wagner (1988) has been completed (Ohto and Iwata 2004, private communication).

a)

0 1 1 1 1
1 0 1 1 0
1 1 0 0 0

b)

actual vertices:
0 = 0
1 = 1
2 = 2 3
3 = 4
realization:

-1	1	-1	0	2
1	-1	-1	-1	-1
-1	-1	-1	-1	3
0	-1	-1	-1	-1
2	-1	3	-1	-1

c)

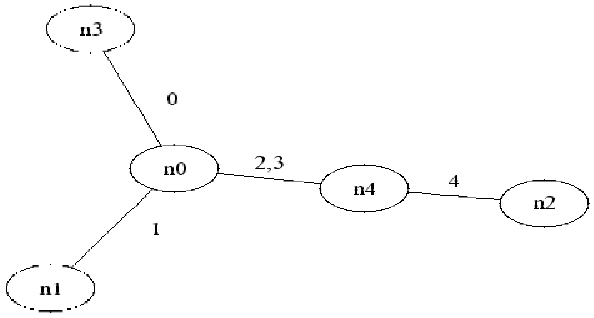


Figure 8 GREAL input and output. (a) An example of an incidence matrix *M*, the input to the GREAL software. *M* shows three xor-genotypes over 5 SNPs. The xor-genotypes (with SNPs numbered from 0 up) are {1,2,3,4}, {0,2,3} and {0,1}. (b) The output (*T*,*f*) of GREAL for the input in (a). The "realization" matrix is the adjacency matrix of $T=(V,E)$. It indicates that $V=\{0,\ldots,4\}$ and that *E* consists of the edges (0,1),(2,4),(0,3),(0,4) with initial labeling of 1,3,0,2 respectively. The "actual vertices" list indicates that the labeling function is the following: f(0)=(0,3), f(1)=(0,1), f(2)= (0,4), f(3)= (0,4) and f(4)= (2,4). (c) The output of GREAL as is presented by *neato*, a software for graph drawing (http://www.research.att.com/sw/tools/graphviz/). In order to obtain this tree drawing GREAL automatically produces output in the format of the input to *neato*.

4.2. Simulated data

We used a standard population genetics simulator due to Hudson (2002) to generate data samples under the perfect phylogeny model. In each run we generated 2400 chromosomes with a prescribed number of SNPs, preserving the default values for all other simulation parameters in the simulator. An important parameter in the experiments was the *minor allele frequency cutoff*, denoted by α: For a given value of α, only SNPs whose less frequent allele among the haplotypes had frequency > α were used. The resulting haplotypes were randomly paired to generate xor-genotypes of individuals.

4.2.1. Results

4.2.1.1. How many individuals are required to get a single solution?

A practical measure of performance is the number of individuals required for a single solution, for a given number of SNPs (the size of a block). We evaluated this measure by adding individuals one by one (starting with 5 individuals) and reapplying GREAL till the solution is unique. The results (Fig 9) show that for $\alpha \geq 0.03$, the number of individuals required to obtain a single solution is roughly constant, irrespective of the number of

SNPs, and is practically bounded by 70. When rare alleles (α=0.01) are present, the behavior is less predictable and the variance is very large.

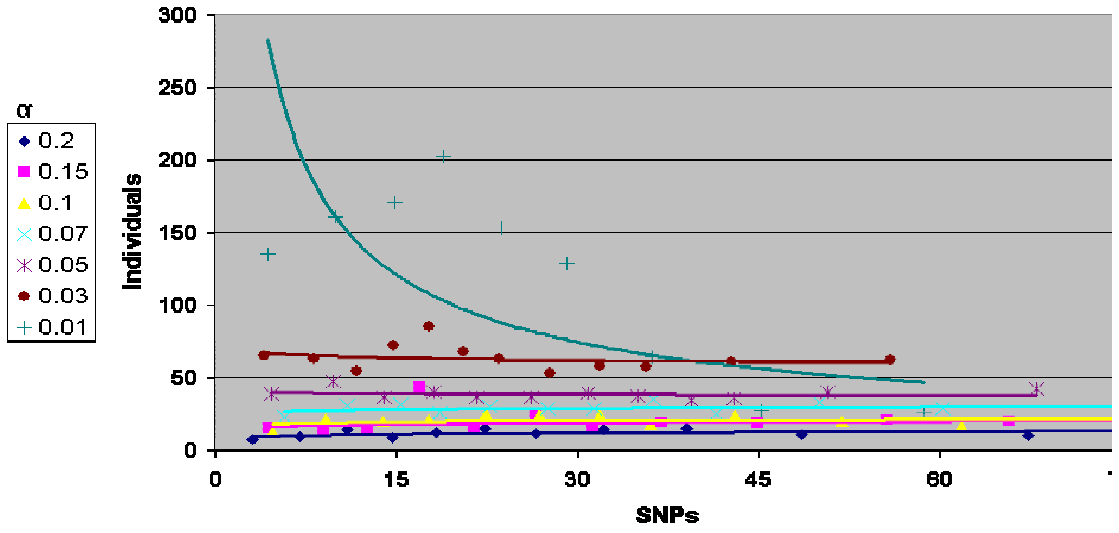


Figure 9 Conditions for solution uniqueness. The plots show the number of xor-genotypes (y-axis) needed for obtaining a single solution for a given number of SNPs (x-axis). Different lines correspond to different thresholds on the minor allele frequency cutoff α.

4.2.1.2. How many xor-equivalence classes are there?

As described in section 3.1, the GR algorithm is applied to the canonic version of the data after computing S^X, and not to the entire S. We tested the number of equivalence classes as a function of the number of SNPs. The number of equivalence classes is a more appropriate measure for the complexity of the problem, as it determines the chance of uniqueness, the number of distinct haplotypes, and the running time of GREAL.

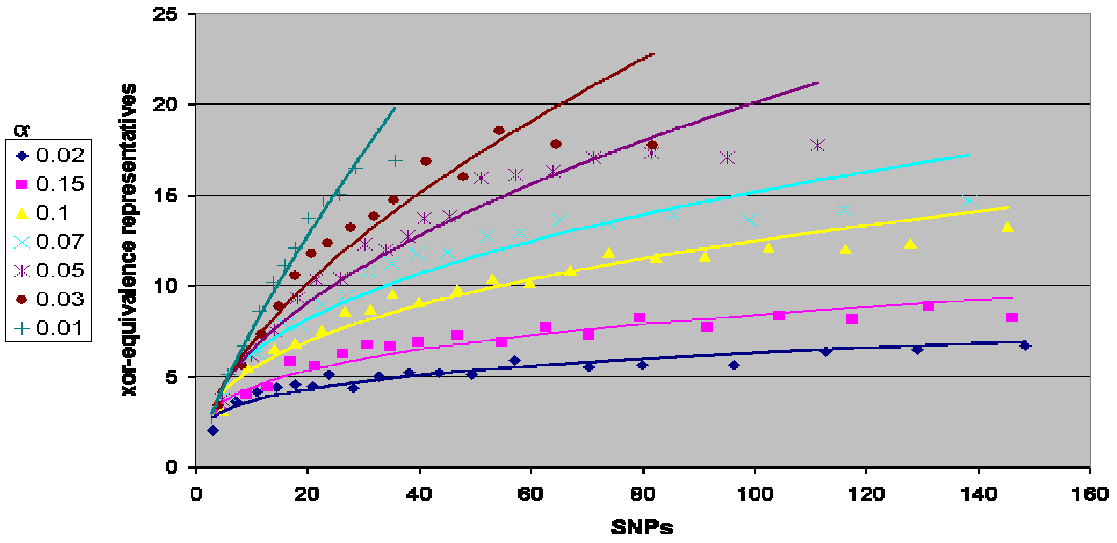


Figure 10 The number of xor-equivalence classes. The plots show the number of xor-equivalence classes (y-axis) obtained for a given number of SNPs (x-axis). Different lines correspond to different thresholds on the minor allele frequency cutoff α.

Fig 10 shows the relation between the size of a block and the number of xor-equivalence classes within it. The results show that α is the key factor in determining the number of classes for a given number of SNPs. The larger α is, the fewest classes there are. For a range of 20-40 SNPs and α=0.03, for example, the number of classes is less than half the number of SNPs.

4.2.1.3. How many individuals does XPPH require compared to PPH?

Since xor-genotypes contain less information than regular (full) genotypes, they may have a potential economic advantage over full genotypes. However, the number of individuals required for obtaining the haplotypes is larger. We compared the number of individuals needed by XPPH and by PPH. Chung and Gusfield (2003) evaluated experimentally the number of individuals required for obtaining a unique solution to PPH (their haplotypes were also generated using Hudson's simulator). We computed the same statistic for XPPH (Fig 11).

For 50 SNPs, 50 xor-genotypes guarantee ~90% chance of uniqueness, and increasing the number of individuals has only a minor effect. Essentially the same results hold for 100 SNPs. In comparison to (Chung and Gusfield, 2003), the chance for a unique XPPH solution with > 50 xor-genotypes is only a few percent lower than for PPH data with the same number of full genotypes.

Consider the following simplified example: suppose that xor-genotyping an individual costs 1, and genotyping an individual costs β. A method for haplotype inference can be the following: Randomly choose 50 individuals to xor-genotype and apply XPPH. If the

solution is unique we are done. Otherwise, randomly choose another 50 individuals to genotype and apply PPH. Then by the probability of a unique solution for 50 individuals (Fig 11), the cost expectation is: $0.9 \cdot 50 + 0.1(50 + 50\beta)$. If however, we decide to use PPH exclusively choosing 50 individuals to genotype, the cost expectation is 50β. Therefore, if $\beta > 1/0.9 \sim 1.1$ then XPPH improves the expected cost.

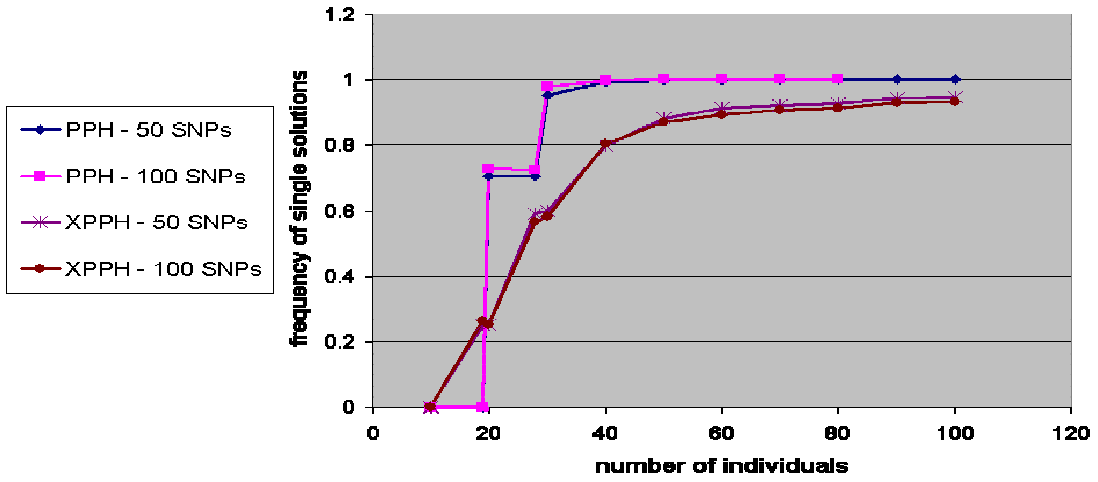


Figure 11 The chance of a unique solution. The plots show the frequency of a unique solution (y-axis) versus the number of individuals tested (x-axis). XPPH statistics are based on 5000 runs for 50 or 100 SNPs after filtering with $\alpha = 0.05$. PPH statistics from (Chung and Gusfield, 2003) are plotted for comparison.

4.2.1.4. How many solutions are there?

We further focused on outlier ambiguous datasets, i.e., those that have multiple solutions despite the examination of many individuals. For such datasets, the number of possible solutions is of much practical interest: If this number is limited, each solution may be tested separately. Indeed, the results (Fig 12) show that for $\alpha \geq 0.03$, when the solution is not unique, there are only a handful of solutions.

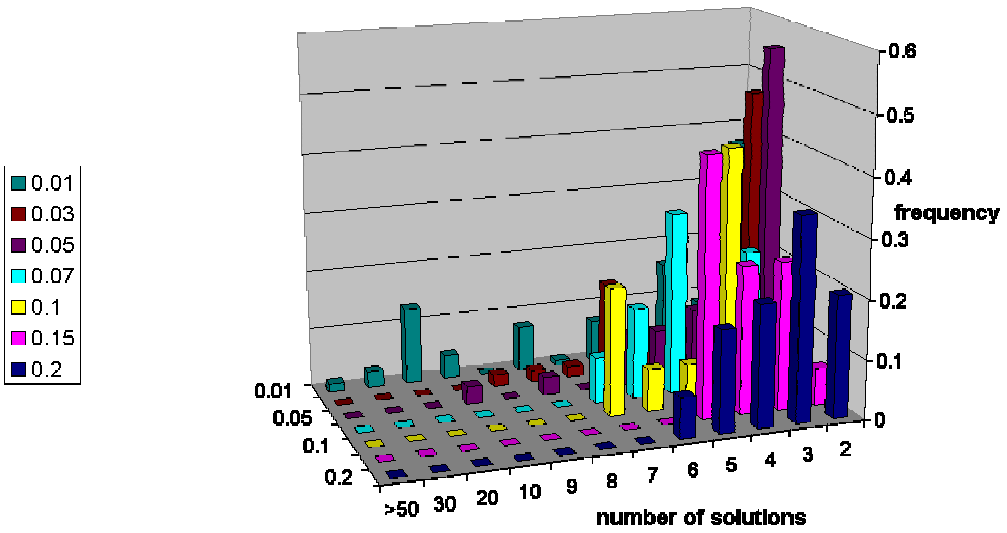


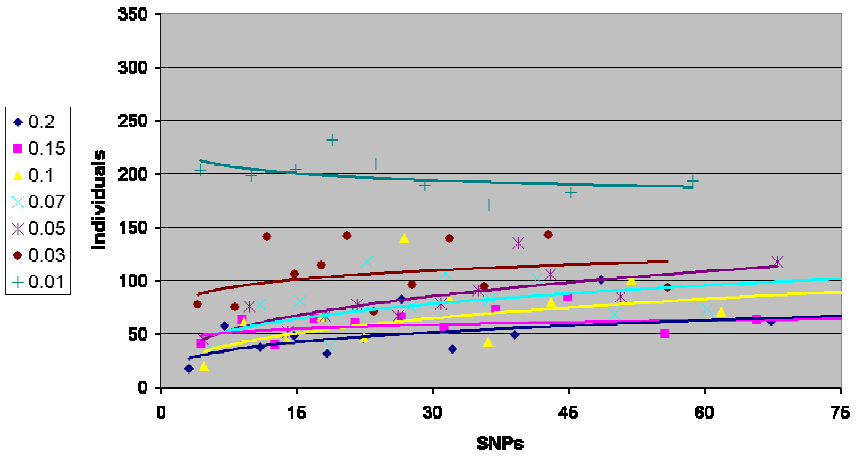
Figure 12 The distribution of the number non-unique solutions in deep coverage studies. We focused here only on instances with multiple solutions. Each bin indicates the frequency (y-axis) of a possible number of solutions (x-axis) for an α-threshold on the minor allele frequency (z-axis). Statistics were collected for varying combinations of the number of SNPs (100-2000) and the number of individuals, where the number of individuals was at least 10 times the number of xor-equivalence classes.

4.2.1.5. Observing the full haplotype diversity in the population

Up until now, we were interested in inferring the haplotypes of a small set of sampled individuals. Specifically, we sampled at random 10-100 individuals out of a simulated population of 1200 individuals, and then attempted to infer their haplotypes. This is indeed the situation in real genetic study: when studying the haplotype variation of a population, one would decide on a threshold on the probability of the rarest haplotype, and then would estimate the number of individuals that should be sampled in order to find (with good probability) all the haplotypes that have larger frequency in the population. Then methods such as regular genotyping or XPPH are used to determine the haplotypes of the sample.

Here we consider a different perspective: How many individuals are required in order to observe all the haplotypes in the population and to obtain a single XPPH solution for them? This is a stricter requirement than the earlier one, but it ensures that the entire diversity of haplotypes is observed.

a)



b)

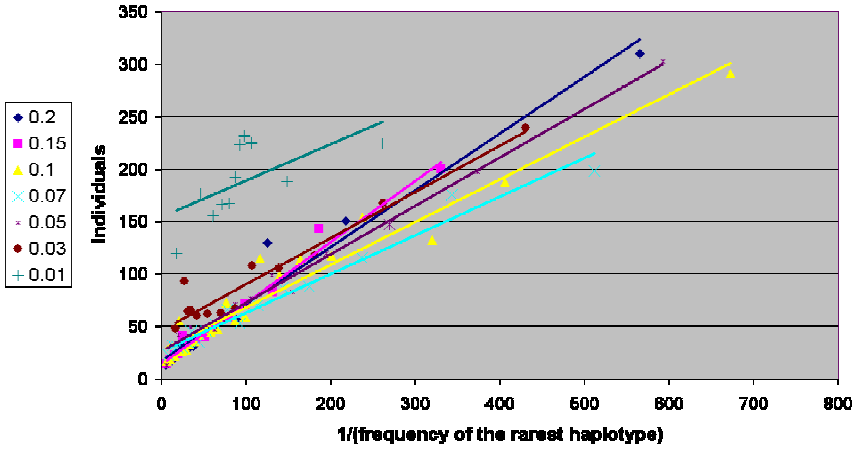


Figure 13 Conditions for solution uniqueness and complete haplotypes sampling. The plots show the number of xor-genotypes (y-axis) needed to observe all the haplotypes in the population and to obtain a single solution (a) for a given number of SNPs (x-axis) and (b) for varying frequencies of the rarest haplotypes (x-axis is the inverse of these frequencies). Different lines correspond to different thresholds on the minor allele frequency cutoff .

We plotted this measure in comparison to the block size, and observed a large variance for it (Fig 13a). Comparing these results to the earlier analysis (Fig 9) we found that the number of individuals required is roughly doubled. In Fig 13b we plotted the number of individuals needed versus the frequency γ of the rarest haplotype in a population of 2400 haplotypes. The plots show markedly lower variance than in Fig 13a, and a linear dependence on $1/\gamma$ (note that γ is the frequency of the complete haplotype and is different from α which is the cutoff for each individual SNP.

Since in these tests we used two criteria for determining the number of individuals needed, a natural question is which of the two criteria is the more demanding. More specifically, which requirement is fulfilled last, when finally the sufficient number of individuals has been reached? We call that requirement “the last straw”.

The results in Fig 14 show that for $\alpha \geq 0.03$ usually a rare haplotype that was missing causes the delay. In contrast, for $\alpha=0.01$, it was the non-uniqueness of the XPPH solution that required more data. Note that for $\alpha=0.01$ the number of xor-genotypes needed is much larger (Fig 9), since many haplotypes are rare. Thus, by the time we have sampled

enough haplotype pairs to have a unique solution, typically all haplotypes have already been observed.

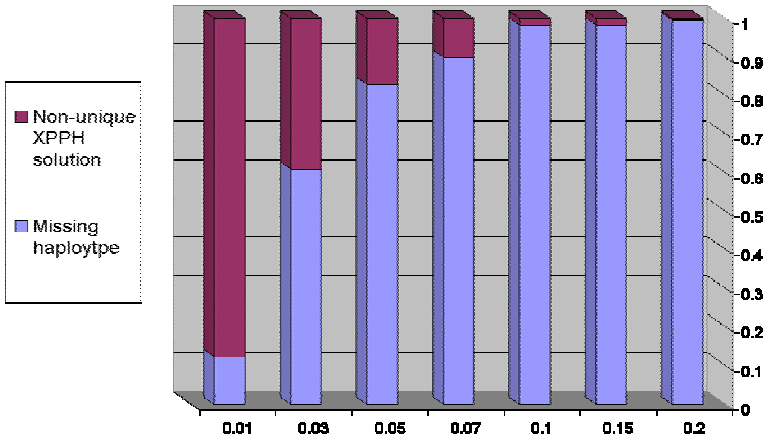


Figure 14 "The last straw". The frequency in which each of the stopping conditions was fulfilled last. Statistics were collected for varying combinations of the number of SNPs (100-2000) and the number of individuals, where the number of individuals was at least 10 times the number of xor-equivalence classes. Different bins correspond to different thresholds on the minor allele frequency cutoff α

In conclusion, for practical threshold values (α=0.05), the use of XPPH does not cause an increase in the needed sample for finding all haplotypes.

4.3. Real data

We applied XPPH on real genetic data to appreciate its qualities in a practical setting. We used mitochondrial DNA (mtDNA), which is the genetic material present in the mitochondria (the organelles that generate energy for the cell). mtDNA is maternally inherited without any recombination; therefore it is appropriate for a study under the perfect phylogeny model. (On the other hand, it is admittedly still not a fully realistic test case, since the haplotypes are not generated by paternal and maternal contributions).

We studied 560 complete European, Asian, and African mtDNA coding-region sequences from unrelated individuals, as were published by (Herrnstadt et al, 2002) and made available on http://www.mitokor.com/science/560mtdnas.php. These sequences were given unaligned and are about 16,000 base pairs long. We took the following preprocessing steps on the data:

1. We aligned the data using ClustalX (Thompson et al., 1994). In the aligned sequences we identified 1395 aligned positions (columns) with differences, and isolated the polymorphisms among them. This resulted in 1352 substitutions (43 of the differences were due to insertions/deletions, which are mutation types that were not handled by this work).
2. We removed 23 tri-allelic polymorphisms, leaving 1329 biallelic SNPs.

3. We removed the SNPs with minor allele frequency of less than 0.05 among the 560 haplotypes, which left us with 54 SNPs (0.05 is a standard threshold). See Fig 15.
4. We heuristically removed possible recurrent-mutations; by applying a greedy approach that iteratively discards a SNP that participates in the largest number of four-gametes; that led to the final removing of 33 SNPs.

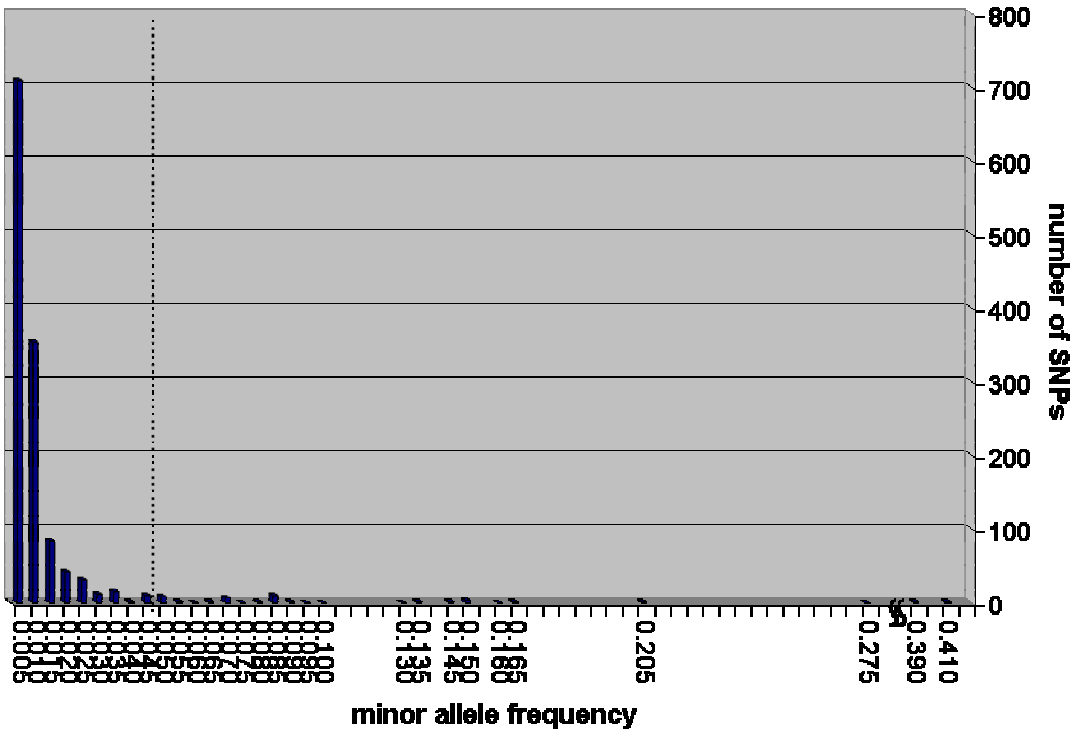


Figure 15 The distribution of minor allele frequencies in the mtDNA data. Each bin indicates the number of SNPs (y-axis) that bear a minor allele frequency within a 0.005 range (x-axis). Note that the vast majority of the SNPs have very rare alleles: some 700 had minor allele frequency < 0.005, and only 54 SNPs out of 1329 had minor allele frequency > 0.05 (the vertical dotted line).

After applying the preprocessing steps we were left with 21 SNPs, for which there are 10 distinct haplotypes among the population of 560 sequences; their perfect phylogeny can be observed in Fig 16. Note that the sharp reduction in the data was due to rare differences (which may be due to sequencing errors, misalignments or truly rare SNPs). The same type of errors as well as solution suboptimality may account for the reduction in the number of SNPs needed to obtain a coalescent data. (The complexity of reducing the least number of SNPs is open, though reducing the least number of individuals is known to be NP-hard for genotype data (Eskin et al., 2003)).

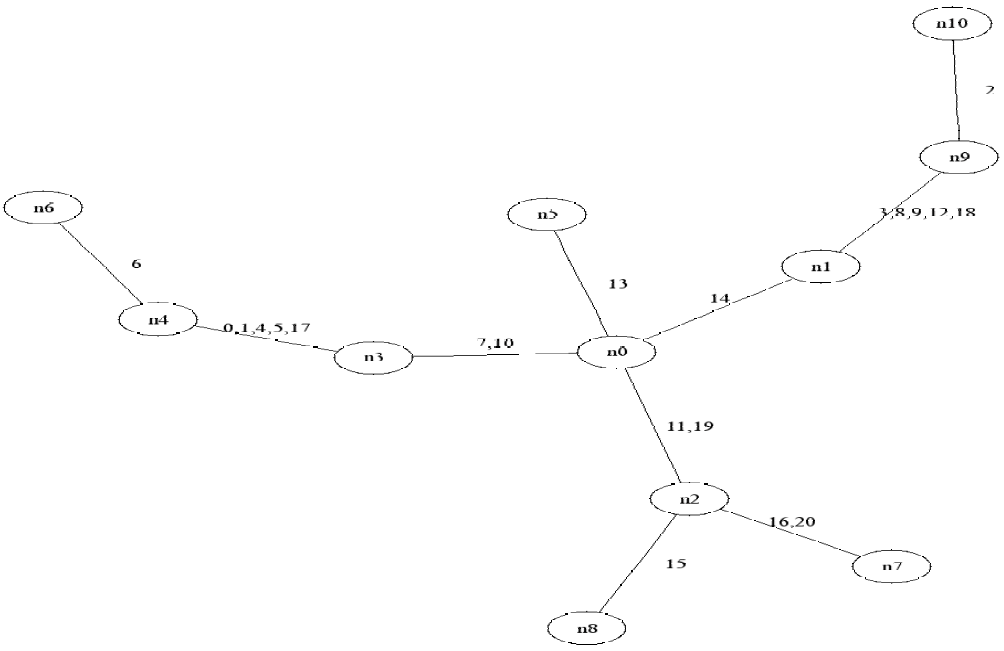


Figure 16 The perfect phylogeny of mtDNA data post processing. This is a perfect phylogeny of 10 observed haplotypes over 21 SNPs, while n3 corresponds to the ancestral haplotype. The distribution of

4.3.1. XPPH

Even though mtDNA does not appear as diploid in our genome, we can create virtual diploids from pairs of mtDNA haplotypes to obtain a sample that mimics real data. We used the processed 560 mtDNA sequences for pairing random choices to generate xor-genotypes of individuals, thereby maintaining haplotype frequencies. We evaluated experimentally the number of individuals required for obtaining a unique solution (Fig 17) and found that 70 individuals guarantee ~90% chance of uniqueness, and increasing the number of individuals has only a minor effect.

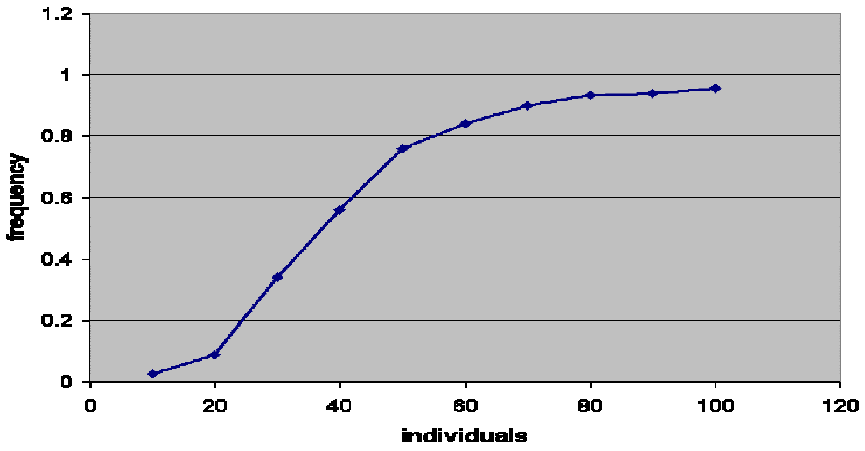


Figure 17 The chance for a unique solution for xor-genotypes from mtDNA data. The plots show the frequency of a unique solution (y-axis) versus the number of individuals tested (x-axis). XPPH statistics are based on 2000 runs.

Chapter 5. Informative SNPs

5.1. Problem Formulation

5.1.1. Informative SNPs

This section first describes the work of Bafna et al. (2003), where informative SNPs are introduced and handled, and then an equivalent formulation that we shall use later.

Let D^s_{ij} be the event that haplotypes i and j have different alleles for SNP s.
The *informativeness* of SNP s with respect to SNP t is defined by $I(s,t)=Pr_{i\neq j}(D^s_{ij}|D^t_{ij})$, where i,j are drawn uniformly from the set of all distinct haplotype pairs.
For $S'\subseteq S$ let $D^{S'}_{ij}$ be the event that haplotypes i and j have different alleles for some SNP $s\in S'$. Define $I(S',t)=Pr_{i\neq j}(D^{S'}_{ij}|D^t_{ij})$.
Finally, for $S',S''\subseteq S$, define $I(S',S'')=\sum_{t\in S''}I(S',t)$.

The following problems arise:

- Minimum informative SNPs (MIS)
  Input: A set $H=\{H_1,\ldots,H_n\}$ of haplotypes over a set $S=\{s_1,\ldots,s_m\}$ of SNPs, a subset $S''\subseteq S$ and an integer k.
  Goal: Find if there exists a subset $S'\subseteq S\backslash S''$ such that $I(S',S'')=|S''|$ and $|S'|\leq k$.
- Block MIS (blMIS): MIS restricted to inputs where S is a complete LD block, i.e., the haplotypes on S are coalescent.

Bafna et al. show that MIS is NP-complete in the general case. They give an $O(nm)$ algorithm for blMIS in the special case where $|S''|=1$. The problem blMIS is left open for the general case where $|S''|=1$, in (Bafna et al. 2003; Halldorsson et al., 2003).

Now we give an equivalent definition for informative SNPs, which we find more direct. In addition, we introduce the notion of cost for individual SNPs, so that one can add more considerations to the choice of informative SNPs.

An alternative definition of informative SNPs is the following (recall previous definition from section 2.6.2):

Definition: Let $H=\{H_1,\ldots,H_n\}$ be a set of haplotypes over a SNP set $S=\{s_1,\ldots,s_m\}$. Let $S''\subseteq S$ be a given subset of interesting SNPs. The set $S'\subseteq S\backslash S''$ is *informative* on H w.r.t. S'' if for each $1\leq k,l\leq n$, whenever there is a SNP $s''\in S''$ for which $H_{ks''}\neq H_{ls''}$, there is a SNP $s'\in S'$ for which $H_{ks}\neq H_{ls'}$ (see Fig 2).

Intuitively, by knowing the alleles of the informative SNPs one can uniquely determine the alleles of the interesting SNPs.

We define the cost of a SNP set by a cost function as follows:

Definition: A *cost function* is a function of type $C:S\rightarrow R^+$. For a cost function $C:S\rightarrow R^+$ we define the *cost* of a subset $S'\subseteq S$ by $C(S')=\sum_{s\in S'}C(s)$.

We generalize the MIS problem as follows:

Problem 6: Minimum-Cost Informative SNPs (MCIS):
Input: (a) A set of haplotypes $H=\{H_1,\ldots,H_n\}$ over a SNP set $S=\{s_1,\ldots,s_m\}$ along with a perfect phylogeny (T,f) for H.
(b) A set of interesting SNPs $S''\subseteq S$.
(c) A cost function C.
Goal: Find a set $S'\subseteq S\backslash S''$ of minimum total cost that is informative w.r.t. S''.

(T,f) may already be known if H was found by solving XPPH. Alternatively, it can be computed in O(mn) time from haplotypes (Gusfield, 1997).

A common task which is related to picking an informative SNP set is to describe all of the haplotype variation in the region (Johnson et al., 2001). Formally, we seek a *tag SNPs set* $S'\subseteq S$ *s.t.* for each $1\leq l,k\leq n$, there is $t\in S'$ for which $H_{kt}\neq H_{lt}$ (see Fig 2). In order to find tag SNPs of minimum cost, one could duplicate the SNP set S and define one of the copies as interesting. A solution to MCIS on the duplicated instance is a tag SNP set of minimum cost. Hence we shall focus on the more general MCIS problem.

5.2. Algorithmic solution

We present here a dynamic programming solution to MCIS, which works in $O(m)$ time and space. We decompose (T,f) into components and solve MCIS independently for each component.

5.2.1. Reducing MCIS to an Ancestral component

Recall that if $T=(V,E)$ is a perfect phylogeny for $H_{n\times m}$ then $\{1,\ldots,n\}\subseteq V$, i.e., the haplotypes of H label nodes in the perfect phylogeny.

Definition: If a node of T is labeled by a haplotype from H we say it is *observed*. Otherwise we say it is *ancestral*.

Ancestral nodes represent haplotypes that have been intermediate stages in evolution but did not survive to the present, or were not collected in the sample.
Notice that informative SNPs need to distinguish the observed haplotypes but needn't distinguish the ancestral haplotypes.

Proposition 4: The leaves of T are observed.

Proof:
Take $(v,u)\in E$ such that u is a leaf of T, and take $s\in f^{-1}(v,u)$. There are $1\le i,j\le n$ such that H_{is} bears the 0-allele for s and H_{js} bears the 1-allele, hence by the key property, the path from i to j in T must go through (v,u). But since u is a leaf, either $u=i$ or $u=j$, therefore u is observed.□

Definition: An *ancestral component* is a subtree of T in which all the internal nodes are ancestral and all the leaves are observed.

Since the leaves of T are observed, T can be represented as a union of edge-disjoint ancestral components, where each union step merges two components by identifying two copies of the same observed node. Two different components can share at most one observed haplotype, but do not share ancestral haplotypes. Partitioning T into ancestral components is straightforward, e.g. by DFS.

We now show that in order to find informative SNPs we can divide the tree into ancestral components and find informative SNPs for each single component separately. The subproblem on a component is defined as follows:

Denote an instance of MCIS by the input tuple $I=(H,S,C,T,f,S'')$. Let $T_1,\ldots,T_p$ be T's ancestral components where $T_i=(V_i,E_i)$. Denote by $S_i\subseteq S$ the SNPs that label E_i. The input tuple for T_i is $I_i=(H_i,S_i,C_i,T_i,f_i,S_i'')$ where the sets and function are the restriction of the original sets and function to S_i.

Theorem 3: Suppose for every i, $IS(I_i)$ is a minimum cost informative SNPs set for the instance I_i. Then $IS(I)=IS(I_1)\cup\ldots\cup IS(I_p)$ is a minimum cost informative SNPs set for I.
Proof:
We shall show that $IS(I)$ is informative w.r.t. S'' iff $IS(I_i)$ is informative w.r.t. S_i'' for all i; The theorem then will follow by the additivity of the cost function.
Assume $IS(I)$ is informative w.r.t. S''. If haplotypes k,l belong to the same observed component T_i, and there is a SNP $s\in IS(I)$ such that $H_{ks}\neq H_{ls}$, then by the key property it must be that $s\in S_i$ hence $s\in IS(I_i)$. Therefore $IS(I_i)$ is informative w.r.t. S_i'' for all i.
Assume $IS(I_i)$ is informative w.r.t. S_i'' for all i. Suppose there are $t\in S''$ and $1\le l,k\le n$ such that $H_{kt}\neq H_{lt}$. Let T_i be the subtree which contains the edge with label t (i.e., $t\in S_i$). Then by the key property, there are l',k' in T_i such that $H_{k't}\neq H_{l't}$ (specifically, l',k' are the observed nodes of T_i that are on the path from k to l in T). But then there is $s'\in IS(I_i)\subseteq IS(I)$ such that $H_{k's'}\neq H_{l's'}$ hence by the key property $H_{ks'}\neq H_{ls'}$.□

5.2.2. Informative SNPs for ancestral component

5.2.2.1. Problem definition

The reduction of MCIS to an ancestral component results in the following formulation:

Problem 6': MCIS for ancestral components (acMCIS):
Input: The same input as in MCIS with the requirement that all the internal nodes of (T,f) are ancestral and all the leaves are observed.

Goal: Find a set $S' \subseteq S \setminus S''$ of minimum total cost that is informative w.r.t. S''.

First we reformulate acMCIS in terms of the tree edges.
Notation: (1) Edges labeled by interesting SNPs are called *target edges*. The set of target edges is $\tau = \{e | f^{-1}(e) \cap S'' \neq \emptyset\}$. It specifies the interesting information in terms of tree edges. (2) An edge is *allowed* if it is labeled by some non-interesting SNP. The set of allowed edges is $\alpha = \{e | f^{-1}(e) \cap (S \setminus S'') \neq \emptyset\}$. These are the edge-analogs of potentially informative SNPs. (3) Edges in $\tau \setminus \alpha$ are called *forbidden*. Forbidden edges cannot be used as informative.

We now expand the definition of the cost function to edges.
Definition: The *cost of an edge e,* denoted $C(e)$*,* is the minimum cost of a non-interesting SNP that labels *e.* For $e \in \tau \setminus \alpha$ define $C(e) = \infty$.

This allows us to provide an equivalent formulation for acMCIS:
Problem 7: Minimum Cost Separating Set (acMCSS)
Input: The same input as for acMCIS.
Goal: Find $E' \subseteq E$ of minimum cost, such that in $G = (V, E \setminus E')$ there are no two observed nodes that are connected by a path containing a target edge.

Proposition 5: acMCIS and acMCSS are equivalent.
Proof: It suffices to show that an informative set for H w.r.t. S'' separates those observed nodes that are connected by a path containing edges from τ, and vice versa. Assume that S' is an informative set for H w.r.t. S''. If the observed nodes k,l are connected by $e \in \tau$, then by the key property $t \in f^{-1}(e) \cap S''$ is such that $H_{kt} \neq H_{lt}$. But then there is $s \in S'$ such that $H_{ks} \neq H_{ls}$, therefore S' separates k,l.
Assume that S' separates those observed nodes that are connected by a path containing edges from τ. If for the observed haplotypes k,l there is $t \in S''$ such that $H_{kt} \neq H_{lt}$, then by the key property k,l are connected by $f(t)$ and by definition $f(t)$ is a target edge. Therefore S' separates k,l.□

5.2.2.2. A solution to acMCSS

We are now ready to outline a dynamic programming algorithm for acMCSS. W.l.o.g. assume $|V| > 2$. Take some internal node $r \in V$ and root T at r. For $v \in V$ denote by $T_v = (V_v, E_v)$ the subtree of T that is rooted at v. For a solution $S_v \subseteq E_v$ of the induced sub instance $I(T_v)$, denote by R_v the connected component which contains v in $G_v = (V_v, E_v \setminus S_v)$. The algorithm will scan T from the leaves up and at each node v form an optimal solution for the subtree T_v based on the optimal solutions for the subtrees of its children. When combining such children solutions, we have to take into consideration the possibility that the combination will generate new paths between observed haplotypes, with or without target edges on them. To do this, we distinguish three types of solutions: S_v is called *empty* if there are no observed haplotypes in R_v. It is called *connected* if some observed haplotypes in G_v are connected to v in R_v by a path containing a target edge. S_v is called *disconnected* otherwise, i.e., if there are observed haplotypes in R_v but there is no path connecting an observed haplotype to v via target edges.

The algorithm

Let N_v, P_v and A_v denote the best empty, connected, or disconnected solutions respectively on T_v. We define recursive formulae for their costs as follows:

- For a leaf node $v \in V$ we initialize: $C(N_v)=\infty$, $C(P_v)=\infty$, $C(A_v)=0$.
- For an internal node $v \in V$ with children $\{u_1,\ldots,u_{k(v)}\}$ we write:

$Tear(i) = \min\{C(N_{u_i}), C(P_{u_i}) + C(v,u_i), C(A_{u_i}) + C(v,u_i)\}$		1
$C(N_v) = \sum_{i=1}^{k(v)} Tear(i)$		2
$C(P_v) = \min\left\{ \min_j \{C(P_{u_j}) + C(N_v) - Tear(j)\}, \min_{j \mid (v,u_j) \in \tau} \{C(A_{u_j}) + C(N_v) - Tear(j)\} \right\}$		3
If $\{i \mid (v,u_i) \notin \tau\} = \phi$ then $C(A_v) = \infty$	Otherwise $j = \arg\min_{i \mid (v,u_i) \notin \tau} (C(A_{u_i}) - Tear(i))$ $C(A_v) = C(A_{u_j}) + \sum_{(v,u_i) \notin \tau, i \neq j} \min\{C(A_{u_i}), Tear(i)\} + \sum_{(v,u_i) \in \tau} Tear(i)$	4

Denote by T_v^i the subtree that consists of v, the edge (v,u_i) and T_{u_i}. Note that T_v is the union of T_v^i, $i=1,\ldots,k(v)$, that is obtained by identifying all copies of v.

The auxiliary value $Tear(i)$ measures the cost of an empty solution for T_v^i.

In computing $C(N_v)$ we must take an empty solution for all T_v^i, $i=1,\ldots,k(v)$.

In computing $C(P_v)$ we must pick one T_v^j to have a connected solution and all the remaining solutions for T_v^i, $i \neq j$, must be empty. A connected solution for T_v^j can be obtained in one of two ways: (a) A connected solution for T_{u_j} (first term in 3) or (b) a disconnected solution for T_{u_j} when (v,u_j) is a target edge (second term).

In computing $C(A_v)$ we must pick one T_v^j to have a disconnected solution and the remaining solutions for T_v^i, $i \neq j$, may either be disconnected or empty. A disconnected solution for T_v^i is possible only when $(v,u_j) \notin \tau$, and in that case it is obtained by taking the disconnected solution for T_{u_j}.

These formulae are implemented in a dynamic program as follows: (1) Visit the nodes in postorder, computing $C(N_v)$, $C(P_v)$ and $C(A_v)$ for each $v \in V$. Upon reaching the root r, compute the minimal cost $\min\{C(N_r),C(P_r),C(A_r)\}$. (2) Compute N_v, P_v and A_v by following the traceback pointers to get all those edges (v,u_j) that were chosen by the minimal cost while taking $C(P_{u_i}) + C(v,u_i)$ or $C(A_{u_i}) + C(v,u_i)$.

Proof of correctness

We need to prove that the formulae for $C(N_v)$, $C(P_v)$ and $C(A_v)$ satisfy our definitions. The correctness will then follow because any solution is either empty, connected or disconnected.

To prove the correctness of the formulae we need to prove the following: (a) A solution for T_v of finite minimal cost implies a solution for each T_{u_i}, which is either empty, connected or disconnected of minimal cost. (b) Empty, connected and disconnected solutions for T_{u_i} with minimal score, can be combined optimally into a solution for T_v by the above formulae.

(a) If k,l are observed nodes in T_{u_i} then the path between k and l is completely in T_{u_i}, therefore if they need to be separated then it would have to be by edges from T_{u_i}. Hence a solution for T_v must induce also a solution for every T_{u_i} (i.e., if E' is a solution on T_v then the induced edge set on T_{u_i}, $E' \cap T_{u_i}$, solves T_{u_i}). Moreover, if a solution for T_v induces an empty solution on T_{u_i}, then any empty solution for T_{u_i} can be used instead. This is also true for connected and disconnected solutions. Therefore the induced solution on T_{u_i} must be the minimum of its type.

(b) Proof by induction on the distance of v from the leaves.
Base: For $v \in V$ that is a leaf of T, T_v is a disconnected component, hence $C(N_v)=\infty$, $C(P_v)=\infty$ and $C(A_v)=0$.
Step: Assume correctness for $C(N_{u_i}), C(P_{u_i}), C(A_{u_i})$, $i=1,\ldots,k(v)$.

1. Disconnecting all the leaves of T_{u_i} from v is done either by disconnecting all those leaves from u_i or by disconnecting (v,u_i). $Tear(i)$ is the minimal cost of such action. Hence $C(N_v)$ is indeed the minimal cost of disconnecting all leaves from v.
2. If there is a leaf of T_{u_i} that is connected to v via a target edge, then only the leaves of T_{u_i} can be connected to v, otherwise two leaves would have been connected trough v via target edge. A leaf of T_{u_i} is connected to v via a target edge if either it is connected to u_i via a target edge or (v,u_i) is a target edge. $C(P_v)$ is indeed the minimal cost of connecting the leaves of one T_{u_i} to v via target edges, and disconnecting all other leaves.
3. For leaves to be connected to v not through target edges, there must be at least one subtree T_{u_j} where leaves are connected to u_j not through target edges, and $(v,u_j) \notin \tau$ must hold. Leaves of other T_{u_i} with $(v,u_i) \notin \tau$ may remain connected or may be disconnected from v, while leaves of T_{u_i} with $(v,u_i) \in \tau$ must be disconnected from v. $C(A_v)$ is indeed the minimal cost of forcing one subtree to be disconnected and allowing the rest to be either disconnected or empty. Notice that if $Tear(i) > C(A_{u_i})$ for all $(v,u_i) \notin \tau$, then by taking $C(A_{u_j})$ instead of $Tear(j)$, we reduce the cost by $C(A_{u_i}) - Tear(i)$, hence this factor needs to be minimized in order to find j.□

Algorithm complexity

The time complexity is $O(m)$: To compute $C(N_v)$, $C(P_v)$ and $C(A_v)$ we visit v's children for a constant number of times, spending $O(1)$ time for each child (due to visiting the tree in postorder). The same hand holds for following the traceback pointers.
The space complexity is $O(m)$: To compute $C(N_r)$,$C(P_r)$ and $C(A_r)$ we visit the tree in postorder, maintaining three costs for each vertex, which totally require $O(m)$ space. Then to compute N_r, P_r and A_r we follow the traceback pointers to get the optimal set of edges, which is of size $O(m)$.

5.3. Tag SNPs from genotypes

Up until now we have followed the standard assumption in the computational literature (Bafna et al., 2003; Sebastiani et al. 2003; Chapman et al. 2003) that tag SNPs need to reconstruct the full binary haplotypes from binary haplotypes of the tag set. As experiments that provide haplotypes are expensive, most studies seek to obtain experimentally only genotypes. For such data, the problem of finding tag SNPs should be reformulated to reflect the fact that the input consists of *genotypes,* rather than haplotypes: Find a subset of SNPs that given their genotype calls, i.e., 0 or 1 for homozygote and 2 for heterozygote, one can completely identify the pair of haplotypes of an individual. We call such subset *phasing tag SNPs*.

Notation: Let H be a set of haplotypes over a set S of SNPs. Denote by $g(k,l)$ the genotype formed from H_k and H_l. Denote by $g(k,l)_{S'}$ the restriction of this genotype to the SNP set $S' \subseteq S$.

Definition: We say that the haplotype pairs $\{i_1,i_2\}$ and $\{j_1,j_2\}$ are *distinct with respect to* S if there is $s \in S$ such that $g(i_1,i_2)_s \neq g(j_1,j_2)_s$. Note that we could distinguish $\{i_1,i_2\}$ from $\{j_1,j_2\}$ given their genotypes.

We now give a formal definition to a phasing tag SNP set (recall previous definition from section 2.6.4):

Definition: $S' \subseteq S$ is a set of *phasing tag SNPs* w.r.t. a set of haplotypes H if every two haplotype pairs from H are distinct with respect to S'.

Intuitively, from the genotype calls of an individual for the set S', one can uniquely determine the exact sequence of the complete set S for each of its two haplotypes.

Problem 8: Phasing tag SNPs:
Input: A set of haplotype H over a set of SNPs S.
Goal: Find a minimum set $S' \subseteq S$ of phasing tag SNPs.

We can define analogously informative and minimum cost informative sets.

In general, the definitions of phasing tag SNPs and tag SNPs differ (see Fig 2). The former is stronger:

Proposition 6: If $S' \subseteq S$ are phasing tag SNPs then they are also tag SNPs.
Proof:
All homozygous genotype-call vectors are distinct w.r.t. S': since S' is a phasing tag SNPs set, for each $i \neq j$, $g(i,i)_{S'} \neq g(j,j)_{S'}$. Hence for every $i \neq j$ there is $s \in S'$ such that $H_{is} \neq H_{js}$.□

We now show that, surprisingly, under the perfect phylogeny model, tag SNPs and phasing tag SNPs are equivalent.

Theorem 4: Suppose that the haplotypes in H satisfy the perfect phylogeny model on S. A set $S' \subseteq S$ is a tag SNPs set if and only if S' is a phasing tag SNPs set.
Proof:
By Proposition 6, it suffices to prove the "only if" direction. Suppose to the contrary that S' are tag SNPs but not phasing tag SNPs. Let $G_i=\{H_1,H_2\}$ and $G_j=\{H_3,H_4\}$ be distinct haplotype pairs with the same genotype call vector for S', i.e., $g(1,2)_{S'}=g(3,4)_{S'}$. Notice that being distinct pairs implies that H_1, H_2, H_3, and H_4 are distinct because G_i and H_1 determine H_2 etc. Since S' is a tag SNP set, it distinguishes H_1 and H_3, so there must be $s' \in S'$ such that G_i and G_j are heterozygous to s_1, and H_1 and H_3 have different alleles for s_1. If w.l.o.g. $H_{1s'}=1$ then $H_{2s'}=0$, $H_{3s'}=0$ and $H_{4s'}=1$. Similarly, there must be $s'' \in S'$ such that G_i and G_j are heterozygous to s'', and H_1 and H_4 have different alleles for s''. Note that $s' \neq s''$ since $H_{1s'}=H_{4s'}$. If w.l.o.g. $H_{1s''}=1$ then $H_{2s''}=0$, $H_{3s''}=1$ and $H_{4s''}=0$. Therefore G_i and G_j are oppositely phased on s' and s'', violating the 4 gamete rule, in contradiction to the perfect phylogeny model.□
Notice that Theorem 4 can be applied to informative SNPs as well.

Theorem 4 justifies the use of tag SNPs on genotype data, which until now was only heuristically argued. Assuming the perfect phylogeny model, we can now apply the ideas and algorithms developed for tag SNPs (in this study and in (Bafna et al., 2003)) on genotypes and actually obtain phasing tag SNPs.

6. Summary and Discussion

We studied here several questions arising in haplotype inference under the perfect phylogeny model. Solutions to these problems contribute to the genetic analysis of a region, along the process from haplotype determination toward their utilization in a genetic study.

We introduced the model of xor-genotypes, and presented the computational foundation for the use of such data: (i) Inference of the sample haplotypes (up to bit-flipping) by adapting graph realization algorithms (ii) Concrete inference of the sample haplotypes by only two or three additional full genotypes.

We argued for the potential economical advantage of xor-genotypes over the full genotypes common today. We showed that both simulated and real genetic data indicate that xor-genotypes are nearly as informative as full genotypes, at a fraction of the cost. Hence, genotyping methods that distinguish only between heterozygotes and homozygotes are competitive and cost-effective, and could potentially be applied to large scale genetic studies.

We provided software for graph realization (and for the xor perfect phylogeny haplotyping), which can hopefully contribute to genetic studies and possibly to other applications as well.

In case the haplotype set is available, we went on to select a small informative SNPs set that fully describes an interesting portion of the sample haplotypes. We introduced per-SNP costs in the selection, which allows picking a set of minimal cost. Typing the set of informative SNPs would then be the most economical method for determining an individual's haplotype. We provided an efficient algorithm for this problem. This extends previous results that could handle only a single interesting SNP and resolves an open problem stated in the literature.

Finally, we have shown how to find tag SNPs for genotype data. We have shown that the set of tag SNPs for haplotype data is equivalent to the set of tag SNPs for genotype data. This result allows the adopting of previous variant methods (including our own) for finding tag SNPs for haplotype data.

This work is an example of how computer science theory can be applied to several practical aspects of a genetic study. The ability to understand and formulate biological needs was combined here with graph theoretical methodology and theory. The result is a set of algorithms and theoretical results that are interesting from the computer science theoretic point of view and are significant from the practical and economical points of view.

This work leaves many interesting problems for further studying. A significant question is with regard to the perfect phylogeny assumption. This assumption has the advantage of

using powerful theoretical mechanisms to take care of many practical peculiarities of the data. The shortcoming of our approach is that perfect phylogeny is only a local approximation to the observed biological reality. Extending our work to incorporate noisy variants of this highly constrained model is desirable. In particular, handling "small deviations" from perfect phylogeny, data errors and missing data, are all open issues.

References

Glossaries:
http://helios.bto.ed.ac.uk/bto/glossary/
http://www.ornl.gov/sci/techresources/Human_Genome/glossary/
http://www.weihenstephan.de/~schlind/genglos.html

Arkin EM and Hassin R. Multiple-choice minimum diameter problems. Unpublished manuscript, 1992.

Bafna V, Gusfield D, Lancia G, and Yooseph S. Haplotyping as Perfect Phylogenty: A direct approach. Technical Report U.C. Davis CSE-2002-21, 2002.

Bafna V, Halldórsson BV, Schwartz R, Clark AG and Istrail S. Haplotypes and informative SNP selection algorithms: don't block out information. *Proceedings of the Seventh Annual International Conference on Computational Biology* 2003 (*RECOMB* '03):19-27.

Bixby R.E and Wagner D.K. An almost linear-time algorithm for graph realization, *Mathematics of Operations Research*, 1988; 13, 99-123.

Chapman JM, Cooper JD, Todd JA, Clayton DG. Detecting disease associations due to linkage disequilibrium using haplotype tags: a class of tests and the determinants of statistical power. *Human Heredity*, 2003; 56(1-3):18-31.

Chung RH and Gusfield D. Perfect Phylogeny Haplotyper: Haplotype Inferral Using a Tree Model. *Bioinformatics*, 2002; 19(6):780-781.

Chung RH and Gusfield D. Empirical Exploration of Perfect Phylogeny Haplotyping and Haplotypers. *Proceedings of the 2003 Cocoon Conference*.

Clark AG. Inference of haplotypes from PCR-amplified samples of diploid populations. *Molecular Biology and Evolution*, 1990; 7(2):111-22

Cummins J. Mitochondrial DNA and the Y chromosome: parallels and paradoxes. *Reproduction, Fertility and Development*, 2001; 13(7-8):533-42.

Daly MJ, Rioux JD, Schaffner SF, Hudson TJ, Lander ES. High resolution haplotype structure in the human genome. *Nature Genetics*, 2001; 29(2):229-32.

Eskin E, Halperin E and Karp RM. Efficient reconstruction of haplotype structure via perfect phylogeny. To appear in the *Journal of Bioinformatics and Computational Biology* (*JBCB*), 2003.

Excoffier L and Slatkin M. Maximum-likelihood estimation of molecular haplotype frequencies in a diploid population. *Molecular Biology and Evolution*, 1995; 12(5):921-7.

Gabriel SB, et al. The structure of haplotype blocks in human genome. *Science*, 2002; 296: 2225-9.

Garey M.R. and Johnson D.S. Computers and Intractability, p. 222 Freeman, New York, 1979.

Gavril F and Tamari R. An algorithm for constructing edge-trees from hypergraphs, *Networks* 1983; 13, 377-388.

Gusfield D. Algorithms on Strings, Trees, and Sequences - Computer Science and Computational Biology. Cambridge University Press 1997

Gusfield D. Haplotyping as Perfect Phylogeny: Conceptual Framework and Efficient Solutions. *Proceedings of the Sixth Annual International Conference on Computational Biology* 2002 (*RECOMB* '02):166-75.

Halldorsson B.V, Bafna V, Edwards N, Lippert R, Yooseph S, Istrail S. Combinatorial Problems Arising in SNP and Haplotype Analysis. *Discrete Mathematics and Theoretical Computer Science*, 2003; (DMTCS '03): 26-47.

Herrnstadt C, Elson J.L, Fahy E, Preston G, Turnbull D.M, Anderson C, Ghosh S.S, Olefsky J.M, Beal M.F, Davis R.E and Howell N. Reduced-Median-Network Analysis of Complete Mitochondrial DNA Coding Region Sequences for the Major African, Asian, and European Haplogroups. *American Journal of Human Genetics*, 2002; 70:1152-1171.

Hudson RR. Generating samples under a Wright-Fisher neutral model of genetic variation. *Bioinformatics*, 2002; 18:337-38.

Jeffreys AJ, Kauppi L and Neumann R. Intensely punctate meiotic recombination in the class II region of the major histocompatibility complex. *Nature Genetics*, 2001; 29:217–22.

Johnson GC, et al. Haplotype tagging for the identification of common disease genes. *Nature Genetics*. 2001 Oct; 29(2): 233-7.

Kwok PY. Genetic association by whole-genome analysis. *Science*, 2001; 294(5547):1669-70.

Nachman MW and Crowell SL. Estimate of the mutation rate per nucleotide in humans. *Genetics*, 2000; 156**:** 297-304.

NIH RFA HG-02-005 (2002) Large-scale genotyping for the haplotype map of the human genome.

Patil N, et al. Blocks of Limited Haplotype Diversity Revealed by High Resolution Scanning of Human Chromosome 21. *Science*, 2001; 294(5547):1719-23

Pe'er I and Beckmann JS. Resolution of haplotypes and haplotype frequencies from SNP genotypes of pooled samples. *Proceedings of the Seventh Annual International Conference on Computational Biology* 2003; (*RECOMB* '03): 237-246

Sachidanandam R, et al. (International SNP Map Working Group). A map of human genome sequence variation containing 1.42 million single nucleotide polymorphisms. *Nature*, 2001; 409(6822): 928-33.

Sebastiani P, Lazarus R, Weiss ST, Kunkel LM, Kohane IS, Ramoni MF. Minimal haplotype tagging. *Proceedings of the National Academy of Sciences of the USA*, 2003; 100(17):9900-5.

Thompson JD, Higgins DG and Gibson TJ. CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, positions-specific gap penalties and weight matrix choice. Nucleic Acids Research, 1994; 22:4673-4680.

Tutte WT. An Algorithm for determining whether a given binary matroid is graphic. *Proceedings of American Mathematical Society*, 1960; 11:905-917.

Xiao W and Oefner PJ, Denaturing high-performance liquid chromatography: A review. *Human Mutation*, 2001; 17(6):439-74.

Zhang K, Deng M, Chen T, Waterman MS and Sun F. (2002) A dynamic programming algorithm for haplotype block partitioning. *Proceedings of the National Academy of Sciences*, 99(11):7335-9.