Haplotyping for Disease Association: A Combinatorial Approach

Giuseppe Lancia, R. Ravi, and Romeo Rizzi

Abstract—We consider a combinatorial problem derived from haplotyping a population with respect to a genetic disease, either recessive or dominant. Given a set of individuals, partitioned into healthy and diseased, and the corresponding sets of genotypes, we want to infer "bad" and "good" haplotypes to account for these genotypes and for the disease. Assume, for example, that the disease is recessive. Then, the resolving haplotypes must consist of *bad* and *good* haplotypes so that 1) each genotype belonging to a diseased individual is explained by a pair of bad haplotypes and 2) each genotype belonging to a healthy individual is explained by a pair of haplotypes of which at least one is good. We prove that the associated decision problem is NP-complete. However, we also prove that there is a simple solution, provided that the data satisfy a very weak requirement.

Index Terms—Combinatorial haplotyping, disease association, dominant disease, recessive disease.

1 Introduction

A *single nucleotide polymorphism* (SNP, pronounced "snip") is a site of the human genome showing a statistically significant variability within a population. Apart from very rare exceptions, at each SNP, only two nucleotides (out of A, T, C, and G) are observed and they are called the SNP *alleles*. SNPs are the predominant form of human polymorphism and their importance can hardly be overestimated. They are widely used in therapeutic, diagnostic, and forensic applications and a SNP consortium exists with the goal of designing a detailed SNP map for the human genome [13], [10].

Humans are *diploid* organisms, i.e., their DNA is organized in pairs of chromosomes. For each pair of chromosomes, one chromosome copy is inherited from the father and the other copy is inherited from the mother. For a given SNP, an individual can be either *homozygous* (i.e., possess the same allele on both chromosomes) or *heterozygous* (i.e., possess two different alleles). The values of a set of SNPs on a particular chromosome copy define a *haplotype*. In Fig. 1, we illustrate a simplistic example of three individuals and four SNPs. The alleles for SNP 1 in this example are C and G. Individual 1, in this example, is heterozygous for SNPs 1, 2, and 3 and homozygous for SNP 4. His or her haplotypes are CCCT and GAGT.

Haplotyping an individual consists of determining his or her two haplotypes, for a given chromosome. With the larger availability in SNP genomic data, recent years have seen the birth of many new computational problems related to haplotyping (see [9] for a survey on haplotyping). These problems are motivated by the fact that it is economically infeasible to determine the haplotypes experimentally. On the other hand, there is a cheap experiment that can determine the (less informative) *genotypes*. A genotype provides information about the multiplicity of each SNP allele, i.e., for each SNP, a genotype specifies if an individual is heterozygous or homozygous (in the latter case, it also specifies the allele). Since genotypes are much cheaper to obtain than haplotypes, a natural solution for haplotyping has been to define an inference problem to be solved algorithmically: Compute the correct haplotypes from the genotypes.

When retrieving haplotypes from genotypes, there is an inherent ambiguity that comes from heterozygous sites. At each heterozygous site, to retrieve the haplotypes, one has to decide how to distribute the two allele values on the two chromosome copies. *Resolving* (or *explaining*) a genotype requires determining the two haplotypes that yield the genotype. Given a set of genotypes, the general (computational) *haplotyping* problem requires determining a set of haplotypes such that each genotype is explained by two haplotypes. Due to their importance, haplotyping problems have been and are being extensively studied, under many objective functions (each with specific biological motivations). Popular formulations include 1) *pure parsimony*, which attempts to minimize the total number of distinct haplotypes used to resolve a given set of genotypes. This variant of the problem is APX-hard and several optimization approaches were proposed for its solution [8], [12], [2], [11], [14]. *Clark's rule* [3] is a common heuristic toward this end. 2) *Perfect phylogeny* haplotyping [1], [6], [5] attempts to resolve the genotypes by a set of haplotypes that admit a perfect phylogeny on them.

One of the main reasons why haplotypes are so important and heavily studied is that they are very useful in diagnostic and medical applications since they are related to the presence/absence of genetic diseases. In a very simplistic way, a genetic disease can be considered as a malfunctioning of a specific gene. A gene does not function

- *G. Lancia and R. Rizzi are with the Dipartimento di Matematica e Informatica, University of Udine, Via delle Scienze 206, 33100 Udine, Italy. E-mail: {lancia, rrizzi}@dimi.uniud.it.*
- *R. Ravi is with The Tepper School of Business, Carnegie Mellon University, Pittsburgh, PA 15213. E-mail: rravi@andrew.cmu.edu.*

Manuscript received 10 Apr. 2007; revised 26 July 2007; accepted 27 Aug. 2007; published online 3 Oct. 2007.
For information on obtaining reprints of this article, please send e-mail to: tcbb@computer.org, and reference IEEECS Log Number TCBB-2007-04-0042. Digital Object Identifier no. 10.1109/TCBB.2007.70255.

Hapl. 1, paternal:	taggtcc**C**tatttt**C**ccaggcgc**C**gtatacttcgacggg**T**ctata
Hapl. 1, maternal:	taggtcc**G**tatttt**A**ccaggcgc**G**gtatacttcgacggg**T**ctata
Hapl. 2, paternal:	taggtcc**C**tatttt**A**ccaggcgc**G**gtatacttcgacggg**T**ctata
Hapl. 2, maternal:	taggtcc**G**tatttt**C**ccaggcgc**G**gtatacttcgacggg**C**ctata
Hapl. 3, paternal:	taggtcc**C**tatttt**A**ccaggcgc**G**gtatacttcgacggg**T**ctata
Hapl. 3, maternal:	taggtcc**G**tatttt**A**ccaggcgc**C**gtatacttcgacggg**C**ctata

Fig. 1. The haplotypes of three individuals, with four SNPs.

properly when its encoding sequence has been mutated with respect to one of its correct versions. Since each gene is present in two copies (a paternal and a maternal copy), it may be the case that either copy is malfunctioning. A genetic disease is called *recessive* if a person shows the symptoms of the disease only when *both* gene copies are malfunctioning. For a recessive disease, one can be a healthy carrier, when one copy is malfunctioning but the other is working properly. Examples of recessive diseases are cystic fibrosis and sickle cell anemia. A genetic disease is called *dominant* if a person shows the symptoms of the disease when *at least one* gene copy is malfunctioning. Examples of dominant diseases are Huntington's disease and Marfan's syndrome.

In this paper, we study the problem of haplotyping the genotypes of a population consisting of healthy and diseased individuals. The haplotypes correspond to the gene sequences. Let us call a haplotype corresponding to a sequence encoding a working gene "good" and a haplotype for which the encoded gene is malfunctioning "bad". Henceforth, in the context of haplotyping with respect to a disease, there should exist a coloring into good and bad of the haplotypes inferred from the genotypes that accounts for the disease. In particular, assuming that the disease under study is recessive,

1. in each pair of haplotypes inferred from a healthy genotype, at least one of the haplotypes is good or
2. in each pair of haplotypes inferred from a diseased genotype, both haplotypes are bad.

Finding the haplotypes and their coloring into good and bad was listed in [7] as an interesting open problem in computational biology. Note that, by reversing the role of good versus bad and of healthy versus diseased, the same conditions can be used to study a dominant disease.

1.1 Our Results

The formal definition of the above problem, called Haplotyping for Disease Association (HDA), will be given in the next section. In this paper, we prove the following main results:

Theorem 1. *The problem HDA is NP-complete even when restricted to instances in which there are no two individuals, one healthy and one diseased, with the same genotype.*

Theorem 2. *If each genotype has at least two heterozygous sites, then the instance is feasible and the problem is polynomially solvable.*

Theorem 3. *If there exists at least one haplotype that can possibly be used to explain all genotypes (i.e., all genotypes are compatible), then the problem is polynomially solvable.*

Let us briefly comment on these results. The negative result of Theorem 1 is, in practice, dominated by the positive result of Theorem 2. In real-life instances, it is expected that each genotype has several heterozygous sites so that each instance should be feasible and polynomially solvable. Moreover, in populations where the most frequent alleles have a much higher frequency than the least frequent ones, it can be shown that genotypes are usually compatible, so Theorem 3 applies.

The existence of a partition of the haplotypes into good and bad is a biologically necessary condition for a solution to be correct and, as such, it has been posed as a combinatorial condition for the haplotyping problem. In our proof of Theorem 2, however, we employ a strictly combinatorial argument that derives a mathematically correct but, most likely, biologically meaningless solution (i.e., bad haplotypes will be such that their alleles, once represented as binary values, sum up to a given remainder modulo 3). Therefore, an important contribution of this work is to show that the partitioning condition alone is too weak to capture the essence of the genetic disease under study and that more constraints must be imposed if the solution is to in fact explain the genetic disease.

1.2 Haplotyping as a Combinatorial Problem

Let n be the number of SNPs we consider. Arbitrarily fix a binary encoding of the two alleles for each SNP (e.g., call the least frequent allele "0" and the other "1"). Once the encoding has been fixed, each haplotype is represented by a binary n-vector.

For a haplotype h, we denote by $h[i]$ the value of its ith component. Given two haplotypes h' and h'', their sum is a vector $h' \oplus h''$, where the binary operator $\oplus$ is defined componentwise as

$$(h' \otimes h'')[i] := \begin{cases} 0, & \text{if } h'[i] = h''[i] = 0, \\ 1, & \text{if } h'[i] = h''[i] = 1, \\ 2, & \text{if } h'[i] \neq h''[i]. \end{cases}$$

Haplotype 1, paternal:	0 1 0 1	
	2 2 2 1	Genotype 1
Haplotype 1, maternal:	1 0 1 1	
Haplotype 2, paternal:	0 0 1 1	
	2 2 1 2	Genotype 2
Haplotype 2, maternal:	1 1 1 0	
Haplotype 3, paternal:	0 0 1 1	
	2 0 2 2	Genotype 3
Haplotype 3, maternal:	1 0 0 0	

Fig. 2. Haplotypes and corresponding genotypes.

Any vector $g \in \{0,1,2\}^n$ is called a *genotype*. Therefore, for two haplotypes h' and h'', $g = h' \oplus h''$ is a genotype. The above notation, which defines the sum of two haplotypes, yielding a genotype, is the standard used in all literature on haplotyping problems.

Definition 1 (resolution). *For g, a genotype, a pair of haplotypes $\{h', h''\}$ such that $g = h' \oplus h''$ is a* resolution *of g. The haplotypes h' and h'' are said to* resolve *g. Let $\mathcal{G}$ be a set of genotypes and $\mathcal{H}'$ and $\mathcal{H}''$ be two sets of haplotypes. We say that $(\mathcal{H}', \mathcal{H}'')$ resolves $\mathcal{G}$ if, for every $g \in \mathcal{G}$, there exist $h_1 \in \mathcal{H}'$ and $h_2 \in \mathcal{H}''$ such that $g = h_1 \oplus h_2$. We call such a resolution a* resolution in $(\mathcal{H}', \mathcal{H}'')$ of g.

Definition 2 (ambiguity). *Let g be a genotype. Each position i such that $g[i] = 2$ is called an* ambiguous position. *By $n_2(g)$, we denote the number of ambiguous positions of g. A genotype is* ambiguous *if it has more than one resolution, i.e., if $n_2(g) \geq 2$.*

In the biological interpretation, genotype entries with a value of 0 or 1 correspond to homozygous SNP sites, while ambiguous positions correspond to heterozygous sites. In Fig. 2, we illustrate a case of three individuals, showing their haplotypes and genotypes.

Definition 3 (compatibility). *A haplotype h is* compatible *with a genotype g if $g[i] = h[i]$ for $g[i] \neq 2$. Two genotypes g and g' are* compatible *if $g[i] = g'[i]$ whenever $g[i] \neq 2$ and $g'[i] \neq 2$ (i.e., if g and g' share at least one compatible haplotype).*

For instance, if $h = 0100$, $g^1 = 0212$, $g^2 = 1222$, and $g^3 = 2211$, then h is compatible with g^1 but not with g^2 or g^3. Moreover, g^1 and g^2 are not compatible, while g^1 and g^3 are.

Clearly, a genotype can be resolved only by compatible haplotypes. Given g and h compatible with g, it is easy to compute the *complement* of h with respect to g. This is the unique haplotype, denoted by $[g \ominus h]$, for which $h \oplus [g \ominus h] = g$.

Definition 4 (shorthand notation). *We denote by $\mathbf{0}$ a genotype/haplotype of all 0s. For $S \subseteq \{1, \ldots, n\}$, we denote by $\mathbf{1}_S$ a genotype/haplotype that is 0 everywhere except at components in S, where it is 1. We shortwrite $\mathbf{1}_i$ for $\mathbf{1}_{\{i\}}$ and $\mathbf{1}_{ij}$ for $\mathbf{1}_{\{i,j\}}$. We denote by $\mathbf{2}_S$ a genotype that is 0 everywhere except at components in S, where it is 2.*

For instance, if $n = 5$, it is $\mathbf{1}_3 = 00100$, $\mathbf{1}_{1,5} = 10001$, and $\mathbf{2}_{1,2,4} = 22020$.

In the problem studied in this paper, the data describe a population of m individuals, of which m_H are healthy and m_D are diseased. An instance consists of two (not necessarily disjoint) sets of genotypes: $\mathcal{G}_H$ (the genotypes from healthy individuals) and $\mathcal{G}_D$ (the genotypes from diseased individuals). Let $\mathcal{G}$ be the multiset obtained by the union of $\mathcal{G}_H$ and $\mathcal{G}_D$. Each genotype appears in G once or twice (when one healthy and one diseased individual have the same genotype). $\mathcal{G}$ can also be viewed as an $m \times n$ matrix with entries in {0, 1, 2}, partitioned in submatrices $\mathcal{G}_D$ of m_D distinct rows (genotypes) and $\mathcal{G}_H$ of m_H distinct rows, with $m = m_D + m_H$.

We consider a recessive disease and we seek two *disjoint* sets of haplotypes, $\mathcal{H}_B$ ("bad" haplotypes) and $\mathcal{H}_G$ ("good" haplotypes), that provide a feasible solution to the following problem:

[HAPLOTYPING FOR DISEASE ASSOCIATION (HDA)]
INSTANCE: A set $\mathcal{G}_D$ of *diseased* genotypes. A set $\mathcal{G}_H$ of *healthy* genotypes.
PROBLEM: Find a set of haplotypes, partitioned into $\mathcal{H}_G$ and $\mathcal{H}_B$ such that

(i) For each $g \in \mathcal{G}_D$, there is a resolution of g in $(\mathcal{H}_B, \mathcal{H}_B)$;
(ii) For each $g \in \mathcal{G}_H$, there is a resolution of g in $(\mathcal{H}_G, \mathcal{H}_G \cup \mathcal{H}_B)$.

Notice that there may be infeasible instances of HDA. The smallest such example, for $n = 1$, is to assume that the genotype $g = 2$ is diseased. Then, the only possible haplotypes, i.e., 0 and 1, must both be bad and, hence, there cannot be healthy genotypes. More interesting examples of infeasible instances can be shown, but, basically, they all result from the same basic issue: If some haplotypes are forced to be bad or forced to be good, then the instance may be infeasible. In the following sections, we will prove that this is the only cause of infeasibility and, when each genotype has more than one resolution, the instance is always feasible.

Example. Consider the following instance of HDA:

$$\mathcal{G}_H = \begin{pmatrix} g^1 \\ g^2 \end{pmatrix} = \begin{pmatrix} 2222 \\ 0220 \end{pmatrix}, \qquad \mathcal{G}_D = \begin{pmatrix} g^3 \\ g^4 \\ g^5 \end{pmatrix} = \begin{pmatrix} 2222 \\ 1222 \\ 2101 \end{pmatrix}.$$

Notice that there are two individuals, one healthy and one diseased, with the same genotype. A possible solution is

$$\mathcal{H}_G = \begin{pmatrix} h^1 \\ h^2 \end{pmatrix} = \begin{pmatrix} 0100 \\ 0010 \end{pmatrix}, \qquad \mathcal{H}_B = \begin{pmatrix} h^3 \\ h^4 \\ h^5 \end{pmatrix} = \begin{pmatrix} 1010 \\ 0101 \\ 1101 \end{pmatrix},$$

with the following resolutions: $g^1 = h^2 \oplus h^5$, $g^2 = h^1 \oplus h^2$, $g^3 = h^3 \oplus h^4$, $g^4 = h^3 \oplus h^5$, and $g^5 = h^4 \oplus h^5$.

2 The Hardness of HDA

In this section, we prove that HDA is a difficult problem. In particular, we prove the following NP-completeness result:

Theorem 1. *Problem HDA is NP-complete even when restricted to instances in which $\mathcal{G}_D$ and $\mathcal{G}_H$ are disjoint.*

The proof is based on a reduction from 3SAT, which was shown to be NP-complete in [4].

Let $\langle U, \mathcal{C}\rangle$ be an instance of 3SAT, where $U = \{u_1, u_2, \ldots, u_n\}$ is a finite set of Boolean variables and $\mathcal{C} = C_1, C_2, \ldots, C_m$ is a collection of clauses, each containing precisely three literals over U. A literal over U is either a variable u_i in U (positive literal) or its negation $\overline{u}_i$ (negated literal). In 3SAT, we are asked to find whether there exists a truth assignment $\Phi : U \mapsto \{true, false\}$ such that each clause in $\mathcal{C}$ contains at least one literal that evaluates to $true$ under Φ.

Given the 3SAT-instance $\langle U, \mathcal{C}\rangle$, we construct an instance of HDA as follows:

We introduce one SNP for each literal over U so that there is a one-to-one correspondence between $\hat{U} := \{u_1, \overline{u}_1, u_2, \overline{u}_2, \ldots, u_n, \overline{u}_n\}$ and the SNPs in the constructed instance of HDA. In our intended interpretation of the reduction, a literal $\hat{u}_i \in \hat{U}$ should be read as $true$ if and only if the haplotype $\mathbf{1}_{\{\hat{u}_i\}}$ is in $\mathcal{H}_B$. It remains to be specified how to define $\mathcal{G}_D$ and $\mathcal{G}_H$. First, the genotype $\mathbf{0}$ is placed in $\mathcal{G}_D$. Next, for each $i = 1, 2, \ldots, n$ and so as to enforce that at most one of the two literals u_i and $\overline{u}_i$ can carry the $true$ value, we place the genotype $\mathbf{1}_{\{u_i, \overline{u}_i\}}$ in $\mathcal{G}_D$ and the genotype $\mathbf{2}_{\{u_i, \overline{u}_i\}}$ in $\mathcal{G}_H$. Finally, for each $c = 1, 2, \ldots, m$ and assuming that $\hat{u}_i$, $\hat{u}_j$, and $\hat{u}_k$ are the three literals occurring in clause C_c, in order to represent the constraint that at least one of these three literals should evaluate to $true$, we place the genotype $\mathbf{2}_{\{\hat{u}_i, \hat{u}_j, \hat{u}_k\}}$ in $\mathcal{G}_D$ and the genotype $\mathbf{1}_{\{\hat{u}_i, \hat{u}_j, \hat{u}_k\}}$ in $\mathcal{G}_H$.

The description of the reduction is complete. It should be clear that the reduction can be performed in polynomial time.

The following two lemmas conclude our NP-completeness proof:

Lemma 2. *Assume that $\langle U, \mathcal{C}\rangle$ admits a satisfying truth assignment. Then, the instance $\langle \mathcal{G}_D, \mathcal{G}_H\rangle$ of the HDA constructed above is a Yes instance.*

Proof. Let Φ be a satisfying truth assignment. First, place $\mathbf{0}$ in $\mathcal{H}_B$. Next, for each $i = 1, 2, \ldots, n$, place $\mathbf{1}_{\{u_i, \overline{u}_i\}}$ in $\mathcal{H}_B$. Moreover, if $\Phi(u_i) = true$, then place $\mathbf{1}_{\{u_i\}}$ in $\mathcal{H}_B$ and $\mathbf{1}_{\{\overline{u}_i\}}$ in $\mathcal{H}_G$; otherwise, do the contrary. For each $c = 1, 2, \ldots, m$, where $\hat{u}_i$, $\hat{u}_j$, and $\hat{u}_k$ are the three literals occurring in clause C_c, and assuming that $\Phi(\hat{u}_i) = true$ as we can also do by possibly renaming the three literals (and remembering that Φ is a satisfying truth assignment), place $\mathbf{1}_{\{\hat{u}_j, \hat{u}_k\}}$ in $\mathcal{H}_B$ and $\mathbf{1}_{\{\hat{u}_i, \hat{u}_j, \hat{u}_k\}}$ in $\mathcal{H}_G$. The reader is invited to check that $(\mathcal{H}_B, \mathcal{H}_B)$ resolves $\mathcal{G}_D$ and $(\mathcal{H}_G, \mathcal{H}_G \cup \mathcal{H}_B)$ resolves $\mathcal{G}_H$. Moreover, $\mathcal{H}_G$ and $\mathcal{H}_B$ are disjoint. □

Lemma 3. *Assume that the instance $\langle \mathcal{G}_D, \mathcal{G}_H\rangle$ of the HDA constructed above is a Yes instance. Then, the 3SAT-instance $\langle U, \mathcal{C}\rangle$ we started from admits a satisfying truth assignment.*

Proof. Let $\mathcal{H}_G$ and $\mathcal{H}_B$ be two disjoint haplotype sets such that $(\mathcal{H}_B, \mathcal{H}_B)$ resolves $\mathcal{G}_D$ and $(\mathcal{H}_G, \mathcal{H}_G \cup \mathcal{H}_B)$ resolves $\mathcal{G}_H$. Clearly, $\mathbf{0} \in \mathcal{H}_B$ since $\mathbf{0} \in \mathcal{G}_D$. Similarly, since $\mathbf{1}_{\{u_i, \overline{u}_i\}} \in \mathcal{G}_D$, $\mathbf{1}_{\{u_i, \overline{u}_i\}} \in \mathcal{H}_B$ for every $i = 1, 2, \ldots, n$. Since $\mathbf{2}_{\{u_i, \overline{u}_i\}} \in \mathcal{G}_H$, both $\mathbf{1}_{\{u_i\}}$ and $\mathbf{1}_{\{\overline{u}_i\}}$ belong to $\mathcal{H}_G \cup \mathcal{H}_B$ and at least one of them belongs to $\mathcal{H}_G$. Consider the truth assignment $\Phi : U \mapsto \{true, false\}$ defined by $\Phi(u_i) = true$ if and only if $\mathbf{1}_{\{\overline{u}_i\}} \in \mathcal{H}_G$. We claim that Φ is a satisfying truth assignment. Indeed, consider the generic clause C_c and let $\hat{u}_i$, $\hat{u}_j$, and $\hat{u}_k$ be the three literals occurring in C_c. Clearly, $\mathbf{1}_{\{\hat{u}_i, \hat{u}_j, \hat{u}_k\}} \in \mathcal{H}_G$ since $\mathbf{1}_{\{\hat{u}_i, \hat{u}_j, \hat{u}_k\}} \in \mathcal{G}_H$. Since $\mathbf{2}_{\{\hat{u}_i, \hat{u}_j, \hat{u}_k\}} \in \mathcal{G}_D$, it must hold that, for at least one of these three literals, say, $\hat{u}_i$, we have $\mathbf{1}_{\{\hat{u}_i\}}, \mathbf{1}_{\{\hat{u}_j, \hat{u}_k\}} \in \mathcal{H}_B$. Without loss of generality, let $\mathbf{1}_{\{\hat{u}_i\}} \in \mathcal{H}_B$ and, from our assignment rule above, it follows that $\Phi(\neg\hat{u}_i) = false$. We have thus argued that, in each clause C_c, there is at least one literal $\hat{u}_i$ such that $\Phi(\hat{u}_i) = true$, that is, Φ is a satisfying truth assignment, as claimed. □

3 Polynomially Solvable Instances of HDA

3.1 All Genotypes with at Least Two Heterozygous Sites

The following theorem says that all instances are feasible, except when some genotypes have only one resolution.

Theorem 4. *Assume that, for each genotype $g \in \mathcal{G}$, $n_2(g) \geq 2$. Then, the instance is feasible and a feasible solution can be readily obtained.*

Proof. Let us divide the set of all possible haplotypes into three classes, $\mathcal{H}_0$, $\mathcal{H}_1$, and $\mathcal{H}_2$, depending on the remainder in the sum of the bits divided by three. More formally,

$$\mathcal{H}_i = \left\{h : \left(\sum_{i=1}^{n} h_i\right) \bmod 3 = i\right\}, \qquad \text{for } i = 0, 1, 2. \quad (1)$$

Similarly, divide all of the genotypes into three classes and define $\mathcal{G}_0$, $\mathcal{G}_1$, and $\mathcal{G}_2$ as follows:

$$\mathcal{G}_i = \{g \in \mathcal{G} : n_2(g) \bmod 3 = i\}. \quad (2)$$

We first describe how to resolve genotypes that have $2 \leq n_2(g) \leq 4$ and, then, we show how the same type of solution can be applied to all genotypes. The main idea is the following: We are going to make all haplotypes in $\mathcal{H}_0$ good. We then need to show that, for each healthy genotype, there is a resolution that uses at least a haplotype in $\mathcal{H}_0$ and, for each diseased genotype, there is a resolution that does not use a haplotype in $\mathcal{H}_0$.

For a genotype g, let us define $\tilde{h}(g)$ as the haplotype obtained by replacing each 2 in g with a 0. Given a g with $2 \leq n_2(g) \leq 4$, each resolution will consist of two haplotypes h' and h'' that "contain" $\tilde{h}(g)$ and of which one has k more 1s and the other has $n_2(g) - k$ more 1s than $\tilde{h}(g)$ does. Depending on g, all of the possibilities for the pair $\{k, n_2(g) - k\}$ are described in Table 1.

Notice that there is always a resolution in which we can add 0, 1, or 2 (modulo 3) to the parity of $\tilde{h}(g)$ and, hence, we can make the resulting haplotype belong to any class $\mathcal{H}_i$ we want. Furthermore, notice that there is always a resolution that skips, in *both* haplotypes, adding 0, 1, or 2 to the parity of $\tilde{h}(g)$ and, hence, we can make the resulting haplotypes *not* belong to any class $\mathcal{H}_i$ we want.

TABLE 1
Resolutions Sorted by the Number of 1s They Introduce

num. 2s	k	$n_2(g)-k$	mod 3	
$n_2(g)=2$	0	2	0 2	(a)
	1	1	1 1	(b)
$n_2(g)=3$	0	3	0 0	(c)
	1	2	1 2	(d)
	0	4	0 1	(e)
$n_2(g)=4$	1	3	1 0	(f)
	2	2	2 2	(g)

In particular, Table 2 shows how we should resolve the genotypes.

Table 2 should be read as follows: For each genotype g, we locate the row that corresponds to the parity classes of g and of $\tilde{h}(g)$ and the column that corresponds to g being healthy or diseased. Then, the table returns a letter that identifies one of the six ways of resolving the genotype described in Table 1. For example, assume that $g = 0212111$ is a healthy genotype. Then, $g \in \mathcal{G}_2$ (since $n_2(g) = 2$) and $\tilde{h}(g) = 0010111 \in \mathcal{H}_1$ so that g should be resolved in the (a) way, i.e., creating two haplotypes of which, in one, we replace the two 2s with two 0s (obtaining 0010111) and, in the other, we replace them with two 1s (obtaining 0111111).

We now verify that the solution obtained by declaring all haplotypes in $\mathcal{H}_0$ good is feasible. Note that, for each healthy genotype, whenever $\tilde{h}(g) \in \mathcal{H}_0$, we always choose a resolution that adds zero 1s in one of the haplotypes, i.e., one of the resolving haplotypes is $\tilde{h}(g)$ itself. If, on the other hand, $\tilde{h}(g) \notin H_0$, we choose a resolution that adds to the parity of $\tilde{h}$ the number of bits needed to obtain a haplotype in $\mathcal{H}_0$. As far as the diseased genotypes are concerned, as we said before, there is always a resolution that, in both haplotypes created, does not add i to the parity of $\tilde{h}(g)$ (where $i = 0, 1, 2$). Therefore, if $\tilde{h}(g)$ belongs to, say, $\mathcal{H}_j$, we need to *not* add to its parity $3 - j$; otherwise, we would get a haplotype in $\mathcal{H}_0$. However, we can always do this, and how to do it is reported in the table.

Finally, consider the general case of g with $n_2(g) > 4$. Then, assume that $n_2(g) = 3p + q$, with $2 \le q \le 4$, so that $g \in \mathcal{G}_q$. Then, replace a set S of $3p$ 2s in g with 1s, obtaining g', and solve g', with $n_2(g') = q$. This yields a solution that has two haplotypes h' and h'' such that $g' = h' \oplus h''$. Let h^0 be obtained by h' by replacing the $3p$ 1s in h' with 0s. Note that the parity of h^0 and h' modulo 3 is the same. Using h^0 in place of h', we have a resolution of g, i.e., $g = h^0 \oplus h''$. □

3.2 Compatible Genotypes

Next, we present a second condition for polynomial-time feasibility.

Theorem 5. *Assume that all genotypes in $\mathcal{G}$ are mutually compatible. Then, the problem is polynomially solvable.*

TABLE 2
How to Resolve the Genotypes

num. 2s		healthy	diseased
$g \in \mathcal{G}_2$	$\tilde{h}(g) \in \mathcal{H}_0$	(a)	(b)
	$\tilde{h}(g) \in \mathcal{H}_1$	(a)	(b)
	$\tilde{h}(g) \in \mathcal{H}_2$	(b)	(a)
$g \in \mathcal{G}_0$	$\tilde{h}(g) \in \mathcal{H}_0$	(c)	(d)
	$\tilde{h}(g) \in \mathcal{H}_1$	(d)	(c)
	$\tilde{h}(g) \in \mathcal{H}_2$	(d)	(c)
$g \in \mathcal{G}_1$	$\tilde{h}(g) \in \mathcal{H}_0$	(e,f)	(g)
	$\tilde{h}(g) \in \mathcal{H}_1$	(g)	(e,f)
	$\tilde{h}(g) \in \mathcal{H}_2$	(e,f)	(g)

Proof. Since all genotypes are mutually compatible, we can assume (possibly after swapping the 1s into 0s along some columns) that the matrix $\mathcal{G}$ has only entries in {0, 2}. Two possibilities arise: Either each row of $\mathcal{G}_D$ has two or more 2s or there exists a row in $\mathcal{G}_D$ with less than two 2s. In the first case, there is a trivial solution: Declare the haplotype **0** good and all other haplotypes bad; we can then resolve each genotype in $\mathcal{G}_H$ with a pair that uses **0** and each genotype in $\mathcal{G}_D$ with a pair that does not use **0**.

Now, consider the case in which a nonempty set B_1 of rows of $\mathcal{G}_D$ has one or no 2s in each row. Let L be the set of columns where nonzeros appear in these rows. Call these the *left* columns and the other columns are the *right* columns. Note that each row in B_1 has a unique resolution and that the forced resolving haplotypes must be bad. Furthermore, each resolution of this type contains the haplotype **0**. Without loss of generality, we replace each diseased genotype in B_1 with the haplotypes that it implies. This way, each diseased genotype has zero or at least two 2s and zero or exactly one 1. We now decompose the data into (at most) six parts: B_1, B_0, and B_2 will be a partition of $\mathcal{G}_D$ and G_1, G_0, and G_2 will be a partition of $\mathcal{G}_H$. The partition of $\mathcal{G}_D$ is described as follows: B_1 has already been defined. B_0 contains all rows of $\mathcal{G}_D$ that have all of their 2s in left columns (note that this implies that each row in B_0 has at least two 2s in left columns). B_2 contains all rows of $\mathcal{G}_D$ that have at least one 2 in a right column (and, in case of precisely one single 2 in a right column, there is also a 2 in a left column by definition of L).

As for $\mathcal{G}_H$, the partition is described as follows: Note that, for each genotype $g \in \mathcal{G}_h$, it must be that $n_2(g) > 1$; otherwise, there is no feasible solution. In G_1, we put all of the rows that have a single 2. Note that none of these rows should have its 2 in a left column; otherwise, the problem is infeasible (the unique resolution of the row would have both haplotypes in B_1 and is hence bad). In G_0, we put the rows of $\mathcal{G}_H$ that have all of their 2s in the left columns (this implies that each row in G_0 has at least two 2s in left columns). Finally, in G_2, we put the remaining rows (that have at least a 2 in a right column). See Fig. 3.

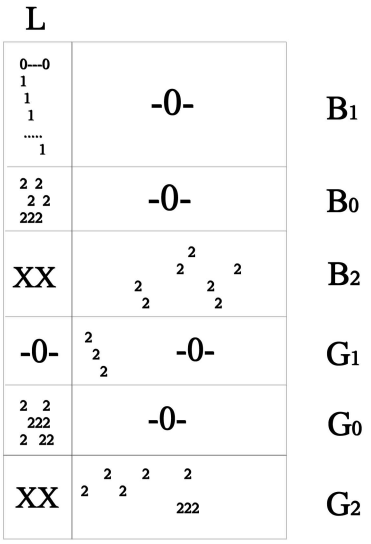

Fig. 3. The generic decomposition. Submatrices labeled XX can contain any type of rows made of 0s and 2s.

Now, resolve the instance as follows: For the diseased, for each $g \in \mathcal{G}_D$, if $g \in B_1$, g is resolved as $g \oplus g$; if $g \in B_0$ and g has exactly two 2s, say, in positions i and j, then resolve g as $\mathbf{1}_i \oplus \mathbf{1}_j$. In all of the remaining cases, resolve g as $\mathbf{0} \oplus [g \ominus \mathbf{0}]$. For the healthy: For each $g \in G_1$, say, $g = \mathbf{2}_i$, resolve g as $\mathbf{0} \oplus \mathbf{1}_i$ and declare $\mathbf{1}_i$ good; for each $g \in G_0$, let $\{i, j\}$, $i \neq j$, be two left columns in which g has a 2. Then, resolve $g = \mathbf{1}_{ij} \oplus [g \ominus \mathbf{1}_{ij}]$ and declare $\mathbf{1}_{ij}$ good. Finally, for each $g \in G_2$, say, g has a nonzero in right column i, resolve g as $g = \mathbf{1}_i \oplus [g \ominus \mathbf{1}_i]$ and declare $\mathbf{1}_i$ good.

We now argue that the solution proposed is feasible. By construction, the bad haplotypes are defined as follows: They either have no 1s or have exactly one 1 in a left column (haplotypes created from B_1 and haplotypes created from rows of B_0 with exactly two 2s in left columns), they have three or more 1s, all in left columns (haplotypes created from rows of B_0 with more than two 2s), or they have at least two 1s, of which at least one is in a right column (haplotypes created from B_2).

Now, each healthy genotype that is not in G_0 has at least one haplotype in its resolution with exactly one 1 and, furthermore, this 1 is in a right column. Since the only bad haplotypes with exactly one 1 have the 1 in a left column, they cannot conflict with the resolution.

Therefore, the only possibility for a conflict comes from healthy genotypes in G_0. Note that such a genotype is resolved by using a haplotype that has exactly two 1s, both in left columns. However (see above), there is no bad haplotype that has exactly two 1s, both in left columns. Hence, there is no conflict possible. □

3.3 Preprocessing

As we have shown in Section 2, HDA is NP-complete, so there can be instances whose feasibility is hard to verify. Because of Theorem 4, these instances must have some forced haplotypes in each solution (caused by nonambiguous genotypes). We should then start any attempt to determine the feasibility of a general instance in which there are nonambiguous genotypes with the following cascade of implications:

1. Let B and G be the set of forced bad and good haplotypes. Initially, B consists of all haplotypes compatible with genotypes in $\mathcal{G}_D$ that have < 2 ambiguous sites. Similarly, G consists of all haplotypes compatible with genotypes in $\mathcal{G}_H$ that have no ambiguous sites. Remove from $\mathcal{G}_H$ and $\mathcal{G}_D$ all genotypes used to derive B and G.
2. If $B \cap G \neq \emptyset$, then stop; the problem is infeasible. Otherwise, loop through 3-6 until, for a complete iteration, neither B nor G changes.
3. Let $\mathcal{G}'$ be the subset of genotypes in $\mathcal{G}_H$ that can be obtained as $h \oplus h'$, with $h, h' \in B$. If, for any $g \in \mathcal{G}'$, all of the compatible haplotypes are in B, stop; the problem is infeasible.
4. For all $g \in \mathcal{G}'$ such that there exists only one haplotype h not in B compatible with g, put h in G and remove g from $\mathcal{G}_H$.
5. Let $\mathcal{G}'$ be the subset of genotypes in $\mathcal{G}_D$ that can be obtained as $h \oplus h'$, with $h, h' \in G$. If, for any $g \in \mathcal{G}'$, all of the possible resolutions use a haplotype that is in G, stop; the problem is infeasible.
6. For all $g \in \mathcal{G}'$ such that there exists exactly one resolution $h \oplus h'$ in which both h and h' are not in G, put h and h' in B and remove g from $\mathcal{G}_D$.

At the end of the cascade, either the problem has been declared infeasible or there are two sets B and G of bad and good haplotypes that must be fixed and all of the unsolved genotypes have a degree of freedom.

As a corollary of Theorem 4, we have that the problem is feasible as long as the haplotypes in B fall in suitable parity classes (e.g., the bad fixed haplotypes skip a certain class modulo 3, which we can then declare good and use to resolve the remaining healthy genotypes).

4 Conclusions

Given a population affected by a genetic disease, it is expected that all haplotypes can be partitioned into working and faulty haplotypes (what we called "good" and "bad" in this paper). This is a biologically necessary condition for a solution to be correct and, as such, it has been posed as a combinatorial condition for the haplotyping problem. We have shown that it is NP-complete to satisfy this condition. Most importantly, however, we have shown that the partitioning condition alone is too weak to capture the essence of the genetic disease under study (it is hardly believable that a genetic illness is due to the sum of alleles being a multiple of 3...). A contribution of our paper is that it proves that more constraints must be imposed in order for the solution to explain the genetic disease. For instance, it is usually required that a set of haplotypes derived from a set of genotypes either fits a perfect phylogeny or is of the smallest possible size, or both. Hence, we could require either of these conditions as an extra condition for HDA, thereby perhaps forbidding a mathematically correct (but biologically meaningless) solution such as that of Theorem 4. Another possible approach is the following: Assume that an HDA instance is feasible. We have seen that this may be due to the large degree of freedom given by heterozygous sites, which we used to give the "coloring" solution of Theorem 4. However,

the critical SNPs that are really associated to the disease may be just a small subset of all SNPs. A natural question then is what is the minimum set of SNPs for which the solution is still feasible? We are then led to the following optimization problem: *Remove the largest number of SNPs so that the instance left is still feasible*. We leave these variants of the HDA problem as directions for future research.

ACKNOWLEDGMENTS

This research was done while Giuseppe Lancia was visiting Carnegie Mellon University. This research was supported by US National Science Foundation (NSF) ITR Grant CCR-0122581 (the ALADDIN project), by NSF Grant CCF-043075, and by MIUR Grant P.R.I.N. "Algoritmi di ottimizzazione per l'analisi comparativa di dati genomici di grandi dimensioni."

REFERENCES

[1] V. Bafna, D. Gusfield, G. Lancia, and S. Yooseph, "Haplotyping as Perfect Phylogeny: A Direct Approach," *J. Computational Biology*, vol. 10, nos. 3-4, pp. 323-340, 2003.
[2] D.G. Brown and J.M. Harrower, "A New Integer Programming Formulation for the Pure Parsimony Problem in Haplotype Analysis," *Proc. Fourth Int'l Workshop Algorithms in Bioinformatics*, pp. 254-265, 2004.
[3] A. Clark, "Inference of Haplotypes from PCR-Amplified Samples of Diploid Populations," *Molecular Biology Evolution*, vol. 7, pp. 111-122, 1990.
[4] S.A. Cook, "The Complexity of Theorem-Proving Procedures," *Proc. Third Ann. ACM Symp. Theory of Computing*, 1971.
[5] Z. Ding, V. Filkov, and D. Gusfield, "A Linear-Time Algorithm for the Perfect Phylogeny Haplotyping Problem," *Proc. Ninth Ann. Int'l Conf. Research in Computational Molecular Biology*, 2005.
[6] E. Eskin, E. Halperin, and R. Karp, "Efficient Reconstruction of Haplotype Structure via Perfect Phylogeny," *J. Bioinformatics and Computational Biology*, vol. 1, no. 1, pp. 1-20, 2003.
[7] H. Greenberg, W. Hart, and G. Lancia, "Opportunities for Combinatorial Optimization in Computational Biology," *INFORMS J. Computing*, vol. 16, no. 3, pp. 1-22, 2004.
[8] D. Gusfield, "Haplotype Inference by Pure Parsimony," *Proc. 14th Ann. Symp. Combinatorial Pattern Matching*, pp. 144-155, 2003.
[9] D. Gusfield and S.H. Orzack, "Haplotype Inference," *Handbook of Computational Molecular Biology*, pp. 1-28, Chapman and Hall/CRC Press, 2005.
[10] L. Helmuth, "Genome Research: Map of the Human Genome 3.0," *Science*, vol. 293, no. 5530, pp. 583-585, 2001.
[11] Y.T. Huang, K.M. Chao, and T. Chen, "An Approximation Algorithm for Haplotype Inference by Maximum Parsimony," *Proc. 20th Ann. ACM Symp. Applied Computing*, pp. 146-150, 2005.
[12] G. Lancia, C. Pinotti, and R. Rizzi, "Haplotyping Populations by Pure Parsimony: Complexity, Exact and Approximation Algorithms," *INFORMS J. Computing*, vol. 16, no. 4, pp. 17-29, 2004.
[13] E. Marshall, "Drug Firms to Create Public Database of Genetic Mutations," *Science*, vol. 284, no. 5413, pp. 406-407, 1999.
[14] L. Wang and Y. Xu, "Haplotype Inference by Maximum Parsimony," *Bioinformatics*, vol. 19, no. 14, pp. 1773-1780, 2003.

Giuseppe Lancia received the MS and PhD degrees in algorithms, combinatorics, and optimization from Carnegie Mellon University. He is an associate professor of operations research at the University of Udine. From 1999 to 2001, he was a visiting scientist at Sandia National Laboratories and then at Celera Genomics, where he worked on the completion of the first draft sequencing of the human genome. His research focuses on combinatorial optimization problems arising in computational molecular biology. He has been an associate editor for special issues of *INFORMS Journal on Computing* and *BMC Bioinformatics* and has served as a program committee member for several international computational biology conferences.

R. Ravi received the PhD degree in computer science from Brown University. He is the Carnegie Bosch professor of operations research and computer science at Carnegie Mellon University. His research interests include combinatorial optimization, approximation algorithms, and computational molecular biology. He serves on the editorial board of *Management Science*, *ACM Transactions on Algorithms*, and *Operations Research* and has been a program committee member for several computer science and operations research international conferences. He has received the US National Science Foundation CAREER Award and the Sigma Xi Outstanding Graduate Research Prize.

Romeo Rizzi received the Laurea degree in electronic engineering from the Politecnico di Milano in 1991 and the PhD degree in computational mathematics and informatics from the University of Padova, Italy, in 1997. Afterward, he held postdoctoral and other temporary positions at research centers like CWI (Amsterdam), BRICS (Aarhus, Denmark), and IRST (Trento, Italy). In 2001, he became an assistant professor at the University of Trento. Since 2005, he has been an associate professor at the University of Udine. He has a background in operations research and his main interests are in combinatorial optimization and algorithms. He is an area editor of *4OR* and acts as a reviewer for the American Mathematical Society. He has published more than 50 articles in a broad range of scientific journals in the areas of discrete mathematics, combinatorics, and algorithms. Since 2004, he has been a trainer of the Italian team for the iOi. His biography was included in the 2007 edition of *Who's Who in the World*.