A Column Generation Approach for Pure Parsimony Haplotyping

Veronica Dal Sasso[1], Luigi De Giovanni[2], and Martine Labbé[3]

1 **Dipartimento di Matematica, Università degli Studi di Padova, via Trieste 63, 35121 Padova, Italy**
`dalsasso@math.unipd.it`

2 **Dipartimento di Matematica, Università degli Studi di Padova, via Trieste 63, 35121 Padova, Italy**
`luigi@math.unipd.it`

3 **GOM, Université Libre de Bruxelles, Bd du Triomphe CP210/01, 1050 Bruxelles, Belgium**
`http://gom.ulb.ac.be/`

Abstract

The knowledge of nucleotides chains that compose the double DNA chain of an individual has a relevant role in detecting diseases and studying populations. However, determining experimentally the single nucleotides chains that, paired, form a certain portion of the DNA is expensive and time-consuming. Mathematical programming approaches have been proposed instead, e.g. formulating the Haplotype Inference by Pure Parsimony problem (HIPP). Abstractly, we are given a set of genotypes (strings over a ternary alphabet $\{0, 1, 2\}$) and we want to determine the smallest set of haplotypes (binary strings over the set $\{0, 1\}$) so that each genotype can be "generated" by some pair of haplotypes, meaning that they are compatible with the genotype and can fully explain its structure.

A polynomial-sized Integer Programming model was proposed by Catanzaro, Godi and Labbé (2010), which is highly efficient but hardly scalable to instances with a large number of genotypes. In order to deal with larger instances, we propose a new model involving an exponential number of variables to be solved via column generation, where variables are dynamically introduced into the model by iteratively solving a pricing problem. We compared different ways of solving the pricing problem, based on integer programming, smart enumeration and local search heuristic. The efficiency of the approach is improved by stabilization and by a heuristic to provide a good initial solution. Results show that, with respect to the linear relaxations of both the polynomial and exponential-size models, our approach yields a tighter formulation and outperforms in both efficiency and effectiveness the previous model for instances with a large number of genotypes.

1998 ACM Subject Classification G.1.6 Optimization

Keywords and phrases computational biology, haplotyping, column generation, integer programming, combinatorial optimization

Digital Object Identifier 10.4230/OASIcs.SCOR.2016.5

1 Introduction

One of the most important achievements of the latest years in biology has been the human genome sequencing, completed in 2001, that has shown how all humans share the 99% of the information contained in the DNA, while all the significant differences are contained in the remaining information. Each site of this 1% portion of the human genome,

that presents a significant variability among the individuals, is called a *Single Nucleotide Polymorphism (SNP)*.

Humans are diploid organisms, meaning that the DNA is organized in pairs of chromosomes, each copy coming from one of the two parents. Every single chain in the DNA is made of a sequence of nucleotides, choosing among the possible four: A, T, C, G. It is known that, regarding human beings, the DNA sites and so also the SNP sites are almost always biallelic, meaning that at each site only two of the four nucleotides can be found. If the nucleotide is equal for both chains, then the SNP is *homozygous*, otherwise it is *heterozygous*. From now on, we denote with *haplotype* the single chain of SNP values for a specific portion of a chromosome copy and with *genotype* the chain providing information regarding the union of the two chromosome copies, that tells us if each SNP in the chain is homozygous or heterozygous. Moreover, we say that two haplotypes resolve a certain genotype if, when paired, the information regarding homozygous and heterozygous sites they give is the same provided by that genotype. Haplotypes have an important role in medical and pharmacologic studies, for example to detect diseases or to study the different behaviour of various individuals to the same therapy. Sequencing them is not practical, as it is very expensive and time consuming, while it is easier to experimentally obtain the information stored in genotypes. We are then facing the *haplotyping* problem, that consists in determining the two haplotypes that resolve a given genotype. Several approaches have been used in order to solve this problem, its difficulty consisting in the fact that, once we have k heterozygous SNPs in the same genotype, we have 2^{k-1} possible pairs of haplotypes that can represent it and we need some criteria to chose the right pair. A classical approach is to apply the *Pure Parsimony* criterion, according to which, given a set of genotypes obtained by a family of individuals, we want to select the minimum number of haplotypes that can resolve all the genotypes.

This problem is called the Haplotype Inference by Pure Parsimony (HIPP) problem. It is well known to be NP-hard [13] and different mathematical programming approaches have been investigated. An exponentially large Integer Programming (IP) formulation with an exponential number of variables and constraints is proposed in [9], able to tackle only small size instances. A combinatorial branch-and-bound algorithm is presented in [18], without great improvements on the efficiency. A different model with an exponential number of variables and constraints can be found in [14], based on a set-covering formulation: variables are related to all possible haplotypes and are dynamically generated by a guided enumeration procedure. Other formulations lead to polynomial-sized models, e.g. [4], where the linear relaxation is weak, [3], that presents a three-index formulation, and [5], where families of valid cuts are derived from the formulation in [10] and hybrid models between existing formulations are proposed. The state-of-the-art polynomial size IP formulation is proposed in [6], which is largely efficient on small and medium-size instances. More efficient non-exact approaches to HIPP have been presented, e.g. [17, 11]. This paper investigates an approach for HIPP to be suitable for large instances. Our contribution consists of a new tighter formulation and basic solution algorithms that outperform previous models on some classes of instances.

The remainder of this paper is organized as follows. Section 2 presents the notation and two new formulations: a slight improvement of the model in [6], polynomial in size and based on class representatives, and a new formulation with an exponential number of variables associated to pairs of haplotypes and genotype subsets and polynomial number of constraints. The latter formulation can be solved via a column-generation approach, whose implementation is detailed in Section 3. Finally, we present computational results showing that our approach is suitable for instances with a large number of genotypes.

2 Formulations

The biallelic property of each SNP allows us to describe a haplotype as a sequence of 0 and 1, where each symbol encodes one of the two possible nucleotides for a specific SNP. A genotype instead is represented as a sequence of symbols chosen among the set $\{0, 1, 2\}$ where 0 and 1 indicate homozygous SNPs in which that specific nucleotide is present and 2 denotes a heterozygous site. A genotype g is resolved by two haplotypes h^1 and h^2 if, for each position p, $h^1_p \neq h^2_p$ whenever $g_p = 2$, and $h^1_p = h^2_p = g_p$ otherwise. A haplotype h and a genotype g are compatible if for every homozygous site p of g we have $g_p = h_p$.

In the HIPP, we are given a set of m genotypes with n SNPs, and we want to determine the least-cardinality set of haplotypes that can be used to resolve all the genotypes.

The two formulations we present in the following stem from the fact that, in a feasible solution to HIPP, each haplotype induces a subset of genotypes that are partially resolved by it, and every genotype belongs to exactly two subsets.

The first formulation slightly improves the one in [6], where each genotype subset is indexed according to the first genotype, in a predefined order, belonging to it. As each genotype belongs to two sets, it can happen that the same genotype g_i should identify two different subsets so that a dummy genotype $g_{i'}$ is created and used to identify the second subset. Thus, we dispose of subsets S_i with index i varying in the set $\bar{K} = K \cup K'$, where $K = \{1, 2, \dots, m\}$ and $K' = \{1', 2', \dots, m'\}$. We also define an ordering such that $1 < 1' < 2 < 2' < \cdots < m < m'$. We then introduce a binary variable x_i, $i \in \bar{K}$, that takes value 1 if in the solution there is a haplotype that induces a subset S_i and 0 otherwise. Genotype subsets are described by variables y^k_{ij}, with $i, j \in \bar{K}$ and $k \in K$, taking value 1 if the k-th genotype belongs to subsets S_i and S_j, 0 otherwise. Further binary variables z_{ip}, $i \in \bar{K}$, $p \in P = \{1, 2, \dots, n\}$ records the value of the p-th SNP in haplotype i. Taking into account symmetries and compatibility issues related to genotypes to be explained by the same haplotype, variables y can be defined on a reduced subset T of triplets (k, i, j) [6]. In particular, if g^k is the k-th genotype, we have $T = \{(k, i, j) \in K \times \bar{K} \times \bar{K} \mid (i < j \leq k \wedge j \neq i' \text{ if } k > i) \vee (i = k \wedge j = k')$ and $\forall\, p \in P,\ (g^i_p = g^j_p = g^k_p) \vee (g^k_p = 2 \wedge g^i_p + g^j_p = 1)\}$. HIPP can be formulated as follows.

$$\text{(PIP)} \quad \min \quad \sum_{i \in \bar{K}} x_i \tag{1}$$

$$\text{s.t.} \quad x_{i'} \leq x_i \qquad \forall\, i \in K \tag{2}$$

$$\sum_{(k,i,j) \in T} y^k_{ij} \geq 1 \qquad \forall\, k \in K \tag{3}$$

$$\sum_{(k,i,j) \in T} y^k_{ij} + \sum_{(k,j,i) \in T} y^k_{ji} \leq x_i \qquad \forall\, k \in K, i \in \bar{K} \tag{4}$$

$$z_{ip} \leq 1 - \sum_{(k,i,j) \in T} y^k_{ij} - \sum_{(k,j,i) \in T} y^k_{ji} \qquad \forall\, k \in K, i \in \bar{K}, p \in P : g^k_p = 0, g^i_p \neq 1 \tag{5}$$

$$z_{ip} \geq \sum_{(k,i,j) \in T} y^k_{ij} + \sum_{(k,j,i) \in T} y^k_{ji} \qquad \forall\, k \in K, i \in \bar{K}, p \in P : g^k_p = 1, g^i_p \neq 0 \tag{6}$$

$$z_{ip} \geq y^k_{ij} \qquad \forall\, (k, i, j) \in T, p \in P : g^k_p = 2, g^i_p \neq 0, g^j_p = 0 \tag{7}$$

$$z_{jp} \geq y^k_{ij} \qquad \forall\, (k, i, j) \in T, p \in P : g^k_p = 2, g^i_p = 0, g^j_p \neq 0 \tag{8}$$

$$z_{ip} \leq 1 - y^k_{ij} \qquad \forall\, (k, i, j) \in T, p \in P : g^k_p = 2, g^i_p \neq 1, g^j_p = 1 \tag{9}$$

$$z_{jp} \leq 1 - y^k_{ij} \qquad \forall\, (k, i, j) \in T, p \in P : g^k_p = 2, g^i_p = 1, g^j_p \neq 1 \tag{10}$$

$$z_{ip} + z_{jp} \geq y^k_{ij} \qquad \forall\, (k, i, j) \in T, p \in P : g^k_p = 2, g^i_p = 2, g^j_p = 2 \tag{11}$$

$$z_{ip} + z_{jp} \leq 2 - y^k_{ij} \qquad \forall\, (k, i, j) \in T, p \in P : g^k_p = 2, g^i_p = 2, g^j_p = 2 \tag{12}$$

$$x_i, y^k_{ij}, z_{ip} \in \{0, 1\} \qquad \forall\, i \in \bar{K}, (k, i, j) \in T, p \in P \tag{13}$$

Table 1 Summary of the notation used.

Sets and data		Variables	
K	set of genotypes' indeces	x_i	set with representative g^i is used
K'	set of dummy genotypes' indeces	y_{ij}^k	g^k is in sets represented by g^i and g^j
$\bar{K}$	set of indeces $K \cup K'$	z_{ip}	record the value of SNPs for haplotype i
$\tilde{K}$	set of heterozygous genotypes	λ^q	Q-pair q is used (or not)
Q	set of Q-pairs $q = (h^q, G^q)$	π^k	dual variables associated to (15)
g^k	k-th genotype	μ_p^k	dual variables associated to (16)
h^q	haplotype associated to Q-pair q	χ^k	g^k is in the solution of PP (or not)
G^q	subset of genotypes associated to q	ζ_p	value of p-th SNP in PP solution

Constraints (2) force the dummy genotype to be used only if the real one is already used as a subset's index, and (3) state that each genotype is resolved. Constraints (4) record whenever the haplotype induced by S_i is used. Constraints (5)-(12) guarantee compatibility issues. With respect to the formulation in [6], we eliminated two sets of redundant constraints and we completed the domains of (7), (8), (9) and (10) to correctly set the values of variables $z_{i,p}$ taking into account all the possible cases for the values of g_p^i, g_p^j and g_p^k.

The second formulation we propose uses an exponential number of binary variables λ^q associated to a pair $q = (h^q, G^q)$ made of a haplotype h^q and a subset of G^q, and taking value 1 if the haplotype h^q is used to resolve all the genotypes in G^q. We denote with Q the set of all possible pairs and refer to any of its elements as a Q-pair. The formulation is derived from a compact quadratic IP model based on two-index variables applying a Dantzig-Wolfe decomposition, as detailed in [8].

We notice that if a genotype has no heterozygous SNPs, it is resolved by taking twice a haplotype equal to the genotype, which must be in the solution (fixed haplotypes). We thus focus on the set $\tilde{K} \subseteq K$ of the genotypes with at least a heterozygous SNP and define, for each Q-pair q, a coefficient c_q equal to 0 if h^q is fixed, 1 otherwise, obtaining the following formulation.

$$\text{(EIP)} \quad \min \sum_{q \in Q} c_q \lambda^q + (m - |\tilde{K}|) \tag{14}$$

$$s.t. \sum_{q \in Q:\ g^k \in G^q} \lambda^q = 2 \qquad \forall\, k \in \tilde{K} \tag{15}$$

$$\sum_{q \in Q:\ g^k \in G^q, h_p^q = 1} \lambda^q = 1 \qquad \forall\, k \in \tilde{K},\ p \in P : g_p^k = 2 \tag{16}$$

$$\lambda^q \in \{0, 1\} \qquad \forall\, q \in Q \tag{17}$$

Constraints (15) ensure that each genotype is resolved by two haplotypes, and constraints (16) ensure that for each heterozygous site of the k-th genotype, only one of the haplotypes used to resolve it has a value 1 in that position, forcing in this way the other haplotype to have a value 0 so that the genotype is correctly resolved. A summary of the notation introduced is shown in Table 1.

3 A column generation approach for EIP

Model PIP can be directly implemented and solved by standard IP solvers, whereas EIP has $O(2^{m\,n})$ variables and solving it with standard solvers can be impractical even for small

size instances. A branch-and-price algorithm should be used [2], where a column generation approach is applied to solve its linear relaxation, that we denote as ELP: at each iteration, we solve a *Reduced Master Problem* (RMP) including a subset of variables and, by applying the theorems on duality, we derive a *pricing problem* (PP) whose aim is to provide either a new variable to possibly improve the solution in the next iteration, or a certificate of optimality.

An initial set of variables is needed, to build a first feasible RMP. We use the following heuristic, based on the approach shown in [7]:

1. initialize the set of haplotypes H to the fixed genotypes, if any
2. for each genotype g with at least one heterozygous site:
   a. look for a haplotype $h \in H$ that can be used to resolve g
   b. if it exists, compute the haplotype v such that h and v resolve g and add v to H
   c. otherwise, build h and v from g by respectively assigning values 0 and 1 to the heterozygous SNPs. Add h and v to H.

3.1 Solving the Pricing Problem

Using the simplex method to solve the RMP, a feasible primal solution to ELP is available, together with a dual solution satisfying the complementary slackness conditions: if the latter solution is dual feasible, then both solutions are optimal. The pricing problem aims at finding any violated dual constraint, corresponding to a primal variable with negative reduced cost. By associating dual variables π and μ to, respectively, constraints (15) and (16), and observing that constraints $\lambda^q \leq 1$ are redundant, we obtain the following dual of ELP:

$$\max \sum_{k=\in\tilde{K}} 2\pi^k + \sum_{k\in\tilde{K}} \sum_{p:g_p^k=2} \mu_p^k \tag{18}$$

$$s.t. \sum_{k:\ g^k\in G^q} \pi^k + \sum_{k:\ g^k\in G^q} \sum_{p:\ g_p^k=2, h_p^q=1} \mu_p^k \leq c_q \qquad \forall\, q \in Q \tag{19}$$

$$\pi^k \geq 0 \qquad \forall\, k \in \tilde{K} \tag{20}$$

Let π_{RM}, μ_{RM} be the dual values from the RMP. The pricing problem, with coefficients (π_{RM}, μ_{RM}), can be formulated as:

$$\text{(PP)}\ \max \sum_{k\in\tilde{K}} \pi_{RM}^k \chi^k - \sum_{k\in\tilde{K}} \sum_{p:g_p^k=2} {\mu_{RM}}_p^k \zeta_p \chi^k - c(\zeta) \tag{21}$$

$$s.t.\ \zeta_p \leq 1 - \chi^k \qquad \forall\, k \in \tilde{K}, p \in P:\ g_p^k = 0 \tag{22}$$

$$\zeta_p \geq \chi^k \qquad \forall\, k \in \tilde{K}, p \in P\ :\ g_p^k = 1 \tag{23}$$

$$\zeta_p, \chi^k \in \{0,1\} \qquad \forall\, k \in \tilde{K}, p \in P \tag{24}$$

where variables ζ and χ describe respectively the haplotype and the genotype subset of the Q-pair, the constraints guarantee compatibility, and $c(\zeta)$ is either 0 or 1, depending on ζ configuring a fixed haplotype or not. The PP can be resolved by first considering the fixed haplotypes, one at a time, and then the other haplotypes. In the first case ζ is given, $c(\zeta) = 0$ and PP can be solved by inspection: for each genotype compatible with the fixed haplotype at hand, evaluate $\pi_{RM}^k + \sum_{p:g_p^k=2, h_p=1} {\mu_{RM}}_p^k$ and set $\chi^k = 1$ if it is non negative; then select the haplotype with the largest value for (21). For non-fixed haplotypes, $c(\zeta) = 1$ and PP has a quadratic objective function and can be directly solved using standard solvers, in case after linearizing by means of a two-index variable to represent the product $\zeta_p \chi^k$. We propose here an alternative approach using a smart enumeration of all possible genotype subsets.

▶ **Proposition 3.1.** *The following* Smart Enumeration *procedure solves PP to optimality:*

1. *for each genotype* g^i, $i \in \tilde{K}$, *in a predefined order* $\succ$
   a. *fix* $\chi^i = 1$, *and* $\chi^k = 0$, $\forall\, k : g^i \succ g^k$
   b. *fix* $\zeta_p = g^i_p, \forall\, p \in P : g^i_p \neq 2$
   c. *set* $G(i) = \{g^i\} \cup \{g^j \in G \mid g^j \succ g^i \text{ and } g^j_p + g^i_p \neq 1, \forall\, p \in P\}$
   d. *solve PP restricted to the genotypes in* $G(i)$*; let* α_i *be the corresponding value*
2. *return the solution related to the maximum* α_i

The condition $g^j_p + g^i_p \neq 1$ *in Step 1c ensures that genotypes* g^j *and* g^i *could be resolved by a common haplotype.*

Proof. By fixing variables χ at Step 1a, we obtain a partition of the solution space of PP into $|\tilde{K}|$ subsets. Let $(\bar{h}, \bar{G})$ be a Q-pair associated to a feasible solution within the i-th subset. Since $g^i \in \bar{G}$, $\bar{h}$ has to be compatible with g^i so that fixing variables as in Step 1b does not exclude any feasible solution. Now, let $g^j \in \bar{G}$ with $g^j \neq g^i$. We necessarily have $g^j \succ g^i$, due to the χ-fixing determining the i-th subset. Moreover, $g^j_p + g^i_p \neq 1, \forall\, p \in P$, as otherwise g^j and g^i cannot be resolved by the same $\bar{h}$. Hence no feasible solution in the i-th is lost by fixing ζ (Step 1b) and by restricting to the genotype subset defined in Step 1c, and α_i is the optimal solution of PP in the i-th subset. ◀

Notice that fixing one genotype in the solution allows us to consistently decrease running times, as we can exploit information on homozygous sites to fix some haplotype coordinates and to choose genotypes in a restricted subset.

Before solving PP exactly, a heuristic can be used to quickly find a variable to be added to RMP. We consider a local search algorithm that starts from the fixed haplotype with associated minimum reduced cost (or a random one, if no fixed ones are available) as current solution. Then all the neighbor solutions defined by flipping one coordinate at a time are generated and evaluated. We notice that evaluating a neighbor solution is equivalent to solving PP for a fixed haplotype, which can be efficiently done by inspection. If the best neighbor solution is better than the current one, it is taken as the new current solution, and the procedure iterates, otherwise the procedure stops.

Once a Q-pair $(\bar{h}, \bar{G})$ with negative reduced cost is available, further Q-pairs to conveniently add to RMP can be determined by taking the same haplotype and extending $\bar{G}$ to include further genotypes. The *Extension Procedure* works as follows: we set an ordering on G and add one genotype compatible with $\bar{h}$ at a time to $\bar{G}$; if the reduced cost associated to the variable corresponding to the new Q-pair (formula (21)) is negative, then it can be added to RMP and the procedure iterates, otherwise the procedure stops. Notice that having Q-pairs with largest genotype subsets may improve the convergence of the column generation procedure, since we give the same haplotype the opportunity to resolve more genotypes, so that the objective function is likely to decrease.

3.2 Stabilization

Solutions to model EIP are often highly degenerate, so that it can take several iterations to recognize that the optimal has been reached, since all the variables with negative reduced cost should be added (this is the so-called tailing-off effect). In order to deal with this issue, we derive a lower bound to be used as an alternative termination criterion. To this aim, we

add the following redundant constraint to EIP:

$$\sum_{q \in Q} \lambda^q \leq M \tag{25}$$

where M is a constant large enough to ensure the constraint is always satisfied. In particular, during the column generation procedure we set its value as tight as possible, updating it at each iteration to the current value of the objective function (14). Notice that the dual of ELP and the pricing problem PP change: in particular, the objective functions (18) and (21) has to be increased by $M\,\nu_{RM}$ and ν_{RM}, respectively, where ν_{RM} is the dual variable associated to (25).

▶ **Proposition 3.2.** *Let $\rho_{RM} = (\pi_{RM}, \mu_{RM}, \nu_{RM})$ be the dual variables associated to the optimal solution of the current RMP and z_{RM} the corresponding optimal value. Let $v(\rho_{RM})$ be the value of the optimal solution of the current PP (including ν_{RM}). Then, $LB(\rho_{RM}) = z_{RM} + \min\{0, M\,v(\rho_{RM})\}$ is a lower bound to ELP.*

Proof. We will show that the lower bound corresponds to the Lagrangian Relaxation of ELP where constraints (15) and (16) (but not (25)) are relaxed. The corresponding Lagrangian function is:

$$L(\rho_{RM}) = \sum_{k \in \tilde{K}} 2\pi^k_{RM} + \sum_{\substack{k \in \tilde{K} \\ p: g^k_p = 2}} {\mu_{RM}}^k_p + \eta \tag{26}$$

The sum of the first two addends is related to the dual objective function of ELP and is equal to $z_{RM} - M\,\nu_{RM}$. The value η is equal to

$$\begin{aligned} \eta = \min \quad & \sum_{q \in Q} c_q \lambda^q - \sum_{k \in \tilde{K}} \pi^k_{RM} \sum_{q: g^k \in G^q} \lambda^q - \sum_{\substack{k \in \tilde{K} \\ p: g^k_p = 2}} {\mu_{RM}}^k_p \sum_{\substack{q: g^k \in G^q \\ h^q_p = 1}} \lambda^q \\ \text{s.t.} \quad & \sum_{q \in Q} \lambda^q \leq M \\ & \lambda^q \geq 0,\ \forall\, q \in Q. \end{aligned}$$

Let

$$\tilde{q} = \arg\min_{q \in Q}\ c_q - \sum_{k \in \tilde{K}:\ g^k \in G^q} \pi^k_{RM} - \sum_{k \in \tilde{K}:\ g^k \in G^q} \sum_{p: g^k_p = 2} {\mu_{RM}}^k_p, \tag{27}$$

then η is obtained by setting $\lambda^q = 0$ for all $q \neq \tilde{q}$, and $\lambda^{\tilde{q}} = 1$ if the minimum in (27) is negative, 0 otherwise. Note that this minimum value is exactly the opposite value of PP plus ν_{RM}. ◀

Further convergence issues are determined by dual degeneracy, which requires stabilization techniques (see, e.g. [15]) to prevent "oscillations" of the dual variables. The technique we adopt solves the PP on a convex combination between the values $\rho = (\pi, \mu, \nu)$ of the current optimal dual variables and a stability center $\bar{\rho} = (\bar{\pi}, \bar{\mu}, \bar{\nu})$. This approach has the advantage of exploiting the lower bound defined above and yields a stabilized column generation procedure that can be sketched as follows [16]:

1. set parameters $0 < \Delta < 1$, $\tau \geq 0$ and $\epsilon > 0$
2. initialize the RMP, the stability centre $\bar{\rho} = \rho_0$, $LB(\bar{\rho}) = -\infty$
3. solve current RMP, obtaining the optimal value z_{RM} and the dual variables ρ_{RM}

4. set $\rho_{ST} = \Delta\,\rho_{RM} + (1-\Delta)\bar{\rho}$ and let z_{ST} be the value of the dual objective function computed in ρ_{ST}
5. solve PP with coefficients ρ_{ST}, obtaining the optimal value $v(\rho_{ST})$ and the Q-pair s
6. compute the lower bound $LB(\rho_{ST}) = z_{ST} + \min\{0, M\,v(\rho_{ST})\}$
7. if $LB(\rho_{ST}) > LB(\bar{\rho})$, update $\bar{\rho} = \rho_{ST}$ and $LB(\bar{\rho}) = LB(\rho_{ST})$
8. if the reduced cost of s with respect to ρ_{RM} is negative, add λ^s to the RMP,
9. if $[(z_{RM} - LB(\bar{\rho}))/LB(\bar{\rho})] \leq \tau$, then set $\Delta = 1$
10. if $z_{RM} - LB(\bar{\rho}) < \epsilon$ then stop, otherwise iterate from 3

It is proved that this algorithm yields the optimal solution [16]. The property is based on the following lemmas guaranteeing that, when a *misprice* happens, that is we do not find a variable to be added to the RMP even if we are not at the optimum, the algorithm is always able to update the stability centre, so that we do not get stuck in a non-optimal solution. Here we adapt the proof of the two lemmas to our lower bound definition.

▶ **Lemma 3.3.** *Let s be the Q-pair defined at Step 5. If the variable λ^s does not have a negative reduced cost, then $LB(\rho_{ST}) \geq LB(\bar{\rho}) + \Delta(z_{RM} - LB(\bar{\rho}))$.*

Proof. Denote with $f(\rho)$ the value of the objective function of PP where the coefficients are ρ and the variables assume the optimal values found when solving PP with coefficients ρ_{ST}. Let $\bar{z}$ be the value of the dual objective function computed in $\bar{\rho}$. We have

$$\begin{aligned} LB(\rho_{ST}) &= z_{ST} + M\min\{0, v(\rho_{ST})\} \geq z_{ST} + Mf(\rho_{ST}) = \\ &= \Delta z_{RM} + (1-\Delta)\bar{z} + \Delta M f(\rho_{RM}) + (1-\Delta)Mf(\bar{\rho}) = \\ &= \Delta(z_{RM} + Mf(\rho_{RM})) + (1-\Delta)(\bar{z} + Mf(\bar{\rho})) \geq \\ &\geq \Delta z_{RM} + (1-\Delta)LB(\bar{\rho}) \end{aligned}$$

where the last inequality holds because $f(\rho_{RM}) \geq 0$ by hypothesis and $f(\bar{\rho}) \geq v(\bar{\rho})$. ◀

▶ **Lemma 3.4.** *When a misprice happens, the gap $z_{RM} - LB(\bar{\rho})$ is reduced by at least a factor of $1/(1-\Delta)$.*

Proof. The sequence $\{z^k_{RM}\}_k$, where k indexes the iterations of the stabilized column generation procedure, is not increasing. Thus, we have

$$\begin{aligned} z^{k+1}_{RM} - LB(\bar{\rho}^{k+1}) &\leq z^k_{RM} - LB(\bar{\rho})^{k+1} \leq z^k_{RM} - LB(\rho^k_{ST}) \overset{(*)}{\leq} \\ &\leq z^k_{RM} - LB(\bar{\rho}^k) - \Delta(z^k_{RM} - LB(\bar{\rho}^k)) = (1-\Delta)(z^k_{RM} - LB(\bar{\rho}^k)) \end{aligned}$$

where inequality (*) holds for the previous lemma. Hence

$$\frac{z^k_{RM} - LB(\bar{\rho}^k)}{z^{k+1}_{RM} - LB(\bar{\rho}^{k+1})} \geq \frac{1}{1-\Delta}\,.$$ ◀

We proved that, whenever a misprice takes place, the lower bound increases, so that according to the stabilization algorithm we need to update the stability center. Moreover, the lower bound increases by a factor big enough to ensure the convergence of the lower bound to the optimal solution.

4 Computational results

In this section we report the results obtained from the computational experiments carried out on instances both from the literature and generated on purpose. The former are taken from

the class *hapmap* used in [5]: they are real instances derived from biological data related to chromosomes 10 and 21 over all four HapMap (International HapMap Consortium 2004) populations. The number of genotypes involved varies between 5 and 68, while the SNPs are either 30, 50 or 75. The latter are random instances characterized by a large number of genotypes (*manygen*). In particular, we generated instances with 80, 90 and 100 genotypes and 10, 20 and 30 SNPs (4 per class, for a total of 36 instances), where SNP is heterozygous with a probability between 10% and 40%, and homozygous sites have the same probability of being 0 or 1.

Model PIP and the stabilized column generation approach for ELP have been implemented in C++ using the SCIP 3.2 [1] library and IBM CPLEX 12.4 [12] solver, and have been tested on an Intel Pentium Dual Core E2160 1.8 GHz processor with 4 GB RAM. A time limit is set to 7200 CPU-seconds for all the implementations. Different variants of the column generation algorithm have been implemented, depending on how the pricing problem is solved. The first variant (*ELP+QPP*) looks for a variable with negative reduced cost by first solving PP on fixed haplotypes, then running the local search procedure and, finally, by linearizing the quadratic PP on general haplotypes and solving it with the standard solver. Notice that a procedure is only executed if the previous one fails. The second variant (*ELP+QPPm*) is similar to the first one, but the extension procedure is applied to add more than one variable at each iteration. The third variant (*ELP+SM*) is as the first one, but the smart enumeration procedure is used instead of the standard solver to solve PP. Finally, a fourth variant (*ELP+SMm*) is as the second one, but using smart enumeration.

Notice that the proposed procedures are highly dependent on the order in which we consider the genotypes. The initial heuristic can end up with sets of different cardinality according to how compatible genotypes are ordered, as can be shown with a very simple example: given the genotypes $g_1 = \{01101\}, g_2 = \{22212\}, g_3 = \{10211\}$, we obtain a better solution if we consider the genotypes in the order g_1, g_3, g_2 instead of g_1, g_2, g_3. In smart enumeration, the ordering of the genotypes affects the size of the problem to be solved at each iteration (if we consider a genotype with many homozygous sites, we have many fixed coordinates and, as a consequence, less decision variables). As for the extending procedure, the order considered can change the set of variables with negative reduced cost that are added at each iteration. In our implementation, we consider genotypes ordered according to the increasing number of heterozygous SNPs.

For the parameters of the stabilization procedure, after preliminary calibration, we set the stabilization parameter $\Delta = 0.15$, the tolerances $\epsilon = 0.1$ and $\tau = 0.1$. Moreover, we set the initial stability center to $\pi_0 = \mathbf{0}$.

Tables 2 and 3 detail our results regarding the LP relaxations of model PIP, indicated with PLP, and EIP, indicated with ELP, for respectively the instances in the *hapmap* and the *manygen* class. The first column of each table identifies the solution approach. The following columns point out the percentage of instances solved within the time limit, the Gap, computed as $(z_{INT} - z_{LR})/z_{INT}$, between the integer solution z_{INT} and the solution of the linear relaxation z_{LR} (average, maximum and percentage of instances having integer linear relaxation), and running time (average, maximum and minimum). Note that all the results regarding the Gap and the running time are referred only to those instances solved within the time limit.

We can see that formulation EIP is tighter than PIP, as the Gap is significantly reduced and the percentage of instances having integer linear relaxation is clearly higher. However, for *hapmap* instances the computational time for the column generation approach is not competitive. When we increase the number of the genotypes, as for the proposed random

Table 2 Results for the *hapmap* class of instances

	% solved	% LR-Gap			time (s)		
		average	max	%0-Gap	average	min	max
PLP	100.00	8.28	25.00	13.04	16.97	0.01	270.14
ELP+ QPP	58.33	4.34	22.22	57.14	1261.18	0.65	5759.27
ELP + QPPm	62.50	4.05	22.22	62.50	819.58	0.76	3446.52
ELP + SM	75.00	3.67	22.22	61.11	1598.46	0.50	6267.11
ELP+ SMm	70.83	3.89	22.22	58.82	789.75	0.47	3557.48

Table 3 Results for *manygen* instances

	% solved	% LR-Gap			time (s)		
		average	max	%0-Gap	average	min	max
PLP	100.00	3.06	32.58	50.00	1263.76	491.78	2287.69
ELP + QPP	61.11	0.22	2.43	86.36	646.63	12.13	7128.80
ELP + QPPm	61.11	0.22	2.43	86.36	647.64	11.96	5996.86
ELP + SM	100.00	1.86	25.40	63.89	269.52	7.83	1677.40
ELP + SMm	100.00	1.86	25.40	63.89	209.09	7.80	1047.98

instances, we can see that the new approach is not only theoretically but also practically efficient. Note that, due to the reduced number of SNPs considered for the *manygen* class, even having a larger number of genotypes results in instances tractable using the same time limit set for the *hapmap* class. Moreover, it can be easily seen that solving the pricing problem with smart enumeration sensibly improves results in terms of number of solved instances within the time limit (in particular, all the *manygen* instances are solved). The effect of the extension procedure can be seen in a reduction of the running times.

5 Conclusions

In this paper, we presented and compared two different formulations for HIPP. The first model PIP is linear and polynomial in size, and refines one previous model in literature. The second model EIP has an exponential number of variables and a relatively small set of constraints. Standard solvers are used for PIP, whereas a column generation approach has been devised to solve the linear relaxation of EIP and implemented, taking into account stabilization techniques to improve its efficiency. Computational tests on real and random instances show that EIP is a consistently tighter formulation than PIP, since its linear relaxation solves a remarkably higher number of instance to integer optimality, and the optimality gap is more than halved on average. From the efficiency point of view, EIP shows promising results on instances with a large number of genotypes, since solving the liner relaxation is faster than PIP in this case. Future work includes integrating the proposed column generation algorithm in a branch-and-price procedure to solve EIP, and investigating specialized branching strategies for both PIP and EIP.

References

1 T. Achterberg. Scip: solving constraint integer programs. *Math. Program. Comput. 1*, 1:1.41, 2009.

2 C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. Savelsbergh, and P. H. Vance. Branch-and-price: Column generation for solving huge integer programs. *Operations Research*, 46(3):316–329, 1998.

3 P. Bertolazzi, A. Godi, M. Labbé, and L. Tininini. Solving haplotyping inference parsimony problem using a new basic polynomial formulation, 2006.

4 D. Brown and I. Harrower. A new integer programming formulation for the pure parsimony problem in haplotype analysis. *Algorithms in Bioinformatics. Springer Berlin Heidelberg*, pages 254–265, 2004.

5 D. Brown and I. Harrower. Integer programming approaches to haplotype inference by pure parsimony. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, 3(2):141–154, 2006. doi:10.1109/TCBB.2006.24.

6 D. Catanzaro, A. Godi, and M. Labbé. A class representative model for pure parsimony haplotyping. *INFORMS Journal on Computing 22.2*, pages 195–209, 2010.

7 A. Clark. Inference of haplotypes from pcr-amplified samples of diploid populations. *Mol. Biol.Evol.*, 7:111–122, 1990.

8 V. Dal Sasso, L. De Giovanni, and M. Labbé. Dantzig-wolfe decomposition of a quadratic ip model for haplotyping by pure parsimony. *Technical report,* http://www.math.unipd.it/∼luigi/manuscripts/HIPP/TR-DM-HIPP-DWdecomposition.pdf, 2016.

9 D. Gusfield. Haplotype inference by pure parsimony. *Springer Lecture Notes in Computer Science No.2676*, pages 144–155, 2003.

10 B.V. Halldórson, B. Bafna, N. Edwards, R. Lippert, S. Yooseph, and S. Istrail. A survey of computational methods for SNPs and haplotype inference. *Proc. DIMACS/RECOMB Satellite Workshop*, pages 26–47, 2004.

11 Y. T. Huang, K. M. Chao, and T. Chen. An approximation algorithm for haplotype inference by maximum parsimony. *Journal of Computational Biology, 12*, 10:1261–1274, 2005.

12 IBM. Cplex optimizer, 1994. URL: http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/.

13 G. Lancia, M.C. Pinotti, and R. Rizzi. Haplotyping populations by pure parsimony: Complexity of exact and approximation algorithms. *INFORMS Journal on computing 16*, 4:348–359, 2004.

14 G. Lancia and P. Serafini. A set-covering approach with column generation for parsimony haplotyping. *INFORMS Journal on Computing 21*, 1:151–166, 2009.

15 M. E. Lübbecke and J. Desrosiers. Selected topics in column generation. *Operations Research, 53*, 6:1007–1023, 2005.

16 A. Pessoa, M. Poggi de Aragao, and R. Rodrigues. Algorithms over arc-indexed formulations for single and parallel machine scheduling problems, 2008. URL: http://www.optimization-online.org/DB_FILE/2008/06/2022.pdf.

17 L. Tininini, P. Bertolazzi, A. Godi, and G. Lancia. Collhaps: a heuristic approach to haplotype inference by parsimony. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 7(3):511–523, 2010. doi:10.1109/TCBB.2008.130.

18 L. Wang and Y. Xu. Haplotype inference by maximum parsimony. *Bioinformatics 19*, 14:1773–1780, 2003.