The Haplotyping Problem: An Overview of Computational Models and Solutions

Paola Bonizzoni* Gianluca Della Vedova† Riccardo Dondi‡ Jing Li§

June 21, 2003

Abstract

The investigation of genetic differences among humans has given evidence that mutations in DNA sequences are responsible for some genetic diseases. The most common mutation is the one that involves only a single nucleotide of the DNA sequence, which is called a *single nucleotide polymorphism (SNP)*. As a consequence, computing a complete map of all SNPs occurring in the human populations is one of the primary goals of recent studies in human genomics. The construction of such a map requires to determine the DNA sequences that from all chromosomes. In diploid organisms like humans, each chromosome consists of two sequences called *haplotypes*. Distinguishing the information contained in both haplotypes when analyzing chromosome sequences poses several new computational issues which collectively form a new emerging topic of Computational Biology known as *Haplotyping*.

This paper is a comprehensive study of some new combinatorial approaches proposed in this research area and it mainly focuses on the formulations and algorithmic solutions of some basic biological problems. Three statistical approaches are briefly discussed at the end of the paper.

1 Introduction

The completion of the Human Genome project [1, 2] has resulted in a draft map of the DNA sequence (which may be thought of as a string over the alphabet {`A,C,G,T`}) present in each human being. At this point one of the main topics of research in Genomics is determining the relevance of all mutations as causes of some genetic diseases.

Mutation in DNA is the principle factor that is responsible for the phenotypic differences among human beings, and SNPs (*single nucleotide polymorphisms*) are the most common mutations, hence it is fundamental to complete a map of all SNPs in the human population. For this purpose a SNP is defined as a position in a chromosome where each one of two (or more) specific nucleotides are observed in at least 10% of the population

*DISCo, via Bicocca degli Arcimboldi 8, Univ. Milano-Bicocca, 20135 - Milano, Italy, bonizzoni@disco.unimib.it

†Dip. Statistica, via Bicocca degli Arcimboldi 8, Univ. Milano-Bicocca, 20135 - Milano, Italy, gianluca.dellavedova@unimib.it

‡DISCo, via Bicocca degli Arcimboldi 8, Univ. Milano-Bicocca, 20135 - Milano, Italy, riccardo.dondi@unimib.it

§Dept. Computer Science, Univ. of California at Riverside, Riverside CA, USA, jili@cs.ucr.edu

[3]. The nucleotides involved in a SNP are called *alleles*. It has been observed that for almost all SNPs only two different alleles are present, in such case the SNP is said *biallelic*, otherwise the SNP is said *multiallelic*. In this survey we will consider exclusively biallelic SNPs.

In diploid organisms, such as humans, each chromosome is made of two distinct copies and each copy is called a *haplotype*. It is known that exactly one haplotype is inherited from the father and the other is from the mother. More precisely, in the absence of *recombinations events*, each haplotype in a child is identical to one of the two haplotypes of each parent. Whenever recombinations occur, a haplotype of the child may consists of portions of both haplotypes of a parent. Recent studies [4, 5] show the block structure in human chromosomes which implies that it is possible to partition a chromosome into blocks where no (or only a few) recombinations have occurred within each block. This observation justifies the fact that different formulations, with or without recombinations, of the biological problem of completing SNP haplotype maps make sense. Furthermore, results in [4, 5] also show that the SNPs within each block induce only a few distinct common haplotypes in the majority of the population, even though the theoretical number of different haplotypes for a block containing n SNPs is exponential in n. The above facts make it interesting to build a SNP haplotype map that consists of the information of haplotype block structure, common haplotypes and their frequencies which, hopefully, could be used to correlate the common haplotypes with common diseases in gene mapping.

Computing a haplotype map requires to determine the possible SNPs combinations that are common in a population, hence it is necessary to analyze data derived by a large scale SNPs screening of single haplotypes in a population. Unfortunately even the most recent technologies are too expensive for large scale analysis or cannot provide good haplotype data from diploid organisms of a large population. Indeed, experimental data only provide the *genotype* for each individual at each SNP site on the chromosome, which is the combined information of the two alleles. For example we are able to know that a SNP occurs at a certain site, and that the two alleles occurring are `A` and `T`, but we are not able to determine to which haplotype the `A` belongs.

This paper focuses on presenting, in a computational framework, some combinatorial problems arising from the following two basic biological issues:

1. to infer haplotypes from genotypes,

2. to infer haplotypes from DNA sequence fragments.

The first problem basically consists of examining the genotypes from an entire population in order to derive the correct haplotypes. Different computational problems may be defined depending on the fact that recombinations are allowed or forbidden and on the fact that some parental relations among the individuals are known or not.

The second biological problem arises in DNA sequencing, where some fragments of two haplotypes are known, and it is desired to compute both haplotypes in their complete form (i.e. the whole sequences).

This paper is organized as follows. In Section 2, some preliminary definitions and notations used in haplotype inference are given. In Section 3 we introduce the problem of inferring haplotypes in a population, firstly by discussing such problem in a general framework in Section 3.1 and then exploiting a natural biological property in Section 3.2.

The problem of haplotype inference given a pedigree is treated in Section 4. In Section 5 some of the recent results regarding the reconstruction of haplotypes from DNA fragments are presented. Finally, Section 6 aims to give the reader some references regarding the use of statistical methods for the haplotyping problem.

2 Preliminary Definitions

We have already pointed out in the introduction that we will restrict ourselves to biallelic SNPs. Without loss of generality we can assume that the values of the two involved alleles of each SNP are always 0 or 1. Since the SNPs are located sequentially on a chromosome, a *haplotype* of length m is a vector $\langle a_1, \cdots, a_m \rangle$ over $\{0,1\}^m$, where each position i is also called a *site* or a *locus*. A *genotype vector*, or simply *genotype*, represents two haplotypes as a sequence of unordered pairs over the set $\{0,1\}$. Each pair represents the nucleotides in a given site, and since the pairs are unordered we are not able to determine the two haplotypes from the genotype alone. For example two haplotypes of length 3 are $\langle 0,1,1 \rangle$ and $\langle 1,0,1 \rangle$ which are combined into the genotype $\langle (0,1),(0,1),(1,1) \rangle$.

Whenever a pair is made of two identical values, then the SNP site is *homozygous*, otherwise it is *heterozygous*. Clearly, by the assumption on the values of the alleles, the pair for a homozygous site is $(0,0)$ or $(1,1)$, while the pair for an heterozygous site is $(0,1)$. Hence a compact representation of the genotype consists of a vector over the alphabet $\{1,0,?\}$, where the first two symbols are used if the site is homozygous, and a ? encodes a heterozygous site. For example, the compact representation of the genotype $\langle (0,1),(1,0),(1,1) \rangle$ is therefore $\langle ?,?,1 \rangle$.

Given a genotype $g = \langle g_1, g_2, \ldots, g_m \rangle$, then a *resolution* of g is a pair $\langle h,k \rangle$ of haplotypes, where $h = \langle h_1, h_2, \ldots, h_m \rangle$ and $k = \langle k_1, k_2, \ldots, k_m \rangle$, such that $h_i = k_i = g_i$ if $g_i \neq ?$ and $h_i, k_i \in \{0,1\}$, $h_i \neq k_i$ if $g_i = ?$. When the above conditions hold we also say that $\langle h,k \rangle$ *resolves* g. Given a genotype g and a haplotype h, h is said *compatible* with g if and only if there exists a haplotype h' such that $\langle h,h' \rangle$ is a resolution of g; in such case the haplotype h' is called the *realization* of g by h, and is denoted by $R(g,h)$. Given a genotype g (a haplotype h, respectively), let us denote by $g[i]$ ($h[i]$) the element of g at site i.

Please notice that, given a genotype g and a haplotype h, there exists exactly one haplotype $R(g,h)$ such that $\langle h,R(g,h) \rangle$ resolves g. Computing $R(g,h)$ is straightforward; in fact for each position i, where $g[i] \neq ?$, $R(g,h)[i] = h[i]$, otherwise $R(g,h)[i] = 1 - h[i]$. The general problem of inferring haplotypes from genotypes can be stated as follows.

Problem 1 (HI). (Haplotype Inference problem)
Input: a set $G = \{g_1, \cdots, g_n\}$ of genotypes.
Output: for each genotype $g \in G$, a pair $\langle h,k \rangle$ of haplotypes resolving g.

The HI problem stated before is actually a metaproblem, in the next sections we will analyze some of the formulations of the general problem that have been proposed in literature.

3 Inferring Haplotype in a Population

Recombination events make a number of problems harder, but experimental data show that human chromosomes can be partitioned into large regions (usually called *blocks*), where no or few recombinations could occur within each block. By definition of a block as a portion of the chromosome where there are few SNPs that account for the differences in the individuals, when we restrict ourselves to the analysis of a specific block in the population, only a relatively small number of distinct haplotypes can be found. The above observations justify some simplifying assumptions that are present in a number of models for inferring haplotypes from genotypes:

- haplotypes consist only of certain portions of the chromosomes or SNP sites [6].
- only a block is considered, consequently no recombinations are allowed [7, 8].

The approaches described in Sections 3.1 and 3.2 make use of these two assumptions. Clearly, exploiting those assumptions depends on a feasible solution to the computational problem of partitioning chromosomes into blocks. The approach proposed in [8] and in [9] faces the problem by computing a partition of the chromosome minimizing the total number of regions or blocks, where each block induces at most a fixed number of distinct haplotypes in the population. This computational problem and other related issues are discussed in [10, 9], where efficient algorithms have been proposed.

When we consider the haplotype inference problem, the alleles at each site i of an individual consist of exactly one allele from each of his parents in the corresponding sites: this behavior is known as *Mendelian Law*, see Fig. 1. When no recombination occurs, each of the two haplotype copies is equal to one of the haplotype copies of its parents, that is all alleles of a haplotype derives from the same haplotype copy of a parent.

If, on the contrary, recombinations occur, then a haplotype can consist of alleles coming from two haplotypes of the same parent. By Mendelian law, the consequence is that the given haplotype derives from two grandparents of the individual. Thus, the parental and grandparental sources of a allele are the basic information to be used to determine and measure recombinations.

Fig. 1 shows a recombination event. A recombination occurs between the second locus and the third locus in the left (paternal) haplotype (the haplotype $1A, 2A, 3B, 4B$) of the child, since under Mendelian Law, the alleles at the first two loci are inherited from the left (paternal) haplotype of the father while the alleles at the last two loci are inherited from the right (maternal) haplotype of the father. Note that no recombination occurs in the right (maternal) haplotype (the haplotype $1D, 2D, 3D, 4D$) of the child, since it is equal to the right (maternal) haplotype of the mother. The number of recombination events in an individual is the total number of such switches of the grandparent source occurring in its haplotypes. In the example of Fig. 1, the number of recombinations is one.

3.1 The Inference Problem: A General Rule

Various methods to infer haplotypes from genotype data have been proposed. Among them the inference method proposed in [11] and later largely discussed in [12] deserves our interest since it is the first approach that points out some basic computational issues related to the haplotype inference problem under a general inference rule.

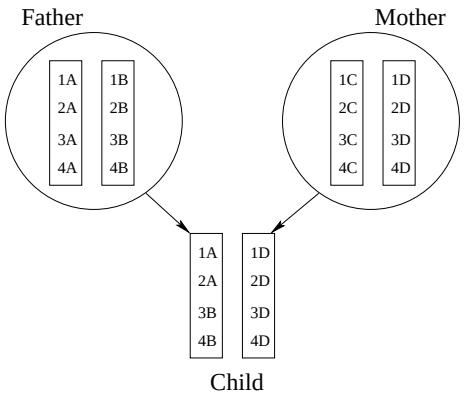

Figure 1: An example of recombination.

The input data of the inference method consists of n individuals and an m-site genotype for each individual. The expected output for each individual genotype is a haplotype pair, among 2^{k-1} different possibilities (where k is the number of heterozygous sites), that resolves the original genotype of the individual.

In [11] a *parsimonious* principle has been used to formalize the HI problem. Such principle has been suggested by empirically observing that a valid solution is usually the one that resolves the largest number of genotypes. Moreover in the same paper a number of experiments suggest the conjecture that there are not two distinct solutions resolving correctly all input genotypes.

Note that, if a genotype g contains only one heterozygous site, then we can infer without ambiguities the two haplotypes that resolve g (the two haplotypes are obtained by computing the only two possible resolutions of the single heterozygous site). Thus a genotype is considered *ambigous* when it contains at least two unresolved sites, that is sites of value ?.

Definition 3.1 (Inference rule). Let $G = \{g_1, \cdots g_m\}$ be a set of genotypes and let H be a nonempty set of haplotypes. The application of the *inference rule* to a haplotype $h \in H$ compatible with a genotype $g \in G$ consists of adding $R(g, h)$ to H and removing g from G.

The computational problem is therefore, given (G, H), finding a sequence of applications of the inference rule that leads to a set H' of haplotypes that resolve all genotypes G (that is all genotypes are removed from G), whenever this is possible. In this case we say that H', with $H' \supseteq H$, resolves G. Example 3.1 shows an instance of the problem.

Example 3.1 (Inference rule). *Let* $G = \{g_1 = 0?0?01, g_2 = 00?00?\}$ *and* $H = \{h_1 = 010101, h_2 = 000101\}$. *The optimal sequence of applications consists of applying the inference rule first to* (g_1, h_1), *and then to* g_2; *this application allows to resolve all genotypes in* G. *In fact* $R(g_1, h_1) = h_3 = 000001$, *after having applied the rule* $G = \{g_2\}$ *and* $H = \{h_1, h_2, h_3\}$. *Now it is possible to apply the rule to* (g_2, h_3), *since* $R(g_2, h_3) = 001000$ *the set* G *becomes empty.*

If we had decided to apply the inference rule first to (g_1, h_2), *the obtained haplotype would not allow to resolve* g_2. *In fact* $R(g_1, h_2) = h_5 = 010001$, *but none of the vectors now in* H *can be used to resolve* g_2.

In [12] a formal framework to analyze and investigate the computational complexity of such problem has been proposed, by stating an optimization problem, whose corresponding decision version is NP-hard. The optimization problem follows:

Problem 2 (MR). (Maximum Resolution problem)
Input: a set G of genotypes and a set H of haplotypes.
Output: a maximum cardinality subset G' of G of genotypes that are removed from G by a sequence of applications of the inference rule starting from G and H.

The computational complexity of some restricted versions of the MR problem is investigated in [12]. Given a set of genotypes G and a set H of haplotypes, H has the *unique expression property w.r.t.* G, in short *UE* property, if for every $g \in G$, there exists at most a pair h_1, h_2 in H such that $R(g, h_1) = h_2$. The consequent problem follows:

Problem 3 (UEMR). (Unique Expression Maximum Resolution problem)
Input: a set G of genotypes and a set H of haplotypes with the UE property.
Output: a maximum-cardinality subset G' of G of genotypes that are removed from G by a sequence of applications of the inference rule starting from G and H that leaves a set H' of haplotypes having the UE property w.r.t. G'.

The proof in [12] that the MR problem is NP-hard makes use of a set H of haplotypes with the *UE* property, thus proving that also the restricted UEMR problem is NP-hard.

In [12] a heuristic for the MR problem is proposed. In particular, the MR problem is reduced through a worst-case exponential time reduction to a new graph problem, which consists of finding some induced subtrees in a graph [12]. A heuristic for this last problem by using integer linear programming is presented.

Some questions related to the MR problem remain open. Mainly, the computational complexity of solving a single genotype from a set containing both genotypes and haplotypes, by iteratively using the inference rule is unknown. Problem 4 is the formalization of the above question which is of fundamental importance, as determining if Problem 4 can be solved in polynomial time, would give some insights on the feasibility of some possible applications of the inference rule that are different from the one suggested in the MR problem.

Problem 4 (SGR). (Single Genotype Resolution problem)
Input: a non empty set H of haplotypes and a distinguished genotype $g \in G$ in a set G of genotypes.
Output: a sequence of applications of the inference rule that resolves a subset of G including g

3.2 The Inference Problem by the Coalescent Model

One of the main drawbacks of the approach presented previously is that no biological assumption has been made, and sometimes biological assumptions allow to restrict the problem so that efficient and more realistic solutions are obtained. Consequently some specific biological models have been introduced recently in the framework of haplotyping. An interesting model has been proposed in [7]: the *coalescent* model, which assumes that the evolutionary history is represented by a rooted tree, where each given sequence labels

one of the leaves of the tree. The *infinite site* model is also assumed, that is at most one mutation can occur in a given site in the whole tree. This last assumption, which forbids recurrent mutations, is suitable to represent the evolutionary history in absence of recombinations and when the basic evolutionary event is changing the value of a SNP site, from 0 to 1. Consequently mutations are directed, that is descendants of individuals in which a mutation has occurred still own the given mutation [13].

The following definition introduces the main combinatorial tool for describing some computational problems related to the coalescent model.

Definition 3.2. Let B be a $n \times m$ $\{0,1\}$-matrix, where each row in B is a binary haplotype and each column i is the n vector of the SNP sites i for the m haplotypes. A *haplotype perfect phylogeny* for B, in short *hpp*, is a rooted tree T with n leaves such that the following properties hold:

1. each leaf of the tree is labeled by a distinct haplotype from B, that is a distinct row of B,
2. each internal edge of T is labeled by at least a SNP site j changing from 0 to 1, while each site labels at most one edge,
3. for each haplotype leaf h, the unique path from the root of T to h specifies exactly all SNP sites that are 1 in h.

Without loss of generality, the root of the phylogeny is assumed to be labelled by $(0, 0, ..., 0)$. Consider now a matrix A, where each row is a genotype, that is A is a $\{0, 1, ?\}$-matrix. Analogously to the case of genotypes vectors, it is possible to give the definition of *realization* of matrix A by a matrix B, that is a matrix B such that each row of A is resolved by a pair of rows of B.

Definition 3.3. A $\{0,1\}$-matrix B is a realization of a $\{0,1,?\}$-matrix A if each row A_j of A is resolved by a pair of rows of B.

The third point of Def. 3.2 implies that each path in the perfect phylogeny T from the root to a haplotype leaf h is a compact representation of the row of matrix B corresponding to h, since it represents all the sites of that row with value 1. Moreover let v be an internal vertex of T, let u be the parent of v and let H_v be the set of all haplotype leaves of the subtree of T that have root in v; then H_v consists of exactly all haplotypes that have value 1 in the SNP sites labeling (u, v). Hence H_v provides a compact representation of column j of matrix B.

In [7, 8] the haplotype inference problem is then stated using the notion of haplotype perfect phylogeny. The basic idea of the common approach in [7, 8] is that n genotypes must be resolved by haplotypes that can be related by a haplotype perfect phylogeny as in Def. 3.2. Formally, the approach described above leads to the following problem as stated in [8].

Problem 5 (PPH). (Perfect Phylogeny Haplotyping problem)
Input: an $n \times m$ matrix A over alphabet $\{0, 1, ?\}$.
Output: a matrix B which is a realization of matrix A and a haplotype perfect phylogeny for B, or decide that such a matrix does not exist.

In [7] the PPH problem is stated by requiring that a realization B of matrix A must be obtained by doubling each row r_j of matrix A, in such a way that rows r_{2j} and r_{2j+1} of B solve row r_j in A. We will call such a realization a *full realization* of matrix A. The definition given above is more general and allows us to define an optimization version of the problem, that derives by applying a parsimonious criterion in inferring the haplotypes: the MPPH problem stated below. Indeed, it seems reasonable to require that the inference process from genotypes should produce a minimum number of distinct haplotypes, as pointed out by the empirical results in [11].

Problem 6 (MPPH). (Minimum Perfect Phylogeny Haplotyping problem)
Input: an $n \times m$ matrix A over alphabet $\{0, 1, ?\}$.
Output: a matrix B which is a realization of matrix A with the smallest number of rows and a haplotype perfect phylogeny for B or decide that such a matrix does not exist.

Given an instance of the PPH problem, a first algorithmic issue concerns the existence of a solution for that instance. Indeed, a haplotype perfect phylogeny induces two relations between pairs of SNP sites labeling edges in the tree, one between siblings and one between an ancestor and a descendant. Those relations do not always allow to find a solution to every instance of the PPH problem. Two sites i, j of an individual haplotype h are related by the ancestor-descendant relation whenever changes 0 to 1 hold in both sites i and j of h: indeed, in the *hpp*, the path from the root to the leaf labeled by h contains two edges labeled i and j. Moreover i and j are 1-0 siblings (0-1 siblings) in an individual haplotype h, whenever the change 0 to 1 occurs in position i of h and not in j (or vice versa, respectively). It is easy to verify that in a *hpp* the two sites i and j that are related in a haplotype by the parenthood relation cannot be 0-1 siblings in a haplotype and 1-0 siblings in another haplotype (see Fig. 3). Formally, this situation is described by the existence of three rows h_1, h_2, h_3 and two columns i, j of a matrix B such that $B[h_1, i] = B[h_1, j] = 1$, while $B[h_2, i] = 1$ and $B[h_2, j] = 0$, $B[h_3, i] = 1$ and $B[h_3, j] = 0$. The submatrix of B induced by i, j and h_1, h_2, h_3 is used [13] to characterize matrices that cannot be represented by a hpp: we call such submatrix the *forbidden* matrix, see Fig. 2.

	i	j
h_1	1	1
h_2	1	0
h_3	0	1

Figure 2: Example of a forbidden matrix M

Lemma 3.1. *Let A be a $n \times m$ matrix over alphabet $\{0, 1\}$. Then, A admits a hpp iff every submatrix of A induced by three rows and a pair of columns is not the forbidden matrix of Fig. 2.*

An analogous of Lemma 3.1 holds for matrices over alphabet $\{0, 1, ?\}$. In a first paper on the PPH problem, a polynomial solution to the problem of computing a full realization of a matrix A based on a reduction to the Graph Realization Problem is proposed [7], while direct algorithms of $O(nm^2)$ time complexity are proposed in [14] and [8]. The

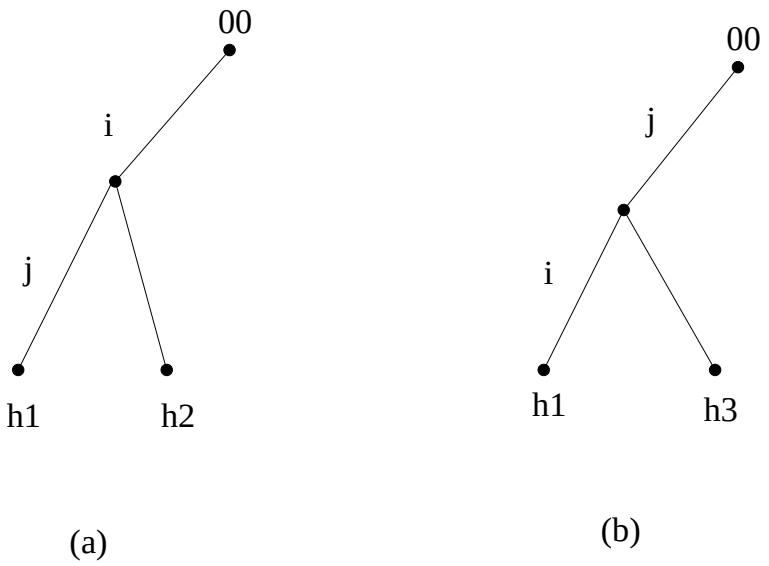

Figure 3: The forbidden matrix M cannot be represented using a perfect phylogeny. In the tree (a) is represented the relation between sites i and j due to the haplotypes h_1 and h_2 of matrix M. In the tree (b) is represented the relation between sites i and j due to the haplotypes h_1 and h_3 of matrix M. Note that in tree (a) i must be an ancestor of j, while in tree (b) j must be an ancestor of i.

complexity of the MPPH problem is still open. An experimental study of the biological validity and relevance of the Coalescent model in haplotype inference is largely discussed in a recent paper [9].

4 Inferring Haplotypes in Pedigrees

In this section, we investigate the HI problem in a pedigree. The difference between population data and pedigree data lies in the relations among the individuals which are regarded independent if taken from a population, while the individuals are related by a parenthood relation if a pedigree is specified. The dependency relationships in pedigree data have implications in two different aspects: 1) a structure (called *pedigree graph*) is imposed over pedigree data, whereas for population data to build such a structure (corresponding to recovering the evolutionary history of the involved haplotypes) might be one of the goals; 2) *Mendelian Law* is assumed, that is each child receives one allele from the father and one from the mother at each site, thus no mutations occur in the pedigree. Consequently Mendelian law can be used to partially resolve some genotypes.

Although the parental information gives some constraints on the reconstruction of haplotypes, there are still too many solutions that are consistent with the genotype data and Mendelian law, especially for biallelic data like SNPs, where in general the probability that more individuals have the same heterozygous genotypes is higher than that on multi-allelic data. Based on the fact that genetic recombinations are rare in human data [4, 5, 15], people believe that haplotypes with fewer recombinations should be preferred in a

haplotype reconstruction [16, 17, 18].

As already pointed out in this paper, the parsimonious principle naturally leads to optimization problems. In this case the computational problem is finding a haplotype configuration with minimum number of recombinants. Again, recombinations make the problem hard; in fact a first formal proof of the NP-hardness of this problem is given in [19] for the general case of pedigree graphs. A heuristic algorithm for this problem and a polynomial exact algorithm for the 0-recombination situation are also presented in that paper. We will discuss those results later, but first we need to formally define the notion of a pedigree graph and several models for the HI problem on pedigree data.

Let us first define the general notion of a pedigree graph, without genotype information that has become widespread in biology.

Definition 4.1. A *pedigree graph* is a weakly connected directed acyclic graph $\mathcal{G} = \langle V, E \rangle$, where $V = M \cup F \cup N$, M stands for the male nodes, F stands for the female nodes, N stands for the mating nodes, and $E = \{e = (u, v)$: $u \in M \cup F$ and $v \in N$ or $u \in N$ and $v \in M \cup F\}$. $M \cup F$ are called the individual nodes. The indegree of each individual node is at most 1. The indegree of a mating node must be 2, with one edge starting from a male node (called father) and the other edge from a female node (called mother), and the outdegree of a mating node must be larger than zero.

In a pedigree, the individual nodes adjacent *to* a mating node (*i.e.* they have edges from the mating node) are called the *children* of the two individual nodes adjacent *from* the mating node (*i.e.* the father and mother nodes, which have edges to the mating node). The individual nodes that have no parents (indegree is zero) are called *founders*. For each mating node, the induced subgraph containing the father, mother, mating, and children nodes is called a *nuclear family*. A *parents-offspring trio* (or simply *trio*) consists of two parents and one of their children. A *mating loop* is a cycle in the graph obtained from G when the directions of edges are ignored.

An equivalent definition of pedigree graph that points out the combinatorial nature of the representation is the following:

Definition 4.2. A *pedigree graph* $\mathcal{G}$ is a weakly connected directed acyclic graph $\langle V, E \rangle$, where each vertex has indegree 2 or 0.

In Def. 4.2 only individual nodes are represented and their gender information are outfitted. The founders are the vertices without incoming edges and a trio is any subgraph of G with 3 vertices u, v, w where (u, v) (here (u, v) stands for an arc from u to v since we are talking about the direct graph) and (w, v) are the only arcs of the pedigree graph; a trio is denoted by the triple $< u, v, w >$. In the following we will mean Def. 4.2 when we refer to the notion of pedigree graph. The only substantial difference with Def. 4.1 regards the definition of mating loops; when Def. 4.2 is considered, a mating loop consists of two distinct paths from a vertex x to a vertex y. Figure 4 shows side by side an example pedigree according to the two definitions of pedigree graph, in particular to the left the common representation of pedigree graphs is reported.

The above pedigree graph definition is very general and there are several restricted versions defined as follows.

Definition 4.3. A pedigree tree $\mathcal{T}$ is a pedigree graph without mating loops.

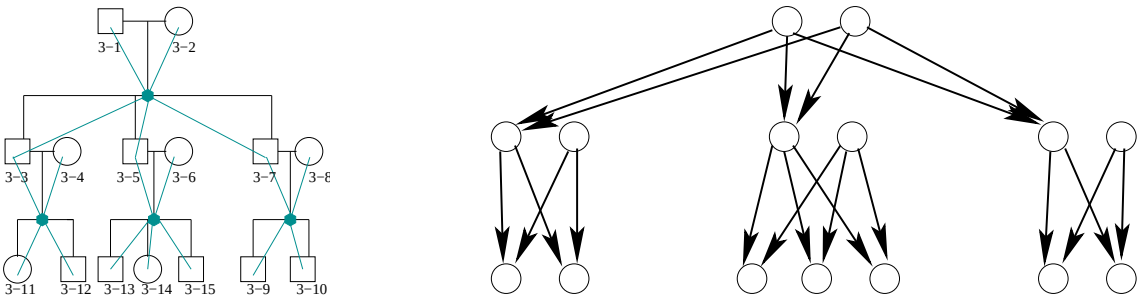

Figure 4: A pedigree with 15 members. (Left) A square represents a male node and a circle represents a female node, and a solid (round) node represents a mating node. The children (*e.g.* 3-3, 3-5 and 3-7) are placed under their parents (*e.g.* 3-1 and 3-2). (Right) The representation of the same pedigree according to Def. 4.2

.

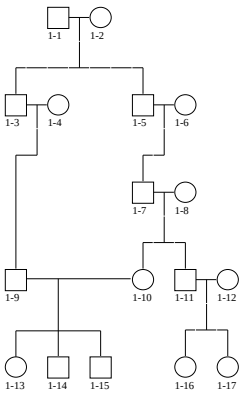

Figure 5: A pedigree with 17 members and a mating loop without showing the mating nodes.

We further distinguish pedigree trees by restricting the number of mating partners each individual node of the tree can have. Given a pedigree tree T, for any vertex v let $\mathcal{P}(v)$ be the set of vertices x such that (x, v) is an arc of T. Then T is a *single-mating* pedigree tree, if for any two vertices v, w, the two sets $\mathcal{P}(v)$ and $\mathcal{P}(w)$ are disjoint or the same set; otherwise the pedigree tree is called *multi-mating*.

The definition of pedigree graph introduced so far allows us to describe the structure of the parental relations. We still need a notion that allow us to relate the actual genotypes to the structure of a pedigree graph.

Definition 4.4. A *genotyped pedigree graph* is a pedigree graph $\mathcal{G}$ where each individual vertex is labeled by a m-site genotype vector.

With a slight abuse of language by pedigree graph we will denote both a labeled (genotyped) and an unlabeled pedigree graph and in the case of labeled pedigree, we use the node itself to denote the genotype vector associated to the node itself. Recall that for any node u, given the genotype vector $< u[1], u[2], \ldots, u[m] >$, then $u[i] \in \{0, 1, ?\}$. If $u[i] \neq ?$, then we say that $u[i]$ is defined. Similarly a *haplotyped* pedigree graph is a pedigree graph where each individual vertex is labeled with two haplotypes.

Definition 4.5. A genotyped pedigree graph $\mathcal{G}$ is *g-valid* if the following *consistency rules* hold. Given a trio $\langle u, v, w \rangle$, then for each i, $1 \leq i \leq m$:

- if $u[i] \neq w[i]$ are both defined, then $v[i] = ?$,
- if $u[i] \neq w[i]$ and only one of $u[i]$ or $w[i]$ is defined, then $v[i] = w[i]$ or $v[i] = u[i]$,
- if $u[i] = w[i] = ?$, then $v[i]$ can be $0, 1$ or $?$,
- $u[i] = v[i] = w[i]$, otherwise.

Then, given a g-valid genotyped pedigree graph $\mathcal{G}$, we are interested in a haplotyped pedigree graph with same sets of vertices and edges, such that the haplotypes labeling a vertex v resolve the genotype of v in $\mathcal{G}$ and each haplotype (paternal/maternal) in a child is inherited from each parent (father/mother) with/without recombinations. In such case we will say that the haplotyped graph is a *realization* of $\mathcal{G}$.

Problem 7 (PHI). (Pedigree Graph Haplotype Inference problem)
Input: a g-valid genotyped pedigree graph $\mathcal{G}$.
Output: a haplotyped pedigree graph which is a realization of $\mathcal{G}$.

Even though a realization of the genotyped graph explicitly associates the two haplotypes of each child to the ones of its parents, a realization might not unambiguously determine for each allele on a given haplotype what is the haplotype of its corresponding parent from which it derives. For instance let us consider Fig. 6, we know that the first allele of a haplotype of the child comes from her mother, but does it come originally from a grandmother or a grandfather? In order to disambiguate those situations we introduce the notion of GS value of each allele which states if it is inherited from parent's paternal haplotype or maternal haplotype.

The introduction of a GS value for each allele is necessary due to the presence of recombinations, because in this case each haplotype is not the exact copy of one haplotype

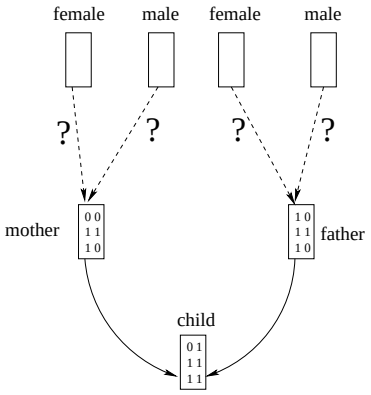

Figure 6: The child in a pedigree tree: it is known that the first allele of a haplotype of the child comes from her mother, but it is not known if it is originally from a grandmother or a grandfather.

of a parent. More formally, given v the child in a trio $< u, v, w >$ where u and w are respectively the father and the mother, and assuming that $h_1(x)$, $h_2(x)$ denote the pair of haplotypes resolving the genotype at node x, $GS(h_1(v[i])) = 0$ if $h_1(v[i]) = h_1(u[i])$ and $GS(h_1(v[i])) = 1$ if $h_1(v[i]) = h_2(u[i])$. Similarly we can define $GS(h_2(v[i]))$.

We are now able to exploit the GS values to count the number of recombinants as follows: for any two alleles that are at adjacent sites and from the same haplotype, they induce a *recombinant* (or *recombination event*) if their GS differ. Formally, given haplotype h, a recombination occurs at site i of h iff $GS(h[i]) \neq GS(h[i+1])$. According to the parsimonious principle we can derive an optimization problem using the above notions.

Problem 8 (GMRHI). (General Minimum Recombination Haplotype Inference problem)
Input: a g-valid genotyped pedigree graph $\mathcal{G}$.
Output: a realization of $\mathcal{G}$ minimizing the number of recombination events.

In [19] it has been proved that GMRHI (originally called MRHC problem in their paper) is in general NP-hard. Actually the proof in the paper shows that even with two sites ($m = 2$), the GMRHI problem is still NP-hard in the case of the pedigree graph allowing mating loops. The reduction is from a stronger version of tridimensional matching which is NP-hard [20]. An iterative heuristic is proposed based on the assumption that recombination events are rare. In [19] also a polynomial time exact algorithm for the restriction of the problem where no recombination occurs is given, that is:

Problem 9 (ZRHI). (Zero-recombinant Haplotype Inference problem)
Input: a pedigree graph $\mathcal{G}$.
Output: A realization of $\mathcal{G}$ such that no recombination events occurs if such realization exists, otherwise report that no realization exists.

The algorithm first identifies all the necessary constraints based on Mendelian Law and the zero recombinant assumption, and represents them using a system of linear equations over the cyclic group Z_2. By using a simple method based on Gaussian elimination, all possible feasible haplotype configurations could be obtained. Since in the case of human

pedigrees, mating loops are very rare, it becomes interesting to give the following more restricted formulations of the general problem.

Problem 10 (MPT-MRHI). (Multi-mating Pedigree Tree Minimum Recombination Haplotype Inference problem)
Input: a multi-mating g-valid genotyped pedigree tree T.
Output: a realization of T minimizing the number of recombination events.

Problem 11 (SPT-MRHI). (Single-mating Pedigree Tree Minimum Recombination Haplotype Inference problem)
Input: a single-mating g-valid genotyped pedigree tree T.
Output: a realization of T minimizing the number of recombination events.

More recently it has been proved in [21] that even SPT-MRHI is NP-hard by a reduction from MAX-CUT [20]. Unlike the NP-hardness proof of GMRHI in [19], where only two sites are needed, the proof of SPT-MRHI does require an unbounded number of sites. It is an interesting problem to study the computational complexity of SPT-MRHI problem when the number of children of any two individual nodes is bounded by a constant.

5 Inferring Haplotypes from Fragments

The Human Genome project has successfully produced a draft version of the DNA present in human beings, through a sequencing process. A different kind of problem arises when the sequencing process aims also to reconstruct haplotypes. Roughly speaking, the sequencing process is made of two phases: in the first phase a number of *fragments* are obtained, where each fragment is a small piece (a few hundreds of bases long) of the examined DNA. Afterwards all such fragments are merged into a chromosome, for example via shotgun sequencing [2]. In its original formulation the sequencing problem assumes that all fragments come from only one copy of the chromosomes of a DNA strand. But this assumption is too weak, in fact not only the fragments come from both copies, but it is not possible to associate the fragments to the copy of the chromosome from which they originate. Thus computing the two sequences that form the haplotypes becomes more challenging. Moreover, the presence of errors in the fragments that need to be assembled makes harder the problem of reconstructing the original sequence from fragments when SNPs are considered.

In [6], the problem of reconstructing the pair of haplotype sequences from fragments of a human chromosome is investigated by introducing some formulations. First of all, each location of a SNP over a fragment is assumed to be known, that is the genomic sequence is thought of as a sequence of positions with one (not a SNP site) or two (a SNP biallelic site) symbols associated to each position. The sequence of SNPs sites along a fragment is described by a vector over a binary alphabet $\{\mathtt{0}, \mathtt{1}\}$ that is used to denote the two distinct alleles of SNP sites contained in the fragment.

The formal definitions of the problems introduced in [6] share the fact that the instance is always a $n \times m$ matrix M where each entry $M[i, j]$ is `0` or `1` or `-`, and where the i-th row corresponds to the i-th *fragment*, conversely the j-th column corresponds to the j-th SNP. When $M[i, j] =$ `-` then the i-th fragment does not *cover* the j-th SNP, which means that the allele of the fragment in position j is unknown: the entry `-` of matrix M

is called a *hole*. Moreover we will say that two fragments *conflict* with each other if they disagree on a SNP covered by both fragments, that is fragments i and j of matrix M are in conflict whenever there exists a SNP site k such that $M[i,k] \neq M[j,k]$ and both $M[i,k]$ and $M[j,k]$ are not holes. A conflict occurring at a given SNP site denotes the fact that the two fragments come from two distinct haplotypes, or more precisely, the SNP site is heterozygous and has two distinct alleles on the two chromosome copies.

If the matrix M represents m fragments obtained from one pair of haplotypes and fragments do not contain errors in the alleles reported in the matrix, conflicts among the fragments can be used to derive a partition of fragments into two sets, each one containing the non conflicting fragments from the same haplotype.

Thus, we define a matrix M *error free* iff there exists a partition of rows of M into two matrices M_1 and M_2 such that both M_1 and M_2 do not contain conflicting fragments.

In [6], the following biological problem is investigated: the reconstruction of the two different haplotypes of a chromosome from SNP values of fragments represented by an instance matrix that is not necessarily error free. It must be pointed out that, even when the matrix is error free, there may be more than one pair of haplotypes whose fragments give the same instance matrix: in this case the reconstruction of the original haplotypes for the chromosome from a SNP matrix is not solvable, as the instance matrix does not contain enough information to disambiguate among all possible haplotypes.

Hence, we restrict ourselves to the problem of finding a pair of haplotypes whose fragments are represented by a given matrix. Then the problem consists of assigning each fragments to a copy of the chromosome.

When the matrix is error free, this problem can be solved easily by computing the *fragment conflict graph*, whose vertices are the fragments and the pair (i,j) is an edge iff the two fragments i and j conflict. Indeed, the graph must be bipartite, with each shore representing all the fragments that are in one of the two copies of the chromosome.

Please notice that the solution to the problem of inferring haplotypes from fragments obtained from the fragment conflict graph is unique iff the graph is connected, in which case the solution consists of a unique pair of haplotypes.

	1	2	3	4	5	6
1	0	1	-	0	-	0
2	0	-	1	-	-	0
3	1	0	-	-	1	1
4	-	1	1	-	0	-
5	1	-	-	1	-	1

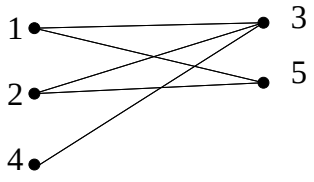

Figure 7: An error-free matrix and its associated fragment conflict graph

In the presence of errors the problem becomes more complex; in fact we look for the minimum number of modifications to the instance matrix to make it error free. Different operations on the matrix may be defined and each of such operations leads to a specific computational problem, as pointed out in [6] where the following problems have been introduced:

	1	2	3	4	5	6
1	0	1	-	0	-	0
2	0	-	1	-	-	0
3	1	0	-	-	1	1
4	-	1	1	-	0	-
5	1	1	-	1	-	1

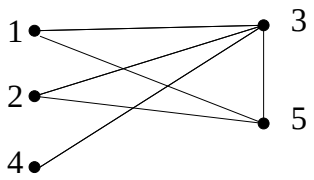

Figure 8: A matrix with errors and its associated fragment conflict graph

Problem 12 (MFR). (Minimum Fragment Removal)
Compute the minimum set of rows to remove from the matrix so that the matrix is error free.

Problem 13 (MSR). (Minimum SNP Removal)
Compute the minimum set of columns to remove from the matrix so that the matrix is error free.

Problem 14 (LHR). (Longest Haplotype Reconstruction)
Compute a set of rows to remove from the matrix so that the matrix is error free and the sum of the lengths of the inferred haplotypes is maximized.

Problem 15 (MEC). (Minimum Error Correction)
Compute the minimum number of corrections on the entries of the matrix so that the resulting matrix is error free.

A single fragment may (but not must) cover SNP sites that are consecutive on the fragment: in such case the fragment is *gapless*. A fragment has k gaps if it covers $k+1$ blocks of consecutive SNPs.

Of particular interests are the cases of 0 or 1 gaps, as these cases are common when sequencing. The classical shotgun sequencing procedure deals with probes that are consecutive nucleotides from a DNA strand. Recent advances of the sequencing technology has made the production of mate pairs feasible, where a mate pair is made of two probes from the same copy of a chromosome and with a distance between the two probes that is approximately known. Representing a mate pair with a 1-gap fragment is immediate. In Table 1 are summarized some results described in [6, 22].

The presence of gaps in fragments and also the number of holes is strictly related to the computational complexity of the above problems (Table 1 reports all known results about the complexity of such problems). Indeed, whenever fragments are gapless the problem is in general polynomial. The 1-gap case is relevant since the presence of at most one gap in each fragment is a sufficient condition to make hard the MFR problem. It is an interesting question to investigate the 1-gap case for the MSR and the other problems.

In the following we describe a graph based method to solve the MSR problem for gapless instance matrices. This method uses the notion of conflict among SNP sites. Given a SNP matrix M, two SNP sites i and j are in conflict in M iff i and j assumes both $0, 1$ values in M and there exist two fragments x and y such that the submatrix

Problems	Gaps	Exact	Approximation
MFR	≤ 1	NP-hard	APX-hard[1]
	0	Solvable in time $O(m^2n + m^3)$	
	k holes	Solvable in time $O(2^{2k}nm^2 + 2^{3k}m^3)$	
MSR	≤ 2	NP-hard	APX-hard[2]
	0	Solvable in time $O(mn^2)$	
	k holes	Solvable in time $O(mn^{2k+2})$,	
LHR	k holes	?	?
	0	Polynomial time	
MEC	n	NP-hard	?
	0	?	

Table 1: Known results for some problem on fragments

induced by rows x and y and columns i and j has three symbols of one type and one of the opposite. SNP conflicts of a SNP matrix M are represented by the *SNP conflict graph* having vertices the SNPs and an edge for each pair (i, j) of conflicting SNPs.

An interesting property relates the SNP matrices without gaps to error free matrices.

Lemma 5.1. *A matrix without gaps is error free iff it has no SNP conflicts.*

By using the above property, that can be easily verified, the problem MSR reduces to solve the Min Vertex Cover problem over the SNP conflict graph, as making a matrix error free means to remove from such graph the minimum number of vertices (matrix columns) so that the graph has no edges. In [6] it is proved that whenever a matrix is without gaps then the SNP conflict graph is perfect [23]. Being the Min Vertex Cover problem on perfect graphs polynomial-time solvable, MSR can be solved efficiently via the above reduction.

On the other hand, when the fragments contain some gaps the reduction does not work as illustrated in the following example.

	1	2	3	4
1	0	-	0	0
2	1	1	-	0
3	-	0	1	1

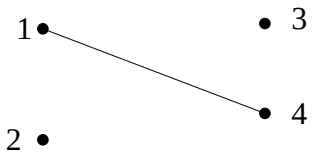

Figure 9: A matrix with gaps and the corresponding SNP conflict graph

Example 5.1. *Assume that M is the matrix of Fig. 9 and G the corresponding SNP conflict graph. Then the vertex 4 is a minimum vertex cover, but the matrix M' obtained from M by removing the column 4 is not error free.*

All algorithms for the gapless cases are via dynamic programming; the main idea is that it is possible to infer the optimal solution from the optimal solution on a submatrix

obtained by removing a SNP column (MSR) or a fragment row (MFR). Let M be a SNP matrix, we denote with $M[1...k]$ the submatrix of M formed by the first k rows of M.

Let us consider the MFR problem. Since the matrix M is gapless, each row of M, consists of a sequence of defined values encoding the actual fragment, preceded or followed by some (possibly zero) holes. For each row f of M we denote by $l(f)$ and $r(f)$ respectively the leftmost and the rightmost positions (SNP) of f that are not holes. The algorithm will exploit the fact that all positions between $l(f)$ and $r(f)$ must be defined. A first preprocessing step of the algorithm is to sort the rows of M in increasing order of $l(f)$, that is $l(i) \leq l(j)$, whenever $i < j$.

A dynamic programming algorithm mainly consists of showing that an optimal solution of an instance can be computed from an optimal solution of some induced subinstance, in our case we will compute the optimal solution over the matrix M by exploiting the optimal solution where the "rightmost" fragment is removed from M. Consequently we will show how to solve the instance $M[1..k]$ from an optimal solution of $M[1..k-1]$ (we recall here that an optimal solution is a minimum-size set of rows that must be removed to obtain an error-free matrix).

The algorithm computes the value $D[i, j, k]$ that is the optimum over instance $M[1...k]$ with the additional restriction that fragment i (respectively j) has the maximum value of $r(i)$ (resp. $r(j)$) among all $r(f)$ for all fragments placed on the first (resp. the second) haplotype. See Fig. 10 for an illustrative example.

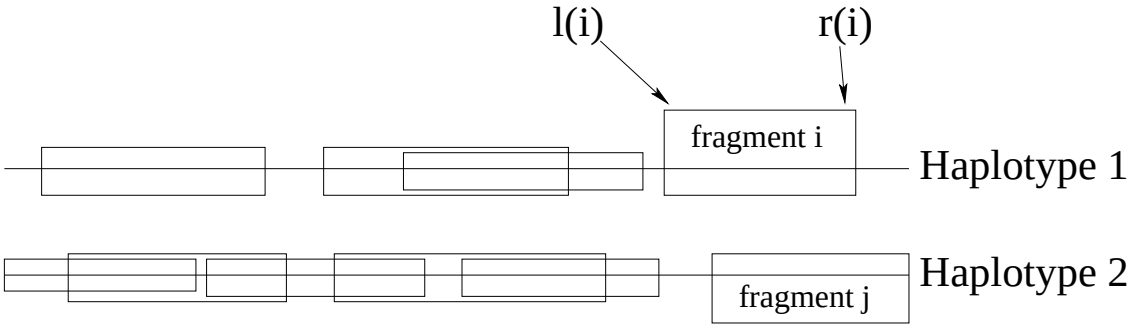

Figure 10: Example of fragments placed on the haplotypes by the algorithm in [22]

Moreover for each row f considered we denote with $OK(f)$ the set of rows with index $g < f$ that agree with row f, that is rows representing fragments that may be on the same haplotype copy. Now we can explicitly show the recurrence that allows to solve the problem. The following possible cases must be considered:

1. $D[i, j, 0] := 0$
2. $k > i$, $k > j$; $D[i, j, k] :=$

$$\begin{cases} D[i,j,k-1] & \text{if } r(k) \leq r(j) \text{ and rows } k \text{ and } j \text{ agree} \\ D[i,j,k-1] & \text{if } r(k) \leq r(i) \text{ and rows } k \text{ and } i \text{ agree} \\ D[i,j,k-1]+1 & \text{otherwise;} \end{cases}$$

3. $k = i$; $D[k, j, k] := min_{h \in OK(k), r(h) \leq r(k)}\{D[h, j, k-1]\}$

4. $k = j$; $D[i,k,k] := min_{h \in OK(k), r(h) \leq r(k)}\{D[i,h,k-1]\}$

The optimal solution will be $min_{h,k}\{D[h,k;m]\}$. In [22] the dynamic programming algorithm described has also been extended as a fixed-parameter algorithm for the same problem, where the parameter is the number of holes contained in the fragments, that is the number of unresolved symbols between the leftmost and the rightmost resolved symbol of each fragment. The algorithm can also be extended to the class of matrices where the columns can be permuted so that in each row the (`0/1`) symbols appear consecutively. The last problem is also a version of the consecutive ones problem, for which a polynomial-time algorithm has been described in [24].

6 A Glimpse over Statistical Methods

In this section we present some basic aspects of the application of statistical methods to tackle the haplotype inference problem. These methods are among the most used by biologists. Indeed, a certain number of results on haplotype inference from real data have been produced based on some statistical models. Anyway here we will only briefly review some of these algorithms, since this survey mainly focuses on combinatorial approaches for haplotype inference. Moreover, some combinatorial problems discussed here aim to address some biological issues that are different from the ones attacked using the algorithms based on statistical models.

The introduction of statistical models is mainly due to the presence of some shortcomings of the method introduced in [11]. In fact, the method in [11] requires an initial set of resolved haplotypes and it also highly depends upon the order by which haplotypes are resolved. The approach of [12, 25] lowers the relevance of the dependency on the order, hence making the whole procedure more reliable. The main idea of statistical models is that haplotypes have an unknown distribution in the target population and the observed genotypes of each individual are simply combination of two haplotypes randomly drawn from the population. The goal of statistical haplotype inference is thus to estimate the haplotype frequencies and the haplotypes of each individual can be easily inferred based on the haplotype frequencies under some biological assumptions (like random mating assumption). Two different formulations of the haplotype inference problem, namely, Maximum-Likelihood inference [26] and Bayesian inference [27, 28] have been investigated and will be briefly discussed here.

Problem 16 (FHI). (Frequencies Haplotype Inference problem)
Input: a sample G of genotypes.
Output: the set of haplotype frequencies $\{h_1, h_2, ..., h_n\}$ (where n is the number of all possible haplotypes) that maximize the likelihood function of observing the genotype sample G.

Problem 17 (BFHI). (Bayesian Frequencies Haplotype Inference problem)
Input: a sample G of genotypes and a priori distribution of the frequencies of haplotypes.
Output: the posterior distribution of the haplotype frequencies given the sample G.

The FHI problem is tackled in [26], where an Expectation-Maximization algorithm (EM) has been proposed to estimate the haplotype frequencies that maximize the likelihood function of genotype sample. It is well known that the general framework of EM

algorithm [29] is to find the maximum-likelihood estimator(s) by iteratively executing the E-step and M-step until convergence in the presence of missing data. Let H_t denote the set of haplotype frequencies and G_t denote the set of probabilities of all the genotypes at time t. The EM algorithm works by arbitrarily assigning an initial value of H_0 (a possible initial set of frequency values is the one corresponding to the assumption that all possible haplotypes are equiprobable). Based on H_0, the expectation of an observing genotype can be easily calculated, which is one of the elements in G_1. The expected genotype frequencies in G_1 are used in turn to estimate the haplotype frequencies at the M-step resulting in a new H_1. Iterate the two steps until convergence (*i.e.*, the difference between H_{t+1} and H_t is smaller than a predefined value.). At each iteration, the solution H_t is improved in the M-step by maximizing the likelihood function of the genotype sample. Different initial values can be taken in order to increase the possibility to obtain a global optimal solution.

The BFHI problem is tackled by an iterative stochastic-sampling strategy, the pseudo Gibbs sampler (PGS) [27], that makes use of the Markov chain Monte Carlo method with the assumption of a coalescent model. A Gibbs sampler iteratively samples a pair of compatible haplotypes for each genotype conditional on the genotypes G and on other haplotypes, and uses these values to update the frequencies of haplotype distribution. This iterative algorithm produces haplotype frequencies $h_1, ..., h_m$ as they were sampled from the desired posterior distribution of the haplotype frequencies given the sample genotypes G.

The two methods described above cannot handle satisfactorily a large number of SNPs, or missing data. These issues are addressed by the method proposed in [28], where a robust Bayesian procedure has been introduced. The method in [28] makes use of the biological model also used in [26], but imposes no assumptions on the population history (comparing to the coalescent model used in [27]). In particular the method introduces a divide-and-conquer technique so that a larger number of haplotypes can be studied. This algorithm partitions the genotypes into units, where each unit has maximum length of 8 loci. The method first constructs a set of most probable partial haplotypes compatible with each unit using the Gibbs sampler. Two adjacent units are then combined to construct a set of the most probable partial haplotypes that are compatible with the genotypes of the two units. The algorithm recursively combines the partial haplotypes until the whole haplotype is created. A detailed comparison of different statistical methods and the combinatorial method in [11] can be found in [28], which is based on an experimental study.

We conclude this section by observing that, since there is an exponential number of possible haplotype solutions for a given set of genotypes, the statistical methods may have to analyze an exponential number of haplotypes in order to generate the solution of the FHI and BFHI problems. Even equipped with advanced numerical methods like EM algorithm and Gibbs sampler, such statistical methods are still very time consuming. Thus, the application of these methods is restricted to a small number of individuals in a population; moreover the maximum number of individuals tractable decreases as the number of SNPs increases. This limitation is not present in other combinatorial approaches based on polynomial algorithmic solutions.

7 Acknowledgements

J.L. is supported by NSF grant CCR-9988353.

References

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

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

[3] N. Patil, A.J. Berno, and et al. Blocks of limited haplotype diversity revealed by high-resolution scanning of human chromosome 21. *Science*, 294(5547):1669–1670, 2001.

[4] M. Daly, J. Roux, S. Schaffer, T. Hudson, and E. Lander. Fine-structure haplotype map of 5q31: implicatins for gene-based studies and genomic ld mapping. 2001.

[5] S. B. Gabriel, S. F. Schaffner, H. Nguyen, J. M. Moore, J. Roy, B. Blumenstiel, J. Higgins, M. DeFelice, A. Lochner, M. Faggart, S. N. Liu-Cordero, C. Rotimi, A. Adeyemo, R. Cooper, R. Ward, E. S. Lander, M. J. Daly, and D. Altshuler. The structure of haplotype blocks in the human genome. *Science*, 296(5576):2225–2229, 2002.

[6] G. Lancia, V. Bafna, S. Istrail, R. Lippert, and R. Schwartz. SNPs problems, complexity and algorithms. In *Proc. 9th European Symp. on Algorithms (ESA)*, pages 182–193, 2001.

[7] D. Gusfield. Haplotyping as perfect phylogeny: Conceptual framework and efficient solutions. In *Proc. 6th Annual Conference on Research in Computational Molecular Biology (RECOMB)*, pages 166–175, 2002.

[8] E. Halperin, E. Eskin, and R. M. Karp. Efficient reconstruction of haplotype structure via perfect phylogeny. *Journal of Bioinformatics and Computational Biology*, to appear.

[9] E. Halperin, E. Eskin, and R. M. Karp. Large scale reconstruction of haplotypes from genotype data. In *Proc. 7th Annual Conference on Research in Computational Molecular Biology (RECOMB)*, pages 104–113, 2003.

[10] K. Zhang, M. Deng, T. Chen, Michael S. Waterman, and F. Sun. A dynamic programming algorithm for haplotype block partitioning. *Proceedings of the National Academy of Sciences USA*, 99(11):7335–7339, 2002.

[11] A. Clark. Inference of haplotypes from pcr-amplified samples of diploid populations. *Molecular Biology and Evolution*, 7(2):111–122, 1990.

[12] D. Gusfield. Inference of haplotypes from samples of diploid populations: Complexity and algorithms. *Journal Of Computational Biology*, 8(3):305–323, 2001.

[13] D. Gusfield. *Algorithms on Strings, Trees and Sequences: Computer Science and Computational Biology*. Cambridge University Press, Cambridge, 1997.

[14] V. Bafna, D. Gusfield, G. Lancia, and S. Yooseph. Haplotyping as perfect phylogeny: A direct approach. *Journal of Computational Biology*, to appear.

[15] L. Helmuth. Genome research: Map of human genome 3.0. *Science*, 5530(293):583–585, 2001.

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

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

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

[19] J. Li and T. Jiang. Efficient rule-based haplotyping algorithms for pedigree data. In *Proc. 7th Annual Conference on Research in Computational Molecular Biology (RECOMB)*, pages 197–206, 2003.

[20] M.R. Garey and D.S. Johnson. *Computer and Intractability: A Guide to the Theory of NP-completeness*. W. H. Freeman, 1979.

[21] K. Doi, J. Li and T. Jiang. Minimum recombinant haplotype configuration on tree pedigrees. Accepted by 3rd international Workshop on Algorithms in Bioinformatics (WABI), Hungary, 2003.

[22] R. Rizzi, V. Bafna, S. Istrail, and G. Lancia. Pratical algorithms and fixed-parameter tractability for the single individual snp haplotyping problem. In *Proceedings of Algorithms in Bioinformatics, Second International Workshop (WABI 2002)*, pages 29–43, 2002.

[23] M. Grötschel, L. Lovasz, and A. Schrijver. A polynomial algorithm for perfect graphs. *Annals of Discrete Mathematics*, 21:325–356, 1984.

[24] K. S. Booth and G. S. Lueker. Testing for the consecutive ones property, interval graphs, and graph planarity using pq-tree algorithms. *Journal of Computer and System Sciences*, 13(3):335–379, 1976.

[25] S. Orzack, D. Gusfield, and V.P. Stanton. The absolute and relative accuracy of haplotype inferral methods and a consensus approach to haplotype inferral. In *51st Annual Meeting of the American Society of Human Genetics*, 2001.

[26] L. Excoffier and M. Slatkin. Maximum-likelihood estimation of molecular haplotype frequencies in a diploid population. *Molecular Biology and Evolution*, 12(5):921–927, 1995.

[27] M. Stephens, N.J. Smith, and P. Donnelly. A new statistical method for haplotype reconstruction from population data. *American Journal of Human Genetics*, 68:978–989, 2001.

[28] T. Niu, Z.S. Qin, X. Xu, and J.S. Liu. Bayesian haplotype inference for multiple linked single-nucleotide polymorphisms. *American Journal of Human Genetics*, 710:157–169, 2002.

[29] T. M. Mitchell. *Machine Learning*. McGraw Hill, New York, 1987.

Authors' Biographies

Paola Bonizzoni is Associate Professor of Computer Science at the Università di Milano-Bicocca in Milan. She received her Master Degree in Computer Science from the Universita' di Milano in 1988 and her Ph. D in Computer Science from the Università di Milano-Torino in 1993. She has been Assistant Professor in Computer Science at the Università di Milano from 1993 to 1999. She has been Visiting Research Associate at the University of Colorado at Boulder (USA).

Gianluca Della Vedova has been appointed as an assistant professor at the Department of Statistics, Università di Milano-Bicocca in 2001. He holds a Ph.D. and a M.Sc. in Computer Science (Università di Milano). His research interests focus on the design of combinatorial algorithms in Bioinformatics and Graph theory. He has published several papers in Bioinformatics and Theoretical Computer Science international journals.

Riccardo Dondi is a Ph.D candidate in Computer Science, at Università di Milano-Bicocca. In 1999 he has received his M.Sc. in Computer Science at Università di Milano. His research area is Computational Biology, in particular the design of algorithms and the study of the computational complexity of some biological problems, mainly the construction and comparison of evolutionary trees. His more recent focus is on haplotype inference problems.

Jing Li currently is a Ph.D candidate in the Department of Computer Science and Engineering at University of California - Riverside. He received a B.S. in Statistics from Peking University, Beijing, P.R.China in July 1995 and a M.S., in Statistical Genetics, from Creighton University in Aug. 2000.

His recent research interest includes Bioinformatics and computational molecular biology, algorithms and statistical genetics. He has published several papers in Bioinformatics and statistical genetics conferences and journals.