Stochastic local search for large-scale instances of the Haplotype Inference Problem by Parsimony

Luca Di Gaspero

DIEGM
University of Udine
via delle Scienze 208, I-33100, Udine, Italy

Andrea Roli

DEIS, Campus of Cesena
University of Bologna
via Venezia 52, I-47023, Cesena, Italy

Abstract

Haplotype Inference is a challenging problem in bioinformatics that consists in inferring the basic genetic constitution of diploid organisms on the basis of their genotype. This information allows researchers to perform association studies for the genetic variants involved in diseases and the individual responses to therapeutic agents. A notable approach to the problem is to encode it as a combinatorial problem (under certain hypotheses, such as the *pure parsimony* criterion) and to solve it using off-the-shelf combinatorial optimization techniques. The main methods applied to Haplotype Inference are either simple greedy heuristic or exact methods (Integer Linear Programming, Semidefinite Programming, SAT and pseudo-boolean encoding) that, at present, are adequate only for moderate size instances. In this paper, we present and discuss an approach based on the combination of local search metaheuristics and a reduction procedure based on an analysis of the problem structure. Some relevant design issues are first described, then a family of local search metaheuristics is defined to tackle the Haplotype Inference. Results on common Haplotype Inference benchmarks show that the approach achieves a good trade-off between solution quality and execution time.

Key words: Metaheuristics, haplotype inference, bioinformatics

Email addresses: `l.digaspero@uniud.it` (Luca Di Gaspero), `andrea.roli@unibo.it` (Andrea Roli).

1 Introduction

A fundamental tool of analysis to investigate the genetic variations in a population is based on *haplotype* data. A haplotype is a copy of a chromosome of a diploid organism (i.e., an organism that has two copies of each chromosome, one inherited from the father and one from the mother). The haplotype information allows to perform association studies for the genetic variants involved in diseases and the individual responses to therapeutic agents. The assessment of a full Haplotype Map of the human genome is indeed one of the current high priority tasks of human genomics [1].

Instead of dealing with complete DNA sequences, usually the researchers are focusing on *Single Nucleotide Polymorphisms* (SNPs), which are the most common mutations among haplotypes. A SNP is a single nucleotide site (allele) where exactly two (out of four) different nucleotides occur in a large percentage of the population.

The haplotype collection is not an easy task: in fact, due to technological limitations it is currently infeasible to directly collect haplotypes in an experimental way, but rather it is possible to collect *genotypes*, i.e., the conflation of a pair of haplotypes. Moreover, instruments can only identify whether the individual is *homozygous* (i.e., the alleles are the same) or *heterozygous* (i.e., the alleles are different) at a given site. Therefore, haplotypes have to be inferred from genotypes in order to reconstruct the detailed information and trace the precise structure of human populations. This process is called *Haplotype Inference* and the goal is to find a set of haplotype pairs so that all the genotypes are *resolved*.

The main approaches to solve the Haplotype Inference are either combinatorial or statistical methods. However, both of them, being of non-experimental nature, need some genetic model of haplotype evolution, which poses some hypotheses to constrain the possible inferences to the ones compatible with Nature. In the case of the combinatorial methods, which are the subject of the present work, a reasonable criterion is the *pure parsimony* approach [2], which searches for the smallest collection of distinct haplotypes that solves the Haplotype Inference problem. This criterion is consistent with current observations in natural populations for which the actual number of haplotypes is vastly smaller than the total number of possible haplotypes. Anyway, some other criteria can be used to assess the quality of the resolving haplotypes, such as entropy measures. In this paper, we will present an approach that makes it possible to easily accommodate different criteria in a single solver.

Current approaches for solving the problem include simple greedy heuristics [3] and exact methods such as Integer Linear Programming [2,4,5,6], Semidefi-

nite Programming [7,8], SAT models [9,10] and Pseudo-Boolean Optimization algorithms [11]. These approaches, however, at present seem not to be particularly adequate for very-large size instances.

To the best of our knowledge, the only attempt to employ metaheuristic techniques for the problem is a recently proposed Genetic Algorithm [12]. However, the cited paper does not report results on real size instances. Anyway, we believe that metaheuristic and hybrid approaches could provide better scalability than exact approaches. Moreover, metaheuristics can be very easily combined with problem specific heuristics and they can also be integrated with tree-based search techniques, thus providing a promising framework for hybrid systems in which a good trade-off between effectiveness and efficiency can be reached.

In this work we present and discuss a metaheuristic approach to tackle the Haplotype Inference problem by pure parsimony. We introduce the problem in Section 2. In Section 3 we propose an analysis of the problem structure. The outcome of the analysis is a reduction procedure that can be combined with the metaheuristic approach developed in Section 4 in order to improve the performance of local search. Experimental results are discusses in Section 5, along with a summary of the search space analysis aimed at explaining the behavior of our algorithms. In Section 6 we compare our technique against the state of the art for Haplotype Inference by parsimony. Finally, we discuss the results and outline future work in Section 7.

2 The Haplotype Inference problem

In the Haplotype Inference problem we deal with *genotypes*, that is, strings of length m that corresponds to a chromosome with m sites. Each value in the string belongs to the alphabet $\{0, 1, 2\}$. A position in the genotype is associated with a site of interest on the chromosome (e.g., a SNP) and it has value 0 (wild type) or 1 (mutant) if the corresponding chromosome site is a homozygous site (i.e., it has that state on both copies) or the value 2 if the chromosome site is heterozygous. A *haplotype* is a string of length m that corresponds to only one copy of the chromosome (in diploid organisms) and whose positions can assume the symbols 0 or 1.

Given a chromosome, we are interested in finding an unordered [1] pair of haplotypes that can explain the chromosome according to the following definition:

Definition 1 (Genotype resolution) *Given a chromosome g, we say that the pair $\langle h, k\rangle$* resolves *g, and we write $\langle h, k\rangle \triangleright g$ (or $g = h \oplus k$), if the following conditions hold (for $j = 1, \ldots, m$):*

$$g[j] = 0 \Rightarrow h[j] = 0 \wedge k[j] = 0 \tag{1a}$$
$$g[j] = 1 \Rightarrow h[j] = 1 \wedge k[j] = 1 \tag{1b}$$
$$g[j] = 2 \Rightarrow (h[j] = 0 \wedge k[j] = 1) \vee (h[j] = 1 \wedge k[j] = 0) \tag{1c}$$

We say that h is a resolvent *of g, and we write $h \trianglelefteq g$, if there exists a companion haplotype k such that $\langle h, k\rangle \triangleright g$. This notation can be extend to sets of haplotypes and we write $H = \{h_1, \ldots, h_l\} \trianglelefteq g$, meaning that $h_i \trianglelefteq g$ for all $i = 1, \ldots, l$.*

The operator $\oplus$ on a single site j can be defined accordingly:

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

and its extension to strings of length m is straightforward.

Conditions (1a) and (1b) require that both haplotypes must have the same value in all homozygous sites, while condition (1c) states that in heterozygous sites the haplotypes must have different values.

Observe that, according to the definition, for a single genotype string the haplotype values at a given site are predetermined in the case of homozygous sites, whereas there is a freedom to choose between two possibilities at heterozygous places. This means that for a genotype string with l heterozygous sites there are 2^{l-1} possible pairs of haplotypes that resolve it.

As an example, consider the genotype $g = (0212)$, then the possible pairs of haplotypes that resolve it are $\langle(0110), (0011)\rangle$ and $\langle(0010), (0111)\rangle$.

[1] In the problem there is no distinction between the maternal and paternal haplotypes.

After these preliminaries we can state the *Haplotype Inference* problem as follows:

Definition 2 (Haplotype Inference problem) *Given a population of n individuals, each of them represented by a genotype string g_i of length m we are interested in finding a set R of n pairs of (not necessarily distinct) haplotypes $R = \{\langle h_1, k_1\rangle, \ldots, \langle h_n, k_n\rangle\}$, so that $\langle h_i, k_i\rangle \triangleright g_i, i = 1, \ldots, n$. We call H the set of haplotypes used in the construction of R, i.e., $H = \{h_1, \ldots, h_n, k_1, \ldots, k_n\}$.*

Of course, from the mathematical point of view, there are many possibilities for constructing the set R since there is an exponential number of possible haplotypes for each genotype. However, on the basis of knowledge about the biological phenomenon, geneticists can provide some criteria to evaluate the quality of solution returned, thus a constrained optimization model of the problem can be defined. One well-known model of the Haplotype Inference problem is the already mentioned *pure parsimony* approach that consists in searching for a solution that minimizes the total number of distinct haplotypes used or, in other words, $|H|$, the cardinality of the set H. A trivial upper bound for $|H|$ is $2n$ in the case of all genotypes resolved by a pair of distinct haplotypes.

It has been shown that the Haplotype Inference problem under the pure parsimony hypothesis is APX-hard [5] and therefore NP-hard. In Figure 1 an example of an instance of Haplotype Inference under pure (or *maximum*) parsimony is depicted. The number of genotypes is 5, among which one is composed only of homozygous sites. Therefore, a set of 9 haplotypes, two per genotype except for the genotype with homozygous sites, could resolve them. Nevertheless, under the hypothesis of pure parsimony, it is possible to find a smaller set of haplotypes that provides a more informative picture to geneticists and biologists.

geneticists

$$
\begin{aligned}
(0221) &= (0001) \oplus (0111)\\
(1011) &= (1011) \oplus (1011)\\
(1022) &= (1000) \oplus (1011)\\
(2102) &= (0100) \oplus (1101)\\
(2211) &= (0111) \oplus (1011)
\end{aligned}
\qquad
\begin{aligned}
H = \{&(0001), (0100), (0111), (1000),\\
&(1011), (1101)\}
\end{aligned}
$$

Figure 1. A solution under pure parsimony of an instance of the Haplotype Inference problem.

It is possible to define a graph that expresses the compatibility between genotypes, so as to avoid unnecessary checks in the determination of the resolvents. Let us build the graph $\mathcal{G} = (G, E)$, in which the set of vertices coincides with the set of the genotypes; in the graph, a pair of genotypes g_1, g_2 are connected by an edge if they are *compatible*, i.e., one or more common haplotypes can resolve both of them. For example, the genotypes (2210) and (1220) are compatible, whereas genotypes (2210) and (1102) are not compatible. The formal definition of this property is as follows.

Definition 3 (Genotypes compatibility) *Let g_1 and g_2 be two genotypes, g_1 and g_2 are* compatible *if, for all $j = 1, \ldots, m$, the following conditions hold:*

$$g_1[j] = 0 \Rightarrow g_2[j] \in \{0, 2\} \tag{3a}$$
$$g_1[j] = 1 \Rightarrow g_2[j] \in \{1, 2\} \tag{3b}$$
$$g_1[j] = 2 \Rightarrow g_2[j] \in \{0, 1, 2\} \tag{3c}$$

The same concept can be expressed also between a genotype and a haplotype as in the following definition.

Definition 4 (Compatibility between genotypes and haplotypes) *Let g be a genotype and h a haplotype, g and h are* compatible *if, for all $j = 1, \ldots, m$, the following conditions hold:*

$$g[j] = 0 \Rightarrow h[j] = 0 \tag{4a}$$
$$g[j] = 1 \Rightarrow h[j] = 1 \tag{4b}$$
$$g[j] = 2 \Rightarrow h[j] \in \{0, 1\} \tag{4c}$$

We denote this relation with $h \mapsto g$, and we write $h[j] \mapsto g[j]$ when the conditions hold for the single SNP j. Moreover with an abuse of notation we indicate with $h \mapsto \{g_1, g_2, \ldots\}$ the set of all the genotypes that are compatible with haplotype h.

Observe that the set of compatible genotypes of a haplotype can contain only mutually compatible genotypes (i.e., they form a clique in the compatibility graph).

Another interesting observation is the following. Due to the resolution definition, when one of the two haplotypes composing the pair, say h, has been selected, then the other haplotype can be directly inferred from h and the genotype g thanks to the resolution conditions.

Proposition 1 (Haplotype complement) *Given a genotype g and a haplotype $h \mapsto g$, there exists a unique haplotype k such that $h \oplus k = g$. The haplotype k is called the* complement *of h with respect to g and is denoted with $k = g \ominus h$.*

Proof 1 *The existence and uniqueness of k is a direct consequence of Conditions (1a)–(1c).* □

3 An analysis of the problem structure

To the best of our knowledge, there have been no attempts to exploit structural properties of the problem which can be deduced from compatibility graphs, or other problem representations. In this section, we perform such an analysis and we present a *reduction procedure* that starts from a set of haplotypes in the complete representation and tries to reduce its cardinality by exploiting compatibility properties of the instance. Other heuristics based on graph representations of the problem will be subject of ongoing work, such as the use of graph density statistics and spectral graph analysis for guiding the choice of the most suitable parameter setting and solver.

3.1 Haplotype cardinality reduction

Let us illustrate this property with an example. Consider the following set of genotypes, which corresponds to the compatibility graph in Figure 2.

$g_1 : (2210212)$	$g_3 : (1212122)$	$g_5 : (1202201)$
$g_2 : (2112110)$	$g_4 : (1222122)$	

Now consider the set $H = \{a, b, c, d, e, f, p, q\}$ of haplotypes that resolves all the genotypes $g_1, \ldots, g_5$. The set consists of 8 distinct haplotypes; genotypes

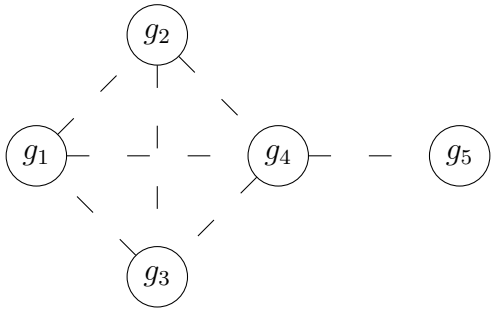

Figure 2. An example of compatibility graph.

currently resolved by a given haplotype are denoted by bold typeface in the resolvent list (i.e., we write $\mathbf{g}$ if $g \trianglerighteq h \in H$).

$$g_1 \triangleleft \begin{cases} a = (1110110) & \mapsto \{\mathbf{g_1}, \mathbf{g_2}, g_3, g_4\} \\ b = (0010010) & \mapsto \{\mathbf{g_1}\} \end{cases} \quad g_4 \triangleleft \begin{cases} f = (1101101) & \mapsto \{\mathbf{g_4}, \mathbf{g_5}\} \\ p = (1010101) & \mapsto \{g_3, \mathbf{g_4}\} \end{cases}$$

$$g_2 \triangleleft \begin{cases} c = (0111110) & \mapsto \{\mathbf{g_2}\} \\ a = (1110110) & \mapsto \{\mathbf{g_1}, \mathbf{g_2}, g_3, g_4\} \end{cases} \quad g_5 \triangleleft \begin{cases} f = (1101101) & \mapsto \{\mathbf{g_4}, \mathbf{g_5}\} \\ q = (1000001) & \mapsto \{\mathbf{g_5}\} \end{cases}$$

$$g_3 \triangleleft \begin{cases} d = (1111101) & \mapsto \{\mathbf{g_3}, g_4\} \\ e = (1010110) & \mapsto \{g_1, \mathbf{g_3}, g_4\} \end{cases}$$

Notice that the haplotype $a = (1110110)$, resolving g_1, is compatible also with genotypes g_2, g_3 and g_4 and, as a matter of fact, in the current resolution it resolves g_2 (but not g_3). This configuration is depicted in the graph of Figure 3 (dashed edges between genotypes represent genotype compatibility) in which the haplotypes are represented by square nodes, the compatibility between haplotypes and genotypes is represented by solid edges and a bold edge represents the current resolution of a genotype by a haplotype. We will call this graph an *extended (compatibility) graph*. The constraint on genotype resolution is mapped onto the extended graph by imposing that every genotype node must have (at least) two bold edges.

The goal of the reduction procedure is to try to decrease the number of distinct haplotypes, i.e., the number of square nodes while satisfying the resolution constraint. The intuition behind the procedure is that a possible way of reducing the haplotype number is to resolve a genotype by a haplotype that is compatible, but not currently resolving it, that is, changing an edge from solid to bold. Of course, this move must be followed by repairing moves in the graph so that the state is still feasible. These moves consist in adding one or more haplotypes and relinking some nodes.

From the situation described above, we can use the resolvent of g_1 to resolve g_3 or g_4, however the situation is different in the two cases. For example, if we use the resolvent a of g_1 to resolve g_3, then the situation will become:

$$g_3 \triangleleft \begin{cases} a = (1110110) & \mapsto \{\mathbf{g_1}, \mathbf{g_2}, \mathbf{g_3}, g_4\} \\ r = (1011101) & \mapsto \{\mathbf{g_3}, g_4\} \end{cases}$$

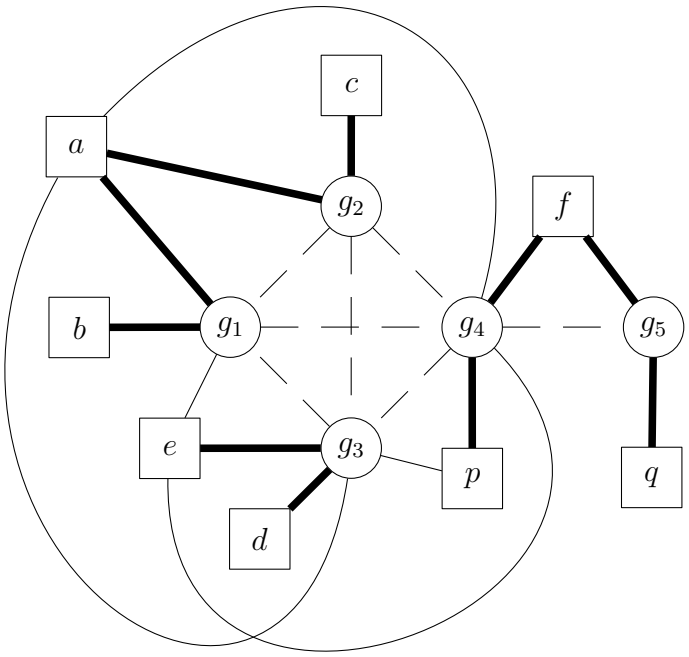

Figure 3. Haplotype resolution graph.

As an effect of the reduction, the total number of haplotypes employed in the solution has been decreased by 1, since now haplotypes d and e resolving g_3 are replaced by a, which was already a member of the haplotype set, and a new haplotype r (see Figure 4).

Instead, if we use the resolvent a to resolve g_4 we obtain the following situation (see Figure 5):

$$g_4 \triangleleft \begin{cases} a = (1110110) & \mapsto \{\mathbf{g_1}, \mathbf{g_2}, \mathbf{g_3}, \mathbf{g_4}\} \\ s = (1001101) & \mapsto \{\mathbf{g_4}\} \end{cases}$$

Unlike the previous reduction, in this case the number of haplotypes in the solution has not changed. The reason for this is that the haplotype f was already shared between the resolutions of g_4 and g_5, therefore the reduction operation removes one haplotype but it introduces also a new one, leaving the total number of haplotypes unchanged.

The situation of the example can be generalized by the following proposition.

Proposition 2 (Haplotype local reduction) *Given n genotypes $G = \{g_1, \ldots, g_n\}$ and the resolvent set $R = \{\langle h_1, b\rangle, \ldots, \langle h_n, k_n\rangle\}$, so that $\langle h_i, k_i\rangle \triangleright g_i$. Suppose there exist two genotypes $g, g' \in G$ such that:*

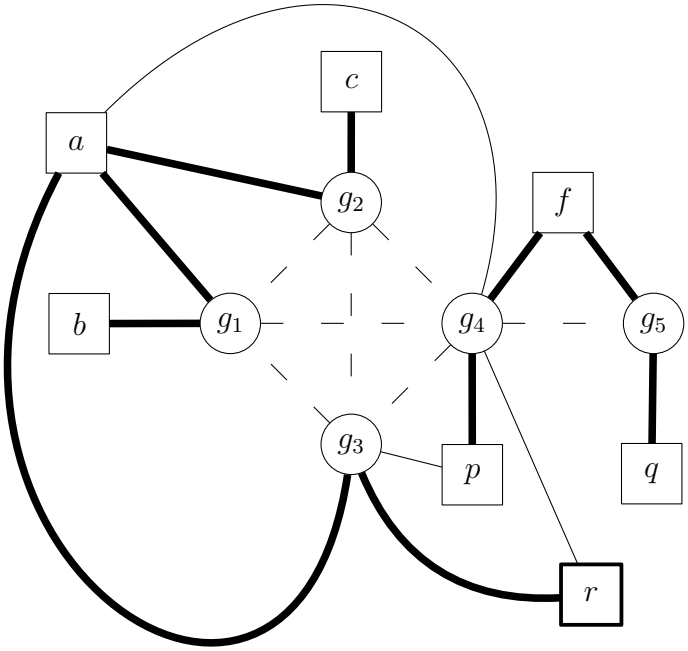

Figure 4. The haplotype resolution graph after the reduction of g_3.

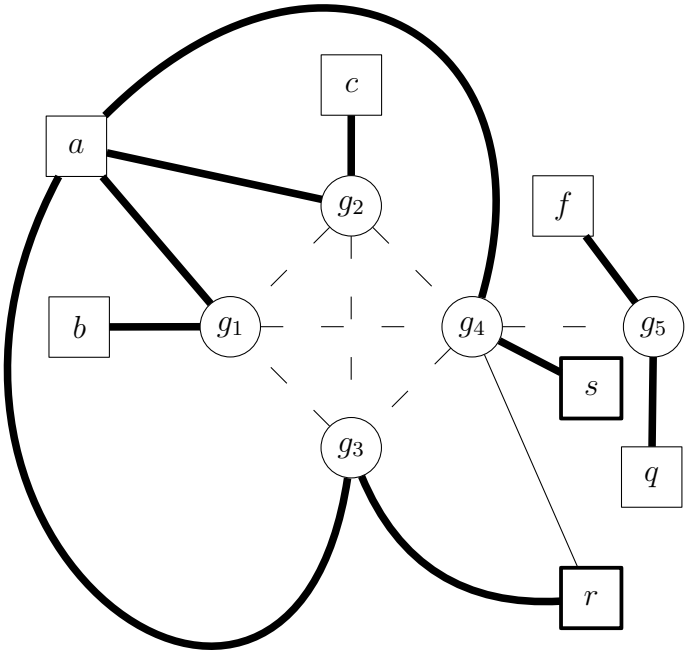

Figure 5. The haplotype resolution graph after the reduction of g_4.

$$g \lhd \begin{cases} h & \mapsto \{g, g', \ldots\} \\ k & \mapsto \{g, \ldots\} \end{cases}, \quad g' \lhd \begin{cases} h' & \mapsto \{g', \ldots\} \\ k' & \mapsto \{g', \ldots\} \end{cases} \tag{5}$$

and $h \neq h'$, $h \neq k'$, $h' \trianglelefteq A$, $k' \trianglelefteq B$.

The replacement of $\langle h', k' \rangle$ *with* $\langle h, g' \ominus h \rangle$ *in the resolution of* g' *is a correct resolution that employs a number of distinct haplotypes according to the following criteria:*

- *if* $|A| = 1$ *and* $|B| = 1$, *the new resolution uses* at most *one less distinct haplotype;*
- *if* $|A| > 1$ *and* $|B| = 1$ *(or symmetrically,* $|A| = 1$ *and* $|B| > 1$*), the new resolution uses* at most *the same number of distinct haplotypes;*
- *in the remaining case the new resolution uses* at most *one more distinct haplotype.*

Proof 2 *The proof of the proposition is straightforward. The resolution is obviously correct because* h *is compatible with* g' *and* $g' \ominus h$ *is the complement of* h *with respect to* g'.

Concerning the validity of the conditions on the cardinality, let us proceed by cases and first consider the situation in which $g' \ominus h$ *does not resolve any other genotype but* g'.

If $|A| = |B| = 1$, *then* h' *and* k' *are not shared with other genotype resolutions so they will not appear in the set* H *after the replacement, therefore since in the new resolution* h *is shared between* g *and* g' *the cardinality of* H *is decreased by one.*

Conversely, if one of the sets $|A|$ *or* $|B|$ *consists of more than a genotype and the other set of just one genotype, there is no guarantee of obtaining an improvement from the replacement. Indeed, since one of the two haplotypes is already shared with another genotype there is just a replacement of the shared haplotype with another one in the set* H.

Finally, when $|A| > 1$ *and* $|B| > 1$ *both* h' *and* k' *are shared with other genotypes therefore the replacement introduces the new haplotype* $g' \ominus h$ *in the set* H.

Moving to the situation in which $g' \ominus h$ *resolves also other genotypes, the same considerations apply; additionally, given that* $g' \ominus h$ *is already present in* H, *the number of distinct haplotypes employed in the resolution is decreased by one. For this reason the estimation of the changes of* $|H|$ *is conservative.* □

4 Local Search techniques for Haplotype Inference

The design process of metaheuristics involves first the definition of local search model and search strategy. In the following we detail our design and implementation choices for a family of stochastic local search techniques to tackle the Haplotype Inference. For a discussion on alternative metaheuristic approaches see [13].

The local search model is defined by specifying three entities, namely the *search space*, the *cost function* and the *neighborhood relation*.

4.1 Search space

As the search space for the Haplotype Inference problem we could either adopt a *complete* or *incomplete* representation as explained below.

Complete representation: This search space is based on a straightforward encoding of the problem. In this representation we consider, for each genotype g, the pair of haplotypes $\langle h, k\rangle$ that resolves it. Therefore in this representation all the genotypes are fully resolved at each state by construction. The search space is therefore the collection of sets R defined as in the problem statement.

Notice that, thanks to complementarity w.r.t. a genotype just one haplotype can be selected to represent a pair of resolvents. Moreover, since the pair is unordered, the haplotype pair $\langle h, k\rangle$ is equal to $\langle k, h\rangle$, therefore we can break this symmetry by selecting the haplotype h as the representative for the state if $h \prec k$, where the symbol $\prec$ indicates the lexicographic precedence of the two strings, otherwise we select k as the representative.

Incomplete representation: In the incomplete representation, instead, we deal with sets of haplotypes that are the elements of the set H. An element $h \in H$ is a *potential resolvent* of a genotype g, and it actually resolves g only with its companion k, so that $\langle h, k\rangle \triangleright g$, also belongs to H. As a consequence, in this representation not necessarily all the genotypes have a resolvent and the search space is the powerset $\mathcal{P}(\bigcup_{i=1}^{n} A_i)$, where $A_i = \{h | \exists k \langle h, k\rangle \triangleright g_i\}$ is the set of all potential resolvents of a genotype g_i of the problem. This formulation is an incomplete representation of the problem, since the genotype resolution is only potential and a solution of the problem must be constructed on the basis of the haplotypes selected in a given state. This representation shares some analogies with the classical representation used when local search is applied to constraint satisfaction problems [14]. Indeed, in that case the state is usually represented as a complete assignment that can be infeasible and the cost function accounts for possible violations.

The difference with Haplotype Inference is that, in this case, infeasibility has to be evaluated as a function of genotypes not resolved by the current tentative solution. To the best of our knowledge, this representation has not yet been used to tackle the Haplotype Inference problem and the design and implementation of metaheuristics using it is subject of ongoing work.

The complete representation has the advantage of making it possible to design anytime algorithms, since the search can be interrupted any moment and return a feasible solution, i.e., a set of haplotypes (not necessarily minimal) that resolve the given genotypes. Moreover, it is in general more adequate for hybridization with constructive methods since it maintains the feasibility (i.e., resolution of all genotypes) at every state of the search. For example it allows to employ local search in a GRASP-like manner [15] or to hybridize it with more informed strategies making use of problem-specific knowledge, as usually done with metaheuristics [16]. For these reasons, we opted for this search space representation.

4.2 Cost function

For the cost function, we identify different components related either to optimality or to heuristic measures.

A natural component is the objective function of the original problem, that is the cardinality $|H|$ of the set of haplotypes employed in the resolution.

$$f_1 = |H| \tag{6}$$

Moreover, we also might want to include some heuristic measure related to the potential quality of the solution. To this respect, a possible measure could be the number of incompatible sites between each genotype/haplotype pair, expressed by the following formula:

$$f_2 = \sum_{h \in H} \sum_{g \in G} \sum_{j=1}^{m} 1 - \chi(h[j] \mapsto g[j]) \tag{7}$$

In the formula, χ denotes the truth indicator function, whose value is 1 when the proposition in parentheses is true and 0 otherwise.

The cost function F is then the weighted sum of the two components:

$$F = \alpha_1 f_1 + \alpha_2 f_2 \tag{8}$$

in which the weights α_1 and α_2 must be chosen for the problem at hand to reflect the trade-offs among the different components. In our experimentation we adopt the values $\alpha_1 = \alpha_2 = 1$.

4.3 Local search strategies

We designed a family of local search strategies, namely *Best improvement*, *Stochastic first improvement*, *Simulated annealing*, and *Tabu search*. The techniques are instances of the general strategies described in [16]. All of them start with a set of haplotypes of cardinality $2n$, where n is the number of genotypes, and they explore the search space by iteratively modifying pairs of resolving haplotypes trying to reduce the number of distinct ones. Best improvement and Stochastic first improvement traverse the search space by moving from a state to a neighboring one with a lower cost function value, by choosing the best and first neighbor respectively. Simulated annealing moves also to worse states than the current one, on the basis of a probabilistic choice function. Finally, Tabu search behaves in principle like Best improvement but restricts the neighborhood by forbidding recently performed moves.

Local moves are defined upon a Hamming neighborhood function. A good trade-off between exploration and execution time is the 1-Hamming distance neighborhood w.r.t. each haplotype in the current solution. The complete exploration of such a neighborhood has a time complexity bounded from above by $O(nk)$, where k is the number of haplotypes and n the number of sites per haplotype. In practice, the time complexity can be further reduced by restricting the number of neighbors to heterozygous sites and haplotypes resolving non isolated genotypes. This kind of move can be thought as a *flip*, performed at a given position in a pair [2] of haplotypes resolving a given genotype.

The drawback of this choice relies in its local character of exploring neighboring solutions, that might not help search escape from areas in which it has been attracted. Indeed, in our experiments Tabu search strikingly outperformed the other local search algorithms we tested, being equipped with an effective diversification mechanism.

5 Experimental results

We developed our solvers with EasyLocal++ [17], a framework for the development of local search algorithms. The algorithms have been implemented

[2] Two haplotypes are involved in the move, since we adopt the complete representation and we must guarantee the pair of resolvents for each genotype.

Table 1
A summary of the main characteristics of the benchmarks.

Benchmark set	**N. of instances**	**N. of genotypes**	**N. of sites**
Harrower uniform	200	10÷100	30÷50
Harrower non-uniform	90	10÷100	30÷50
Harrower hapmap	24	5÷68	30÷75
Marchini SU1	100	90	179
Marchini SU2	100	90	171
Marchini SU3	100	90	187
Marchini SU-100kb	29	90	18

in C++ and compiled with *gcc 3.2.2* and run on a Intel Xeon CPU 2.80GHz machine with SUSE Linux 2.4.21-278-smp. Each algorithm was run on every instance one time and we allotted 300 seconds for each execution of the algorithms. Since Tabu search (TS) showed superior performance over the other local search algorithms, we only discuss results of this technique.

Our Tabu search implementation considers as tabu all the moves that insist on a pair of haplotypes that recently changed. We also experimented with a more relaxed criterion that prohibits only to apply the flip on the same haplotypes and in the same position of a recent move, however we realized that this criterion had an insufficient impact in the ability to diversify the search. The tabu list scheme adopted is a *dynamic* one, that is for each move performed we consider it as prohibited for a number of iterations that randomly varies between two values k_{min} and k_{max}. The values of these parameters were chosen according to the results of an exploratory analysis based on the F-Race method [18], and were set to $k_{min} = 10$, $k_{max} = 20$. These settings have shown to be quite robust across the variety of instances tested.

The benchmark instances are composed of two parts. The first one, composed of the sets Harrower uniform, Harrower non-uniform and Harrower hapmap, is the benchmark used in [6]. The second part of the instances, namely Marchini SU1, Marchini SU2, Marchini SU3 and Marchini SU-100kb, were taken from the website `http://www.stats.ox.ac.uk/~marchini/phaseoff.html`. The main characteristics of the instance sets are summarized in Table 1.

The cases of interest for our analysis are the local search run with or without the initial reduction procedure. The plots in Figures 6 and 7 show a comparison of the two approaches; a point (x, y) in the plot represents the number of haplotypes in the best solution returned by TS without and with reduction, respectively. A point below the line means that the solution returned by the algorithm corresponding to the y-axis is better than the one returned by the

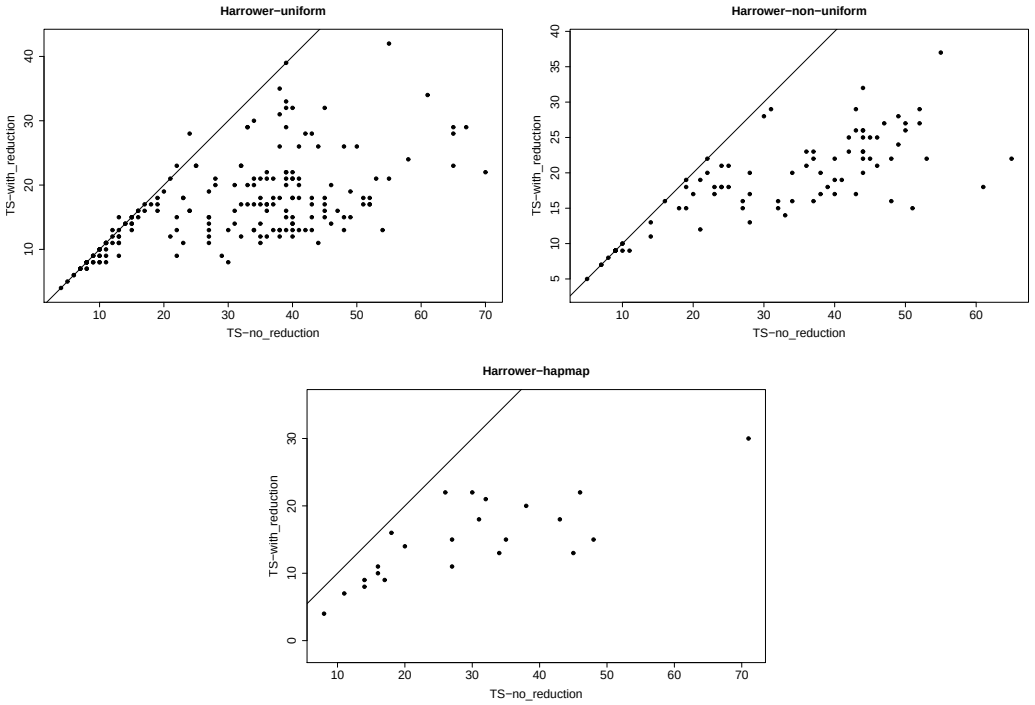

Figure 6. Comparison between TS without and with reduction procedure on Harrower's instances (in terms of number of haplotypes in the best solution).

algorithm associated to the x-axis.

The algorithm that incorporates the initial graph reduction clearly outperforms the one without graph reduction. This behavior shows that the exploitation of graph structure is particularly effective.

5.1 Search space analysis

The behavior of the local search algorithms we developed can be explained by analyzing the properties of the search space. Among the main search space parameters are the number and density of local and global minima and the basins of attraction of minima.

Informally, the basin of attraction of a state s is the set of initial states that make search reach s. Basins of attraction of minima represent a finer piece of information than minima density, because basins of attraction also take into account possible anisotropies of the search space. By studying the basins of attraction (BOA) of local and global minima, it is possible to estimate the probability of reaching a global minimum [19,20].

More formally, given a deterministic algorithm, such as Best improvement, the basin of attraction $\mathcal{B}(\overline{s})$ of a minimum $\overline{s}$, is defined as the set of states

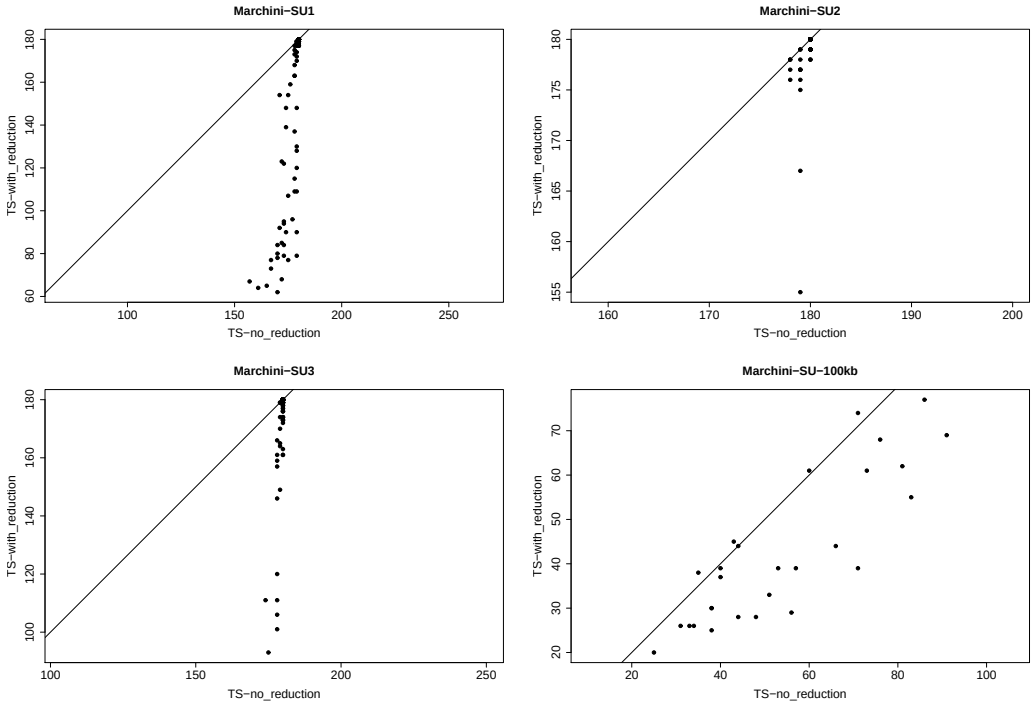

Figure 7. Comparison between TS without and with reduction procedure on Marchini's instances (in terms of number of haplotypes in the best solution).

that, taken as initial states, give origin to trajectories that ends at point $\overline{s}$. The cardinality of $\mathcal{B}(\overline{s})$ represents its size. The quantity $rBOA(\overline{s})$, defined as the ratio between the size of $\mathcal{B}(\overline{s})$ and the search space size, is an estimation of the reachability of state $\overline{s}$.

Given the set S^* of the global optima, the union of the BOA of global optima $I^* = \bigcup_{s \in S^*} \mathcal{B}(s)$ represents the set of desirable initial states of the search. Indeed, a search starting from $s \in I^*$ will eventually find an optimal solution. Since it is usually not possible to construct an initial solution that is guaranteed to be in I^*, the ratio $rGBOA = |I^*|/|S|$ can be taken as an indicator of the probability to find an optimal solution. On the extreme case, if we start from a random solution, the probability to find a global optimum is exactly $|I^*|/|S|$. Therefore, the higher this ratio, the higher the probability of success of the algorithm. TS incorporates stochastic decision mechanisms and moreover it is also able to escape from local minima by using the tabu criteria, therefore the estimation of basins of attraction size related to Best improvement provides a lower bound on the probability of reaching the global optimum when using TS.

We estimated the basins of attraction (w.r.t. Best improvement) of minima in the search space of some representative instances by uniformly sampling the search space. We observed two different scenarios: one in which many different very small basins exist, and, in some instances, most of them lead to attractors

Table 2
Estimation of basins of attraction of minima in the search space some representative instances from Harrower's benchmarks. Results are taken from up to 5000 uniform samples.

Instance	**Sol. value**	**rGBOA**
Harrower uniform 30_50.00	54	0.001
	55	0.002
	56	0.005
	57	0.018
	58	0.038
	59	0.075
	60	0.145
	61	0.222
	62	0.218
	63	0.181
	64	0.094
Harrower non-uniform 30_50.00	52	0.001
	54	0.004
	55	0.008
	56	0.020
	57	0.045
	58	0.091
	59	0.149
	60	0.215
	61	0.218
	62	0.153
	63	0.082
	64	0.016

at the same cost value (Harrower's benchmarks and Marchini SU-100kb); and the other scenario, in which no sample could find a state in a global minimum attraction basin (Marchini SU1÷SU3). Tables 2 and 3 report some statistics of representative cases of this analysis. Each line in the table reports a given solution value and its corresponding *rGBOA*, i.e., the estimated fraction of initial states which lead to that cost value.

The outcome of this analysis explains why TS outperforms the other local search algorithms. Indeed, in the first scenario, *rGBOA* is very small, hence a simple Best improvement strategy has a very low probability of finding an optimal or a near-optimal solution. On the contrary, TS can effectively traverse such a landscape and reach a (near-)optimal solution. In the second scenario, the situation is even more dramatic because the frequency of success is zero, therefore TS is definitely preferable over a simple Best improvement.

Table 3
Estimation of basins of attraction of minima in the search space some representative instances from Marchini's benchmarks. Results are taken from up to 5000 uniform samples.

Instance	**Sol. value**	**rGBOA**
Marchini SU1 genos.haps.1	170	1.0
Marchini SU2 genos.haps.1	180	1.0
Marchini SU3 genos.haps.1	180	1.0
Marchini SU-100kb genos.haps.1	105	0.009
	106	0.013
	107	0.009
	108	0.017
	109	0.035
	110	0.009
	111	0.061
	112	0.061
	113	0.035
	114	0.078
	115	0.082
	116	0.113
	117	0.069
	118	0.065
	119	0.108
	120	0.065
	121	0.043
	122	0.048
	123	0.048
	124	0.013
	125	0.009
	126	0.004
	127	0.009

6 Comparison with the state of the art

In order to estimate the quality of solutions produced by TS, we need to compute the optimal solution of the benchmark instances. We tackled the instances with *rpoly* [11], a state-of-the-art exact solver for the Haplotype Inference. We run the solver on the same benchmark instances and on the same machine. We allotted *rpoly* 24 hours of computation for each instance. The instances of the set Harrower uniform, Harrower non-uniform, Harrower hapmap, Marchini SU1 and Marchini SU2 were completely solved. From Marchini SU3 and Marchini SU-100kb only a portion of the instances were solved. Overall, most of the instances could be solved with a runtime higher than 12 hours. A summary of the fraction of solved instances is reported in Table 4.

Table 4
Fraction of instances solved by *rpoly* from each benchmark.

Benchmark set	Fraction of solved instances
Harrower uniform	200/200
Harrower non-uniform	90/90
Harrower hapmap	24/24
Marchini SU1	20/100
Marchini SU2	100/100
Marchini SU3	89/100
Marchini SU-100kb	23/29

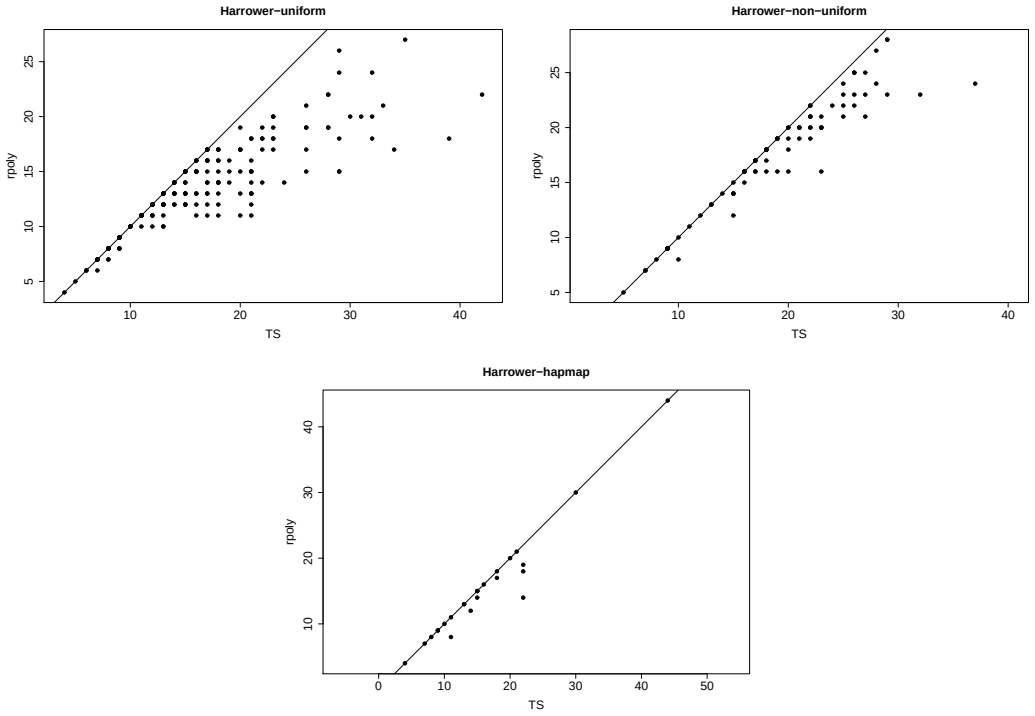

Figure 8. Comparison between TS and rpoly (on the instances solved by rpoly) in terms of number of haplotypes in the best solution.

In Figures 8 and 9 the plots showing the comparison between TS and *rpoly* on the solved instances are drawn.

We can observe that the solution quality achieved by TS (with reduction) approximates the optimal one returned by *rpoly* on some benchmarks, namely Harrower sets and Marchini SU-100kb, whilst the performance on Marchini SU2 is considerably inferior. The performance on benchmarks Marchini SU1 and Marchini SU3 is inferior, but it has to be taken into account that TS returned a feasible solution to all the instances of the sets, whilst *rpoly* solved

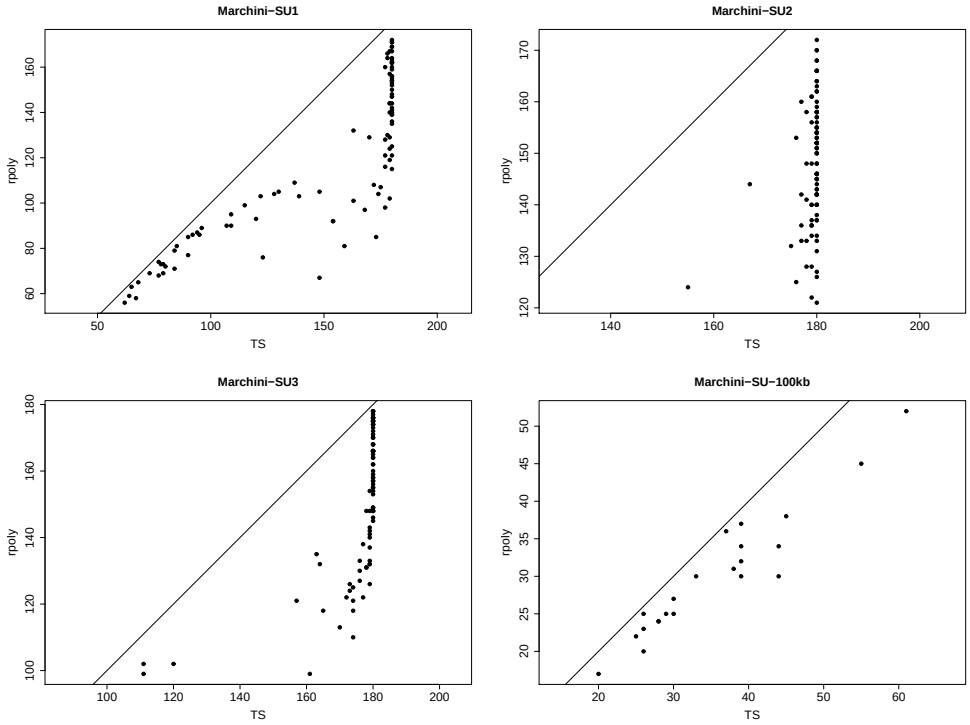

Figure 9. Comparison between TS and rpoly (on the instances solved by rpoly) in terms of number of haplotypes in the best solution.

only a fraction of the instances of Marchini SU3. We also observe that our approach scales very smoothly. Conversely, in instances with a high number of individuals and sites, the complete approach has not a good performance in terms of running times and it even fails to find a solution in 24 hours, whereas TS is able to find always a solution (even though its optimality cannot be guaranteed) within 300 seconds.

These results enlighten the complementarity of the two approaches: the algorithm that also returns the proof of optimality is definitely preferable over the incomplete one when the execution time allotted can be large, while we can resort to the approximate algorithm to have a feasible and (hopefully) near-optimal solution in very short time. Therefore, a possible powerful combination of the two techniques can be an hybrid algorithm that interleaves the execution of *rpoly* and local search, using the solution returned by latter as an upper bound for the former.

The local search approach discussed in this work has also the advantage of enabling the developer to easily specify different objective functions or also a weighted combination of objectives. This characteristic can be very useful to explore different solutions to the Haplotype Inference, making it possible for biologists to compare different candidate solutions to the real problem. In this respect, we are currently experimenting with several quality measures.

7 Discussion and future work

We have presented a metaheuristic approach to tackle the Haplotype Inference problem by pure parsimony and, in particular, we showed and discuss results of Tabu search. The performance of local search can be enhanced by introducing a reduction procedure that exploits features of the compatibility graph. Results are encouraging, although on some benchmarks our algorithm is dominated by the pseudo-boolean optimization solver *rpoly*. Nevertheless, it has to be considered that our algorithm quickly finds feasible solutions to all the instances and it can be taken as a starting point to design hybrid efficient algorithms to find near-optimal solutions.

We are currently working on improvements to our stochastic local search and testing our approach on Haplotype Inference by parsimony and other quality measures, such as entropy. Moreover, hybrid metaheuristics, combining local search and constructive procedures exploiting instance structure, are subject of ongoing work [13].

Acknowledgments

We thank Inês Lynce and Ana Sofia Graça for kindly providing us their instances and solvers, and we also thank Ian M. Harrower for sending us his datasets. We are grateful to three anonymous referees whose insightful comments allowed us to improve the paper.

References

[1] The International HapMap Consortium, The international HapMap project, Nature 426 (2003) 789–796.

[2] D. Gusfield, Haplotype inference by pure parsimony., in: R. A. Baeza-Yates, E. Chávez, M. Crochemore (Eds.), Combinatorial Pattern Matching (CPM 2003), Proceedings of the 14th Annual Symposium, Vol. 2676 of Lecture Notes in Computer Science, Springer-Verlag, Berlin-Heidelberg, Germany, 2003, pp. 144–155.

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

[4] B. V. Halldórsson, V. Bafna, N. Edwards, R. Lippert, S. Yooseph, S. Istrail, A survey of computational methods for determining haplotypes., in: S. Istrail, M. S. Waterman, A. G. Clark (Eds.), Computational Methods for SNPs and

Haplotype Inference, Vol. 2983 of Lecture Notes in Computer Science, Springer, 2002, pp. 26–47.

[5] G. Lancia, M. C. Pinotti, R. Rizzi, Haplotyping populations by pure parsimony: Complexity of exact and approximation algorithms., INFORMS Journal on Computing 16 (4) (2004) 348–359.

[6] D. G. Brown, I. M. Harrower, Integer programming approaches to haplotype inference by pure parsimony., IEEE/ACM Transactions on Computational Biology and Bioinformatics 3 (2) (2006) 141–154.

[7] K. Kalpakis, P. Namjoshi, Haplotype phasing using semidefinite programming., in: BIBE, IEEE Computer Society, 2005, pp. 145–152.

[8] Y.-T. Huang, K.-M. Chao, T. Chen, An approximation algorithm for haplotype inference by maximum parsimony., in: H. Haddad, L. M. Liebrock, A. Omicini, R. L. Wainwright (Eds.), Proceedings of the 2005 ACM Symposium on Applied Computing (SAC 2005), ACM, 2005, pp. 146–150.

[9] I. Lynce, J. Marques-Silva, SAT in bioinformatics: Making the case with haplotype inference., in: A. Biere, C. P. Gomes (Eds.), SAT, Vol. 4121 of Lecture Notes in Computer Science, Springer-Verlag, Berlin-Heidelberg, Germany, 2006, pp. 136–141.

[10] I. Lynce, J. Marques-Silva, Efficient haplotype inference with boolean satisfiability., in: Proceedings of the 21st National Conference on Artificial Intelligence and the Eighteenth Innovative Applications of Artificial Intelligence Conference, AAAI Press, Menlo Park, CA, USA, 2006.

[11] A. Graça, J. Marques-Silva, I. Lynce, A. L. Oliveira, Efficient haplotype inference with pseudo-boolean optimization, in: H. Anai, K. Horimoto, T. Kutsia (Eds.), Algebraic Biology, Second International Conference, AB 2007, Castle of Hagenberg, Austria, July 2-4, 2007, Proceedings, Vol. 4545 of Lecture Notes in Computer Science, Springer-Verlag, Berlin-Heidelberg, Germany, 2007, pp. 125–139.

[12] R.-S. Wang, X.-S. Zhang, L. Sheng, Haplotype inference by pure parsimony via genetic algorithm, in: X.-S. Zhang, D.-G. Liu, L.-Y. Wu (Eds.), Operations Research and Its Applications: the Fifth International Symposium (ISORA'05), Tibet, China, August 8–13, Vol. 5 of Lecture Notes in Operations Research, Beijing World Publishing Corporation, Beijing, People Republic of China, 2005, pp. 308–318.

[13] L. Di Gaspero, A. Roli, A preliminary analysis on metaheuristics methods applied to the haplotype inference problem, Tech. Rep. DEIS-LIA-07-006, University of Bologna (Italy), lIA Series no. 84 (2007).

[14] H. H. Hoos, T. Stützle, Stochastic Local Search Foundations and Applications, Morgan Kaufmann Publishers, San Francisco, CA (USA), 2005.

[15] M. G. C. Resende, C. C. Ribeiro, Greedy randomized adaptive search procedures, in: F. Glover, G. A. Kocheberger (Eds.), Handbook of

Metaheuristics, Vol. 57 of International Series in Operations Research & Management Science, Kluwer Academic Publishers, Norwell, MA, USA, 2003, pp. 219–249.

[16] C. Blum, A. Roli, Metaheuristics in combinatorial optimization: Overview and conceptual comparison, ACM Computing Surveys 35 (3) (2003) 268–308.

[17] L. Di Gaspero, A. Schaerf, EasyLocal++: An object-oriented framework for flexible design of local search algorithms, Software—Practice and Experience 33 (8) (2003) 733–765.

[18] M. Birattari, T. Stützle, L. Paquete, K. Varrentrapp, A racing algorithm for configuring metaheuristics, in: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2002), Morgan Kaufmann Publishers, New York (NY), USA, 2002, pp. 11–18.

[19] S. Prestwich, A. Roli, Symmetry breaking and local search spaces, in: Proceedings of CPAIOR 2005, Vol. 3524 of Lecture Notes in Computer Science, Springer–Verlag, 2005.

[20] A. Roli, Symmetry-breaking and local search: A case study, in: SymCon'04 – 4th International Workshop on Symmetry and Constraint Satisfaction Problems, 2004.