Tractable cases of $(*,2)$-bounded parsimony haplotyping

Tractable Cases of $(*,2)$-Bounded Parsimony Haplotyping

Judith Keijsper and Tim Oosterwijk

Abstract—Parsimony haplotyping is the problem of finding a set of haplotypes of minimum cardinality that explains a given set of genotypes, where a genotype is explained by two haplotypes if it can be obtained as a combination of the two. This problem is NP-complete in the general case, but polynomially solvable for (k,l)-bounded instances for certain k and l. Here, k denotes the maximum number of ambiguous sites in any genotype, and l is the maximum number of genotypes that are ambiguous at the same site. Only the complexity of the $(*,2)$-bounded problem is still unknown, where $*$ denotes no restriction. It has been proved that $(*,2)$-bounded instances have compatibility graphs that can be constructed from cliques and circuits by pasting along an edge. In this paper, we give a constructive proof of the fact that $(*,2)$-bounded instances are polynomially solvable if the compatibility graph is constructed by pasting cliques, trees and circuits along a bounded number of edges. We obtain this proof by solving a slightly generalized problem on circuits, trees and cliques respectively, and arguing that all possible combinations of optimal solutions for these graphs that are pasted along a bounded number of edges can be enumerated efficiently.

Index Terms—Drug design, hereditary disease, health, haplotyping, graph, polynomial time algorithm, shortest path, matching problem, path partition

1 Introduction

Understanding the common variations that occur in the human genome is important for e.g. drug design and the study of hereditary diseases [1], [5]. Diploid organisms such as humans have two copies of each chromosome that are not necessarily identical. Each copy is called a *haplotype.* Any two copies of the human genome are identical in almost all sites. Single sites where variations do commonly occur are known as *single nucleotide polymorphisms* or SNPs. If we assume that the other sites are indeed identical in all individuals, the only interesting information in a haplotype is the contents of its SNPs. Since there are most often only two different DNA bases found at an SNP, we may encode these as 0 and 1 and view a haplotype as a binary vector with an entry from $\{0,1\}$ for each SNP.

The combined data of both haplotypes is referred to as the *genotype* of an individual, which is responsible for inherited characteristics. We may view a genotype as a ternary vector with an entry from $\{0,1,2\}$ for each SNP. This vector has an entry 0 at a particular SNP if both haplotypes have a 0 at that site, and a 1 if both have a 1 (such sites where the haplotypes agree are called homozygous sites). The genotype has a 2 if the haplotypes do not agree on that site, i.e., if one has a 0 and one has a 1 (such a site is called a heterozygous site). Note that if the genotype of an individual is known, one can generally not derive the haplotypes of that individual, since it is unclear for each heterozygous site which of the two haplotypes has a 0 and which has a 1. If two haplotypes $h, h' \in \{0,1\}^n$ combine to a given genotype $g \in \{0,1,2\}^n$ they are said to *explain* or *resolve* that genotype. For brevity, we write $g = h \oplus h'$. The haplotypes h and h' are said to be each other's complements with respect to g. A haplotype h is said to be *consistent* with a genotype g if there exists another haplotype h' such that $g = h \oplus h'$.

With current techniques, genotype data is easier to obtain than haplotype data. Most applications ask for haplotype data, however [8]. Consequently, the problem of inferring the haplotypes of a population from genotype data is a well-studied problem in bioinformatics. An important approach to this inference problem is *parsimony haplotyping* (*PH*), where the task is to find a smallest set of haplotypes explaining a given set of genotypes. The parsimony principle is used to argue that such a set of minimum cardinality is more likely to be the real set of haplotypes in the population than any bigger set.

The input of the parsimony haplotyping problem is specified by a *genotype matrix*, a $\{0,1,2\}$-matrix whose rows are the given genotypes. In [12], the notion of a (p,q)-*bounded instance* has been introduced. This is an instance of the parsimony haplotyping problem with at most p 2s per genotype (row of the input matrix) and at most q 2s per site (column of the input matrix), where an asterisk indicates no constraints. We denote the (p,q)-bounded parsimony haplotyping problem by $PH(p,q)$ for short.

The computational complexity of $PH(p,q)$ has been established for most p and q. The problem $PH(3,3)$ is NP-hard and even APX-hard [10], while $PH(2,*)$ is polynomial [4], [11] and $PH(*,1)$ is polynomial [10]. This leaves $PH(*,2)$ as the only case where the complexity is still open.

Two genotypes are said to be *compatible* if there exists a haplotype which is consistent with both genotypes. From this, the notion of a *compatibility graph* arises for a given set of genotypes, which has a vertex for every genotype and an

- *J. Keijsper is with the Department of Mathematics, Technische Universiteit Eindhoven, Eindhoven, Netherlands. E-mail: j.c.m.keijsper@tue.nl.*
- *T. Oosterwijk is with the School of Business and Economics, Maastricht University, Maastricht, Netherlands.*
  E-mail: t.oosterwijk@maastrichtuniversity.nl.

Manuscript received 21 Jan. 2014; revised 25 July 2014; accepted 30 July 2014. Date of publication 25 Aug. 2014; date of current version 30 Jan. 2015.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TCBB.2014.2352031

edge between any two compatible genotypes. In [13], it has been proved that $PH(*,2)$ is solvable in polynomial time for instances where the compatibility graph is a clique. In [13], it has been shown that $PH(*,2)$ is solvable in polynomial time if the compatibility graph can be constructed from two cliques by pasting them along an edge. In [2], it has been proved that $PH(*,2)$ is solvable in polynomial time if the compatibility graph has bounded treewidth.

In [2] and [3], the structure of compatibility graphs of general $PH(*,2)$ instances has been studied. It turns out in [3] that such a compatibility graph can be constructed from cliques and circuits by iteratively *pasting along* an edge (or more formally, along a 2-clique). Here, the operation of pasting along an edge takes two vertex disjoint graphs and identifies an edge (including both endpoints) of one graph with an edge of the other graph, or it takes a vertex of one graph and identifies it with a vertex of the other graph. Any tree, for example, can be constructed from cliques of size 2 by identifying vertices, so it can be considered a result of repeatedly applying this pasting operation to 2-cliques. In this paper, we extend the results from [13] and [13] and prove that $PH(*,2)$ is polynomial time solvable in case the compatibility graph is pasted from any bounded number of cliques, trees, and circuits. Note that in this statement, one tree might be pasted from an unbounded number of 2-cliques, so we obtain a stronger result here if we consider the trees as a separate class of basic graphs.

Our proof method consists of generalizing the haplotyping problem at hand in a way that resembles the constrained haplotype inference problem studied in [6] and [7]. In the constrained problem from these papers, besides a set of genotypes G, a set H' of plausible haplotypes is given, and the solution H is required to be a minimum cardinality subset of this set H' that explains every genotype in G. We also work with a set of 'plausible' or rather 'known' haplotypes H', but we do not assume knowledge of a complete set of possible haplotypes. We do assume that the given set H' is correct. This leads to a 'rooted' version rather than a constrained version of the parsimony haplotyping problem: given a set of genotypes G and a 'known' set of haplotypes H', determine a set of haplotypes H of minimum cardinality explaining G such that $H' \subseteq H$. This is equivalent to determining a set H with a minimum number of haplotypes outside the given set H'. We denote the (p,q)-bounded rooted parsimony haplotyping (RPH) problem by $RPH(p,q)$. If the set H' of known haplotypes is empty, this problem is exactly the classical parsimony haplotyping problem.

The rooted haplotyping problem (Gusfield [9] calls this problem the Modified-Parsimony Haplotype Inference problem) is of practical relevance, since it introduces the possibility to take existing knowledge into account. Indeed, if some haplotypes are known to occur in a population (for example because there are individuals whose genotypes are homozygous at every site, or heterozygous at only a single site), those haplotypes must be added to any solution found for the inference problem. So in the end, a most parsimonious solution is obtained by adding to the known haplotypes a minimum cardinality set of unknown haplotypes (i.e., haplotypes distinct from the known ones) that together with the known haplotypes explain all observed genotypes.

Moreover, a theoretical reason for studying the rooted problem is that it allows us to extend polynomiality results for $PH(*,2)$ instances with certain basic compatibility graphs (cliques, trees and circuits) to $PH(*,2)$ instances with compatibility graphs that are pasted along a bounded number of edges from any number of those basic graphs. The fact that the rooted haplotyping problem forms a useful induction hypothesis has already been exploited in [10] to prove polynomiality of $PH(*,1)$, where compatibility graphs are 1-sums of cliques.

In Section 3 of this paper, we first deal with the rooted haplotyping problem for circuits. We prove that an $RPH(*,2)$ instance is polynomial time solvable if the compatibility graph associated with the given set G of genotypes is a circuit. Since a circuit has bounded treewidth, this was already known for classical $PH(*,2)$ circuit instances [2], [13]. We reduce the rooted haplotyping problem for circuits to a sequence of shortest path problems. This also suggests a new (special-purpose) polynomial-time solution method for classical $PH(*,2)$ circuit instances.

Alternatively, the polynomial time dynamic programming algorithm from [13] for enumerable $PH(*,*)$ instances can be slightly modified to solve enumerable $RPH(*,*)$ instances, and also the method from [2] for constructing in polynomial time a set of haplotypes of polynomial size containing an optimal solution for a $PH(*,2)$ instance can be adapted to the case of an $RPH(*,2)$ instance. This implies that the $RPH(*,2)$ problem is polynomial-time solvable on instances of bounded treewidth, an extension of the result from [2]. Polynomiality of $RPH(*,2)$ tree and circuit instances follows immediately.

Next, in Section 4, we deal with the rooted haplotyping problem for cliques. We prove that an $RPH(*,2)$ instance is polynomial-time solvable if the compatibility graph associated with the given set G of genotypes is a clique, thereby generalizing the result from [13]. We obtain this result by reducing the haplotyping problem to a matching problem. This reduction is quite technical and due to page limits we moved a large part of this reduction to the appendix, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TCBB.2014.2352031.

Finally, in Section 5, we show how the results from all previous sections can be combined to our main theorem, stating that any (rooted) $PH(*,2)$ instance with a compatibility graph that is pasted from trees, circuits and cliques along a bounded number of edges can be solved in polynomial time. Clearly, it follows that any (rooted) $PH(*,2)$ instance with a compatibility graph that is pasted from any bounded number of circuits and cliques can be solved in polynomial time. We leave the complexity of the general $PH(*,2)$ problem open.

2 Preliminary Results

2.1 Problem Statement and Our Contribution

In this section, we introduce the Parsimony Haplotyping problem and the Rooted Parsimony Haplotyping problem as mathematical problems. To be able to do this, we first give strict mathematical definitions for all biological concepts involved.

A *genotype of length k* is a vector from $\{0,1,2\}^k$, and a *haplotype of length k* is a vector from $\{0,1\}^k$.

So by this abstract definition, any haplotype is a genotype without any 2-entries. When viewed as genotypes, haplotypes will be referred to as *unambiguous genotypes* and genotypes with at least one 2-entry will be called *ambiguous genotypes*.

If $g \in \{0,1,2\}^k$ is a genotype of length k, we denote its ith entry ($1 \leq i \leq k$) by $g(i)$. Two haplotypes h and h' of length k are said to *explain* or *resolve* g, denoted by $g = h \oplus h'$, if $g(i) \in \{0,1\}$ implies $h(i) = h'(i) = g(i)$ and $g(i) = 2$ implies $h(i) \neq h'(i)$ for every $1 \leq i \leq k$. The haplotypes h and h' are then said to be each other's *complements with respect to g*, or each other's *g-mates*. A haplotype h of length k is *consistent* with the genotype g if $h(i) = g(i)$ whenever $g(i) \in \{0,1\}$. Note that h is consistent with g if and only if there is a unique haplotype h' such that $g = h \oplus h'$. If g is unambiguous, then $g = g \oplus g$ and g is the only haplotype consistent with g. A *set* of genotypes G is *explained* or *resolved* by a set of haplotypes H if for every $g \in G$ there exist $h, h' \in H$ such that h and h' explain g. Two genotypes are said to be *compatible* if there exists a haplotype which is consistent with both genotypes. The *compatibility graph* $\text{comp}(G)$ for a set of genotypes G is the graph with vertex set G and edge set the compatible pairs of genotypes from G. The *Parsimony Haplotyping problem* is the following problem.

Parsimony Haplotyping Problem (PH)
Given: a set G of genotypes from $\{0,1,2\}^k$
Find: a set H' of haplotypes from $\{0,1\}^k$ of minimum cardinality that explains G.

Note that the input set G could contain unambiguous genotypes. If H is the set of unambiguous genotypes (haplotypes) contained in G, then $H \subset H'$ for any solution H' to *PH*.

A *(p,q)-bounded instance* of *PH* is an instance where there are at most p 2's in every given genotype and at most q 2's in every position (so for each position $i \in \{1, \dots k\}$ there are at most p genotypes g with $g(i) = 2$ among the given genotypes). An asterisk instead of p or q denotes no constraint on the number of 2's in the genotypes or positions. We denote by $PH(p,q)$ the (p,q)-bounded version of the Parsimony Haplotyping problem. By a $PH(*,2)$ clique (circuit, tree) instance we mean a $(*,2)$-bounded instance G where $\text{comp}(G)$ is a clique of size at least 3 (or a circuit, or a tree, respectively).

If we explicitly distinguish between the given unambiguous and the given ambiguous genotypes, we can reformulate the Parsimony Haplotyping problem as follows.

Rooted Parsimony Haplotyping Problem (RPH)
Given: a pair (G,H), where G is a set of ambiguous genotypes from $\{0,1,2\}^k$ and H is a set of unambiguous genotypes from $\{0,1\}^k$.
Find: a set H' of haplotypes from $\{0,1\}^k$ of minimum cardinality that explains G and contains H.

By $RPH(p,q)$ we denote the (p,q)-bounded version of the Rooted Parsimony Haplotyping problem, where at most p 2's are allowed in any genotype and at most q 2's are allowed in any position (in all given genotypes together). By an $RPH(*,2)$ clique (circuit, tree) instance (G,H) we mean a $(*,2)$-bounded instance where $\text{comp}(G)$ is a clique (or circuit, tree, respectively).

Note that $PH(p,q)$ and $RPH(p,q)$ define the same collection of problems: if (G,H) is an $RPH(p,q)$-instance then $G \cup H$ is an equivalent $PH(p,q)$-instance, and if G is a $PH(p,q)$-instance then $(G, \emptyset)$ is an equivalent $RPH(p,q)$-instance. However, as the compatibility graphs $\text{comp}(G)$ and $\text{comp}(G \cup H)$ are not the same if $H \neq \emptyset$, an $RPH(p,q)$ clique (circuit, tree) instance is generally not a $PH(p,q)$ clique (circuit, tree) instance.

Therefore the following theorem, which we will prove in Section 4 of this paper, is a proper generalization of the corresponding result for $PH(*,2)$ clique instances from [13].

Theorem 1. *$RPH(*,2)$ is solvable in polynomial time for clique instances.*

Similarly, the next theorems, which we will prove in Section 3 of this paper, are proper generalizations of the corresponding results for $PH(*,2)$ from [2].

Theorem 2. *$RPH(*,2)$ is solvable in polynomial time for circuit instances.*

Theorem 3. *$RPH(*,2)$ is solvable in polynomial time for tree instances.*

In fact, it is proved in [2] that $PH(*,2)$ is solvable in polynomial time whenever the compatibility graph has bounded treewidth, and we will also generalize that result in Section 3 to the following theorem.

Theorem 4. *$RPH(*,2)$ is solvable in polynomial time for instances (G,H) where the compatibility graph comp(G) has bounded treewidth.*

To prove these theorems, we will use generalized concepts and techniques from [13]. We first recall some definitions and results from [13] and extend them to the rooted case.

Lemma 1 ([13]). *For every clique K in the compatibility graph comp(G) of an $RPH(*,2)$ instance (G,H), there exists a haplotype that is consistent with every genotype in K. If $|K| \geq 3$, this haplotype is unique.*

Proof. All genotypes in K are pairwise compatible, so for each position i one may pick one value $h(i) \in \{0,1\}$ such that either $g(i) = h(i)$ or $g(i) = 2$ for each $g \in K$. This defines a haplotype h, consistent with all $g \in G$. Since the instance is $(*,2)$-bounded, there are at most two genotypes having a 2 in a position i. So if $|K| \geq 3$, then for each i there is at least one $g \in G$ with $g(i) \neq 2$ so $h(i) = g(i)$ is uniquely defined in this case. □

For a clique K of size at least 3 in the compatibility graph of an $RPH(*,2)$ instance, we will call the unique haplotype consistent with every genotype in K the *clique haplotype* of K and denote it by h^c_K. If the compatibility graph is a clique of size at least 3, or if it is clear to what clique in the compatibility graph we are referring, we simply denote the clique haplotype by h^c. Moreover, we say that a haplotype h is a *clique haplotype* of the $RPH(*,2)$ instance (G,H) if $h = h^c_K$ for some clique K of cardinality $|K| \geq 3$ in $\text{comp}(G)$.

The second statement from the previous lemma can now be reformulated as follows.

Lemma 2. *In an $RPH(*, 2)$ clique instance (G, H), every haplotype that is not the clique haplotype is consistent with at most 2 genotypes.*

Trivially, the same holds for an $RPH(*, 2)$ instance where the compatibility graph is a tree or a circuit of length at least 4.

Lemma 3. *If the compatibility graph of an $RPH(*, *)$ instance is a tree or a circuit of length at least 4, then every haplotype is consistent with at most two genotypes.*

Proof. If the haplotype h is shared by three genotypes g_1, g_2, and g_3 from the input set of ambiguous genotypes G, then in the compatibility graph there is a triangle on the vertices g_1, g_2, and g_3. Contradiction. □

The reason for our special interest in circuit, tree, and clique instances is that every compatibility graph of an $RPH(*, 2)$ instance is built up from these basic building blocks. The structure of $(*, 2)$-bounded compatibility graphs was studied in [3]. The following definitions and structural results are from that paper.

A k-clique is a clique of size k. A graph G is obtained from two graphs G_1 and G_2 by *pasting along a k-clique* if for some $r \leq k$ there exist two r-cliques $\{x_1, \ldots, x_r\}$ in G_1 and $\{y_1, \ldots, y_r\}$ in G_2 such that G is isomorphic to the graph obtained from the disjoint union of G_1 and G_2 by identifying x_i with y_i for all $i = 1, \ldots, r$ [3]. Note that the pasting operation can be applied iteratively. A graph obtained from a collection of graphs by iteratively pasting members of that collection along k-cliques is called a *k-paste* of that collection. Also note that $r = 0$ is allowed, resulting in the disjoint union of two graphs. So a disconnected graph can be viewed as a k-paste of its components, for any $k \geq 0$. We will only be concerned with 2-pastes in this paper. To state our main result below, it will be convenient to specify a bound on the number of vertices involved in any of the pasting operations to form a 2-paste. For this purpose, we define a *k-bounded 2-paste* of a collection of graphs to be a graph G that is a 2-paste of the collection, with the restriction that the number of vertices of G that are the result of at least one identification in a 2-pasting operation is at most k.

A *graph of separability at most k* is defined as a graph in which every two non-adjacent vertices can be separated by removing at most k other vertices, or equivalently (by Menger's theorem) in which there are at most k internally vertex disjoint paths between any two non-adjacent vertices. A graph is said to be a *(p, q)-bounded compatibility graph* if it is the compatibility graph $\text{comp}(G)$ of some (p, q)-bounded (rooted) parsimony haplotyping instance. This could be either a $PH(p, q)$ instance G, or an $RPH(p, q)$ instance (G, H). The following result from [3] links $(*, k)$-boundedness and the notion of separability.

Lemma 4. *A graph is of separability at most k if and only if it is a $(*, k)$-bounded compatibility graph.*

The structure theorem for graphs of separability at most 2 proved in [3] thus implies the following result on $RPH(*, 2)$ compatibility graphs.

Lemma 5. *The compatibility graph of any $RPH(*, 2)$ instance is built from cliques and chordless circuits of length at least 4 by iteratively pasting along 2-cliques.*

For reasons outlined below, we prefer to reformulate this result as follows.

Lemma 6. *The compatibility graph of any $RPH(*, 2)$ instance is built from cliques of size at least 3, trees, and chordless circuits of length at least 4 by iteratively pasting along 2-cliques.*

Proof. This follows from Lemma 5, since a 2-paste of 2-cliques is a tree. □

Lemma 6 suggests the following definition. A *basic graph* is a clique of size at least 3, a tree, or a chordless circuit of length at least 4. Lemma 6 then claims that any compatibility graph of any $RPH(*, 2)$ instance is a 2-paste of basic graphs.

The main result of this paper is the following theorem, which will be derived in Section 5 from Theorems 1, 2, and 3.

Theorem 5. *For any positive integer k, there is a polynomial time algorithm for $RPH(*, 2)$ instances for which the compatibility graph is a k-bounded 2-paste of basic graphs.*

Here it becomes apparent why we prefer to view trees as separate basic graphs: any tree is a 2-paste of 2-cliques, but not necessarily a k-bounded 2-paste of 2-cliques. So we obtain a stronger result by taking trees to be one type of basic graphs. An even stronger result can be obtained by allowing building blocks that are more general than trees and circuits: graphs of bounded treewidth. This theorem will be derived in Section 5 from Theorems 1 and 4.

Theorem 6. *For any positive integer k, there is a polynomial time algorithm for $RPH(*, 2)$ instances for which the compatibility graph is a k-bounded 2-paste of cliques and graphs of bounded treewidth.*

Finally, since bounded treewidth is equivalent to bounded clique number for graphs of separability at most 2 by a result from [3], we can replace the building blocks of bounded treewidth by building blocks of bounded clique number to obtain the following formulation of our main result. The *clique number* of a graph is the size of its largest clique.

Theorem 7. *For any positive integer k, there is a polynomial time algorithm for $RPH(*, 2)$ instances for which the compatibility graph is a k-bounded 2-paste of cliques and graphs of clique number at most k.*

Note that an $RPH(*, 2)$ instance (G, H) satisfies the condition of this theorem if $\text{comp}(G)$ contains at most k vertices that are in at least one maximal clique of size more than k, and in at least one other maximal clique.

Corollary 1. *Any $RPH(*, 2)$ instance for which the compatibility graph is a 2-paste of a bounded number of cliques, trees and circuits can be solved in polynomial time in the input size.*

To obtain these results, we first solve $RPH(*, 2)$ on basic graphs in Sections 3 and 4.

2.2 Inference Paths

In [13], it was shown for $PH(*, 2)$ clique instances that partitioning the compatibility graph (clique) into so-called *clique-inference cycles* corresponds to an optimal way of resolving the input-genotypes. If the clique cannot be partitioned into clique-inference cycles, an optimal way of resolving is to explain every genotype by the clique haplotype and its mate.

To solve the more general $RPH(*,2)$ problem on cliques we generalize the notions of *clique-inference paths* and clique-inference cycles from [13]. The same definition is used for circuits and trees. The following notion accurately describes the possible sharing of haplotypes in resolutions for all basic $RPH(*,2)$ instances.

An *H-inference walk* for a basic $RPH(*,2)$ instance (G,H) is a walk $(g_1, g_2, \dots, g_r)$ in the basic graph $\mathrm{comp}(G)$ (so the g_i are sequentially compatible genotypes) together with an associated sequence of haplotypes $(h_1, h_2, \dots, h_r, h_{r+1})$ such that $h_i \oplus h_{i+1} = g_i$ for every $1 \le i \le r$, and moreover such that if h_i is a clique haplotype or a haplotype from H, then $i = 1$ or $i = r+1$. In other words, as soon as a clique haplotype or a haplotype from H is encountered, the walk is terminated.

To show that (self-)intersection of H-inference walks is restricted, we study the transitions that H-inference walks are composed of. For a basic $RPH(*,2)$ instance (G,H), a *transition* through some $g \in G$ is a triple (g_1, g, g_2), with $g \in G$ and $g_1, g_2 \in G \cup H$ three distinct genotypes, such that there exist haplotypes h_1 and h_2 consistent with g_1 and g_2 respectively, that are each other's g-mates: $g = h_1 \oplus h_2$. Note that if g_1 and g_2 are both input genotypes from G, then (g_1, g, g_2) is a transition if and only if it corresponds to an H-inference walk in the compatibility graph of G. If $g_1 = h \in H$ and $g_2 \notin H$, then (g_1, g, g_2) is a transition if and only if (g, g_2) is an H-inference walk having an associated haplotype sequence that starts with h. If $g_1 = h \in H$ and $g_2 = h' \in H$ then (g_1, g, g_2) is a transition if and only if (g) is an H-inference walk with associated haplotype sequence (h, h').

A transition through g in a basic $RPH(*,2)$ problem suggests a unique resolution of g. This is clear in case $g_1 \in H$, $g_2 \in H$, or both, since any haplotype has a unique g-mate. It is also true if both g_1 and g_2 are ambiguous genotypes, as was observed in [2].

Lemma 7 ([2]). *Let (G,H) be an $RPH(*,2)$ instance, and let (g_1, g, g_2) be a transition for this instance, with $g \in G$, and $g_1, g_2 \in G \cup H$. Then there is at most one pair of haplotypes $\{h_1, h_2\}$ that resolves g such that h_i is consistent with g_i, for $i = 1, 2$.*

Proof. Suppose $g = h_1 \oplus h_2$, $g = h'_1 \oplus h'_2$, where $h_1 \neq h'_1$ are both consistent with g_1, and h_2, h'_2 are both consistent with g_2. Then $h_2 \neq h'_2$ because their g-mates are distinct. We may assume that $h_2(i) = 0$, $h'_2(i) = 1$ for some position i. Then $g_2(i) = 2$ and $g(i) = 2$ because these genotypes are consistent with both haplotypes. But then $h_1(i) = 1$ and $h'_1(i) = 0$, so $g_1(i) = 2$, contradicting the $(*,2)$-boundedness. □

We will call the unique haplotypes h_1 and h_2 corresponding to the transition (g_1, g, g_2) the *transition haplotypes* of this transition.

Next, we claim that if an H-inference walk in a basic instance (G,H) intersects itself, then the set of all edges of $\mathrm{comp}(G)$ traversed by the walk is the set of edges of either a circuit or a simple path in $\mathrm{comp}(G)$ (a single edge may be traversed more than once by the walk, though). For a tree instance this follows from the fact that a tree contains no circuits at all, and for a circuit instance, this follows from the fact that the only possible circuit is the entire circuit. For cliques this claim follows from the following lemma.

Lemma 8. *Let (G,H) be an $RPH(*,2)$ clique instance, and let (g_1, g, g_2) be a transition through $g \in G$, with $g_1, g_2 \in G$. Then if g is compatible with a genotype g_3 distinct from g, g_1, g_2, the only haplotype shared by g and g_3 is the clique haplotype h^c.*

Proof. Let $g = h_1 \oplus h_2$, where h_i is consistent with g_i, $i = 1, 2$. Suppose that $h_3 \neq h^c$ is a haplotype consistent with both g and g_3. Then h_3 must be different from h^c in at least one position i. Without loss of generality assume $h^c(i) = 0$ and $h_3(i) = 1$. Since g is consistent with both h^c and h_3, it follows that $g(i) = 2$. Similarly, $g_3(i) = 2$. Because of the $(*,2)$-boundedness, $g_1(i)$ and $g_2(i)$ are both not equal to 2, and because they are both consistent with h^c, we have $g_1(i) = 0$, $g_2(i) = 0$, and therefore $h_1(i) = 0$, $h_2(i) = 0$. However, $g = h_1 \oplus h_2$ now contradicts $g(i) = 2$. □

It follows that under the conditions of this lemma, no H-inference walk traverses the edge $\{g, g_3\}$, since the clique haplotype by definition terminates any such walk. So in a clique instance, H-inference walks do not (self-)intersect, except possibly at their very ends.

Thus, we have shown that an H-inference walk in a clique or circuit instance traverses only the edges of a simple path or the edges of a circuit, possibly back and forth. Next, we will show that self-intersections of H-inference walks are even more restricted. The next lemma implies together with Lemma 7 that if an H-inference walk in a clique or circuit instance returns on its footsteps (i.e., contains a sequence $(\dots g_1, g_2, g_1, \dots)$, where g_1 and g_2 are distinct genotypes), it terminates after at most one repeated genotype. Indeed, by Lemma 7 the walk does not contain a sequence of the form $(\dots, g_1, g_2, g_3, g_2, g_1, \dots)$, and by the next lemma the walk does not contain a sequence of the form $(\dots, g_1, g_2, g_1, g_2, \dots)$, so it does not turn back twice in consecutive vertices. For a tree-instance the next lemma only implies that an H-inference walk does not turn back twice in consecutive vertices.

Lemma 9. *Let (G,H) be an $RPH(*,*)$ instance, and suppose that h_1, h_2, and h_3 are three distinct haplotypes consistent with two genotypes g_1 and g_2. Suppose moreover that $g_1 = h_1 \oplus h_2$, and $g_2 = h_2 \oplus h_3$. Then $g_1 = g_2$.*

Proof. Suppose first that for a position i, the three haplotypes are identical: $h_1(i) = h_2(i) = h_3(i) = 0$, say. Then it follows from $g_1 = h_1 \oplus h_2$ and $g_2 = h_2 \oplus h_3$ that $g_1(i) = g_2(i) = 0$. Now suppose that $h_1(i) = h_2(i) = 0$, and $h_3(i) = 1$. Then it follows from $g_1 = h_1 \oplus h_2$ that $g_1(i) = 0$. This contradicts the fact that h_3 is consistent with g_1. So this case does not occur, and similarly, the case that $h_1(i) = 1$ and $h_2(i) = h_3(i) = 0$ does not occur. Now suppose that $h_1(i) = h_3(i) = 0$, and $h_2(i) = 1$. Then it follows from $g_1 = h_1 \oplus h_2$ and $g_2 = h_2 \oplus h_3$ that $g_1(i) = g_2(i) = 2$. The remaining cases are symmetric. We conclude that $g_1(i) = g_2(i)$ for every position i. □

Together, Lemmas 7, 8, and 9 imply that an H-inference walk in a clique or a circuit instance does not repeat vertices, except possibly if the walk bites its own tail (i.e., is of the form $(g_1, g_2, \ldots, g_r, g_1, \ldots)$, where $g_1, \ldots, g_r$ are distinct vertices), or if it returns on its footsteps (i.e., is of the form $(g_1, g_2, \ldots, g_{i-1} g_i, g_{i-1}, \ldots)$, where $g_1, \ldots, g_i$ are distinct vertices). Moreover, if such a walk bites its own tail, it repeats its own genotype sequence as well as its haplotype sequence from the start, or it terminates after repeating at most the first two genotypes from its genotype sequence. Moreover, if an H-inference walk in a clique or a circuit instance returns on its footsteps it terminates after exactly one repeated genotype. Also, if the walk bites its own tail and then returns on its footsteps, it must terminate after that, having repeated at most two distinct genotypes. There are no other possibilities for an H-inference walk in a clique or a circuit instances to repeat vertices. In particular, if in such a walk a genotype is repeated, then no new genotypes will come up in the genotype sequence of the walk after that point. This means that with respect to the covering of genotypes by H-inference walks, we might as well consider only walks without repetitions.

To this end, define an *H-inference path* in a basic $RPH(*, 2)$ instance to be a simple H-inference walk, that is an H-inference walk without repeated vertices.

So formally, an H-inference path for a basic $RPH(*, 2)$ instance (G, H) is a pair

$$((g_1, g_2, \ldots, g_r), (h_1, h_2, \ldots, h_r, h_{r+1}))$$

of a path $(g_1, g_2, \ldots, g_r)$ in the basic graph $\mathrm{comp}(G)$ (where the g_i are distinct sequentially compatible genotypes) and an associated sequence $(h_1, h_2, \ldots, h_r, h_{r+1})$ of haplotypes such that $h_i \oplus h_{i+1} = g_i$ for every $1 \leq i \leq r$, and moreover such that if h_i is a clique haplotype or a haplotype from H, then $i = 1$ or $i = r + 1$. The fact that the genotypes on an H-inference path are distinct also means that the haplotypes in the associated haplotype sequence $(h_1, h_2, \ldots, h_r, h_{r+1})$ are distinct, except that the first haplotype (h_1) might be equal to the last (h_{r+1}). Indeed, any haplotype in the haplotype sequence is consistent with at most two of the (distinct) genotypes by Lemmas 2 and 3, except possibly h_1 and h_{r+1} (which might be clique haplotypes).

Now, define the *length* of an H-inference path $((g_1, g_2, \ldots, g_r), (h_1, h_2, \ldots, h_r, h_{r+1}))$ with all g_i distinct to be r. In other words, the length of an H-inference path is equal to the number of vertices in the compatibility graph it covers (its 'revenue'). Let us now define the *cost* of an H-inference path to be the number of haplotypes outside H in its associated haplotype sequence $(h_1, h_2, \ldots, h_r, h_{r+1})$ (since only h_1 and h_{r+1} can be in H, and since all h_i are distinct, except possibly $h_1 = h_{r+1}$, this cost is at least $r - 1$ and at most $r + 1$).

Although formally, an H-inference path is a pair $((g_1, g_2, \ldots, g_r), (h_1, h_2, \ldots, h_r, h_{r+1}))$ of a path and an associated haplotype sequence, often we will just refer to the path $(g_1, g_2, \ldots, g_r)$ in the compatibility graph and not mention the associated haplotype sequence if it is clear that such a sequence exists. The cost of the path is then (well-) defined as the minimum number of haplotypes outside H in any associated haplotype sequence for the path.

We say that an H-inference path $((g_1, g_2, \ldots, g_r), (h_1, h_2, \ldots, h_r, h_{r+1}))$ *suggests* the resolution $g_i = h_i \oplus h_{i+1}$ for every g_i on the path. So every H-inference path suggests an explanation for the genotypes it covers using a number of haplotypes outside H equal to its cost.

Any H-inference path $P = ((g_1, g_2, \ldots, g_r), (h_1, h_2, \ldots, h_r, h_{r+1}))$ is of one of four possible types.

1) If $h_1, h_{r+1} \in H$ (possibly $h_1 = h_{r+1}$), then P is called an *H_2-path*. An H_2-path of length r has cost $r - 1$ (resolving the genotypes as suggested by the path takes $r - 1$ haplotypes outside H). If $h_1 = h_{r+1}$, the H_2-path corresponds to a circuit in the compatibility graph.
2) If $h_1 \in H$ and $h_{r+1} \notin H$, or if $h_1 \notin H$ and $h_{r+1} \in H$, then P is called an *H_1-path*. An H_1-path of length r has cost r. Note however that if $h_{r+1} = h^c$ (or $h_1 = h^c$) is the clique haplotype in a clique instance, then sharing of h^c with other genotypes is still possible. For example, if the haplotype sequences of k distinct H_1-paths all end in the clique haplotype h^c, then explaining the genotypes on all these paths together in the way the paths suggest takes a number of haplotypes outside H equal to the sum of the costs of the paths minus $k - 1$. Also note that h_{r+1} might be consistent with g_1, in which case the H_1-path corresponds to a circuit in the compatibility graph.
3) If $h_1 \notin H$ and $h_{r+1} \notin H$, and $h_1 \neq h_{r+1}$, then P is called an *H_0-path*. An H_0-path of length r has cost $r + 1$. Again, sharing of h_1 or h_{r+1} with other paths is possible if $h_1 = h^c$ or $h_{r+1} = h^c$. Also, h_{r+1} might be consistent with g_1, in which case the H_0-path corresponds to a circuit in the compatibility graph.
4) If $h_1 \notin H$, and $h_1 = h_{r+1}$, then P is called an *H_0-circuit*. An H_0-circuit of length r has cost r. An H_0-circuit always corresponds to a circuit in the compatibility graph. If $h_1 = h^c$ sharing of this haplotype with other H-inference paths is possible.

Note that an associated haplotype sequence of an H_0-circuit in a clique instance can contain the clique haplotype h^c only if h^c is not contained in H (if such a circuit would have an associated haplotype sequence containing $h^c \in H$, this particular circuit would in fact be an H_2-path, and its cost would be $r - 1$).

Two H-inference paths are considered equal if their genotype sequences are the same, or if one is the reverse of the other. Two H_0-circuits are also considered equal if the genotype sequence of one is a cyclic permutation of the genotype sequence of the other. Two H-inference paths that are not considered to be equal by one of the previous two rules, are said to be *distinct*. If an H-inference path has more than one associated haplotype sequence, its cost determines what kind of H-inference path it is (so it is an H_2-path if there exists a haplotype sequence starting and ending with haplotypes from H, it is an H_1-path if there is no such haplotype sequence but there is one starting *or* ending with a haplotype from H etc.). This means that every H-inference path can be considered to be of one unique type.

An H-inference path P *contains* another H-inference path P' if the sequence of genotypes of P', or the reversed sequence, is a subsequence of the genotypes of P. P *strictly contains* P' if P contains P' and has greater length than P'. For example, every prefix $(g_1, \ldots, g_i)$ $(1 \leq i \leq r - 1)$ of an H_2-path $P = (g_1, \ldots, g_r)$ is an H_1-path strictly contained in it. Similarly, every suffix $(g_i, \ldots, g_r)$ $(2 \leq i \leq r)$ is an H_1-path strictly contained in P, and if P has length at least 3 then every subsequence $(g_i, \ldots, g_j)$ with $2 \leq i \leq r-1$ is an H_0-path strictly contained in it.

A *maximal* H-inference path for a basic $RPH(*, 2)$ instance is an H-inference path that is not strictly contained in any longer H-inference path, so it cannot be extended to either side without repeating a genotype in the genotype sequence or using a clique haplotype or a haplotype from H in the associated haplotype sequence.

Given a genotype g and a haplotype h consistent with g, there are at most two maximal H-inference paths that have g in their genotype sequence and h in an associated haplotype sequence. They can be constructed as follows.

To construct the first path, start with the genotype sequence (g) and the haplotype sequence (h), then add the (unique) g-mate h' of h to the right of the haplotype sequence, and add a genotype $g' \neq g$ consistent with h' if h' is not in H and not the clique haplotype, if such a $g' \neq g$ exists (there exists at most one such g'), to the right of the genotype sequence; then add the g'-mate h'' of h' to the right of the haplotype sequence, and continue in this way,

$$((g, g', \ldots), (h, h', h'', \ldots)),$$

extending both sequences to the right, until a haplotype in $H \cup \{h^c\}$ or a previously visited genotype is encountered.

If $h \in H$ or if h is the clique haplotype, we are done: we have obtained the unique maximal H-inference path inferred from g and h. If not, proceed by extending the sequences to the left as much as possible: add a genotype $g_1 \neq g$ consistent with h, if such a $g_1 \neq g$ exists (there exists at most one such g_1), to the left of the genotype sequence, and the g_1-mate h_1 of h to the left of the haplotype sequence. Continue in this way, extending both sequences to the left,

$$((\ldots, g_1, g, g', \ldots), (\ldots, h_1, h, h', h'', \ldots)),$$

until a haplotype in $H \cup \{h^c\}$ or a previously visited genotype is encountered.

If h is not in $H \cup \{h^c\}$, there is possibly a second path that is slightly different from the first. To construct it, start again with the genotype sequence (g) and the haplotype sequence (h), but now start extending these sequences to the left first: add a genotype $g_1 \neq g$ consistent with h, if such a $g_1 \neq g$ exists (there exists at most one such g_1), to the left of the genotype sequence, and the g_1-mate h_1 of h to the left of the haplotype sequence. Continue extending both sequences to the left until a haplotype in $H \cup \{h^c\}$ or a previously visited genotype is encountered. When the sequence has been maximally extended to the left, proceed to extend it to the right: add the (unique) g-mate h' of h to the right of the haplotype sequence, and add a genotype $g' \neq g$ consistent with h' if h' is not in H and not the clique haplotype, if such a $g' \neq g$ exists (there exists at most one such g'), to the right of the genotype sequence; then add the g'-mate of h' to the right of the haplotype sequence, and continue extending both sequences to the right until a haplotype in $H \cup \{h^c\}$ or a previously visited genotype is encountered.

The second path is different from the first path only if there is a genotype $\hat{g}$ that can be inferred by going either right, or left from g, h. Since H-inference walks do not self-intersect nontrivially (Lemmas 7, 8, 9), the inference both ways is necessarily terminated at this genotype $\hat{g}$ then. If in this case the path is extended maximally to the right first, then $\hat{g}$ becomes the rightmost genotype in the genotype sequence; if the path is extended maximally to the left first, then $\hat{g}$ becomes the leftmost genotype in the genotype sequence. The resulting genotype sequences are distinct, so the two H-inference paths are only considered to be equal if these genotype sequences are cyclic permutations of each other and belong to the same H_0-circuit.

More precisely, the only way in which the two constructions yield distinct maximal H-inference paths is when the vertices of both paths coincide with the vertices of a circuit $(g_1, \ldots g_r)$ of length $r \geq 3$ in comp(G), and one path is

$$((g_1, g_2, \ldots, g_r), (h_1, h_2, \ldots, h_r, h_{r+1})),$$

while the other is

$$((g_2, \ldots, g_r, g_1), (h_2, \ldots, h_r, h_{r+1}, h_{r+2})),$$

where $h_1 \neq h_{r+1}$ and $h_2 \neq h_{r+2}$, and where $h_1 \neq h_{r+2}$ are two distinct haplotypes consistent with g_1 that are in $H \cup \{h^c\}$ or not consistent with any other genotype besides g_1. Because $h_1 \neq h_{r+1}$ and $h_2 \neq h_{r+2}$, none of these two paths is an H_0-circuit. Since in addition h_2 and h_{r+1} are clearly not contained in H, each of the two paths is either a H_1-path or an H_0-path. In this situation, we will refer to the two maximal H-inference paths together (each covering the same circuit of comp(G)) as a *pseudocycle.* The situation can also occur for $r = 2$, but in this degenerated case the 'circuit' is actually an edge $\{g_1, g_2\}$ in the compatibility graph, and the two paths (g_1, g_2) and (g_2, g_1) cover this edge using different sets of haplotypes (namely $\{h_1, h_2, h_3\}$ and $\{h_2, h_3, h_4\}$ respectively, where h_2 and h_3 are both consistent with both g_1 and g_2, and h_1 and h_4 are both consistent with g_1). Here, the genotype sequence of the first path is the reversed genotype sequence of the second path. Therefore the paths are simply considered to be the same H-inference path (an H_1-path if h_1 or h_4 is in H, and an H_0-path otherwise), and we will not use the term pseudocycle for this situation.

Now, Lemma 7 implies that two distinct maximal H-inference paths in a basic instance do not overlap in two consecutive edges, unless they form a pseudocycle.

Corollary 2. *Two distinct maximal H-inference paths in a basic $RPH(*, 2)$ instance do not have any transitions in common, or they cover the exact same circuit in the compatibility graph and form a pseudocycle.*

Proof. Since the transition (g_1, g, g_2) has a unique pair of transition haplotypes (h_1, h_2), there are at most two maximal H-inference paths containing it that can be inferred from g and h_1 (or equivalently from g and h_2), as explained above, by extending to the right first, or respectively by extending to the left first. These two maximal paths are identical in all situations except if both paths cover the same circuit of length at least 3 in comp(G) and form a pseudocycle, as defined above. □

2.3 Reductions

Using the notion of H-inference paths, we can essentially view the problem $RPH(*, 2)$ in a basic instance (G, H) as a minimum cost path partition problem (MCPP) on the compatibility graph $\text{comp}(G)$.

Minimum Cost Path Partition Problem (MCPP)
Given: a finite graph R, and a finite collection of paths $\mathcal{P}$ in the graph, such that each path P from the collection $\mathcal{P}$ has an associated cost $c(P)$.
Find: a partition of the vertex set of R into paths from $\mathcal{P}$ of minimum total cost.

The reduction of a basic $RPH(*,2)$ instance to such a minimum cost partition problem is fairly straightforward in case the compatibility graph is not a clique, or if the clique haplotype h^c is contained in the given set H.

Lemma 10. *Let (G,H) be a basic $RPH(*,2)$ instance such that $h^c \in H$ if it is a clique instance. Then this problem is equivalent to the Minimum Cost Path Partition Problem that asks for a minimum cost path partition of $\text{comp}(G)$ by H-inference paths.*

Proof. Let $\mathcal{P}$ be the collection of all H-inference paths in $\text{comp}(G)$, and for any $P \in \mathcal{P}$, let $c(P)$ be the cost of the H-inference path, as defined above (i.e., it is the number of haplotypes outside H in an associated haplotype sequence of the path P).

Then a path partition of $\text{comp}(G)$ by paths from $\mathcal{P}$ of total cost K corresponds to a solution H' for (G,H) of cardinality $|H|+K$. Indeed, resolving each genotype from G (i.e. each vertex from $\text{comp}(G)$) as suggested by the unique path from the partition that traverses it, leads by definition to a solution H' that contains in total exactly K haplotypes outside H (since associated haplotype sequences of distinct paths only share haplotypes from H, the costs of the individual paths add up to the total number of haplotypes used outside H).

Next, let H' be a solution of cardinality $|H|+K$ to the instance (G,H) (with $H \subseteq H'$). For each $g \in G$, fix a resolution $g = h_1 \oplus h_2$ of g with $h_1, h_2 \in H'$. This fixes a resolution of $G \cup H$. Now define the *solution graph* of this instance as the graph with vertex set $G \cup H$ and edges between compatible genotypes from G that share a haplotype outside H in the fixed resolution, and edges between a genotype $g \in G$ and a haplotype $h \in H$ if g is resolved using h in the fixed resolution. Each $g \in G$ is resolved by one fixed pair of haplotypes in the fixed resolution, and each of those two haplotypes is either $h^c \in H$, or is shared with at most one other genotype from G in the resolution, by Lemmas 2 and 3. This means that each vertex in the solution graph, except if it corresponds to the clique haplotype $h^c \in H$ has degree at most 2. So, if we delete the vertices from H from the solution graph, the remaining graph consists of paths and circuits covering G: in fact it defines a path partition of $\text{comp}(G)$ by H-inference paths of total cost at most K (at most, because the fixed resolution yields associated haplotype sequences for the paths in the partition that might not be of minimum cost).

So, there is a $1-1$ correspondence between optimal path partitions of total cost K and optimal solutions to (G,H) of cardinality $|H|+K$ in case $h^c \in H$. □

In Section 3.1, we will apply the above lemma to circuit instances, and in Section 4.2 to clique instances (G,H) with $h_c \in H$. The reduction of RPH to MCPP leads to efficient algorithms in these cases because this reduction can be performed in time polynomial in the size of the RPH-instance, and leads to a MCPP instance of polynomial size, according to the following lemma.

Lemma 11. *Let (G,H) be an $RPH(*,2)$ basic instance with $|G| = n$ and $|H| = m$, and k the length of the genotypes. Then the compatibility graph of this instance can be constructed in polynomial time. The total number of distinct H-inference paths in the compatibility graph is bounded by a polynomial in n and m. Moreover, this polynomial collection of H-inference paths (with associated haplotype sequences) can be constructed in time polynomial in n, m, and k.*

Proof. The compatibility graph of G can be constructed in polynomial time by checking all pairs of genotypes for compatibility in $O(kn^2)$ time.

We have described before how to construct the at most two maximal H-inference paths using a given genotype g in their genotype sequence and a given haplotype h consistent with g in their haplotype sequence. From this description, it is not hard to see that constructing these paths takes at most time $O(kn^2)$.

We will now describe for which combinations of a genotype g and a consistent haplotype h we need to repeat this construction to obtain all distinct maximal H-inference paths. First, we construct all H-inference paths starting their haplotype sequence with some $h \in H \setminus \{h^c\}$. From Lemma 2 we know that every $h \in H \setminus \{h^c\}$ can start the haplotype sequence of at most two H-inference paths. So the number of maximal H_2-paths and H_1-paths arising from those is at most $2m$. Next, we add to our collection all n H-inference paths starting their haplotype sequence with h^c, if our instance is a clique instance of size at least 3 (h^c is consistent with each genotype). Some of these paths might identify (and form an H_2-path or an H_0-circuit in the graph). Some pairs of H_1-paths might form a pseudocycle. In total, at most $O(2m+n)$ maximal H_1-paths and H_2- paths are starting their haplotype sequence with a haplotype in $H \cup \{h^c\}$. Moreover, these are all maximal H_1-paths and H_2- paths.

Next, we construct all maximal H_0-paths and H_0-circuits of length at least 3 by extending all possible transitions to maximal H-inference paths. We may obtain some of the H_1 and H_2-paths and H_0-circuits from the previous step again, but we also discover all maximal H_0-paths and remaining H_0-circuits of length at least 3 in this way. Some new pseudocycles (involving H_0-paths) might be discovered. Every vertex (genotype) g is in the middle of at most $O(n^2)$ transitions, so there are at most $O(n^3)$ maximal H_0 paths that we add to our collection in this step.

It remains to construct maximal H_0-paths of length 2. For this we consider each edge (that is not yet covered by any of the previously constructed maximal paths) of the compatibility graph and view it as a maximal H_0-path. To construct a maximal H_0-path associated with an edge

$\{g, g'\}$, simply construct any haplotype that is consistent with both g and g' (in $O(k)$ time), and then find its g-mate and its g'-mate (again in $O(k)$ time). There are at most $O(n^2)$ maximal H_0-paths of length 2, since there are at most $O(n^2)$ edges in the graph.

Finally, there are at most n isolated vertices in the graph that correspond to maximal H_0-paths of length 1 (for a genotype g find any consistent haplotype and its g-mate, in time $O(k)$).

Since every maximal H-inference path in the graph is either an H_1-path, an H_2-path, part of a pseudocycle, an H_0-path of length at least 3, of length 1 or 2, or an H_0-circuit, every possible maximal H-inference path has been constructed in one of the previous steps (except possibly if it is an H_0-path of length 1 or 2 contained in a previously constructed path, in which case we will discover it in the next step where we add all subpaths of constructed paths to our collection).

To finish our collection of H-inference paths, we now construct all subpaths of the paths constructed so far. We have constructed at most $O(m + n^3)$ maximal H-paths and they all have maximal length n. Since a path of length n has at most n^2 subpaths, the total number of subpaths contained in all these paths together is at most $O((m + n^3)n^2)$.

Hence, there are at most $O(mn^2 + n^5)$ H-inference paths in the compatibility graph, and constructing them all takes polynomial time. □

In a general $RPH(*, 2)$-instance (G, H) there might be clique haplotypes that are not contained in H. In this general case, there is no direct $1 - 1$ correspondence between optimal solutions H' and partitions of the compatibility graph by H-inference paths. However, if we construct associated haplotype sequences for all possible H-inference paths in the basic building blocks of comp(G), the set of haplotypes used in these sequences still constitutes a set of polynomial size (as was argued in the previous lemma), and moreover this set is still guaranteed to contain an optimal solution by the next lemma. In [2], it was already proved that for any $PH(*, 2)$ instance, a collection of haplotypes of polynomial size exists that contains an optimal solution to the instance. Here, we simply adapt this result, and its proof, to our slightly more general RPH framework.

Lemma 12. *Given an $RPH(*, 2)$ instance (G, H), one can compute in polynomial time for each $g \in G$ a polynomial collection of haplotypes H'_g such that there exists an optimal solution in which each genotype $g \in G$ is explained by two haplotypes from H'_g. More specifically, for any subset $G' \subseteq G$ of genotypes an optimal solution to the $RPH(*, 2)$ instance (G', H) is contained in the set*

$$H'_{G'} := \bigcup_{g \in G'} H'_g \cup H$$

of at most $O((|G'| + |H|)^3)$ haplotypes.

Proof. We may assume that $G' = G$ (otherwise simply replace G by G' in the following arguments). We start with $H'_g = \emptyset$ for $g \in G$, and $H'_G = H$. In several steps we add haplotypes to the H'_g, and update H'_G such that $H'_G = \cup_{g \in G} H'_g \cup H$ remains valid at all times.

First we add to H'_g all haplotypes in H that are consistent with g, together with their g-mates. In total, we have added at most $O(|H|)$ haplotypes to H'_g and at most $|G||H|$ haplotypes to H'_G. Then, we construct the compatibility graph of G and compute all the clique haplotypes of maximal cliques of size at least 3. By Lemma 5, there are at most $|G|$ such cliques, and they can be found in polynomial time. For every clique K we add to H'_g the clique haplotype h^c_K for which $g \in K$, along with its g-mate. In total, we have added at most $O(|G|^2)$ haplotypes to H'_g and at most $O(|G|^3)$ haplotypes to H'_G in this step.

Next, for every transition (g_1, g, g_2) we add to H'_g the unique pair of transition haplotypes $\{h_1, h_2\}$ for which $g = h_1 \oplus h_2$ and h_i is consistent with g_i for $i = 1, 2$. Moreover, for $i = 1, 2$, if h_i is not a clique haplotype (in which case we have already added it and its complements in the previous step), we add h_i and its g_i-mate to H'_{g_i}, for $i = 1, 2$. Note that if h_i is not a clique haplotype, then g and g_i are the only genotypes consistent with it (by Lemmas 2, 3 and 5), so with each haplotype added in this step we have also added all its complements. Any $g \in G$ is in the middle of at most $O((|G| + |H|)^2)$ transitions. So in total, at most $O((|G| + |H|)^2)$ haplotypes are added to H'_g (and the same amount is added to all the H_{g_i} together, where g_i are neighbours of g on the transition), so at most $O((|G| + |H|)^3)$ haplotypes are added to H'_G in this step.

Next, for each edge $\{g_1, g_2\}$ of the compatibility graph, if there is no haplotype in H' yet that is consistent with both g_1 and g_2, then we add one such haplotype h to H'_{g_1} and to H'_{g_2}, and we add its g_i-mate to H'_{g_i}, $i = 1, 2$. In total, at most $O(|G|)$ haplotypes are added to each H'_g and at most $O(|G|^2)$ haplotypes are added to H'_G in this step.

Finally, for each input genotype g not resolved by haplotypes added to H' so far (these g are isolated vertices in comp(G)), we add to H'_g a pair of haplotypes resolving g. In total, at most 2 haplotypes are added to H'_g and at most $O(|G|)$ haplotypes are added to H'_G in this step. Since the haplotypes added to H'_G in this step are consistent with a single genotype, we have included with each haplotype all of its complements too.

Note that for every $g \in G$ and $h \in H'_g$, the g-mate of h is also in H'_g. Moreover, for each $h \in H'_G$, the complement of h with respect to every possible $g \in G$ is again in H'_G. Also note that the collection H'_G contains a minimum cost associated haplotype sequence for every possible H-inference path in the compatibility graph of G.

We can now argue as in [2] that if $\overline{H}$ is an optimal solution to the $RPH(*, 2)$ instance (G, H) that has maximum intersection with H'_G, then it is completely contained in H'_G.

Indeed, if h would be a haplotype in $\overline{H} \setminus H$ but not in H'_G, then h is consistent with at most two genotypes from G, since otherwise h is a clique haplotype by Lemma 2, and is by construction contained in H'_G.

If h is consistent with two distinct genotypes g_1 and g_2 from G, and h_1 and h_2 denote the complements of h with respect to those genotypes, then both h_1 and h_2 are not in H'_G, so not in H, and both are not consistent with any $g \in G$, since otherwise h would be a transition haplotype and would by construction be contained in H'_G. But then we can replace $\{h, h_1, h_2\}$ in $\overline{H}$ by any three haplotypes $\{h', h'_1, h'_2\}$ from H'_G such that h' is consistent with g_1 and g_2, and h'_i is the g_i-mate of h, $i = 1, 2$ (such haplotypes are by construction present in H'_G because of the edge $\{g_1, g_2\}$ in the compatibility graph). The solution $\left(\overline{H} \setminus \{h, h_1, h_2\}\right) \cup \{h', h'_1, h'_2\}$ is still optimal, but has a larger intersection with H'_G than $\overline{H}$. Contradiction.

If h is consistent with exactly one genotype $g \in G$, then the g-mate h' of h is also contained in $\overline{H}$, but h' is not contained in H'_G because otherwise its complement h would have been included as well. So $h' \notin H$. Also, h' is not consistent with any other genotype in G because then for h' one of the previous (impossible) cases would hold. But now we can replace $\{h, h'\}$ in $\overline{H}$ by any two haplotypes from H'_G that resolve g, obtaining an optimal solution having a larger intersection with H'_G. Contradiction.

This argument shows that the polynomial-size collection H'_G contains an optimal solution to the instance (G, H). □

An immediate corollary is the following.

Corollary 3. *$RPH(*,2)$ is solvable in polynomial time on instances where $|G|$ and $|H|$ are bounded by a fixed number k.*

Proof. Let (G, H) be such a bounded $RPH(*,2)$ instance. Then we can simply check each subset of the collection H'_G from the previous lemma for being an optimal solution, and this takes polynomial time. □

3 Circuits and Trees

In [2] it was argued that $PH(*,2)$ can be solved in polynomial time if the compatibility graph is of bounded treewidth. This implies that $PH(*,2)$ is polynomial time solvable on circuit instances. In Section 3.1, we will show that also $RPH(*,2)$ circuit instances are polynomial time solvable. In fact, also the more general result about graphs of bounded treewidth extends to RPH, and we will sketch the proof of that claim in Section 3.2.

3.1 Circuits

In this section, we will give a special-purpose algorithm for $RPH(*,2)$ circuit instances that we think is interesting in its own right.

In the case of a circuit, the upper bound on the number of H-inference paths from Lemma 11 can be improved.

Lemma 13. *Let (G, H) be an $RPH(*,2)$-circuit instance with $|G| = n$ and $|H| = m$. Then the collection Q of all H-inference paths in the circuit has cardinality at most $O((m+n)n^2)$ and can be constructed in polynomial time.*

Proof. This follows from the proof of Lemma 11 by noting that if the graph is a circuit, there is no clique haplotype, every vertex is in at most one transition, and there are only n edges to consider as H_0-paths. □

In case of a circuit, the Minimum Cost Path Partition Problem can be solved by solving a series of shortest path problems in an acyclic directed graph.

Lemma 14. *Let C be a circuit, Q a collection of paths in this circuit, each with an associated cost. Then a minimum cost path partition of C by paths from Q can be found in $O(|Q|^3)$ time.*

Proof. Orient the circuit clockwise. Let us represent the vertices on the circuit C consecutively (following the orientation) by the integers $0, 1, \ldots, n-1$. Now denote a path, following the orientation of the circuit, from vertex a to vertex b, where $a \leq b$ by the interval $[a, b]$. Also, if $b < a$ the interval $[a, b]$ denotes the path $a \ldots n-1, 0, 1, \ldots b$ that traverses the vertex 0.

For every path $P = [n-i, j]$ in the collection Q that covers the vertex 0 (so with $j < n-i$) do the following. Construct a weighted acyclic directed graph D_P with a vertex for P, and with a vertex for every path $P' = [a, b]$ in Q such that $j < a \leq b < n-i$, i.e., for every path P' with $P \cap P' = \emptyset$. Add an arc from $P_1 = [a, b]$ to $P_2 = [c, d]$, with the cost of P_1 as its length, if and only if $c = b+1$ and $P_2 \neq P$. Note that by construction P becomes a source. Also add a sink S_P, with arcs from vertices $P' = [l, n-i-1]$ to S_P with the cost of P' as length. Now clearly, a shortest (i.e. smallest length) path from P to S_P in the auxiliary graph D_P corresponds to a minimum cost path partition of the circuit C among the path partitions that use the path P.

Since any path partition uses some P that covers 0, to find the overall minimum cost path partition of the circuit we can simply try every P from Q that covers 0 and take the one for which the shortest path in D_P has minimum length.

For each fixed P constructing the graph D_P takes time $O(|Q|^2)$, and then finding a shortest path in this acyclic digraph with $O(|Q|^2)$ arcs takes time $O(|Q|^2)$ again. Repeating this for every path P in Q covering 0 yields a total complexity of $O(|Q|^3)$ time. □

The polynomiality result for circuit instances is immediate.

Theorem 2. *$RPH(*,2)$ is solvable in polynomial time on a circuit instance (G, H).*

Proof. Since a circuit instance does not have clique haplotypes, Lemma 10 is applicable. Thus, an optimal solution to this instance corresponds to a minimum cost path partition of the circuit by H-inference paths. By Lemma 13, the collection of all H-inference paths is of polynomial size and can be constructed in polynomial time. By Lemma 14, a minimum cost partition of the circuit by paths from such a polynomial collection can be found in polynomial time. □

3.2 Trees

In [13], a polynomial time dynamic programming algorithm is given that solves $PH(*,*)$ on enumerable instances (where there is a polynomial number of haplotypes consistent with each given genotype) of bounded treewidth. We claim (and this is easy to check) that essentially the same polynomial time dynamic programming algorithm solves $RPH(*,*)$ on enumerable instances of bounded treewidth.

In [2], it is observed that the polynomiality of the problem remains valid if instead of an enumerable instance of $PH(*,*)$ we only require the instance to be such that a polynomially-sized set of haplotypes that contains an optimal solution can be computed in polynomial time. This observation also extends to $RPH(*,*)$.

In [2], this observation is used to prove that $PH(*,2)$ is polynomial time solvable on instances of bounded treewidth. Using Lemma 12, we obtain the generalization to $RPH(*,2)$ of that result.

Theorem 4. *$RPH(*,2)$ is solvable in polynomial time on instances where the compatibility graph has bounded treewidth.*

Proof. By Lemma 12, a polynomially-sized set of haplotypes that contains an optimal solution can be computed for any $RPH(*,2)$ instance in polynomial time. By the above observations, this implies that $RPH(*,2)$ can be solved in polynomial time by a dynamic programming algorithm. □

This immediately implies polynomial time solvability of tree-instances.

Theorem 3. *$RPH(*,2)$ is solvable in polynomial time on tree instances.*

Also the polynomial time solvability of circuit-instances, Theorem 2, can be obtained as a consequence of Theorem 4.

4 Cliques

This section is devoted to the proof of the following theorem.

Theorem 1. *$RPH(*,2)$ is solvable in polynomial time for clique instances.*

We distinguish between $RPH(*,2)$ clique instances (G,H) where the clique haplotype is contained in H (in Section 4.2), and instances where this is not the case (in the appendix, available online, due to space constraints). We start with some preliminaries specific to clique instances.

4.1 Preliminaries

In this section, we will introduce some notation and prove some preliminary results on $RPH(*,2)$-clique instances. More specifically, we will prove a result on the disjointness of H-inference paths in $RPH(*,2)$ clique instances, and from that we will derive a lower bound on the number of haplotypes in any solution to an $RPH(*,2)$ clique instance.

Throughout this section on clique instances, we will assume that in the $RPH(*,2)$ clique instance at hand, no $g \in G$ can be completely resolved by haplotypes in H (if $g = h \oplus h'$ for $h, h' \in H$ then we could simply delete g from G and obtain an equivalent problem).

Now define $N(H) \subseteq G$ as the set of all genotypes $g \in G$ that are consistent with some $h \in H$, the "neighbourhood" of H.

Some H-inference paths deserve a special name as we will use them a lot. An H_2-path of length 2 is called a 2-path. An H_1-path of length 2 is called a 1-path. We call 2-paths, 1-paths and even shorter H_2-paths and H_1-paths *short paths*. H_2-paths and H_1-paths of length 1 are also called *very short paths*. H-inference paths that are not short are called *long paths*. Note that an H_0-path of length 2 is a long path, but H_1-paths and H_2-paths of length 2 are short paths. Note that a genotype is on a very short path if and only if it is in $N(H)$. The assumption that no genotype is resolved within H implies that no very short H_2-paths exist. Now call an H-path *2-path free* if it does not strictly contain a 2-path. In particular, a 2-path is 2-path free by this definition.

If $P = (g_1, \dots, g_r)$ is an H-inference path, then $g_1, \dots g_r$ are its *vertices*. If P is an H_0-, an H_1-, or an H_2-path, then g_1 and g_r are called its *endpoints* whereas $g_2, g_3, \dots g_{r-1}$ are called its *internal vertices*. If P is an H_0-circuit, then all its vertices are internal vertices. Two H-inference paths are said to be *internally vertex disjoint* if no internal vertex of one path is a vertex of the other path. Two H-inference paths are said to be *completely vertex disjoint* or simply *vertex disjoint* if they do not have any vertices in common.

Lemma 15. *In an $RPH(*,2)$-clique instance, any two distinct maximal H-inference paths are internally vertex disjoint, or one is a short path completely contained in the other, or they are the two maximal H-paths of a pseudocycle.*

Proof. The statement is trivially true if both paths are short because short paths do not have internal vertices.

Now consider two distinct maximal H-inference paths P_1 and P_2 that do not constitute a pseudocycle, but do intersect at some $g \in G$ which is an internal vertex of P_1 (so P_1 is long). Since g is an internal vertex of P_1, it has two neighbours g_1 and g_2 from G on P_1, or in other words (g_1, g, g_2) is a transition contained in P_1.

Since P_2 is distinct from P_1, by Corollary 2 it does not contain the transition (g_1, g, g_2), so either P_2 is very short and contained in P_1, or P_2 contains one of the edges $\{g, g_1\}$ and $\{g, g_2\}$ but has g as an endpoint, or P_2 contains at least one neighbour g_3 of g that is distinct from both g_1 and g_2, and it traverses the edge $\{g, g_3\}$. However, by Lemma 8, no H-inference path traverses such an edge $\{g, g_3\}$ because $h_3 = h^c$ is the only haplotype consistent with both g and g_3, and the clique haplotype by definition terminates any H-inference path.

So we may assume that g is the first vertex, and $\{g, g_1\}$ is the first edge of the path P_2. If P_2 is short, this means it is contained in P_1. If P_2 is long, it contains a transition (g, g_1, g') with $g' \in G$. By Lemma 8, there is no $g_3 \in G$ distinct from g, g_1, g' such that P_1 traverses the edge $\{g_3, g_1\}$. This means that g_1 is an endpoint of P_1. So $P_1 = g_1, g, g_2, \dots$ and $P_2 = g, g_1, g', \dots$, with g_2 and g' distinct from g_1 and g (both paths are long).

Assume that haplotype sequences for P_1 and P_2 are given by $(h_1, h'_1, h_2, h'_2, \dots)$ and $(h, h_3, h'_3, h', \dots)$, respectively, with $h_1, h \in H \cup \{h^c\}$, so that $g_1 = h_1 \oplus h'_1$, $g = h'_1 \oplus h_2$, and $g = h \oplus h_3$, $g_1 = h_3 \oplus h'_3$, etc. Now suppose that $h'_1 \neq h_3$, say $h'_1(i) = 0, h_3(i) = 1$, and moreover suppose that $h^c(i) = 0$ (without loss of generality). Then since g and g_1 are consistent with both h'_1 and h_3, $g(i) = 2$, $g_1(i) = 2$, and hence $h_2(i) = 1$. Since g_2 is consistent with both h_2 and h^c, $g_2(i) = 2$ which contradicts the $(*,2)$-boundedness. Finally, suppose that $h'_1 = h_3$, then also $h_2 = h$ (g-mates) and

$h_1 = h'_3$ (g_1-mates). So $h_2, h'_3 \in H \cup \{h^c\}$, contradicting the assumption that P_1 and P_2 are long paths. □

In the next lemma, we will derive a lower bound on the cardinality of any solution to an $RPH(*, 2)$-clique instance.

To formulate the lower bound, we introduce the following notation.

Definition 1. *Let (G, H') be an $RPH(*, 2)$-clique instance with clique haplotype h^c. Let $H = H' \cup \{h^c\}$. Then m_c is by definition the maximum number of vertex disjoint H_2-paths in comp(G) (note that H_2-paths are defined here with respect to the extended set of haplotypes H).*

Lemma 16. *Any solution to an $RPH(*, 2)$-clique instance (G, H') has cardinality at least $|G| + |H'| - m_c$, where m_c is defined above.*

Proof. Let (G, H') be an $RPH(*, 2)$-clique instance, and let $H = H' \cup \{h^c\}$. Let H'' be any solution to the instance (G, H'). H'' either contains the clique haplotype h^c or it does not.

If H'' contains h^c, then in fact H'' is also a solution to the instance (G, H). By Lemma 10, H'' gives rise to a path partition of comp(G) by H-inference paths, such that $|H''|$ is greater or equal to $|H|$ plus the cost of this path partition. If the path partition contains q H_0-paths and m H_2-paths, its cost is $|G| + q - m$. Since $q \geq 0$ and $m \leq m_c$, it follows that

$$|H''| \geq |H| + |G| + q - m \geq$$
$$|H| + |G| - m_c \geq |H'| + |G| - m_c$$

in this case.

If H'' does not contain h^c, then as in the proof of Lemma 10, a fixed resolution of $G \cup H'$ using haplotypes from H'' gives rise to a solution graph associated with this resolution: a graph with vertex set $G \cup H'$ and edges between compatible genotypes from G that share a haplotype outside H' in the fixed resolution, and edges between a genotype $g \in G$ and a haplotype $h \in H'$ if g is resolved by h and its g-mate (outside H') in the fixed resolution. By Lemma 2, any vertex corresponding to a haplotype $h \in H'$ (so $h \neq h^c$) is adjacent to at most two vertices $g \in G$ in this solution graph. Each $g \in G$ is resolved by one fixed pair of haplotypes in the fixed resolution, and each of those two haplotypes is shared with at most one other genotype from G in the resolution, again by Lemma 2 (note that h^c is not used in this resolution). This means that each vertex in the solution graph has degree at most 2. So, if we delete the vertices from H' from the solution graph, the remaining graph consists of vertex disjoint paths and circuits covering G: it is a path partition of comp(G) by H'-inference paths. Since $h_c \notin H''$, the haplotype sequences (obtained from the fixed resolution) associated with distinct paths from this partition do not have any haplotypes outside H' in common. So $|H'|$ plus the cost of the path partition (from optimal haplotype sequences) is a lower bound on the cardinality of the solution H''. If the path partition contains q H'_0-paths and m H'_2-paths, then its cost is $|G| + q - m$. Since $q \geq 0$ and $m \leq m_c$ (each H'_2-path is also an H_2-path because $H' \subset H$), it follows also in this case that

$$|H''| \geq |H'| + |G| + q - m \geq |H'| + |G| - m_c.$$

□

In the remainder of this section, we will distinguish between two cases for a given clique instance (G, H').

If $h^c \in H'$, then $H = H'$ in Lemma 16, and we will prove in Section 4.2 that the lower bound in Lemma 16 is attained in this case, and that moreover an optimal solution attaining the lower bound can be determined in polynomial time.

If $h^c \notin H'$, then $H = H' \cup \{h^c\}$ in Lemma 16, and we will show, with some more effort, that also in this case an optimal solution can be determined in polynomial time. Due to space constraints, this proof is postponed to the appendix, available online.

4.2 Cliques with $h^c \in H$

In this section we prove that an optimal solution for any $RPH(*, 2)$-clique instance (G, H) with $h^c \in H$ can be determined in polynomial time. Note that in case $h^c \in H$, any set of k nontrivial genotypes can be trivially resolved using exactly k haplotypes outside H by just resolving every g by $h^c \in H$ and its g-mate outside H. Therefore the only H-inference paths worth consideration in this case are H_2-paths, as these have a cost strictly smaller than their length.

Indeed, the following lemma claims that a minimum cardinality solution (attaining the lower bound of Lemma 16) can be found by determining a maximum number of (completely) vertex disjoint H_2-paths in the compatibility graph.

Lemma 17. *For an $RPH(*, 2)$-clique instance (G, H) with $h^c \in H$, there exists a solution of cardinality $|G| + |H| - m_c$. This solution is optimal.*

Proof. Consider a collection of m_c disjoint H_2-paths. Resolve the p ambiguous genotypes on them as the paths suggest with $p - m_c$ haplotypes outside H. Then resolve the remaining $|G| - p$ ambiguous genotypes using $h^c \in H$ and its complements outside H, so with $|G| - p$ haplotypes outside H in total, one for each genotype. To resolve the unambiguous genotypes in H we are forced to add the haplotypes in H to the solution. This results in a solution of cardinality $|G| + |H| - m_c$. Because this cardinality matches the lower bound for the cardinality of any solution given in Lemma 16, this solution is optimal. □

The next lemma shows that a maximum collection of vertex disjoint H_2-paths in the compatibility graph can be determined in polynomial time because it can be reduced to a matching problem.

Lemma 18. *Let (G, H) be an $RPH(*, 2)$-bounded clique instance with $h^c \in H$. Let G_m be the graph on vertex set $N(H)$ that has an edge between two vertices from $N(H)$ if and only if there exists a 2-path free H_2-path between them in comp(G). Then the problem of finding a maximum collection of vertex disjoint H_2-paths in comp(G) is equivalent to the maximum cardinality matching problem in G_m.*

Proof. On the one hand, note that distinct 2-path free H_2-paths are internally vertex disjoint by Lemma 15. So such paths intersect at their endpoints in $N(H)$ or they do not intersect at all. In other words, the edge set of G_m

completely captures all intersections between such paths. It is now clear that any matching in G_m corresponds to a collection of vertex disjoint (2-path free) H_2-paths of the same cardinality.

On the other hand, we show that there exists a maximum cardinality collection of vertex disjoint H_2-paths consisting entirely of 2-path free paths, i.e., corresponding to a matching in G_m. Indeed, let $\mathcal{P}$ be a maximum collection of vertex disjoint H_2-paths such that the number of paths in $\mathcal{P}$ that are not 2-path free is minimized. Suppose $\mathcal{P}$ contains a path P that strictly contains a 2-path T. Then $\mathcal{P}$ does not contain T, as P and T are not vertex disjoint. Moreover, every $P' \in \mathcal{P} \setminus \{P\}$ is vertex disjoint from T as it is vertex disjoint from P. So, as T is 2-path free, $(\mathcal{P} \setminus \{P\}) \cup \{T\}$ is a maximum collection of vertex disjoint H_2-paths containing fewer paths that are not 2-path free than $\mathcal{P}$. This contradiction proves that $\mathcal{P}$ consists of 2-path free H_2-paths.

So every matching in G_m corresponds to a collection of vertex disjoint H_2-paths in comp(G) of the same cardinality, and there exists a maximum collection of vertex disjoint H_2-paths that corresponds to a matching in G_m of the same cardinality.

This implies that any maximum cardinality matching in G_m corresponds to a maximum cardinality collection of vertex disjoint H_2-paths. □

Combining algorithmic versions of these lemmas results in the main theorem of this section.

Theorem 8. *Let (G, H) be an $RPH(*, 2)$-clique instance with $h^c \in H$. A minimum cardinality solution $H' \supseteq H$ to this instance can be determined in time polynomial in $|G|$, $|H|$, and the length k of the genotypes in $G \cup H$, i.e in time polynomial in the problem size.*

Proof. Note that as $|G| \geq 3$, the genotypes in G uniquely define the clique haplotype h^c, and it is in fact easy to construct h^c from any three genotypes in G. Also all H_2-paths (with their haplotype sequences) can be constructed in time polynomial in $|G|$, $|H|$, and k, by Lemma 11, and then it can be checked in polynomial time which 2-paths are contained in which long H_2-paths. So the graph G_m (of size polynomial in $|G|$ and $|H|$) from Lemma 18 can be constructed in polynomial time.

Since a maximum cardinality matching in a graph can be found in time polynomial in the graph size, Lemma 18 implies that one can find a maximum collection of vertex disjoint H_2-paths in polynomial time. The proof of Lemma 17 suggests how an optimal solution can be derived from this collection of disjoint paths together with their haplotype sequences, in polynomial time. □

4.3 Cliques with $h^c \notin H$

It remains to prove that any RPH(*,2) clique instance (G, H) where $h^c \notin H$ can be optimally solved in polynomial time. Due to space constraints, the long and technical proof of this theorem can be found in the appendix, available online.

Theorem 9. *Let (G, H) be an $RPH(*, 2)$-clique instance having $h^c \notin H$. A minimum cardinality solution $H' \supseteq H$ to this instance can be found in polynomial time.*

Proof. See the appendix, available online. □

Together, Theorem 8 and Theorem 9 imply the following.

Theorem 1. *Let (G, H) be an $RPH(*, 2)$-clique instance. A minimum cardinality solution $H' \supseteq H$ to this instance can be found in polynomial time in the problem size.*

5 Main Result

In this section, we derive our main theorem from the results on basic instances that we proved in the previous sections.

Theorem 5. *For any positive integer k, there is a polynomial time algorithm for $RPH(*, 2)$ instances for which the compatibility graph is a k-bounded 2-paste of basic graphs.*

Proof. Let (G, H) be an $RPH(*, 2)$ instance for which comp(G) is 2-pasted from q basic graphs (trees, circuits and cliques) $G_1, G_2, \ldots, G_q$, where $q \leq |G|$. Let S be the set of shared vertices, i.e., the set of vertices (genotypes) that belong to at least two of the basic graphs G_i. By definition of k-bounded 2-paste, $|S| \leq k$.

By Lemma 12, we can construct in polynomial time a polynomially sized set H' of haplotypes containing an optimal solution to the instance (G, H). Consider all possible resolutions of the genotypes in S by haplotypes from H'. There are at most $|H'|^{|S|} \leq |H'|^k$ such resolutions, which is polynomial in $|H'|$, as k is fixed. For each such resolution, do the following. Define $H'' \subseteq H'$ as the set of haplotypes used in this resolution for the genotypes in S. For each $i = 1 \ldots q$, solve the basic $RPH(*, 2)$ instance given by $(V(G_i), H'')$ in polynomial time in such a way that the optimal solution obtained is a subset of H'. This is possible by Theorems 1, 2 and 3. Note that the set H' is constructed in such a way in Lemma 12 that the haplotype sequences of the H''-inference paths for each basic instance $(V(G_i), H'')$ only contain haplotypes from H' (see also the proof of Lemma 11), and thus indeed a solution contained in H' is obtained by applying these theorems. Denote the optimal solution obtained for $(V(G_i), H'')$ by H_i. Then $H^* := \cup_{i=1}^{q} H_i \subset H'$ is a solution to the instance (G, H). Now take the solution H^* that has minimum cardinality among all solutions found after trying all possible resolutions for the genotypes in S. This is an optimal solution to the instance (G, H) since it is the best solution contained in H', and H' by construction contains an optimal solution.

Clearly, since k is fixed and $q \leq |G|$, the complete algorithm described above runs in polynomial time. □

An even stronger result can be obtained by using Theorem 4 instead of Theorems 2 and 3.

Theorem 6. *For any positive integer k, there is a polynomial time algorithm for $RPH(*, 2)$ instances for which the compatibility graph is a k-bounded 2-paste of cliques and graphs of bounded treewidth.*

Since the compatibility graph of an $RPH(*, 2)$ instance has bounded treewidth if and only if it has bounded clique number, this result can be formulated equivalently as follows.

Theorem 7. *For any positive integer k, there is a polynomial time algorithm for $RPH(*, 2)$ instances for which the compatibility graph is a k-bounded 2-paste of cliques and graphs of clique number at most k.*

Consequently, we have proved that any $PH(*, 2)$ instance for which the compatibility graph is a bounded 2-paste of cliques and graphs of bounded treewidth (or clique number) can be solved in polynomial time. This immediately implies the following Corollary.

Corollary 1. *Any $RPH(*, 2)$ instance for which the compatibility graph is a 2-paste of a bounded number of cliques, trees and circuits can be solved in polynomial time in the input size.*

We leave open the question whether the general $PH(*, 2)$ problem, or more generally the $RPH(*, 2)$ problem, is polynomial time solvable.

Acknowledgments

The first author would like to thank Steven Kelk and Martin Milanič for early stage discussions on the topic of this paper. Both authors would like to thank an anonymous referee for helpful suggestions.

References

[1] [Online]. Available: www.hapmap.org, 2013.
[2] F. Cicalese and M. Milanič, “On parsimony haplotyping,” Technischen Fakultät, Universität Bielefeld, Bielefeld, Germany, Tech. Rep. 04, Apr. 2008.
[3] F. Cicalese and M. Milanič, “Graphs of separability at most 2,” *Discrete Appl. Math.*, vol. 160, no. 6, pp. 685–696, 2012.
[4] R. Cilibrasi, L. Van Iersel, S. Kelk, and J. Tromp, “On the complexity of several haplotyping problems,” in *Algorithms in Bioinformatics*, volume 3692 of *Lecture Notes Computer Science*. Berlin, Germany: Springer, 2005, pp. 128–139.
[5] The International HapMap Consortium, “The international hapmap project,” *Nature*, vol. 426, pp. 789–796, 2003.
[6] M. R. Fellows, T. Hartman, D. Hermelin, G. M. Landau, F. Rosamond, and L. Rozenberg, “Haplotype inference constrained by plausible haplotype data,” *IEEE/ACM Trans. Comput. Biol. Bioinformat.*, vol. 8, no. 6, pp. 1692–1699, Nov./Dec. 2011.
[7] R. Fleischer, J. Guo, R. Niedermeier, J. Uhlmann, Y. Wang, M. Weller, and X. Wu, “Extended islands of tractability for parsimony haplotyping,” in *Proc. 21st Annu. Conf. Combinatorial Pattern Matching*, 2010, pp. 214–226.
[8] D. Gusfield and S. H. Orzack, “Haplotype inference,” in *Handbook on Bioinformatics*, S. Aluru, Ed. London, U.K.: Chapman & Hall, 2006, pp. 18-1–18-25.
[9] D. Gusfield, “Haplotype inference by pure parsimony,” in *Combinatorial Pattern Matching*, volume 2676 of *Lecture Notes in Computer Science*. Berlin, Germany: Springer, 2003, pp. 144–155.
[10] L. van Iersel, J. Keijsper, S. Kelk, and L. Stougie, “Shorelines of islands of tractability: Algorithms for parsimony and minimum perfect phylogeny haplotyping problems,” *IEEE/ACM Trans. Comput. Biol. Bioinformat.*, vol. 5, no. 2, pp. 301–312, Apr. 2008.
[11] G. Lancia and R. Rizzi, “A polynomial case of the parsimony haplotyping problem,” *Oper. Res. Lett.*, vol. 34, no. 3, pp. 289–295, 2006.
[12] R. Sharan, B. V. Halldórsson, and S. Istrail, “Islands of tractability for parsimony haplotyping,” *IEEE/ACM Trans. Comput. Biol. Bioinformat.*, vol. 3, no. 3, pp. 303–311, Jul. 2006.
[13] S. M. Stefánsdóttir, “Models for solving minimum parsimony haplotyping,” Master’s thesis, Reykjavík University, Reykjavík, Iceland, Jun. 2007.

Judith Keijsper received the master’s and the PhD degrees from the Universiteit van Amsterdam, in 1994 and 1998, respectively, where she worked with Alexander Schrijver on combinatorial algorithms for graph problems. After working as a postdoc at Leibniz-IMAG, Grenoble, France, and as an assistant professor at the Universiteit Twente, the Netherlands, for short periods of time, she moved to the Technische Universiteit Eindhoven, the Netherlands, in 2000, where she is an assistant professor in the Combinatorial Optimization group of the Mathematics Department. Her research interest is combinatorial optimization in general, and the application of combinatorial algorithms to problems from computational biology in particular.

Tim Oosterwijk received the master’s degree from the Technische Universiteit Eindhoven, the Netherlands, in 2013, where he worked with Nikhil Bansal on the k-set packing problem. He is currently working toward the PhD degree at Maastricht University, the Netherlands. His research interests are discrete optimization problems, combinatorial algorithms for graph problems, game theory and the application of local search techniques to find approximation algorithms.