Acta Mathematicae Applicatae Sinica, English Series
Vol. 25, No. 3 (2009) 475–486
DOI: 10.1007/s10255-008-8821-3
http://www.ApplMath.com.cn

Acta Mathematica Applicatae Sinica, English Series

Efficient Haplotype Inference Algorithms in One Whole Genome Scan for Pedigree Data with Non-genotyped Founders

Yongxi Cheng, Hadi Sabaa, Zhipeng Cai, Randy Goebel, Guohui Lin*

Department of Computing Science, University of Alberta. Edmonton, Alberta T6G 2E8, Canada
(E-mail: yongxi@cs.ualberta.ca, sabaa@cs.ualberta.ca, zhipeng@cs.ualberta.ca, goebel@cs.ualberta.ca, *ghlin@cs.ualberta.ca)

Abstract An efficient rule-based algorithm is presented for haplotype inference from general pedigree genotype data, with the assumption of no recombination. This algorithm generalizes previous algorithms to handle the cases where some pedigree founders are not genotyped, provided that for each nuclear family at least one parent is genotyped and each non-genotyped founder appears in exactly one nuclear family. The importance of this generalization lies in that such cases frequently happen in real data, because some founders may have passed away and their genotype data can no longer be collected. The algorithm runs in $O(m^3n^3)$ time, where m is the number of single nucleotide polymorphism (SNP) loci under consideration and n is the number of genotyped members in the pedigree. This zero-recombination haplotyping algorithm is extended to a maximum parsimoniously haplotyping algorithm in one whole genome scan to minimize the total number of breakpoint sites, or equivalently, the number of maximal zero-recombination chromosomal regions. We show that such a whole genome scan haplotyping algorithm can be implemented in $O(m^3n^3)$ time in a novel incremental fashion, here m denotes the total number of SNP loci along the chromosome.

Keywords Haplotype inference; efficient algorithm; pedigree; whole genome scan
2000 MR Subject Classification 68R05; 92D10

1 Introduction

Genetic fine-mapping for complex diseases (such as cancers, diabetes, mental illness etc.) is currently a great challenge for geneticists. With the availability of millions of *single nucleotide polymorphisms* (SNPs), research sees new potentials via using these common variants — the *genome-wide association studies* (GWAS) under the "*common disease – common variant*" (CDCV) hypothesis. In the past a few years, GWAS has achieved a great deal of successes[1]. Nevertheless, the unphased genotype data is recognized as a fundamental bottleneck in general association studies, in particular for rare diseases. The ideal case is to build the dense SNP haplotype (phased genotype) map from the unphased genotype data, to provide finer and deterministic haplotype allelic information for the mapping of important disease-susceptible genes. However, due to the diploid structure of the human (and many other organisms) genome, it is the genotype data instead of haplotype data that are collected in practice, due to cost considerations. Therefore, efficient and accurate computational methods for the inference of haplotypes from genotype data are highly demanded.

Manuscript received Feberuary 27, 2009. Revised April 16, 2009.
This research is supported in part by AARI, AICML, ALIDF, iCORE, and NSERC.
*Corresponding author.

There is a rich and growing literature on the inference of haplotypes from genotype data, which is also commonly referred to as *phasing*, or *haplotyping*. Some works focus on unrelated individuals, or population datasets[4−6,11,13−15,17]; Research on related individuals, or pedigree datasets, include those based on either exact-likelihood computations[3,8,16,18], approximate-likelihood computations[7,16,18], or rule-based strategies[2,9,10,12,19]. The likelihood-based methods usually cannot handle large (in many cases, moderately large) datasets, owing to the extensive computations required. Additional information and assumptions, such as marker recombination rates and Hardy-Weinberg equilibrium, are generally required to calculate the likelihood. The rule-based methods are *ad hoc*, but they rely on fewer assumptions and generally run faster than likelihood-based methods.

The computational problem to find a haplotype configuration containing the minimum number of breakpoints for pedigree genotype data is in general NP-hard[9]. That is, there is unlikely a polynomial time algorithm that reconstructs such haplotype configurations. When no recombination (or crossover, or breakpoint) events are allowed, the problem becomes to infer zero-recombination haplotypes from given pedigree genotype data, which is called *zero-recombination haplotype configuration* (ZRHC) problem. The ZRHC problem turns out to be polynomial time solvable on *full pedigrees*, in which every non-founder member has both genotyped parents in the pedigree. Li and Jiang[9] proposed an $O(m^3n^3)$ time algorithm solving ZRHC by reducing the problem to solving a system of linear equations over the cyclic group Z_2, where m is the number of SNP loci and n is the number of members in the pedigree. Following several major advances[2,19], Liu and Jiang[12] recently gave a linear time (i.e., $O(mn)$ time) algorithm which generates a particular solution to the ZRHC problem. The algorithm can also generate a general solution (which represents the set of all solutions) in $O(mn^2)$ time.

All the above mentioned zero-recombination haplotyping algorithms work on full pedigree genotype datasets, disallowing any missing genotype data or genotype errors (Mendelian violations). In this work we give a generalization to these algorithms such that, our new rule-based algorithm can handle incomplete pedigrees where some founders are not genotyped, provided that for each nuclear family at least one parent is genotyped and each non-genotyped founder appears in exactly one nuclear family. The importance of this generalization lies in that such cases frequently happen in real data, because some founder members may have passed away and their genotype data can no longer be collected. Our new algorithm produces a general solution to this generalized ZRHC problem in $O(m^3n^3)$ time.

We extend our new zero-recombination haplotyping algorithm to a maximum parsimoniously haplotyping algorithm in one whole genome scan, for general pedigree genotype data to minimize the total number of breakpoint sites, or equivalently, the number of maximal zero-recombination chromosomal regions. The whole genome scan algorithm determines all maximal zero-recombination chromosomal regions as well as their respective haplotype configurations. We prove that the algorithm can be implemented in $O(m^3n^3)$ time in an incremental fashion, here m denotes the total number of SNP loci along the chromosome. Our implementation demonstrates very good performance on simulated datasets, in terms of the recovered breakpoint sites and the recovered haplotype allele *identical-by-descent* (IBD).

The rest of this paper is organized as follows. We first introduce the necessary biological background in Section 2. Then we present the new zero-recombination haplotyping algorithm for incomplete pedigrees and its extension to the whole genome scan haplotyping algorithm in Section 3 and Section 4, respectively. We conclude the paper in Section 5 with some future work.

2 Biological Background and Terminologies

We briefly introduce the necessary biological concepts and terminologies for rule-based haplotype inference on pedigree genotype data. A *pedigree* describes the parent-offspring relationships among the individuals. In a pedigree, the members who have no parents are called *founders*. A *nuclear family* consists of a couple and all their children. A *trio* is a subset of a nuclear family that consists of the two parents and one of their children.

The genome of an organism consists of chromosomes that are double-stranded DNA. A *single nucleotide polymorphism* (SNP) refers to a chromosomal locus where different individuals might have different nucleotides. Each possible nucleotide for the locus is referred to as an *allele* (or *state*, used interchangeably) at the locus. All SNP loci and their alleles define the *genetic map* of the organism. SNPs are considered as the finest *genetic markers* describing the genetic variants, and considered superior to other markers in genetic mapping. In this paper, we deal with bi-allelic SNPs, those having two alleles, and we use identification numbers 1 and 2 to denote them for each marker.

In *diploid* organisms such as humans, chromosomes come in pairs. For each individual, the two alleles at a SNP locus of a pair of homologous chromosomes is called the *genotype* of this locus. The genotype of a locus does not specify which allele comes from which one of the two chromosomes. Thus, the genotype of a locus can be denoted as an unordered pair of two alleles $\{a, b\}$, where $a, b \in \{1, 2\}$, and the genotype of a pair of homologous chromosomes can be viewed as a sequence of unordered pairs of SNP alleles. On the contrary, the *haplotype* of a SNP locus specifies which allele comes from which one of the two homologous chromosomes, and a haplotype of a chromosome consists of all the alleles, one for each SNP locus, on the chromosome. A SNP locus is *homozygous* if its two alleles are the same, or *heterozygous* otherwise.

The Mendelian law of inheritance states that, at each locus of a pair of homologous autosomes of the child, one allele is inherited from the father (the *paternal* allele) and the other from the mother (the *maternal* allele). In general, a child does not inherit a complete parental chromosome from each parent, as *recombination* (or *crossover*) events may occur. During the meiosis process where the two parental chromosomes get duplicated and shuffled and four chromatids are made, one of the chromatids is passed on to the child. Between any two consecutive loci along the chromosome, if recombination occurs then we say that there is a *breakpoint* between these two loci on the chromatid the child inherits. Similarly as in [9], for each locus of each individual in the pedigree, we use *parental source* (PS) value to indicate at this locus which allele comes from which parent. For a homozygous locus the PS value will always be 0. For a heterozygous locus, the PS value can be 0 or 1, where 0 indicates that at this locus the allele with the smaller identification number is from the father and the allele with the larger identification number is from the mother, and 1 indicates the opposite.

3 A New Algorithm Dealing with Non-genotyped Founders

In this section we present a new algorithm for calculating all recombination-free haplotype configurations for a given pedigree genotype dataset. Our new algorithm can handle the pedigrees in which some founder members are not genotyped, provided that for each nuclear family at least one parent is genotyped and each non-genotyped founder appears in exactly one nuclear family. For ease of presentation, we call the pedigrees that can be handled the *general* pedigrees. We want to point out that the previous algorithms only work on full pedigrees, in which every

non-founder member has both parents genotyped[2,9,12,19]. This new algorithm is then used as a subroutine in a maximum parsimoniously whole genome haplotyping algorithm for general pedigree genotype data, described in Section 4.

3.1 Brief Review of the PedPhase Algorithm and New Observations

Li and Jiang[9] presented a constraint-based algorithm for calculating all recombination-free haplotype configurations on full pedigree data. They have also implemented the algorithm in C++ as a part of the *PedPhase* program. In this paper, we will simply call this algorithm the *PedPhase* algorithm as there is no confusion. Briefly, they described three levels of constraints (level 2, 3 constraints in their paper) on the PS values, and proved that every feasible PS assignment corresponds to a feasible recombination-free haplotype configuration. These constraints are defined over the trios extracted from the pedigree (and thus the pedigree has to be full), based on the Mendelian law and the zero-recombination assumption. The constraints can all be expressed as linear equations on the binary PS variables over the cyclic group Z_2. Therefore, the problem of finding all feasible recombination-free haplotype configurations is reduced to solving a system of linear equations over Z_2.

To generalize the PedPhase algorithm to deal with general pedigrees, we observe that all the constraints defined on trios in PedPhase can be rewritten as constraints over parent-child pairs. This essentially enables us to deal with general pedigrees, in which trios are not guaranteed. That is, we will write down all the constraint equations over the parent-child pairs for which the parent is genotyped. Upon full pedigrees, the system of linear equations over parent-child pairs can be proven to be equivalent to the system over trios written in PedPhase. For the sake of preciseness, these constraints over the parent-child pairs are listed in detail in Table 1, which are referred to as the *basic constraints*. Similarly as in [9], in Table 1, constraints for Cases 1–2 are for a locus p where parent x is homozygous and child z is heterozygous; constraints for Cases 3–9 are for two consecutive (but not necessarily adjacent) heterozygous loci p and q of parent x, that is, x is homozygous at every locus in between p and q. The necessity and sufficiency of these constraints can be proven in a similar way as Theorem 3 in [9].

3.2 Dealing with Nuclear Families Having Only One Genotyped Parent

Next we present a method to write down more constraint equations, besides those in Table 1, for a nuclear family having only one genotyped parent in the general pedigrees. This method is the basis of our new zero-recombination haplotyping algorithm for general pedigree genotype datasets.

Consider a nuclear family having only one genotyped parent in the pedigree, in which x is the parent and $c_1, c_2, \cdots, c_d$ are all the d ($d > 2$) children of x and y, where y is the other parent not genotyped. Assume without loss of generality that x is the father and y is the mother. Under the Mendelian law and the zero-recombination assumption, each child inherits one of the two haplotypes of x, which can be guaranteed by the basic constraints listed in Table 1, and one of the two haplotypes of y. Due to the fact that y is not genotyped, all we can tell is that the number of distinct maternal haplotypes of all d children is no more than two. A haplotype configuration for this nuclear family satisfying this additional constraint on the number of distinct maternal haplotypes is said *feasible*. Note that when $d \leq 2$, this additional constraint is automatically satisfied, and therefore the interesting case is $d > 2$. Clearly, for any three distinct children c_i, c_j, c_k among all these d children, this additional constraint on

Table 1. All the possible basic constraints over parent-child pairs. The first (and second) line in a square bracket represents the genotype at locus p (and locus q) of the member. x_p and x_q are the binary PS variables for locus p and locus q of x respectively; z_p and z_q are the binary PS variables for locus p and locus q of z respectively. These constraints are derived from the Mendelian law of inheritance and the assumption that no recombination happens in between loci p and q.

Case	Parent x	Child z	Constraint equations
1	$\begin{bmatrix} 1 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 2 \end{bmatrix}$	$z_p = 0$, if x is the father; $z_p = 1$, if x is the mother
2	$\begin{bmatrix} 2 & 2 \end{bmatrix}$	$\begin{bmatrix} 1 & 2 \end{bmatrix}$	$z_p = 1$, if x is the father; $z_p = 0$, if x is the mother
3	$\begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$	$\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$ or $\begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix}$	$x_p + x_q = 0$
4	$\begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$	$\begin{bmatrix} 1 & 1 \\ 2 & 2 \end{bmatrix}$ or $\begin{bmatrix} 2 & 2 \\ 1 & 1 \end{bmatrix}$	$x_p + x_q = 1$
5	$\begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$	$\begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$	$x_p + x_q + z_p + z_q = 0$
6	$\begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$	$\begin{bmatrix} 1 & 2 \\ 1 & 1 \end{bmatrix}$	$x_p + x_q + z_p = 0$, if x is the father; $x_p + x_q + z_p = 1$, if x is the mother
7	$\begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$	$\begin{bmatrix} 1 & 2 \\ 2 & 2 \end{bmatrix}$	$x_p + x_q + z_p = 1$, if x is the father; $x_p + x_q + z_p = 0$, if x is the mother
8	$\begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$	$\begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix}$	$x_p + x_q + z_q = 0$, if x is the father; $x_p + x_q + z_q = 1$, if x is the mother
9	$\begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$	$\begin{bmatrix} 2 & 2 \\ 1 & 2 \end{bmatrix}$	$x_p + x_q + z_q = 1$, if x is the father; $x_p + x_q + z_q = 0$, if x is the mother

the number of distinct maternal haplotypes must also be satisfied. The following theorem says that a haplotype configuration is feasible if and only if the induced haplotype configuration on every quadruple (x, c_i, c_j, c_k) is feasible. For the quadruple (x, c_i, c_j, c_k) and any two distinct loci p and q, the eight genotype values of x, c_i, c_j, and c_k at loci p and q form a *claw*.

Theorem 1. *For a nuclear family consisting of parent x and children $c_1, c_2, \cdots, c_d$ of x and y, where y is the other parent not genotyped, a haplotype configuration for x and $c_1, c_2, \cdots, c_d$ is feasible if and only if the haplotype configuration restricted to every claw is feasible.*

Proof. Again we assume without loss of generality that x is the father and y is the mother. The *only if* part is obvious. For the *if* part, we prove by contradiction. Suppose restricted to each claw the haplotype configuration is feasible. Then, the paternal haplotype of each child much be equal to one of the two haplotypes of x, which can be proved in the same way as in the proof of Theorem 3 in [9]. Since the haplotype configuration is not feasible for the whole nuclear family consisting of x and $c_1, c_2, \cdots, c_d$, we conclude that the number of different maternal haplotypes of $c_1, c_2, \cdots, c_d$ is at least three. Further assume that the three maternal haplotypes of children c_i, c_j, c_k are distinct from each other. Then, there must exist a locus p at which the three maternal SNP alleles of c_i, c_j and c_k are not the same. Without loss of generality we can assume that at locus p, c_i and c_j have maternal SNP allele 1, and c_k has maternal SNP allele 2. A step further, since the maternal haplotypes of c_i and c_j are distinct, there must exist another locus q at which the maternal SNP alleles of c_i and c_j differ. These indicate that the haplotype configuration restricted to the claw defined by quadruple (x, c_i, c_j, c_k) and loci p and q is infeasible, a contradiction.

3.3 Dealing with Claws: Three Scenarios

Let parent x (assumed to be the father) and three children z, u, v be the four members in a

claw, which involves loci p and q. In the following, for a heterozygous locus $i \in \{p, q\}$ of each member $w \in \{x, z, u, v\}$, we define a binary variable w_i to represent her PS value at the locus. We then present the extra constraints on w_i, besides the basic constraints in Table 1, that have to be satisfied by any feasible haplotype configurations for the claw. We also prove that by satisfying all these extra constraints, a haplotype configuration is feasible for the claw. There are three distinct scenarios of a claw that need to be discussed separately.

In the first scenario, every haplotype configuration satisfying all the basic constraints is already feasible for this claw and thus we do not need to introduce any extra constraints for this claw. For example, assume that $p < q$ and every member is heterozygous at both loci p and q, i.e., $x = \begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$, $z = \begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$, $u = \begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$, and $v = \begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$ (where for each member the first line is the genotype at locus p and the second line is for locus q). Then, for this claw we have the following basic constraints: $x_p + x_q + z_p + z_q = 0$, $x_p + x_q + u_p + u_q = 0$, and $x_p + x_q + v_p + v_q = 0$. There are essentially two possible haplotype configurations satisfying all these basic constraints as follows (in each case the paternal and maternal haplotypes of x, z, u, and v can be swapped), and one can check that both of them are feasible for this claw:

$$x = \begin{bmatrix} 1 \mid 2 \\ 1 \mid 2 \end{bmatrix}, \quad z = \begin{bmatrix} 1 \mid 2 \\ 1 \mid 2 \end{bmatrix}, \quad u = \begin{bmatrix} 1 \mid 2 \\ 1 \mid 2 \end{bmatrix}, \quad v = \begin{bmatrix} 1 \mid 2 \\ 1 \mid 2 \end{bmatrix}, \tag{1.1}$$

$$x = \begin{bmatrix} 1 \mid 2 \\ 2 \mid 1 \end{bmatrix}, \quad z = \begin{bmatrix} 1 \mid 2 \\ 2 \mid 1 \end{bmatrix}, \quad u = \begin{bmatrix} 1 \mid 2 \\ 2 \mid 1 \end{bmatrix}, \quad v = \begin{bmatrix} 1 \mid 2 \\ 2 \mid 1 \end{bmatrix}. \tag{1.2}$$

In the second scenario, in addition to all the basic constraints, some extra constraints are required to ensure that the haplotype configuration is feasible for this claw. For example, assume that $p < q$ and the genotype of the quadruple x, z, u, v at loci p and q are as follows: $x = \begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$, $z = \begin{bmatrix} 1 & 2 \\ 1 & 1 \end{bmatrix}$, $u = \begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$, and $v = \begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$. Then, all the basic constraints for this claw are as follows: $x_p+x_q+z_p = 0$, $x_p+x_q+u_p+u_q = 0$, and $x_p+x_q+v_p+v_q = 0$. Consider the following two haplotype configurations, both of which satisfy all three basic constraints:

$$x = \begin{bmatrix} 1 \mid 2 \\ 2 \mid 1 \end{bmatrix}, \quad z = \begin{bmatrix} 2 \mid 1 \\ 1 \mid 1 \end{bmatrix}, \quad u = \begin{bmatrix} 1 \mid 2 \\ 2 \mid 1 \end{bmatrix}, \quad v = \begin{bmatrix} 1 \mid 2 \\ 2 \mid 1 \end{bmatrix}, \tag{2.1}$$

$$x = \begin{bmatrix} 1 \mid 2 \\ 2 \mid 1 \end{bmatrix}, \quad z = \begin{bmatrix} 2 \mid 1 \\ 1 \mid 1 \end{bmatrix}, \quad u = \begin{bmatrix} 1 \mid 2 \\ 2 \mid 1 \end{bmatrix}, \quad v = \begin{bmatrix} 2 \mid 1 \\ 1 \mid 2 \end{bmatrix}. \tag{2.2}$$

One can check that configuration (2.1) is feasible for this claw while configuration (2.2) is not, since in the second configuration z, u, v have 3 different maternal haplotypes. Nevertheless, if we add an extra constraint $u_q + v_q = 0$ to the above three basic constraints, which essentially says that u and v should have the same inheritance at locus q, then it will rule out configuration (2.2). Since z is homozygous at locus q, for this example, any haplotype configuration is feasible for this claw if and only if it satisfies all these four constraints. That is, for a claw of this genotype configuration, we need to introduce an extra constraint $u_q + v_q = 0$ to guarantee that the satisfying haplotype configuration is feasible for this claw.

In Table 2, all the essentially different seven cases that can be classified into this scenario and their corresponding extra constraints are listed (the above example corresponds to Case 6). For each of these cases, one can validate both the necessity and sufficiency for satisfying haplotype configurations being feasible. We note that there are six other symmetric cases by swapping the genotype configuration for loci p and q in Cases 1–6 in Table 2, as well as permuting the roles of z, u, v.

Table 2. All the possible extra constraints over a claw on loci p and q, in which x is the parent and z, u, v are three children of x and the other parent y not genotyped. Genotype alleles $a, b, c \in \{1, 2\}$, with an $*$ indicates an arbitrary allele. Note that swapping the genotype configuration at loci p and q in Cases 1–6 gives another six symmetric cases. These constraints are derived from the Mendelian law of inheritance and the assumption no recombination happening in between loci p and q.

Case	Parent x	z	u	v	Constraint equations
1	$\begin{bmatrix} a & a \\ 1 & 2 \end{bmatrix}$	$\begin{bmatrix} a & a \\ b & b \end{bmatrix}$	$\begin{bmatrix} a & a \\ 1 & 2 \end{bmatrix}$	$\begin{bmatrix} 1 & 2 \\ * & * \end{bmatrix}$	$u_q = b$, if x is the father; $u_q = b+1$, if x is the mother
2	$\begin{bmatrix} a & a \\ 1 & 2 \end{bmatrix}$	$\begin{bmatrix} a & a \\ * & * \end{bmatrix}$	$\begin{bmatrix} 1 & 2 \\ b & b \end{bmatrix}$	$\begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$	$v_q = b$, if x is the father; $v_q = b+1$, if x is the mother
3	$\begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$	$\begin{bmatrix} a & a \\ b & b \end{bmatrix}$	$\begin{bmatrix} 1 & 2 \\ c & c \end{bmatrix}$	$\begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$	$v_q = b$, if x is the father; $v_q = b+1$, if x is the mother
4	$\begin{bmatrix} a & a \\ 1 & 2 \end{bmatrix}$	$\begin{bmatrix} a & a \\ 1 & 2 \end{bmatrix}$	$\begin{bmatrix} a & a \\ 1 & 2 \end{bmatrix}$	$\begin{bmatrix} 1 & 2 \\ * & * \end{bmatrix}$	$z_q + u_q = 0$
5	$\begin{bmatrix} a & a \\ 1 & 2 \end{bmatrix}$	$\begin{bmatrix} a & a \\ * & * \end{bmatrix}$	$\begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$	$\begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$	$u_q + v_q = 0$
6	$\begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$	$\begin{bmatrix} 1 & 2 \\ a & a \end{bmatrix}$	$\begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$	$\begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$	$u_q + v_q = 0$
7	$\begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$	$\begin{bmatrix} a & a \\ 1 & 2 \end{bmatrix}$	$\begin{bmatrix} 1 & 2 \\ b & b \end{bmatrix}$	$\begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$	$x_p + x_q = a + b + 1$

In the third scenario, none of the haplotype configurations satisfying the basic constraints is feasible for the claw. That is, the existence of a claw of such a scenario implies no existence of any feasible haplotype configuration under the Mendelian law of inheritance and the zero-recombination assumption. For example, assume that $p < q$ and the genotype of the quadruple x, z, u, v at loci p and q are as follows: $x = \begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$, $z = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix}$, $u = \begin{bmatrix} 2 & 2 \\ 1 & 2 \end{bmatrix}$, and $v = \begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$. Then, for this claw we have the following basic constraints: $x_p + x_q + z_q = 0$, $x_p + x_q + u_q = 1$, and $x_p + x_q + v_p + v_q = 0$. There are essentially two different haplotype configurations satisfying these constraints (in each of them the paternal and maternal haplotypes of x and v can be swapped):

$$x = \begin{bmatrix} 1 \mid 2 \\ 1 \mid 2 \end{bmatrix}, \quad z = \begin{bmatrix} 1 \mid 1 \\ 1 \mid 2 \end{bmatrix}, \quad u = \begin{bmatrix} 2 \mid 2 \\ 2 \mid 1 \end{bmatrix}, \quad v = \begin{bmatrix} 1 \mid 2 \\ 1 \mid 2 \end{bmatrix}, \tag{3.1}$$

$$x = \begin{bmatrix} 1 \mid 2 \\ 2 \mid 1 \end{bmatrix}, \quad z = \begin{bmatrix} 1 \mid 1 \\ 2 \mid 1 \end{bmatrix}, \quad u = \begin{bmatrix} 2 \mid 2 \\ 1 \mid 2 \end{bmatrix}, \quad v = \begin{bmatrix} 1 \mid 2 \\ 2 \mid 1 \end{bmatrix}. \tag{3.2}$$

One can check that none of the above haplotype configurations is feasible, since in each of them children z, u, v have three different maternal haplotypes.

All the possible genotype configurations of a claw falling into this scenario are listed in Table 3. We note that by an essentially exhaustive enumeration of all the $3^8 = 6,561$ genotype configurations for claws, any of them is successfully classified into one of the above three scenarios.

3.4 The New Zero-recombination Haplotyping Algorithm for General Pedigrees

Given a general pedigree genotype dataset, our new zero-recombination haplotyping algorithm first checks whether there exists a claw belonging to the third scenario, and if so then no zero-recombination haplotype configuration exists and the algorithm halts. Otherwise, the algorithm collects all the basic and extra constraints to form a system of linear equations over Z_2, and solves it using the Gaussian elimination method. On halting, each feasible solution to the system specifies a feasible haplotype configuration, spelled by the values of the binary PS variables in

Table 3. All the possible genotype configurations for a claw with no feasible haplotype configuration. Genotype alleles $a, b \in \{1, 2\}$, with an $*$ indicates an arbitrary allele. Note that swapping the genotype configuration at loci p and q in each case gives another symmetric case. The infeasibility conclusion is drawn from the Mendelian law of inheritance and the assumption that no recombination happens in between loci p and q.

Case	Parent x	z	u	v
1	$\begin{bmatrix}1&2\\a&a\end{bmatrix}$	$\begin{bmatrix}1&1\\a&a\end{bmatrix}$	$\begin{bmatrix}2&2\\a&a\end{bmatrix}$	$\begin{bmatrix}*&*\\1&2\end{bmatrix}$
2	$\begin{bmatrix}1&2\\a&a\end{bmatrix}$	$\begin{bmatrix}b&b\\a&a\end{bmatrix}$	$\begin{bmatrix}1&1\\1&2\end{bmatrix}$	$\begin{bmatrix}2&2\\1&2\end{bmatrix}$
3	$\begin{bmatrix}1&2\\a&a\end{bmatrix}$	$\begin{bmatrix}1&2\\a&a\end{bmatrix}$	$\begin{bmatrix}1&1\\1&2\end{bmatrix}$	$\begin{bmatrix}2&2\\1&2\end{bmatrix}$
4	$\begin{bmatrix}1&2\\1&2\end{bmatrix}$	$\begin{bmatrix}1&1\\2&2\end{bmatrix}$	$\begin{bmatrix}2&2\\1&1\end{bmatrix}$	$\begin{bmatrix}1&2\\a&a\end{bmatrix}$
5	$\begin{bmatrix}1&2\\1&2\end{bmatrix}$	$\begin{bmatrix}1&1\\1&1\end{bmatrix}$	$\begin{bmatrix}2&2\\2&2\end{bmatrix}$	$\begin{bmatrix}1&2\\a&a\end{bmatrix}$
6	$\begin{bmatrix}1&2\\1&2\end{bmatrix}$	$\begin{bmatrix}1&2\\1&2\end{bmatrix}$	$\begin{bmatrix}1&1\\1&2\end{bmatrix}$	$\begin{bmatrix}2&2\\1&2\end{bmatrix}$
7	$\begin{bmatrix}1&2\\1&2\end{bmatrix}$	$\begin{bmatrix}a&a\\a&a\end{bmatrix}$	$\begin{bmatrix}1&1\\1&2\end{bmatrix}$	$\begin{bmatrix}2&2\\1&2\end{bmatrix}$
8	$\begin{bmatrix}1&2\\1&2\end{bmatrix}$	$\begin{bmatrix}a&a\\a&a\end{bmatrix}$	$\begin{bmatrix}b&b\\1&2\end{bmatrix}$	$\begin{bmatrix}1&2\\b&b\end{bmatrix}$
9	$\begin{bmatrix}1&2\\1&2\end{bmatrix}$	$\begin{bmatrix}1&1\\2&2\end{bmatrix}$	$\begin{bmatrix}1&1\\1&2\end{bmatrix}$ or $\begin{bmatrix}1&2\\1&1\end{bmatrix}$	$\begin{bmatrix}1&2\\2&2\end{bmatrix}$ or $\begin{bmatrix}2&2\\1&2\end{bmatrix}$

the solution. Note that there might be no feasible solution to the system, indicating no existence of zero-recombination haplotype configuration. The correctness of our new algorithm follows directly from the above analysis in this section.

Next, we show that the new algorithm runs in $O(m^3n^3)$ time, where m is the number of SNPs and n is the size of the pedigree. Firstly, since a claw consists of a parent and three children and two loci, there are no more than $O(m^2n^3)$ claws. For each claw, checking whether or not it belongs to the third scenario takes constant time. This means the process of checking the existence of a *bad* claw can be done in $O(m^2n^3)$ time. At the no existence of bad claws, the algorithm collects all the basic constraints for parent-child pairs and all the extra constraints for claws falling into the second scenario. As there are less than $2n$ parent-child pairs, and from Table 1 for each parent-child pair there are at most m constraints in Cases 1–2 and at most $m-1$ constraints in Cases 3–9, we conclude that the total number of basic constraints is less than $4mn$. The next Lemma 2 shows that all the extra constraints can be written into a system of at most $3mn$ linear equations.

Lemma 2. *All the extra constraints can be written into a system of at most* $3mn$ *linear equations over the binary PS variables.*

Proof. We deal with the seven cases of extra constraints in Table 2, the other six symmetric cases by swapping the genotype configurations at loci p and q in Cases 1–6, not listed in the table, and other symmetric cases by swapping the genotype configurations of the three children all together.

Firstly, all extra constraints of Cases 1–3 are of the form $w_i = 0$ or $w_i = 1$, where $w \in \{z, u, v\}$ and $i \in \{p, q\}$. We only need to keep a record for each such variable w_i and its value. Clearly, there are at most mn such variables.

Secondly, all extra constraints of Cases 4–6 are of the form $w_i + w'_i = 0$, or equivalently $w_i = w'_i$, where $w, w' \in \{z, u, v\}$. Consider all the children $c_1, c_2, \cdots, c_d$ of x (and the other parent y not genotyped) as a group. At each locus p, based on all the extra constraints of Cases 4–6 that involve locus p and members of $\{c_1, c_2, \cdots, c_d\}$, the binary PS variables $w_p, w \in \{c_1, c_2, \cdots, c_d\}$, can be divided into disjoint subsets such that all the variables within

a subset should be equal to each other. This implies that the extra constraints of Cases 4–6 involving members of group $\{c_1, c_2, \cdots, c_d\}$ at each locus can be re-written using no more than $d-1$ linear equations. Therefore, at most mn linear equations are necessary to re-write all the extra constraints of Cases 4–6.

Finally, for extra constraints of Case 7, they are all of the form $x_p + x_q = 0$ (i.e., $x_p = x_q$) or $x_p + x_q = 1$ (i.e., $x_p \neq x_q$), for some single parent x. For x, consider all the extra constraints of Case 7 of the form $x_p + x_q = 0$. Similarly as in the last paragraph, all the variables involved in these constraints can be divided into disjoint subsets such that all the variables within a subset should be equal to each other. View each such subset as a node. Two such nodes are connected by an edge if and only if there is a variable from each node such that these two variables are in an extra constraint $x_p + x_q = 1$ (i.e., $x_p \neq x_q$) of Case 7. Let G denote the resulting graph. Clearly, if G is not bipartite, then there is no feasible haplotype configuration. In the other case, we may similarly re-write all the extra constraints of Case 7 on single parent x using at most $m-1$ linear equations. It follows that at most mn linear equations are necessary to re-write all the extra constraints of Case 7.

Summing up, all the extra constraints on claws can be re-written into a system of less than $3mn$ linear equations.

From Lemma 2, we conclude that the system to be solved via the Gaussian elimination method contains no more than $7mn$ linear equations. Therefore, the Gaussian elimination method will take $O(m^3n^3)$ time to terminate. Also, by Tables 1 and 2 and the above proof of Lemma 2, collecting all these $O(mn)$ linear equations can be done within $O(m^3n^3)$ time. We have thus established the running time of our algorithm summarized in the following.

Theorem 3. *The running time of our new zero-recombination haplotyping algorithm on general pedigree genotype datasets is $O(m^3n^3)$, where m is the number of SNPs under consideration and n is the size of the general pedigree.*

4 A Haplotyping Algorithm in One Whole Genome Scan

In this section, we extend the above zero-recombination haplotyping algorithm for general pedigree genotype datasets to a maximum parsimoniously haplotyping algorithm in one whole genome scan. Note that zero-recombination assumption holds for only small chromosomal regions, but not necessarily for a whole chromosome, even on sparse SNP genotype datasets. Therefore, we drop the zero-recombination assumption while to infer the haplotype configurations using the minimum number of breakpoint sites, at which either one or multiple members require a crossover event. Such an objective function is equivalent to minimizing the number of zero-recombination chromosomal regions along the whole chromosome.

Naively, one may chop out any contiguous chromosomal region, represented by a pair of a starting SNP locus and an ending SNP locus, then call the above zero-recombination haplotyping algorithm on it. A little better way is to fix one end of the chromosomal regions to search for the other end for determining a maximal zero-recombination region. This involves solving $O(m)$ datasets and thus the overall running time is $O(m^4n^3)$, where m is the total number of SNPs along the chromosome. Next, we introduce a whole genome scan algorithm for this purpose in $O(m^3n^3)$ time.

Our whole genome scan algorithm starts from the first locus, takes sequentially the next locus into account in each step, and invokes the above zero-recombination haplotyping algorithm to see whether a zero-recombination haplotype configuration exists for the current chromosomal

region under consideration. If so, it continues to consider the next locus. Otherwise, it outputs a zero-recombination haplotype configuration for the last chromosomal region, which does not include the current locus, and starts with the current to consider the next locus. Overall, each locus is considered at most twice by the zero-recombination haplotyping algorithm, one together with its preceding locus and one with its succeeding locus. The following lemma establishes the optimality of our whole genome scan haplotyping algorithm, that it achieves the minimum number of breakpoint sites for the given general pedigree genotype dataset.

Lemma 4. *The whole genome scan haplotyping algorithm achieves the minimum number of breakpoint sites for any given general pedigree genotype dataset.*

Proof. Assume the SNP loci are indexed by integers 1 to m, and the whole genome scan haplotyping algorithm reports k breakpoint sites: $p_1, p_2, \cdots, p_k$, where p_i is located between loci ℓ_i and $\ell_i + 1$ ($1 \leq \ell_1 < \ell_2 < \cdots < \ell_k < m$). Let $\ell_0 = 0$. For each $i = 0, 1, \cdots, k-1$, the chromosomal region starting with locus $\ell_i + 1$ and ending at locus $\ell_{i+1} + 1$ is not a zero-recombination region, from the fact that our zero-recombination haplotyping algorithm is an exact algorithm. This says that there are at least k breakpoint sites along the chromosome (or at least $k + 1$ maximal zero-recombination chromosomal regions).

To guarantee the $O(m^3n^3)$ running time of the whole genome scan haplotyping algorithm, we need to carefully execute every step of the zero-recombination haplotyping algorithm. We do this in a novel incremental fashion. For each new locus taken into consideration, the algorithm first checks all the claws involving this locus to see whether or not any of them belongs to the third scenario. If so, a new breakpoint site is found right before this new locus (the current locus). Otherwise, the algorithm examines all the basic and all the extra constraints involving the current locus, and collects the corresponding linear equations. The algorithm keeps a record of the matrix which is the result of applying the Gaussian elimination method to the system of all the linear equations collected before considering the current locus, and such a matrix had been reduced into the row echelon form. The algorithm adds to this recorded matrix the newly collected linear equations and applies the Gaussian elimination method only to these newly added linear equations (or the newly added rows in the matrix) to further reduce it into the row echelon form. If failed, that is, the newly expanded system of linear equations is found to be infeasible, then a new breakpoint site is found right before the current locus. Otherwise, the row echelon form of the matrix is updated (and recorded for next iteration) and the algorithm proceeds to consider the next locus, and so on.

Recall the analysis of the running time of the zero-recombination haplotyping algorithm in Section 3.4. In this whole genome scan haplotyping algorithm to consider the current locus, the total number of claws to be examined, on whether or not any of them belongs to the third scenario, is still $O(m^2n^3)$. If no such existence, the algorithm moves on to collect the basic and the extra constraints. The number of basic constraints involving the current locus is trivially $O(n)$. The number of linear equations that together re-write the extra constraints involving the current locus could be $O(mn)$; Nevertheless, if we only write down the linear equations that are "*independent*" of all the previously written linear equations, from the proof of Lemma 2, for a maximal zero-recombination region containing m SNPs, the whole genome scan haplotyping algorithm only writes down $O(mn)$ linear equations to cover all the extra constraints. It follows that again the total number of linear equations been written down by the whole genome scan haplotyping algorithm is $O(mn)$, implying an $O(m^3n^3)$ running time of the algorithm. We have the following theorem.

Theorem 5. *The whole genome scan haplotyping algorithm runs in $O(m^3n^3)$ time and achieves the minimum number of breakpoint sites on any given general pedigree genotype dataset, where m is the number of SNPs in the dataset and n is the size of the general pedigree.*

5 Conclusions and Future Work

In this paper, we presented an alternative writing of the linear equations to constrain the haplotype allele inheritance, which was based on trios and now based on parent-child pairs extracted from the given pedigree, under the zero-recombination assumption. The impact of this new observation is big in that it enables us to deal with general pedigree structures, in which some founder members are not genotyped provided that for each nuclear family at least one parent is genotyped and each non-genotyped founder appears in exactly one nuclear family. We presented a new rule-based zero-recombination haplotyping algorithm for such general pedigree genotype datasets, through imposing extra constraints on the inheritance among members of each single parent nuclear family. We further demonstrated that these extra constraints can be re-written into a system of $O(mn)$ linear equations, where m is the number of SNPs under consideration and n is the size of the pedigree. Consequently, our new algorithm runs in $O(m^3n^3)$ time. We further extended this algorithm to haplotype the whole chromosome using the minimum number of breakpoint sites, in one whole genome scan. This whole genome scan haplotyping algorithm, which has been implemented as the iBDD program, is proven to run in $O(m^3n^3)$ time as well and performed very efficiently and effectively in haplotype recovery in an extensive simulation experiment.

In the real cattle genotype datasets that we are working on, pedigrees of various sizes exist. Interestingly, none of them is a full pedigree (each non-founder member has both presents genotyped), but many of them are of the general pedigree structures we deal with in this paper. Yet in some of them, a cattle could have both parents not genotyped, or a cattle is the father of multiple nuclear families but itself is unfortunately not genotyped. For both cases, we have investigated some small instances and found out that the numbers of feasible solutions are not exact powers of 2. Given that the number of feasible solutions to a system of linear equations over the cyclic group Z_2 is always an exact power of 2, we thus doubt that the same approach of defining one binary variable for each heterozygous locus of each member and expressing constraints as a *single* system of linear equations can be generalized to cover these two cases.

Nevertheless, it is practically interesting to generalize our algorithms, and previous ones, to handle missing genotype data in the general pedigree genotype datasets considered in this paper, since the real datasets often contain a few percents of missing values.

References

[1] Altshuler, D., Daly, M.J., Lander, E.S. Genetic mapping in human disease. *Science*, 322: 881–888 (2008)
[2] Chan, M.Y., Chan, W., Chin, F., Fung, S., Kao, M. Linear-time haplotype inference on pedigrees without recombinations. In: Proceedings of the 6th Annual Workshop on Algorithms in Bioinformatics (WABI'06), 2006, 56–67
[3] Du, F.X., Woodward, B.W., Denise, S.K. Haplotype construction of sires with progeny genotypes based on an exact likelihood. *Journal of Dairy Sciences*, 81: 1462–1468 (1998)
[4] Excoffer, L., Slatkin, M. Maximum-likelihood estimation of molecular haplotype frequencies in a diploid population. *Molecular Biology and Evolution*, 12: 921–927 (1995)
[5] Gusfield, D. Inference of haplotypes from samples of diploid populations: complexity and algorithms. *Journal of Computational Biology*, 8: 305–323 (2001)
[6] Hawley, M.E., Kidd, K.K. HAPLO: a program using the EM algorithm to estimate the frequencies of

multi-site haplotypes. *Journal of Heredity*, 86: 409–411 (1995)
[7] Kruglyak, L., Daly, M.J., Reeve-Daly, M.P., Lander, E.S. Parametric and nonparametric linkage analysis: a unified multipoint approach. *American Journal of Human Genetics*, 58: 1347–1363 (1996)
[8] Lander, E.S., Green, P. Construction of multilocus genetic linkage maps in human. *Proceedings of National Academy of Sciences of USA*, 84: 2363–2367 (1987)
[9] Li, J., Jiang, T. Efficient rule-based haplotyping algorithms for pedigree data. In: Proceedings of the 7th Annual Conference on Research in Computational Molecular Biology (RECOMB'03), 2003, 197–206
[10] Lin, G., Wang, Z., Wang, L., Lau, Y.L., Yang, W. Identification of linked regions using high-density SNP genotype data for linkage analyses. *Bioinformatics*, 24: 86–93 (2008)
[11] Lin, S., Cutler, D.J., Zwick, M.E., Chakravarti, A. Haplotype inference in random population samples. *American Journal of Human Genetics*, 71: 1129–1137 (2002)
[12] Liu, L., Jiang, T. A linear-time algorithm for reconstructing zero-recombinant haplotype configuration on pedigrees without mating loops. *Journal of Combinatorial Optimization*, 2009. Online first.
[13] Long, J.C., Williams, R.C., Urbanek, M. An E-M algorithm and testing strategy for multiple-locus haplotypes. *American Journal of Human Genetics*, 56: 799–810 (1995)
[14] Niu, T., Qin, Z.S., Xu, X., Liu, J.S. Bayesian haplotype inference for multiple linked single-nucleotide polymorphisms. *American Journal of Human Genetics*, 70: 157–169 (2002)
[15] Qin, Z., Niu, T., Liu, J. Partitioning-ligation-expectation maximization algorithm for haplotype inference with single nucleotide polymorphisms. *American Journal of Human Genetics*, 71: 1242–1247 (2002)
[16] Sobel, E., Lange, K., O'Connell, J.R., Weeks, D.E. Haplotype algorithms. In T.P. Speed and M.S. Waterman, editors. Genetic Mapping and DNA Sequencing. IMA Volumes in Mathematics and Its Applications, P. 89–110. Springer, New York, 1995
[17] Stephens, M., Smith, N., Donnelly, P. A new statistical method for haplotype reconstruction from population data. *American Journal of Human Genetics*, 68: 978–989 (2001)
[18] Weeks, D.E., Sobel, E., O'Connell, J.R., Lange, K. Computer programs for multilocus haplotyping of general pedigrees. *American Journal of Human Genetics*, 56: 1506–1507 (1995)
[19] Xiao, J., Liu, L., Xia, L., Jiang, T. Fast elimination of redundant linear equations and reconstruction of recombination-free Mendelian inheritance on a pedigree. In: Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'07), 2007, 655–664