

# Improved Recombination Lower Bounds for Haplotype Data

Vineet Bafna and Vikas Bansal

*Dept. of Computer Science and Engineering, University of California at San Diego, La Jolla CA*

*email: {vbafna,vibansal}@cs.ucsd.edu*

**ABSTRACT** Recombination is an important evolutionary mechanism responsible for the genetic diversity in humans and other organisms. Recently, there has been extensive research on understanding the fine scale variation in recombination rates across the human genome using DNA polymorphism data. A combinatorial approach towards this is to estimate the minimum number of recombination events in any history of the sample. Recently, Myers and Griffiths [30] proposed two measures,  $R_h$  and  $R_s$ , that give lower bounds on the minimum number of recombination events. In this paper, we provide new and improved methods (both in terms of running time and ability to detect past recombination events) for computing lower bounds on the minimum number of recombination events. Our principal results include:

- We show that computing the lower bound  $R_h$  is NP-hard using a reduction from the minimum test collection problem [12]. We adapt the greedy algorithm for the set cover problem [24] to give a polynomial time algorithm for computing a diversity based bound, which we call  $R_g$ . This algorithm is several orders of magnitude faster than the Recmin program of Myers and Griffiths [30] and the bound  $R_g$  matches the bound  $R_h$  almost always.
- We also show that computing the lower bound  $R_s$  for a given matrix is also NP-hard using a reduction from MAX-2SAT. We give a  $O(m \cdot 2^n)$  time algorithm for exactly computing  $R_s$  for a dataset with  $n$  haplotypes and  $m$  SNP's. We propose a new bound  $R_I$  which extends the history based bound  $R_s$  using the notion of intermediate haplotypes. This bound detects more recombination events than both  $R_h$  and  $R_s$  bounds on many real datasets.
- We extend our algorithms for computing  $R_g$  and  $R_s$  to obtain lower bounds for haplotypes with missing data. These methods can detect more recombination events for the LPL dataset [32] than previous bounds and provide stronger evidence for the presence of a recombination hotspot.
- We apply our lower bounds methods to a real dataset [22] and demonstrate that these can provide a good indication for the presence and the location of *recombination hotspots*.

## 1 INTRODUCTION

Recombination is one of the major evolutionary mechanisms responsible for genetic diversity in many organisms. Although all genetic variation starts from mu-

tation, recombination can give rise to new variants by combining types already present in the population. Recombination tends to break the dependence among alleles on either side of the crossover and hence reduce the Linkage Disequilibrium (LD). Recent studies of DNA polymorphism data from the human genome (see e.g. [11], [4], [23]) suggested an interesting block like structure of the genome, where long stretches of the human genome known as *LD blocks* (with high LD) show signs of little or no recombination and the recombination events occur in so called *recombination hot-spots*. In humans, various recombination hotspots have been identified using experimental techniques. Jeffreys et. al. [22] analyzed a 216-kb region from the major histocompatibility complex (MHC) using sperm typing and identified clusters of recombination hotspots separated by long regions (60-90 kbs) of low diversity. However, the experimental determination of recombination rates at high resolution is technically difficult and costly. An alternative approach is to use population genetics data to infer the fine-scale variations in recombination rates. A variety of statistical methods based on different population genetics models have been proposed to estimate recombination rates from DNA sequences sampled from human populations. (see e.g. [14], [9], [20], [28], [27]). The emergence of genome-wide diversity studies, such as the HapMap project[17], has accelerated efforts towards constructing a fine-scale recombination map of the human genome. More recently, two large scale studies [29, 3] have shown fine-scale recombination rate variation and recombination hotspots to be a ubiquitous feature of the human genome.

In contrast to model based methods to infer recombination rates, an alternative approach for characterizing the heterogeneity in recombination in the human genome is to obtain a direct count of past recombination events from population genetics data. Population genetics data, in particular haplotype data contains signature patterns left behind by past recombination events. A parsimonious approach to counting recombination events is to compute the minimum number of recombination events required to explain any evolutionary history of the sample assuming that each segregating sites mutates only once. This may be achieved by trying to reconstruct the

underlying graph or phylogenetic network that uses the minimum number of recombination events. This problem is computationally challenging and has resisted efforts for even an exponential time algorithm [18, 19, 35] (see [39], [5], [6], [15] for some recent work on efficient algorithms for restricted versions of this problem). Therefore, research in this area has focused on computing lower bounds on the minimum number of recombination events. One should remember that most recombination events that took place in the history cannot possibly be recovered due to multiple reasons; small sample size, absence of past haplotypes, etc. However, one expects that the number of recombination events detected for a particular genomic region is a good indicator of the underlying recombination rate for that region. Myers and Griffiths [30] demonstrated the  $R_h$  lower bound to be much more powerful than previous lower bounds in detecting recombination events through simulation studies and found a strong clustering of recombination events in the center of the lipoprotein lipase gene [32]. This region has previously been characterized to be a putative recombination hotspot [38]. Fearnhead et. al. [10] applied the  $R_h$  method of Myers and Griffiths [30] to detect recombination events in the  $\beta$ -globin gene cluster which has a well-characterized recombination hotspot. They found that the results obtained using this method were consistent with their estimates obtained using a full likelihood method and moreover gave a better evidence for the recombination hotspot than the pairwise LD summaries.

### 1.1 Our contribution

In this paper, our objective is to explore the problem of computing lower bounds on the number of recombination events both from an algorithmic and application perspective. We demonstrate that recombination lower bounds, although conservative, are a good indicator of the underlying heterogeneity in recombination rates across a given genomic region and can indicate the presence of recombination hotspots.

We provide a theoretical formulation for the lower bound,  $R_h$  and show that it is NP-hard to compute this bound. However, on the positive side, using the greedy algorithm for the set cover problem [24], we present a  $O(mn^2)$  time algorithm which computes a lower bound  $R_g$  for a dataset with  $n$  rows and  $m$  segregating sites. The algorithm for computing  $R_g$  outperforms the Recmin implementation of Myers and Griffiths [30] by several orders of magnitude on large datasets (such as the one by Daly et. al. [4]) and finds the same lower bounds.

We also show that computing the lower bound  $R_s$  is

NP-hard using a reduction from MAX-2SAT. We give an  $O(m2^n)$  time algorithm for computing  $R_s$  which enables us to apply it to real datasets. The previous implementation of Myers and Griffiths [30] had only an  $\Omega(m \cdot n!)$  bound and is intractable for more than 10-15 haplotypes. Next, we show that the lower bound  $R_s$  can underestimate the true number of recombination events since it does not consider missing haplotypes. We propose a new bound  $R_I$  which extends  $R_s$  using the notion of intermediate haplotypes. This bound finds the optimal bound of 7 for the haplotypes from the ADH locus of *Drosophila Melanogaster* [26] and detects more recombination events than the  $R_s$  method on several datasets from the SeattleSNP database [31].

Most real haplotype datasets have some amount of missing data. A simple way of handling missing data is to not consider SNP's which have missing alleles for some haplotypes. However, this can possibly lead to reduced bounds. We provide extensions of the bounds  $R_g$  and  $R_s$  for efficiently computing bounds for haplotype datasets with missing data. These bounds applied to the LPL dataset [32] detect many more recombination events (in comparison to the number detected by ignoring the sites with missing data) which provide strong support for the presence of a recombination hotspot [38]. Finally, we apply our methods to the polymorphism data from the MHC region and show plots which clearly indicate the presence of recombination hotspots that were detected by Jeffreys et. al. [22] through sperm typing. We also find that the location of the hotspots (determined using sperm typing) are in good agreement with the values obtained using recombination lower bounds.

## 2 BASIC DEFINITIONS AND PREVIOUS WORK

A single nucleotide polymorphism (commonly known as a SNP) is a position in the genome where multiple (predominantly two) bases are observed in the population. Very few polymorphic sites (about 0.1% ) in humans have been found to be tri-allelic, i.e. having more than two different bases at the given site. In fact, tri-allelic polymorphism detection is often used to flag possible experimental error (see for e.g. [34]). Since many of the SNP loci are neutral, a significant violation of the infinite-sites assumption should result in a much higher fraction of tri-allelic polymorphic sites. Therefore, it is reasonable to make the *infinite-sites* or no-homoplasy assumption while dealing with human polymorphism data. As there are only two alleles at every site (the ancestral and the mutant), the extant data is represented by a binary matrix  $M$  with  $n$  rows and  $m$  columns, with

the two nucleotides arbitrarily renamed 0 and 1. Hence, all our results on binary character data are applicable to real haplotype data.

## 2.1 Recombination and Phylogenetic Networks

A recombination event at site  $p$ , between two haplotypes  $A$  and  $B$ , produces a recombinant sequence  $C$ , which is either a concatenation of sites  $A[1 \dots p]$  with  $B[p+1 \dots m]$  or  $B[1 \dots p]$  with  $A[p+1 \dots m]$ . A phylogenetic network  $G$  for a set  $M$  of  $n$  sequences is a directed acyclic graph with a root. The root has no incoming edges. Each node in  $G$  is labeled by a  $m$ -length binary sequence where  $m$  is the number of sites. Each leaf of this graph is labeled by a sequence in  $M$ . Each node other than the root has either one or two incoming edges. A node with two incoming edges is called a *recombination* node. Some of the edges are labeled by the columns (sites) of  $M$  which correspond to a mutation event at that site. For a non-recombination node  $v$ , let  $e$  be the single incoming edge into  $v$ . The sequence labeling  $v$  can be obtained from the sequence labeling  $v$ 's parent by changing the value at the sites which label the edge  $e$  from 0 to 1 (assuming that the root sequence is all-0). Each recombination node  $v$  is associated with an integer  $r_v$  (in the range  $[2, m]$ ), called the recombination point for  $v$ . Corresponding to the recombination at node  $v$ , one of the two sequences labeling the parents of  $v$  is denoted as  $P$  and the other one as  $S$ . The sequence labeling node  $v$  is a concatenation of the first  $r_v - 1$  characters of  $P$  with the last  $m - r_v + 1$  characters of  $S$ . The sequences labeling the leaves of the phylogenetic network are referred to as *extant* sequences.

A phylogenetic network  $G$  explains a set  $M$  of  $n$  haplotypes iff each sequence labels exactly one of the leaves of  $G$ . For a given set of haplotypes, there can be many possible phylogenetic networks with varying number of recombination events which explain the set. We define  $m_M$  to be the *minimum number of recombinations required to explain  $M$* , i.e. there exists a phylogenetic network with  $m_M$  number of recombinations which explains  $M$  and there is no phylogenetic network with fewer number of recombination events that explains  $M$ .

## 2.2 Recombination Lower Bounds

Currently, there is no algorithm that can efficiently compute  $m_M$  for a given dataset of reasonable size. However, one is not always interested in the exact number of events and lower bounds on the minimum number of recombination events are informative. The lower bound  $R_m$ , introduced by Hudson and Kaplan [21] is based on

the *four-gamete test*; if for a pair of SNP's with ancestral and mutant alleles  $a/b$  and  $c/d$  respectively, all four possible gametes ( $ac$ ,  $ad$ ,  $bc$ ,  $bd$ ) are present, then at least one recombination event must have happened between the pair of loci under the assumption that no site mutates more than once (*infinite sites model of mutation*). Based on this idea, one can find all intervals in which recombination must have occurred and choose the largest set of non-overlapping intervals from this collection. The bound  $R_m$  is the number of intervals in this set. However,  $R_m$  is a conservative estimate of the minimum number of recombination events that have occurred in the history of the sample, since it only considers pairs of SNP's to infer recombination events.

One can use haplotype diversity to infer more than one recombination event in an interval. Consider an interval with  $m$  segregating sites. If  $n (> m + 1)$  distinct haplotypes are observed in this interval, then at most  $m$  haplotypes can be explained using mutation events. Assuming that the ancestral haplotype is present in the sample, the remaining  $n - m - 1$  haplotypes must arise due to recombination events. Hence, one can infer a lower bound of  $n - m - 1$  for the interval. Moreover, one can choose any subset of segregating sites for an interval and compute this difference to obtain another lower bound for that region. Taking the maximum bound over all subsets of segregating sites in a particular region, gives the best lower bound,  $R_h$  [30]<sup>1</sup>.

The bounds  $R_m$  and  $R_h$  do not explicitly consider possible histories of the sample. The lower bound  $R_s$ , proposed by Myers and Griffiths [30], computes for every history (an ordering of the haplotypes), a simplified number of recombination events, such that any a phylogenetic network that is consistent with this history, requires more recombination events than this number. By minimizing over all possible histories, one obtains a lower bound on the minimum number of recombination events. Myers and Griffiths [30] provide an algorithmic definition for the bound  $R_s$ . Their algorithm performs three kinds of operations on a given matrix: row deletion, column deletion and non-redundant row removal. A *row deletion* can be performed if the given row is identical to another row in the matrix. Such a row is also referred to as a *redundant* row. A *column deletion* can be done if the column (site) is *non-informative* (all but one rows have the same allele at this site). A *non-redundant row removal* is a row removal when there are

<sup>1</sup>Myers and Griffiths refer to the global bound obtained by combining these local lower bounds as the bound  $R_h$ . In this paper, we refer to the lower bound described above as  $R_h$  for ease of exposition

no non-informative sites in the matrix and no redundant rows. Given an ordering of the  $n$  rows, the algorithm performs a sequence of column deletions, row deletions and non-redundant row removals until there is no row left in the matrix  $M$ . The minimum number of non-redundant row removal events over all possible histories gives the bound  $R_s$ . Since, the procedure considers all  $n!$  histories, the worst case complexity of this procedure is  $\Omega(m.n!)$ .

Song and Hein [36] proposed a set-theoretic lower bound which computes the optimal bound for the Kreitman dataset [26]. Recently, the authors of this paper [1] and Gusfield et. al. [16] independently showed that the number of connected components in the conflict graph of the set of segregating sites is also a lower bound on the number of recombination events. While being computationally efficient, this bound can never be better than  $R_s$ .

### 2.3 Combining Local Recombination Bounds

Myers and Griffiths [30] presented a general framework for computing recombination lower bounds from haplotype data. This framework can combine local recombination bounds on continuous subregions of a larger region to obtain recombination bounds for the larger parent region. Consider a matrix  $M$  with  $m$  segregating sites labeled 1 to  $m$ . Suppose that one has computed, for every interval  $(i, j)$  ( $1 \leq i < j \leq m$ ), a lower bound  $b_{ij}$  on the number of recombination events between the sites  $i$  and  $j$ . Each local lower bound  $b_{ij}$  can be computed by any lower bound method described previously and bounds for different intervals may be obtained by different methods.

In the second step, which is essentially a dynamic programming algorithm, one computes a new lower bound  $B_{ij}$  on the minimum number of recombination events between the sites  $i$  and  $j$  using the local bounds  $b_{i'j'}$ ,  $i' \leq i < j \leq j'$ . A recursive description of the algorithm is:

$$B_{ij} = \max_{k=i+1}^{j-1} (B_{ik} + b_{kj})$$

Note that the combined lower bound  $B_{ij}$  can be substantially better than the corresponding local bound  $b_{ij}$  for an interval  $(i, j)$ . In particular,  $B_{1m}$  gives the best lower bound for the whole region. It is important to note that all the practical results in this paper are obtained by computing lower bounds (by using the corresponding lower bound method) for all intervals of length  $w$  (specified as a parameter) for the given dataset, and combining them using the dynamic programming algorithm.

In the next couple of sections, we describe various methods for computing the local bounds.

## 3 BOUNDS BASED ON HAPLOTYPE DIVERSITY

Consider a matrix  $M$  and let  $S' \subseteq S$  be a subset of sites in  $M$ . For a subset  $S'$  of segregating sites, we denote the set of distinct haplotypes induced by  $S'$  as  $H(S')$ . The  $R_h$  bound of Myers and Griffiths[30] is based on the observation that  $|H(S')| - |S'| - 1$  is a lower bound on the number of recombinations for every subset  $S'$ . Since the number of subsets is  $2^w$  for a region of width  $w$ , Myers and Griffiths [30] use the approach of computing this difference for subsets of size at most  $s$  where  $s < w$  is a specified parameter. Increasing  $s$  can provide better bounds with an increase in computation time since the running time is exponential in  $s$ . We define the algorithmic problem associated with the computation of the bound  $R_h$  as follows:

### MDS: Most Discriminative SNP subset problem

**Input:** A binary matrix  $M$  and an integer  $k$ , where  $S$  is the set of columns of  $M$ .

**Output:** Is there a subset  $S'$  of  $S$ , such that  $|H(S')| - |S'| - 1 \geq k$ .

The  $R_h$  bound can simply be defined as:

$$R_h(M) = \max_{S' \subseteq S} (|H(S')| - |S'| - 1)$$

We show that Most Discriminative SNP subset problem is NP-complete by a reduction from the *Test Collection Problem*[12]. An instance of the test collection problem consists of a collection  $\mathcal{C}$  of subsets of a finite set  $\mathcal{S}$  and an integer  $k$ , and the objective is to decide if there is a subcollection  $\mathcal{C}' \subseteq \mathcal{C}$  such that for each  $x, y \in \mathcal{S}$  there exists  $c \in \mathcal{C}'$  that contains exactly one of  $x$  and  $y$  and  $|\mathcal{C}'| \leq k$ . An instance of the test collection problem can be encoded as a binary matrix  $M$  of size  $|\mathcal{S}| \times |\mathcal{C}|$ . Each row of the matrix corresponds to an element of the finite set  $\mathcal{S}$  and  $M[x, c] = 1$  if the subset  $c$  contains the element  $x$  and 0 otherwise. Here, the objective is to find a subset  $S'$  of the columns of  $M$  of size at most  $k$  such that for every pair of rows in  $M$ , there is a column in  $S'$  that can distinguish between them, i.e.  $|S'| \leq k$  and  $H(S') = |\mathcal{S}|$ . We use this encoding in the NP-completeness proof of the MDS problem below.

**Lemma 1:** The MDS problem is NP-complete.

**Proof:** Given an instance  $(\mathcal{S}, \mathcal{C}, k)$  of the test collection problem, we construct an instance  $(M, |\mathcal{S}| - k - 1)$  of the MDS problem using the matrix encoding described above.

Consider a solution  $\mathcal{C}' \subseteq \mathcal{C}$  of the test collection problem, i.e. there is a subset  $S'$  of the columns of  $M$  such that  $|H(S')| = |\mathcal{S}|$  and  $|S'| \leq k$ . It follows that for

the subset  $S'$ ,  $|H(S')| - |S'| - 1 \geq |\mathcal{S}| - k - 1$ . Hence,  $S'$  is a valid solution for the instance of the MDS problem.

Now, let  $S' \subseteq S$  ( $S$  is the set of columns of  $M$ ) be a solution of the instance  $(M, |\mathcal{S}| - k - 1)$  of the MDS problem, i.e.  $|H(S')| - |S'| - 1 \geq |\mathcal{S}| - k - 1$ . Consider a row  $h \in M$  such that  $h \notin H(S')$ . For such a row, there is exactly one row  $h'$  in  $H(S')$  that is identical to  $h$ , since all rows in  $H(S')$  are distinct by definition. Also, there is a column  $s \in S - S'$  such that the value at this column in  $h$  is different from the value at this column in row  $h'$ . Hence, we can add the column  $s$  to  $S'$  and the row  $h$  to  $H(S')$  to get another set  $S''$  such that  $|S''| = |S'| + 1$  and  $|H(S'')| \geq |H(S')| + 1$ . Clearly,  $S''$  is a valid solution for the  $(M, |\mathcal{S}| - k - 1)$ -MDS problem. Inductively, we can add all rows not present in  $H(S')$  to obtain a subset of columns  $S^*$  such that  $H(S^*) = H$ . Therefore,  $|S^*| \leq |H(S^*)| - (|\mathcal{S}| - k) = k$ . Hence  $S^*$  is a solution for the test collection problem. Therefore, the test collection problem and  $(M, |\mathcal{S}| - k - 1)$ -MDS problem are equivalent. The NP-completeness of the MDS problem follows. ♣

### 3.1 The lower bound $R_g$

From the above reduction, it is easy to see that computing the bound  $R_h$  is equivalent to finding a smallest subset of columns  $C$  such that for every pair of haplotypes (rows)  $(x, y)$  in  $M$ , there is at least one column  $c \in C$  such that  $M[x, c] \neq M[y, c]$ . We can write down an Integer Linear Program for computing  $R_h$  as follows:

$$\begin{aligned} \max \quad & \left( |H| - \sum_{c \in C} x_c \right) \\ \text{s.t.} \quad & \sum_{c \in C} [M(h_1, c) \oplus M(h_2, c)] x_c \geq 1 \quad \forall (h_1, h_2) \in H^2 \\ \text{and} \quad & x_c \in \{0, 1\} \quad \forall c \in C \end{aligned}$$

It is computationally difficult to solve this ILP exactly. However, we note that any feasible solution for this ILP gives a lower bound on  $R_h$  and hence is a valid lower bound. We adapt the standard greedy algorithm for the set cover problem [24] to devise an algorithm (described in Figure 1) for computing a lower bound; denoted as  $R_g$ .

### 3.2 Results: Comparison for Daly trios

It is well known that the greedy algorithm gives a  $1 + 2 \ln n$  approximation for the test collection problem where  $n = |\mathcal{S}|$ , the size of the ground set. However, this approximation ratio does not apply to the MDS problem. Although, in general  $R_g \leq R_h$ , we found that the overall bound (obtained by combining the local bounds computed using  $R_g$ ) was equal to the corresponding

COMPUTE\_  $R_g(M)$

1. Repeat
  - If two rows in  $M$  are identical, coalesce them.
  - If a site  $s$  is non-informative, remove the site  $s$ .
  - For a pair of sites  $(a, b)$ , if  $P(a, b) = 2$ , remove site  $a$  until possible.
2. Let  $M'$  be the reduced matrix with  $n$  rows and  $m$  sites
3. Initialize  $d(x, y) = 0$  for all pairs of rows and  $I = \phi$
4. **while**  $d(x, y) = 0$  for some pair
5.     Let  $s'$  be the column that can distinguish between the
6.     maximum pairs for which  $d(x, y) = 0$
7.     set  $d(x, y) = 1$  for all  $(x, y)$  s.t.  $M'[x, s'] \neq M'[y, s']$
8.      $I = I \cup \{s'\}$
9. Return  $|H(I)| - |I| - 1$

Figure 1: The greedy algorithm for computing the bound  $R_g$ . For a pair of columns  $(a, b)$ ,  $P(a, b)$  denotes the number of distinct pairs in the two columns.

Parameters	Recmin program [30]		$R_g$ (greedy algorithm)	
	Bound	time	Bound	time
default(w=15 s=6)	134	4 secs	180	01 secs
w=20 s=10	183	2.5 mins	188	03 secs
w=25 s=10	186	31 mins	198	06 secs
w=30 s=15	200	29 hrs	199	11 secs
w=35	-	-	203	15 secs

Table 1: Comparison of the performance of the Recmin program [30] and our  $R_g$  bound for the Daly haplotypes with different values of the parameters; maximum subset size (s) and maximum width (w). Note that the  $R_g$  bound requires only the parameter  $w$ .

bound returned by the Recmin program [30] for almost all datasets we did the comparison for (we believe that this is due to the effect of combining the local bounds). The running time of the Recmin program [30] is proportional to  $\sum_{i=2}^s \binom{w}{i}$  where  $w$  is the maximum number of segregating sites in a region for which the local bound is computed and  $s$  is the maximum subset size used for computing the  $R_h$  bound. In contrast, in order to compute the best bound by combining the local  $R_g$  bounds, we require only one parameter, i.e maximum width and the overall running time is  $O(n^2mw^2)$ . To illustrate the kind of improvements we obtain using  $R_g$ , we compare the bounds (Recmin and  $R_g$ ) for the phased haplotypes (258 haplotypes on 103 SNP's) of the Daly [4] dataset obtained from the Hap Webserver [8, 7] (see table 1).

## 4 HISTORY BASED LOWER BOUNDS

Myers and Griffiths[30] only give a procedural definition of the bound  $R_s$ , and their description is somewhat informal. The time complexity of their procedure (as described in Algorithm 3 in [30]) is  $O(mn!)$ , where  $n$  is

the number of rows, and  $m$  the number of columns. We give a theoretical formulation of the bound  $R_s$  which allows us to develop an exponential time algorithm for computing it and also show that computing  $R_s$  is NP-hard.

We define a history for a set of  $n$  rows as simply an ordering of the rows. We start by redefining  $R_s$  in terms of appropriate cost of a row in a given history. Consider a history  $H = r_1 \rightarrow r_2 \dots \rightarrow r_n$ . The cost of row  $r_i$  in the history, denoted by  $C_s(r_i)$ , is 0 if after removing non-informative columns from  $r_1, r_2, \dots, r_i$ , the row  $r_i$  turns out to be identical to one of the rows  $r_1, \dots, r_{i-1}$  and 1 otherwise. Then we have

$$C_s(H) = \sum_i C_s(r_i)$$

and

$$R_s(M) = \min_{\text{history } H} C_s(H)$$

We defer the discussion of why  $R_s$  is a lower bound to Theorem 3 (where we prove that  $R_I$  is a lower bound). Consider a bit vector  $\vec{r}$  of lengths  $n$ . Let  $M_{\vec{r}}$  denote a submatrix of  $M$  which contains only rows  $i$  such that  $r_i = 1$ . Define a partial order on the vectors as follows:  $\vec{v}_1 \leq \vec{v}_2$  if  $v_2[i] = 1$  whenever  $v_1[i] = 1$ . Define the vector  $\vec{v}_{-i}$  as the  $\vec{v}$  with the  $i$ -th bit set to 0. Let  $R_S[\vec{v}]$  denote the  $R_s$  bound for the corresponding submatrix. The procedure in Figure 2 gives an  $O(m2^n)$  algorithm for computing  $R_s$ . This dynamic programming algorithm can bring significant improvements in running time. Indeed, the  $R_s$  implementation of Myers and Griffiths [30] is practical for only datasets with 10-15 haplotypes and the default in the Recmin program is to compute  $R_h$  instead of  $R_s$ .

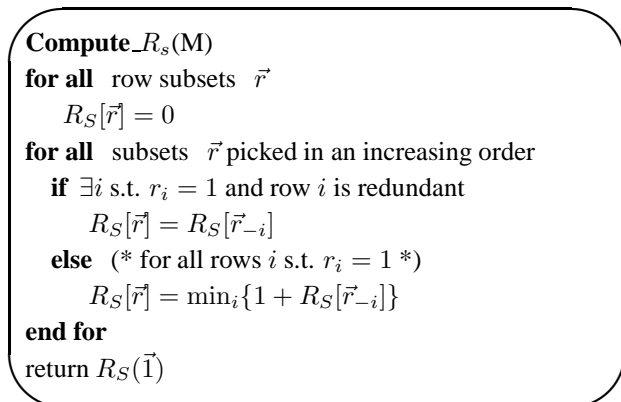


Figure 2: An  $O(m2^n)$  algorithm for computing  $R_s$  ( $\vec{1}$  refers to all-ones vector of length  $n$ )

Using a non-trivial reduction from the MAX-2SAT problem we show that computing the bound  $R_s$  for a matrix is NP-hard (for the proof, see Appendix).

**Theorem 2:** Computing  $R_s(M)$  is NP-hard.

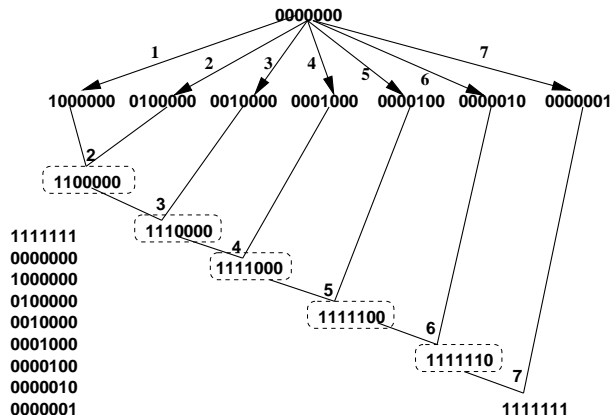


Figure 3: A set of 9 haplotypes for which  $R_s$  is 1 and a phylogenetic network for the set of haplotypes with 6 recombination events ( $R_I = 6$ )

The  $R_s$  bounds searches over possible histories of the set of haplotypes and one would expect the bound to be better than the diversity based bound  $R_h$ . Myers and Griffiths [30] gave a small example of eight haplotypes where  $R_s$  improves over  $R_h$  and remarked that the bound  $R_s$  is always better than the bound  $R_h$ . However, in practice, both  $R_h$  and  $R_s$  underestimate the true bound in many instances.

#### 4.1 Recombinant Intermediates and the bound $R_I$

We use an example to demonstrate how  $R_s$  can be improved. Consider the set of  $n + 2$  haplotypes with  $n$  sites shown in Figure 3. For illustration  $n = 7$ .

Note that if the history was forced to start with the first two haplotypes, each of the following  $n$  rows could only be removed through a non-redundant row removal, and we would have a recombination bound of  $n$ . However, if we choose 1111111 to be the last haplotype in the history, then removing it makes every column non-informative. As  $R_s$  is the minimum over all histories,  $R_s(M) = 1$ . However, at least 6 recombinations are needed. Note that for this particular example, we can boost the  $R_s$  bound to the correct value by applying the dynamic programming algorithm [30] for combining local bounds. However, the example illustrates a problem with  $R_s$ , which is that in explaining a non-redundant row-removal, we only charge a *SINGLE* recombination event. Therefore, if 1111111 was indeed the last haplotype in the true history, then adding it would require 5 recombinants (the haplotypes in dashed boxes) NOT from the current set (as explained in Figure 3).

We use this idea to improve the  $R_S$  bound. Consider a history  $H = r_1 \rightarrow r_2 \dots \rightarrow r_n$ . Let  $\mathcal{I}_j(H)$  denote the

minimum number of recombination events in obtaining  $r_j$ , given any phylogenetic network for  $r_1, \dots, r_{j-1}$ . We allow the use of recombinant intermediates, and so  $\mathcal{I}_j(H)$  can be greater than one. In general, the use of recombinant intermediates is tricky because the intermediates may help explain some of the existing haplotypes by simple mutations. In order to prove a lower bound, we introduce the concept of a *direct recombination*. We define  $C_d(r_i)$  for a haplotype  $r_i$  in a given history  $H$  as follows:

$$C_d(r_i) = \begin{cases} 0 & r_i \text{ is different from all } r_{j < i} \text{ in a} \\ & \text{non-informative column.} \\ 0 & r_i \text{ is identical to } r_{j < i} \text{ after removing} \\ & \text{non-informative columns} \\ 1 & \text{Otherwise} \end{cases} \quad (1)$$

We observe that the definition of  $C_d(r_i)$  holds for a set of haplotypes  $\{r_1, r_2, \dots, r_{i-1}, r_i\}$  and denote this generic definition as  $C_d(r_i, \{r_1, r_2, \dots, r_{i-1}\})$ . Note that  $C_d(r_i) \leq C_s(r_i)$  for all  $i$  in a history. However,  $C_d$  can be used to give a new lower bound on the total number of recombinations.

**Theorem 3:** Let  $\mathcal{H}$  denote the set of all histories over the set of haplotypes  $M$ . Then

$$R_I = \min_{H \in \mathcal{H}} \max_j \left\{ \sum_{i < j} C_d(r_i) + \mathcal{I}_j(H) + \sum_{i \geq j} C_s(r_i) \right\}$$

is a lower bound on the number of recombinations.

**Proof:** Recall that  $m_M$  denotes the minimum number of recombinations in any history of  $M$ . We construct one history  $H = r_1 \rightarrow r_2 \dots \rightarrow r_n$  in which which  $\sum_{i < j} C_d(r_i) + \mathcal{I}_j(H) + \sum_{i > j} C_s(r_i)$  is a lower bound on  $m_M$  for all choices of  $j$ . This is sufficient because we minimize over all histories. Consider an phylogenetic network  $\mathcal{A}$  that explains  $m_M$  with a minimum number of recombinations. Each node  $v$  in the phylogenetic network corresponds to a haplotype  $r_v$ , which may or may not be in  $M$ . Haplotype  $r \in M$  is a *direct witness* for a recombinant node  $v$  if  $r = r_v$ . It is an *indirect witness* if it can be derived from  $r_v$  solely by mutation events. A predecessor relationship  $<_P$  is defined for some haplotypes  $r_i, r_j \in M$ . Specifically  $r_i <_P r_j$  if  $r_i$  is a (direct or indirect) witness to a recombinant node on a path from the root to  $r_j$ . Note that  $<_P$  is a partial order.

Next, choose a history  $H$  (a total ordering) that is consistent with  $<_P$ . Note that  $C_s(r_i) = 1$  if and only if  $r_i$  is the first witness to a recombination node in  $A$  to appear in  $H$  (thereby proving that  $R_s(M)$  is a lower bound). Likewise  $C_d(r_i) = 1$  if and only if  $r_i$  is the first

direct witness to a recombination node in  $A$  to appear in  $H$ . As each recombination node contributes at most 1,  $R_s = \sum_i C_s(r_i)$  is a valid lower bound on the number of recombinations. Consider an arbitrary  $r_j$  with  $C_s(r_j) = 1$ . Instead of charging 1 to the number of recombination events, we charge a value  $\mathcal{I}_j(H)$  equal to the minimum number of recombinations needed to obtain  $r_j$  from  $r_1, r_2, \dots, r_{j-1}$ . Consider the sequence of intermediate recombination events that were used to obtain  $r_j$ . None of these nodes have a direct witness. Therefore the nodes in  $r_1, r_2, \dots, r_{j-1}$  that had a  $C_d$  value of 1 correspond to other recombination nodes.

Next, the haplotypes  $r_{i > j}$  that follow  $r_i$  are charged  $C_s(r_i)$ . Whenever,  $C_s(r_i) = 1$ , it is because  $r_i$  is the first witness to a recombination node in  $A$  to appear in  $H$ . By construction, this recombination node is not on any path from root to  $r_j$ , and therefore wasn't charged when considering intermediates for  $r_j$ . Therefore, each recombination node is charged at most once and the bound holds. ♣

#### Compute $R_I(M)$

```

for all row subsets  $\vec{r}$ :  $R_d[\vec{r}] = 0$ ;  $R_I[\vec{r}] = 0$ 
for all subsets  $\vec{r}$  chosen in an increasing order
  if  $\exists i$  s.t.  $r_i = 1$  and row  $i$  is redundant
     $R_d[\vec{r}] = R_d[\vec{r}_{-i}]$ ;  $R_I[\vec{r}] = R_I[\vec{r}_{-i}]$ 
  else
    for all rows  $i$  s.t.  $r_i = 1$ 
       $R_{d,i} = \min_i \{C_d(r_i, \vec{r}_{-i}) + R_d[\vec{r}_{-i}]\}$ 
       $R_{I,i} = \min_i \{\max\{1 + R_I[\vec{r}_{-i}], R_d[\vec{r}_{-i}] + \mathcal{I}_i[\vec{r}_{-i}]\}\}$ 
    end for
     $R_d[\vec{r}] = \min_i \{R_{d,i}\}$ ;  $R_I[\vec{r}] = \min_i \{R_{I,i}\}$ 
  end if
end for
return  $R_I(M)$ 

```

Figure 4: An  $O(2^n I(m, n))$  algorithm for computing  $R_I$ . The bit-vector  $\vec{r}$  is used to describe a subset of haplotypes. Subsets are ordered so that the ones of smaller size are picked before the larger ones.  $\mathcal{I}_i[\vec{r}_{-i}]$  denotes the minimum number of recombinant intermediates needed to compute haplotype  $r_i$  given the subset  $\vec{r}$  with  $r_i$  removed.

It is easy to see that  $R_I \geq R_s$ . In order to compute  $R_I$ , we need to compute  $\mathcal{I}_j(H)$  for all haplotypes  $j$ , and all histories  $H$ . To do this more efficiently, we define  $\mathcal{I}_j$  over subsets, instead of histories. We denote a subset of haplotypes by the bit-vector  $\vec{r}$  of size  $n$  where  $\vec{r}_i = 1$  iff  $r_i$  is in the subset. Define  $\mathcal{I}_j[\vec{r}]$  as the minimum number of recombination events needed to obtain  $r_j$ , over any history of the haplotypes in  $\vec{r}$ . Likewise, define  $R_d(\vec{r})$  as the minimum number of direct recom-

binations in any history of the haplotype subset  $\vec{r}$ . The algorithm in Figure 4 describes how to compute  $R_I$  in time  $O(n2^n I(m, n))$  time, where  $I(m, n)$  is the time to compute  $\mathcal{I}_j[\vec{r}]$  for any subset  $\vec{r}$ .

## 4.2 Computing Recombinant Intermediates

Our goal is to compute  $\mathcal{I}_i[\vec{r}]$  efficiently. Haplotype  $i$  is assumed to arise later in history than in  $\vec{r}$  and is therefore a mosaic of sub-intervals of the haplotypes in  $\vec{r}$ . The mosaic can be expressed by a sequence of pairs  $M = (h_1, j_1), (h_2, j_2), \dots, (h_k, j_k)$  interpreted as follows: In  $h_i$ , columns  $1, \dots, j_1$  came from haplotype  $h_1$ , columns  $j_1 + 1, \dots, j_2 + 1$  from  $h_2$ , and so on. If  $M$  were the true mosaic, then  $h_i$  would need  $k - 1$  recombinant intermediates. Thus, we need to minimize this.

First, we can ignore all columns that are identical for all haplotypes in  $\vec{r}$ . If  $h_i$  has a different value in any of these columns, it can be explained by a mutation. If it has the identical value, the column can be explained using any haplotype and will not contribute to recombination. Ignoring these columns, the following is true: if columns  $j_1, \dots, j_2$  of  $h_i$  arise from haplotype  $h$ , then the values of  $h$  and  $h_i$  must be identical in columns  $j_1$  through  $j_2$ . If any column  $c$  was different ( $h_i[c] \neq h[c]$ ), to explain it by a mutation would violate the infinite-sites assumption. This observation allows us to solve the problem of computing  $\mathcal{I}_i[\vec{r}]$  efficiently.

For column  $c$ ,  $1 \leq c \leq m$  and haplotype  $h$ , let  $I[c, h]$  denote the minimum number of recombinations needed to explain the first  $c$  columns of haplotype  $h_i$  such that the  $c$ -th column arose from haplotype  $h$ . This is sufficient because

$$\mathcal{I}_i[\vec{r}] = \min_h \{I[m, h]\}$$

$I[c, h]$  can be computed using the following recurrence:

$$I[c, h] = \begin{cases} 0 & c = 0 \\ \infty & h_i[c] \neq h[c] \\ \min \left\{ \begin{array}{l} I[c-1, h], \\ \min_{h' \neq h} \{1 + I[c-1, h']\} \end{array} \right\} & o/w \end{cases}$$

## 4.3 Results for $R_I$ bound

Besides the simulated example (in Figure 3), real datasets are known where  $R_s$  and  $R_h$  are sub-optimal. As an example, the  $R_h$  and  $R_s$  bounds for Kreitman's data [26] from the ADH locus of *Drosophila Melanogaster* are both 6. Song and Hein [36] showed that their set theoretic lower bound gave a bound of 7 and proved this to be optimal by actually constructing an phylogenetic network which requires 7 recombination events. Our new

Dataset	Size	$R_s$	$R_I$
CSF3	15x17	3	4 (optimal)
MMP3	21x41	6	9
EPHB6	31x62	23	25
ABO	68x197	70	73
DCN	31x117	16	19
HMOX1	34x53	14	16
F2RL3	28x29	10	11
F13B	24x77	22	23

Table 2: Comparison of the number of detected recombination events using  $R_s$  and  $R_I$  for the phased haplotype datasets for various genes obtained from the SeattleSNP project [31]

lower bound  $R_I$  also returns the optimal bound of 7. However, the set theoretic-bound [36] does not have an explicit algorithmic description. On the other hand, the  $R_I$  bound can be computed for large datasets ( $100 \times 500$  matrix can be analyzed in about an hour on a standard PC) and gives improved bounds for a number of real datasets (see Table 2 for a partial list).

## 5 BOUNDS FOR HAPLOTYPES WITH MISSING DATA

A complete haplotype is an element of  $\{0, 1\}^m$  where  $m$  is the number of SNP's and the  $j$ -th component indicates the nucleotide at that position. However, due to errors or other reasons, the allele at a particular position for an individual is sometimes not available. In such a scenario, some of the haplotypes are partial or incomplete. A partial haplotype is an element of  $\{0, 1, ?\}^m$  where  $?$  represents the positions where the allele is unknown. Since most of the real haplotype data has missing entries, it is important to find efficient methods to find recombination lower bounds for haplotypes with missing data. The lower bounds  $R_h$  and  $R_s$  do not naturally extend for an incomplete haplotype matrix. However, in this section, we show how both the greedy algorithm for computing  $R_d$  and the exponential algorithm for computing  $R_s$  can be extended for an incomplete matrix without much increase in the computational complexity. We first need to modify the definitions of non-informative site and redundant row. A site is non-informative if it has all but one alleles of one type (ignoring the missing alleles). For comparing two rows, we define  $M[x, a] \neq M[y, b]$  if and only if  $M[x, a] \neq ?$  and  $M[y, b] \neq ?$  and  $M[x, a] \neq M[y, b]$ .

### 5.1 $R_d$ bound for an incomplete matrix

In the last step of the greedy algorithm (Figure 1), the algorithm returns the bound  $H(I) - I - 1$ . For a matrix with missing entries, it is not straightforward to compute  $H(I)$ . However, consider an assignment to the '?'s

that minimizes  $H(I)$ . Then the difference  $H(I) - I - 1$  gives a valid lower bound, i.e. a bound which is valid for all possible assignments to the missing entries. However, one then needs to solve the minimum haplotype completion problem; where given an haplotype matrix with missing entries, the objective is to complete the missing entries so as to minimize the number of distinct haplotypes. This problem was shown to be NP-hard by Kimmel. et. al. [25]. However, for our purposes, we use a simple heuristic to find the minimum number of rows that can be distinguished using the non-missing entries. This gives a valid lower recombination lower bound that is easy to compute in practice.

## 5.2 $R_{sm}$ : history based bound for missing data

Consider a set of haplotypes  $M$  where some of the haplotypes are incomplete. We define a completion to be assignment of 0 or 1 to every missing allele in  $M$ . Clearly, there exists a completion  $M'$  of  $M$  and a corresponding ordering  $H$  for that complete matrix  $M'$ , such that the number of row removal operations is minimum over all completions and all orderings. In other words, the definition of the  $R_s$  lower bound has to be modified as follows:

$$R_s(M) = \min_{\text{completion } M'} [R_s(M')]$$

Since a complete matrix is a special case of a incomplete matrix, it follows that it is also NP-hard to compute the modified version of  $R_s$  for an incomplete matrix. We extend the  $O(m \cdot 2^n)$  algorithm for  $R_s$  to obtain a lower bound  $R_{sm}$  (this bound may not exactly equal  $R_s$  as defined for an incomplete matrix) for an incomplete matrix (see Figure 5). The next lemma shows this is a valid lower bound for the matrix  $M$  over all possible completions of the matrix.

**Lemma 4:** For a incomplete haplotype matrix  $M$ ,

$$R_{sm}(M) \leq R_s(M) \leq m_M$$

**Proof:** See Appendix. ♣

## 5.3 Application to haplotype data from LPL locus

A 9.7-kb region in the human LPL gene was sequenced by Nickerson. et. al. [32] in 71 individuals from three different populations comprising of 24 individuals from Jackson, Mississippi, 24 from North Karelia, Finland and 23 from Rochester, Minnesota. The haplotype data comprised of 88 haplotypes defined by 69 variable sites with about 1.2% missing data.

This data has previously been analysed for haplotype diversity and recombination by Clark et. al. [2]

```

Compute_ $R_{sm}(M)$ 
for all row subsets  $\vec{r}$  :  $R_{sm}[\vec{r}] = 0$ 
for all subsets  $\vec{r}$  picked in an increasing order
  if  $\exists$  a redundant row in  $\vec{r}$ 
     $R_{sm}[\vec{r}] = \min_i \{R_{sm}[\vec{r}_{-i}]\}$  (* for all rows  $i$  s.t.  $i$ 
    is redundant *)
  else
     $R_{sm}[\vec{r}] = \min_i \{1 + R_{sm}[\vec{r}_{-i}]\}$  (* for all rows  $i$ 
    s.t.  $r_i = 1$  *)
  end if
end for

```

Figure 5: An  $O(m2^n)$  algorithm for computing a history based bound for an incomplete haplotype matrix

Region	Site Range			
	106-2987	2987-4872	4872-9721	Full
Jackson	10(10)	11(9)	17(13)	39(36)
Finland	2(2)	13(13)	13(11)	31(27)
Rochester	1(1)	13(13)	7(7)	22(21)
Combined	13(12)	37(22)	36(28)	87(70)

Figure 6: The number of detected recombination events using methods for missing data for the LPL datasets. The number in bracket indicates the corresponding lower bound obtained by ignoring sites with missing alleles [30]. The region (2987-4872) corresponds to the suggested hotspot [38]

and Templeton et. al. [38]. Myers and Griffiths [30] also applied their  $R_h$  method to compute lower bounds for the three different populations and the combined dataset. In figure 6, we compare the bounds obtained for different sub-regions of the LPL region for various populations. The overall bound for the whole region is 70 if one ignores the sites with missing data (see [30]), while we obtain a much improved bound of 87 by applying our  $R_h/R_{sm}$  bounds along with the dynamic programming framework. Templeton et. al. [38] had found the 29 recombination events detected using their method to be clustered near the center of the region (approximately between the sites 2987 and 4872). It is interesting to note that number of detected recombination events (37) in this region increases significantly (from 22) when one takes into account the sites with missing alleles. Thus, the bounds obtained using our improved methods which can handle missing data, seem to provide strong support for the presence of a recombination hotspot suggested by Templeton et. al. [38]. This demonstrates that the ability to extract past recombination events can be crucial to detecting regions with elevated recombination rates.

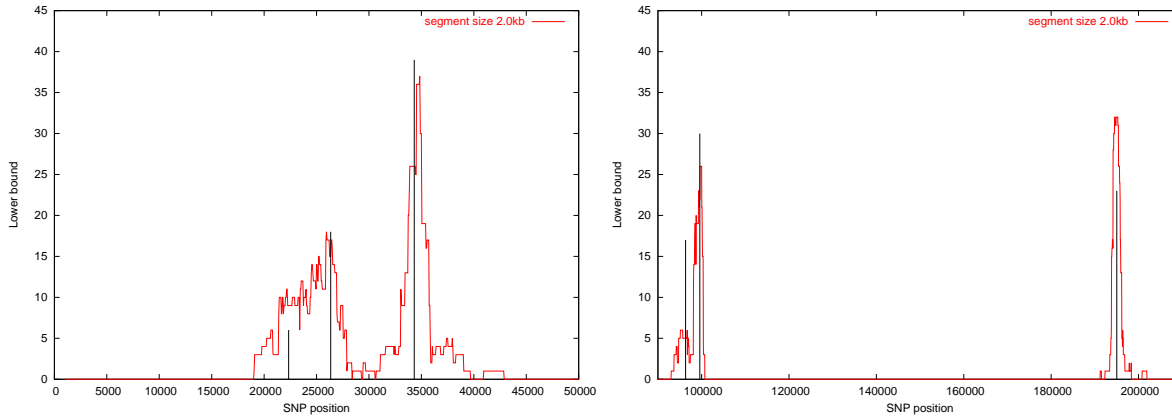


Figure 7: Plot of recombination lower bounds (on a 2-kb scale) for the 216-kb segment of the class II region of the major histocompatibility complex (MHC). The vertical black lines (height scaled by logarithm of the mean recombination rate obtained from sperm typing for that hotspot) show the approximate locations of the center of the six hotspots inferred using sperm crossover analysis by Jeffreys et. al. [22]. The TAP2 hotspot [23] is the last hotspot near the 200-kb region.

## 6 LOWER BOUNDS & RECOMBINATION HOTSPOTS

In humans, pedigree studies have shown variation in recombination rates on a megabase scale, and analyses of sperm crossovers in males [23, 22] have identified hotspots of length 1-2kbs where recombination events cluster. However, characterizing fine-scale variation in recombination rates using pedigree studies (at the kb scale) is difficult and experimental difficulties limit the large-scale application of sperm analyses. After several studies [4, 11, 33] observed a block-like structure in patterns of linkage disequilibrium in the human genome, it has been speculated that most or all recombination occurs in recombination hotspots [13]. The problem of detecting recombination hotspots (roughly defined as a region in which the recombination rate is much higher than the average recombination rate) using DNA polymorphism data has been considered by several studies [10, 27] which proposed statistical based methods to give quantitative estimates of recombination rates.

Here, we apply our lower bounds to the population data from a 216-kb segment of the class II region of the Major histocompatibility complex (MHC). Jeffreys et. al. sequenced 50 individuals from UK in this region and identified six recombination hotspots using sperm crossover analysis. Since the available data is unphased, we applied our lower bounds to the haplotypes estimated by the PHASE program [37]. Three separate studies [3, 10, 29] have applied their methods to infer recombination hotspots for this dataset. Although, sperm typing and recombination lower bounds measure very different things, we find that the lower bounds are able to locate most of the recombination hotspots with

high accuracy (see Figure 7). Five regions show excellent evidence of a hotspot with excellent agreement with the center of the corresponding hotspots (as found by Jeffreys et. al. [22]), with only one of the characterized hotspots (DMB1 near the 96kb region) showing a weak signal. This clearly demonstrates the ability of recombination lower bound methods to provide first hand indication of the presence and the location of hotspots. One criticism of lower bound methods is that we do not model events such as repeat mutations and gene conversion. However, such events are rare and our results (see also Myers and Griffiths [30]) suggest that this has only moderate effects on the bounds.

## 7 Discussion and Future Work

In this paper, we proposed new and improved methods for computing recombination lower bounds from haplotype data (also handling missing data). These methods are much more efficient than previous methods and computationally tractable for genome-scale datasets that are becoming increasingly available. We demonstrated that these methods can detect more recombination events than previous lower bounds on various real datasets. We believe that lower bounds can provide good indication about the location and intensity of hotspots given haplotype data with sufficient number of haplotypes and high enough SNP density. We demonstrated this using a real dataset from the MHC region where several recombination hotspots have previously been characterized. Obtaining efficiently computable lower bounds that are close to the true bound is an interesting problem for future work.

The algorithms discussed in this paper were implemented in C++ and the code is available on request.

## 8 Acknowledgements

The authors would like to thank Simon Myers for some useful clarifications about lower bounds, Andy Clark for providing the LPL dataset and Paul Fearnhead and Peter Donnelly for providing the datasets used in their paper.

## References

- [1] V. Bafna and V. Bansal. The number of recombination events in a sample history: Conflict graph and lower bounds. *To appear in IEEE Transactions on Computational Biology and Bioinformatics*, 2004.
- [2] A.G. Clark, K.M. Weiss, D.A. Nickerson, S.L. Taylor, A. Buchanan, J. Stengard, V. Salomaa, E. Vartiainen, M. Perola, E. Boerwinkle, and C.F. Sing. Haplotype structure and population genetic inferences from nucleotide-sequence variation in human lipoprotein lipase. *American Journal of Human Genetics*, 63:595–612, 1998.
- [3] D.C. Crawford, T. Bhangale, N. Li, G. Hellenthal, M.J. Rieder, D.A. Nickerson, and M. Stephens. Evidence for substantial fine-scale variation in recombination rates across the human genome. *Nature Genetics*, 36:700–706, 2004.
- [4] M. J. Daly, J. D. Rioux, S. F. Schaffner, T. J. Hudson, and E. S. Lander. High-resolution haplotype structure in the human genome. *Nature Genetics*, 29:229–232, 2001.
- [5] D. Gusfield, S. Eddhu, and C. Langley. Efficient reconstruction of phylogenetic networks with constrained recombination. In *IEEE Computer Society Bioinformatics Conference*, pages 363–374, 2003.
- [6] D. Gusfield, S. Eddhu, and C. Langley. The fine structure of galls in phylogenetic networks. In *To Appear in INFORMS J. of Computing: Special Issue on Computational Biology*, 2004.
- [7] E. Eskin and E. Halperin. Haplotype reconstruction from genotype data using imperfect phylogeny. *Bioinformatics*, 20(12):1842–9, 2003.
- [8] E. Eskin, E. Halperin, and R. Karp. Efficient reconstruction of haplotype structure via perfect phylogeny. *Journal of Bioinformatics and Computational Biology*, 1:1–20, 2003.
- [9] P. Fearnhead and P. Donnelly. Estimating recombination rates from population genetic data. *Genetics*, 159:1299–1318, 2001.
- [10] P. Fearnhead, R.M. Harding, J.A. Schneider, S. Myers, and P. Donnelly. Application of coalescent methods to reveal fine-scale rate variation and recombination hotspots. *Genetics*, 167:2067–2081, 2004.
- [11] S. B. Gabriel, S. F. Schaffner, H. Nguyen, J. M. Moore, J. Roy, B. Blumenstiel, J. Higgins, M. DeFelice, A. Lochner, M. Faggart, S. N. Liu-Cordero, C. Rotimi, A. Adeyemo, R. Cooper, R. Ward, E. S. Lander, M. J. Daly, and D. Altschuler. The structure of haplotype blocks in the human genome. *Science*, 296(5576):2225–2229, 2002.
- [12] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*. W.H. Freeman and Company, 1979.
- [13] D. B. Goldstein. Islands of linkage disequilibrium. *Nature Genetics*, 29:109–111, 2001.
- [14] R. C. Griffiths and P. Marjoram. Ancestral inference from samples of DNA sequences with recombination. *Journal of Computational Biology*, 3(4):479–502, 1996.
- [15] D. Gusfield. Optimal, efficient reconstruction of root-unknown phylogenetic networks with constrained and structured recombination. *UC Davis Technical Report*, 2004.
- [16] D. Gusfield and D. Hickerson. A fundamental, efficiently-computed lower bound on the number of recombinations needed in phylogenetic networks. *UC Davis Technical Report*, 2004.
- [17] The International HapMap Project. <http://www.hapmap.org/>, October 2004.
- [18] J. Hein. Reconstructing Evolution of sequences subject to recombination using parsimony. *Math. Biosci.*, 98:185–200, 1990.
- [19] J. Hein. A Heuristic Method to Reconstruct the History of Sequences Subject to Recombination. *J. Mol. Evol.*, 20:402–411, 1993.
- [20] R. R. Hudson. Two-locus sampling distributions and their applications. *Genetics*, 159:1805–1817, 2001.

- [21] R. R. Hudson and N. L. Kaplan. Statistical properties of the number of recombination events in the history of a sample of DNA sequences. *Genetics*, 111:147–164, 1985.
- [22] A. J. Jeffreys, L. Kauppi, and R. Neumann. Intensely punctate meiotic recombination in the class II region of the major histocompatibility complex. *Nature Genetics*, 29(2):217–222, 2001.
- [23] A.J. Jeffreys, A. Ritchie, and R. Neumann. High resolution analysis of haplotype diversity and meiotic crossover in the human *tap2* recombination hotspot. *Human Molecular Genetics*, 9:725–733, 2000.
- [24] D. Johnson. Approximation algorithms for combinatorial problems. *Journal of Comput. System Sci.*, 9:256–278, 1972.
- [25] G. Kimmel and R. Shamir. The incomplete perfect phylogeny haplotype problem. In *Proceedings of the Second RECOMB Satellite Workshop on Computational Methods for SNPs and Haplotypes*, 2004.
- [26] M. Kreitman. Nucleotide Polymorphism at the Alcohol Dehydrogenase Locus of *Drosophila Melanogaster*. *Nature*, 304:412–417, 1983.
- [27] N. Li and M. Stephens. Modeling linkage disequilibrium and identifying recombination hotspots using single-nucleotide polymorphism data. *Genetics*, 165:2213–2233, 2003.
- [28] G.A. McVean, T.P. Awadella, and P. Fearnhead. A coalescent method for detecting recombination from gene sequences. *Genetics*, 160:1231–1241, 2002.
- [29] G.A. McVean, S.R. Myers, S. Hunt, P. Deloukas, D.R. Bentley, and P. Donnelly. The fine-scale structure of recombination rate variation in the human genome. *Science*, 304:581–584, 2004.
- [30] S.R. Myers and R.C. Griffiths. Bounds on the Minimum Number of Recombination Events in a Sample History. *Genetics*, 163:375–394, 2003.
- [31] SeattleSNPs. NHLBI Program for Genomic Applications, UW-FHCRC, Seattle, WA. <http://pga.gs.washington.edu>, October 2004.
- [32] D.A. Nickerson, S.L. Taylor, K.M. Weiss, A.G. Clark, R.G. Hutchinson, J. Stengard, V. Salomaa, E. Vartiainen, E. Boerwinkle, and C.F. Sing. DNA sequence diversity in a 9.7-kb region of the human lipoprotein lipase gene. *Nature Genetics*, 19:233–240, 1998.
- [33] N. Patil, A. J. Berno, D. A. Hinds, W. A. Barrett, J. M. Doshi, C. R. Hacker, C. R. Kautzer, D. H. Lee, C. Marjoribanks, D. P. McDonough, B. T. N. Nguyen, M. C. Norris, J. B. Sheehan, N. Shen, D. Stern, R. P. Stokowski, D. J. Thomas, M. O. Trulson, K. R. Vyas, K. A. Frazer, S. P. A. Fodor, and D. R. Cox. Blocks of limited haplotype diversity revealed by high resolution scanning of human chromosome 21. *Science*, 294:1719–1723, 2001.
- [34] Genomics of Cardiovascular Development, Adaptation, and Remodeling. NHLBI Program for Genomic Applications, Harvard Medical School. [http://cardiogenomics.med.harvard.edu/component-detail?project\\_id=240](http://cardiogenomics.med.harvard.edu/component-detail?project_id=240), October 2004.
- [35] Y.S. Song and J. Hein. Parsimonious Reconstruction of Sequence Evolution and Haplotype Blocks: Finding the Minimum Number of Recombination Events. In *In Algorithms in Bioinformatics, WABI 2003*, pages 287–302, 2003.
- [36] Y.S. Song and J. Hein. On the minimum number of recombination events in the evolutionary history of dna sequences. *Journal of Mathematical Biology*, 48:160–186, 2004.
- [37] M. Stephens, N. J. Smith, and P. Donnelly. A new statistical method for haplotype reconstruction from population data. *American Journal of Human Genetics*, 68:978–989, 2001.
- [38] A.R. Templeton, A.G. Clark, K.M. Weiss, D.A. Nickerson, E. Boerwinkle, and C.F. Sing. Recombinational and mutational hotspots within the human lipoprotein lipase gene. *American Journal of Human Genetics*, 66:69–83, 2000.
- [39] L. Wang, K. Zhang, and L. Zhang. Perfect phylogenetic networks with recombination. *Journal of Computational Biology*, 8(1):69–78, 2001.

## 9 Appendix

### 9.1 Proof of Theorem 2 (NP-completeness of computing $R_s$ )

We begin by stating the following trivial lemma.

**Lemma 5:** If a history were changed by moving a row  $r$  up,  $C_s(r)$  cannot increase.

We will exploit a simple gadget that ensures a partial ordering of rows. Consider a set of rows  $R$  as follows:

```

1 : 11111111
2 : 10000000
3 : 01000000
4 : 00100000
   ...
k : 00000001

```

Clearly, the best bound here is obtained by row 1 being the last row. The following lemma is trivial

**Lemma 6:** Consider a history  $H$  of  $R$  in which row 1 is the first row. Then,  $C_s(H) = k - 3$ .

**Proof:** Note that the  $C_s(H)$  is independent of column order, so by appropriate permutation of columns, every history in which row 1 is the first row is equivalent to the rows 1 through  $k$  appearing in order. It is easy to see that every row after row 3 gets a cost of 1. ♣

We reduce from MAX 2SAT. Consider a Max2SAT instance  $\phi(c, v)$  with  $c$  clauses and  $v$  variables. We will denote the  $2v$  literals as  $X_1, \bar{X}_1, X_2, \bar{X}_2, \dots, X_v, \bar{X}_v$ . We will construct special rows corresponding to the clauses, and the literals.

## 9.2 Clause rows

Each clause corresponds to 8 special rows. Figure 8 describes the structure, and the global structure is given in Figure 9. The 8 rows corresponding to clause  $C_j$  are all 0 except for the following columns. From  $8j - 7$  through column  $8j$ , the 8 rows correspond to the structure in Figure 8. Also, the 8 rows are all 1 from column  $8c + 1$  to column  $8c + v(c + 2)$ , and from column  $8c + v(c + 2) + 2v + 1$  to  $8c + v(c + 2) + 2v + 14vc$ .

## 9.3 Literal Rows

There are a total of  $2v(c + 2)$  rows, with  $c + 2$  rows for each literal. The  $c + 2$  rows are identical (denoted *copies* except in columns  $8c + 1$  to columns  $8c + v(c + 2)$ ).

Consider the  $k$ -th copy of literal  $X_j$ , and literal  $\bar{X}_j$ .

1. From columns  $8c + 1$  through  $8c + v(c + 2)$ , both rows have a 1 in column  $v(j - 1) + k$ , and a 0 everywhere else.
2. In columns  $8c + v(c + 2) + 1$  to  $8c + v(c + 2) + 2v$ : all copies of  $X_j$  are 1 in column  $8c + c(v + 2) + (2j - 1)$ , and 0 everywhere else. all copies of  $\bar{X}_j$  are 1 in column  $8c + c(v + 2) + (2j)$ , and 0 everywhere else.

3. Columns  $8c + v(c + 2) + 2v + 1$  to  $8c + v(c + 2) + 2v + 14vc$ : All literal rows are 1 in all of these columns.
4. Columns 1 to  $8c$ : If a variable does not appear in clause  $C_j$ , the two literal rows and all copies are 0 in columns  $8j - 7$  to  $8j$ . Otherwise, consider clause  $C_j = (X + Y)$ . The sub-matrix of rows corresponding to literals  $X, \bar{X}, Y, \bar{Y}$  and the columns  $8j - 7$  through  $8j$  is shown in Figure 10. Note that all copies again have identical values.

In addition, we have an extra set of  $14cv$  *early* rows, which are all 0 in the first  $8c + v(c + 2)$  columns. In columns  $8c + v(c + 2) + 1$  to  $8c + v(c + 2) + 2v$ , they are all 1's. In columns  $8c + v(c + 2) + 2v + 1$  through  $8c + v(c + 2) + 2v + 14vc$ , they form a diagonal of 1. See Figure 9 for the complete arrangement. All lines correspond to 0's, with 1's in appropriate places, and the boxed regions correspond to special arrangements as described earlier.

## A high level proof

A high level view of the proof is as follows. The early rows are designed to be the first set of rows. Their presence ensures that the first occurrence of a clause or literal row has a cost of 1, while their copies may have cost 0 or 1. The literal and the clause rows are designed so that in an optimum history, all clause rows come together, all literal copies come together, and for every variable  $X$ , either all rows corresponding to  $X$ , or all rows corresponding to  $\bar{X}$  are above the clause rows, and the complementary set is below the clause rows. Thus there is a one to one correspondence between an optimal history and an assignment of truth values to the variables. For a variable  $X$ , if  $X$  is above the clause rows, and  $\bar{X}$  below, it is set to true, else false. Finally, the clause rows are designed so that for every clause, the set of rows has a cost of 8 if neither of the literals is true, and 7 if at least one is.

**Lemma 7:** In any optimum history, the early rows are always the first set of rows.

**Proof:** Note that if all of the early rows were indeed the first set of rows, they would all have a cost 0, so the maximum cost of such a history would be the cost of the remaining  $8c + v(c + 2) < 12cv$ . This implies that the first  $cv$  rows must be early rows. If not, then at least  $13cv$  rows lie under a literal, or a clause row. By lemma 6, all of these early rows have a cost of 1, contradicting optimality.

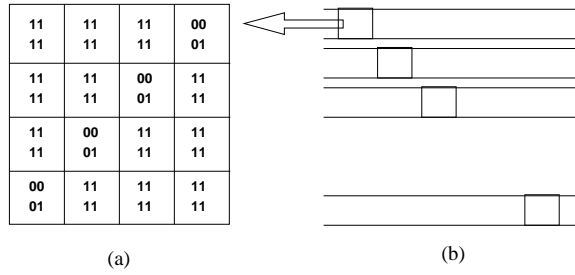


Figure 8: The 8 rows corresponding to a clause.

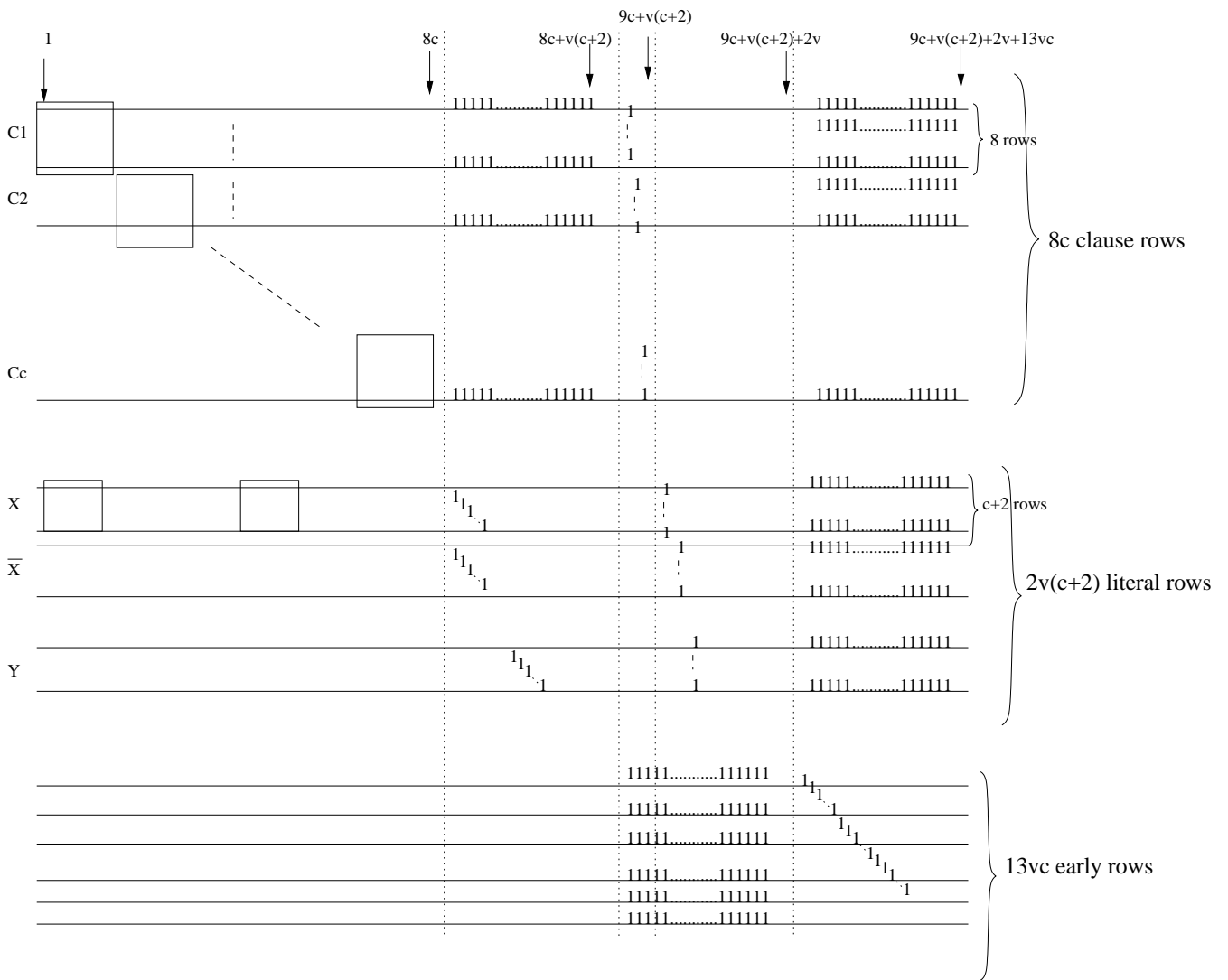


Figure 9: The global structure of the rows corresponding to an instance of Max2SAT.

$$C = (X+Y)$$

$X$	<b>11</b>	<b>00</b>	<b>00</b>	<b>00</b>	}	$c+2$ copies
	$\vdots$	$\vdots$	$\vdots$	$\vdots$		
	<b>11</b>	<b>00</b>	<b>00</b>	<b>00</b>		
$\bar{X}$	<b>00</b>	<b>11</b>	<b>11</b>	<b>11</b>		
	$\vdots$	$\vdots$	$\vdots$	$\vdots$		
	<b>00</b>	<b>11</b>	<b>11</b>	<b>11</b>		
$Y$	<b>00</b>	<b>00</b>	<b>00</b>	<b>11</b>		
	$\vdots$	$\vdots$	$\vdots$	$\vdots$		
	<b>00</b>	<b>00</b>	<b>00</b>	<b>11</b>		
$\bar{Y}$	<b>11</b>	<b>11</b>	<b>11</b>	<b>00</b>		
	$\vdots$	$\vdots$	$\vdots$	$\vdots$		
	<b>11</b>	<b>11</b>	<b>11</b>	<b>00</b>		

Figure 10: The 4 literal rows corresponding to a clause.

Suppose the first  $cv$  rows in an optimum history are early rows, but other early rows appear after literal or clause rows. Then, take any early row  $r$  that lies below a clause or a literal row, and move it to the top. The cost of  $r$  decreases from 1 to 0. Note that none of the early rows that have cost 0 are affected by this. Consider a clause or a literal row  $r'$  with cost 1. Moving  $r$  up can decrease the cost of  $r'$  only if  $r'$  was a copy of  $r_1$  in the non-trivial columns. As that is clearly not the case for any literal or clause row, the cost of clause rows does not decrease. Therefore, there is a net decrease in cost, contradicting optimality. ♣

**Theorem 8:** Consider an arbitrary literal row  $l$  in a history. Then  $C_s(l) = 1$  if and only if one of the following occurs:

1. None of the copies of  $l$  appears before it.
2. There is a clause row before it in history.
3. Either its complementary literal appears before it, or for every copy  $l_c$  above it, the complement of that copy also appears above it.

**Proof:** Consider a literal row  $l$ . If it is the first copy in this history, then note that the presence of the early rows keeps columns  $8c+v(c+2)$  through  $8c+v(c+2)+2v$  informative, and as it is different from the other literal/clause or early rows, it must get a cost of 1. Likewise, if it is after a clause row, then it will have a cost of

1. Finally, if  $l$  is not the first copy, then it differs from each of the copies  $l_c$  above it in only two columns, which occur between  $8c+1$ , and  $8c+v(c+2)$ . If the complement of  $l$ , or of  $l_c$  appear above  $l$ , then one of these two columns becomes informative. Thus if for every copy this is true, then  $l$  must get a cost of 1. If there is some copy  $l_c$  where this is not true, then after removing non-informative columns,  $l$  is identical to  $l_c$  and must have a cost 0. ♣

**Lemma 9:** Consider a history. If a literal corresponding to variable  $X$  is moved up or down, the cost of a literal row  $l$  corresponding to a different variable does not change.

**Proof:** Note that none of the conditions that determine the cost of  $l$  (Theorem 8) have changed. ♣

**Lemma 10:** The cost of the literal rows corresponding to a variable in any history is at least  $c+3$ .

**Proof:** From Theorem 8 the cost of the first occurrence of a distinct literal row, and the first occurrence of its complement is 1. The  $2(c+1)$  remaining copies can be paired into literal rows and their complements. From Theorem 8, one of the two must have a cost of 1. Thus the total cost is  $c+1+2$ . ♣

**Lemma 11:** A history  $H$  can be transformed in polynomial time into a history  $H'$  in which the clause rows are adjacent to each other, all literal copies are adjacent to each other, and  $C_s(H) \geq C_s(H')$ .

**Proof:** (By construction). Start with an optimum history  $H$ . Consider the first occurrence of a clause row  $r$ . Transform this history by bringing up all clause rows of the clause and insert them in order after row  $r$ . We need to show that

$$C_s(H') \leq C_s(H)$$

From lemma 5, the cost of the clause rows cannot increase. Also, from Theorem 8, the cost of the literal rows that follow  $r$  in the history is already 1.

For every variable  $X$ , consider the first occurrence of either of its two literals, and denote it as  $l$ . Move all copies of  $l$  to be below  $l$ . If  $l$  was below the clause rows, all copies of both literals of variable  $X$  as well as other affected literal rows have a cost of 1, so the cost does not increase. If  $l$  was above the clause rows, then it has a cost of 1, but all of its  $c + 1$  copies have a cost of 0. Thus the cost of that variable is at most  $2(c+2) - (c+1) = c+3$ , which is optimal (Lemma 10). From Lemma 9, none of the other literals are affected. Finally, clause rows are affected by literal rows, but do not change by adding or removing copies of literal rows. Next, consider the complement of  $l$ , and bring all of its copies together to be just below the first occurrence of a complementary row. As these rows already have a cost of 1, they do not contribute to an increase in total cost. Once again, none of the other literal rows or clause rows are affected. ♣

**Lemma 12:** Consider an optimum history in which all clause rows are together. The cost of rows for each of the clauses is either 7, or 8.

**Theorem 13:** Consider an optimum history with all clause rows adjacent to each other, and all literal rows adjacent to each other. For each variable  $X$ , either the rows corresponding to  $X$ , or to  $\bar{X}$  must lie above the clause rows.

**Proof:** Assume otherwise. Then, there exists a variable  $X$ , such that the rows corresponding to both  $X$ , and  $\bar{X}$  are below the clause rows. Together, their cost to the history is  $2(c+2)$ . Now move all of the  $c+2$  rows of  $X$  above the clause rows. From Theorem 8, the total cost of the  $c + 2$  copies of literal rows corresponding to  $X$  is now 1, thus reducing the net cost by  $c + 1$ . Even if all clause rows increased in cost from 7 to 8 because of this move, there is a decrease in total cost of at least 1, a contradiction. ♣

**Corollary 14** There exists an optimum history in which all clause rows occur together, all copies of literals occur together, and for each variable  $X$ , exactly one of  $X$  or  $\bar{X}$  appears above the clause rows.

**Proof:** From Theorem 13, at least one of  $X$  or  $\bar{X}$  appear above the clause rows. If both do, note that each row in the lower clause already has a cost of 1, and so we can move the lower literal down without changing the cost. ♣

**Theorem 15:** Consider an optimum history  $H$  of instance derived from a Max2SAT instance  $\phi(c, v)$ . Then, an assignment can be computed which will satisfy exactly  $v(c + 1) + 2v + 8c - C_s(H)$  clauses.

**Proof:** Transform  $H$  into an optimum history  $H'$  in which all clause rows are adjacent, and all literal copies are adjacent, and for each variable  $X$ , exactly one of  $X$  or  $\bar{X}$  appears above the clause rows. Clearly the total cost of the early rows is 0, those of the literal clauses, is  $v(c + 1) + 2$ . Consider the set of 8 rows for each clause.

Note that each set of clause rows scores a 7 if at least one of the two literals in it is above (or set to true) it, and 8 otherwise. Thus the cost of the clause rows is exactly 8 if the clause is not satisfied, and 7 if it is. The total cost is  $C_s(H') = C_s(H) = v(c + 3) + 8c - c'$ , where  $c'$  is the number of satisfied clauses. ♣

#### 9.4 Proof of Lemma 4

**Proof:** Let  $M'$  be the completion of  $M$  and  $H' = h_1 \rightarrow h_2 \dots \rightarrow h_{n'}$  be the history for  $M'$  which corresponds to the value  $R_s(M)$ . It also follows that,  $R_s(M') = C_s(H')$ . Now, every haplotype  $h$  in the incomplete matrix  $M$  is mapped to a haplotype in the complete matrix  $M'$ . Let  $n$  be the number of haplotypes in the matrix  $M$  and  $n'$  be the number of haplotypes in  $M'$  where  $n' \leq n$ . We denote the set of haplotypes in  $M$  whose completion is identical to haplotype  $h_i$  by  $C(h_i)$ . Note that,  $C(h_i) \geq 1$ ,  $1 \leq i \leq n'$ . Consider a history  $H = C(h_1) \rightarrow C(h_2) \dots \rightarrow C(h_{n'})$  for the incomplete matrix  $M$  where we ignore the ordering of the haplotypes within each set of haplotypes  $C(h_i)$ . We show that the following holds for every  $i$ ,  $1 \leq i \leq n'$ ,

$$R_{sm}[C(h_1), C(h_2) \dots C(h_i)] \leq R_s[h_1, h_2 \dots h_i]$$

The proof is by induction on  $i$ . The base case is trivial. Let us assume that

$$R_{sm}[C(h_1), C(h_2) \dots C(h_{i-1})] \leq R_s[h_1, h_2 \dots h_{i-1}]$$

holds. Since,  $H'$  is assumed to be the optimal history,  $R_s[h_1, h_2 \dots h_i]$  is either  $R_s[h_1, h_2 \dots h_{i-1}]$  or  $R_s[h_1, h_2 \dots h_{i-1}] +$

1 accordingly as  $C_s(h_i) = 0$  or 1. Since, all rows in the set of haplotypes  $C(h_i)$  are identical to each other, we have

$$R_{sm}[C(h_1), C(h_2) \dots C(h_i)] \leq R_{sm}[C(h_1), C(h_2) \dots h_i']$$

where  $h_i'$  is one of the haplotypes in  $C(h_i)$ . Note that this holds, since we choose the minimum over all possible redundant row removals in the algorithm for computing  $R_{sm}$ . Also,  $h_i'$  is identical to  $h_i$  treating the missing entries as don't cares. If  $C_s(h_i) = 0$ , then  $C_s(h_i')$  is also zero. If  $C_s(h_i) = 1$ ,  $C_s(h_i')$  can either be 0 or 1. Using the induction hypothesis, we get

$$R_{sm}[C(h_1), C(h_2) \dots C(h_{i-1}), h_i'] \leq R_s[h_1, h_2 \dots h_{i-1}, h_i]$$

Combining with the previous inequality, it follows that

$$R_{sm}[C(h_1), C(h_2) \dots C(h_i)] \leq R_s[h_1, h_2 \dots h_i]$$

