ISTITUTO DI ANALISI DEI SISTEMI ED INFORMATICA
"Antonio Ruberti"
CONSIGLIO NAZIONALE DELLE RICERCHE

L. Tinini, P. Bertolazzi, A. Godi, G. Lancia

COLLHAPS 2.0: A HEURISTIC APPROACH TO HAPLOTYPE INFERENCE BY PARSIMONY

R. 08-05, 2008

Leonardo Tininini – Istituto di Analisi dei Sistemi ed Informatica del CNR, viale Manzoni 30 - 00185 Roma, Italy. Email: `tininini@iasi.cnr.it`.

Paola Bertolazzi – Istituto di Analisi dei Sistemi ed Informatica del CNR, viale Manzoni 30 - 00185 Roma, Italy. Email: `bertola@iasi.cnr.it`.

Alessandra Godi – Istituto di Analisi dei Sistemi ed Informatica del CNR, viale Manzoni 30 - 00185 Roma, Italy. Email: `godi@iasi.cnr.it`.

Giuseppe Lancia – Dipartimento di Informatica e Matematica, Universitá di Udine, Udine, Italy. EMAIL: `lancia@dei.unipd.it`.

ISSN: 1128–3378

Abstract

Haplotype data play a relevant role in several genetic studies, e.g. mapping of complex disease genes, drug design and evolutionary studies on populations. However, the experimental determination of haplotypes is expensive and time-consuming. This motivates the increasing interest in techniques for inferring haplotype data from genotypes, which can instead be obtained quickly and economically. Several such techniques are based on the maximum parsimony principle, which has been justified by both experimental results and theoretical arguments. However, the problem of haplotype inference by parsimony was shown to be NP-hard, thus limiting the applicability of exact parsimony-based techniques to relatively small datasets. In this paper we introduce *collapse rule*, a generalization of the well-known Clark's rule, and describe a new heuristic algorithm for haplotype inference (implemented in a program called CollHaps), based on parsimony and the iterative application of collapse rules. The performance of CollHaps is tested on several datasets. The experiments show that CollHaps enables the user to process large datasets obtaining very "parsimonious" solutions in short processing times. They also show a correlation, especially for large datasets, between parsimony and correct reconstruction, supporting the validity of the parsimony principle to produce accurate solutions.

1. Introduction and preliminaries

Recent work in genome sequencing has shown that all humans share a large percentage of their DNA [1], the main differences lying in the number of copies of particular regions in the genome, commonly known as CNVs (Copy Number Variations) and in nucleotide variants at specific base positions, commonly known as SNPs (Single Nucleotide Polymorphisms). The variants at SNP sites are called *alleles* and SNPs are almost always biallelic, i.e. only two of the four possible variants have a non-negligible frequency (1% or greater). The exact sequence of alleles along each of the two chromosome copies in diploid organisms (e.g. humans) is called a *haplotype* and each SNP site is called either *homozygous* or *heterozygous*, depending on whether the alleles are the same or different on the two haplotypes.

The determination of individual haplotypes is fundamental in many practical contexts, e.g. detecting diseases and drug design, as well as in evolutionary studies on populations [2, 3]. Unfortunately, current routine sequencing technology can only determine conflated sequences, known as *genotypes*, where information on which are the alleles that came from the same chromosome copy is unknown for heterozygous sites. Experimental techniques to obtain individual haplotypes are very expensive, time-consuming and labor-intensive. This motivates the increasing interest in computational techniques for *haplotype inference*, i.e., techniques aimed at determining haplotype pairs from the corresponding genotypes [4, 5, 6].

As SNPs in humans are almost always biallelic, the two alleles at any SNP site are commonly encoded by the symbols 0 and 1, independently of the actual bases constituting the two variants. A haplotype h^* with m SNPs is therefore represented by an m-dimensional vector, such that each component $h^*_j \in \{0,1\}$. Similarly, a genotype g is represented by an m-dimensional vector, where each component $g_j \in \{0,1,2\}$: 0 and 1 relate to homozygous sites, while heterozygous sites are denoted by 2. We introduce the conflate operator $\oplus : \{0,1\} \to \{0,1,2\}$, defined as follows:

$\oplus$	0	1
0	0	2
1	2	1

which generalizes to vectors in the obvious way: given an m-dimensional genotype g and a pair of m-dimensional haplotypes h^*_1 and h^*_2,

$$g = h^*_1 \oplus h^*_2 \iff g_j = h^*_{1,j} \oplus h^*_{2,j} \ (j = 1, \ldots, m)$$

An instance of the Haplotype Inference Problem (*HIP-instance*) is constituted by an $n \times m$ matrix G such that each row g_i is an m-dimensional genotype. A corresponding (candidate) solution is a $2n \times m$ matrix H^* such that each row h^*_i is an m-dimensional haplotype and:

$$g_i = h^*_{2i-1} \oplus h^*_{2i} \ (i = 1, \ldots, n)$$

We call the pair of haplotypes h^*_{2i-1} and h^*_{2i} the *genotype solution* of g_i.

Given two haplotypes $h^*_1 = 10100$ and $h^*_2 = 00111$, the genotype g resulting from their conflation is given by $g = 10100 \oplus 00111 = 20122$. Viceversa, given the genotype $g = 20122$, the two haplotypes $h^*_1 = 10100$ and $h^*_2 = 00111$ constitute one of the possible genotype solution of g. □

One of the first (and most widely known) haplotype inference techniques was proposed by Clark [7]. The technique is based on the iterative application of a well-known inference rule and its main limitation is that it requires a suitable initial set of already inferred haplotypes. In this paper we introduce a generalization of Clark's inference rule, called *collapse rule*, and show some relevant properties related to it.

Several authors have proposed haplotype inference techniques based on the *maximum parsimony principle* [4]. According to this principle, one of the possible candidate solutions is sought, such that the number of distinct haplotypes in $\{h^*_1, h^*_2, \ldots, h^*_{2n}\}$ is minimum.

Maximum parsimony principle has been justified by both experimental results and theoretical arguments. As noted by Gusfield and Orzack [4], "the use of this principle is consistent with the fact that the number of observed distinct haplotypes in natural populations is vastly smaller than the number of possible haplotypes; this is also expected given the plausible assumption that the mutation rate at each site is small and recombination rates are low." Broadly speaking, since the number of haplotypes in natural populations is (relatively) small, if the "sample" of genotypes is sufficiently large, the reconstruction using the minimum number of distinct haplotypes has a high probability to be the right one. Furthermore, Clark's method itself [7] has been described by some authors as relying on a parsimony criterion [8, 9], and even Phase [10] (one of the currently most accurate program for haplotype inference) has been explained in terms of parsimony [11]. However, the problem of haplotype inference by parsimony was shown to be NP-complete [12] and APX-hard even in very restricted cases [6, 13], thus limiting the applicability of exact parsimony-based techniques to relatively small datasets.

In this paper we present a new heuristic algorithm for haplotype inference, based on the maximum parsimony principle and on the iterative application of collapse rules. In contrast to Clark's technique we show that the algorithm can be effectively applied even when the initial set of inferred haplotypes is empty. Furthermore, the heuristic approach enables the user to process large datasets (up to several hundreds of genotypes and SNPs, dimensions that are unprecedented for the parsimony problem) and hence to solve those above mentioned "sufficiently large" problem instances, where maximum parsimony has a high probability to produce good reconstructions.

Our approach is also in line with what conjectured by Gusfield and Orzack [4]: "It is conceivable that a program that adds heuristics to the Pure Parsimony criterion would produce results that are competitive with programs such as Phase." and our experimental results confirm this conjecture. The performance of the proposed algorithm is indeed tested on several datasets, which demonstrate: (i) its scalability properties, enabling the user to process large datasets in short processing times; (ii) the excellent compliance to the maximum parsimony principle (in the vast majority of tested datasets optimal parsimony solutions were obtained); and more importantly (iii) an apparent correlation, especially for large datasets, between parsimony and correct reconstruction, further confirming the validity of the parsimony principle to produce accurate solutions.

The paper is organized as follows. Related works are presented in Section 2. Section 3 provides a detailed description of the proposed algorithm, implemented in the CollHaps software. Experimental results are presented in Section 4, focused on both parsimony compliance and accuracy. Some conclusions are drawn in Section 5.

2. Related works

Clark [7] proposed a haplotype inference technique based on the iterative application of an inference rule, commonly known as "Clark's rule". The algorithm starts by identifying an initial set of unambiguously resolvable genotypes (i.e., the genotypes that have at most one heterozygous site each) and building the corresponding initial set R of resolved haplotypes. At each step, the algorithm attempts to resolve one of the residual genotypes g by using one of the haplotypes $\hat{h}_1$ in R and an inferred haplotype $\hat{h}_2$, such that $g = \hat{h}_1 \oplus \hat{h}_2$. If not already present, $\hat{h}_2$ is inserted into R. Probably, the most relevant problem of Clark's technique is that it strongly depends on the starting set R, which may or not result "powerful" enough for reaching good solutions: in extreme cases, e.g., if R is initially empty, the algorithm can not even be started and some completely random initial choices may be necessary. In our experiments, we compared the performance of our program CollHaps with Hapinferx, an implementation of the above-mentioned algorithm kindly provided by Prof. Clark.

A fairly wide class of techniques are based on the *maximum parsimony principle* described above and justified by both experimental results and theoretical arguments. A branch and bound algorithm based on maximum parsimony was proposed by Wang and Xu [14], while several Integer Linear Programming techniques have been proposed in the literature, e.g. those in [15, 16]. More recently, some techniques based on SAT-based formulations of the problem and SAT-solvers were also proposed [17, 18].

The main problem of these approaches is that haplotype inference by parsimony was shown to be NP-complete [12] and APX-hard even in very restricted cases [6, 13]. Their applicability is therefore limited to relatively small datasets. In their survey, Gusfield and Orzack [4] state that the exact approaches in the literature are practical for genotype data of up to 30 sites and 50 individuals. The recent work by Brown and Harrower [16] basically confirms such statement with the largest instances ranging from 30 sites and 50 individuals to 100 sites and 30 individuals. In fact the actual applicability of these techniques strictly depends on the heterozigosity (percentage of 2's) of the considered instance: even for datasets of only 28 genotypes and 23 SNPs we could not obtain any solution by using the ILP formulations proposed in the literature, due to the high heterozigosity of the considered datasets.

We show that CollHaps can produce approximate solutions (most times coincident, and in the remaining cases very close, to the most parsimonious) for problems larger by two or three orders of magnitude (i.e., such that $nm = 10^6$). This is very important because our experiments clearly show that the effectiveness of the maximum parsimony principle increases (i.e., the maximum parsimony solutions correspond to the actual haplotypes) with increasing number of individuals, e.g., hundreds or more. In other words, CollHaps can provide very parsimonious solutions for large problem instances, where parsimony leads to good error rates, but conventional parsimony-based techniques are unable to obtain any solution at all.

A broad variety of alternative approaches is based on statistical methods, trying to determine the haplotype frequencies iteratively and then to infer the haplotype pairs. In the methods based on expectation-maximization (EM) the haplotype frequency estimates are iteratively updated, starting from an initial guess and trying to maximize a likelihood function [19, 20, 21]. "Pure"-EM methods have severe limitations in terms of number of SNPs. More recently, however, Kimmel and Shamir [22] proposed a new EM-based algorithm for the simultaneous identification of haplotypes and SNP blocks, and implemented it in the Gerbil software. Gerbil has been shown to have good accuracy and fast processing times even on large datasets, and hence is included in our performance tests.

Other statistical methods are based on Bayesian inference and on the adoption of a more or less biologically-based prior, to obtain more accurate estimates of haplotype frequencies and consequently of the genotype reconstructions. The well-known program Phase [10] uses a prior based on the coalescent theory [23] and Gibbs sampling to derive the unknown haplotype frequencies. More recently, a slightly modified technique was proposed and implemented in the FastPhase software [24]. FastPhase assumes some forms of clustering among haplotypes, thus achieving faster processing times, although with slightly less accurate solutions. Gibbs sampling is also used by the program Haplotyper [9] which adopts Partition-Ligation (a divide-conquer-combine technique) to partition the haplotypes into smaller segments, thus obtaining fairly good processing times. Phase and Haplotyper are considered as two of the most accurate programs for haplotype inference, and are therefore included in our experiments. FastPhase was also considered for one family of datasets.

3. The CollHaps Algorithm

In this section we illustrate a new haplotype inference heuristic algorithm based on the maximum parsimony principle. The algorithm is based on the collapse rule concept (a generalization of Clark's rule) and an effective technique for the solution's progressive improvement.

3.1. The symbolic haplotype matrix

The CollHaps algorithm is based on the definition of a matrix (in one-to-one correspondence with the HIP-instance), which is constituted by both constants and variables, and continuously updated during the algorithm's execution. In this section we illustrate how to build this matrix, while in the following sections we show how such matrix will be progressively modified by the iterative application of collapse rules.

Let us consider a HIP-instance, i.e. an $n \times m$ matrix G of genotypes. Denote by V the overall number of '2' in G. We associate a distinct variable x_p (with p ranging from 1 to V) to each '2' in G. Let us denote with μ the bijective mapping that, given a pair (i, j) such that $g_{i,j} = 2$, returns the corresponding number $p = \mu(i, j)$ within $1, .., V$.

For each genotype g_i two *symbolic haplotypes* h_{2i-1} and h_{2i} are consequently derived, defined by:

$$h_{2i-1,j} = \begin{cases} 0 & \text{if } g_{i,j} = 0 \\ 1 & \text{if } g_{i,j} = 1 \\ x_{\mu(i,j)} & \text{if } g_{i,j} = 2 \end{cases}$$

$$h_{2i,j} = \begin{cases} 0 & \text{if } g_{i,j} = 0 \\ 1 & \text{if } g_{i,j} = 1 \\ \bar{x}_{\mu(i,j)} & \text{if } g_{i,j} = 2 \end{cases}$$

The variables x_p take values in $\{0,1\}$ and $\bar{x}_p$ is a shorthand for $1-x_p$ (obviously $\bar{\bar{x}}_p = x_p$). The variables x_p and $\bar{x}_p$ are called *complementary* and the $2n \times m$ matrix H, whose rows are the h_i's is called the *symbolic haplotype matrix.* According to the maximum parsimony principle, we want to determine a variable assignment for the variables x_p such that the resulting number of distinct haplotypes (i.e. the distinct rows of the matrix H) is minimum.

In the following example we show the symbolic haplotype matrix corresponding to a HIP-instance and two variable assignments (i.e. two candidate solutions) for such instance, one of which is maximally parsimonious, i.e. optimal in terms of the maximum parsimony principle.

Let us consider a HIP-instance constituted by the three genotypes: 1022, 2220, 2202, i.e. by the matrix G:

$$G = \begin{pmatrix} 1 & 0 & 2 & 2 \\ 2 & 2 & 2 & 0 \\ 2 & 2 & 0 & 2 \end{pmatrix}$$

The corresponding symbolic haplotype matrix is given by:

$$H = \begin{pmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \end{pmatrix} = \begin{pmatrix} 1 & 0 & x_1 & x_2 \\ 1 & 0 & \bar{x}_1 & \bar{x}_2 \\ x_3 & x_4 & x_5 & 0 \\ \bar{x}_3 & \bar{x}_4 & \bar{x}_5 & 0 \\ x_6 & x_7 & 0 & x_8 \\ \bar{x}_6 & \bar{x}_7 & 0 & \bar{x}_8 \end{pmatrix}$$

A candidate solution is obtained by considering an assignment for the variables $x_1, \ldots, x_8$. For example, the following variable assignment: x_1=0, x_2=0, x_3=1, x_4=0, x_5=0, x_6=1, x_7=0, x_8=0 corresponds to a candidate solution using 4 haplotypes, namely $h_a^* = h_1^* = h_3^* = h_5^* = 1000$, $h_b^* = h_2^* = 1001$, $h_c^* = h_4^* = 0110$ and $h_d^* = h_6^* = 0101$:

$$H_1^* = \begin{pmatrix} h_a^* \\ h_b^* \\ h_a^* \\ h_c^* \\ h_a^* \\ h_d^* \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}$$

while the variable assignment: x_1=0, x_2=1, x_3=1, x_4=0, x_5=1, x_6=0, x_7=1, x_8=0 corresponds to a (maximally parsimonious) solution using 3 haplotypes, namely $h_e^* = h_1^* = h_6^* = 1001$, $h_f^* = h_2^* = h_3^* = 1010$ and $h_g^* = h_4^* = h_5^* = 0100$:

$$H_2^* = \begin{pmatrix} h_e^* \\ h_f^* \\ h_f^* \\ h_g^* \\ h_g^* \\ h_e^* \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix}$$

$\square$

The *collapse rule*, to be described in the next section, arises from the following simple observation: each variable assignment on H induces a partition of its rows (the symbolic haplotypes h_1, h_2, …, h_{2n}) into classes: two rows (symbolic haplotypes) h_p and h_q are in the same partition class iff the variable assignment maps them to the same haplotype.

In this perspective a variable assignment is optimal (with respect to the maximum parsimony principle) iff the number of classes induced by such assignment is minimum. A collapse rule forces two symbolic haplotypes (matrix rows) h_p and h_q to be in the same partition class, by assigning the minimum possible number of variables in h_p and h_q to constants. As a consequence, h_p and h_q may still contain some variables, even after they have been collapsed (forced to be equal).

3.2. The collapse rule

In this section we introduce the concept of collapse rule and show that it generalizes Clark's rule to symbolic haplotypes. A collapse rule corresponds to the minimum set of variable assignments which lead two symbolic haplotypes to become identical, i.e. to be in the same partition class mentioned above. A prerequisite for the application of a collapse rule to a pair of symbolic haplotypes h' and h'' is their *compatibility.*

(compatible symbolic haplotypes) We say that two k-dimensional symbolic haplotypes h' and h'' are compatible iff for each $j \in \{1, \dots, k\}$ one of the following holds:

1. $h'_j = h''_j = 0$
2. $h'_j = h''_j = 1$
3. either h'_j or h''_j, but not both, is a variable
4. both h'_j and h''_j are variables and not complementary

$\square$

In other words, two symbolic haplotypes are compatible if they can be made identical by an opportune assignment on the variables.

Let us consider the three symbolic haplotypes:

$$\begin{array}{ll} h' & = (\ 0 \quad 1 \quad x_2 \quad 0\ ) \\ h'' & = (\ \bar{x}_3 \quad 0 \quad 0 \quad x_5\ ) \\ h''' & = (\ x_1 \quad x_4 \quad \bar{x}_2 \quad 0\ ) \end{array}$$

h' is incompatible with h'' (as $h'_2 = 1$ but $h''_2 = 0$) and h''' (as h'_3 and h'''_3 are complementary and any assignment on x_2 will always make $h' \neq h'''$), while h'' and h''' are compatible (e.g. they can be made equal to the haplotype 0000 by the variable assignment $x_1 = 0$, $x_2 = 1$, $x_3 = 1$, $x_4 = 0$, $x_5 = 0$). $\square$

Given a pair of compatible symbolic haplotypes, we want to determine a variable assignment that makes them identical. We are especially interested in the assignment forcing the *minimum* number of variables to be bound to constants. This is captured by the concept of collapse rule.

(collapse rule) Given two compatible symbolic haplotypes h', h'' the collapse rule (for h', h'') is the variable assignment ϑ defined as follows:

1. if $h'_j = x_p$ and h''_j is a constant $c \in \{0, 1\}$ then $\vartheta(x_p) = c$
2. if $h'_j = \bar{x}_p$ and h''_j is a constant $c \in \{0, 1\}$ then $\vartheta(x_p) = 1 - c$
3. if h'_j is a constant $c \in \{0, 1\}$ and $h''_j = x_q$ then $\vartheta(x_q) = c$
4. if h'_j is a constant $c \in \{0, 1\}$ and $h''_j = \bar{x}_q$ then $\vartheta(x_q) = 1 - c$
5. if ($h'_j = x_p$ and $h''_j = x_q$) or ($h'_j = \bar{x}_p$ and $h''_j = \bar{x}_q$) then $\vartheta(x_q) = x_p$
6. if ($h'_j = x_p$ and $h''_j = \bar{x}_q$) or ($h'_j = \bar{x}_p$ and $h''_j = x_q$) then $\vartheta(x_q) = \bar{x}_p$
7. ϑ is the identity for any other variable

□

As already stated, a collapse rule corresponds to the set of variable assignments, which lead two symbolic haplotypes to become identical, although requiring the minimum number of assignments to constant values.

Given the two symbolic haplotypes h'' and h''' of the previous example:

$$\begin{aligned} h'' &= (\ \bar{x}_3\ \ 0\ \ 0\ \ x_5\ ) \\ h''' &= (\ x_1\ \ x_4\ \ \bar{x}_2\ \ 0\ ) \end{aligned}$$

the corresponding collapse rule (for h'', h''') is given by: $\vartheta(x_1) = \bar{x}_3$ (subrule 6), $\vartheta(x_4) = 0$ (subrule 3), $\vartheta(x_2) = 1$ (subrule 4), $\vartheta(x_5) = 0$ (subrule 1), $\vartheta(x_3) = x_3$ (subrule 7), and the result of its application make both of them equal to the following symbolic haplotype

$$(\ \bar{x}_3\ \ 0\ \ 0\ \ 0\ )$$

where the variable x_3 is not bound to any constant value and can be arbitrarily assigned. □

In general the same variable will have multiple occurrences within a given symbolic haplotype matrix and the assignment expressed by a collapse rule will have to be applied on all occurrences of that variable. In other words the application of a collapse rule is not "local" to the two collapsed symbolic haplotypes, but needs to be propagated to all matrix rows (symbolic haplotypes), sharing one or more variables with the collapsed ones. This process constitutes a collapse step.

(collapse step: application of collapse rule on a symbolic haplotype matrix) Given a symbolic haplotype matrix H and the collapse rule ϑ (corresponding to a pair of compatible symbolic haplotypes h' and h'' in H), the application of the collapse rule ϑ on all symbolic haplotypes in H is called collapse step and the resulting matrix denoted by $\vartheta(H)$. □

Let us consider the following symbolic haplotype matrix (corresponding to the HIP-instance constituted by the two genotypes 1022 and 2220):

$$H = \begin{pmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \end{pmatrix} = \begin{pmatrix} 1 & 0 & x_1 & x_2 \\ 1 & 0 & \bar{x}_1 & \bar{x}_2 \\ x_3 & x_4 & x_5 & 0 \\ \bar{x}_3 & \bar{x}_4 & \bar{x}_5 & 0 \end{pmatrix}$$

By collapsing h_2 with h_3 also h_1 and h_4 are affected, as they share variables with h_2 and h_3. Therefore the resulting symbolic haplotype matrix is the following:

$$\vartheta(H) = \begin{pmatrix} 1 & 0 & x_1 & 1 \\ 1 & 0 & \bar{x}_1 & 0 \\ 1 & 0 & \bar{x}_1 & 0 \\ 0 & 1 & x_1 & 0 \end{pmatrix}$$

Note that after collapsing h_2 with h_3, they are necessarily equal, but the variable x_1 can still be assigned. □

It is easily seen that Clark's rule is a particular form of collapse rule, corresponding to the case where one of the two haplotypes to be collapsed is unambiguous (i.e. does not contain variables). In other words Clark's rule is just a special case of collapse rule, but the converse does not hold.

As noted above, a collapse step (for h', h'') makes the symbolic haplotypes h' and h'' equal and hence after the application of a collapse rule the number of distinct rows in the symbolic haplotype matrix H is decreased by (at least) one. The following proposition states a fundamental property of the collapse rule (in the following we denote by "maximally parsimonious" a solution which is optimal according to the maximum parsimony principle, i.e. a solution using the minimum possible number of distinct haplotypes).

Given one maximally parsimonious solution for an HIP-instance, there always exists a sequence of collapse steps which produces a set of maximally parsimonious solutions including at least the given one.

Proof. The construction of the sequence is simple: we have seen that an HIP solution is a variable assignment for the variables in the symbolic haplotypes which defines a partition on the list of symbolic haplotypes. For each of the N partition classes defined by the given maximally parsimonious solution, we apply a sequence of collapse steps on the member haplotypes in order to make them all equal (if the class has cardinality n_c this is performed in $n_c - 1$ steps and the order of application is immaterial). The sequence of collapse steps will yield exactly N distinct haplotypes, which may be still partially symbolic, i.e. contain some variables. Any solution obtained by arbitrarily assigning 0/1 values to the residual variables is maximally parsimonious and one of them is obviously the given one. ∎

Given the HIP-instance of Example 3.1 and its optimal solution, one possible sequence of collapse steps (the order by which the rule is applied to the single pairs is unimportant) is: (h_1, h_6), (h_2, h_3), (h_4, h_5). The initial matrix of symbolic haplotypes is the following:

$$\begin{pmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \end{pmatrix} = \begin{pmatrix} 1 & 0 & x_1 & x_2 \\ 1 & 0 & \bar{x}_1 & \bar{x}_2 \\ x_3 & x_4 & x_5 & 0 \\ \bar{x}_3 & \bar{x}_4 & \bar{x}_5 & 0 \\ x_6 & x_7 & 0 & x_8 \\ \bar{x}_6 & \bar{x}_7 & 0 & \bar{x}_8 \end{pmatrix}$$

The first application of the collapse rule (for h_1, h_6) corresponds to the following variable assignments: x_1=0, x_6=0, x_7=1, x_8=$\bar{x}_2$ and produces the following matrix:

$$\begin{pmatrix} 1 & 0 & 0 & x_2 \\ 1 & 0 & 1 & \bar{x}_2 \\ x_3 & x_4 & x_5 & 0 \\ \bar{x}_3 & \bar{x}_4 & \bar{x}_5 & 0 \\ 0 & 1 & 0 & \bar{x}_2 \\ 1 & 0 & 0 & x_2 \end{pmatrix}$$

The second application (for h_2, h_3) corresponds to the variable assignment: x_2=1, x_3=1, x_4=0, x_5=1 and produces the following matrix:

$$\begin{pmatrix} 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix}$$

The final application (to h_4, h_5) is not necessary, as h_4 and h_5 have already been made identical by the previous assignments. □

Proposition 3.2 is of great theoretical relevance, as it ensures that a maximally parsimonious solution can always be reached by performing an opportune sequence of collapse steps. However, it does not provide a practical strategy to obtain the optimal sequence of collapse steps. Since the exhaustive exploration of all collapse sequences is infeasible in practice, a heuristic strategy is required.

Clearly, the maximum parsimony principle corresponds to perform the maximum possible number of collapse steps (as already noted, at each collapse step the number of distinct rows in the symbolic haplotype matrix H is decreased by one). Broadly speaking, the basic idea of the algorithm is to defer the binding of a variable to a specific constant value as much as possible: this is intuitive, as the probability of introducing incompatibility increases with the binding of variables to constants, while we are trying to avoid incompatibility in order to perform more collapse steps. The actual strategy is in fact more complex and includes several refinements for local exploration of promising sequences and for escaping from local minima. The overall strategy is described below and was implemented in a program called CollHaps.

CollHaps performs a given number of complete attempts, updating the overall solution whenever an improvement is obtained. By default, the number of complete attempts is set to 30, but a different value can be specified by setting a specific command-line parameter. Each complete attempt starts the computation from scratch, i.e. from the original matrix H and performs a sequence of collapse steps (driven by a randomized quasi-greedy strategy) until possible (see Sect. 3.4), followed by a reduction of the haplotypes used, based on a simple greedy technique (see Sect. 3.5). An iterative improvement of this "local" best solution is then attempted. This is obtained by some "precollapse" steps based on the haplotypes most extensively used in the local best (see Sect. 3.6), followed by conventional collapse steps based on the above mentioned randomized quasi-greedy strategy. The local search is interrupted after maxNoLocal consecutive unsuccessful iterations. By default, maxNoLocal is set to 15 but a different value can be used by setting a specific command-line parameter. Whenever necessary, residual variables in the final overall solution are finally removed to produce the result (see Sect. 3.7).

3.3. The preprocessing

Like most programs for HIP, CollHaps performs an initial preprocessing step, trying to reduce the number of symbolic haplotypes on which the collapse rules are to be applied. This is obtained by removing duplicate genotypes, and is transparent to the user. Furthermore, the access to haplotype data is mediated by an indexing structure to speed-up the computation: in practice, at each step one of the two collapsed haplotypes should be removed from the computation, as it has been made identical to the other one and this may cause a substantial memory overhead. Instead, collapsed haplotypes are only virtually removed (by modifying their position in the indexing structure) during the processing, thus enabling the program to only access the relevant data and greatly reducing the amount of memory accesses, as if the matrix size was progressively reduced by the computation.

3.4. The heuristic sequence of collapse steps

As a collapse rule is a generalized version of Clark's rule, Clark's algorithm can be seen as a sequence of collapse steps. Its main drawback is due to the specific criterion used to choose the pairs of haplotypes to be collapsed: as one of the two haplotypes must be unambiguous (i.e. must not contain variables) a sufficiently large initial set of unambiguous haplotypes is required to start the algorithm, and even in this case some genotypes may, at the end, remain unexplained.

We propose a different criterion to choose the haplotypes to be collapsed, which does not require any initial set of unambiguous haplotypes. In contrast to Clark's algorithm, we try to defer the binding of a variable to a specific constant value as much as possible: this is intuitive, as the probability of introducing incompatibility increases with the binding of variables to constants, while we are trying to avoid incompatibility in order to perform more collapse steps (note that each collapse step decreases by at least one the number of distinct haplotypes and, consequently, the objective of performing the maximum number of collapse steps is almost coincident with that of maximum parsimony). Consequently, the pair to be collapsed is chosen from among those which produce the smallest number of variable bindings: a distance matrix D is maintained by the algorithm, where the generic cell $D_{i,j}$ represents the "distance" $d(h_i, h_j)$, i.e., the number of variables that must be assigned to constants to collapse h_i and h_j. The distance matrix D is obviously symmetric and if two haplotypes are incompatible a conventional "infinite" value is assumed for their distance.

Given the symbolic haplotype matrix H of Example 3.1 the corresponding distance matrix D is the following:

$$H = \begin{pmatrix} 1 & 0 & x_1 & x_2 \\ 1 & 0 & \bar{x}_1 & \bar{x}_2 \\ x_3 & x_4 & x_5 & 0 \\ \bar{x}_3 & \bar{x}_4 & \bar{x}_5 & 0 \\ x_6 & x_7 & 0 & x_8 \\ \bar{x}_6 & \bar{x}_7 & 0 & \bar{x}_8 \end{pmatrix} D = \begin{pmatrix} - & \infty & 3 & 3 & 3 & 3 \\ \infty & - & 3 & 3 & 3 & 3 \\ 3 & 3 & - & \infty & 2 & 2 \\ 3 & 3 & \infty & - & 2 & 2 \\ 3 & 3 & 2 & 2 & - & \infty \\ 3 & 3 & 2 & 2 & \infty & - \end{pmatrix}$$

For example the distance indicated in the 3-rd column of the second row of the distance matrix is 3, because three variables need to be assigned to constants to make h_2 equal to h_3, namely $x_2 = 1$, $x_3 = 1$

and $x_4 = 0$. Note that the assignment of variables to other variables (in the example above $x_5 = \bar{x}_1$) is not considered for distance contributions. $\square$

Since a purely-greedy technique can sometimes lead the algorithm to (incorrect) local minima, a randomized quasi-greedy choice of the haplotypes to be collapsed was adopted. The randomization is such that pairs at the minimum distance d in D have a probability of being chosen which is twice the probability of a pair at distance $d+1$, which in turn is twice the probability of a pair at distance $d+2$, etc. In other words, distance acts as a preference measure, so that pairs at lower distance have a higher probability of being randomly chosen than those at higher distance. The sequence of collapse steps stops when the distance matrix contains infinite values for all haplotype pairs and hence no further collapse steps can be performed.

Let us consider a sequence of collapse steps corresponding to the following symbolic haplotype matrix H (i.e. to the set of genotypes $G = \{122, 102, 121\}$) and distance matrix D. Note that after h' and h'' have been collapsed (and hence made equal), the latter is "virtually" removed (see below) from the following steps. In particular the corresponding rows and columns in the distance matrix are not updated (as they are not considered by the algorithm) and have been conventionally indicated with asterisks.

$$H = \begin{pmatrix} 1 & x_1 & x_2 \\ 1 & \bar{x}_1 & \bar{x}_2 \\ 1 & 0 & x_3 \\ 1 & 0 & \bar{x}_3 \\ 1 & x_4 & 1 \\ 1 & \bar{x}_4 & 1 \end{pmatrix} D = \begin{pmatrix} - & \infty & 1 & 1 & 1 & 1 \\ \infty & - & 1 & 1 & 1 & 1 \\ 1 & 1 & - & \infty & 2 & 2 \\ 1 & 1 & \infty & - & 2 & 2 \\ 1 & 1 & 2 & 2 & - & \infty \\ 1 & 1 & 2 & 2 & \infty & - \end{pmatrix}$$

$$\Downarrow (h_1, h_3)$$

$$H = \begin{pmatrix} 1 & 0 & x_2 \\ 1 & 1 & \bar{x}_2 \\ 1 & 0 & x_2 \\ 1 & 0 & \bar{x}_2 \\ 1 & x_4 & 1 \\ 1 & \bar{x}_4 & 1 \end{pmatrix} D = \begin{pmatrix} - & \infty & * & \infty & 2 & 2 \\ \infty & - & * & \infty & 2 & 2 \\ * & * & * & * & * & * \\ \infty & \infty & * & - & 2 & 2 \\ 2 & 2 & * & 2 & - & \infty \\ 2 & 2 & * & 2 & \infty & - \end{pmatrix}$$

$$\Downarrow (h_1, h_5)$$

$$H = \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix} D = \begin{pmatrix} - & \infty & * & \infty & * & \infty \\ \infty & - & * & \infty & * & \infty \\ * & * & * & * & * & * \\ \infty & \infty & * & - & * & \infty \\ * & * & * & * & * & * \\ \infty & \infty & * & \infty & * & - \end{pmatrix}$$

We suppose in the first step h_1 and h_3 are randomly chosen to be collapsed by the algorithm (they are one of the pairs at the minimum distance 1 and hence with the highest probability to be chosen). In the second step the best pairs are at distance 2, so let us suppose that h_1 and h_5 are randomly chosen to be collapsed. At this point the four distinct haplotypes are all pairwise at infinite distance and no further collapse steps can therefore be performed. The solution can be directly extracted from the symbolic haplotype matrix, namely:

$$\begin{aligned} 122 &= 101 \oplus 110 \\ 102 &= 101 \oplus 100 \\ 121 &= 101 \oplus 111 \end{aligned}$$

However, we show in the following section on haplotype set reduction that this direct solution can be improved by defining a proper subset of the haplotypes in the haplotype matrix, still capable of explaining all genotypes in the considered HIP-instance. A reduction can be obtained only if at least one of the genotypes can be explained by two or more pairs taken from the obtained set of haplotypes.

For example in our case the set of haplotypes is $\mathcal{H} = \{101, 110, 100, 111\}$ and the genotype 122 can be explained by $101 \oplus 110$, as well as by $100 \oplus 111$. In particular, it will be shown in the next section that the direct solution found here (using 4 distinct haplotypes) can be improved by discarding one of the 4 haplotypes and producing a different solution that uses only 3 haplotypes. □

The computation and update of the distance matrix, as well as the determination of the haplotype pair to be collapsed, are key issues of the proposed technique and a substantial effort was consequently devoted to improve their efficiency. The computation from scratch of D is performed only at the beginning. In all following steps only the cells that could have been modified by the last collapse step are examined and updated.

Furthermore, only one of the two symbolic haplotypes made identical by a collapse step is maintained in the computation (the other one is "virtually" removed by an index structure) and the size of the distance matrix to be updated is consequently decreased by one at each step. In order to support the determination of the pair to be collapsed (which requires the number of pairs at each distance value and for each symbolic haplotype) a cardinality matrix is computed (and progressively updated) storing for each row of the haplotype matrix the number of haplotypes at the various distance values. In this way an improvement of an order of magnitude (and even more for large instances) was obtained with respect to an initial "naive" implementation.

3.5. Haplotype set reduction

The sequence of collapse rule applications produces a set $\mathcal{H}$ of haplotypes on which a further reduction can be applied, as some genotypes can be explained in more than one way by using the haplotypes in $\mathcal{H}$. For each genotype a list of possible solutions is built using the haplotypes in $\mathcal{H}$. The list for each genotype may contain a unique solution (if there is only one pair of haplotypes in $\mathcal{H}$ which can explain it) or some alternatives. Note also that the set $\mathcal{H}$ produced by a sequence of collapse steps may still contain some (at least partially) symbolic haplotypes, i.e. containing one or more variables.

The reduction is based on a greedy technique: a set $\mathcal{U} \subseteq \mathcal{H}$ of haplotypes is initially built comprising all haplotypes which are in unique solutions. At each step: (i) either at least one genotype with multiple genotype solutions can be explained by a pair of haplotypes in $\mathcal{U}$ (and in this case all other solutions are discarded) or (ii) the "most useful" pair of haplotypes in one of the solution lists is promoted and inserted into $\mathcal{U}$. The usefulness of a pair (genotype solution) is computed taking into account the number of genotype solutions in which the component haplotypes are involved, and the number of other genotype solutions discarded by that solution (since the haplotypes in $\mathcal{H}$ may contain variables, the adoption of a specific solution may require some variable assignments, which produce incompatibilities and hence the discarding of other solutions).

Let us consider the set $\mathcal{H} = \{101, 110, 100, 111\}$ of haplotypes produced by the sequence of collapse steps in Example 3.4. The initial lists of possible solutions using the haplotypes in $\mathcal{H}$ are the following:

- $g_1 : \{(100, 111), (101, 110)\}$
- $g_2 : \{(100, 101)\}$
- $g_3 : \{(101, 111)\}$

As g_2 and g_3 have unique solutions, the corresponding haplotypes are included in $\mathcal{U}$, i.e. $\mathcal{U} = \{100, 101, 111\}$. g_1 can now be explained by using only haplotypes in U and the alternative solution $(101, 110)$ is discarded. After reduction the solution is therefore the following:

$$\begin{aligned} 122 &= 100 \oplus 111 \\ 102 &= 100 \oplus 101 \\ 121 &= 101 \oplus 111 \end{aligned}$$

where the number of required haplotypes has been reduced from 4 to 3. □

This reduction is typically more effective in the first steps of the computation, when the quality (in terms of parsimony) of the solution is still fairly low. Conversely, it produces less relevant (but still important) reductions when the solution has been refined by several consecutive alternate applications of precollapsing (see below) and collapsing steps.

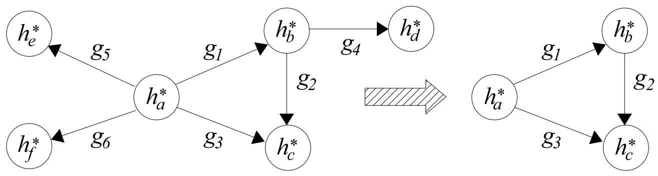

Figure 1: Selecting some "promising" haplotypes

3.6. Precollapsing

When an iteration produces an improved solution, a further exploration is initiated partially reusing that solution. In practice, given the previous solution, a subset of the genotype solutions is extracted corresponding to pairs of "promising" haplotypes (i.e. haplotypes that are used in several solutions and paired with other promising haplotypes). This subset is used to perform a preliminary sequence of collapse steps, followed by a conventional sequence (i.e. driven by the distance matrix).

Let us suppose that an improved solution was determined in the last iteration:

- $g_1 : \{(h_a^*, h_b^*)\}$
- $g_2 : \{(h_b^*, h_c^*)\}$
- $g_3 : \{(h_a^*, h_c^*)\}$
- $g_4 : \{(h_b^*, h_d^*)\}$
- $g_5 : \{(h_a^*, h_e^*)\}$
- $g_6 : \{(h_a^*, h_f^*)\}$

This solution is graphically illustrated in Figure 1. Edges have been oriented to graphically distinguish the first and second haplotype of a solution pair: although the order in the pair is irrelevant in general, it is important to correctly determine the precollapsing sequence. The figure also illustrates the set of "promising" haplotypes selected from the initial ones: more specifically, such set is given by $\mathcal{P} = \{(h_a^*, h_b^*, h_c^*)\}$, as these haplotypes are used at least twice in the solutions and paired with each other. Instead, h_d^*, h_e^* and h_f^* are used only once and removed. In practice, since a maximum parsimony solution corresponds to a graph having the genotypes as edges (and hence fixed) and which maximizes the average node degree, the algorithm tries to preserve subgraphs having a relatively high average node degree.

The genotypes (oriented edges in the figure) that can be explained by using only haplotypes in $\mathcal{P}$ (i.e. g_1, g_2 and g_3) determines the sequence of precollapsing. According to the rightmost graph of Figure 1 we impose:

- the first symbolic haplotype of g_1 to be equal to the first symbolic haplotype of g_3
- the second symbolic haplotype of g_1 to be equal to the first symbolic haplotype of g_2
- the second symbolic haplotype of g_3 to be equal to the second symbolic haplotype of g_2

Rephrased in terms of the initial symbolic haplotype matrix H (having the symbolic haplotypes $h_1, h_2, \ldots, h_{12}$ as rows) this means that the following precollapsing steps will be performed:

- collapse h_1 (first symbolic haplotype of g_1) with h_5 (first symbolic haplotype of g_3)
- collapse h_2 (second symbolic haplotype of g_1) with h_3 (first symbolic haplotype of g_2)
- collapse h_4 (second symbolic haplotype of g_2) with h_6 (second symbolic haplotype of g_3)

$\square$

Once the precollapsing steps have been completed, the algorithm performs a conventional sequence of collapse steps as illustrated above, by using the distance matrix and the randomized quasi-greedy strategy.

3.7. Postprocessing: removing residual variables

Even if the algorithm has reached an optimal number of haplotypes some of them may still contain some variables. This corresponds to multiple (equivalent in terms of the maximum parsimony principle) solutions. In these cases, CollHaps produces a double output: one without variables, where a specific constant value is assigned to each variable, and one with variables (symbolic solution), corresponding to a set of possible solutions. It can be easily shown that:

Given a symbolic solution, if r is the number of residual variables and k is the number of genotypes whose explanation contains at least one variable, then the number of corresponding distinct solutions is given by 2^{r-k}.

According to the maximum parsimony principle, each of these 2^{r-k} distinct solutions is optimal and could be randomly assigned. However, to obtain better performance in terms of error rates, a different strategy (inspired by the coalescent model) was adopted: given a haplotype h_v containing some variables, the (variable-free) haplotype h_f requiring the fewest SNP switches to be transformed into h_v is searched for, and the variables in h_v are assigned according to the corresponding constant values in h_f.

Let us consider an HIP-instance constituted by only two genotypes $g_1 = 2220$ and $g_2 = 0222$. The (partially symbolic) solution produced in this case (by a single collapse step for the first and the third row of H) is the following:

$$\begin{pmatrix} 0 & x_2 & x_3 & 0 \\ 1 & \bar{x}_2 & \bar{x}_3 & 0 \\ 0 & x_2 & x_3 & 0 \\ 0 & \bar{x}_2 & \bar{x}_3 & 1 \end{pmatrix}$$

In this case no variable-free haplotype is available and the residual variables have to be randomly assigned. There are $2^{(2-1)} = 2$ possible distinct non-symbolic solutions, namely:

$$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{pmatrix} \text{ and } \begin{pmatrix} 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}$$

$\square$

4. Experimental Results

CollHaps, the program based on the above illustrated algorithm, was tested on a large number of datasets (built from real and software-generated haplotypes) to study its performance in terms of parsimony, effectiveness, efficiency and scalability.

4.1. Assessing parsimony compliance

Since the algorithm is based on the maximum parsimony principle, a first set of experiments was devoted to assess if CollHaps solutions are actually maximally parsimonious, or at least close to the maximally parsimonious ones. To this aim we used some families of datasets obtained by randomly combining a set of haplotypes generated by the well-known Hudson's software "ms" and with variable levels of recombination. The sizes of the datasets used in the experiments were chosen according to the current thresholds of applicability for ILP-based techniques [4]. To test parsimony compliance, we used the optimal solutions and lower bounds (whenever an optimal solution could not be reached in reasonable processing time) provided by a novel integer linear programming approach [25].

The parent haplotypes were generated using Hudson's software "ms" [26] with recombination level r values of 0, 4, 16 and 40, as in [15, 16], and 100, as in [14]. For each r value 20 datasets were generated, each with 50 genotypes and 30 sites. As mentioned above, these values constitute the limit of applicability of current exact approaches based on ILP formulations.

The results are shown in Table 1. All instances with recombination levels ranging from 0 to 16 were solved by the ILP formulation and for 58 out of 60 instances CollHaps could find a maximally parsimonious

The CollHaps algorithm

```
for i = 0 to maxNoAttempts
    Perform a heuristically determined sequence of
        collapse steps (Sec. 3.4)
    Reduce the set of haplotypes used (Sec. 3.5)
    Update the best local and (if necessary) overall
        solutions
    repeat
        Execute a precollapse based on the previous
            best local solution (Sec. 3.6)
        Perform a heuristically determined sequence
            of collapse steps (Sec. 3.4)
        Reduce the set of haplotypes used (Sec. 3.5)
        If necessary update the best local and overall
            solutions
    until no improvements have been obtained in the
        last maxNoLocal searches
endfor
If necessary remove the residual variables (Sec. 3.7)
```

solution in less than 2 seconds. In only two cases CollHaps found non-optimal solutions using 20 and 16 haplotypes instead of the optimal respectively 19 and 15.

Table 1: Performance on ms-generated datasets ($n = 50$, $m = 30$)

Recombination level	Number of datasets optimally resolved by ILP	Number of datasets optimally resolved by CollHaps	Number of CollHaps solutions certainly non-optimal	CollHaps solutions improving the ILP U-B	CollHaps solutions equal to the ILP U-B
0	20	18	2	-	-
4	20	20	-	-	-
16	20	20	-	-	-
40	16	18	-	1	1
100	8	15	-	4	1

For $r = 40$ the ILP could find an exact solution in 16 cases and only provided a pair (Lower-Bound, Upper-Bound) in the remaining 4. CollHaps could find 18 optimal solutions, i.e. the 16 provided by the ILP plus 2 solutions using a number of haplotypes equal to the Lower-Bound (and hence optimal). In the remaining 2 cases the solutions provided by CollHaps were at least as good as the Upper-Bounds provided by the ILP.

For $r = 100$ the ILP could find an exact solution in 8 cases and only provided a pair (Lower-Bound, Upper-Bound) in the remaining 12. CollHaps could find 15 optimal solutions, i.e. the 8 provided by the ILP plus 7 solutions using a number of haplotypes equal to the Lower-Bound (and hence optimal). In the remaining 5 cases the solutions provided by CollHaps were at least as good as the Upper-Bounds provided by the ILP, in particular in 4 cases it could improve the Upper-Bound determined by the ILP. In all datasets CollHaps processing times ranged between 1 and 3 seconds.

4.2. Assessing accuracy: performance measures and haplotype inference programs

Determining good approximations of maximum parsimony solutions is certainly an interesting result by itself. However, our aim was also to investigate the relationships between maximum parsimony and correct haplotype reconstruction, in contexts that could not be studied in the past due to the size limitations imposed by exact ILP-based techniques. In this perspective, parsimony is not an aim, but used as a means of obtaining good haplotype reconstructions. Our experimental results seem to confirm the already mentioned conjecture that a "a program that adds heuristics to the Pure Parsimony criterion would produce results that are competitive with programs such as Phase" [4].

CollHaps performance in terms of error rates and speed was compared with four widely used programs for haplotype inference, namely: Hapinferx [7], kindly provided by Prof. Clark, and Haplotyper [9], Phase [10], Gerbil [22], available at the authors' websites. All programs were run with their default values and Haplotyper was run with the ROUND parameter set to 20, as suggested by the authors. The version 2.1 of the Phase software was used, where relevant improvements were introduced with respect to 1.x versions [27]. With the largest instances, where the long processing times of PHASE are a significant issue, we also tested the FastPhase [24] software, which is based on a slightly modified technique assuming some forms of clustering among haplotypes, achieving faster processing times although with slightly less accurate solutions.

When comparing the above mentioned programs several measures were considered: (genotype and switch) error rates, number of haplotypes, processing time and size constraints. The two most important ones are obviously the error rates, which will be defined and illustrated below. The number of haplotypes used by the solution was mainly considered to analyze the possible correlation between error rates and parsimony. Processing time along with explicit or implicit size constraints were used as indicators of the algorithm's scalability and practical applicability: even for medium-scale problem instances, some existing problems may require days (and even years) of processing time, or even crash due to memory allocation problems.

The *genotype error rate* is very commonly used to assess the accuracy of haplotype inference algorithms and defined as the percentage of incorrectly inferred genotypes. Although very easy to define and compute, this measure is not very accurate to assess how "close" an incorrect genotype solution is to the actual one: an incorrectly solved genotype is just an error, independently of the number of incorrect SNPs.

The *switch error rate* can be considered as a more accurate version of the previous measure and was used also in [28, 10]. The idea is to measure the "distance" of each proposed genotype solution from the actual one. Suppose for example $g = 000000 \oplus 111111 = 222222$. Any solution having exactly one '0' in one of the two haplotypes (e.g. $011111 \oplus 100000$, $101111 \oplus 010000$, etc.) has distance one from the actual solution, as it suffices to switch the single '0' with a '1' (and viceversa on the other haplotype) to obtain the actual solution. Similarly, solutions with two '0' (e.g. $001111 \oplus 110000$, $010111 \oplus 101000$, etc.) have distance 2. The worst case is that of solutions having three '0' in one of the haplotypes, requiring three switches to obtain the actual solution (note that solutions with four '0' are obviously at distance 2). More generally, given a genotype g_i with s_i heterozygous sites, the worst-case number of switches required to obtain the actual genotype solution is therefore

$$\left\lfloor \frac{s_i}{2} \right\rfloor$$

In the case of a HIP-instance comprising n genotypes, the worst-case total number of switches required to recover the original haplotypes is therefore expressed by

$$\sum_{i=1}^{n} \left\lfloor \frac{s_i}{2} \right\rfloor$$

The switch error rate is defined as the ratio between the total number of switches measured on the inferred solution and the worst-case number of switches expressed above.

Note that in general a better genotype error rate does not imply a better switch error rate and viceversa, as illustrated by the following example.

Let us consider a HIP-instance constituted by only two genotypes: 0222222 and 2222221, i.e. by the matrix G:

$$G = \begin{pmatrix} 0 & 2 & 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 2 & 2 & 1 \end{pmatrix}$$

and suppose the actual 2 pairs of haplotypes generating them are the following:

$$H_a^* = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

Let us consider two candidate solutions and the corresponding error rates:

$$H_1^* = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

$$H_2^* = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

In H_1^* both genotypes are incorrectly resolved and the genotype error rate is therefore the worst possible, i.e. 100%. However, the worst possible number of switches is $3+3=6$ and only 2 switches are sufficient to obtain H_a^* from H_1^*, hence the switch error rate for the first solution is only 2/6, i.e. 33%. In H_2^* the second genotype is correctly resolved, producing a genotype error rate of 1/2, i.e. 50%. However, three switches are required to obtain H_a^* from H_2^* and consequently the switch error rate is given by 3/6, i.e. 50%. In other terms the first solution is worse in terms of genotype error rate, although producing a better switch error rate.

In the following sections the results from three families of datasets will be analyzed: (1) the ACE dataset is a fairly "easy" dataset and will be considered to illustrate CollHaps' symbolic solutions and the fact that insufficiently large datasets produce unsatisfactory error rates, independently of the adopted haplotype inference program. (2) The CFTR-based datasets are examples of relatively "hard" (despite the relatively small number of genotypes and SNPs) instances and show a first example of how parsimony and good error rates are correlated when the instance size is sufficiently large. This is further confirmed by the third family of datasets, (3) the 5q31-based datasets, which are examples of those large instances that can be effectively and efficiently solved by using CollHaps. □

4.3. Angiotensin Converting Enzyme Dataset

Angiotensin Converting Enzyme (encoded by the gene DCP1, also known as ACE) catalyzes the conversion of angiotensin I to the physiologically active peptide angiotensin II, which controls fluid-electrolyte balance and systemic blood pressure. [29] completed the genomic sequencing of DCP1 from 11 individuals, identifying 78 varying sites in 22 chromosomes that were resolved into 13 distinct haplotypes. As in [14, 10] we only considered the 52 biallelic sites of these haplotypes.

We performed 20 runs for each program (changing the random seed each time, when possible). The results are summarized in Table 2. All algorithms correctly found maximum parsimony solutions using 13 haplotypes, but the genotype error rate is nevertheless quite high for all algorithms. The symbolic solution produced by CollHaps accounts for such poor performance: 3 out of the 11 haplotype pairs are partially symbolic (contain variables), corresponding to 2^{17} (see Proposition 3.7 above) distinct solutions, all using 13 haplotypes and hence parsimony-equivalent. Furthermore, each haplotype of these 3 pairs is incompatible with any other symbolic haplotype already in the initial haplotype matrix, thus requiring

Table 2: Performance comparison on ACE dataset ($n = 11$, $m = 52$)

	Number of used haplotypes	Average genotype error rate	Average switch error rate	Average proc. time (in sec.)
Gerbil	13	.181	**.042**	7.8
Hapinferx	13	.272	.073	**0.1**
Haplotyper	13	**.159**	.057	6.1
Phase	13	.181	.063	116.2
CollHaps	13	.272	.053	0.2

The results for the best-performing algorithms in each column are in bold. CollHaps produced a partially symbolic solution corresponding to 2^{17} distinct parsimony-equivalent solutions.

the introduction of 6 distinct and unique haplotypes. This stresses the importance of sufficiently large samples to obtain good error rates.

4.4. Cystic Fibrosis Transmembrane-Conductance Regulator (CFTR) Gene dataset

Cystic fibrosis is one of the most common autosomal recessive diseases in Caucasian populations, occurring approximately once in every 2000 live births. Nearly 70% of mutations in cystic fibrosis patients have been shown to correspond to a specific 3-bp deletion in the CFTR gene on chromosome 7 (region q31) by [30], who collected data from affected and healthy individuals on 23 SNPs in a 1.8 Mb candidate region.

As in [9, 10] we considered a subset of 57 haplotypes (of which 29 are distinct) with no missing data from the 94 experimentally identified disease haplotypes. In [9, 10] the haplotypes were combined to form datasets of size 28 and the high error rates obtained in the experiments were attributed to the low number of genotypes with respect to the number of distinct haplotypes. In our experiments 8 sizes (ranging from 28 to 56 genotypes) were therefore considered and for each size, 20 distinct datasets were generated.

Table 3: Performance comparison on CFTR data ($m = 23$)

n	28	32	36	40	44	48	52	56
				average genotype error rate				
Gerbil	.579	.550	.531	.504	.472	.450	.454	.442
Hapinferx	.941	.892	.836	.795	.743	.703	.681	.677
Haplotyper	.298	.275	.221	.138	.118	.098	.082	.079
Phase	**.264**	**.195**	**.153**	**.114**	.091	.086	.079	.069
CollHaps	.359	.261	.189	.126	**.083**	**.063**	**.054**	**.037**
				average switch error rate				
Gerbil	.345	.316	.307	.300	.280	.264	.265	.255
Hapinferx	.930	.840	.770	.725	.660	.618	.593	.586
Haplotyper	.185	.160	.134	.085	.071	.058	.049	.047
Phase	**.154**	**.114**	**.090**	**.066**	.054	.048	.045	.039
CollHaps	.219	.163	.117	.080	**.053**	**.041**	**.033**	**.022**
				average number of haplotypes used				
Gerbil	39.00	40.95	44.50	47.35	49.30	50.35	52.05	53.20
Hapinferx	52.75	55.35	58.25	62.55	63.75	66.10	68.05	72.40
Haplotyper	26.50	27.45	27.85	28.25	28.55	28.55	28.70	28.85
Phase	28.55	28.60	28.75	28.85	28.90	29.05	29.10	29.16
CollHaps	26.45	27.10	27.70	28.15	28.50	28.55	28.55	28.75

The results for the best-performing algorithms in each column are in bold.

The results in terms of parsimony and error rates are illustrated in Table 3. For low n values, most haplotypes are "used" only once and the few sample genotypes are apparently insufficient for their correct inference. Programs that incorporate biological models, like Phase, outperform the other algorithms, but the error rates are nevertheless intolerably high for all techniques. This confirms what has already been noted by Fallin and Schork [31], i.e. that much of the overall error in haplotype inference is due to sampling error, rather than to haplotyping techniques.

The table shows that CollHaps is the "most parsimonious" program (but obviously in this context this is not necessarily a positive characteristic) and Phase has the best error rates for low n values. However, when the sample size increases, error rates progressively decrease for all algorithms and parsimony leads to better solutions: CollHaps outperforms the other algorithms for both the genotype and the switch error rate.

Gerbil and Haplotyper have similar processing times (all from 3 to 10 seconds), while Phase requires some minutes (from 4 to 10) to process a single instance. CollHaps solved all instances in less than 2 seconds. Hapinferx is very fast (instances are always processed in less than one second) but has unacceptable error rates. Gerbil's error rates are also particularly high; this may be related to the characteristics of these haplotypes and Gerbil's assumption of rare recombination events in blocks.

4.5. Chromosome 5q31 dataset

[32] studied a 500-kb region on human chromosome 5q31, which is implicated as containing a genetic risk factor for Crohn's disease. The original diploid data contain 129 pedigrees (mother, father and child) each genotyped at 103 SNPs. As in [22] we considered the 258 haplotypes from the children and used them to generate datasets of size (number of genotypes) 500, 600, 700, 800, 900 and 1000 (20 datasets were generated for each size). To produce more difficult input sets, where completely resolved genotypes (i.e. genotypes without heterozygous sites) are less likely, the duplicate haplotypes were eliminated and only 178 unique haplotypes were considered for random sampling and pairing.

Table 4: Performance comparison on 5q31 dataset ($m = 103$)

n	500	600	700	800	900	1000
			average genotype error rate			
Gerbil	.76030	.75783	.75807	.75425	.75406	.75445
Hapinferx	.63900	.62192	.57071	.57075	.53694	.52830
FastPhase	.37670	.36467	.35764	.35544	.34628	.34085
Phase	.00630	.00342	.00314	.00263	.00194	.00185
CollHaps	**.00430**	**.00083**	**.00021**	**.00013**	**.00000**	**.00000**
			average switch error rate			
Gerbil	.27657	.27503	.27325	.27180	.27380	.27470
Hapinferx	.29981	.29498	.21054	.20695	.16626	.16484
FastPhase	.09871	.09251	.09110	.09032	.08847	.08735
Phase	**.00150**	.00072	.00052	.00035	.00025	.00022
CollHaps	.00200	**.00042**	**.00011**	**.00004**	**.00000**	**.00000**
			average number of haplotypes used			
Gerbil	746.4	866.9	983.7	1093.6	1203.3	1307.9
Hapinferx	505.4	583.4	573.6	635.6	631.9	675.5
FastPhase	455.3	499.9	540.5	584.9	617.0	652.9
Phase	179.9	178.7	178.8	178.6	178.2	178.1
CollHaps	178.5	178.1	178.0	178.0	178.0	178.0

Haplotyper was unable to obtain any solutions.
The results for the best-performing algorithms in each column are in bold.

Haplotyper was unable to obtain any solutions due to specific input size limitations. Gerbil required processing times of 15 to 25 minutes, while Hapinferx required 5 minutes at most to process each problem instance. However, for both programs error rates were fairly high. Phase processing times ranged from

12 hours to one day. As a consequence, FastPhase output was also examined, obtaining times ranging from one to 2 hours, but error rates quite worse with respect to Phase. Finally, CollHaps processing times ranged between 7 and 15 minutes.

The table shows that Phase and CollHaps performances were clearly superior to the other programs and that parsimony is clearly related to accuracy when the sample size is sufficiently large (in terms of number of genotypes), like in this case. The poor performances of both Gerbil and FastPhase are probably related to the assumptions on population made by the two programs. Both Phase and CollHaps nearly always produced solutions with 178 haplotypes (which are very likely to be maximally parsimonious, as 178 is also the number of haplotypes used to generate the genotypes) and the correlation between parsimony and error rates is apparent. For instances with 600 genotypes or more CollHaps outperforms the other programs and could always determine the correct reconstruction for instances with 900 genotypes or more.

5. Conclusions and future work

In this paper we presented a new algorithm for the haplotype inference problem, based on the maximum parsimony principle and a generalized version of Clark's rule, named collapse rule. Some relevant properties of this rule were illustrated, particularly that any optimal solution can be obtained by a suitable sequence of collapse rule applications. Finally, the implementation of the proposed algorithm (CollHaps) was tested on several datasets built from real and software-generated haplotypes. The experiments show that CollHaps achieves good performance in terms of parsimony (number of haplotypes used), processing times and above all error rates: indeed, parsimony and correct reconstruction appear to be strictly correlated, when the datasets are sufficiently large. We are currently working on further developments of the software to cope with incomplete data and identify blocks in haplotype sequences. We are also planning to use the algorithm to determine good upper bounds in exact parsimony techniques, based on Integer Linear Programming formulations of the problem.

Acknowledgment

The authors wish to thank the authors of the programs used in the experiments for providing their software.

References

[1] J. C. e. a. Venter, "The sequence of the human genome," *Science*, vol. 291, pp. 1304–1351, 2001.

[2] L. Jin, P. A. Underhill, V. Doctor, R. W. Davis, P. Shen, L. L. Cavalli-Sforza, and P. J. Oefner, "Distribution of haplotypes from a chromosome 21 region distinguishes multiple prehistoric human migiration," *Proc Natl Acad Sci*, vol. 96, pp. 3796–3800, 1999.

[3] M. R. Hoehe, K. Köpke, B. Wendel, K. Rohde, C. Flachmeier, K. K. Kidd, W. H. Berrettini, and C. G. M, "Sequence variability and candidate gene analysis in complex disease: association of μ opioid receptor gene variation with substance dependence," *Hum Mol Genet*, vol. 9, pp. 2895–2908, 2000.

[4] D. Gusfield and S. H. Orzack, "Haplotype inference," in *Handbook of Computational Molecular Biology*, A. S, Ed. CRC Press, 2006, p. 18.

[5] B. V. Halldorsson, V. Bafna, N. Edwards, R. Lippert, S. Yooseph, and I. S, "A survey of computational methods for determining haplotypes," in *Proceedings of the DIMACS/RECOMB Satellite Workshop on Computational Methods for SNPs and Haplotype Inference*, 2002, pp. 26–47.

[6] G. Lancia, M. C. Pinotti, and R. Rizzi, "Haplotyping populations by pure parsimony: Complexity of exact and approximation algorithms," *INFORMS Journal on Computing*, vol. 16, pp. 348–359, 2004.

[7] S. Clark, "Inference of haplotypes from pcr-amplified samples of diploid populations," *Mol Biol Evol*, vol. 7, pp. 111–122, 1990.

[8] R. M. Adkins, "Comparison of the accuracy of methods of computational haplotype inference using a large empirical dataset," *BMC Genetics*, vol. 5:22, pp. –, 2004.

[9] T. Niu, Z. S. Qin, X. Xu, and J. S. Liu, "Bayesian haplotype inference for multiple linked single-nucleotide polymorphisms," *Am J Hum Gen*, vol. 70, pp. 157–169, 2002.

[10] M. Stephens and P. Donnelly, "A comparison of bayesian methods for haplotype reconstruction from population genotype data," *Am J Hum Gen*, vol. 73, pp. 1162–1169, 2003.

[11] P. Donnelly, "Comments made in a lecture given at the dimacs conference on computational methods for snps and haplotype inference," November 2002.

[12] E. Hubbell, "Finding a maximum parsimony solution to haplotype phase is np-hard," personal communication, 2000.

[13] R. Sharan, B. V. Halldorsson, and S. Istrail, "Islands of tractability for parsimony haplotyping," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 3, pp. 303–311, 2006.

[14] L. Wang and Y. Xu, "Haplotype inference by maximum parsimony," *Bioinformatics*, vol. 19, pp. 1773–1780, 2003.

[15] D. Gusfield, "Haplotype inference by pure parsimony," in *Proceedings of the 14th Annual Symposium on Combinatorial Pattern Matching (CPM 2003)*, 2003, pp. 144–155.

[16] D. G. Brown and I. M. Harrower, "Integer programming approaches to haplotype inference by pure parsimony," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 3, pp. 141–154, 2006.

[17] I. Lynce and J. Marques-Silva, "Sat in bioinformatics: Making the case with haplotype inference," in *Proceedings of the International Conference on Theory and Applications of Satisfiability Testing (SAT)*, 2006, pp. 283–296.

[18] ——, "Haplotype inference with boolean satisfiability," *International Journal on Artificial Intelligence Tools*, vol. 17, pp. 355–387, 2008.

[19] L. Excoffier and M. Slatkin, "Maximum-likelihood estimation of molecular haplotype frequencies in a diploid population," *Mol Bio Evol*, vol. 12, pp. 921–927, 1995.

[20] M. E. Hawley and K. K. Kidd, "Haplo: a program using the em algorithm to estimate the frequencies of multi-site haplotypes," *J Heredity*, vol. 86, pp. 409–411, 1995.

[21] J. C. Long, R. C. Williams, and M. Urbanek, "An e-m algorithm and testing strategy for multiple-locus haplotypes," *Am J Hum Gen*, vol. 56, pp. 799–810, 1995.

[22] G. Kimmel and R. Shamir, "Gerbil: Genotype resolution and block identification using likelihood," *Proc Natl Acad Sci*, vol. 102, pp. 158–162, 2005.

[23] R. R. Hudson, "Gene genalogies and the coalescent process," in *Oxford Surveys in Evolutionary Biology*, F. D and A. J, Eds. Oxford University Press, 2006, vol. 7, pp. 1–44.

[24] P. Scheet and M. Stephens, "A fast and flexible statistical model for large-scale population genotype data: applications to inferring missing genotypes and haplotypic phase," *Am J Hum Gen*, vol. 78, pp. 629–644, 2006.

[25] G. Lancia and P. Serafini, "A covering approach with column-generation for parsimony haplotyping," 2007, Technical Report 4-07 Dip. di Matematica e Informatica, University of Udine (also submitted for publication).

[26] R. R. Hudson, “Generating samples under a wright-fisher neutral model of genetic variation,” *Bioinformatics*, vol. 18, pp. 337–338, 2002.

[27] M. Stephens, N. Smith, and P. Donnelly, “A new statistical method for haplotype reconstruction from population data,” *Am J Hum Gen*, vol. 68, pp. 978–989, 2001.

[28] S. Lin, D. J. Cutler, E. Zwick, and A. Chakravarti, “Haplotype inference in random population samples,” *Am J Hum Gen*, vol. 71, pp. 1129–1137, 2002.

[29] M. J. Rieder, S. L. Taylor, A. G. Clark, and D. A. Nickerson, “Sequence variation in the human angiotensin converting enzyme,” *Nature Gen*, vol. 22, pp. 59–62, 1999.

[30] B. Kerem, J. M. Rommens, J. A. Buchanan, D. Marliewicz, T. K. Cox, A. Chakravarti, M. Buchwald, and L. Tsui, “Identification of the cystic fibrosis gene: Genetic analysis,” *Science*, vol. 245, pp. 1073–1080, 1989.

[31] D. Fallin and N. J. Schork, “Accuracy of haplotype frequency estimation for biallelic loci, via the expectation-maximization algorithm for unphased diploid genotype data,” *Am J Hum Genet*, vol. 67, pp. 947–959, 2000.

[32] M. J. Daly, J. D. Rioux, S. F. Schaffner, T. J. Hudson, and E. S. Lander, “High-resolution haplotype structure in the human genome,” *Nature Genetics*, vol. 29, pp. 229–232, 2001.