PedPhase: Haplotype Inference for Pedigree Data

Jing Li * Tao Jiang †

Abstract

Summary: We have developed a computer program consisting of four algorithms for inferring haplotypes from (unphased) genotypes on pedigree data. These algorithms are designed based on a combinatorial formulation of haplotype inference, namely the *minimum-recombinant haplotype configuration (MRHC)* problem, and are effective for different types of data. One of the algorithms, called block-extension, is an efficient heuristic algorithm for MRHC that performs very well when the input data requires few recombinants. The other three are exact algorithms for MRHC under various restrictions. **Availability:** Executable code for Windows and Linux is available at website `http://www.cs.ucr.edu/~jili/haplotyping.html`. **Contact:** jili@cs.ucr.edu.

1 Introduction

With the completion of the Human Genome Project [5, 13], an (almost) complete human genomic DNA sequence has become available, which is essential to understand the functions and characteristics of human genetic material. An important next step in human genomics is to determine genetic variations among humans and the correlation between genetic variations and phenotypic variations (such as disease status, quantitative traits, *etc.*). To achieve this goal, an international collaboration, namely, the international HapMap project, was launched in October, 2002. The main objective of the HapMap project is to identify the *haplotype* structure of humans and common haplotypes among populations. However the human genome is a diploid and, in practice, *genotype* data instead of haplotype data are collected directly, especially in large scale sequencing projects, mainly due to cost considerations. Hence, efficient and accurate computational methods and computer programs for inferring haplotype information from genotype data are highly demanded. A recent comprehensive review of computational methods for haplotype inference can be found in [2].

Genotype data used in haplotyping can be divided into two categories: *pedigree* data and *population* data. Although collecting population data and reconstructing haplotypes according to some evolutionary model seems an economic way for haplotype inference, studies [1] show that for a larger number of marker loci, haplotype estimations from small pedigrees such as nuclear families (*i.e.* two-level pedigree consisting of only parents and children) can be more efficient than estimations from independent individuals in a population on a per genotype base. Furthermore, some family based statistical tests like TDT (*i.e. Transition Disequilibrium Test*) and its variants require haplotype data as input. In this paper, we are only interested pedigree data.

The existing computational methods for haplotyping can be divided into two categories: statistical methods and rule-based (*i.e. combinatorial*) methods. Statistical methods [8, 11] often infer haplotype configurations with some likelihood estimation, but they are usually very time consuming and not suitable for large data sets. These methods work well for data with sparse markers separated by large genetic (and physical) distances, because they rely on the linkage equilibrium assumption. This assumption does not hold for dense markers like SNPs (*i.e. single nucleotide polymorphisms*), for which linkage disequilibrium between adjacent markers is common. On the other hand, rule-based methods are usually very fast although they normally do not provide any numerical assessment of the reliability of their results. Nevertheless, by utilizing some reasonable biological assumptions, such as the minimum recombination principle (*i.e.* haplotype configurations with the minimum recombinants are likely to be the true haplotype configurations), rule-based methods have proven to be powerful and practical, especially for data with dense markers [9, 10, 12].

In a recent paper [10], Qian and Beckmann proposed a rule-based algorithm for the *Minimum-Recombinant Haplotype Configuration (MRHC) problem*, where we are given a pedigree with genotype data and we are required to find the haplotype

*Department of Computer Science, University of California, Riverside, CA.
†Department of Computer Science, University of California, Riverside, CA, and Shanghai Center for Bioinformatics Technology.

configurations for each member of the pedigree such that the total number of recombinants (or recombination events) in the whole pedigree is minimized. Figure 1 illustrates an example input and the expected output of the MRHC problem. The diagram on the left shows the pedigree structure and genotype information. The diagram on the right shows the (paternal or maternal) origin of each allele in a child (*i.e.* the inferred haplotype/phase information). Notice that, there is a recombinant between loci 6 and 7 of the maternal haplotype in member 3-4.

In recent papers [6, 7, 4], we showed that the MRHC problem is in general NP-hard, even for pedigrees without mating loops,[1] and developed an iterative heuristic algorithm, called block-extension, that is much more efficient than the algorithm MRH given in [10]. Preliminary experiments showed that the algorithm is often able to compute an optimal solution or nearly optimal solution when the minimum number of recombinants required is small [6, 7]. However, its performance decreases when more recombinants are required. We have also devised an efficient exact algorithm for solving MRHC on pedigree data that requires no recombinants, *i.e.* the algorithm can find all haplotype configurations incurring no recombinants [6, 7]. More recently, two dynamic programming algorithms are developed for (general) pedigrees of small sizes and loopless pedigrees with a small number of marker loci [4]. In this paper, we report a computer program implementing these algorithms.

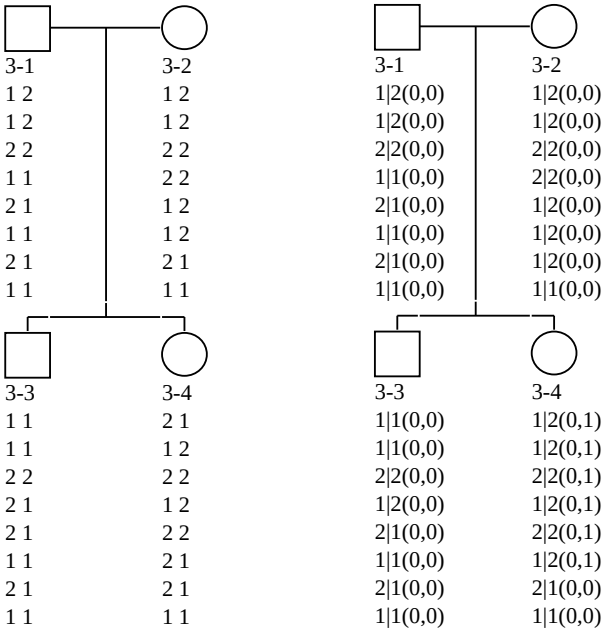

Figure 1: An illustration of the input and expected output of the MRHC problem. In the pedigree, a square represents a male and a circle represents a female. The children (*e.g.* 3-3 and 3-4) are placed under their parents (*e.g.* 3-1 and 3-2). The member ID and genotype data are placed under each member. On the left, the blank between two alleles at a locus indicates that the locus is phase-unknown. On the right, a | separates the paternal allele from the maternal allele. For each phase-resolved locus, we further use two numbers in parentheses to indicate that the alleles come from the parents' paternal (represented as 0) and maternal (represented as 1) alleles. The number of recombinants can be easily derived from this information.

2 Methods and implementation

The above four algorithms are implemented in a program, called PedPhase, in C++. Executable codes for Windows and Linux are available free of cost for non-commercial use. The following gives a brief synopsis of the algorithms.

- **Block-extension algorithm** The block-extension algorithm described in [6, 7] first attempts to resolve the haplotype configuration of all unambiguous loci using the *Mendelian law* of inheritance. Some sensible greedy strategies (such as avoiding double recombinants within a small region of loci) are then used to resolve loci that are adjacent to the previously resolved loci, resulting in *blocks* of consecutive resolved loci. The algorithm then uses the longest block in the pedigree to resolve more unresolved loci under the minimum recombination principle. This may extend some blocks into longer blocks. The process is repeated until no blocks can be extended. The algorithm then fills the remaining gaps between blocks in each member by considering the haplotype information at the other members of the same nuclear family. The time complexity of the above algorithm is $O(dmn)$, where n is the size of the pedigree, m the number of loci, and d the largest number of children in a nuclear family.

[1] A mating loop is a substructure of a pedigree that shows two descendants of the same member mate and produce children.

- **Constraint-finding algorithm for 0-recombinant data** The constraint-finding algorithm [6, 7] first determines if an MRHC instance has 0-recombinant haplotype configurations and identify all such configurations if it does. More precisely, the algorithm first identifies all necessary (and sufficient) constraints on the haplotype configurations derived from the Mendelian law and the zero-recombinant assumption, represented as a system of linear equations on binary variables over the cyclic group Z_2 (*i.e.* integer addition `mod` 2), and then solves the equations to obtain all consistent haplotype configurations satisfying the constraints, using a simple method based on Gaussian elimination. These consistent haplotype configurations are shown to be feasible 0-recombinant solutions. The running time for representing and solving the equations is $O(m^3n^3)$ and the time for enumerating all configurations is proportional to the number of feasible 0-recombinant solutions.

Observe that the complexity of an instance of MRHC is defined by two independent parameters, namely, the number of members in the pedigree (*i.e.* the size of the pedigree) and the number of marker loci in each member. A polynomial time algorithm for MRHC can be constructed if one of these parameter is bounded by a constant (which is often the case in practice). This gives rises to two dynamic programming algorithms introduced in [4].

- **Locus-based dynamic programming algorithm** This algorithm assumes that the number of marker loci is bounded by a small constant and performs dynamic programming on the members of the input pedigree. It works only for pedigrees without mating loops. The algorithm takes advantage of the tree structure of the input pedigree and has a running time linear in the size of the pedigree. It could be very useful for solving MRHC in practice because most real pedigrees are loopless and involve a small number of marker loci (in each haplotype block).
- **Member-based dynamic programming algorithm** This algorithm assumes that the input pedigree is small and performs dynamic programming on the marker loci in each member of the pedigree simultaneously. The algorithm works for any input pedigree (without or with mating loops) and has a running time linear in the number of marker loci. It could be useful as a subroutine for solving MRHC on small pedigrees, *e.g.* nuclear families from a large input pedigree or independent nuclear families from a (semi-)population data.

All the algorithms have been tested on simulated data and real data. The test results [6, 7, 4] demonstrate that the programs are more efficient than the previously known algorithms, and perform as well as those algorithms. In summary, the block-extension algorithm works well for large dataset in a tight linked region, although it does not guarantee optimality of its solution. The program has simple missing allele imputation capabilities. The other three algorithms are exact algorithms, but they are efficient only for pedigree data with certain restrictions. They also assume that no genotype data is missing.

The input to PedPhase is simply a text file. Each line represents a member, specifying its parental and genotype information. The output (text) file gives the parental source of each allele in a member and the parental source of the allele in the involve parent. The text files can be used to draw pedigree diagrams (similar to the ones in Figure 1) using WPEDRAW [3]. The detailed specifications and instructions on the use of the program can be found in the *User's Guide*, available along with the package.

3 Acknowledgement

JL is supported by NSF grant CCR-9988353. TJ is supported by NSF Grants CCR-9988353, ITR-0085910, DBI-0133265, and National Key Project for Basic Research (973).

References

[1] T. Becker and M Knapp, Efficiency of haplotype frequency estimation when nuclear family information is included. *Hum Hered* 54:45-53, 2002.

[2] P. Bonizzoni, G. D. Vedova, R. Dondi and J. Li, The haplotyping problem: an overview of computational models and solutions. Accepted by *J Comp Sci Tech*.

[3] D. Curtis A program to draw pedigrees using linkage or linksys data files. *Annals of Human Genetics*, 54:365–367, 1990.

[4] K. Doi, J. Li and T. Jiang, Minimum recombinant haplotype configuration on tree pedigrees. To appear *WABI'03*.

[5] International Human Genome Sequencing Consortium. Initial sequencing and analysis of the human genome. *Nature*, 409(6822):860–921, 2001.

[6] J. Li and T. Jiang, Efficient rule-based haplotyping algorithms for pedigree data. *Proc. RECOMB'03*, 197–206, 2003.

[7] J. Li and T. Jiang, Efficient inference of haplotypes from genotypes on a pedigree. *J Bioinfo and Comp Biol* 1(1):41-69, 2003.

[8] S. Lin and T. P. Speed. An algorithm for haplotype analysis. *J Comput Biol*, 4(4):535–546, 1997.

[9] J. R. O'Connell. Zero-recombinant haplotyping: applications to fine mapping using snps. *Genet Epidemiol*, 19 Suppl 1:S64–70, 2000.

[10] D. Qian and L. Beckman. Minimum-recombinant haplotyping in pedigrees. *Am J Hum Genet*, 70(6):1434–1445, 2002.

[11] E. Sobel, K. Lange, J.R. O'Connell and D.E. Weeks. Haplotyping algorithms. T. Speed and M. Waterman, eds., *Genetic Mapping and DNA Sequencing, IMA Vol in Math and its App*, Springer-verlag, New York, 81:89-110, 1996.

[12] P. Tapadar, S. Ghosh, and P. P. Majumder. Haplotyping in pedigrees via a genetic algorithm. *Hum Hered*, 50(1):43–56, 2000.

[13] J. C. Venter, M. D. Adams, E. W. Myers, and *et al.* The sequence of the human genome. *Science*, 291(5507):1304–1351, 2001.