Complexity and Approximation of the Minimum Recombination Haplotype Configuration Problem

Lan Liu [1], Xi Chen[2], Jing Xiao[2] and Tao Jiang[1]

[1]Department of Computer Science and Engineering, University of California, Riverside, CA 92521, USA
{lliu, jiang}@cs.ucr.edu
[2]Department of Computer Science and Technology, Tsinghua University, Beijing 100084, P.R.China
{xichen00, xiaojing00}@mails.tsinghua.edu.cn

Abstract. We study the complexity and approximation of the problem of reconstructing haplotypes from genotypes on pedigrees under the Mendelian Law of Inheritance and the minimum recombinant principle (MRHC). First, we show that MRHC for simple pedigrees where each member has at most one mate and at most one child (*i.e.* binary-tree pedigrees) is NP-hard. Second, we present some approximation results for the MRHC problem, which are the first approximation results in the literature to the best of our knowledge. We prove that MRHC on two-locus pedigrees or binary-tree pedigrees with missing data cannot be approximated (the formal definition is given in section 1.2) unless P=NP. Next we show that MRHC on two-locus pedigrees without missing data cannot be approximated within any constant ratio under the Unique Games Conjecture and can be approximated within ratio $O(\sqrt{\log(n)})$. Our L-reduction for the approximation hardness gives a simple alternative proof that MRHC on two-locus pedigrees is NP-hard, which is much easier to understand than the original proof. We also show that MRHC for tree pedigrees without missing data cannot be approximated within any constant ratio under the Unique Games Conjecture, too. Finally, we explore the hardness and approximation of MRHC on pedigrees where each member has a bounded number of children and mates mirroring real pedigrees.

Keywords: Haplotyping, pedigree, recombinant, SNP, complexity, approximation, L-reduction, positive result, negative result, bounded number, children, mates

1 Introduction and Definitions

The secret mechanism behind phenotypic variation and inheritance has intrigued the study of genetic markers. With the discovery of genetic markers such as microsatellite DNA sequences and Single Nucleotide Polymorphisms (SNPs), it is now possible to provide a unique genetic map to track the variation and inheritance of genetic markers. The international HapMap project launched in October 2002, aims to discover the haplotype structure of human beings and examine the common haplotypes among populations [17].

Homologous recombination, the combination of genetic material between chromosome pairs during meiosis, is essential in diploid organisms such as humans [7]. Unfortunately, the diploid structure of humans makes it very expensive to collect haplotype data directly to display the recombination events. In a large-scale

sequencing project, genotype data instead of haplotype data are collected. However, haplotype data are required in many genetic marker applications, such as linkage disequilibrium analysis and disease association mapping to name a few [12,13]. Therefore, combinatorial algorithms and statistical methods to reconstruct haplotypes from genotypes (*i.e.* the haplotype phasing or inference problem) are urgently needed.

The input data for this problem can be SNP fragments from an individual, genotype data in a population or genotype data in a family [8,9,10,11,15]. There are many combinatorial [1,2,14,16] and statistical ways [11,19] of tackling the phasing problem. They are usually quite computationally demanding.

Some of the commonly used combinatorial methods [1,2,14,16] take advantage of the availability of pedigree data. In other words, given a pedigree and the genotype information, they reconstruct a haplotype configuration for each individual in the pedigree by trying to solve the Minimum Recombinant Haplotype Configuration (MRHC) problem [1]. During the process of reconstruction, the minimum recombinant criterion is used as the objective function. Because this objective attempts to reduce the number of candidate haplotype configurations, it naturally preserves common haplotype structures.

All the existing methods to the MRHC problem are time and space consuming for realistic applications. For example, a Pentium IV computer with 256MB RAM is used to solve MRHC on a input pedigree with 29 members and 51 SNP markers. An effective combinatorial algorithm ILP takes about 5 hours to find an exact solution, whereas a well-known statistical approach SimWalk2 takes even more than 6 days to find a haplotype configuration with the maximum likelihood [21]. While over 5 millions of SNPs have been identified in the public database dbSNP [17], there is a great need for efficient algorithms that could scale up to the whole genome level. This difficulty motivates us to analyze the hardness and approximability of MRHC problems from a theoretical point of view.

1.1 Formal definition of the MRHC problem

In this subsection, we give a formal definition of the MRHC problem as well as the issue of pedigree representation and biological background. We follow the conventions in [1].

Definition 1. *A pedigree graph is a connected directed acyclic graph (DAG) $G=\{V, E\}$, where $V= M \cup F \cup N$, M represents the male nodes, F represents the female nodes, N represents the matting nodes, and $E= \{ (u, v): u \in M \cup F$ and $v \in N$ or $u \in N$ and $v \in M \cup F\}$. $M \cup F$ is called individual nodes. The in-degree of each individual node is at most one. The in-degree of a mating node must be two, with one edge starting from a male (called the father) node and the other edge from a female node (called the mother) and the out-degree of a mating node must be larger than zero.*

In a pedigree, the individual nodes outgoing from a mating node are called the *children*. The individual nodes with zero in-degree are called the *founders*. The induced subgraph by the father, the mother and one child adjacent to the same mating node is called a *family trio*. If there are two node-disjoint paths between two mating

nodes in the pedigree graph, this pedigree has a *mating loop*. A pedigree without mating loops is called a *tree pedigree*. A pedigree where each member has at most one mate and at most one child looks like a binary tree, so this kind of pedigree is called a *binary-tree pedigree*. Fig. 1 demonstrates an example pedigree drawn in both the formal and conventional ways. In the conventional way, the mating nodes are omitted. For convenience, we use conventional drawings of pedigrees throughout this paper.

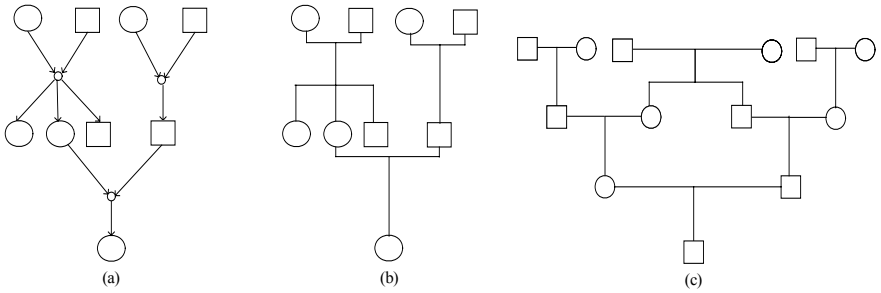


Fig. 1. (a) A pedigree drawn in the formal way. (b) The pedigree drawn in the conventional way. (c) A pedigree with a mating loop

A *genetic marker* is a short non-redundant discriminative DNA sequence that can be used to trace inheritance. Some common genetic markers are microsatellite DNA sequences or SNP data. Each polymorphism state of a genetic marker is called an *allele*. Different kinds of markers have different numbers of alleles. For instance, a microsatellite marker has multiple possible alleles occurring at a locus, which is called *multi-allelic*. An SNP marker commonly has only two possible alleles occurring at a locus, which is called *bi-allelic*. We will mostly be interested in bi-allelic markers because they are becoming the most popular markers in practice. Bi-alleles can be in exactly one of the two alternative states, such as 1 or 2. If an allele is missing at some locus, it is denoted as a "*".

In diploid organisms, because chromosomes come in pairs, at each locus there is a pair of alleles, which is referred to the *genotype* of this locus. If these alleles are the same, the genotype at this locus is *homozygous*; otherwise, the genotype is *heterozygous*. The alleles on the same chromosome form a *haplotype*. Each individual has a pair of haplotypes.

If there is no genetic mutation in a meiosis process, the child inherits one haplotype from the mother and the other one from the farther. This is the well-known *Mendelian law of inheritance.* The haplotype inherited from the mother is called the *maternal haplotype* while the one from the father is called the *paternal haplotype*. Given a pair of haplotypes of an individual, if it is known which one was inherited from his (or her) father and which was from his (or her) mother, the haplotypes and the inheritance information together are called a *haplotype configuration* (*i.e.* a *configuration* in short); otherwise, the haplotypes without inheritance information form a *haplotype grouping* (*i.e.* a *grouping* in short).

Usually, an entire haplotype of the mother's (or father's) haplotype pair is passed onto the child during meiosis. However, crossover between the haplotype pair might

occur, where the haplotype pair gets shuffled and one of the mixed haplotypes is passed onto the child. This crossover is called a recombinant.

A PS (or phase) value represents the paternal or maternal information about the alleles at a locus. The PS value can take the values 0 or 1, where 1 means that the allele with the smaller identification number is from the mother and the allele with the larger identification number is from the father, and 0 otherwise. Thus, the reconstruction of haplotype configuration for an input pedigree can be viewed as assigning PS values to each locus of every member of the pedigree.

We give the mathematical notations of the concepts mentioned above for convenience (see the appendix). Now, the MRHC problem is defined as follows:

Definition 2 (MRHC [1])**.** *Given a pedigree and genotype information for each member of the pedigree, find a haplotype configuration of the pedigree that obeys the Mendelian law of inheritance and requires the minimum number of recombinants.*

1.2 Variants of MRHC and some related problems

We give the definitions of the variants of MRHC and list the related problems that are going to be discussed later in the paper.

Definition 3. *MRHC(k, j) is defined the same as MRHC except that each member in the pedigree has at most k mates and at most j children with each mate. Binary-tree-MRHC is defined as MRHC on a binary-tree pedigree. Binary-tree -MRHC* is defined the same as binary-tree-MRHC except it is allowed to have missing alleles. 2-locus-MRHC is MRHC on a two-locus pedigree without missing data. 2-locus-MRHC* is defined the same as 2-locus-MRHC except it is allowed to have missing data. Tree-MRHC is MRHC on a pedigree without mating loops or missing data.*

In order to discuss the hardness and approximation of the variants, we are going to make use of some related problems or properties, such as the Min UnCut [5] (*i.e.* 2-Linear-Equations Mod 2 [4]), Min UnCut(*k*) (the same definition as Min UnCut except that each variable occurs at most *k* times), Min 2CNF Deletion [4, 5] problems, consistency and satisfiablility property (see the appendix). The Min UnCut and Min 2CNF Deletion problems are known to be NP-hard [5]. We will show that the Min UnCut(*k*) problem is also NP-hard in this paper.

For any NP-hard minimization (or maximization) problem, if there is some polynomial time algorithm to give a solution with the objective value no more (or less, respectively) than $f(n)\cdot$OPT (or OPT/$f(n)$, respectively), where $f(n)$ can be any function of the input size n, the problem can be approximated within ratio $f(n)$; otherwise, the problem cannot be approximated.

1.3 Previous complexity results on MRHC

Qian and Bechmann proposed a ruled-based algorithm to reconstruct haplotype configurations based on six rules [16]. Their algorithm is a heuristic without theoretical analysis. Li and Jiang first proved that MRHC on two-locus pedigree is NP-hard [1]. Doi, Li and Jiang further proved that MRHC on tree pedigrees is also NP-hard in the general case [2], even though MRHC can be solved by dynamic programming algorithms when the number of members or loci in the input pedigree is bounded by a constant. However, the NP-hardness proof requires pedigrees containing individuals with an unbounded number of mates or children. It was left as an open question if the proof can be improved to work for tree pedigrees where every individual has a bounded number of mates and children.

Consistency checking of the Mendelian law of inheritance (*i.e.* the Mendelian law checking problem) is closely related to the MRHC problem. The purpose of Mendelian law checking is to determine whether the given genotype data obey the classic Mendelian law of inheritance. Mendelian law checking usually needs to be done ahead of phasing haplotype configurations. Aceto *et al.* showed that the Mendelian law checking problem is NP-hard in general, although checking the consistency on pedigrees with bi-allelic data or with no mating loops [3] can be done in polynomial time.

In this paper, we consider a simple variant of MRHC, which involves pedigrees with members that has at most one mate and one child (*i.e.* binary-tree-MRHC). It is an open question if binary-tree MRHC is NP-hard. A polynomial-time algorithm for it, if exists, could be useful for solving the general-case MRHC problem. Another important question is whether a good approximation algorithm exists for MRHC. Here, in terms of computing the minimum-recombinant haplotype, the accuracy is sacrificed to improve the efficiency. Previously, there is no known polynomial-time approximation algorithm for MRHC with guaranteed ratio.

Table 1. The known hardness results of the Mendelian law checking and MRHC problems

Problem \ Pedigree	Loop?	Multi-allelic?	Unbounded number of loci?	Unbouned number of members?	Hardness
Mendelian law checking	Yes	Yes			NP-hard [3]
		No			P [3]
	No				P [3]
MRHC	Yes	No	No	Yes	NP-hard [1]
	No	No	No	Yes	P [2]
	No	No	Yes	No	P [2]
	No	No	Yes	Yes	NP-hard [2]

1.4 Our results

We will consider pedigrees with bi-allelic genotype data throughout this paper. First, we reduce $\neq$3SAT to the binary-tree-MRHC problem and show that this problem is NP-hard, which answers an open question in [2]. Second, we study the approximability of MRHC on pedigree data with the following restrictions: (*I*) 2-locus genotype data with missing alleles, (*II*) binary tree pedigrees with missing alleles, (*III*) 2-locus genotype data without missing alleles, and (*IV*) tree pedigrees without missing alleles. These four restricted cases of MRHC are NP-hard problems shown

either in the literature [1,2] or in this paper. We demonstrate that for MRHC in the former two cases *I* and *II* cannot be approximated unless P = NP. We also prove that it is NP-hard to approximate problems *III* and *IV* within any constant ratio under the Unique Games Conjecture [4]. Moreover, we show that problem *III* can be approximated with ratio $O(\sqrt{\log(n)})$ in polynomial time by reducing it to the Min 2CNF Deletion problem, Finally, we discuss the approximation of MRHC on pedigrees where each member has a bounded number of children and mates, mirroring pedigrees in real applications.

1.5 Organization of the paper

The paper is organized as follows. We briefly give definitions of the MRHC problem and other closely related problems, introduce the related biological background in section 1. We prove that binary-tree-MRHC is NP-hard and state the approximatability of MRHC on pedigrees with missing data in section 2. We show the approximation lower bound of MRHC on pedigrees without missing data and the approximation upper bound of 2-locus-MRHC in section 3. In section 4, we tentatively explore the approximation hardness of MRHC on the pedigrees where each member has a bounded number of mates and children. We organize our hardness results and conclude this paper with a few remarks in section 5. Due to space limitation, some of the proofs are omitted in the main text and given in the appendix.

2 Approximation of MRHC on pedigrees with missing data

In this section, we prove the hardness of approximating MRHC on pedigree data with missing alleles. Two variants are considered.

Lemma 1. *If it is NP-hard to decide whether OPT(R)=0 for a minimization problem R, R cannot be approximated unless P=NP.*

Proof: See the appendix. □

2.1 Hardness and approximation of binary-tree-MHRC(*)

Theorem 2. *Binary-tree-MRHC is NP-hard.*

Proof: See the appendix. □

Theorem 3. *It is NP-hard to decide whether OPT(binary-tree-MRHC*)=0.*

Proof: See the appendix. □

Corollary 4. *Binary-tree- MRHC* cannot be approximated unless P=NP.*

Proof: It follows obviously from Lemma 1 and Theorem 3. □

2.2 Approximation of 2-loop-MHRC*

Theorem 5. *It is NP-hard to decide whether OPT(2-locus-MRHC*)=0*

Proof: See the appendix. □

Corollary 6. *2-locus-MRHC* cannot be approximated unless P=NP.*

Proof: This follows immediately from Lemma 1 and Theorem 5. □

3 Approximation of MRHC on pedigrees without missing data

In this section, we consider the approximability of the same variants of MRHC without missing data. In order to show the negative result, we need to use some gap-introducing reduction (or gap-preserving reduction) for MRHC. We will use the concept of *L-reduction* proposed by Papadimitriou and Yannakakis [18].

3.1 Approximation of tree-MRHC

Lemma 7. *There is an L-reduction from Min UnCut to tree-MRHC that transforms a set of Boolean constraints φ to a tree pedigree ξ such that:*
(i) $OPT_{Min\ UnCut}(\varphi) = OPT_{tree\text{-}MRHC}(\xi)$, and
(ii) Given a haplotype solution for ξ with k recombinants, we can construct a solution for φ with at most k unsatisfied clauses.

Proof: See the appendix. □

Theorem 8. *It is NP-hard to approximate tree-MRHC within any constant ratio under the Unique Games Conjecture* [4].

Proof: It is known NP-hard to approximate the Min UnCut problem within any constant ratio under the Unique Games Conjecture [4]. The property of L-reduction in Lemma 7 guarantees the NP-hardness of approximating tree-MRHC. □

3.2 Approximation of 2-locus-MRHC

We will present a lower bound and an upper bound on the approximation ratio for the 2-locus-MRHC problem.

3.2.1 Negative result for approximating 2-locus-MRHC

Lemma 9. *There is a polynomial-time L-reduction from Min UnCut to 2-locus-MRHC that transforms a Boolean constraints set φ to a pedigree ξ such that*

(i) $OPT_{Min\ UnCut}(\varphi) = OPT_{2\text{-}locus\text{-}MRHC}(\xi)$, and

(ii) Given any haplotype solution for ξ with k recombinants, we can find in polynomial time a truth assignment for φ with at most k unsatisfied constraints.

Proof: See the appendix. □

Theorem 10. *It is NP-hard to approximate 2-locus-MRHC within any constant ratio under the Unique Games Conjecture* [4].

Proof: The result follows from the above lemma and the fact that Min UnCut has no constant ratio approximation unless P = NP under the Unique Games Conjecture [4].□

3.2.2 Positive result for approximating 2-locus-MRHC

We first would like to reduce an instance of 2-locus-MRHC so that each member of the pedigree can be described by one Boolean variable. Since only two loci are involved, there are three types of members in a pedigree: (*I*) both loci are homozygous, (*II)* one locus is homozygous, and (*III)* both loci are heterozygous. A type *I* (or *II*) member has a fixed haplotype grouping. A type *III* member has a variable haplotype grouping.

Agarwal and Charikar recently presented a randomized polynomial-time $O(\sqrt{\log(n)})$ approximation algorithm for the Min 2CNF Deletion problem [5], where *n* is the number of variables in the input 2CNF constraints.

Lemma 11. *There is a randomized polynomial-time $O(\sqrt{\log(n)})$ approximation algorithm for 2-locus-MRHC, where n is the number of members in the input pedigree.*

Proof: The main idea of the proof is to transform an instance ξ of the 2-locus-MRHC problem to an instance φ of Min 2CNF Deletion whose solution corresponds to a haplotype solution of ξ in an equivalent way (in terms of cost). Moreover, the number of variable in φ is no more than the number of members in ξ. This would give rise to an L-reduction from 2-locus-MRHC to Min 2CNF Deletion, which has an $O(\sqrt{\log(n)})$ approximation algorithm. In the reduction, we represent the haplotype grouping of each type *III* member of pedigree ξ as a Boolean variable. Clauses of size two are then used to capture all grouping combinations in a parents-child trio that would cause recombinants.

Without loss of generality, we represent grouping HG_{TRUE} by TRUE and another grouping HG_{FALSE} by FALSE in the same way as in Theorem 5. We use the following table to show how to map each parents-child trio to a clause so that the number of recombinants is translated to the number of unsatisified clauses.

Note that, if both parents of a trio are members of types *I* or *II*, the trio cannot incur any recombinants, although we need check the Mendelian consistency of the genotypes in the trio. Hence, such trios are omitted in Table 2. Let φ denote the 2CNF constraints consisting of all the clauses created above. Then it is easy to see that there is a one-to-one correspondence between recombinants in the pedigree and unsatisfied

clauses, and the reduction is in fact an L-reduction. Hence, we have a randomized polynomial-time $O(\sqrt{\log(n)})$ approximation algorithm for 2-locus-MRHC.

Table 2. Mapping a parents-child trio to clauses (Here, alleles X and Y can be either 1 or 2, and X are Y are different.)

Genotype of the Mother (***A***)	Genotype of the Father (***B***)	Genotype of the Child (***C***)	2CNF Constraint
1 2 1 2	1 2 1 2	1 1 2 2 1 1 2 2	$2(A \vee B)$ $(A \vee \overline{B})$ $(\overline{A} \vee B)$
		2 2 1 1 1 1 2 2	$2(\overline{A} \vee \overline{B})$ $(A \vee \overline{B})$ $(\overline{A} \vee B)$
		2 2 1 1 1 2 1 2 1 2 1 2 2 2 1 1	$(\overline{A} \vee \overline{B})$ $(A \vee B)$
		1 2 1 2	$(\overline{A} \vee C)$ $(A \vee \overline{C})$ $(\overline{B} \vee C)$ $(B \vee \overline{C})$
1 2 1 2	X X Y X Y X X X	1 1 2 2 1 1 2 2	A
		2 2 1 1 1 1 2 2	$\overline{A}$
		1 2 1 2	$(\overline{A} \vee C)$ $(A \vee \overline{C})$
	X X Y X	Y X X X	A
		Y X Y Y	$\overline{A}$
	Y X X X	X X X Y	A
		Y Y X Y	$\overline{A}$

□

Observe that the results in this section show that, in terms of approximability, the 2-locus-MRHC problem is easier than the Min 2CNF Deletion problem and harder than the Min UnCut problem. Also, Lemma 9 presents an alternative proof that 2-locus-MRHC is NP-hard, which is much easier to understand than the original proof in [1].

4 Approximation of MRHC(*k*, *j*)

The proof of Lemma 7 uses a pedigree that contains members with a variable number of children, although every member in the pedigree has only one mate. Can we get the same hardness result for tree-MRHC if we bound the number of mates instead of the number of children? In addition, the pedigrees in the proofs of Theorem 5 and Lemma 9 contain members with a variable number of children or mates. Another question is whether MRHC on two-locus pedigrees with a bounded number of children and mates leads to the same hardness result. In this section, we discuss the approximation of MRHC on pedigrees with bounded number of children and mates. For the convenience of comparison, we state strengthened versions of the previous theorems in the order they appear in this paper. We use *u* to present an integer variable.

First, we refine Theorem 5. The hardness result in this theorem holds for 2-locus-MRHC(*u*,1), because some member might appear in every clause gadget and every member has at most one child in the proof of Theorem 5.

Theorem 12. *2-locus-MRHC*(4,1) cannot be approximated unless P=NP.*

Proof: According to the L-reduction from Max 3SAT to Max 3SAT(29) and the L-reduction from Max 3SAT(29) to MAX 3SAT(3) [6], deciding whether there is a satisfiable solution for 3SAT(3) is NP-hard. In the proof of Theorem 5, rather than reducing 3SAT to 2-locus-MRHC*, we can reduce 3SAT(3) to 2-locus-MRHC*. Then, in the pedigree constructed, each member has at most four mates (*i.e.* one mate in the variable gadgets and at most three mates in the clause gadgets). The construction of the pedigree introduces at most one child for each member with each mate. Hence, Theorem 5 can be tightened to 2-locus-MRHC*(4,1)). □

Next, let us look at Lemma 7. This lemma actually works for tree-MRHC(1,*u*). It is natural to consider tree-MRHC on pedigrees where members have a bounded number of children with each mate. In order to decrease the number of children and mates in the pedigree, we need a bounded version of Min UnCut like the one for Max 3SAT.

In fact, there is an L-reduction from Min UnCut to Min UnCut(15) that transforms a Boolean constraints set φ to another Boolean constraints set ψ such that

(*i*) $OPT_{UnCut}(\varphi) = OPT_{UnCut(15)}(\psi)$, and

(*ii*) Given any truth assignment for ψ with k unsatisfied constraints, we can find in polynomial time a truth assignment for φ with at most k unsatisified constraints.

This L-reduction from Min UnCut to Min UnCut(15) can be constructed using the same idea as the L-reduction that transforms Max 3SAT to Max 3SAT(29) in [6] with just a few minor modifications. The details of this L-reduction are omitted here. Based on the property of this L-reduction, we know that it is NP-hard to approximate Min UnCut(15) within any constant ratio under the Unique Games Conjecture [4].

Theorem 13. *It is NP-hard to approximate tree-MRHC(u,1) within any constant ratio under the Unique Games Conjecture [4].*

Proof: See the appendix. □

Finally, we consider Lemma 9. The hardness result actually holds for 2-locus-MRHC(*u*, *u*), because neither the number of mates nor the number of children for a member is bounded by any constant.

Theorem 14. *It is NP-hard to approximate 2-locus-MRHC(16,15) within any constant ratio under the Unique Games Conjecture [4].*

Proof: Similar to the proof of Lemma 9. But we reduce from Min UnCut(15) instead of Min UnCut. Each member in the constructed pedigree has at most 16 mates (*i.e.* one mate in the variable gadgets and at most 15 mates in the clause gadgets) and 15 children with each mate. The details are omitted here. □

5 Discussion and Conclusion

The results presented in this paper are organized in Table 3. First, we showed that binary-tree-MRHC is NP-hard. Binary-tree-MRHC is a simplest variant of MRHC

because one mate and one child are the minimum requirement to express the inheritance of human beings. Second, we showed some approximability results concerning the MRHC problem. With the presence of missing data, it is NP-hard to tell if an instance of 2-locus-MRHC* and binary-tree-MRHC* requires any recombinant. This gives an interesting contrast to the results in [1] where the problem of finding a zero-recombinant haplotype solution for MRHC was shown to be solvable in polynomial time. This result also implies that 2-locus-MRHC* and binary-tree-MRHC* is not approximable in polynomial time. Without the presence of missing data, 2-locus-MRHC can be approximated with the ratio $O(\sqrt{\log(n)})$. In addition, it is NP-hard to approximate 2-locus-MRHC and tree-MRHC within any constant ratio under the Unique Games Conjecture [4]. Our final results concern the inapproximability of MRHC on pedigrees where each member has a bounded number of mates and/or a bounded number of children with each mate.

Table 3. Our hardness and approximation results for MRHC with bi-alleles

	Loop ?	Miss-ing data?	Unbounded Number of loci?	Unbounded number of members?	Hardness	Lower bound of approx. ratio	Assumption	The lower bound holds for	Upper bound of approx. ratio
Binary-tree-MRHC	No	No	Yes	Yes	NP				
2-locus-MRHC*	Yes	Yes	No	Yes		Any f(n)	$P \neq NP$	2-locus-MRHC* (4,1)	
Binary-tree-MRHC*	No	Yes	Yes	Yes		Any f(n)	$P \neq NP$	Binary-tree-MRHC*	
2-locus-MRHC	Yes	No	No	Yes		Any constant	$P \neq NP$, the Unique Games Conjecture	2-locus-MRHC (16,15)	$O(\sqrt{\log(n)})$
Tree-MRHC	No	No	Yes	Yes		Any constant	$P \neq NP$, the Unique Games Conjecture	Tree-MRHC(1,u) Tree-MRHC(u,1)	

Acknowledgement

We would like to thank Dr. Neal Young and Dr. Marek Chrobak for their valuable suggestions and discussion.

References

[1] J. Li and T. Jiang. Efficient rule-based haplotyping algorithm for pedigree data. *Proc. of the 7th Annual Conference on Research in Computational Molecular Biology (RECOMB'03)*, pages 197-206, 2003.

[2] K. Doi, J. Li and T. Jiang. Minimum recombinant haplotype configuration on tree pedigrees. *Proc. of the 3rd Annual Workshop on Algorithms in Bioinformatics (WABI'03)*, pages 339-353, 2003.

[3] L. Aceto *et al*. The complexity of checking consistency of pedigree information and related problems. *J. Comp. Sci. Tech.*, 19(1): 42–59, 2004.

[4] S. Khot. On the power of 2-Prover 1-Round Games. *Proc. of the 34th ACM Symposium on Theory of Computing (STOC'02)*, pages 767-775, 2002.

[5] A. Agarwal, M. Charikar. $O(\sqrt{\log(n)})$ approximation algorithms for min UnCut, min 2CNF deletion, and directed cut problems. *Proc. STOC'05,* pages 573-581, 2005.

[6] G. Ausiello *et al*. *Complexity and approximation: combinatorial optimization problems and their approximability properties*, pages 276-279, Springer, 1999.

[7] L. Jorde. Where we are hot, they are not. *Science*, Volume 308, pages 60-62, 2005.

[8] L. Li, J.H. Kim, and M. S. Waterman. Haplotype reconstruction from SNP alignment. *Proc. RECOMB'03*, pages 207-216, 2003.

[9] R. Lippert, *et al*. Algorithmic strategies for the single nucleotide polymorphism haplotype assembly problem. *Briefings in Bioinformatics* 3(1): 23-31, 2002.

[10] E. Eskin E. Halperin, and R.M. Karp. Large scale reconstruction of haplotypes from genotype data. *Proc. RECOMB'03,* pages 104-113, 2003.

[11] L. Excoffier and M. Slatkin. Maximum–likelihood estimation of molecular haplotype frequencies in a diploid population. *Mol. Biol. Evol.*, 12: 921-927, 1995.

[12] H. Seltman, K. Roeder, and B. Devlin. Transmission/disequilibrium test meets measured haplotype analysis: family-based association analysis guided by evolution of haplotypes. *Am. J . Hum. Genet.*, 68(5): 1250-1263, 2001.

[13] S. Zhang *et al*. Transmission/ disequilibrium test based on haplotype sharing for tightly linked markers. *Am. J. Hum. Genet.*, 73(3): 556-579, 2003.

[14] J. Li and T. Jiang. An exact solution for finding minimum recombinant haplotype configurations on pedigrees with missing data by integer linear programming. *Proc. RECOMB'04*, pages 20-29, 2004.

[15] J.R. O'Connell. Zero-recombinant haplotyping: applications to fine mapping using SNPs. *Genet. Epidemiol.*, 19 Suppl. 1: S64-70, 2000.

[16] D. Qian and L. Beckmann. Minimum-recombinant haplotyping in pedigrees. *Am. J. Hum. Genet.*, 70(6): 1434-1445, 2002.

[17] The International HapMap Consortium. The International HapMap Project. *Nature*, Volume 426, pages 789-796, December 2003.

[18] C.H. Papadimitriou and M. Yannakakis. Optimization, Approximation, and Complexity Classes. *J. Comp. System Sci.*, pages 425-440, 1991.

[19] M. Stephens, N. J. Smith, and P. Donnelly. A new statistical method for haplotype reconstruction from population data. *Am. J. Hum. Genet.*, 68(4):978–989, 2001.

[20] T.J. Schaefer. The complexity of satisfiability problems. *Proc. of the 10th STOC*, pages 216-226, 1978.

[21] J. Li and T. Jiang. Computing the Minimum Recombinant Haplotype Configuration from incomplete genotype data on a pedigree by integer linear programming. *Proc. RECOMB'04*, pages 20-29, 2004.

Appendix

Mathematical notations and some related definition

We use the following mathematical notations to represent the critical concepts mentioned in section 1.1. Let n denote the number of loci, and $a_i, b_i, c_i, d_i, e_i, f_i \in \{1,2, *\}$, $1 \le i \le n$. A vector is defined as $(a_1,a_2,\ldots,a_n)^T$. $\vec{A}$ and $\vec{B}, \vec{C}, \vec{D}, \vec{E}, \vec{F}$ are defined similarly. We represent a genotype, haplotype, configuration and haplotype grouping by the following forms $G(\vec{A},\vec{B}), H(\vec{A}), HC(\vec{A},\vec{B})$ and $HG(\vec{A},\vec{B})$ respectively.

$$G(\vec{A},\vec{B}) = \begin{Bmatrix} a_1 & b_1 \\ \ldots & \ldots \\ a_n & b_n \end{Bmatrix} \quad H(\vec{A}) = \begin{bmatrix} a_1 \\ \ldots \\ a_n \end{bmatrix} \quad HC(\vec{A},\vec{B}) = \begin{bmatrix} a_1 & | & b_1 \\ \ldots & | & \ldots \\ a_n & | & b_n \end{bmatrix} \quad HG(\vec{A},\vec{B}) = \begin{bmatrix} a_1 & / & b_1 \\ \ldots & / & \ldots \\ a_n & / & b_n \end{bmatrix}.$$

For an individual, the *genotype* is denoted by a vector of allele pairs at every locus with the form $G(\vec{A},\vec{B})$; the *haplotype* is denoted by a vector of alleles at every loci with the form $H(\vec{A})$; the *haplotype configuration* is denoted by an ordered haplotype pair with the form $HC(\vec{A},\vec{B})$, where the former haplotype comes from the mother and the latter one comes from the father; the *haplotype grouping* is denoted by an unordered haplotype pair with the form $HG(\vec{A},\vec{B})$.

For the Min 2CNF Deletion [4, 5], Min UnCut [5], ≠3SAT [20] and other satisfiablility problems, we usually need to check the following two properties to make sure that a truth assignment is a feasible solution. The following term "satisfiability" means different things in different problems. For instance, in the Min 2CNF Deletion and Min UnCut problems, a constraint is satisfied if it is true; but in the ≠3SAT problem, a clause is satisfied if its three literatures do not have the same value.

Definition 4. *Given a variable x_i, any occurrence of x_i and $\overline{x_i}$ have different values; all occurrences of x_i (or $\overline{x_i}$) must have the same values (i.e. consistency property). Given a constraint, clause or monomial, it is satisfied (i.e. satisfiablility property)*

Proof of Lemma 1: This lemma is a folklore. We prove it for the completeness of the paper. Suppose that there is an approximation algorithm *A* with ratio *f(n)*, where *f(n)* is a function of input size n. On any instance, algorithm *A* can generate a solution *S* with cost smaller than $f(n)\cdot OPT(R)$ in polynomial time. If $OPT(R)=0$, solution *S* has cost *0*. Otherwise, solution *S* has a positive cost. This gives rise to a polynomial time algorithm to decide whether $OPT(R)=0$, which contradicts the assumption that it is NP-hard to decide whether $OPT(R)=0$. This Lemma holds. □

Proof of Theorem 2: We reduce ≠3SAT to binary-tree-MRHC. Given an instance of ≠3SAT, we assume that the Boolean formula φ has *m* clauses and each clause has three literals. In order to construct an instance of binary-tree-MRHC to enforce the consistency and satisfiability properties of a assignment to the literals in φ, we consider all pairs of literals in φ for each variable x_k (for consistency checking) and all pairs of literals (x_p , x_q), (x_p , x_s) and (x_q , x_s) for each clause $x_p \vee x_q \vee x_s$ of φ (for satisfiability checking). Suppose that *t* pairs are required for consistency checking. Clearly, *t* is smaller than m^2. We check exactly *3m* literals pairs for the satisfiability property. These pairs of literals are indexed sequentially starting from 1.

We construct a pedigree with genotype data consisting of *6m+1* loci, indexed from 1 through *6m+1*. First, we introduce an individual node M_0. Each even-numbered locus of M_0 has a heterozygous genotype {1 2} and corresponds to a literal in the formula φ. We observe that an assignment of PS values to the even-numbered loci in M_0 naturally divides the loci (and thus the literals) into two groups, which also bipartitions the corresponding literals in φ. The literals in one group can be considered to have value TRUE and the others FALSE. Therefore, the PS values of the even-numbered loci in M_0 encode an assignment to the literals in the formula φ. The constructed pedigree is illustrated in Fig. 2. This construction can be done in $O(m^3)$ time. Satisfiability checking or consistency checking involving complementary literals is called type *I* checking; it is type *II* checking, otherwise.

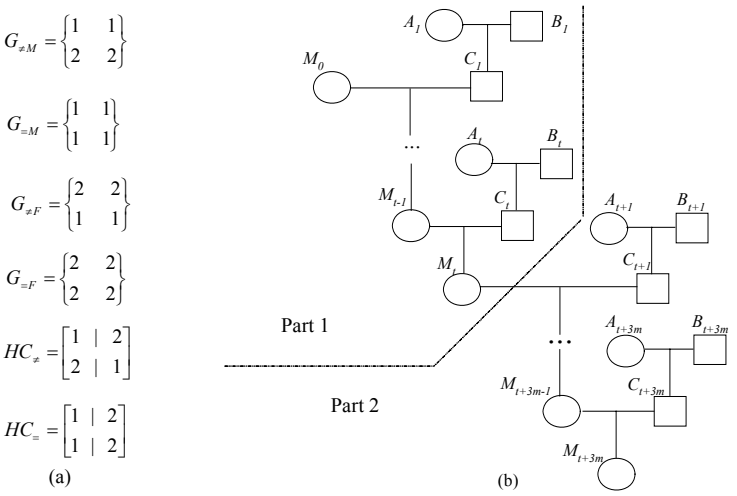


Fig. 2. (a) $G_{\neq M}$, $G_{=M}$, $G_{\neq F}$, $G_{=F}$, $HC_{\neq}$, and $HC_{=}$. (b) The pedigree used in the reduction from ≠3SAT to binary-tree-MRHC

Table 4. How C_i's configuration works in the reduction.

C_i's config.	C_i, M_{i-1} and M_i's configuration (or grouping)	
	at least 1 recombinant	0 recombinant
$\begin{matrix} 1 & 2 \\ 2 & 1 \end{matrix}$ (*i.e.* $HC_{\neq}$)	M_{i-1}: $\begin{matrix} 1/2 \\ 1/2 \end{matrix}$, C_i: $\begin{matrix} 1 & 2 \\ 2 & 1 \end{matrix}$, M_i: $\begin{matrix} 1/2 \\ 1/2 \end{matrix}$ (a)	M_{i-1}: $\begin{matrix} 1/2 \\ 1/2 \end{matrix}$, C_i: $\begin{matrix} 1 & 2 \\ 2 & 1 \end{matrix}$, M_i: $\begin{matrix} 1/2 \\ 2/1 \end{matrix}$ (b)	M_{i-1}: $\begin{matrix} 1/2 \\ 2/1 \end{matrix}$, C_i: $\begin{matrix} 1 & 2 \\ 2 & 1 \end{matrix}$, M_i: $\begin{matrix} 1/2 \\ 2/1 \end{matrix}$ (c)
$\begin{matrix} 1 & 2 \\ 1 & 2 \end{matrix}$ (*i.e.* $HC_{=}$)	M_{i-1}: $\begin{matrix} 1/2 \\ 2/1 \end{matrix}$, C_i: $\begin{matrix} 1 & 2 \\ 1 & 2 \end{matrix}$, M_i: $\begin{matrix} 1/2 \\ 1/2 \end{matrix}$ (d)	M_{i-1}: $\begin{matrix} 1/2 \\ 2/1 \end{matrix}$, C_i: $\begin{matrix} 1 & 2 \\ 1 & 2 \end{matrix}$, M_i: $\begin{matrix} 1/2 \\ 2/1 \end{matrix}$ (e)	M_{i-1}: $\begin{matrix} 1/2 \\ 1/2 \end{matrix}$, C_i: $\begin{matrix} 1 & 2 \\ 1 & 2 \end{matrix}$, M_i: $\begin{matrix} 1/2 \\ 1/2 \end{matrix}$ (f)

Suppose that the i^{th} pair of literals indexed in the above is (x_j, x_k). The two loci representing the occurrences of these two literals are denoted as the $j_o{}^{th}$ and $k_o{}^{th}$ loci. The pedigree has four types of individuals A_i, B_i, C_i and M_i. For convenience, we define four genotypes $G_{\neq M}$, $G_{=M}$, $G_{\neq F}$, $G_{=F}$ and two configurations $HC_{\neq}$, $HC_{=}$ in Fig. 2.

The genotype of each individual is constructed as follows:

- A_i: All odd-numbered loci have genotype {2 2} and all even-numbered loci have genotype {1 2} except the two loci j_o and k_o. The genotype of these two loci is $G_{\neq M}$ if the i^{th} literal pair is for type *I* checking; the genotype is $G_{=M}$ otherwise.
- B_i: All odd-numbered loci have genotype {2 2} and all even-numbered loci have genotype {1 2} except the two loci j_o and k_o. The genotype of these two loci is $G_{\neq F}$ if the i^{th} pair is for type *I* checking; the genotype is $G_{=F}$ otherwise.

- C_i: All odd-numbered loci have genotype {2 2} and all even-numbered loci have genotype {1 2}. Due to the genotype value of A_i and B_i, the two loci j_o and k_0 of C_i will be forced to have the haplotype configuration $HC_{\neq}$ if the i^{th} pair is for type I checking; they will be forced to have the configuration $HC_{=}$ otherwise.
- M_i: All loci have genotype {1 2}.

There are two main parts in the pedigree. Part 1 is used to enforce the consistency property of a truth assignment, while part 2 is used to verify the satisfiability property.

Part 1: For the i^{th} ($1 \le i \le t$) pair of literals indexed above, we introduce a gadget consisting of the parents-child trio A_i, B_i and C_i to do consistency checking. The two loci k_o and j_o standing for the occurrences of the i^{th} pair are shown in Table 4. If and only if these two loci in M_{i-1} have the consistent PS value, there can be zero recombinants in this parents-child trio. Therefore, if and only if all the variables are consistent, there can be zero recombinants in part 1.

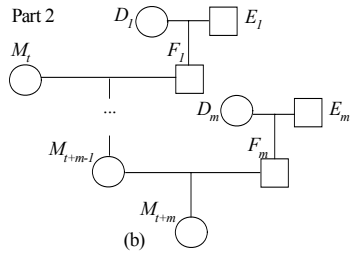
Part 2: For the j^{th} ($1 \le j \le m$) clause $x_p \vee x_q \vee x_s$, we need three adjacent trios A_{t+3j-r}, B_{t+3j-r}, C_{t+3j-r} ($0 \le r \le 2$) to verify its satisfiability. We check whether at least one of the three pairs (x_p , x_q), (x_p , x_s) and (x_q , x_s) has different PS values. Each trio is constructed in the same way as the above. If all three literals in a clause have the same truth value, the satisfiability checking in all three corresponding trios will fail and thus at least two recombinants will be required in the pedigree. Otherwise, exactly one checking fails and thus only one recombinant would be required (which takes place when a C_i passes its haplotypes to the M_i, $1 \le i \le t+3m$). The recombinants required by satisfibility checking for different clauses cannot be shared. For any haplotype configuration, there are at least m recombinants in part 2 and this bound can be reached if and only if all clauses can be satisfied by some truth assignment.

We claim that φ has a satisfiable assignment if and only if there are exactly m recombinants in the whole pedigree – zero in Part 1 and m in Part 2. The details are omitted here due to the space limitation. Thus binary-tree-MRHC is NP-hard. □

Proof of Theorem 3: Again, we reduce ≠3SAT to binary-tree-MRHC. Given an instance of ≠3SAT, assume that the Boolean formula φ has m clauses and each clause has three literals. We use a similar pedigree as the one in Theorem 2. The main structure of the two parts of the pedigree remains the same. However in Part 2, instead of using three parents-child trios to check the satisfiablility of the j^{th} ($1 \le j \le m$) clause, we use just one trio containing members D_j, E_j and F_j. We can force F_j's *(6j-4)th, (6j-2)th* and *(6j)th* loci to have configuration HC_{SAT} defined in Fig. 3, by using the same technique as the one Theorem 2 (*i.e.* setting D_j and E_j's genotype appropriately). This construction can still be done in $O(m^3)$ time.

$$HC_{SAT} = \begin{bmatrix} * & | & 1 \\ 1 & | & 2 \\ 2 & | & * \end{bmatrix}$$

(a)



(b)

Fig. 3. (a) HC_{SAT} (b) Part 2 of the constructed pedigree in the reduction

This modified part 2 of the new pedigree is illustrated in Fig. 3. For the j^{th} ($1 \le j \le m$) clause $x_p \vee x_q \vee x_s$, Part 2 contains a member F_j that can be used to check if all x_p, x_q and x_s have equal truth values. If the literals in this clause do not have the same value, we can always assign appropriate alleles to the missing data positions without causing recombinants in the parents-child trio containing M_{j-1}, F_j and M_j; otherwise, there must be at least one recombinant, for the arguments similar to those demonstrated in Table 4.

We claim that if and only if φ has a satisfying assignment, there is a zero-recombinant solution for the entire pedigree. The details are omitted here. This completes the proof. □

Proof of Theorem 5: We reduce 3SAT to 2-locus-MRHC*. Given an instance of 3SAT, we assume that the Boolean formula φ has n variables and m clauses, where each clause has three literals. We construct gadgets for each variable and clause and define a genotype G_0, two grouping HG_{TRUE} and HG_{FALSE}, a haplotype H_{TRUE} illustrated in Fig. 4. The logical value TRUE and FALSE are represented by two alternative haplotype groupings for the genotype G_0. Without loss of generality, we will represent TRUE by HG_{TRUE} and FALSE by HG_{FALSE}.

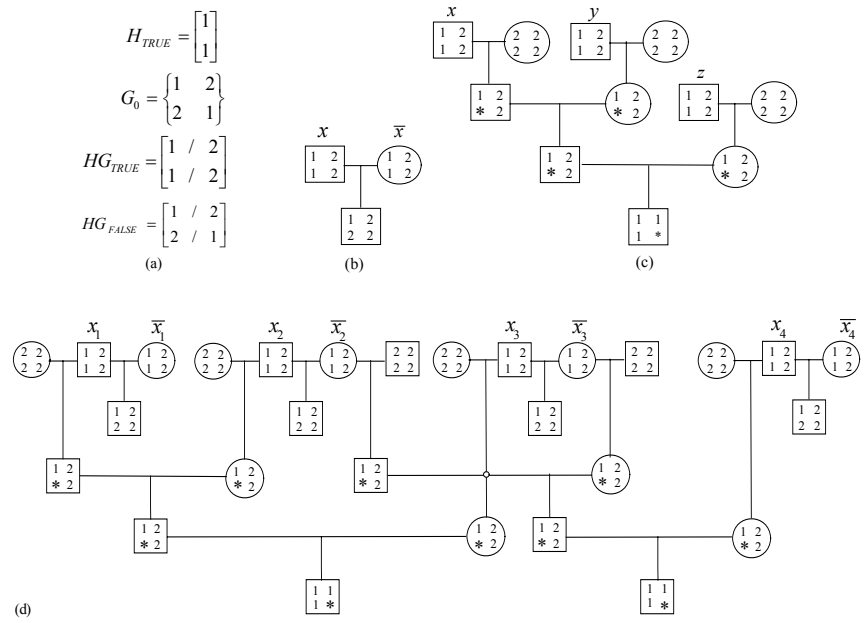


Fig. 4. (a) G_0, H_{TRUE}, HG_{TRUE} and HG_{FALSE}. (b) Gadget for variable x. (c) Gadget for clause (d) An example of the reduction from 3SAT to 2-locus-MRHC* for $(x_1 \vee x_2 \vee x_3) \wedge (\overline{x_2} \vee \overline{x_3} \vee x_4)$

We create a founder for all negative occurrences of each variable and a founder for all positive literals, and add appropriate offsprings according to the variable and clause gadgets in Fig. 4. In each variable gadget, we let the positive(or negative) literal be male (or female, respetively). In each clause gadget, we carefully set each variable member's mate be the complementary gender. By this means, we can

construct a pedigree in O(m+n) time. Fig. 4 also shows an example of the construction.

If and only if φ has a consistent assignment, there is a solution where no recombinant is needed in the variable gadgets. Moreover, if and only if some assignment has at least one literal with TRUE value in each clause, at least one founder has the grouping HG_{TRUE} in the corresponding clause gadget; so that the haplotype H_{TRUE} is passed through intermediate members to the last generation without incurring any recombinant. Then we claim that φ has a satisfiable truth assignment if and only if there is a zero-recombinant haplotype solution for the entire pedigree. This completes the proof. □

Proof of Lemma 7: To simplify the proof, let φ represent both an instance of Min UnCut (or 2CNF Deletion) on a set of constraints (or clauses, respectively) and the input constraint (clause) set, ξ represent both an instance 2-locus-MRHC and also the input pedigree.

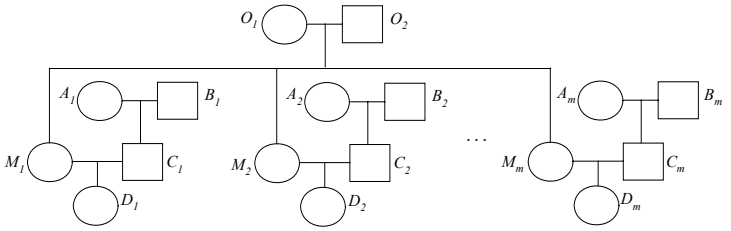


Fig. 5. The reduction from Min UnCut to tree-MRHC

The idea here is borrowed from an NP-hardness proof in Doi, Li and Jiang [2]. Given an instance of Min UnCut, we assume that the Boolean constraint set φ has n variables and m constraints. We construct a pedigree ξ with n loci, each of which represents a variable, as illustrated in Fig. 5. All loci in O_1, O_2, M_i, C_i and D_i($1 \le i \le m$) are heterozygous. Two haplotype configurations $HC_=$ and $HC_{\neq}$ are defined in the same way as in Theorem 2. For the i^{th} constraint of φ, if the constraint has the form $x_{j_1} \oplus x_{j_2} = 0$, C_i is be forced to have the configuration $HC_=$ at the $j_1{}^{th}$ and the $j_2{}^{th}$ loci. Otherwise, if the constraint has the form $x_{j_1} \oplus x_{j_2} = 1$, C_i is forced to have the configuration $HC_{\neq}$ at these two loci. All other loci of A_i and B_i are heterozygous.

Using the same arguments as the proofs in [2], it is easy to show this reduction is an L-reduction. The details are omitted here. □

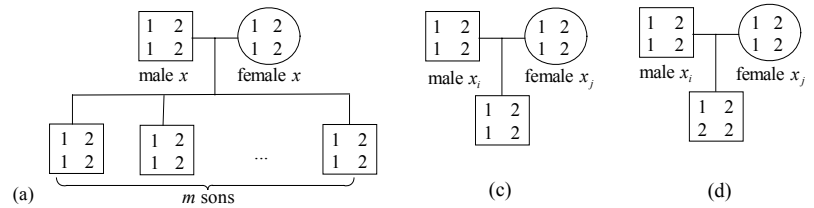


Fig. 6. (a) Gadget for variable x (b) Gadget for constraint $x_i \oplus x_j = 0$ (c) Gadget for constraint $x_i \oplus x_j = 1$

Proof of Lemma 9: Given an instance φ of Min UnCut, we assume that the Boolean constraint set has n variables and m constraints. We create a male member (M_x) and a female member (F_x) for each variable x as the founders, and add appropriate offsprings according to the variable and constraint gadgets shown in Fig. 6. We can construct the pedigree ξ for 2-locus-MRHC in O(mn) time. We denote TRUE by HG_{TRUE} and FALSE by HG_{FALSE} in the same way as in Theorem 5.

Observe that we can let M_x and F_x have the same haplotype grouping for each variable x and obtain a haplotyope solution S with at most m recombinants in the constraint gadgets. Moreover, we claim that it is always advantageous for each pair of members M_x and F_x to have the same haplotype grouping. This is because if M_x and F_x have different haplotype groupings in some haplotype solution, each of their sons in the corresponding variable gadget would incur a recombinant and thus the variable gadget would contain a total of m recombinants. Clearly, we will a better solution by changing the haplotype grouping of M_x or F_x.

It is easy to see that we can translate in polynomial time any haplotype solution for ξ with k recombinants to a solution for φ with at most k unsatisfied constraints. This correspondence trivially constitutes an L-reduction. □

Proof of Theorem 13: Given an instance φ of Min UnCut(15) with n variable and m constraints, we construct a pedigree ξ as illustrated in Fig. 7. We use the same notation and idea to construct every member as those in Theorem 2. All M_i's children are designed to have the same genotype as M_i. Members C_1, C_2, …, C_t are used to check the consistency of each variable. C_{t+j} ($1 \leq i \leq m$) is designed to check the satisfiability of the j^{th} constraint.

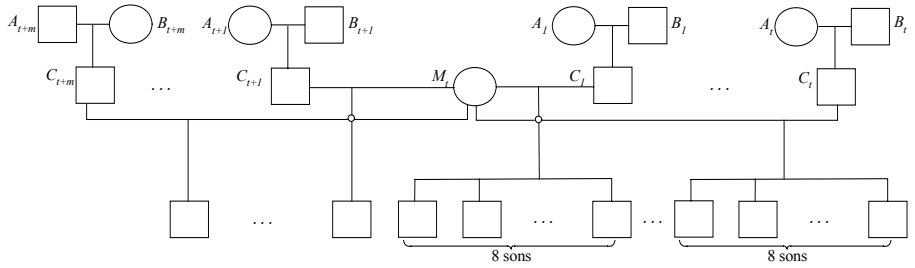


Fig. 7. The reduction from UnCut(15) to tree-MRHC(u,8).

Using the arguments similar to those in Lemma 9, we can prove that an optimal truth assignment for the Boolean constraints φ can be mapped to an optimal haplotype solution for the pedigree ξ with the same cost and vice versa. Given a haplotype solution for ξ with k recombinants, we can construct a solution for φ with at most k unsatisfied clauses easily. In other words, the above is an L-reduction. The details are omitted here. Hence, the approximability result holds tree-MRHC(u,8).

In the above reduction, tree-MRHC(u,8) can be further tightened to tree-MRHC(u,1). We can create 8 new mates with only one child with each mate to replace C_i ($1 \leq i \leq m$) and its 8 children. These 8 new mates are duplicates of the old C_i. It's easy to see that the approximability result still works for tree-MRHC(u,1). □