OPTIMAL IMPERFECT PHYLOGENY RECONSTRUCTION AND HAPLOTYPING (IPPH)

Srinath Sridhar
Computer Science Department, Carnegie Mellon University
Pittsburgh, PA 15213. USA.
Email: srinath@cs.cmu.edu

Guy E. Blelloch
Computer Science Department, Carnegie Mellon University
Pittsburgh, PA 15213. USA.
Email: guyb@cs.cmu.edu

R. Ravi
Tepper School of Business, Carnegie Mellon University
Pittsburgh, PA 15213. USA.
Email: ravi@cmu.edu

Russell Schwartz
Department of Biological Sciences, Carnegie Mellon University
Pittsburgh, PA 15213. USA.
Email: russells@andrew.cmu.edu

The production of large quantities of diploid genotype data has created a need for computational methods for large-scale inference of haplotypes from genotypes. One promising approach to the problem has been to infer possible phylogenies explaining the observed genotypes in terms of putative descendants of some common ancestral haplotype. The first attempts at this problem proceeded on the restrictive assumption that observed sequences could be explained by a perfect phylogeny, in which each variant locus is presumed to have mutated exactly once over the sampled population's history. Recently, the perfect phylogeny model was relaxed and the problem of reconstructing an imperfect phylogeny (IPPH) from genotype data was considered. A polynomial time algorithm was developed for the case when a single site is allowed to mutate twice, but the general problem remained open. In this work, we solve the general IPPH problem and show for the first time that it is possible to infer optimal q-near-perfect phylogenies from diploid genotype data in polynomial time for any constant q, where q is the number of "extra" mutations required in the phylogeny beyond what would be present in a perfect phylogeny. This work has application to the haplotype phasing problem as well as to various related problems in phylogenetic inference, analysis of sequence variability in populations, and association study design. Empirical studies on human data of known phase show this method to be competitive with the leading phasing methods and provide strong support for the value of continued research into algorithms for general phylogeny construction from diploid data.

1. INTRODUCTION

Sophisticated computational methods for data refinement and interpretation have become a core component of modern studies in human genetics. Computational methods have long been central to the study of phylogenetics, or evolutionary tree building, particularly the parsimony variants best suited to inferences of relationships over short time scales. The more specialized problem of haplotype reconstruction has also benefited tremendously from contributions from the fields of discrete algorithms and combinatorial optimization. In haplotype reconstruction, also called *phasing*, one seeks to separate the alleleic contributions of two chromosomes observed together in a diploid genotype assay. If we use the symbols 0 and 1 to represent homozygous and 2 to represent heterozygous alleles then '0221' is a genotype typed on four loci. Two pairs of haplotypes 0001, 0111 and 0011, 0101 are both consistent with the genotype and the goal of phasing is to correctly infer the true haplotypes, given the genotypes. The problem has relevance to basic research into population histories as well as to applied problems such as statistically linking haplotypes to disease phenotypes.

The field of computational haplotype reconstruction began with a fast and simple iterative method based on the idea that a haplotype that is observed in one individual is likely to be found in other individuals as well[8]. Various statistically motivated algorithms based on heuristic optimization techniques such as expectation maximization[28] and Markov chain Monte Carlo methods[35] have since been developed. Such algorithms provide significantly improved robustness and accuracy, although still poor scaling in problem size. Several combinatorial optimization methods have also been formulated based on Clark's original method[20], although these have all been intractable in theory and practice. The advent of large-scale genotyping created a need for new methods designed to scale to chromosome-sized data sets.

Recently, a new avenue towards haplotype reconstruction was developed based on phylogenetic methods. In such an approach, one seeks to identify the ancestral history that could have produced a set of haplotype sequences from a common ancestor such that the inferred haplotypes could give rise to the observed genotypes. Gusfield[19] showed that it is possible to directly and efficiently infer phylogenies that could explain an observed set of diploid genotypes provided the phylogenies are *perfect*, meaning that each character mutates only once from the common ancestor over the entire phylogeny. This problem is referred to as *Perfect Phylogeny Haplotyping* (PPH). Several subsequent results simplified and improved on this original method[2, 3, 12, 13]. The work produced a fast, practical method for large-scale haplotyping, in which one breaks a large sequence into blocks consistent with perfect phylogenies and uses the phylogenies to phase those sequences.

The perfect phylogeny assumption is quite restrictive, though, and several approaches have been taken to adapt the perfect phylogeny method to more realistic models of molecular evolution. Data inconsistent with perfect phylogenies can arise from multiple mutations of a base over the history of a species or through the processes of recombination or gene conversion, which can assemble hybrid chromosomes in which different segments have different phylogenies. Heuristic methods have allowed some tolerance to recurrent mutations (for example, the work by Halperin and Eskin[23]) resulting in the generation of *imperfect phylogenies*. Imperfect phylogeny haplotyping has proven to be very fast and competitive with the best prior methods in accuracy. Some recent work has been directed at provably optimal methods for imperfect phylogeny haplotyping (IPPH), resulting in a polynomial-time algorithm for the case when a single site is allowed to mutate twice (or a single recombination is present), under a practical assumption on the input data[33]. But the general IPPH problem remained open. It appears for the moment intractable in both theory and practice to infer recombinational histories in non-trivial cases.

Our Contributions: In this work, we solve the general IPPH problem and show for the first time that it is possible to infer imperfect phylogenies with any constant number q of recurrent mutations in polynomial time in the number of sequences and number of sites typed. Our approach builds on both the prior theory on phylogeny construction from diploid data[13, 33] and a separate body of theory on the inference of imperfect phylogenies from haplotype data[4, 14, 21, 25, 31]. Our algorithm reconstructs a *q-near-perfect phylogeny* in time $nm^{O(q)}$ where m is the number of characters (variant sites typed), n is the number of taxa (sequences examined), and q is the *imperfectness* of the solution, defined below. The prior method of Song et al.[33] that solves the 1-near-perfect phylogeny haplotyping problem relied on an empirically but not theoretically supported assumption that an embedded perfect phylogeny problem will have a unique solution. We relax this assumption to allow a polynomial number of such solutions and show that the relaxed assumption holds with high probability if the population obeys a common assumption of population genetics theory called Hardy-Weinberg equilibrium. We thus provide a theoretical basis for Song et al.'s empirical observation. We demonstrate and validate our algorithm by comparing with leading phasing methods using a collection of large-scale genotype data of known phase[29]. We find that our method is efficient and more accurate on blocks with small q. We further provide strong empirical support for the value of continuing research into accelerated algorithms for phylogeny construction from diploid data.

2. PRELIMINARIES

We begin by introducing definitions for parsimony-based phylogeny reconstruction. In such problems, we wish to study the relationship between a set of n taxa, each of which is defined over a set of m binary characters. Input I will be represented by a matrix, where row set $R(I)$ represent taxa and column set $C(I)$ represent characters. In a *haplotype matrix* all taxa $r_i \in \{0,1\}^m$ and in a *genotype matrix* $r_i \in \{0,1,2\}^m$.

Definition 2.1. A *phylogeny* for $n \times m$ binary input matrix I is a tree $T(V,E)$ and a label function $l : V(T) \rightarrow \{0,1\}^m$ with the following properties: $R(I) \subseteq l(V(T))$ and for all $(u,v) \in E(T)$, $d(l(u), l(v)) = 1$ where d is the Hamming distance. That is, every input taxon appears in T and the Hamming distance between adjacent vertices is 1.

Definition 2.2. We define the following *terms* for a phylogeny T on input I:

- character $\mu(e)$ represents mutation on edge $e = (u,v)$ s.t. $l(u)[\mu(e)] \neq l(v)[\mu(e)]$.
- vertex $v \in V(T)$ is *terminal* if $l(v) \in R(I)$ and *Steiner* otherwise.
- $\texttt{length}(T) = |E(T)|$.
- phylogeny T is *optimal* if $\texttt{length}(T)$ is minimized.
- $\texttt{penalty}(T) = \texttt{length}(T) - m$.
- phylogeny T is *q-near-perfect* if $\texttt{penalty}(T) = q$ and *perfect* if $\texttt{penalty}(T) = 0$.

We say that a character i mutates on edge e if $\mu(e) = i$. We will assume that both states $0, 1$ are present in all characters. Therefore the length of an optimum phylogeny is at least m. This provides a natural motivation for the *penalty* of a phylogeny as defined above. For simplicity, we will drop the label function $l(v)$ and use v to refer to both a vertex and the taxon it represents.

The IPPH problem: The input to the problem is an $n \times m$ matrix G, where each row $g_i \in \{0,1,2\}^m$ represents a (genotype) taxon and each column represents a character. The output is a $2n \times m$ matrix H in which each row $h_i \in \{0,1\}^m$ represents a haplotype. Furthermore corresponding to every taxon $g_i \in R(G)$, there are two taxa $h_{2i-1}, h_{2i} \in R(H)$ with the following properties:

- if $g_i[c] \neq 2$ then $h_{2i-1}[c] = h_{2i}[c] = g_i[c]$
- if $g_i[c] = 2$ then $h_{2i-1}[c] \neq h_{2i}[c]$

The objective of the IPPH problem is to find an output matrix H such that the length of the optimum phylogeny on H is minimized. This problem is clearly NP-hard, since if matrix G contains no 2s, then the problem is equivalent to reconstructing the most parsimonious phylogenetic tree[15]. We therefore consider the following parameterized version of the problem. Given matrix G and parameter q, we return matrix H such that there exists an optimal phylogeny T^* on H with $\textit{penalty}(T^*) \leq q$, under the assumption that such a matrix H exists. Note that the PPH problem is a restriction of IPPH when $q = 0$.

Definition 2.3. For a set of binary state taxa S and a set of characters C, the set of *gametes* $GAM(S,C)$ is the projection of S on characters C. In other words $(x_1, \ldots, x_{|C|}) \in GAM(S,C)$ i.f.f. there exists $s \in S$ with $s[c_i] = x_i$ for all $c_i \in C$. A set of characters C *shares* k *gametes* if $|GAM(S,C)| = k$. A pair of characters i, j *conflict* if $|GAM(S, \{i,j\})| = 4$.

Pre-processing: We perform a well established pre-processing step that ensures that for any optimal output matrix H, $(0,0) \in GAM(H, \{i,j\})$ for all $i, j \in C(H)$ (See the work of Eskin et al.[13]). We assume that the input matrix has no duplicate rows, since such rows do not change the optimal solution. Note that such a matrix should have at most $q+1$ more taxa than characters, since otherwise, it does not have a solution to the IPPH problem.

3. ALGORITHM

At a high level, our algorithm has the same spirit as the algorithm of Song et al.[33]. The algorithm guesses characters that mutate more than once and removes them from the input matrix. It then solves the perfect phylogeny haplotyping problem on the remainder of the matrix. Finally, our algorithm adds the removed characters back and performs haplotyping on the full matrix. In this section, we will use the same assumption of Song et al.[33]: for any subset of characters of the input matrix, if a solution to perfect phylogeny haplotyping (PPH) exists, then it is

function `solveIPPH(input matrix` G`)`

(1) guess $Q = \{i \in C(G) | \exists e_1, e_2 \in E(T^*), e_1 \neq e_2, \mu(e_1) = \mu(e_2) = i\}$
(2) let M be matrix G restricted to characters $C(G) \setminus Q$
(3) let M' be the unique solution to perfect phylogeny haplotyping of M
(4) let H' be matrix M' defined on characters $C(G)$ s.t. $\forall h'_{2i-1}, h'_{2i} \in R(H')$ and corresponding taxa $g_i \in R(G).\forall \hat{c} \in Q.h'_{2i-1}[\hat{c}] = h'_{2i}[\hat{c}] = g_i[\hat{c}]$
(5) guess $\kappa = \{i \in C(G) \mid \exists j \in C(G),\ |GAM(t(T^*), \{i, j\})| = 4\}$
(6) guess $GAM(t(T^*), \kappa)$
(7) loop for $c \in C(H') \setminus \kappa$
  (a) $H' :=$ `processMatrix`(H', c)
(8) for all couples h'_{2i-1}, h'_{2i} in H'
  (a) resolve state 2 on h'_{2i-1}, h'_{2i} s.t.$GAM(\{h'_{2i-1}, h'_{2i}\}, \kappa) \subseteq GAM(t(T^*), \kappa)$

function `processMatrix(matrix` H'`, character` c`)`

(1) initialize the set $\Delta := \{c\}$
(2) while $|\Delta| > 0$ do
  (a) extract a character c from Δ
  (b) for all $(2, c)$-couples h_{2i-1}, h_{2i}
    i. for all $\hat{c} \in Q$ with $h_{2i-1}[\hat{c}] = 2(= h_{2i}[\hat{c}])$
    A. if $|IND(H', \{c, \hat{c}\})| = 3$ then $\mathcal{G}(c, \hat{c}) = IND(H', \{c, \hat{c}\})$
      else guess three gametes $\mathcal{G}(c, \hat{c})$
    B. resolve state 2 in $\hat{c}$ on h_{2i-1}, h_{2i} based on $\mathcal{G}(c, \hat{c})$
    C. $\Delta := \Delta \cup \{c' \notin \kappa | h_{2i-1}[c'] \neq h_{2i}[c']\}$
      i.e. add to Δ characters $c' \notin \kappa$ for which the current couple is also a $(2, c')$-couple
  (c) if $\exists \hat{c} \in Q$. s.t.$(c, \hat{c})$ conflict or
    for the set $H2$ of all $(2, c)$-couples $GAM(H2, \kappa) \nsubseteq GAM(t(T^*), \kappa)$ then
    return `no solutions`
  (d) remove c from $C(H')$
(3) return H'

Fig. 1. Algorithm to solve IPPH

unique. In Section 4, we show that the number of PPH solutions is polynomial with high probability.

Throughout the paper, we fix an arbitrary optimal phylogeny T^*, which we will use as a reference for expository purposes. Let $t(T^*)$ be the set of all taxa (Steiner vertices included) in T^*. Let Q be the set of characters that mutate more than once in T^*. Since $|Q| \leq q$, we can find Q by brute force in time $O(\sum_{i=1}^{q} \binom{m}{i}) = O(qm^q)$.

After finding Q, we can remove the characters to obtain matrix M with character set $C(G) \setminus Q$. We now use a prior method to solve the perfect phylogeny haplotyping (PPH) problem on M in time $O(nm)$[12]. Let M' be the unique solution to the PPH problem. The solution matrix M' contains $2n$ taxa and $m - |Q|$ characters. We can now add the characters Q to matrix M' to obtain H'. For all $j \in Q$ and $h'_{2i-1}, h'_{2i} \in H'$ and $g_i \in G$, taxa $h'_{2i-1}[j] = h'_{2i}[j] = g_i[j]$. Note that matrix H' contains state 2 only on the characters Q (see Fig 3(b) for an example).

Definition 3.1. A pair of taxa $h'_{2i-1}, h'_{2i} \in R(H')$ is defined as a *couple*. For any $c \in C(H')$, if $h'_{2i-1}[c] = 2(= h'_{2i}[c])$ or $h'_{2i-1}[c] \neq h'_{2i}[c]$ then h'_{2i-1}, h'_{2i} is a $(2, c)$-couple.

Note that if c contains both 0 and 1 in a couple of H', then the couple contained state 2

at character c in the input (before perfect phylogeny haplotyping). Let $\kappa = \{i \in C(G) | \exists j \in C(G), |GAM(t(T^*), \{i,j\})| = 4\}$ be the set of characters that conflict with some other character in $t(T^*)$. Note that $Q \subseteq \kappa$. To complete the description and analysis of the algorithm we borrow the following definition from prior work[13].

Definition 3.2. We say that a (unordered) couple r_1, r_2 induces (x, y) at characters (c, c') if $r_i[c] = x, r_i[c'] = y$ or $r_1[c] = r_2[c] = 2, r_1[c'] = r_2[c'] = y$ or $r_1[c] = r_2[c] = x, r_1[c'] = r_2[c'] = 2$. We define $IND(H', \{c, c'\})$ to denote the set of *gametes induced* by the couples of H' at c, c'.

The goal now is to convert the {0, 1, 2} matrix H' to a {0, 1} matrix H such that the following **correctness conditions** are satisfied:

(1) for every $(2, c)$-couple in H' one of the two taxa should contain state '1' and the other '0' on character c in H
(2) $GAM(H, \kappa) \subseteq GAM(t(T^*), \kappa)$, i.e. the set of gametes on κ in H is a subset of the set of gametes on κ in T^*.
(3) $(|GAM(H, \{c, c'\})| = 4) \Longrightarrow c, c' \in \kappa$, i.e. a pair of characters share four gametes in H only if they are both in κ.

In matrix H', if a couple contains state 2 at character c, then replacing it with state 0 on one taxa and 1 on the other is informally referred to as a *resolution*. The algorithm to solve IPPH is summarized in Figure 1. We now go into the details of the steps. The following lemma shows that Steps 5 and 6 of function `solveIPPH` can be implemented efficiently:

Lemma 3.1. *Sets κ and $GAM(t(T^*), \kappa)$ can be found in time $m^{2q}q^{O(q)} + O(nm)$.*

Proof. We can easily identify κ by brute force in time $O(m^{|\kappa|})$. Since we do not know $|\kappa|$, this step can take time $O(2^m)$. We can however do better by performing such an enumeration over phylogenies as illustrated in Figure 2. First we construct the unique perfect phylogeny T for the matrix H' restricted to $C(G) \setminus Q$ which contains $m - |Q| + 1$ vertices in time $O(nm)$[12]. Note that contracting edges $e \in T^*$ with $\mu(e) \in Q$ results in tree T (see Figure 2(b)). We will begin with T and add the edges $e, \mu(e) \in Q$ to obtain T^* as follows. Since we know Q already, we can identify the set of $|Q| + q$ edges (labels μ) in time $O(\binom{|Q|+q}{q}) = O(\binom{2q}{q}) = O(4^q)$. There are $m - |Q| + 1$ locations to add an edge that mutates a character in Q. All possible edge assignments can be enumerated in time $O((m - |Q| + 1)^{|Q|+q}) = O(m^{2q})$. Each enumeration assigns a set of edges (multi-set on characters) Q_v to each vertex v of T. We now enumerate all possible rooted trees T_v with edge labels in Q_v for all vertices $v \in T$ in time $O((|Q| + q)^{|Q|+q}) = O((2q)^{2q})$. Since the mutations in $C(G) \setminus Q$ are already fixed by the perfect phylogeny, the states in all vertices on characters $C(G) \setminus Q$ are known. For every root of T_v, we guess the states in all Q characters in time $O(2^{q^2})$, which can be improved to $O(q^q)$ (since this is equivalent to enumerating all tree structures with edges mutating Q). Note that since we guessed the states at the root of T_v, we know the states on all characters for all vertices in T_v (for all v). Further, for any two roots of $T_v, T_{v'}$, the path that connects them is given by the path connecting v, v' in the perfect phylogeny T (see Figure 2(c)). Therefore, we can identify the edges that lie between two mutations of a character in Q. We can now find κ since for all $i \in \kappa \setminus Q$, there exists $j \in Q$ and $e_1, e_2, e_3 \in T^*$ such that $\mu(e_1) = \mu(e_3) = j$, $\mu(e_2) = i$ and e_2 lies in the path connecting e_1, e_3.

Since we know states in all characters of T_v and T'_v, we know the states in all the characters for every vertex in the path connecting them as well. We now consider the set of all vertices V_κ that lie in the path connecting two mutations of the same character in T. It is easy to see that $GAM(V_\kappa, \kappa) = GAM(t(T^*), \kappa)$. We can therefore identify κ and $GAM(t(T^*), \kappa)$ in time $m^{2q}q^{O(q)} + O(nm)$. □

Figure 3 illustrates how the algorithm performs haplotyping given κ and $GAM(t(T^*), \kappa)$. For reference Figures 3(a) and 3(b) represent input matrix G and matrix H' as found in Step 4 of `solveIPPH`. Function `solveIPPH` selects $c = 2$ (Step 7). Function `processMatrix` determines $\mathcal{G}(2, 6) = IND(H', \{2, 6\}) = \{(0, 0), (0, 1), (1, 1)\}$ and guesses $\mathcal{G}(2, 7) = \{(0, 0), (0, 1), (1, 1)\}$ (Step 2(b)iA). Based on these two sets of three gametes, the $(2, 2)$-couples, rows 9 to 12, are resolved (Step 2(b)iB), Fig 3(c). Character 2 is removed (Step 2d). No-

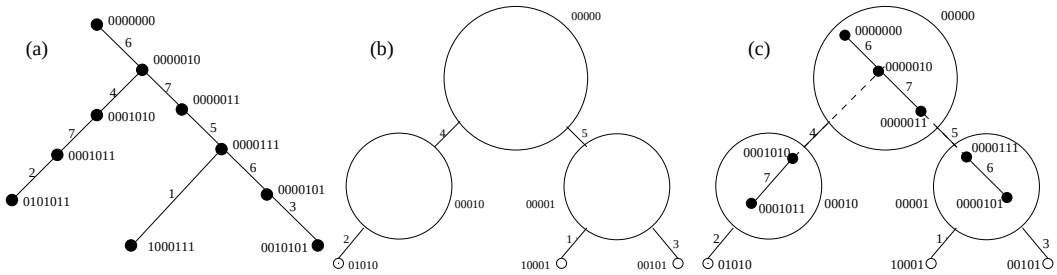

Fig. 2. Algorithm to determine κ and $GAM(t(T^*), \kappa)$ efficiently. (a) optimal phylogeny T^* with $Q = \{6, 7\}, \kappa = \{4, 5, 6, 7\}$(b) perfect phylogeny T on characters $C(G) \setminus Q$ (c) the four edges mutating characters 6 and 7 are assigned to three vertices of T; rooted trees T_v are constructed on the assigned edges; states on the three roots $0000000, 0001010, 0000111$ determine edges $\{4, 5\}$ that lie between two mutations of $6, 7$. Filled vertices form V_κ.

tice that a character c is removed only when all the $(2, c)$-couples are completely resolved (rows 9 to 12). Function `solveIPPH` then selects $c = 3$ (Step 7), Fig 3(d). Function `processMatrix` guesses $\mathcal{G}(3, 6) = \{(0, 0), (1, 0), (0, 1)\}$ and determines $\mathcal{G}(3, 7) = IND(H', \{3, 7\}) = \{(0, 0), (0, 1), (1, 1)\}$ (Step 2(b)iA), and using them $(2, 3)$-couple, rows 7 and 8, are resolved(Step 2(b)iB), Fig 3(e). Since the pair of rows (7, 8) is also a $(2, 1)$-couple character 1 is added to Δ (Step 2(b)iC). Character 3 is removed (Step 2d) and $c = 1$ is extracted from Δ (Step 2a), Fig 3(f). Since there are no $(2, 1)$-couples, character 1 is removed (Step 2d), Fig 3(g). This exhausts all $c \in C(H') \setminus \kappa$. Function `solveIPPH` then resolves state 2 in the first couple resulting in $0000, 0010$ which are both in $GAM(t(T^*), \kappa)$ (Step 8a), Fig 3(h). Since the next couple has no state 2, we resolve the third couple which results in $0101, 0111$ (Step 8a), completing the algorithm, Fig 3(i). We now prove the main lemma that bounds the running time of our algorithm.

Theorem 3.1. *If* `penalty`$(T^*) \leq q$*, then the algorithm described in Figure 1 returns a solution matrix H that obeys all correctness conditions in time $nm^{O(q)}$.*

Proof. To prove this theorem, we first need three simple lemmas.

Lemma 3.2. *If $c_1 \in \kappa$ and $c_2 \in C(H') \setminus \kappa$, then c_1, c_2 share exactly three gametes in $t(T^*)$.*

Proof. Every pair of characters share at least three gametes in T^*. Characters, c_1, c_2 cannot share four gametes since $c_2 \notin \kappa$. □

Lemma 3.3. *For a pair of characters $c \notin \kappa, \hat{c} \in Q$, given a set of three gametes $\mathcal{G}(c, \hat{c})$, there exists a unique resolution of state 2 in character $\hat{c}$ for any $(2, c)$-couple s.t. for resulting matrix H', $GAM(H', \{c, \hat{c}\}) \subseteq \mathcal{G}(c, \hat{c})$*

Proof. Let $r_{2i-1}, r_{2i} \in R(H')$ be a $(2, c)$-couple. By definition, $r_{2i-1}[c] \neq r_{2i}[c]$. If $r_{2i-1}[\hat{c}] = r_{2i}[\hat{c}] = 2$ then the resolution will either create $GAM(\{r_{2i-1}, r_{2i}\}, \{c, \hat{c}\}) = \{(0, 0), (1, 1)\}$ or $\{(0, 1), (1, 0)\}$. Only one of the two can be contained in set $\mathcal{G}(c, \hat{c})$ established between c and $\hat{c}$. □

Lemma 3.4. *In matrix H', for $c \notin \kappa, \hat{c} \in Q$ if every $(2, c)$-couple is in $\{0, 1\}^m$ then $GAM(H, \{c, \hat{c}\})$ is fixed where H is any matrix obtained from H' by resolution of 2s.*

Proof. For any couple in H' if $h'_{2i-1}[\hat{c}] = h'_{2i}[\hat{c}] = 2$ then according to the condition of the lemma, $h'_{2i-1}[c] = h'_{2i-1}[c] = s$. Therefore, for any couple h_{2i-1}, h_{2i} in H obtained by resolving state 2 on $\hat{c}$ will have the property that $GAM(\{h_{2i-1}, h_{2i}\}, \{c, \hat{c}\}) = \{(s, 0), (s, 1)\}$. □

Corollary 3.1. *Although $\hat{c}$ can contain state 2 in matrix H', Step 2c of function* `processMatrix` *in Figure 1 can test if $c, \hat{c}$ are conflicting. Furthermore, if $c, \hat{c}$ do not conflict at Step 2c, then the final matrix obtained by the algorithm will not have a conflict between $c, \hat{c}$.*

We now return to the proof of the main theorem. Step 1 requires at most $q\binom{m}{q}$ enumerations. Lemma 3.1 shows that given the correct Q, both κ and $GAM(t(T^*), \kappa)$ can be found in time $m^{2q}q^{O(q)} +$

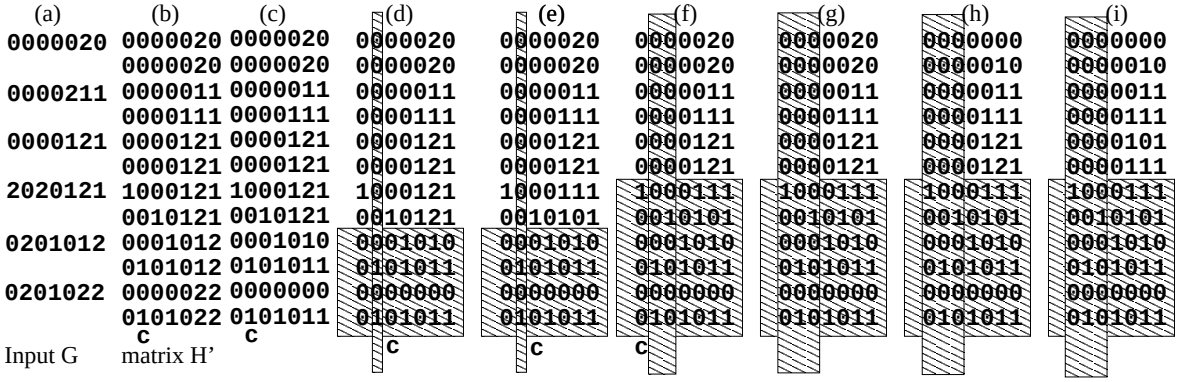

Fig. 3. Given $Q = \{6, 7\}$, $\kappa = \{4, 5, 6, 7\}$, $GAM(t(T^*), \kappa) = \{0000, 0010, 0011, 1011, 1010, 0111, 0101\}$, the algorithm chooses $c = 2, 3, 1$. Based on a set of three gametes, it resolves states 2 in $\hat{c} \in Q$, in all $(2, c)$-couples. Shaded regions represent deleted (or ignored) characters and couples completely resolved by the algorithm. After exhausting all characters $c \in C(G) \setminus \kappa$, the algorithm considers the remaining couples and iteratively resolves states 2 s.t. the couples are in $GAM(t(T^*), \kappa)$.

$O(nm)$. We now bound the run time for function `processMatrix`.

Lemma 3.5. *Total run-time for all calls to function* ***processMatrix*** *is* $O(2^q nm^2)$.

Proof. First notice that once a character c is removed from Δ at Step 2a of function `processMatrix`, it is never added back throughout the execution of the algorithm. This is because, function `processMatrix` resolves all the states 2 present in $(2, c)$-couples and subsequently the character is deleted (or ignored) for the rest of the algorithm in Step 2d. This bounds the number of times Step 2a is executed throughout the algorithm as $O(m)$. To prove the lemma, we now bound the time for executing Steps 2a through 2d as $O(2^q nm)$.

The number of $(2, c)$-couples is $O(n)$ and therefore Step 2b loops for $O(n)$ times. The cardinality of Q is at most q and therefore the loop in Step 2(b)i executes $O(|Q|) = O(q)$ times. The time bound for guessing the set of gametes $\mathcal{G}(c, \hat{c})$ shared between c and $\hat{c}$ is the hardest to analyze.

Consider any one call to function `processMatrix`. Let c_i's represent the characters added into Δ during the execution of a call. We show that if $\mathcal{G}(c_j, \hat{c})$ is guessed at Step 2(b)iA then for all future c_k encountered during the execution of Step 2(b)iA, $|IND(H', \{c_k, \hat{c}\})| = 3$. When the algorithm guessed $\mathcal{G}(c_j, \hat{c})$, by definition of $(2, c_j)$-couples, $h_{2i-1}[c_j] \neq h_{2i}[c_j]$ and by the condition on Step 2(b)i, $h_{2i-1}[\hat{c}] = h_{2i}[\hat{c}] = 2$. Also $h_{2i-1}[c_k] = h_{2i}[c_k] = x$ since otherwise $|IND(H', \{c_k, \hat{c}\})| = 3$ (c_k and $\hat{c}$ cannot be identical characters). Let c_l be the character extracted from Δ and used in the loop of Step 2b during which c_k was added into Δ. This implies the existence of a couple h'_{2i-1}, h'_{2i} which is both a $(2, c_l)$-couple and a $(2, c_k)$-couple. However $h'_{2i-1}[\hat{c}] = h'_{2i}[\hat{c}] = y$. Again, otherwise $|IND(H', \{c_k, \hat{c}\})| = 3$. Therefore matrix H' induces gametes $(x, 0), (x, 1), (0, y), (1, y)$ on characters $(c_k, \hat{c})$. For all values of $x, y \in \{0, 1\}$, this results in three induced gametes. The above proof therefore shows that the `if` condition in Step 2(b)iA fails at most q times for any function call to `processMatrix`. Therefore the probability of all guesses performed at Step 2(b)iA being correct for any single call to `processMatrix` is at least 2^{-q}. Equivalently, we suffer a multiplicative factor of 2^q in the run-time because of this step.

The check performed at Step 2c takes time $O(nm)$. Assuming $q < m$, the total running time for all calls to `processMatrix` combined is $O(2^q nm^2)$. □

Using Lemma 3.2, we know that c shares exactly three gametes with all characters $\hat{c} \in Q$ in $t(T^*)$. For character c, Step 2(b)iA (guesses) iterates over the set of all possible gametes shared between $c, \hat{c}$. Given the set of three gametes, Lemma 3.3 shows that there is a unique resolution of states 2 in the $(2, c)$-couples and this is performed in Step 2(b)iB. Correctness condition 1 holds by the definition of resolution. Step 2c of function `processMatrix` checks for conditions 2 and 3. Using Corollary 3.1, we know that c and $\hat{c}$ cannot conflict because of the reso-

lution of any of the remaining 2s (ensures correctness condition 3). Finally, there can be no 2s in character c (since $c \notin Q$) or in any of the $(2, c)$-couples since it was just resolved. Step 8 of function `solveIPPH` iterates n times. At each iteration, it performs a brute-force step of computing all possible ways of resolving the 2s. Since only the characters in Q can contain state 2 at this point, Step 8a takes $O(m2^q)$ time. This step also checks if the resulting gametes on characters in κ are in the predicted set $GAM(t(T^*), \kappa)$(ensures correctness condition 3). This shows that any matrix found by the algorithm obeys the correctness conditions and the running time is $nm^{O(q)}$. Finally, we know that there exists a set of three gametes $\mathcal{G}(c, Q)$ (as defined by $t(T^*)$) s.t. resolving based on $\mathcal{G}$ will ensure that $c, \hat{c}$ do not conflict and $GAM(H', \kappa) \subseteq GAM(t(T^*), \kappa)$. Using these two observations, we know that if $\texttt{penalty}(T^*) \leq q$, then the algorithm finds matrix H' that obeys the correctness conditions in the stated time.

We now prove the correctness of our algorithm:

Theorem 3.2. *Any solution matrix H obeying all the correctness conditions is optimal.*

Proof. The proof is constructive and demonstrates the procedure to construct a q-near-perfect phylogeny for H. The phylogeny along with correctness condition 1 guarantees that the returned matrix H is an optimal solution.

Matrix H satisfies the following two properties: if a pair of characters c, c' conflict in H, then $c, c' \in \kappa$ (third correctness condition); a q-near-perfect phylogeny can be constructed on $GAM(H, \kappa)$ (since $GAM(H, \kappa) \subseteq GAM(t(T^*), \kappa)$, second correctness condition). It can be shown that a q-near-perfect phylogeny can be constructed for any matrix that satisfies the above two properties (see Section 7 of Gusfield and Bansal [21]). Such a phylogeny is obtained by constructing the q-near-perfect phylogeny on on $GAM(H, \kappa)$, contracting the phylogeny to a vertex and constructing a perfect phylogeny on the remaining characters. □

Theorem 3.3. *The algorithm of Figure 1 returns an optimal solution H to the IPPH problem in time $nm^{O(q)}$.*

Proof. The proof follows directly from Theorems 3.1 and 3.2. □

4. SOLUTIONS TO PPH

We assumed in the preceding sections, following prior work[33], that the perfect phylogeny stage of the algorithm will find a unique solution, but this assumption does not necessarily hold. To guarantee optimality, the algorithm would need to enumerate over all solutions to the PPH sub-problem, increasing run-time proportionally. Prior work showed that the number of PPH solutions is at most 2^k, where k is the number of characters of G that do not contain the homozygous minor allele[13, 19]. In the worst case, this could be as large as m and therefore the number of PPH solutions can be 2^m. This however should not occur in practice since the underlying population typically follows a random mating model.

Hardy-Weinberg equilibrium states that the two haplotypes of any given individual are selected independently of one another at random. For any fixed character, let p be the minor allele frequency and $(1-p)$ be the major allele frequency. Consider G, an $n \times m$ input genotype matrix to the PPH problem. The probability under Hardy-Weinberg that none of the n taxa contain the homozygous minor allele is $(1-p^2)^n$. This probability could be large for very small values of p.

It is reasonable to assume that the value of $p > c$ for some constant c since otherwise SNPs cannot be detected. In this case, with high probability in n (at least $1-(1-c^2)^n$), a specific character contains the homozygous minor allele. Therefore in expectation the number of characters without a homozygous minor allele is at most $m(1-c^2)^n$. This expectation, exponentially tends to zero with n.

We now consider a more general setting when the value of p is assumed to be uniformly distributed in $[0, 0.5]$. Now, the probability that a specific character does not contain the homozygous minor allele is:

$$2\int_0^{0.5}(1-p^2)^n dp$$

$$= 2\int_0^{1/\sqrt{n}}(1-p^2)^n dp + 2\int_{1/\sqrt{n}}^{0.5}(1-p^2)^n dp$$

$$\leq \frac{2}{\sqrt{n}} + 2\int_{1/\sqrt{n}}^{0.5}(1-p^2)^n dp$$

$$\leq \frac{2}{\sqrt{n}} + 2\sum_{i=1}^{\infty}\int_{i/\sqrt{n}}^{(i+1)/\sqrt{n}}(1-p^2)^n dp$$

$$\leq \frac{2}{\sqrt{n}} + \frac{2}{\sqrt{n}}\sum_{i=1}^{\infty}(1-\frac{i^2}{n})^n$$

$$\leq \frac{2}{\sqrt{n}} + \frac{2}{\sqrt{n}}\sum_{i=1}^{\infty}e^{-i^2} \leq \frac{4}{\sqrt{n}}$$

Now, if $n = \Omega(m^2)$, then using Chernoff bounds we can show that with high probability the number of characters that lack a homozygous minor allele is bounded by $2\log m$ and therefore the number of solutions to the PPH problem is bounded by m^2.

This discussion answers the question raised by Gusfield on the theoretical estimate for the number of PPH solutions to expect from a coalescent model[19]. Furthermore, since PPH is always performed on SNP blocks with low diversity, it is not unreasonable to assume $n >> m$. Since the number of solutions returned by PPH is $O(m^2)$, the IPPH algorithm described above with high probability just suffers $O(m^2)$ overhead for finding the optimal extension of each PPH solution.

5. EMPIRICAL VALIDATION

We demonstrate and validate our algorithm by comparing with leading phasing methods using a collection of large-scale genotype data of known phase from a high resolution scan of human chromosome 21[29]. The study identified common single nucleotide polymorphisms (SNPs) and typed them on 20 sequences through a method that directly identifies haplotypes rather than genotypes. The SNPs were partitioned into 4135 haplotype blocks by the authors of that study using a diversity test. We extracted haplotypes from the 'haplotype pattern' file provided in the supplementary material, which identifies distinct haplotypes in each block and provides some inference of missing data when it can be done unambiguously. We ignored haplotypes which still had significant missing data, replacing them with haplotypes randomly selected from the multinomial distribution of fully-resolved haplotypes in the corresponding block in order to maintain 20 chromosomes per block. The haplotype blocks were divided into four sets based on their imperfectness($q = 0, 1, 2, 3+$) using a prior method[31]. A large majority of blocks (98%) are 0, 1, or 2 near-perfect. We do not solve optimally for the 2% with imperfectness 3 or greater because the run-time for optimal solutions would be prohibitive. Such blocks can be solved non-optimally in practice by subdividing into smaller blocks of lower imperfectness, but such data would not be comparable to those solved optimally and is therefore omitted from our analysis. We then randomly paired haplotypes to produce 10 diploid genotypes per block as our final test set. Figure 5 shows that the length of the blocks is related to q, the imperfectness. The tails are heavier for larger values of q as expected. For instance, this shows that a significant fraction of the $q = 2$ blocks have large number of characters when compared with $q = 1, 0$.

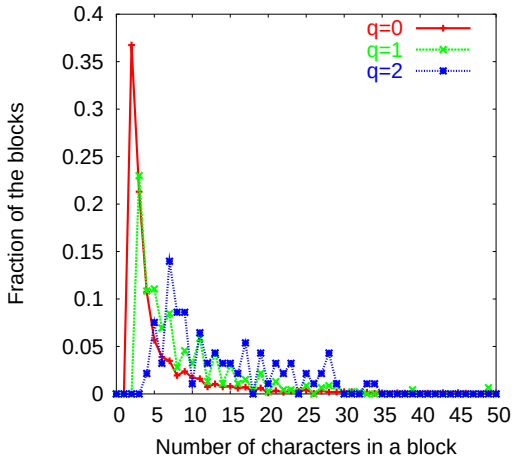

Fig. 5. Distribution of the fraction of blocks as a function of the number of characters in the block. Data for more than 50 characters not shown.

We compared our method with two popular phasing packages. We ran the `haplotyper`[28] package, which uses an expectation-maximization heuristic, with 20 rounds (the recommended value). We also ran the `PHASE`[35] package, which employs a Markov Chain Monte Carlo method. Although

			Error rate			Total Run Time(secs)		
Perfectness	#Blocks	#SNPs	PHASE	haplotyper	our alg	PHASE	haplotyper	our alg
0	3497	20816	0.11	0.11	0.17	3521.33	337.18	17.21
1	461	4211	0.53	0.47	0.35	805.62	80.62	8.77
2	93	1266	0.83	0.68	0.55	268.02	59.28	1111.18

Fig. 4. Empirical results on Chromosome 21. Blocks with 0, 1, 2 near-perfectness (accounting for 4051 out of 4135 blocks) were analyzed separately using the three algorithms. Error rate is the number of switch errors divided by the number of blocks. Total run time is the sum over all blocks.

PHASE can make use of additional information such as SNP positions, we provided it only the genotypes as input.

For our own method, with any given q, we enumerated all possible phylogenies that are at most q-near-perfect. Note that this does not guarantee finding all possible output matrices that are consistent with a q-near perfect phylogeny. Where multiple solutions were obtained, we selected the one that minimized the entropy of the haplotype frequencies. For the PPH sub-problem we implemented a fast prior algorithm[13]. We note that for simplicity, we do not implement the algorithm exactly as described. Rather, function processMatrix does not add characters into Δ to be processed iteratively. The implementation however always returns a haplotype output matrix and we report the switch errors[33] for the output.

Table of Figure 4 summarizes the test results[a]. All methods provide comparable accuracy when blocks are perfect. We attribute the slightly worse performance of our method on perfect input to the fact that we do not use any procedure to select a maximum-likelihood output matrix among all perfect matrices, as is done for prior perfect phylogeny methods[13]. All three methods degrade in accuracy with increase in imperfectness. Our method, however, scales much better with imperfectness than do the other two, clearly outperforming them on 1-near-perfect and 2-near-perfect inputs. Imperfectness can result from recurrent mutations, recombinations or incorrect SNP inferences and we attribute our method's superior performance on imperfect data to the fact that it explicitly handles one of these factors while the others suffer from all three. Our method is extremely fast for $q = 0$, where it reduces to the perfect phylogeny algorithm of Eskin et al.[13] and also substantially outperforms the competing methods for $q = 1$. Our method's run time rapidly increases with larger q, though, as expected from the theoretical bounds.

6. DISCUSSION AND CONCLUSIONS

We have developed a theory for reconstructing phylogenetic trees directly from genotypes that is optimal in the number of mutations. As an immediate application, we solve the general IPPH problem. We demonstrate practical results that show great promise in accurately inferring phase from real data sets. Our results suggest that imperfect phylogeny approaches can lead to significant improvements in accuracy over other leading phasing methods. Run time, while very fast for perfect and almost perfect data, remains an obstacle for even modest q; this observation suggests a need for further research in improving theoretical and practical bounds for general q. Our new method has several immediate applications in computational genetics:

- Phasing: At present, the method is competitive with the most widely used tools in accuracy and, with optimizations, should become competitive in run-time for larger q. Both PHASE and haplotyper return confidence scores on its results, which might allow a slower, high-accuracy method such as ours to function as a fall-back for regions that those methods cannot infer with confidence.
- Phylogeny Construction: Our run times are competitive with typical times to be expected from other optimal phylogeny reconstruction algorithms, even when the input consists of haplotypes. Our approach may thus be considered

[a] Haplotyper crashes on 18 blocks which were ignored in the calculation of its accuracy.

preferable to the standard practice of inferring haplotypes then fitting them to phylogenies.

- Haplotype Blocks Inference: Our method could serve as an improved means of identifying recombination-free haplotype blocks for purposes of association study design by more accurately distinguishing recurrent mutation from recombination. The blocks might thus be useful in improving statistical power in haplotype-based association testing.

Future empirical studies, enabled by our method, will be needed to better establish the nature of imperfectness in real genotype data and the degree to which better handling of recurrent mutations will be of use in practice.

Acknowledgments

This work is supported in part by NSF ITR grant CCR-0122581 (The ALADDIN project) and NSF grant CCF-043075.

References

1. R. Agarwala and D. Fernandez-Baca. A Polynomial-Time Algorithm for the Perfect Phylogeny Problem when the Number of Character States is Fixed. *SIAM Journal on Computing*, 23 (1994).
2. V. Bafna, D. Gusfield, G. Lancia, and S. Yooseph. Haplotyping as perfect phylogeny: A direct approach. *Journal of Computational Biology*, 10 (2003).
3. V. Bafna, D. Gusfield, G. Hannenhalli, and S. Yooseph. A note on efficient computation of haplotypes via perfect phylogeny. *Journal of Computational Biology*, 11 (2004).
4. G. E. Blelloch, K. Dhamdhere, E. Halperin, R. Ravi, R. Schwartz and S. Sridhar. Fixed Parameter Tractability of Binary Near-Perfect Phylogenetic Tree Reconstruction. To appear in proc *International Colloquium on Automata, Languages and Programming* (2006).
5. H. Bodlaender, M. Fellows and T. Warnow. Two Strikes Against Perfect Phylogeny. In proc *International Colloquium on Automata, Languages and Programming* (1992).
6. H. Bodlaender, M. Fellows, M. Hallett, H. Wareham and T. Warnow. The Hardness of Perfect Phylogeny, Feasible Register Assignment and Other Problems on Thin Colored Graphs. *Theoretical Computer Science* (2000).
7. M. Bonet, M. Steel, T. Warnow and S. Yooseph. Better Methods for Solving Parsimony and Compatibility. *Journal of Computational Biology*, 5(3) (1992).
8. A. G. Clark. Inference of haplotypes form PCR-amplified samples of diploid populations. *Molecular Biology and Evolution* 7:1111-122 (1990).
9. W. H. Day and D. Sankoff. Computational Complexity of Inferring Phylogenies by Compatibility. *Systematic Zoology* (1986).
10. P. Damaschke. Parameterized Enumeration, Transversals, and Imperfect Phylogeny Reconstruction. In proc *International Workshop on Parameterized and Exact Computation* (2004).
11. R.G. Downey and M. R. Fellows. Parameterized Complexity. *Monographs in Computer Science* (1999).
12. Z. Ding, V. Filkov and D. Gusfield. A Linear Time Algorithm For Perfect Phylogeny Haplotyping. In proc *Research in Computational Molecular Biology* (2005).
13. E. Eskin, E. Halperin and R. M. Karp. Efficient Reconstruction of Haplotype Structure via Perfect Phylogeny. *Journal of Bioinformatics and Computational Biology* (2003).
14. D. Fernandez-Baca and J. Lagergren. A Polynomial-Time Algorithm for Near-Perfect Phylogeny. *SIAM Journal on Computing*, 32 (2003).
15. L. R. Foulds and R. L. Graham. The Steiner problem in Phylogeny is NP-complete. *Advances in Applied Mathematics* (3) (1982).
16. G. Ganapathy, V. Ramachandran and T. Warnow. Better Hill-Climbing Searches for Parsimony. In proc *Workshop on Algorithms in Bioinformatics* (2003).
17. D. Gusfield. Efficient Algorithms for Inferring Evolutionary Trees. In: *Networks*, 21 (1991).
18. D. Gusfield. Algorithms on Strings, Trees and Sequences. *Cambridge University Press* (1999).
19. D. Gusfield. Haplotyping as a Perfect Phylogeny: Conceptual Framework and Efficient Solutions. In proc *Research in Computational Molecular Biology* (2002).
20. D. Gusfield. An Overview of Combinatorial Methods for Haplotype Inference. *Lecture Notes in Computer Science*, 2983 (2004).
21. D. Gusfield and V. Bansal. A Fundamental Decomposition Theory for Phylogenetic Networks and Incompatible Characters. In proc *Research in Computational Molecular Biology* (2005).
22. D. Gusfield, S. Eddhu and C. Langley. Efficient Reconstruction of Phylogenetic Networks with Constrained Recombination. In proc *IEEE Computer Society Bioinformatics Conference* (2003).
23. E. Halperin and E. Eskin. Haplotype Reconstruction from Genotype Data using Imperfect Phylogeny. *Bioinformatics* (2004).
24. D. A. Hinds, L. L. Stuve, G. B. Nilsen, E. Halperin, E. Eskin, D. G. Ballinger, K. A. Frazer, D. R. Cox. Whole Genome Patterns of Common DNA Variation in Three Human Populations. `www.perlegen.com`. *Science* (2005).
25. D. Huson, T. Klopper, P. J. Lockhart, M. A. Steel. Reconstruction of Reticulate Networks from Gene

Trees. In proc *Research in Computational Molecular Biology* (2005).
26. The International HapMap Consortium. The International HapMap Project. `www.hapmap.org`. *Nature* 426 (2003)
27. S. Kannan and T. Warnow. A Fast Algorithm for the Computation and Enumeration of Perfect Phylogenies. *SIAM Journal on Computing*, 26 (1997).
28. T. Niu, Z. S. Qin, X. Xu, and J. Liu. Bayesian Haplotype Inference for Multiple Linked Single Nucleotide Polymorphisms. *American Journal of Human Genetics* (2002)
29. N. Patil et al. Blocks of Limited Haplotype Diversity Revealed by High-Resolution Scanning of Human Chromosome 21. In *Science* (2001).
30. H. J. Promel and A. Steger. The Steiner Tree Problem: A Tour Through Graphs Algorithms and Complexity. *Vieweg Verlag* (2002).
31. S. Sridhar, K. Dhamdhere, G. E. Blelloch, E. Halperin, R. Ravi and R. Schwartz. Simple Reconstruction of Binary Near-Perfect Phylogenetic Trees. In proc *International Workshop on Bioinformatics Research and Applications* (2006).
32. C. Semple and M. Steel. Phylogenetics. *Oxford University Press* (2003).
33. Y. Song, Y. Wu and D. Gusfield. Algorithms for Imperfect Phylogeny Haplotyping with a Single Homoplasy or Recombination Event. In proc *Workshop on Algorithms in Bioinformatics* (2005).
34. M. A. Steel. The Complexity of Reconstructing Trees from Qualitative Characters and Subtrees. *Journal of Classification*, 9 (1992).
35. M. Stephens, N. Smith and P. Donnelly. A new statistical method for haplotype reconstruction from population data. *American Journal of Human Genetics* (2001).