Beaches of Islands of Tractability: Algorithms for Parsimony and Minimum Perfect Phylogeny Haplotyping Problems*

Leo van Iersel[1], Judith Keijsper[1], Steven Kelk[2], and Leen Stougie[1,2]

[1] Technische Universiteit Eindhoven (TU/e), Den Dolech 2, 5612 AX Eindhoven, Netherlands
l.j.j.v.iersel@tue.nl, j.c.m.keijsper@tue.nl
http://www.tue.nl

[2] Centrum voor Wiskunde en Informatica (CWI), Kruislaan 413, 1098 SJ Amsterdam, Netherlands
steven.kelk@cwi.nl, leen.stougie@cwi.nl
http://www.cwi.nl

Abstract. The problem *Parsimony Haplotyping* (PH) asks for the smallest set of haplotypes which can explain a given set of genotypes, and the problem *Minimum Perfect Phylogeny Haplotyping* ($MPPH$) asks for the smallest such set which also allows the haplotypes to be embedded in a *perfect phylogeny* evolutionary tree, a well-known biologically-motivated data structure. For PH we extend recent work of [17] by further mapping the interface between "easy" and "hard" instances, within the framework of (k, l)-*bounded instances*. By exploring, in the same way, the tractability frontier of $MPPH$ we provide the first concrete, positive results for this problem, and the algorithms underpinning these results offer new insights about how $MPPH$ might be further tackled in the future. In both PH and $MPPH$ intriguing open problems remain.

1 Introduction

The computational problem of inferring biologically meaningful haplotype data from the genotype data of a population continues to generate considerable interest at the interface of biology and computer science/mathematics. A popular underlying abstraction for this model (in the context of diploid organisms) represents a genotype as a string over a $\{0, 1, 2\}$ alphabet, and a haplotype as a string over $\{0, 1\}$. The precise goal depends on the biological model being applied but a common, minimal algorithmic requirement is that, given a set of genotypes, a set of haplotypes must be produced which resolves the genotypes.

In this paper we focus on two different models. The first model, the *parsimony haplotyping* (PH) model [10], asks for a smallest (i.e., most parsimonious) set of haplotypes to resolve the input genotypes. To be precise, we are given a *genotype matrix* G with elements in $\{0, 1, 2\}$, the rows of which correspond to genotypes,

* Supported by the Dutch BSIK/BRICKS project.

P. Bücher and B.M.E. Moret (Eds.): WABI 2006, LNBI 4175, pp. 80–91, 2006.

while its columns correspond to sites on the genome, called SNP's. A *haplotype matrix* has elements from $\{0,1\}$, and rows corresponding to haplotypes. Haplotype matrix H *resolves* genotype matrix G if for each row g_i of G, containing at least one 2, there are two rows h_{i_1} and h_{i_2} of H, such that $g_i(j) = h_{i_1}(j)$ for all j with $h_{i_1}(j) = h_{i_2}(j)$ and $g_i(j) = 2$ otherwise, in which case we say that h_{i_1} and h_{i_2} resolve g_i, we write $g_i = h_{i_1} + h_{i_2}$, and we call h_{i_1} the *complement* of h_{i_2} with respect to g_i, and vice versa. A row g_i without 2's is itself a haplotype and is uniquely resolved by this haplotype, which therefore has to be contained in H.

The *Parsimony Haplotyping* problem (PH) is given a genotype matrix G to find a haplotype matrix H with a minimum number of rows that resolves G. There is a rich literature in this area, of which recent papers such as [5] give a good overview. The problem is APX-hard [13,17] and the best known approximation algorithms are rather weak, yielding approximation guarantees of 2^{k-1} where k is the maximum number of 2's appearing in a row of the genotype matrix [13,14]. The lack of success in finding strong approximation guarantees has led many authors to consider methods based on Integer Linear Programming (ILP) [5,10,11,13]. A different response to the hardness is to search for "islands of tractability" amongst special, restricted cases of the problem, exploring the frontier between hardness and polynomial-time solvability. In the literature available in this direction [6,14,17], this investigation has specified classes of (k,l)-*bounded instances*: in a (k,l)-*bounded instance* the input genotype matrix G has at most k 2's per row and at most l 2's per column (cf. [17]). If k or l is a "$*$" we mean instances that are bounded only by the number of 2's per column or per row, respectively. This paper aims to supplement this "tractability" literature with mainly positive results, and doing so almost completes the bounded instance complexity landscape.

Next to the PH model we study a related model: the *Minimum Perfect Phylogeny Haplotyping* ($MPPH$) model [2]. Again a minimum-size set of resolving haplotypes is required but this time under the additional, biologically-motivated restriction that the produced haplotypes permit a *perfect phylogeny* i.e., that they can be placed at the leaves of an evolutionary tree within which each site mutates at most once. Haplotype matrices admitting a perfect phylogeny are completely characterised [8,9] by the absence of the forbidden submatrix

$$F = \begin{bmatrix} 1 & 1 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

The *Minimum Perfect Phylogeny Haplotyping* problem ($MPPH$) is given a genotype matrix G find a haplotype matrix H with a minimum number of rows that resolves G and admits a perfect phylogeny.

The feasibility question (PPH)—given a genotype matrix G, find any haplotype matrix H that resolves G and admits a perfect phylogeny, or state that no such H exists—is solvable in linear-time [7,19]. Researchers in this area are now moving on to explore the PPH question on phylogenetic *networks* [18].

The $MPPH$ problem, however, has so far hardly been studied beyond an NP-hardness result [2] and occasional comments within PH and PPH literature [4][19][20]. In this paper we thus provide what is one of the first attempts to analyse the parsimony optimisation criteria within a well-defined and widely applicable biological framework. We seek namely to map the $MPPH$ complexity landscape in the same way as the PH complexity landscape: using the concept of(k, l)-boundedness. We write $PH(k, l)$ and $MPPH(k, l)$ for these problems restricted to (k, l)-bounded instances.

In [13] it was shown that $PH(3, *)$ is APX-hard. In [6][14] it was shown that $PH(2, *)$ is polynomial-time solvable. Recently in [17], it was shown (amongst various other results) that $PH(4, 3)$ is APX-hard. In this paper, we bring the boundaries between hard and easy classes closer by showing that $PH(3, 3)$ is APX-hard and that $PH(*, 1)$ is polynomial-time solvable.

As far as $MPPH$ is concerned there have been, prior to this paper, no concrete results beyond the above mentioned NP-hardness result. We show that $MPPH(3, 3)$ is APX-hard and that, like their PH counterparts, $MPPH(2, *)$ and $MPPH(*, 1)$ are polynomial-time solvable (in both cases using a reduction to the PH counterpart.)

For both problems the $(*, 2)$-bounded versions remain the intriguing open case. Analogous to a result from [17] for a subclass of $PH(*, 2)$, we show here that $MPPH(*, 2)$ is solvable in polynomial-time if the *compatibility graph* of the input genotype matrix is a clique. The compatibility graph $C(G)$ of a genotype matrix G has vertices representing the rows (genotypes) of G, and there is an edge between two vertices if the corresponding two genotypes coincide in each column in which none of the two has a 2. Our prediction is that learning the complexity of $PH(*, 2)$ and $MPPH(*, 2)$ in the case where the compatibility graph is a (graph-theoretical) sum of two or three cliques, will reveal the complexity of the full classes $PH(*, 2)$ and $MPPH(*, 2)$.

As explained by Sharan et al. in their "islands of tractability" paper [17], identifying tractable special classes can be practically useful for constructing high-speed subroutines within ILP solvers, but perhaps the most significant aspect of this paper is the analysis underpinning the results, which - by deepening our understanding of how this problem behaves - assists the search for better, faster approximation algorithms and for determining the exact beaches of the islands of tractability. Indeed, the continuing absence of approximation algorithms with strong accuracy guarantees underlines the importance of such work. Furthermore, the fact that (prior to this paper) concrete and positive results for $MPPH$ had not been obtained (except for rather pessimistic modifications to ILP models [5]), means that the algorithms given here for the $MPPH$ cases, and the data structures used in their analysis (e.g. the *restricted compatibility graph* in Section 3), assume particular importance.

Finally, this paper yields some interesting open problems, of which the outstanding $(*, 2)$ case (for both PH and $MPPH$) is only one; prominent amongst these questions (which are discussed at the end of the paper) is the question

of whether $MPPH$ and PH instances are inter-reducible, at least within the bounded-instance framework.

The paper is organised as follows. In Section 2 we give the hardness results, in Section 3 we present the polynomial-time solvable cases, and we finish in Section 4 with conclusions and open problems. A full version of the paper including all proofs is available online [12].

2 Hard Problems

Theorem 1. *$MPPH(3,3)$ is APX-hard.*

Proof. The proof in [2] that $MPPH$ is NP-hard uses a reduction from Vertex Cover. Using the same construction, but reducing instead from the APX-hard problem 3-Vertex Cover (i.e., where every vertex has at most degree 3) [1][15], gives a (3,3)-bounded instance. In such a case it is not too difficult to show that (for $\epsilon > 0$) a $(1+\epsilon)$ approximation for the constructed $MPPH$ instance can be used to create a $(1+8\epsilon)$ approximation for the size of the minimum vertex cover on the input graph. We defer the details to a full version of the paper [12]. □

Theorem 2. *$PH(3,3)$ is APX-hard.*

Proof. We observe that in the proof that $PH(4,3)$ is APX-hard, by Sharan et al in [17], the leftmost 2 of an *element genotype* is actually only necessary if the element in question appears in fewer than three triples. This slight modification thus yields a (3,3)-bounded instance, and the reduction used in [17] is otherwise unchanged. We defer the proof of correctness to a full version of the paper [12].□

3 Polynomial-Time Solvability

3.1 Parsimony Haplotyping

The following result shows the polynomial-time solvability of PH on (*,1)-bounded instances.

We say that two genotypes g_1 and g_2 are *compatible*, denoted as $g_1 \sim g_2$, if $g_1(j) = g_2(j)$ or $g_1(j) = 2$ or $g_2(j) = 2$ for all j. A genotype g and a haplotype h are *consistent* if h can be used to resolve g, ie. if $g(j) = h(j)$ or $g(j) = 2$ for all j. The *compatibility graph* is the graph with vertices for the genotypes and an edge between two genotypes if they are compatible. Proof of the following two lemmas is omitted.

Lemma 1. *If g_1 and g_2 are rows of a genotype matrix with at most one 2 per column and g_1 and g_2 are compatible then there exists exactly one haplotype that is consistent with both g_1 and g_2.* □

We use the notation $g_1 \sim_h g_2$ if g_1 and g_2 are compatible and h is consistent with both. We prove that the compatibility graph has a specific structure. A *1-sum* of two graphs is the result of identifying a vertex of one graph with a vertex of

the other graph. A 1-sum of $n+1$ graphs is the result of identifying a vertex of a graph with a vertex of a 1-sum of n graphs. See Figure 1 for an example of a 1-sum of three cliques (K_3, K_4 and K_2).

Lemma 2. *If G is a genotype matrix with at most one 2 per column then every connected component of the compatibility graph of G is a 1-sum of cliques, where edges in the same clique are labelled with the same haplotype.* □

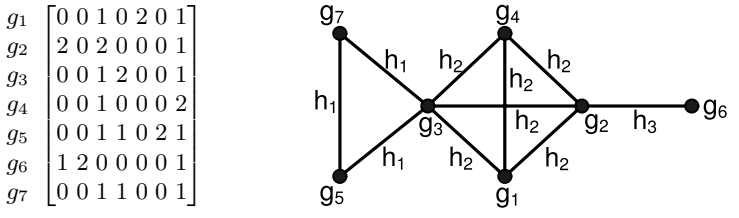

Fig. 1. Example of a genotype matrix and the corresponding compatibility graph, with $h_1 = (0,0,1,1,0,0,1)$, $h_2 = (0,0,1,0,0,0,1)$ and $h_3 = (1,0,0,0,0,0,1)$

From this lemma, it follows directly that in $PH(*,1)$ the compatibility graph is *chordal*, meaning that all its induced cycles are triangles. Every chordal graph has a *simplicial* vertex, a vertex whose (closed) neighbourhood is a clique. Deleting a vertex in a chordal graph gives again a chordal graph (see for example [3] for an introduction to chordal graphs). The following lemma leads almost immediately to polynomial solvability of $PH(*,1)$. We use set-operations for the rows of matrices: thus, e.g., $h \in H$ says h is a row of matrix H, $H \cup h$ says h is added to H as a row, and $H' \subset H$ says H' is a submatrix consisting of rows of H.

Lemma 3. *Given haplotype matrix H' and genotype matrix G with at most one 2 per column it is possible to find, in polynomial time, a haplotype matrix H that resolves G, has H' as a submatrix and has a minimum number of rows.*

Proof. The proof is constructive. Let problem (G, H') denote the above problem on input matrices G and H'. Let C be the compatibility graph of G, which implied by Lemma 2 is chordal. Suppose g corresponds to a simplicial vertex of C. Let h_c be the unique haplotype consistent with any genotype in the closed neighbourhood clique of g. We extend matrix H' to H'' and update graph C as follows.

1. If g has no 2's it can be resolved with only one haplotype $h = g$. We set $H'' = H' \cup h$ and remove g from C.
2. Else, if there exist rows $h_1 \in H'$ and $h_2 \in H'$ that resolve g we set $H'' = H'$ and remove g from C.
3. Else, if there exists $h_1 \in H'$ such that $g = h_1 + h_c$ we set $H'' = H' \cup h_c$ and remove g from C.
4. Else, if there exists $h_1 \in H'$ and $h_2 \notin H'$ such that $g = h_1 + h_2$ we set $H'' = H' \cup h_2$ and remove g from C.

5. Else, if g is not an isolated vertex in C then there exists a haplotype h_1 such that $g = h_1 + h_c$ and we set $H'' = H' \cup \{h_1, h_c\}$ and remove g from C.
6. Otherwise, g is an isolated vertex in C and we set $H'' = H' \cup \{h_1, h_2\}$ for any h_1 and h_2 such that $g = h_1 + h_2$ and remove g from C.

The resulting graph is again chordal and we repeat the above procedure for $H' = H''$ until all vertices are removed from C. Let H be the final haplotype matrix H''. It is clear from the construction that H resolves G.

The proof that H has a minimum number of rows is by induction on the number of genotypes and deferred to a full version of the paper [12]. □

Theorem 3. *The problem $PH(*, 1)$ can be solved in polynomial time.*

Proof. The proof follows from Lemma 3. Construction of the compatibility graph takes $O(n^2m)$ time, for an n times m input matrix. Finding an ordering in which to delete the simplicial vertices can be done in time $O(n^2)$ (see [16]) and resolving each vertex takes $O(n^2m)$ time. The overall running time of the algorithm is therefore $O(n^3m)$. □

3.2 Minimum Pure Parsimony Haplotyping

Polynomial-time solvability of PH on $(2, *)$-bounded instances has been shown in [6] and [14]. We prove it for $MPPH(2, *)$. We start with a definition.

Definition 1. *For two columns of a genotype matrix we say that a* reduced resolution *of these columns is the result of applying the following rules as often as possible to the submatrix induced by these columns: deleting one of two identical rows and the replacement rules, for $a \in \{0, 1\}$,*

$$[2\ a] \to \begin{bmatrix} 1 & a \\ 0 & a \end{bmatrix},\ [a\ 2] \to \begin{bmatrix} a & 1 \\ a & 0 \end{bmatrix},\ [2\ 2] \to \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix} \text{ and } [2\ 2] \to \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Note that two columns can have more than one reduced resolution if there is a genotype with a 2 in both these columns. The reduced resolutions of a column pair of a genotype matrix G are submatrices of (or equal to) F and represent all possibilities for the submatrix induced by the corresponding two columns of a minimal haplotype matrix H resolving G, after collapsing identical rows.

Theorem 4. *The problem $MPPH(2, *)$ can be solved in polynomial time.*

Proof. We reduce $MPPH(2, *)$ to $PH(2,*)$, which can be solved in polynomial time (see above). Let G be an instance of $MPPH(2, *)$. We may assume that any two rows are different.

Take the submatrix of any two columns of G. If it does not contain a [2 2] row, then in terms of Definition 1 there is only one reduced resolution. If G contains two or more [2 2] rows then, since by assumption all genotypes are different, G must have $\begin{bmatrix} 2 & 2 & 0 \\ 2 & 2 & 1 \end{bmatrix}$ and therefore $\begin{bmatrix} 2 & 0 \\ 2 & 1 \end{bmatrix}$ as a submatrix, which can only be

resolved by a haplotype matrix containing the forbidden submatrix F. It follows that in this case the instance is infeasible. If it contains exactly one [2 2] row, then there are clearly two reduced resolutions. Thus we may assume that for each column pair there are at most two reduced solutions.

Observe that if for some column pair all reduced resolutions are equal to F the instance is again infeasible. On the other hand, if for all column pairs none of the reduced resolutions is equal to F then $MPPH(2,*)$ is equivalent to $PH(2,*)$ because any minimal haplotype matrix H that resolves G admits a perfect phylogeny. Finally, consider a column pair with two reduced resolutions, one of them containing F. Because there are two reduced resolutions there is a genotype g with a 2 in both columns. For any such g, replace g in G by h_1 and h_2, where h_1 and h_2 are the haplotypes that correspond to the resolution of g that does not lead to F. This ensures that a minimal haplotype matrix H resolving G can not have F as a submatrix in these two columns.

Repeating this procedure for every column pair either tells us that the matrix G was an infeasible instance or creates a genotype matrix G' such that any minimal haplotype matrix H resolves G' if and only if H resolves G, and H admits a perfect phylogeny. □

Theorem 5. *The problem $MPPH(*,1)$ can be solved in polynomial time.*

Proof. As in the proof of Theorem 4 we reduce $MPPH(*,1)$ to $PH(*,1)$. We defer it to a full version of the paper. □

The open complexity problems in PH and $MPPH$ are now $PH(*,2)$ and $MPPH(*,2)$. Unfortunately, we have not found the answer to these complexity questions. However, the borders have been pushed slightly further. In [17] $PH(*,2)$ is shown to be polynomially solvable if the input genotypes have the complete graph as compatibility graph, we call this problem $PH(*,2)$-$C1$. We will give the counterpart result for $MPPH(*,2)$-$C1$.

Let G be an $n \times m$ $MPPH(*,2)$-$C1$ input matrix. Since the compatibility graph is a clique, every column of G contains only one symbol besides possible 2's. If we replace in every 1-column of G (a column containing only 1's and 2's) the 1's by 0's and mark the SNP corresponding to this column 'flipped', then we obtain an equivalent problem on a $\{0,2\}$-matrix G'. To see that this problem is indeed equivalent, suppose H' is a haplotype matrix resolving this modified genotype matrix G' and suppose H' does not contain the forbidden submatrix F. Then by interchanging 0's and 1's in every column of H' corresponding to a flipped SNP, one obtains a haplotype matrix H without the forbidden submatrix which resolves the original input matrix G. And vice versa. Hence, from now on we will assume, without loss of generality, that the input matrix G is a $\{0,2\}$-matrix.

If we assume moreover that $n \geq 3$, which we do from here on, the *trivial haplotype* h_t defined as the all-0 haplotype of length m is the only haplotype consistent with all genotypes in G. We define the *restricted* compatibility graph $C_R(G)$ of G as follows. As in the normal compatibility graph, the vertices of $C_R(G)$ are the genotypes of G. However, there is an edge $\{g, g'\}$ in C_R(G) only

if $g \sim_h g'$ for some $h \neq h_t$, or, equivalently, if there is a column where both g and g' have a 2.

Lemma 4. *If G is a feasible instance of $MPPH(*,2)$-$C1$ then every vertex in $C_R(G)$ has degree at most 2.*

Proof. Any vertex of degree higher than 2 in $C_R(G)$ implies the existence in G of submatrix:

$$B = \begin{bmatrix} 2 & 2 & 2 \\ 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{bmatrix}$$

It is easy to verify that no resolution of this submatrix permits a perfect phylogeny. □

Suppose that G has two identical columns. There are either 0, 1 or 2 rows with 2's in both these columns. In each case it is easy to see that any haplotype matrix H resolving G can be modified, without introducing a forbidden submatrix, to make the corresponding columns in H equal as well (simply delete one column and duplicate another). This leads to the first step of the algorithm **A** that we propose for solving $MPPH(*,2)$-$C1$:

Step 1 of A: Collapse all identical columns in G.

From now on, we assume that there are no identical columns. Let us partition the genotypes in G_0, G_1 and G_2, denoting the set of genotypes in G with, respectively, degree 0,1, and 2 in $C_R(G)$. For any genotype g of degree 1 in $C_R(G)$ there is exactly one genotype with a 2 in the same column as g. Because there are no identical columns, it follows that any genotype g of degree 1 in $C_R(G)$ can have at most two 2's. Similarly any genotype of degree 2 in $C_R(G)$ has at most three 2's. Accordingly we define G_1^1 and G_1^2 as the genotypes in G_1 that have one 2 and two 2's, respectively, and similarly G_2^2 and G_2^3 as the genotypes in G_2 with two and three 2's, respectively.

The following lemma states how genotypes in these sets must be resolved if no submatrix F is allowed in the solution. If genotype g has k 2's we denote by $g[a_1, a_2, \ldots, a_k]$ the haplotype with entry a_i in the position where g has its i-th 2 and 0 everywhere else.

Lemma 5. *In a feasible solution to the problem $MPPH(*,2)$-$C1$ all genotypes are resolved in one of the following ways:*

1. *a genotype $g \in G_1^1$ is resolved by $g[1]$ and $g[0] = h_t$;*
2. *a genotype $g \in G_2^2$ is resolved by $g[0,1]$ and $g[1,0]$;*
3. *a genotype $g \in G_1^2$ is either resolved by $g[0,0] = h_t$ and $g[1,1]$ or by $g[0,1]$ and $g[1,0]$; or*
4. *a genotype $g \in G_2^3$ is either resolved by $g[1,0,0]$ and $g[0,1,1]$ or by $g[0,1,0]$ and $g[1,0,1]$ (assuming that the two neighbours of g have a 2 in the first two positions where g has a 2).*

Proof. A genotype $g \in G_2^2$ has degree 2 in $C_R(G)$, which implies the existence in G of a submatrix:

$$D = \begin{matrix} g \\ g' \\ g'' \end{matrix} \begin{bmatrix} 2\ 2 \\ 2\ 0 \\ 0\ 2 \end{bmatrix}.$$

Resolving g with $g[0,0]$ and $g[1,1]$ clearly leads to the forbidden submatrix F. Similarly, resolving a genotype $g \in G_2^3$ with $g[0,0,1]$ and $g[1,1,0]$ or with $g[0,0,0]$ and $g[1,1,1]$ leads to a forbidden submatrix in the first two columns where g has a 2. It follows that resolving the genotypes in a way other than described in the lemma yields a haplotype matrix which does not admit a perfect phylogeny.

Now suppose that all genotypes are resolved as described in the lemma and assume that there is a forbidden submatrix F in the solution. Without loss of generality, we assume F can be found in the first two columns of the solution matrix. We may also assume that no haplotype can be deleted from the solution. Then, since F contains [1 1], there is a genotype g starting with [2 2]. Since there are no identical columns there are only two possibilities. The first possibility is that there is exactly one other genotype g' with a 2 in exactly one of the first two columns. Since all genotypes different from g and g' start with [0 0], none of the resolutions of g can have created the complete submatrix F. Contradiction. The other possibility is that there is exactly one genotype with a 2 in the first column and exactly one genotype with a 2 in the second column, but these are different genotypes, i.e., we have the submatrix D. Then $g \in G_2^3$ or $g \in G_2^2$ and it can again be checked that none of the resolutions in (2) and (4) leads to the forbidden submatrix. □

Lemma 6. *Let G be an instance of $MPPH(*, 2)$ and G_1^2, G_2^3 as defined above.*

1. *any nontrivial haplotype is consistent with at most two genotypes in G; and*
2. *A genotype $g \in G_1^2 \cup G_2^3$ must be resolved using at least one haplotype that is not consistent with any other genotype.*

Proof. For the first statement, let h be a nontrivial haplotype. There is a column where h has a 1 and there are at most two genotypes with a 2 in that column. For the second statement, a genotype $g \in G_1^2 \cup G_2^3$ has a 2 in a column that has no other 2's. Hence there is a haplotype with a 1 in this column and this haplotype is not consistent with any other genotypes. □

A haplotype that is only consistent with g is called a *private haplotype* of g. Based on (1) and (2) of Lemma 5 we propose the next step of **A**:

Step 2 of A: Resolve all $g \in G_1^1 \cup G_2^2$ by the unique haplotypes allowed to resolve them according to Lemma 5. Also resolve each $g \in G_0$ with h_t and the complement of h_t with respect to g. This leads to a partial haplotype matrix H_2^p.

The next step of **A** is based on Lemma 6 (2).

Step 3 of A: For each $g \in G_1^2 \cup G_2^3$ with $g \sim_{h'} g'$ for some $h' \in H_2^p$ that is allowed to resolve g according to Lemma 5, resolve g by adding the complement h'' of h' w.r.t. g to the set of haplotypes, i.e., set $H_2^p := H_2^p \cup \{h''\}$, and repeat this step as long as new haplotypes get added. This leads to partial haplotype matrix H_3^p.

Notice that H_3^p does not contain any haplotype that is allowed to resolve any of the genotypes that have not been resolved in Steps 2 and 3. Let us denote this set of leftover, unresolved haplotypes by GL, the degree 1 vertices among those by $GL_1 \subseteq G_1^2$, and the degree 2 vertices among those by $GL_2 \subseteq G_2^3$. The restricted compatibility graph induced by GL, which we denote by $C_R(GL)$ consists of paths and circuits. We first give the final steps of algorithm **A** and argue optimality afterwards.

Step 4 of A: Resolve each cycle in $C_R(GL)$, necessarily consisting of GL_2-vertices, by starting with an arbitrary vertex and, following the cycle, resolving each next pair g, g' of vertices by haplotype $h \neq h_t$ such that $g \sim_h g'$ and the two complements of h w.r.t. g and g' respectively. In case of an odd cycle the last vertex is resolved by any pair of haplotypes that is allowed to resolve it. Note that h has a 1 in the column where both g and g' have a 2 and otherwise 0. It follows easily that g and g' are both allowed to use h (and its complement) according to (4) of Lemma 5.

Step 5 of A: Resolve each path in $C_R(GL)$ with both endpoints in GL_1 by first resolving the GL_1 endpoints by the trivial haplotype h_t and the complements of h_t w.r.t. the two endpoint genotypes, respectively. The remaining path contains only GL_2-vertices and is resolved according to Step 6.

Step 6 of A: Resolve each remaining path by starting in (one of) its GL_2-endpoint(s), and following the path, resolving each next pair of vertices as in Step 4. In case of a path with an odd number of vertices, resolve the last vertex by any pair of haplotypes that is allowed to resolve it in case it is a GL_2-vertex, and resolve it by the trivial haplotype and its complement w.r.t. the vertex in case it is a GL_1 vertex.

By construction the haplotype matrix H resulting from **A** resolves G. In addition, from Lemma 5 follows that H admits a perfect phylogeny. To argue minimality of the solution, first observe that the haplotypes added in Step 2 and Step 3 are unavoidable by Lemma 5 (1) and (2) and Lemma 6 (2). Lemma 6 tells us moreover that the resolution of a cycle of k genotypes in GL_2 requires at least $k + \lceil \frac{k}{2} \rceil$ haplotypes that can not be used to resolve any other genotypes in GL. This proves optimality of Step 4. To prove optimality of the last two steps we need to take into account that genotypes in GL_1 can potentially share the trivial haplotype. Observe that to resolve a path with k vertices one needs at least $k + \lceil \frac{k}{2} \rceil$ haplotypes. Indeed **A** does not use more than that in Steps 5 and 6. Moreover, since these paths are disjoint, they cannot share haplotypes for resolving their genotypes except for the endpoints if they are in GL_1, which can share the trivial haplotype. Indeed, **A** exploits the possibility of sharing the trivial haplotype in a maximal way, except on a path with an even number of

vertices and one endpoint in GL_1. Such a path, with k (even) vertices, is resolved in $\mathbf{A}$ by $3\frac{k}{2}$ haplotypes that can not be used to resolve any other genotypes. The degree 1 endpoint might alternatively be resolved by the trivial haplotype and its complement w.r.t. the corresponding genotype, adding the latter private haplotype, but then for resolving the remaining path with $k-1$ (odd) vertices only from GL_2 we still need $k-1+\lceil\frac{k-1}{2}\rceil$, which together with the private haplotype of the degree 1 vertex gives $3\frac{k}{2}$ haplotypes also (not even counting h_t).

Theorem 6. *$MPPH(*,2)$ is solvable in polynomial time if the compatibility graph is a clique.* □

4 Postlude

There remain a number of open problems. The complexity of $PH(*,2)$ and $MPPH(*,2)$ is still unknown. An approach that might raise the necessary insight is studying $PH(*,2)$-Ck and $MPPH(*,2)$-Ck variants of these problems (i.e., where the compatibility graph is the sum of k cliques) for small k.

Another intriguing open question concerns the relative complexity of PH and $MPPH$ instances. Has $PH(k,l)$ always the same complexity as $MPPH(k,l)$, in terms of well-known complexity measurements (polynomial-time solvability, NP-hardness, APX-hardness)? For hard instances, do approximability ratios differ? There do not yet exist any approximation algorithms for $MPPH$ and an immediate question is whether the weak 2^{k-1} approximation ratio for PH can be attained (or improved) for $MPPH$. A related question is whether it is possible to directly encode PH instances as $MPPH$ instances, and/or vice-versa, and if so whether/how this affects the bounds on the number of 2's in columns and rows.

For hard $PH(k,l)$ instances it would also be interesting to determine if the 2^{k-1} approximation ratio can be improved for fixed l. Finally, with respect to $MPPH$, it could be good to explore how parsimonious the solutions are that are produced by the various PPH feasibility algorithms, and whether searching through the entire space of PPH solutions (as proposed in [19]) yields practical algorithms for solving $MPPH$.

References

1. Alimonti, P., Kann, V., Hardness of approximating problems on cubic graphs, Proc. of the 3rd Italian Conf. on Algorithms and Complexity, 288–298 (1997)
2. Bafna, V., Gusfield, D., Hannenhalli, S., Yooseph, S., A Note on Efficient Computation of Haplotypes via Perfect Phylogeny, *J. of Computational Biology*, 11(5), pp. 858–866 (2004)
3. Blair, J.R.S., Peyton, B., An introduction to chordal graphs and clique trees, in *Graph theory and sparse matrix computation*, pp. 1–29, Springer (1993)
4. Bonizzoni, P., Vedova, G.D., Dondi, R., Li, J., The haplotyping problem: an overview of computational models and solutions, *J. of Computer Science and Technology* 18(6), pp. 675–688 (2003)

5. Brown, D., Harrower, I., Integer programming approaches to haplotype inference by pure parsimony, *IEEE/ACM Transactions on Computational Biology and Informatics* 3(2) (2006)
6. Cilibrasi, R., Iersel, L.J.J. van, Kelk, S.M., Tromp, J., On the Complexity of Several Haplotyping Problems, Proc. 5th Workshop on Algorithms in Bioinformatics (WABI 2005), LNBI 3692, Springer Verlag, Berlin, pp. 128–139 (2005)
7. Ding, Z., Filkov, V., Gusfield, D., A linear-time algorithm for the perfect phylogeny haplotyping (PPH) problem, *J. of Computational Biology*, 13(2) pp. 522–533 (2006)
8. Gusfield, D., *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*, Cambridge University Press (1997)
9. Gusfield, D., Efficient algorithms for inferring evolutionary history, *Networks* 21, pp. 19–28 (1991)
10. Gusfield, D., Haplotype inference by pure parsimony, Proc. 14th Symp. Combinatorial Pattern Matching CPM'03, pp. 144–155 (2003)
11. Halldórsson, B.V., Bafna, V., Edwards, N., Lippert, R., Yooseph, S., Istrail, S., A survey of computational methods for determining haplotypes, Proc. DIMACS/RECOMB Satellite Workshop: Computational Methods for SNPs and Haplotype Inference, pp. 26–47 (2004)
12. Iersel, L.J.J. van, Keijsper, J.C.M., Kelk, S.M., Stougie, L., Beaches of islands of tractability: Algorithms for parsimony and minimum perfect phylogeny haplotyping problems, technical report, http://www.win.tue.nl/bs/spor/2006-09.pdf (2006)
13. Lancia, G., Pinotti, M., Rizzi, R., Haplotyping populations by pure parsimony: complexity of exact and approximation algorithms, *INFORMS J. on Computing* 16(4) pp. 348–359 (2004)
14. Lancia, G., Rizzi, R., A polynomial case of the parsimony haplotyping problem, *Operations Research Letters* 34(3) pp. 289–295 (2006)
15. Papadimitriou, C.H., Yannakakis, M., Optimization, approximation, and complexity classes, *J. Comput. System Sci.* 43, pp. 425–440 (1991)
16. Rose, D.J., Tarjan, R.E., Lueker, G.S., Algorithmic aspects of vertex elimination on graphs, *SIAM J. Comput.* 5, pp. 266–283 (1976)
17. Sharan, R., Halldórsson, B.V., Istrail, S., Islands of tractability for parsimony haplotyping, *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, to appear
18. Song, Y.S., Wu, Y., Gusfield, D., Algorithms for imperfect phylogeny haplotyping (IPPH) with single haploplasy or recombination event, Proc. 5th Workshop on Algorithms in Bioinformatics (WABI 2005), LNBI 3692, Springer Verlag, Berlin, pp. 152–164 (2005)
19. VijayaSatya, R., Mukherjee, A., An optimal algorithm for perfect phylogeny haplotyping, *J. of Computational Biology*, to appear
20. Xian-Sun Zhang, Rui-Sheng Wang, Ling-Yun Wu, Luonan Chen, Models and Algorithms for Haplotyping Problem, *Current Bioinformatics* 1, pp. 105–114 (2006)