Efficient Algorithms for Constructing Phylogenetic Networks with Restricted Recombinations

Wei-Shun Su Tso-Ching Lee[*] Yaw-Ling Lin[†]

Dept. of Comput. Sci. and Info. Management, College of Computing and Informatics, Providence University, 200 Chung Chi Road, Shalu, Taichung, Taiwan 433.
E-mail: g9371011@cs.pu.edu.tw, yllin@pu.edu.tw

Abstract

A current high-priority phase of human genomics involves the development of a full Haplotype Map of the human genome [15]. It will be used in large-scale screens of populations to associate specific haplotypes with specific complex genetic-influenced diseases. Phylogenetic networks are models of sequence evolution that go beyond trees, allowing biological operations that are not tree-like. The problem is to find a phylogenetic network that derives an input set of sequences.

Recently Ding and Gusfield [4] propose a linear time algorithm for the perfect phylogeny tree *inference problem. In this paper, we generalize the linear time tree inference algorithm as the base case and propose efficient algorithms that infer* phylogenetic networks *satisfying given pre-specified types and/or conditions of recombination by using the* conflict table *data structure.*

Keywords: *bioinformatics, SNP, haplotype inference, phylogenetic network, recombination.*

1 Introduction

Building a Haplotype Map of the human genome has become a central NIH promoted goal [15]. The international Haplotype Map Project is focussed on determining the common SNP haplotypes in several diverse human populations. It is widely expected that the relation between specific haplotypes and genotypes (such as certain disease) will allow the rapid location of gene that influence those disease. However, collecting genotypes is cheaper and easier than collecting haplotypes. So it is the main approach that collecting genotype data and computationally inferring to haplotype data pairs.

1.1 SNP, Genotype, Haplotype

Mutation in DNA is the principle factor that is responsible for the phenotypic differences among human beings, and SNPs (*single nucleotide polymorphisms*) are the most common mutations. A SNP is defined as a position in a chromosome where each one of two (or more) specific nucleotides is observed in at least 10% of the population [18]. The nucleotides involved in a SNP are called *alleles*.

In diploid organisms, such as human, there are two "copies" of each chromosome. A description of the data from a single cope is called a haplotype, and the mixed data on the two copies is called a genotype. In complex disease (affected by more than single gene,) it is more informative to have haplotype data than to have genotype data. On the other hand, it is not feasible to examine the two haplotypes separately in general, and genotype data rather than haplotype data is usually obtained. Computational methods for inference of haplotype information from the observed genotype data are thus highly demanding in the current trend of computational biology.

We represent each of n genotypes as a vector, each with m site, where each value in a site is either 0,1 or 2. A site i in genotype vector g has a value 0 (respectively 1) if site i has value 0 (respectively 1) on both the underlying haplotypes (*homozygous site*), and has value 2 otherwise (*heterozygous site*).

2 The Phylogenetic Networks

Despite the existence of many well-studied efficient algorithms [9, 1, 5, 4] for inferring the (perfect)

[*]Address: Taichung Veterans General Hospital. 160, Sec. 3, Taichung Kang Road, Taichung, Taiwan 407. E-mail: tclee@vghtc.gov.tw
[†]Corresponding author. This work is supported in part by the National Science Council, Taiwan, R.O.C, grant NSC-94-2213-E-126-003 and PU-93-11100-5162-006-002.

phylogenetic tree haplotype information, the growth of genetic data reveals that much haplotyped (SNP) sequences do not fit the evolutionary tree (coalescent) models. A *phylogenic network* is a generalization of a phylogenetic tree, allowing structural properties that are not tree-like. In the phylogenetic networks, *recombination* is an important situation between the evolution. It means a chimeric sequence derived by combining another two gene sequences. In populations it is the key element underlying techniques that are widely hoped to locate genes influencing genetic diseases. In graph theory, it can be represented as an cycle in a undirect graph. The phylogenetic network problem (with recombination) was introduced by Hein [13, 14]. The goal is to construct a phylogenetic network that derives a given set of binary sequences with minimizing the number of recombinations used. The minimization criterion is motivated by the general utility of parsimony in biological problems, and because most evolutionary histories are thought to contain a small number of observable recombinations [10]. Unfortunately, the problem is shown to be NP-hard [19].

In a phylogenic network, there are four components needed to specify: a directed acyclic graph, an assignment of sites on edges, an assignment of a sequence to each non-recombination, an assignment of a recombination point, and a sequence to each recombination node. Figure 1 is an example for a phylogenetic network. The phylogenetic network generated by a matrix $M[n, m]$ is shown at the right. There are exactly n nodes without incoming edges. Each node except root has either one or two incoming edges. A node with two incoming edges is call a "recombination node". There are two labels P and S on the incoming edges of recombination node; P means the "prefix" and S means the "suffix" to the recombination sequence, and the number on the recombination node means the site of sequence that recombined.

The recombination events have some special or important biological meaning, and also are the requisite role in evolution of species. In this paper, we propose some algorithms to reconstruct the recombination sequences and the phylogenetic networks. First, we use the linear time algorithm proposed by Gusfield [4] to construct the *perfect phylogeny tree*, and then add the recombination sequences derived by our algorithms to form a phylogeny network. Given the biologically specified weights, the algorithm distinguish more important or meaningful sequences from the others; thus higher weighted sequences are considered as higher priority in processing the genotype matrix. The proposed phylogenetic network inference algorithms thus generate results that reflects these priorities.

3 The Linear-Time Perfect Phylogeny Haplotyping Algorithm (LPPH)

Given an input set of n genotype vectors of length m, the *Haplotyping Inference (HI)* problem is to find $2n$ binary haplotype vectors such that these n genotypes can be generated by the associated pairs of haplotype vectors. This would be impossible without the implicit or explicit use of some genetic model, either to assess the biological fidelity of any proposed solution, or to guide the algorithm in constructing a solution. The most powerful such genetic model is the population-genetic concept of a *coalescent* [20, 16]. The coalescent model is an evolution history of $2n$ haplotypes without recombination; it can be viewed as a rooted tree with $2n$ leaves, and each of the m sites labels exactly one edge on the tree. In computer science terminology, the coalescent model says that the $2n$ haplotypes fit a *perfect phylogeny*.

The PPH problem was introduced in [9] with a solution whose running time is $O(nm\alpha(nm))$. The nearly linear time algorithm is based on the linear-time reduction of the PPH problem to the graph realization problem. But the graph realization problem is some what complicate resulting a PPH algorithm hard to understand and implement. In [9], it is conjectured that the real linear-time algorithm to PPH problem should be possible.

Another two slower and easier methods in [1, 5] were based on "conflict-pairs" rather than graph models. The running time of both methods are $O(nm^2)$. Related research has examined extensions, modifications or specializations of the PPH Problem [2, 3, 6, 11, 17], or examined the problem when the data or solutions are assumed to have some special form [12, 8, 7].

Recently, a really linear-time algorithm for PPH problem was proposed in [4]. The algorithm makes no assumptions about the form of data or the solution. It is based on a directed, rooted graph, called "*shadow tree*". The algorithm uses some standard operations which could be executed rapidly to ensure the properties of the shadow tree. Furthermore, the algorithm has been fully implemented. The method provides an implicit representation of all the PPH solutions. The value of a linear-time algorithm for the PPH problem is partly conceptual and partly for use in the inner-loop of algorithm for more complex problem, where the PPH problem must be solved repeatedly.

3.1 The Shadow Tree

The shadow tree is a kind of data structure which can express distinct PPH solutions by flipping the class. In shadow tree, there are two types of edges: *tree edge* and *shadow edge*, which are both directed towards to root. The tree and shadow edges are labelled by columns from the genotype matrix M (with the

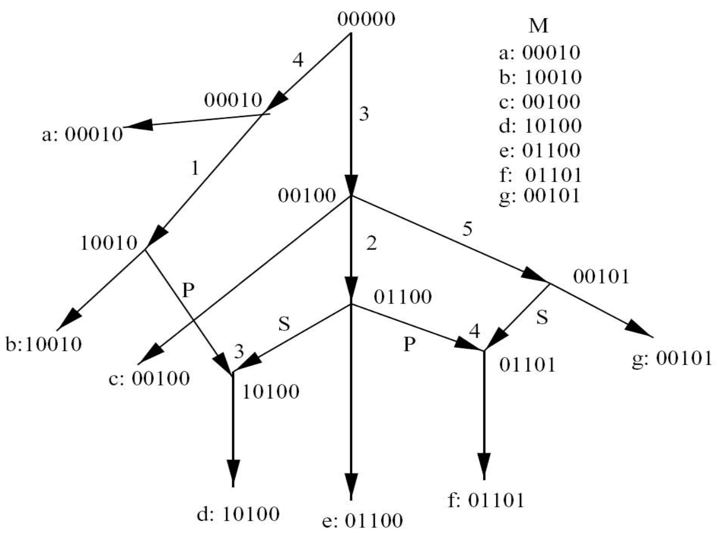

Figure 1: A phylogenetic Network with Two Recombination Nodes

shadow edge having bars over the labels for distinction). For each tree edge i, there is a shadow edge $\bar{i}$ in the shadow tree. The tree edge i means that the site i occurred a mutation in history. Relatively, the path from ancestor to the leaves which across the shadow edge $\bar{i}$ means the site i has no mutation in the historical path.

There is another important data structure in shadow tree, called "*links*". In the graph theory standpoint, the links are also edges in a tree, but in order to distinguishing between the tree (shadow) edges and links, the word "edge" is reversed for tree and shadow edges. However, there are also two types of links: *free link* and *fixed link*. Tree edges, shadow edges, and fixed links are organized into *classes*. In a *class*, the relation of position of these edges is fixed. A free link connects two classes, and a fixed link is contained in a single class.

There are three operations used to modify the shadow tree- *edge addition*, *class flipping*, and *class merging*. The three operations can be executed rapidly to ensure the properties of shadow tree.

3.2 Invariant Properties

The LPPH problem processes the input genotype matrix M one row a time, starting at the first row. At every step, the algorithm should ensure the correctness of the following properties and the running time of the a shadow tree.

Property 1 : For any column i in M, the edge labelled by i is in the shadow tree if and only if the shadow tree $\bar{i}$ is; i and $\bar{i}$ are in the same class, and are in different subtrees of the class (expect for the root class).

Property 2 : Each class (expect for the root class) attaches to exactly one other class, and the two join points are in different subtrees of the parent class unless it is the root class.

Property 3 : Along any directed path towards the root the column numbers of the edges (tree or shadow edges) strictly decrease. Also, for any two edges E and E', if E was added to the shadow tree while processing a row k, and E' was added when processing a row greater than k, then the E' can never be above E on a path to the root in the shadow tree.

According above properties, the algorithm can create shadow tree in linear time. The more details and proofs described in [4].

4 The Phylogenetic network algorithms

Prior to the LPPH algorithms, columns of the genotype matrix M are arranged according to decreasing leaf count, with the column containing the largest leaf count on the left, and the position of rightmost 1-entry in each row is arranged decreasingly, with the first row containing the rightmost 1-entry in M. According to the arrangement of M, LPPH algorithm can construct a larger shadow tree. The means of "larger" is more rows in M could be processed and the shape of shadow tree is larger.

In the course of processing the genotype matrix M containing the recombination sequences, LPPH algorithm stops at certain point and report a recombination occurred in the evolution history. The condition of recombination can be checked by the method, called *three-gamete test*. If any two columns i, j in haplotype sequences contain three rows with the pair $(1, 0)$,

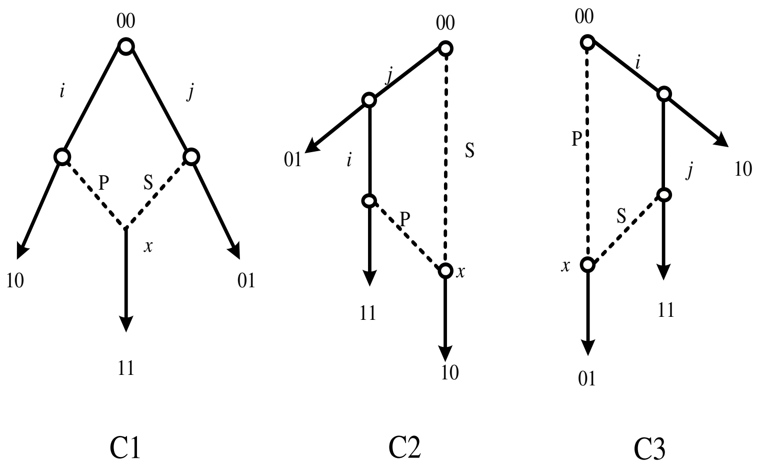

Figure 2: x is a recombination node, and we use a dotted line to represent the recombination. In type C1, site i and j mutate separately and recombine the sequence $1, 1$. In type C2, site j mutates before i does, the recombination sequence is $1, 0$. And in type C3, site i mutates before j, the recombination sequence is $0, 1$

$(0, 1)$, and $(1, 1)$, then the two columns are called *conflict*. The sequences containing conflict pairs imply recombination events. There are three cases of recombination, as illustrated in Figure 2.

Theorem 1 *Two sites i and j ($j > i$) is a conflict pair. There will be three recombination types in the phylogenetic networks.*

Note that the algorithm arrange the matrix M before peforming the LPPH algorithm; it follows that more useful sequences are processed (as coalescent sequences) earlier. If the recombination occurred, less important sequence would be detected and the algorithm stopped. So we can get a recombination sequence by executing the LPPH once, and this sequence is useless or less important than the other two sequences. We use a structure call “conflict table” to record all sequences and construct the recombination sequences in the following steps.

4.1 Conflict Table

The conflict table CT is maintained as a two dimensional array such that each entry is a *conflict link*. The rows of CT are indexed by three kinds of conflict pairs: $(0, 1)$, $(1, 0)$, and $(1, 1)$. Each column of CT is indexed by each pair of (i, j)-sites within the SNP sequences for each $1 < i < j < m$. Each value in CT is a link list to the row numbers of haplotype sequences. As an example, suppose the given haplotype matrix M is represented as the following.

$$M : \begin{bmatrix} 100 \\ 010 \end{bmatrix}$$

It follows that the corresponding conflict table for M is the following

CT(M)	(1,2)	(2,3)	(1,3)
10	1	2	1
01	2	0	0
11	0	0	0

Table 1

Note that the entries in CT are the linked list of row numbers. If certain recombination sequence add in CT, it must causes all values of certain column of CT are greater than 0. We put the column numbers into a set of recombination site, called “RS”. The recombination sequence can be construct by most $|RS| - 1$ recombing. For example, if the third row 110 added in M, it cause the recombination occurred. The table $CT(M)$ become:

CT(M)	(1,2)	(2,3)	(1,3)
10	1	2,3	1,3
01	2	0	0
11	3	0	0

Table 2

In column 1, the values of three rows are all greater than 0, the column pair is added into RS; thus $RS = \{1, 2\}$. The sequence of row 3 of M can be derived by recombing the two arbitrary sequences of column 1 and row $1, 2$ of CT. The LPPH algorithm stops when it encounters a recombination sequence; so we know which sequence is derived by recombing. We can delete this sequence from M, and restart the LPPH algorithm. Finally, a shadow tree for matrix $M \setminus R$ can be constructed, where R is the set of recombination sequences. We use LPPH algorithm to select recombination sequences, and the conflict table to construct these sequences; until finally produce a phylogenetic network after linking these recombined sequences on the initial perfect phylogeny tree derived by LPPH. It

```
PN($M, R$)
Input: A arranged genotype matrix $M$ and recombination sequence set $R$.
Output: The haplotype sequences $H$.
1  $LH \leftarrow$ LPPH($M \setminus R$)
2  if LPPH stop in row $i$ before finishing
3    then $R \leftarrow R + i$
4         PN($M, R$)      $\triangleright$ Restart this algorithm.
5  return CT($LH, R$)
```

Figure 3: The Phylogenetic Network Algorithm.

```
CT($LH, R$)
Input: The sequences $LH$ derived from LPPH and recombination sequences $R$.
Output: The haplotype sequences $H$.
1  $H \leftarrow LH$
2  ADD-CT($LH$)
3  for each row $k \in R$ do
4    ADD-CT($k$)
5    if $CT[n, (i, j)] > 0, \forall n \in CT$
6      then $RS \leftarrow RS + i + j$
7           $rs \leftarrow$ recombine two sequences of column $(i, j)$
8           $H \leftarrow H + rs$
9           ADD-CT($rs$)
10 return $H$

ADD-CT($S$)
Input: The sequences $S$.
1  for each row $k$ and column $i, j \in S, (\forall j > i)$ do
2    if $(S[k, i], S[k, j]) = (1, 0)$
3      then Insert $k$ into the $CT[(1, 0), (i, j)]$ linked list
4    elseif $(S[k, i], S[k, j]) = (0, 1)$
5      then Insert $k$ into the $CT[(0, 1), (i, j)]$ linked list
6    elseif $(S[k, i], S[k, j]) = (1, 1)$
7      then Insert $k$ into the $CT[(1, 1), (i, j)]$ linked list
```

Figure 4: The Conflict Table Algorithm.

is easily verified that the total time complexity for getting a perfect phylogeny tree and those recombination sequences is $O(mn^2)$. The time needed for constructing the recombination sequences is $O(m^2n)$. Thus the running time of our phylogenetic networks algorithm is $O(mn^2 + m^2n)$.

4.2 The Phylogenetic Network Algorithm

Our phylogenetic network algorithm is shown in Figure 3. The input is genotype matrix M and recombination sequences R, R is $\varnothing$ initially. The basic idea is to utilize LPPH algorithm into producing a perfect phylogeny tree for sequences without recombination. Later on, the conflict table is used to determine sequences and sites implying recombinations. First, we use the LPPH algorithm to deal with the matrix M. Once the algorithm stops in row i before finishing, it means row i is a recombination sequence. We add i into recombination matrix R in line 3 and restart the PN algorithm. Note that the input of LPPH algorithm is matrix $M \setminus R$ in second running; thus the row i is excluded as an input of LPPH. We use the CT procedure to construct the recombination sequences.

In the conflict table algorithm CT shown in Figure 4, we first call ADD-CT to record all sequences of LH into conflict table. If the value pair of row k and column i, j is $(0, 1)$, add the number k in to the link list of $CT[(0, 1), (i, j)]$. There is no conflict situation occurred now. Then, the steps 3 to 9, record the sequence one by one of the recombination matrix R and determine the conflict situation. If the all values of certain column (i, j) in CT (one column) are greater than 0, that means the conflict occurred in the three sequences of column (i, j) and separate rows (10), (01), and (11). We put the number i, j into RS, and construct the sequence rs by recombining some sequences. For example, there is a conflict situation in Table 2, and the 3rd row can be constructed by recombining the 1st and

2nd sequences at site $1, 2$. Finally, add rs into matrix H and return the sequences matrix H.

In the phylogenetic network algorithm, we use LPPH algorithm to construct the sequences with no recombination and determine the recombination sequences. Then use the conflict table to determine the sequences and site which recombination occurred and reconstruct the recombination sequences.

5 The Priority Sequences

It is possible to associate each sequence of the given genomic data with a weight as an indication of the biological importance, probability of coalescent (non-recombinant,) or reliability of each obtained genotype sequence. These factors can be useful and considered in the construction of haplotype inference. Thus we can arrange the genotype matrix M by adding the priority of each row, and arrange M by decreasing the priority, with the highest priority sequence on the first row. Then process one row at a time, start at first row. This method can avoid the important sequences be removed form M. For an example, we consider four sequences in a genotype matrix M. The order of processing of LPPH algorithm is A, B, C, D, respectively, as shown in the following.

$$M: \begin{array}{c} A \\ B \\ C \\ D \end{array} \begin{bmatrix} 1100 \\ 0011 \\ 0110 \\ 1110 \end{bmatrix}$$

Sequence C is conflicted with A and B on sites pair $(2, 3)$, and sequence B is conflict with A and D on $(1, 3)$ or $(2, 3)$. In LPPH, B and C sequences are determined as the recombination sequences; however, other different possibilities exist. For example, if the priority of these four sequences is D, A, B, C. By processing the sequences according the priority, the sequence B will be determined a recombination sequence, but the next sequence C is allowed.

	LPPH	Priority
non-recombination	A, D	A, C, D
recombination	B, C	B

Table 3

Although the LPPH algorithm can determine the more important sequences A and D, but the number of non-recombination sequences is less than the priority method. The method retains the meaningful sequences and determines the recombination sequences which conflicted with these sequences.

5.1 The Phylogenetic Network Algorithm with Priority Sequences

In the line 1 to 3 of phylogenetic network algorithm with priority sequences shown in Figure 5, we add the weight of priority for each sequence and arrange the matrix M with priority. The priority may be the meaning of biology or the relationship for some special disease or populations. Then, the following steps are the same with the PN algorithm. In this algorithm, the more important sequences would be proceeded by LPPH earlier, relatively the chance to be retained is higher than others.

6 Conclusion and Further Work

In this paper we propose phylogenetic network inference algorithm by using the LPPH algorithm to determine a maximal non-recombinant phylogenetic sequences in building a perfect phylogenetic tree as the base case. In producing the shadow tree corresponded to genotype matrix, it can also determine sequences that is constituted through the recombination event. We can use the conflict table to find the sequences and sites causing recombination and construct the recombination sequence, and the time complexity is $O(mn^2 + m^2n)$. As a variant of our phylogentic network algorithm, our algorithm also deals with sequences with different weights that reflect different meaning or information, importance, or reliability. In such case, we consider priorities of sequences and retain the most sequences in the course of computing.

Most of common genetic related diseases are caused by more than one gene. The distance between

```
PSPN($M$)
Input: The genotype matrix $M$.
Output: The haplotype sequences $H$.
1  for each row $i \in M$ do
2      Add the weight of priority or importance on row $i$
3  Sort the matrix $M$ by decreasing the degree of priority
4  $H \leftarrow$ PN($M, \varnothing$)
5  return $H$
```

Figure 5: The Phylogenetic Network Algorithm with Priority Sequences.

those genes may be very far, and the possibility of recombination between those genes is higher. For setting up the course of evolving, the situation of recombination should be considered. In the further, we will collect the data for the specific disease in some true families, and try to compute the possible evolutionary history for the various specific diseases.

References

[1] V. Bafna, D. Gusfield, G. Lancia, and S. Yooseph. Haplotyping as perfect phylogeny: A direct approch. *J. Computational Biology*, 10:323–340, 2003.

[2] T. Barzuza, J.S. Beckmann, R. Shamir, and I. Pe'er. Computational problem in perfect phylogeny haplotyping: Xor-genotypes and tag snp's. In *Proceedings of CPM*, 2004.

[3] P. Damaschke. Fast perfect phylogeny haplotype inference. In *14th symp. on Fundamentals of Comp. Theory FCT'2003*, volume 2751, pages 183–194, 2003.

[4] Z. Ding, V. Filkov, and D. Gusfield. A linear-time algorithm for perfect phylogeny haplotyping. In *Proceedings of RECOMB 2005*, 2005.

[5] E. Eskin, E. Halperin, and R.M. Karp. Efficient reconstruction of haplotype structure via perfect phylogeny. *J. Computational Biology*, 1(1):1–20, 2003.

[6] E. Eskin, E. Halperin, and R. Sharan. Optimally phasing long genome regions using local haplotype predictions. In *Proceedings of the second RECOMB Satellite Workshop on Computational Method for SNPs and Haplotypes*, Pittsburg, USA, Feburary 20-21 2004.

[7] J. Gramm, T. Nierhoff, and T. Tantau. Perfect path phylogeny haplotyping with missing data is fixed-parameter tractable. In *The First international Workshop on Parametrized and Exact Computation (IWPEC 2004)*, Bergen, Norway, September 2004.

[8] J. Gramm, T. Nierhoff, T. Tantau, and R. Sharan. On the complexity of haplotyping via perfect phylogeny. In *The Second RECOMB Satellite Workshop on Computational Method for SNPs and Haplotypes*, Pittsburg, USA, Feburary 20-21 2004.

[9] D. Gusfield. Haplotyping as perfect phylogeny: Conceptual framework and efficient solution (extended abstract). In *Proceedings of RECOMB 2002: The Sixth Annual International Conference on Research in Computational Molecular Biology*, pages 166–175, 2002.

[10] D. Gusfield, D. Hickerson, and S. Eddhu. An efficiently computed lower bound on the number of recombinations in phylogenetic networks: Theory and empirical study. Technical report, To appear *Discrete Applied Math*, Special Issue on Computational Biology, 2005.

[11] E. Halperin and E. Eskin. Haplotype reconstruction from genotype data using imperfect phylogeny. *Bioinformatics*, 20:1842–1849, 2004.

[12] E. Halperin and R.M. Karp. Perfect phylogent and haplotype assignment. In *Prceedings of RECOMB 2004: The Eighth Annual International Conference on Research in Computational Molecular Biology*, pages 10–19, 2004.

[13] J. Hein. Reconstructing evolution of sequences subject to recombination using parsimony. *Math. Biosci*, 98:185–200, 1990.

[14] J. Hein. A heuristic method to reconstruct the history of sequences subject to recombination. *J. Mol. Evol*, 36:396–405, 1993.

[15] L. Helmuth. Genome research: Map of the human genome 3.0. *Science*, 293(5530):583–585, 2001.

[16] R. Hudson. Gene genealogies and the coalescent process. *Oxford Survey of Evolutionary Biology*, 7:1–44, 1990.

[17] G. Kimmel and R. Shamir. The incomplete perfect phylogeny gaplotype problem. In *Proceedings of the second RECOMB Satellite Workshop on Computational Method for SNPs and Haplotypes*, Pittsburg, USA, Feburary 20-21 2004.

[18] N. Patil, A. J. Berno, D. A. Hinds, et al. Blocks of limited haplotype diversity revealed by high resolution scanning of human chromosome 21. *Science*, 294:1719–1723, 2001.

[19] L. Wang, K. Zhang, and L. Zhang. Perfect phylogenetic networks with recombination. *Journal of Computational Biology*, 8:69–78, 2001.

[20] C. Wiuf. Inference onrecombination and block structure using unphased data. *Genetics*, 166(1):537–545, January 2004.