# HAPLOTYPING POPULATIONS: COMPLEXITY AND APPROXIMATIONS

Giuseppe Lancia, Cristina Pinotti and Romeo Rizzi

October 2002

Technical Report # DIT-02-0080

# Haplotyping Populations: Complexity and Approximations

Giuseppe Lancia[1] Cristina M. Pinotti[2] Romeo Rizzi[2]

*1. Dipartimento di Ingegneria dell'Informazione, Università di Padova*
lancia@dei.unipd.it
*2. Dipartimento di Informatica e Telecomunicazioni, Università di Trento*
{romeo,pinotti}@science.unitn.it

**Abstract**

We study the computational complexity of the following haplotyping problem. Given a set of genotypes $\mathcal{G}$, find a minimum cardinality set of haplotypes which explains $\mathcal{G}$. Here, a genotype $g$ is an $n$-ary string over the alphabet $\{A, B, -\}$ and an haplotype $h$ is an $n$-ary string over the alphabet $\{A, B\}$. A set of haplotypes $\mathcal{H}$ is said to explain $\mathcal{G}$ if for every $g \in \mathcal{G}$ there are $h_1, h_2 \in \mathcal{H}$ such that $h_1 + h_2 = g$. The position-wise sum $h_1 + h_2$ indicates the genotype which has a $-$ in the positions where $h_1$ and $h_2$ disagree, and the same value as $h_1$ and $h_2$ where they agree. We show the APX-hardness of the problem even in the case the number of $-$ symbols is at most 3 for every $g \in \mathcal{G}$. We give a $\sqrt{|\mathcal{G}|}$-approximation algorithm for the general case, and a $2^{k-1}$-approximation algorithm when the number of $-$ symbols is at most $k$ for every $g \in \mathcal{G}$.

**Keywords**: computational biology, SNPs, haplotyping, approximation algorithms.

## 1 Problem Description

With the completion of the sequencing of the human genome has come the confirmation that all humans are almost identical at DNA level (99% and greater identity). Hence, small regions of differences must be responsible for the observed diversities at phenotype level. The smallest possible variation is at a single nucleotide, and is called *Single Nucleotide Polymorphism*, or SNP (pronounced "snip"). Broadly speaking, a polymorphism is a trait, common to everybody, whose value can be different but drawn in a limited range of possibilities, called *alleles*. A SNP is a specific nucleotide, placed in the middle of a DNA region which is otherwise identical for all of us, whose value varies within a population. In particular, each SNP shows a variability of only two alleles. These alleles can be different for different SNPs.

Recent studies have shown that SNPs are the predominant form of human variation [3] occurring, on average, every thousand bases. Their importance cannot be overestimated for therapeutic, diagnostic and forensic applications.

Since DNA of *diploid* organisms is organized in pairs of chromosomes, for each SNP one can either be *homozygous* (same allele on both chromosomes) or *heterozygous* (different alleles). The values of a set of SNPs on a particular chromosome copy define a *haplotype*. *Haplotyping* an individual consists in determining a pair of haplotypes, one for each copy of a given chromosome. The pair provides full information of the SNP fingerprint for that individual at the specific chromosome.

With the larger availability in SNP genomic data, the recent years have also seen the birth of a set of new combinatorial and optimization problems related to SNPs [8, 12, 13, 6, 7]. In particular, since it is infeasible (or, at least, astronomically expensive) to perform the complete sequencing of an individual's genome as a routine experiment, most combinatorial problems are related to haplotyping individuals without sequencing their genomes. Furthermore, even in the case of a fully sequenced genome, unavoidable errors in the data lead to the definition of mathematical haplotyping problems such as the *Single Individual Haplotyping Problem*, studied by Lancia et al. [8], Lippert et al. [9] and Rizzi et al. [12].

In this paper we consider the problem of haplotyping a *population* (i.e., a set of individuals) from ambiguous *genotype* data. Genotype data provides, for an individual, only information about the multiplicity of each allele at each SNP: I.e., we only know, for each SNP, if the person is homo- or hetero- zygous. The ambiguity comes from heterozygous sites, since we have to decide how to distribute the two allele values on the two chromosome copies, to retrieve the haplotypes. 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 [4, 5, 6, 7].

*Resolving* (or *explaining*) a genotype $g$ requires to determine two haplotypes such that, if they are assumed to be the ones on the two chromosome copies, then, computing the multiplicity of each SNP allele we obtain exactly the genotype $g$. The *Population Haplotyping Problem* (PHP) is then the following: given a set $\mathcal{G}$ of genotypes, determine a set $\mathcal{H}$ of haplotypes such that each genotype $g \in \mathcal{G}$ is explained by two haplotypes $h', h'' \in \mathcal{H}$.

Note that without specifying an objective to optimize, or a set of constraints on $\mathcal{H}$, the problem is trivial. A natural objective for (PHP) is to require that $\mathcal{H}$ has the minimum possible cardinality. We are lead to this objective by a famous principle, the *Maximum Parsimony*, that has been adopted on innumerable occasions in computational biology. The principle, known also as Occam's razor, states that under many explanations of an observed phenomenon, we should choose the one that requires the fewest assumptions. Here, we are trying to determine what is the minimum number of different elements (haplotypes) that, recombined in pairs among them during time, have given rise to a set of observed diversities (genotypes). To our surprise (and to the best of our knowledge), this optimization problem has never been studied before in the form stated.

The most closely related problem has been studied by Gusfield in [5, 6]. In this version of the problem, the solution $\mathcal{H}$ is required to be obtained via successive applications of a rule known as *Clark's rule*. In 1990, the biologist A. Clark suggested a greedy rule [4] to resolve genotypes. This rule starts from a minimal "bootstrap" set of haplotypes and uses them to explain as many genotypes as possible, while greedily introducing new haplotypes when needed. Hence, one can heuristically expect the final set $\mathcal{H}$ to have a "small" cardinality, although it is easy to find examples for which this is not true. Gusfield studied the problem of how to apply Clark's rule in an optimal way, showing this problem to be APX-hard and suggesting an Integer Programming formulation for its solution.

Finally, another version of (PHP) has been proposed for which it is required that $\mathcal{H}$ admits a *perfect phylogeny*. The problem was shown to be polynomial [7].

### 1.1 Our results

In this paper we study the following version of (PHP): find a set $\mathcal{H}$ of haplotypes of *smallest* cardinality to explain a given set $\mathcal{G}$ of $m$ genotypes. The binary nature of SNPs leads to a nice combinatorial problem. As described in the sequel, this problem is defined over a matrix with elements in $\{0, 1, -\}$.

We first address the complexity of the problem, showing, via a reduction from NODE-COVER that the problem is APX-hard.

In light of this result, we then try to determine a good approximation algorithm. After showing that it is easy to obtain a $\sqrt{m}$-approximation, we obtain better approximation algorithms, whose ratio depends on the number of "$-$" in the input. In particular, by using a randomized algorithm, we show the following:

**Las Vegas.** Let $k$ be the maximum number of "-" characters in a genotype. We give a randomized algorithm, which, assuming that $OPT \leq m^{\alpha}$, will return a feasible solution $\mathcal{H}_{good}$ with $|\mathcal{H}_{good}| \leq 2^{k+2} m^{\alpha} \log 2m$. The running time of the algorithm is a random variable pretty much concentrated and with expected value bounded by a polynomial in the size of the input instance.

**Monte Carlo.** We give a randomized algorithm, which will return almost surely (i.e. with probability at least $\frac{m-1}{m}$) a feasible solution $\mathcal{H}_{good}$ with $|\mathcal{H}_{good}| \leq 2^{k+2} \log 2m \; OPT$. The running time of the algorithm is bounded by a polynomial in the size of the input instance.

Moreover, by using a linear programming formulation, we could more recently show the following:

**LP formulation.** Let $k$ be the maximum number of "-" characters in a genotype. We give a deterministic $2^{k-1}$-approximation algorithm, which is based on Linear Programming.

We arrived at the third algorithmic solution listed here above only recently, and we are at present trying to derive a combinatorial primal-dual algorithm. Concerning the two randomized algorithms, in deriving the second algorithm from the first (in which we guess the optimum value), we use an observation (Observation 16) that allows us to assume nothing about the optimum. Basically, we do binary search of its value, while showing that this search is still polynomially bounded. We wonder if this way of proceeding is somehow novel.

## 2 Preliminaries: SNPs, haplotypes and genotypes

A *Single Nucleotide Polymorphism*, or SNP, is a position in the genome at which some of us have a certain base while the others have a different one. The two base values are called *alleles*. Due to the binary nature of SNPs, we can encode, for each SNP, the two alleles with the bits 0 and 1. *Diploid* genomes (such as the human genome) are organized into pairs of chromosomes (a paternal and a maternal copy) which have nearly identical content and carry (paternal and maternal) copies of the same genes. For each SNP, an individual is *homozygous* if the SNP has the same allele on both chromosome copies, and otherwise the individual is *heterozygous*. The values of a set of SNPs on a particular chromosome copy define a *haplotype*. Here we give a simplistic example of a chromosome with three SNP sites.

```
Chrom. c, paternal:   ataggtccCtatttccaggcgcCgtatacttcgacgggActata
Chrom. c, maternal:   ataggtccGtatttccaggcgcCgtatacttcgacgggTctata

Haplotype 1 →                 C             C               A
Haplotype 2 →                 G             C               T
```

The individual is heterozygous at SNPs 1 and 3 and homozygous at SNP 2. The haplotypes are CCA and GCT. Under a given encoding of the alleles, these two haplotypes could be represented, e.g., as the binary strings 010 and 111. In this encoding, a "0" at SNP 1 stands for C and a "1" at SNP 1 stands for G.

Haplotype data is particularly sought after in the study of complex diseases (those affected by more than a single gene), since it can give complete information about which set of gene alleles are inherited together. However, polymorphism screens are conducted on large populations where it is not feasible to examine the two copies of each chromosome separately, and *genotype* data rather than haplotype data is usually obtained. A genotype describes the multiplicity of each SNP allele for the chromosome of interest. At each SNP, three possibilities arise: either one is homozygous for the allele 0, or homozygous for the allele 1, or heterozygous (a situation that we shall denote with the symbol -). Hence a genotype is a string over the alphabet $\{0, 1, -\}$, where each position of the letter - is called an *ambiguous* position. We say that a genotype $g$ is *resolved* (or *explained*) by the pair of haplotypes $\{h, q\}$ if, for each SNP $j$, $g[j] = 0$ implies $h[j] = q[j] = 0$, $g[j] = 1$ implies $h[j] = q[j] = 1$, and $g[j] = -$ implies $h[j] \neq q[j]$. A genotype $g$ is said *compatible* with (or *good* for) a haplotype $h$ if $h$ agrees with $g$ at all unambiguous (i.e., non "$-$") positions.

The *Population Haplotyping Problem* (in its *Maximum Parsimony* version) is the following: given a set $\mathcal{G}$ of $m$ genotypes over $n$ SNPs, find a set $\mathcal{H}$ of haplotypes such that each genotype is resolved by one pair of haplotypes in $\mathcal{H}$, and the cardinality of $\mathcal{H}$ is minimum.

## 3 Terminology

A *genotype* is a finite string over the alphabet $\Sigma_3 = \{0, 1, -\}$. An *haplotype* is a finite string over the alphabet $\Sigma_2 = \{0, 1\}$. In the sequel, we assume all given strings to have the same length $n$.

When $h_1$ and $h_2$ are haplotypes, then their *sum* $g = h_1 + h_2$ is a genotype and is defined as follows:

$$g[i] = \begin{cases} h_1[i] & \text{if } h_1[i] = h_2[i] \\ - & \text{otherwise} \end{cases} \quad (i = 1, \dots, n)$$

We are given as input a *population*, that is, a family $\mathcal{G}$ of genotypes. We will consider that $\mathcal{G}$ is encoded by means of an $m \times n$ matrix $M$ with entries in $\Sigma_3 = \{0, 1, -\}$, where $m$ denotes $|\mathcal{G}|$.

We say that a set $\mathcal{H}$ of haplotypes *explains* $\mathcal{G}$ if for every $g \in \mathcal{G}$ there exist $h_1, h_2 \in \mathcal{H}$ such that $g = h_1 + h_2$. We consider the problem where, given $\mathcal{G}$, we want to find an $\mathcal{H}$ which explains $\mathcal{G}$ and with minimum possible size.

**Problem 1 (HAPLOTYPING POPULATION).** *Given a family $\mathcal{G}$ of genotypes, find a minimum size family $\mathcal{H}$ which explains $\mathcal{G}$.*

In the following, $\phi$ denotes the minimum size of an $\mathcal{H}$ which explains $\mathcal{G}$.

**Fact 2 (first upper bound: $\phi \leq 2m$).** *Given any input family $\mathcal{G}$, there always exists an $\mathcal{H}$ which explains $\mathcal{G}$ with $|\mathcal{H}| \leq 2|\mathcal{G}|$.*

*Proof:* Where $g$ is a genotype, let $h_1$ be the haplotype which is 0 whenever $g$ is 0 and 1 in all other positions. Let $h_0$ be the haplotype which is 1 whenever $g$ is 1 and 0 in all other positions. Then $g = h_0 + h_1$. $\square$

**Fact 3 (first lower bound: $m \leq \frac{\phi(\phi-1)}{2}$).** *Assume that $\mathcal{H}$ explains $\mathcal{G}$. Then $|\mathcal{G}| \leq \binom{|\mathcal{H}|}{2}$, that is, $|\mathcal{G}| \leq \frac{|\mathcal{H}|(|\mathcal{H}|-1)}{2}$.*

In other words, $\phi \geq \sqrt{2m}$.

**Corollary 4.** *Combining Facts 2 and 3 we get a $\sqrt{m}$-approximation algorithm.*

*Proof:* The proof of Fact 2 is constructive. $\square$

In fact, any poly-time algorithm which yields a feasible solution $\mathcal{H}$ can be turned into a $\sqrt{m}$-approximation algorithm (without worsening the quality of $\mathcal{H}$) by the following observation. (Where a family $\mathcal{H}$ is said to be *a minimal family which explains* $\mathcal{G}$ if $\mathcal{H}$ explains $\mathcal{G}$ and there exists no $\mathcal{H}' \subset \mathcal{H}$ that explains $\mathcal{G}$).

**Observation 5.** *Let $\mathcal{H}$ be a minimal family which explains $\mathcal{G}$. Then $|\mathcal{H}| \leq 2|\mathcal{G}|$.*

## 4 NP-completeness proof

An haplotype $h$ is said to be *good* for a genotype $g$ when there exists an haplotype $\overline{h}$ such that $g = h + \overline{h}$. In other words, $h$ is good for $g$ when $h[j] = g[j]$ whenever $g[j] \in \{0, 1\}$.

In this section, we give a simple APX-hardness proof which holds already when we have at most three "-" symbols per row.

The reduction is from NODE COVER, where we are given as input an undirected graph $G = (V, E)$ and we are asked to find a node cover $X \subseteq V$ of smallest possible cardinality. NODE COVER is known [11, 1, 2] to be APX-hard already for $\Delta(G) \leq 3$, that is, even when the input instances are restricted to be graphs of degree at most three. Our arguments also involve a classical theorem on node covers by Nemhauser and Trotter [10].

### 4.1 NODE COVER: the theorem of Nemhauser and Trotter

Let $G = (V, E)$ be an undirected simple graph on $\hat{n}$ nodes and $\hat{m}$ edges. A *node cover* is a vertex-set $X \subseteq V$ such that every edge in $E$ has at least one endpoint in $X$. Denote by $\Delta_G$ the maximum degree of a node and by $\sigma_G$ the minimum size of a node cover in $G$.

As usual, when $S \subset V$, then $G[S]$ denotes *the subgraph of $G$ induced by $S$*, i.e. $G[S] = (S, \{uv \in E : u, v \in S\})$. Moreover, when $X$ is a node cover of $G$, then we say that $X$ *covers* $G$. Here follows a classical theorem on node covers by Nemhauser and Trotter [10].

**Theorem 6 (Nemhauser and Trotter [10]).** *Given a graph $G = (V, E)$, introduce a new node $v'$ for every node $v \in V$. Let $V' = \{v' : v \in V\}$ and $F = \{uv' : uv \in E\}$. Consider the bipartite graph $H = (V, V'; F)$ on $2\hat{n}$ node and $2\hat{m}$ edges. Let $X$ be a minimum node cover of $H$. Let $Y = \{v : v \in X \wedge v' \in X\}$ and $Z = \{v : v \in X \vee v' \in X\}$. Then the following properties hold:*

**(i)** *if a set $D \subseteq Z$ covers $G[Z]$ then $D \cup Y$ covers $G$;*

**(ii)** *there exists an optimum cover of $G$ which contains $Y$;*

**(iii)** $\sigma_{G[Z]} \geq |Z|/2$.

Since a minimum node cover on a bipartite graph can be found in polynomial time, then in virtue of the above theorem we can always assume that $\sigma_G \geq \hat{n}/2$. (Indeed, if $Y \neq \emptyset$, then we can reduce the given instance by (ii). When eventually $Y = \emptyset$, then $Z = V$ or again we can reduce the given instance by (i). When eventually $Y = \emptyset$ and $Z = V$, then $\sigma_G \geq \hat{n}/2$ by (iii).) This inequality will be crucial in the derivation of our bounds.

## 4.2 APX-hardness proofs based on NODE COVER

Given a graph $G = (V, E)$, we construct a matrix $M$ with $\hat{m} + \hat{n}$ rows (genotypes) and $\hat{n}$ columns. The rows will be indexed by the elements in $V \cup E$ and the columns will be indexed by the elements in $V \cup \{s\}$, where $s$ is a special symbol. The entries are defined by the following equations.

$$\begin{array}{ll} M[u,u] = 0 & \text{for every } u \in V \\ M[u,v] = 1 & \text{for every } u, v \in V \text{ with } u \neq v \\ M[u,s] = 0 & \text{for every } u \in V \\ M[e,v] = - & \text{for every } v \in V \text{ and } e \in \delta(v) \\ M[e,v] = 1 & \text{for every } v \in V \text{ and } e \in E \text{ not incident at } v \\ M[e,s] = - & \text{for every } e \in E \end{array}$$

**Lemma 7.** *Let $X$ be a node cover of $G$. Then there exists an $\mathcal{H}_X$ explaining $M$ with $|\mathcal{H}_X| = \hat{n} + |X|$.*

*Proof:* For every $v \in V$, denote by $h_v$ the haplotype which is everywhere 1 except in column $v$, where $h_v[v] = 0$. Let $X$ be a node cover of $G$. Consider the family $\mathcal{H}_X$ which contains as haplotypes the $\hat{n}$ genotypes (rows) of $M$ indexed by elements in $V$ and all the $|X|$ haplotypes of the form $h_v$, $v \in X$. Note that $\mathcal{H}_X$ explains $M$. $\square$

**Lemma 8.** *Let $\mathcal{H}$ be an optimal haplotype family explaining $M$ and such that the number of 0's in haplotypes in $\mathcal{H}$ is minimum possible. Then $G$ admits a node cover $X_{\mathcal{H}}$ with $|X_{\mathcal{H}}| = |\mathcal{H}| - \hat{n}$.*

*Proof:* For every $v \in V$, denote by $h_v$ the haplotype which is everywhere 1 except in column $v$, where $h_v[v] = 0$. Let $\mathcal{H}$ be an optimal haplotype family explaining $M$. Then $\mathcal{H}$ contains as haplotypes all the $\hat{n}$ genotypes (rows) of $M$ indexed by elements in $V$, since these do not contain any "-" symbol. At this point we can observe that any single genotype in $M$ is either indexed by $V$ (and hence is already explained), or is indexed by $E$ (and hence can be explained by introducing a single haplotype of the form $h_v$, where $v \in E$. Therefore, the lemma follows once we have proven the following claim.

**Claim.** Let $h$ be an haplotype in $\mathcal{H}$ which is good for more than one genotype in $M$ indexed by $E$. Then $h$ is of the form $h_v$ for some $v \in V$.

**Proof:** Clearly, $h$ contains at most three 0's, otherwise $h$ would not be good for any genotype in $M$. Actually, $h$ contains at most two 0's, since otherwise $h$ would be good for at most one single genotype in $M$ indexed by $E$ (we can clearly assume that $G$ is simple). The same argument actually shows that if $h$ has precisely two 0's, than one 0 is in the column indexed by $s$, and assume that the other 0 is in the column indexed by node $v$. But then all genotypes in $M$ indexed by $E$ and compatible with $h$ would all correspond to edges incident at $v$ and would hence be all explained by $h_v$ plus the genotypes indexed by $V$ which are anyhow in $\mathcal{H}$. Hence, $\mathcal{H} \setminus \{h\} \cup \{h_v\}$ would also explain $M$ and would contradict the minimality assumptions for $\mathcal{H}$.

Assume $h$ has no component set to 0, that is, $h = \mathbf{1}$. Note however that haplotype $\mathbf{1}$ would combine only with haplotypes $h'$ which have precisely three 0's, two of which in correspondence of the endnodes of some edge $e_{h'}$ in $E$. Moreover, as mentioned above, $h'$ could not explain any genotype other than the one indexed by $e$. We can then remove $h$ from $\mathcal{H}$ if we also replace each such $h'$ in $\mathcal{H}$ with an haplotype $h_v$, with $v$ endpoint of $e_{h'}$. This contradicts the optimality of $\mathcal{H}$.

Finally, assume that $h$ has precisely one component set to 0. But then this component is not $s$, since otherwise this haplotype would combine only with haplotypes $h'$ which have precisely two 0's in correspondence of the endnodes of some edge $e_{h'}$ in $E$. Moreover $h'$ could not explain any genotype other than the one indexed by $e$. We can then remove $h$ from $\mathcal{H}$ if we also replace each such $h'$ in $\mathcal{H}$ with an haplotype $h_v$, with $v$ endpoint of $e_{h'}$. This contradicts the optimality of $\mathcal{H}$. □

**Lemma 9.** *Assume to have a $(1+\epsilon)$-approximation algorithm for HAPLOTYPING which holds on the restricted kind of instances involved in the reduction above proposed. Then one can develop a $(1+3\epsilon)$-approximation algorithm for NODE COVER.*

*Proof:* Assuming the minimum node cover has size OPT, then there exists an $\mathcal{H}_{opt}$ explaining $M$ with $|\mathcal{H}_{opt}| = \hat{n} + \text{OPT}$. By running the $(1+\epsilon)$-approximation algorithm for HAPLOTYPING we are hence guaranteed to find a solution $\mathcal{H}'$ with $|\mathcal{H}'| \leq (\hat{n}+\text{OPT})(1+\epsilon)$. And out from this $\mathcal{H}'$ we have shown how one can find (in polynomial time) a node cover $X$ of $G$ of size at most

$$|X| \leq (\hat{n}+\text{OPT})(1+\epsilon) - \hat{n} \leq \epsilon\hat{n} + \text{OPT} + \epsilon\ \text{OPT} \leq 2\epsilon\text{OPT} + \text{OPT} + \epsilon\ \text{OPT} = (1+3\epsilon)\ \text{OPT}.$$

## 5 $2^{k-1}$-Approximation algorithms from an LP formulation

Denote by $\hat{\mathcal{H}}_{\mathcal{G}}$, or more compactly by $\hat{\mathcal{H}}$, the set of those haplotypes which are compatible with some genotype in $\mathcal{G}$. We associate a binary 0/1 variable $x_h$ to every $h \in \hat{\mathcal{H}}$. The intended meaning of $x_h = 1$ is that $h$ is taken into the solution, whereas if $x_h = 0$ then $h$ is not taken into the solution Fix any total order on $\hat{\mathcal{H}}$ and denote by $\mathcal{P}$ the set of those pairs $(h_1, h_2)$ with $h_1, h_2 \in \hat{\mathcal{H}}$, $h_1 < h_2$. Introduce a binary 0/1 variable $y_{h_1,h_2}$ for every $(h_1, h_2) \in \mathcal{P}$. Moreover, for every $g \in \mathcal{G}$, let $\mathcal{P}_g := \{(h_1, h_2) \in \mathcal{P} \,|h_1 + h_2 = g\}$.

Consider the following IPL formulation of the HAPLOTYPING problem.

$$opt(\mathcal{G}) := \quad \min \textstyle\sum_{h\in\hat{\mathcal{H}}} x_h \qquad\qquad (1)$$
$$\begin{cases} \sum_{(h_1,h_2)\in\mathcal{P}_g} y_{h_1,h_2} \geq 1 & \text{for every } g \text{ in } \mathcal{G} \\ x_{h_1} \geq y_{h_1,h_2} & \text{for every } (h_1, h_2) \in \mathcal{P} \\ x_{h_2} \geq y_{h_1,h_2} & \text{for every } (h_1, h_2) \in \mathcal{P} \\ x \in \{0,1\}^{\hat{\mathcal{H}}},\ y \in \{0,1\}^{\mathcal{P}} \end{cases}$$

Let us report the dual of the fractional relaxation of the above LPI 1.

$$opt' := \quad \max \textstyle\sum_{g\in\mathcal{G}} \lambda_g \qquad\qquad (2)$$
$$\begin{cases} \sum_{(h_1,h_2)\in\mathcal{P}:h\in\{h_1,h_2\}}(\alpha_{h_1,h_2} + \beta_{h_1,h_2}) \leq 1 & \text{for every } h \text{ in } \hat{\mathcal{H}} \\ \lambda_g - \alpha_{h_1,h_2} - \beta_{h_1,h_2} \leq 0 & \text{for every } (h_1, h_2) \in \mathcal{P}_g \\ \lambda, \alpha, \beta \geq \mathbf{0} \end{cases}$$

Which can be more conveniently rewritten as follows.

$$opt' := \quad \max \textstyle\sum_{g\in\mathcal{G}} \lambda_g \qquad\qquad (3)$$
$$\begin{cases} \sum_{(h_1,h_2)\in\mathcal{P}:h\in\{h_1,h_2\}} \gamma_{h_1,h_2} \leq 1 & \text{for every } h \text{ in } \hat{\mathcal{H}} \\ \lambda_g - \gamma_{h_1,h_2} \leq 0 & \text{for every } (h_1, h_2) \in \mathcal{P}_g \\ \lambda, \gamma \geq \mathbf{0} \end{cases}$$

Assume that every genotype $g \in \mathcal{G}$ contains at most $k$ "-" symbols. Then the number of variables and equations in the above ILP is polynomial in the size of the input instance and we can solve the LP relaxation to optimality in polynomial time. Assume now to perform the following two actions in their sequential order:

1. scale up the value of every variable by a factor of $2^{k-1}$;

2. if the value of a variable is at least 1, then round it to 1, otherwise set the value of that variable to 0.

It is easy to check that operation 1. will not destroy feasibility of the solution at hand. Moreover, the value of the objective function will scale up by precisely a factor of $2^{k-1}$. It is easy to check that operation 2. will not destroy feasibility of the solution at hand. This is becouse, after operation 1., the left terms of the first kind of constraints will be at least $2^{k-1}$, and the number of variables contributing to each of such terms is at most $2^{k-1}$, hence at least one such variable per term will have value at least 1. Moreover, the validity of the second and third kind of constraints can clearly not be affected during operation 2. Finally, operation 2. can only decrease the objective function.

## 6 A randomized algorithm for haplotyping a population

The main results of this section are the following (remember that $m = |\mathcal{G}|$ and $\phi$ denotes the optimal solution value):

**Result 10 (Las Vegas).** *Let $k$ be the maximum number of "-" characters into a same row. We give a randomized algorithm, which, assuming that $\phi \leq m^\alpha$, will return a feasible solution $\mathcal{H}_{good}$ with $|\mathcal{H}_{good}| \leq 2^{k+2} m^\alpha \log 2m$. The running time of the algorithm is a concentrated random variable with expected value bounded by a polynomial in the size of the input instance.*

**Result 11 (Monte Carlo).** *We give a randomized algorithm, which will return almost surely (i.e. with probability at least $\frac{m-1}{m}$) a feasible solution $\mathcal{H}_{good}$ with $|\mathcal{H}_{good}| \leq 2^{k+2}\phi \log 2m$. The running time of the algorithm is bounded by a polynomial in the size of the input instance.*

### 6.1 The Las Vegas algorithm which knows about $\alpha$

Consider the following algorithm (Algorithm 6.1).

**Algorithm 1** HAPLOTYPE ($\mathcal{G}$)

```
 0.  $\mathcal{H}_{good} \leftarrow \emptyset$; $\mathcal{G}' \leftarrow \mathcal{G}$;
 1.  while $\mathcal{G}' \neq \emptyset$ do
 2.      repeat
 3.          $Temp \leftarrow \emptyset$;
 4.          for $4m^\alpha \log m$ times do
 5.              choose (uniformly) at random a $g \in \mathcal{G}'$;
 6.              put all haplotypes compatible with $g$ into $Temp$;
 7.      until $Temp$ explains at least $\frac{1}{2}$ of the genotypes in $\mathcal{G}'$;
 8.      $\mathcal{H}_{good} \leftarrow \mathcal{H}_{good} \cup Temp$;
 9.      remove from $\mathcal{G}'$ the genotypes already explained by $\mathcal{H}_{good}$;
10.  return $\mathcal{H}_{good}$.
```

In this lemma we settle the correctness and approximation guarantee issues.

**Lemma 12.** *When Algorithm 6.1 terminates, then it returns an haplotype family $\mathcal{H}_{good}$ which explains $\mathcal{G}$ and such that $|\mathcal{H}_{good}| \leq 2^{k+2} m^\alpha \log 2m$.*

*Proof:* When Algorithm 6.1 terminates, then by the condition of the while-loop 1–9 at step 1. we have that $\mathcal{G}' = \emptyset$, and hence $\mathcal{H}_{good}$ explains all genotypes in $\mathcal{G}$. Furthermore, by the condition of the repeat-loop 2–7 at step 7., we have that $|\mathcal{G}'|$ at least halves at every loop, hence the number of loops is bounded by $\log m$. Since at every loop we choose at most $4m^\alpha \log m$ different genotypes in $\mathcal{G}$, then $|Temp| \leq 2^k 4m^\alpha \log m = 2^{k+2} m^\alpha \log m$ whenever the repeat-loop 2–7 is exit. But then, at termination, $|\mathcal{H}_{good}| \leq 2^{k+2} m^\alpha \log 2m$. □

We remain with the analysis of the running time, which will take the remainder of this subsection.

By Fact 3, $\alpha \geq \frac{1}{2}$. We can also assume that $\alpha < 1$, since otherwise we can exploit Fact 2 to achieve a 2-approximation. Denote by $\mathcal{H}_{opt}$ an optimal solution to the haplotyping problem under consideration. Hence, $\phi = |\mathcal{H}_{opt}|$. Denote by $\hat{\mathcal{H}}_{\mathcal{G}}$ the set of those haplotypes which are compatible with some genotype in $\mathcal{G}$.

Even if not necessary in the analysis to follow, we like to think of a graph on node set $\hat{\mathcal{H}}_{\mathcal{G}}$ and having an edge $h_1h_2$ labeled $g$ whenever $g = h_1 + h_2$. (This graph is simple and the set of edges labeled with a same genotype is a matching). This motivates denoting by $d(h)$ the number of genotypes in $\mathcal{G}$ compatible with $h$. Clearly, $\mathcal{H}_{opt} \subseteq \hat{\mathcal{H}}_{\mathcal{G}}$. We define a node in $\mathcal{H}_{opt}$ *to be good* when $d(h) \geq \frac{m}{2m^\alpha}$ and *bad* otherwise.

**Observation 13.** *When $h$ is good, then the fraction of genotypes in $\mathcal{G}$ compatible with $h$ is at least $\frac{1}{2m^\alpha}$.*

*Proof:* By definition, $m := |\mathcal{G}|$. The ratio is therefore

$$\frac{d(h)}{|\mathcal{G}|} \geq \frac{\frac{m}{2m^\alpha}}{m} = \frac{1}{2m^\alpha}.$$

□

As a consequence, when at step 5. we choose a random $g \in \mathcal{G}$, then the probability that a good haplotype $h$ will be put in *Temp* at the following step 6. is at least $\frac{1}{2m^\alpha}$. Since within loop 4–6 this experiment is repeated $4m^\alpha \log m$ times, then it is almost sure that all good haplotypes end up into *Temp* during any full execution of loop 4–6. This is more formally stated into the following lemma.

**Lemma 14.** *Let $h$ be a good haplotype. Consider the random experiment of running the $4m^\alpha \log m$ iterations of loop 4–6. Then, when we get to the test in step 7., we have that $P[h \notin Temp] \leq \left(\frac{1}{m}\right)^2$. Furthermore, the probability that any good haplotype in $\mathcal{H}_{opt}$ is not in Temp is at most $\frac{1}{m}$. In other words, with high probability all good haplotypes are in Temp.*

*Proof:* Since $h$ is good, and where we assume that the random choices are independent, then

$$\begin{aligned} P[h \notin Temp] &\leq \textstyle\prod_{i=1}^{4m^\alpha \log m} \left(1 - \frac{1}{2m^\alpha}\right) = \left(1 - \frac{1}{2m^\alpha}\right)^{4m^\alpha \log m} = \\ &= \left(\left(1 - \frac{1}{2m^\alpha}\right)^{2m^\alpha}\right)^{2 \log m} \leq \left(\frac{1}{e}\right)^{2 \log m} = \left(\frac{1}{m}\right) 2 \end{aligned}$$

To bound the probability that any good haplotype does not make it into *Temp*, we employ the most simple fact that the probability of an union of events is at most the sum of the probabilities of the events themselves, no matter what are the dependencies between these events.

Hence, where $\mathcal{H}_{opt}^{good}$ is the set of good haplotypes in $\mathcal{H}_{opt}$, then

$$\begin{aligned} P[\exists h \in \mathcal{H}_{opt}^{good} \setminus Temp] &\leq \textstyle\sum_{h \in \mathcal{H}_{opt}^{good}} P[h \notin Temp] \leq \sum_{h \in \mathcal{H}_{opt}^{good}} \left(\frac{1}{m}\right)^2 = |\mathcal{H}_{opt}^{good}| \left(\frac{1}{m}\right)^2 \leq \\ &\leq m^\alpha \left(\frac{1}{m}\right) 2 = \left(\frac{1}{m}\right)^{2-\alpha} \leq \frac{1}{m}. \end{aligned}$$

**Corollary 15.** *Each time we get to the test at step 7., the probability to pass the test and exit the repeat-loop is at least* $\frac{1}{m}$.

*Proof:* By Lemma 14, we know that when the test at step 7. is faced, then the probability that any good haplotype into $\mathcal{H}_{opt}$ has not made it into $Temp$ is at most $\frac{1}{m}$. This means that with probability at least $\frac{m-1}{m}$ all genotypes which are not explained by $Temp$ have at least one endnode into bad haplotypes of $\mathcal{H}_{opt}$. Since $|\mathcal{H}_{opt}| \leq m^{\alpha}$ and since $d(h) < \frac{m}{2m^{\alpha}}$ holds for every bad node, this implies that the genotypes in $\mathcal{G}$ which are not explained by $Temp$ are at most

$$|\mathcal{H}_{opt}|d(h) < m^{\alpha}\frac{m}{2m^{\alpha}} = \frac{m}{2}$$

□

Now, where $T$ is the time for the whole algorithm (a random variable), $T_{try}$ is the time (a deterministic value) needed for a single attempt (loop 4–6), and if $p < \frac{1}{m}$ is the probability that the attempt goes wrong, then the expected running time is $E[T] = \sum_{i=0}^{\infty} p^i T'$ which is $O(T)$. Moreover, the distribution of the random variable $T$ is pretty much concentrated. (I.e. the probability that in a random experiment the actual value of $T$ at least doubles on $T' \leq E[T]$ is at most $p = \frac{1}{m}$, and actually even less).

## 5.2 The Monte Carlo algorithm which knows nothing

The Monte Carlo algorithm is simply a binary search on $\alpha$ within the interval $[\frac{1}{2}, 1)$. We know that when $\alpha$ is not guessed by defect, then it is almost sure that the attempt loop in the Las Vegas algorithm in the previous section goes good. We hence need only to show that, even assuming that all attempts are good, the number of steps in the binary search required to essentially get to the correct value of $\alpha$ is not too big. One could object that a real $\alpha$ can be approximated arbitrarily close, so that infinite steps are needed. However, consider the following observation.

**Observation 16.** *Let* $\epsilon = \frac{1}{t}$*. Then* $m^{\alpha+\epsilon} \leq 2m^{\alpha}$ *as soon as* $t \geq \log_2 m$.

*Proof:* Clearly, $m^{\alpha+\epsilon} \leq 2m^{\alpha}$ if and only if $m^{\frac{1}{t}} = m^{\epsilon} \leq 2$. And clearly, $m^{\frac{1}{t}} \leq 2$ if and only if $\frac{1}{t} \leq \log_m 2 = \frac{1}{\log_2 m}$. This holds when $t \geq \log_2 m$. □

Observation 16 amount perhaps to observing that the smallest $\alpha$ such that $\phi \leq m^{\alpha}$ occurs is a rational. We hence can obtain a Monte Carlo algorithm which does not assume the knowledge of $\alpha$ by just doubling the constant in the approximation ratio and by running the whole Algorithm 6.1 at most $\log m$ times. It is possible (omitted for space reasons) to shave a $\log m$ factor from the approximation ratio. We believe that with a deeper analysis, it is possible to shave also the remaining $\log m$ and achieve linear approximation.

## References

[1] P. Alimonti, V. Kann, Hardness of approximating problems on cubic graphs. Proc. 3rd Italian Conf. on Algorithms and Complexity, *Lecture Notes in Comput. Sci.* 1203, Springer-Verlag (1997) 288-298.

[2] P. Berman, M. Karpinski, On some tighter inapproximability results. ECCC Report TR98-029, June 1998. http://www.eccc.uni-trier.de/eccc/index.html

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

[4] A. Clark, Inference of haplotypes from PCR–amplified samples of diploid populations, *Molecular Biology Evolution* 7 (1990) 111–122.

[5] D. Gusfield, Inference of haplotypes from PCR–amplified samples of diploid populations: Complexity and algorithms, Tech. Rep. cse–99–6, U.C.S.D. (1999).

[6] D. Gusfield, A Practical Algorithm for Optimal Inference of Haplotypes from Diploid Populations, Proc. 8th Internat. Conf. on Intelligent Systems for Molec. Biol., AAAI Press (2000) 183–189.

[7] D. Gusfield, Haplotyping as Perfect Phylogeny: Conceptual Framework and Efficient Solutions, Proc. 6th ACM Internat. Conf. on Computational Biology (RECOMB), ACM Press (2002) 166–175.

[8] G. Lancia, V. Bafna, S. Istrail, R. Lippert and R. Schwartz, SNPs Problems, Complexity and Algorithms, Proc. 9th European Symposium on Algorithms (ESA), *Lecture Notes in Comput. Sci.*, 2161, Springer-Verlag (2001) 182–193.

[9] R. Lippert, R. Schwartz, G. Lancia and S. Istrail, Algorithmic Strategies for the SNPs Haplotype Assembly Problem, *Briefings in Bioinformatics* 3 (2002) 23–31.

[10] G.L. Nemhauser, L.E., Jr. Trotter, Vertex packings: structural properties and algorithms. *Math. Programming* 8 (1975), 232–248.

[11] C.H. Papadimitriou, M. Yannakakis, Optimization, approximation, and complexity classes. *J. Comput. System Sci.* 43 (1991), 425–440.

[12] R. Rizzi, V. Bafna, S. Istrail, and G. Lancia, Practical Algorithms and Fixed-Parameter Tractability of the Single Individual SNPs Haplotyping Problem, Proc. 2nd Workshop on Algorithms in Bioinformatics (WABI), *Lecture Notes in Comput. Sci.*, Springer-Verlag (2002).

[13] R. Schwartz, A. Clark and S. Istrail, Methods for Inferring Block-wise Ancestral History from Haploid Sequences: The Haplotyping Coloring Problem, Proc. 2nd Workshop on Algorithms in Bioinformatics (WABI), *Lecture Notes in Comput. Sci.*, Springer-Verlag (2002).