Available online at www.sciencedirect.com

SCIENCE DIRECT®

Operations Research Letters 34 (2006) 289–295

Operations Research Letters

www.elsevier.com/locate/orl

A polynomial case of the parsimony haplotyping problem

Giuseppe Lancia[a,*], Romeo Rizzi[b]

[a]Dipartimento di Matematica e Informatica, Università di Udine, Via delle Scienze 206, 33100 Udine, Italy
[b]Dipartimento di Informatica e Telecomunicazioni, Università di Trento, Italy

Received 3 February 2005; accepted 7 May 2005
Available online 27 June 2005

Abstract

The *parsimony haplotyping* problem was shown to be NP-hard when each *genotype* had $k \leqslant 3$ *ambiguous positions*, while the case for $k \leqslant 2$ was open. In this paper, we show that the case for $k \leqslant 2$ is polynomial, and we give approximation and FPT algorithms for the general case of $k \geqslant 0$ ambiguous positions.
© 2005 Elsevier B.V. All rights reserved.

Keywords: Totally unimodular matrices; Approximation algorithms; Fixed parameter tractability; Parsimony haplotyping; Computational biology

1. Introduction

In this paper we focus on a combinatorial problem defined on ternary vectors and derived from a molecular genetics procedure known as *haplotyping*. We will briefly describe the biological motivations behind the problem definition in Section 1.1. The starting point of the investigations reported here is to settle the complexity for a special class of instances which had resisted prior complexity classification, but whose relevance to real-life applications appears minor. However, the deeper mathematical insight gained into these borderline instances will allow us to derive simple and practical combinatorial approximation algorithms for a wider and more significant class of instances. Moreover, FPT algorithms for this more relevant class of instances will also follow as a byproduct of our core structural results. Therefore, we prefer to focus on the mathematical aspects of the result more than on the implications that our algorithm can have for the practical solution of the biology problem.

The problem data consist in a set $\mathcal{G}$ of n *genotypes* $g_1, \ldots, g_n$, corresponding to n individuals in a population. Each genotype g is a vector with entries in $\{0, 1, 2\}$. Each position where a 2 appears is called an *ambiguous* position. For each genotype g we must determine a pair of *haplotypes* h_{P} and h_{M} (h_{P} stands for the *paternal* haplotype and h_{M} stands for the *maternal* haplotype), which are binary vectors such that $g = h_{\mathrm{P}} \oplus h_{\mathrm{M}}$ (where the definition of the $\oplus$ operation will be given later). A set $\mathcal{H}$ of haplotypes is said to

* Corresponding author.
E-mail address: lancia@dimi.uniud.it (G. Lancia).

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

Fig. 1. The haplotypes of 3 individuals, with 4 SNPs.

explain $\mathcal{G}$ if for each $g \in \mathcal{G}$ there is at least one pair of haplotypes $h', h'' \in \mathcal{H}$ such that $g = h' \oplus h''$.

Given a set $\mathcal{G}$ of genotypes, the *haplotyping* problem consists in finding a set $\mathcal{H}$ of haplotypes that explains $\mathcal{G}$. In this paper we will pursue the *parsimony* objective function, i.e., we will be interested in finding a set $\mathcal{H}$ of smallest possible cardinality.

1.1. Polymorphisms and biological motivations

A *single nucleotide polymorphism* (SNP) is a site of the human genome (i.e., the position of a specific nucleotide) showing a statistically significant variability within a population. Besides very rare exceptions, at each SNP site we observe only two (out of the possible four, i.e., A, T, C and G) nucleotides, called the SNP *alleles*. The recent completion of the sequencing phase of the Human Genome Project [15,12] has shown that the genome of any two individuals is the same in about 99% of the positions, and that most polymorphisms (i.e., differences at genomic level) are in fact SNPs [3].

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 haplotypes are `CCCT` and `GAGT`.

Haplotyping an individual consists of determining his two haplotypes, for a given chromosome. With the larger availability in SNP genomic data, recent years have witnessed the emergence of a set of new computational problems related to 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 reasonable experiment which can determine the (less informative and often ambiguous) genotypes, from which the haplotypes must then be retrieved computationally.

A *genotype* provides, for an individual, information about the multiplicity of each SNP allele, i.e., for each SNP site, the genotype specifies whether the individual is heterozygous or homozygous (in the latter case, it also specifies the allele).

The ambiguity comes from heterozygous sites, since, 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 g requires determining two haplotypes such that, if they are assumed to be the two chromosome copies, then, computing the multiplicity of each SNP allele, we obtain exactly the genotype g. Given a set $\mathcal{G}$ of genotypes, the *population haplotyping problem* requires to determine a set $\mathcal{H}$ of haplotypes such that each genotype $g \in \mathcal{G}$ is explained by two haplotypes $h', h'' \in \mathcal{H}$. For its importance (as we said, haplotyping from genotype data is nowadays the only viable way) the population haplotyping problem has been and is being extensively studied, under many objective functions, among which are: *Perfect phylogeny* [1,5], *Clark's rule* [6,7] and *Parsimony* [8,13,2].

Each model and objective function has specific biological motivations, which are discussed in the cited references. In this paper, we focus on the parsimony haplotyping problem. Under the parsimony objective, it is required that $\mathcal{H}$ has the minimum possible cardinality. The objective is based on the principle that, under many explanations of an observed phenomenon, one should choose the one that requires the fewest assumptions. This problem has been introduced by Gusfield [8], who adopted an integer-programming formulation for its practical solution. The problem is NP-hard, as first shown by Hubbel [11].

Among the several objective functions for haplotyping, pure parsimony is the most recent model, whose importance is now being recognized as crucial in the solution of more complex haplotyping problems. In fact, when observed at the largest possible scale (e.g., considering several thousands SNPs at once), haplotypes are not inherited as units, but there is a certain level of recombination (i.e., new haplotypes are created by merging pieces of other haplotypes). A *block* is a region where no recombination has occurred in any of the haplotypes. Biological reasons suggest that the number of different haplotypes observed within a block should be minimum [8].

1.2. Notation and results

Given a set of n SNPs, fix arbitrarily a binary encoding of the two alleles for each SNP (i.e., call one of the two alleles '0' and the other '1'). Once the encoding has been fixed, each haplotype corresponds to (with a slight abuse of terminology, hereafter, we will say that each haplotype *is*) a binary vector of length n.

For a haplotype h, we denote by $h[i]$ the value of its ith component, with $i=1, \dots, n$. Given two haplotypes h' and h'', their sum is defined as a vector $g := h' \oplus h''$. The vector g has length n, and its components can take only values in $\{0, 1, 2\}$, according to the following rule:

- if $h'[i] = h''[i] = 0$ then $g[i] = 0$;
- if $h'[i] = h''[i] = 1$ then $g[i] = 1$;
- if $h'[i] \neq h''[i]$ then $g[i] = 2$.

We will call a vector g with entries in $\{0, 1, 2\}$ a *genotype*. Each position i such that $g[i]=2$ is called an *ambiguous position* (or *ambiguous site*). A resolution of a genotype g is given by a pair of haplotypes h' and h'' such that $g = h' \oplus h''$. Such haplotypes are said to resolve g, and are called g-mates. A haplotype h is said to be *compatible* with a genotype g if h admits a g-mate. A genotype is ambiguous if it has more than one possible resolution, i.e., if it has at least two ambiguous positions. Biologically, genotype entries of value 0 or 1 correspond to homozygous SNP sites, while entries of value 2 correspond to heterozygous sites. In Fig. 2 we illustrate a case of three individuals, showing their haplotypes and genotypes.

The *parsimony haplotyping problem* (PHP) we consider is the following: find a set $\mathscr{H}$ of haplotypes of *smallest* cardinality that explains a given set $\mathscr{G}$ of genotypes.

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.

It had been previously shown that the above problem is NP-hard [11] and, in fact, APX-hard, even when each input genotype is restricted to possess at most three ambiguous sites [13]. The complexity of the case for two ambiguous sites, however, was unknown. In Section 2, we prove that the problem when each genotype has at most two ambiguous sites is in fact polynomial, by using an argument from Linear Programming. In Section 3, we give an alternative, combinatorial proof by means of a reduction to a classical combinatorial problem which also leads to a practical and effective algorithm to solve this polynomial special case of PHP. In Section 3.2, we discover that this new combinatorial approach scales up well to yield 2^{k-1}-approximation algorithms for the version of the problem in which there are at most k ambiguous sites. We finally point out that, as a consequence of our results, we are able to conclude that this version of the problem is in FPT for any fixed k.

2. A linear program naturally integer

In this section, we give a first proof that PHP can be solved in polynomial time when each genotype has at most two ambiguous positions. The proof consists in showing the total unimodularity of the matrix involved in a known integer linear programming formulation for the problem. In the next section, we will give an alternative, combinatorial proof, leading to an effective combinatorial algorithm.

We start by recalling an integer-programming formulation of PHP [8,13]. This formulation has an exponential number of variables and constraints in the general case but has a polynomial size when there are at most two ambiguous sites per genotype (note that polynomial-size formulations for the general case are also possible; see, e.g., [13,2]). Denote by $\hat{\mathscr{H}}_{\mathscr{G}}$ the set of haplotypes that are compatible with some genotype in $\mathscr{G}$. Let $\mathscr{G}^{\mathrm{N}}$ be the set of non-ambiguous genotypes (i.e., those with at most one ambiguous position) and let $\hat{\mathscr{H}}_{\mathscr{G}}^{\mathrm{N}} \subseteq \hat{\mathscr{H}}_{\mathscr{G}}$ be the set of haplotypes compatible with genotypes in $\mathscr{G}^{\mathrm{N}}$. Clearly, $\hat{\mathscr{H}}_{\mathscr{G}}^{\mathrm{N}}$ must be contained in every feasible solution of the problem. Associate a binary variable x_h with every $h \in \hat{\mathscr{H}}_{\mathscr{G}}$ (where $x_h = 1$ means that h is taken in the solution, whereas $x_h = 0$ means that h is not taken). After fixing a total ordering on $\hat{\mathscr{H}}_{\mathscr{G}}$, denote by $\mathscr{P}$ the set of those pairs (h_1, h_2) with $h_1, h_2 \in \hat{\mathscr{H}}_{\mathscr{G}}$, $h_1 < h_2$. For every $g \in \mathscr{G}$, let $\mathscr{P}_g := \{(h_1, h_2) \in \mathscr{P} | h_1 \oplus h_2 = g\}$. Note that $\mathscr{P}_g \cap \mathscr{P}_{g'} = \emptyset$ whenever $g \neq g'$. For every $g \in \mathscr{G} \backslash \mathscr{G}^{\mathrm{N}}$, introduce a binary variable y_{h_1,h_2} for every $(h_1, h_2) \in \mathscr{P}_g$. The meaning of y_{h_1,h_2} is to select a pair (h_1, h_2) used to resolve some ambiguous genotype. The following is a valid ILP formulation of the PHP problem [8,13]:

$$\min \sum_{h \in \hat{\mathscr{H}}_{\mathscr{G}}} x_h \tag{1}$$

s. t.

$$\sum_{(h_1,h_2) \in \mathscr{P}_g} y_{h_1,h_2} \geqslant 1 \quad \forall g \in \mathscr{G} \backslash \mathscr{G}^{\mathrm{N}}, \tag{2}$$

$$y_{h_1,h_2} \leqslant x_{h_1} \quad \forall (h_1, h_2) \in \bigcup_{g \in \mathscr{G} \backslash \mathscr{G}^{\mathrm{N}}} \mathscr{P}_g, \tag{3}$$

$$y_{h_1,h_2} \leqslant x_{h_2} \quad \forall (h_1, h_2) \in \bigcup_{g \in \mathscr{G} \backslash \mathscr{G}^{\mathrm{N}}} \mathscr{P}_g, \tag{4}$$

$$x_h = 1 \quad \forall h \in \hat{\mathscr{H}}_{\mathscr{G}}^{\mathrm{N}}, \tag{5}$$

$$x \in \{0, 1\}^{\hat{\mathscr{H}}_{\mathscr{G}}}, \quad y \in \{0, 1\}^{\mathscr{P}}. \tag{6}$$

Constraints (2) impose that each ambiguous genotype is resolved by at least one pair of haplotypes. Non-ambiguous genotypes are resolved by forced haplotypes, which are all included in the solution because of (5). Constraints (3) and (4) ensure that a pair (h_1, h_2) can be used to resolve a genotype only if both h_1 and h_2 are included in the solution.

We now show that the constraint matrix of the model is totally unimodular when each genotype has at most two ambiguous sites. This implies that any basic optimal solution to the linear programming relaxation of (1)–(6) is in fact an optimal integer solution. Furthermore, the LP relaxation can be solved in polynomial time.

A classical result on totally unimodular matrices is the following sufficient condition [9]:

Theorem 1. *A $\{0, 1, -1\}$-matrix is totally unimodular if both the following conditions hold:*

- *Each row has at most two nonzero entries.*
- *The set of columns can be partitioned into two subsets C_1 and C_2 such that*

  1. *if a row contains two nonzero entries with the same sign, then one of these entries belongs to a column in C_1, while the other belongs to a column in C_2; and*
  2. *if a row contains two nonzero entries of opposite sign, then the two columns containing these two elements are in the same subset.*

Using Theorem 1, we can now prove the following result:

Theorem 2. *The constraint matrix of* (2)–(5) *is totally unimodular when each genotype has at most two ambiguous sites.*

Proof. Call a haplotype h *even* if $\sum_{i=1}^{n} h[i]$ is even, and *odd* otherwise. Call a variable x_h even if h is even and odd otherwise. The key observation is that, for every $g \in \mathscr{G} \backslash \mathscr{G}^{\mathrm{N}}$, and every pair $(h_1, h_2) \in \mathscr{P}_g$, h_1 and h_2 are either both odd or both even. This is because every ambiguous genotype has precisely two ambiguous sites. Call a variable y_{h_1,h_2} even if h_1 and h_2 are both even and odd otherwise. Note furthermore that when $g \in \mathscr{G} \backslash \mathscr{G}^{\mathrm{N}}$, then $|\mathscr{P}_g| = 2$, and of the two resolutions that g admits, one is the sum of two odd haplotypes, and the other is the sum of two even haplotypes. Hence, constraints (2) involve an odd and an even variable, while constraints (3) and (4) involve variables which are both odd or both even. With C_1 the set of odd variables and C_2 the set of even variables, the conditions of Theorem 1 are satisfied, and the constraint matrix is totally unimodular.

Corollary 3. *When each genotype has at most two ambiguous positions, the PHP problem can be solved in polynomial time.*

This is because any basic optimal solution of the Linear Programming relaxation of (1)–(6), which can be obtained in polynomial time, is in fact an optimal solution of PHP.

3. Combinatorial algorithms

As in the previous section, let $\mathscr{G}$ be a set of input genotypes $\mathscr{G}$, and denote by $\hat{\mathscr{H}}_{\mathscr{G}}$ the set of haplotypes which are compatible with some genotype in $\mathscr{G}$.

Clearly, when $\mathscr{H}$ is a minimal set of haplotypes which explains $\mathscr{G}$, then $\mathscr{H} \subseteq \hat{\mathscr{H}}_{\mathscr{G}}$. Hence, our goal is to find a minimum subset of $\hat{\mathscr{H}}_{\mathscr{G}}$ which explains $\mathscr{G}$. When each genotype in $\mathscr{G}$ has at most k ambiguous positions, then the size of $\hat{\mathscr{H}}_{\mathscr{G}}$ is at most $2^k|\mathscr{G}|$. Given a genotype g with t ambiguous positions ($t \leqslant g$), and a $\{0, 1\}$ vector s of length precisely t, we denote by $g(s)$ the haplotype obtained from g by substituting, for every $i \in \{1, \dots, t\}$, the ith "2" symbol in g with the ith component of s.

3.1. At most two ambiguous positions: an exact algorithm

Assume, to begin with, that every genotype in $\mathscr{G}$ has precisely two ambiguous positions. We show how to reduce the problem of finding a minimum size subset of $\hat{\mathscr{H}}_{\mathscr{G}}$ which explains $\mathscr{G}$ to the problem of finding a minimum node cover in a bipartite graph (for which effective algorithms are known; see, e.g., [10]). For every genotype $g \in \mathscr{G}$, we consider a graph $G_g = (V_g, E_g)$ on 4 nodes and 4 edges (a square), whose nodes are $V_g = \{g(00), g(01), g(11), g(10)\}$ and whose edges are

$$E_g = \{g(00)g(01), g(01)g(11), g(11)g(10), g(10)g(00)\}. \tag{7}$$

Each node of the square G_g in Fig. 3 is a haplotype. Note that every node cover of G_g contains at least two opposite nodes of the square, whence a pair of g-mates (i.e., a pair of haplotypes whose sum yields g). Consider now the graph $G = (V, E)$ with node set $V = \bigcup_{g\in\mathscr{G}} V_g = \hat{\mathscr{H}}_{\mathscr{G}}$ and edge set $E = \bigcup_{g\in\mathscr{G}} E_g$.

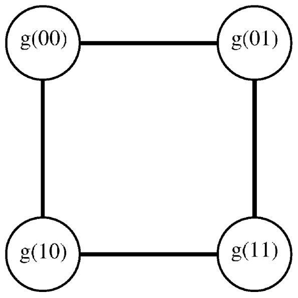

Fig. 3. The square G_g.

Lemma 4. *Let $\mathscr{H}$ be any subset of $\hat{\mathscr{H}}_{\mathscr{G}}$. Then $\mathscr{H}$ explains $\mathscr{G}$ if and only if $\mathscr{H}$ is a node cover of G.*

Proof. Assume that $\mathscr{H}$ is a node cover of G. Then, for any genotype g, $\mathscr{H}$ is a node cover for the square G_g. This implies that $\mathscr{H}$ explains each genotype $g \in \mathscr{G}$.

For the converse, assume that $\mathscr{H}$ explains g. This means that either $g(0\,0), g(1\,1) \in \mathscr{H}$ or $g(1\,0), g(0\,1) \in \mathscr{H}$ holds; hence, $\mathscr{H}$ covers E_g. Since $\mathscr{H}$ explains each genotype $g \in \mathscr{G}$, it follows that $\mathscr{H}$ covers $E = \bigcup_{g\in\mathscr{G}} E_g$. □

We have shown that, when each genotype has precisely two ambiguous sites, then our problem amounts to finding a minimum node cover in the graph G. The following observation closes the issue about the complexity of PHP in this special case.

Observation 5. *The graph G is bipartite.*

Proof. The graph G is actually a subgraph of the hypercube since two haplotypes are adjacent only if at Hamming distance 1. □

Since the minimum node cover problem is polynomial for bipartite graphs, we obtain that the special case of PHP in which each genotype has two ambiguous positions is polynomial as well.

It is now easy to also accommodate genotypes containing at most one ambiguous position (i.e., the non-ambiguous genotypes). If g has no ambiguous position, then $h = g$ must be taken in each solution. This corresponds to searching for a node cover X in $G\backslash g$ and then returning $X \cup \{h\}$. If g contains precisely one ambiguous position, then $h_0 = g(0)$ and $h_1 = g(1)$

must both be taken. This corresponds to searching for a node cover X in $G \backslash \{h_0, h_1\}$ and then returning $X \cup \{h_0, h_1\}$. More generally, the following algorithm solves our problem.

1. Input $\mathscr{G}$;
2. Let $\mathscr{G}^{\mathrm{N}}$ be the set of non-ambiguous genotypes in $\mathscr{G}$;
3. Let $\hat{\mathscr{H}}^{\mathrm{N}}_{\mathscr{G}}$ be the set of haplotypes compatible with genotypes in $\mathscr{G}^{\mathrm{N}}$;
4. Find a minimum node cover X in the bipartite graph $G[\hat{\mathscr{H}}_{\mathscr{G}} \backslash \hat{\mathscr{H}}^{\mathrm{N}}_{\mathscr{G}}]$; and
5. Return $X \cup \hat{\mathscr{H}}^{\mathrm{N}}_{\mathscr{G}}$.

Remark 6. Note that $|V(G)| = \mathrm{O}(|\mathscr{G}|)$ and $|E(G)| = \mathrm{O}(|\mathscr{G}|)$. Where n is the number of SNPs (i.e., the genotype length), then the graph G can be built in $\mathrm{O}(n|\mathscr{G}|)$ time. A minimum node cover of G is computed in $\mathrm{O}(\sqrt{|V(G)|}|E(G)|)$ time by the algorithm in [10]. The total running time of our algorithm is therefore $\mathrm{O}(n|\mathscr{G}| + |\mathscr{G}|^{3/2})$.

3.2. *At most k ambiguous sites: an approximation algorithm*

Consider an ambiguous genotype g and let $\hat{\mathscr{H}}_{\{g\}}$ be the set of haplotypes which are compatible with g. Since any haplotype can have at most one g-mate, the haplotypes in $\hat{\mathscr{H}}_{\{g\}}$ are paired up by the "being g-mate" relation. We denote by $P(g)$ the family of those subsets of $\hat{\mathscr{H}}_{\{g\}}$ which take precisely one haplotype from each pair of g-mates. Hence, when g contains k ambiguous sites, then $|\hat{\mathscr{H}}_{\{g\}}| = 2^k$ and $P(g)$ contains $2^{2^{k-1}}$ sets of size 2^{k-1}. As a special case, when g has no ambiguous site, we assume that $P(g) = \{\{g\}\}$. A *transversal*, or *hitting set*, of $P(g)$ is a minimal set of haplotypes that intersects all elements of $P(g)$.

Observation 7. *Let g be an ambiguous genotype. Then the transversals of $P(g)$ are precisely the pairs of g-mates.*

Proof. Clearly, all pairs of g-mates are transversals of $P(g)$. Conversely, let X be any transversal of $P(g)$. We will show that X contains a pair of g-mates. Indeed, if on the contrary X misses at least one haplotype from each pair of g-mates, then the set of missed haplotypes contains a member of $P(g)$, contradicting the fact that X is a transversal of $P(g)$.

With some background in transversal theory, we could have come to the same conclusion by duality after considering that the sets in $P(g)$ are the transversals of the clutter made by all pairs of g-mates. □

We consider the hypergraph $G = (V, E)$ with node set $V = \hat{\mathscr{H}}_{\mathscr{G}}$ and edge set $E = \bigcup_{g \in \mathscr{G}} P(g)$. We then have the following lemma.

Lemma 8. *Let $\mathscr{H}$ be any subset of $\hat{\mathscr{H}}_{\mathscr{G}}$. Then $\mathscr{H}$ explains $\mathscr{G}$ if and only if $\mathscr{H}$ is a node cover of G.*

Proof. Assume that $\mathscr{H}$ is a node cover of G. Consider any genotype g in $\mathscr{G}$. We will show that $\mathscr{H}$ explains g. Indeed, since $\mathscr{H}$ covers every hyperedge in $P(g)$, then $\mathscr{H}$ must contain a pair of g-mates, or g itself in case g is not ambiguous.

For the converse, assume that $\mathscr{H}$ explains g. If g is not ambiguous, then $\mathscr{H}$ contains g and hence covers $P(g) = \{g\}$. Otherwise, if g is ambiguous, then $\mathscr{H}$ contains a pair of g-mates and hence covers $P(g)$. Since $\mathscr{H}$ explains each genotype $g \in \mathscr{G}$, it follows that $\mathscr{H}$ covers $E = \bigcup_{g \in \mathscr{G}} P(g)$. □

Note that, when each $g \in \mathscr{G}$ has at most k ambiguous positions, then every edge in G has size at most 2^{k-1}. It is well known [14] that for such instances of node cover on hypergraphs there exist extremely simple and effective combinatorial primal-dual 2^{k-1}-approximation algorithms. These algorithms take linear time in the size of the input hypergraph, i.e., in the sum of the cardinalities of the hyperedges. Our hypergraph has $|\hat{\mathscr{H}}_{\mathscr{G}}| \leqslant 2^k|\mathscr{G}|$ nodes and at most $\sum_{g \in \mathscr{G}} |P(g)| \leqslant 2^{2^{k-1}}|\mathscr{G}|$ hyperedges. Each hyperedge has at most 2^{k-1} nodes. We can hence produce a 2^{k-1}-approximate solution in $2^{2^{k-1}+k-1}\mathrm{O}(|\mathscr{G}|)$ time and space. It is also well known [4] that a whole class of FPT algorithms exists to solve the node cover problem to optimality on hypergraphs with hyperedges of bounded size. While these FPT algorithms do not appear to provide a viable approach in this setting due to the doubly exponential dependence on k that ultimately results in their running times, still these remarks show that the problem is in FPT whenever the number of ambiguous sites is bounded by a constant and may suggest a possible means of dealing with the problem in practice.

References

[1] V. Bafna, D. Gusfield, G. Lancia, S. Yooseph, Haplotyping as perfect phylogeny: a direct approach, J. Comput. Biol. 10 (3–4) (2003) 323–340.

[2] D.G. Brown, J.M. Harrower, A new integer programming formulation for the pure parsimony problem in haplotype analysis, in: Proceedings of Annual Workshop on Algorithms in Bioinformatics (WABI), Lecture Notes in Computer Science, Springer, 2004.

[3] A. Chakravarti, It's raining SNP, hallelujah?, Nat. Genet. 19 (1998) 216–217.

[4] R.G. Downey, M.R. Fellows, Parameterized Complexity, Monographs in Computer Science, Springer, New York, 1999.

[5] E. Eskin, E. Halperin, R. Karp, Efficient reconstruction of haplotype structure via perfect phylogeny, J. Bioinformatics Comput. Biol. 1 (1) (2003) 1–20.

[6] D. Gusfield, A practical algorithm for optimal inference of haplotypes from diploid populations, in: R. Altman, T.L. Bailey, P. Bourne, M. Gribskov, T. Lengauer, I.N. Shindyalov, L.F. Ten Eyck, H. Weissig (Eds.), Proceedings of the Annual International Conference on Intelligent Systems for Molecular Biology (ISMB), AAAI Press, Menlo Park, CA, 2000, pp. 183–189.

[7] D. Gusfield, Inference of haplotypes from samples of diploid populations: complexity and algorithms, J. Comput. Biol. 8 (3) (2001) 305–324.

[8] D. Gusfield, Haplotype inference by pure parsimony, in: Proceedings of the Annual Symposium on Combinatorial Pattern Matching (CPM), Lecture Notes in Computer Science, vol. 2676, Springer, Berlin, 2003, pp. 144–155.

[9] I. Heller, C.B. Tompkins, An extension of a theorem of Dantzig's, in: H.W. Kuhn, A.W. Tucker (Eds.), Linear Inequalities and Related Systems, Princeton University Press, 1956, pp. 246–254.

[10] J.E. Hopcroft, R.M. Karp, An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs, SIAM J. Comput. 2 (1973) 225–231.

[11] E. Hubbel, Unpublished manuscript, 2002.

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

[13] G. Lancia, C. Pinotti, R. Rizzi, Haplotyping populations by pure parsimony: complexity, exact and approximation algorithms, INFORMS J. Comput. 16 (4) (2004) 17–29.

[14] V. Vazirani, Approximation Algorithms, Springer, Berlin, 2001.

[15] J.C. Venter, et al., The sequence of the human genome, Science 291 (2001) 1304–1351.