Efficient Reconstruction of Phylogenetic Networks with Constrained Recombination

Dan Gusfield and Satish Eddhu
Department of Computer Science
University of California, Davis
{gusfield,eddhu}@cs.ucdavis.edu

Charles Langley
Division of Evolution and Ecology
University of California, Davis
chlangley@ucdavis.edu

Abstract

A phylogenetic network is a generalization of a phylogenetic tree, allowing structural properties that are not tree-like. With the growth of genomic data, much of which does not fit ideal tree models, there is greater need to understand the algorithmics and combinatorics of phylogenetic networks [10, 11]. However, to date, very little has been published on this, with the notable exception of the paper by Wang et al.[12]. Other related papers include [4, 5, 7]

We consider the problem introduced in [12], of determining whether the sequences can be derived on a phylogenetic network where the recombination cycles are node disjoint. In this paper, we call such a phylogenetic network a "galled-tree". By more deeply analysing the combinatorial constraints on cycle-disjoint phylogenetic networks, we obtain an efficient algorithm that is guaranteed to be both a necessary and sufficient test for the existence of a galled-tree for the data. If there is a galled-tree, the algorithm constructs one and obtains an implicit representation of all the galled trees for the data, and can create these in linear time for each one. We also note two additional results related to galled trees: first, any set of sequences that can be derived on a galled tree can be derived on a true tree (without recombination cycles), where at most one back mutation is allowed per site; second, the site compatibility problem (which is NP-hard in general) can be solved in linear time for any set of sequences that can be derived on a galled tree.

The combinatorial constraints we develop apply (for the most part) to node-disjoint cycles in any phylogenetic network (not just galled-trees), and can be used for example to prove that a given site cannot be on a node-disjoint cycle in any phylogenetic network. Perhaps more important than the specific results about galled-trees, we introduce an approach that can be used to study recombination in phylogenetic networks that go beyond galled-trees.

1 Introduction to phylogenetic networks and galled-trees

With the growth of genomic data, much of which does not fit ideal tree models, and the increasing appreciation of the genomic role of such phenomena as recombination, recurrent and back mutation, horizontal gene transfer, gene conversion, and mobile genetic elements, there is greater need to understand the algorithmics and combinatorics of phylogenetic networks [10, 11]. Recombination is particularly important, because it is the key element needed for techniques that are widely hoped to locate genes influencing genetic diseases. The key to locating these genes is to understand and use the patterns of recombination in the genetic "experiments" done by nature and history. However, to date, very little has been published on phylogenetic networks, with the notable exception of the paper by Wang et al.[12]. Other related papers include [4, 5, 7].

1.1 Formal definition of a phylogenetic network

There are four components needed to specify a phylogenetic network: a directed acyclic graph (no directed cycles, but the underlying undirected graph can have cycles); an assignment of mutations or sites (integers) to edges; an assignment of a sequence to each non-recombination node; an assignment of a recombination point and a sequence to each recombination node. We will define each of these components in turn. See Figure 1 for an example of a phylogenetic network.

An (n, m)-phylogenetic network N is built on a directed acyclic graph containing exactly one node (the root) with no incoming edges, a set of internal nodes that have both incoming and and outgoing edges, and exactly n nodes (the leaves) with no outgoing edges. Each node other than the root has either one or two incoming edges. A node x with two incoming edges is called a "recombination" node.

Each integer (site) from 1 to m is assigned to exactly one edge in N, but for simplicity of exposition, none are assigned to any edge entering a recombination node.

Each node in N is labeled by an m-length binary sequence, starting with the root node which is labeled with the all-0 sequence. Since N is acyclic, the nodes in N can be topologically sorted into a list, where every node occurs in the list only after its parent(s). Using that list, we can constructively define the sequences that label the non-root nodes, in order of their appearance in the list, as follows:

> a) For a non-recombination node v, let e be the single edge coming into v. The sequence labeling v is obtained from the sequence labeling v's parent by changing from 0 to 1 the value at position i, for every integer i assigned to edge e. This corresponds to a mutation at site i occurring on edge e.
>
> b) Each recombination node x is associated with an integer r_x (denoted r, when x is clear by context) between 2 and m inclusive, called the "recombination point" for x. For the recombination at node x, one of the two sequences labeling the parents of x must be designated P and the other designated S. Then the sequence labeling x consists of the first $r_x - 1$ characters of P, followed by the last $m - r_x + 1$ characters of S. Hence P contributes a *Prefix* and S contributes a *Suffix* to x's sequence. The resulting sequence that labels x is called a "recombinant sequence".

The sequences labeling the leaves of N are the extant sequences, i.e., the sequences that can be observed.

Definition 1.1 *An (n, m)-phylogenetic network N derives (or explains) a set of n sequences M if and only each sequence in M labels exactly one of the leaves of N. We use the terms "site" and "column" interchangeably.*

The biological interpretation of a phylogenetic network N that derives M is that N is a possible history of the evolution of the sequences in M, under the assumptions that there is a single, known ancestral sequence (assumed to be all-0 for convenience); that for any site in the sequences there is exactly one point in the history where that state of that site mutates (due to a point-mutation) from 0 to 1; and that two sequences are permitted to recombine in an equal-crossover event. Each site in the sequence represents a SNP (single nucleotide polymorphism), i.e., a site where two of the four possible nucleotides appear in the population with a frequency above some set threshold. With these definitions, a classic perfect phylogeny is a phylogenetic network which is topologically a directed, rooted tree, i.e., lacking any cycles in the underlying (undirected) graph.

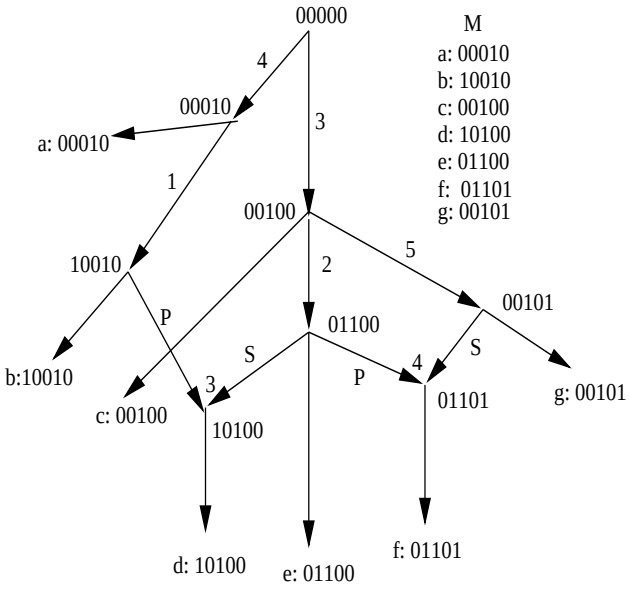

Figure 1. A phylogenetic network N with two recombination nodes. The matrix of sequences M that are derived by N is shown at the right. Note that the node with sequence label 01100 is sequence S for the left recombination node, and is sequence P for the right recombination node. The recombination points are 3 and 4 for the left and right recombination nodes respectively, and are written just above the recombination nodes . In this example, every label of an interior node also labels a leaf, but that is not a general property of phylogenetic networks.

Interest in phylogenetic networks comes partly from a desire to reconstruct the evolutionary history of a set of molecular sequences under a model that is more complete than the perfect phylogeny (tree) model. But there also more applied uses of phylogenetic networks. For example, in a population of "unrelated" individuals, we want to determine which parts of the individuals genomes came from a common ancestor. This determination helps locate regions in the genome associated with genes contributing to an observable trait (for example, a disease). Recombination in the population is key to this determination, and understanding the history of the recombinations is the key to doing this kind of mapping.

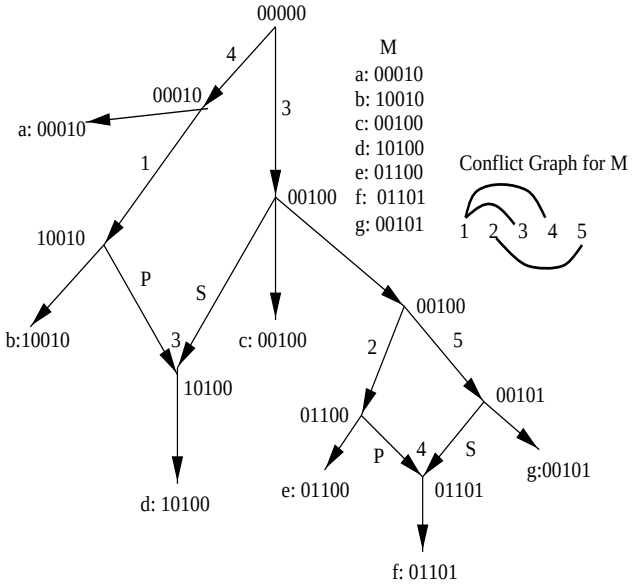

Figure 2. A galled-tree deriving the same sequences as the phylogenetic network in Figure 1. Unlike the example shown here, in general the recombinant sequence exiting a gall may be on a path that reaches another gall.

1.2 Which Phylogenetic Networks are Biologically Informative?

It is easy to show that for every binary matrix M, there is a phylogenetic network N that derives M using $\Theta(nm)$ recombination nodes, but that is not of great interest because in most evolutionary histories the number of recombinations is thought to be relatively small (on the order of the number of mutations). Hence a more biologically informative problem is to find, for input M, a phylogenetic network that generates M, and that either has some biologically-motivated structure, or uses the *minimum* number of recombinations. We call that number m_M. Wang et al. [12] showed that the general problem of computing m_M is NP-hard, and Hudson and Kaplan [6] and Myers and Griffiths [9] give combinatorial methods for computing lower-bounds on m_M.

1.2.1 Galled-trees: A biological and algorithmically motivated structural restriction

Given the NP-hardness of the problem of computing m_M, Wang et al. suggested a structural restriction on the permitted phylogenetic networks which has both biological and algorithmic appeal.

Definition 1.2 *In a phylogenetic network N, let w be a node that has two paths out of it that meet at a recombination node x. Those two paths together define a "recombination cycle" Q. Node w is called the "coalescent node" of Q, and x is the recombination node of Q.*

Definition 1.3 *A recombination cycle in a phylogenetic network that shares no nodes with any other recombination cycle is called a "gall" (imagine a wasp's gall in a tree). We say a site i "appears" on a gall Q if i labels one of the edges of Q. We use the term "recombination cycle" for general phylogenetic networks.*

Definition 1.4 *A phylogenetic network is called a "galled-tree" if every recombination cycle is a gall. See Figure 2.*

A galled-tree defines a phylogenetic history where the recombination cycles are node-disjoint, using at most $m/2$ recombinations. A phylogenetic network is likely to be a galled-tree if the level of recombination is moderate, or if most of the observable recombinations are recent. In Human populations, both conditions are believed to hold. Other examples of galled-trees arise in the data reported in [8]. The simplest case is when we study an interval in the genome where only a single recombination has occurred; the true history of the sequences in that interval takes the form of a galled-tree. More generally, it is important (in disease association studies, for example) to find regions of the genome where the subsequences in a population exhibit moderate recombination, and the galled-tree algorithm can be used to find such regions. We will show that when sequences can be derived on a galled-tree, the galled-tree is "essentially-unique". Thus, if the sequences did derive historically on a galled-tree, the algorithm will correctly capture the essential history of the sequences.

Further motivation for galled-trees comes from the fact that if M can be derived by a galled-tree, then it can be derived by a true tree (no underlying undirected cycles) with at most one back mutation per site. A tree with limited back mutations is another model of interest that deviates from the perfect phylogeny model.

Galled-Tree Problem: Given a set M of n binary sequences, each of length m, determine if there exists a galled-tree T that derives M, and if there is one, construct one.

Wang et al. [12] give an $O(nm+n^4)$-time algorithm that was intended to solve the Galled-Tree Problem. This work is seminal as it is the first paper to introduce a biologically motivated structural restriction for a phylogenetic network that allows a polynomial time algorithm. Unfortunately, the algorithm in [12] is incorrect, and only provides a sufficient condition for the existence of a galled-tree for M.

Main Result Here we develop a faster algorithm ($O(nm+n^3)$-time) that completely solves the Galled-Tree

Problem. We also show that if there is a galled-tree for M, then all galled-trees for M use the same number of recombinations, which we conjecture, but have not proved, is m_M. Our "canonical" solution in "essentially-unique", minimizes the number of sites on the recombination cycles, and can be used to count and produce all the galled-trees for M.

In obtaining these results, we develop combinatorial constraints that apply to galls in any phylogenetic network (whether a galled-tree or not). This is useful as a first step in understanding phylogenetic networks in general, and for specific tasks, such as proving that a given site cannot be on any gall in any phylogenetic network. We also show that if M can be derived by a galled-tree, then it can be derived by a true tree (without underlying cycles) with at most one back mutation per site, and that the problem of removing the minimum number of sites of M, so that the remaining sites have a perfect phylogeny (an NP-hard problem in general) can be solved in linear time.

2 Combinatorial definitions and observations

We organize M into a matrix, where each row contains a sequence in M, and assume there are no duplicate columns. We also assume for simplicity of exposition that there are no duplicate columns, and that each column has at least one entry that is 1.

2.1 Combinatorial Background and Major Combinatorial Tool

Definition 2.1 *Two columns (or sites) in M are said to "conflict" if and only if the two columns contain three rows with the pairs 1,1; 0,1; and 1,0. A site is called "conflicted" if it is involved in at least one conflict, and is otherwise called "unconflicted".*

Recall that a perfect phylogeny is a phylogenetic network without recombinations. Hence, as a graph, it is a directed rooted tree. The following is the classic necessary and sufficient condition for the existence of a perfect phylogeny deriving a set of sequences M. See [2, 3] for one exposition.

Theorem 2.1 *There is a perfect phylogeny deriving M if and only if matrix M contains no conflicting sites. Further, if there is a perfect phylogeny for M and all columns of M are distinct, then there is a unique perfect phylogeny for M, and each edge is labeled by at most one site. If there are identical columns, then the perfect phylogeny is unique up to any ordering given to multiple sites that label the same edge.*

Hence it is the existence of conflicts in M that require a deviation from the perfect phylogeny model, and in this paper, require recombinations in order to derive a history of M.

Major Tool: The Conflict Graph and its Connected Components

The central contribution of this paper is to observe that there is combinatorial structure in the pattern of conflicts between columns, and that this structure can be represented and exploited to obtain insights about recombination in phylogenetic networks. We now introduce the *conflict graph*, which represents and exposes some of the combinatorial structure.

Definition 2.2 *The conflict graph G contains one node for each site in M. We label each node of G by the site it represents. Two nodes i and j are connected by an undirected edge if and only if sites i and j conflict. See Figure 2.*

Overview: The connected components of G are particularly important. We will show that there is a one-one correspondence between the non-trivial connected components of G and the galls in a galled-tree: more generally, every gall in any phylogenetic network contains all the sites of one (non-trivial) connected component, and contains no sites from another (non-trivial) connected component. Further, no gall need contain any unconflicted sites. It follows that every galled-tree for M uses the same number of galls, and the same number of recombinations.

2.2 Combinatorial Constraints on Galls

In order to prove the claims made in the overview, we next begin an examination of the combinatorial constraints on galls and galled-trees. We state the needed lemmas and theorems but omit the longer proofs for lack of space.

Lemma 2.1 *Let Q be a gall in a phylogenetic network N and v be a node on Q. Define N' as the subnetwork of N consisting of all nodes and edges reachable by directed paths from v, not using any edges in Q, i.e., the maximal subnetwork branching off of Q at v. If site i appears on Q, then the state of site i at every node in N' is the same as at node v.*

Proof Suppose that at some node in N', the state of i is different than it is at v. Let x' be such a node with the property that at every ancestor of x' in N', the state of i is the same as at node v. Since i only mutates once (on Q), the state of i cannot change in N' due to mutation, and can therefore only change due to recombination. Hence, x' must be a recombination node. Now if both parents of x' were in

N', then by the choice of x', the state of i at both parents would be the same as the state at v, and that state would be unchanged at x' regardless of where the recombination point r_x is. So one of the parents of x', call it p, must be outside of N'. Now consider a path from p back towards the root, and let w be the first ancestor of v reached on this path. Note that w could be v, and in that case, the path also intersects a descendent of v on Q. But the path from w to x' through p, together with the path from w to x' through v, forms a recombination cycle that shares at least one edge with Q, contradicting the assumption that Q is a gall. So the state of site i at every node in N' must be the same as at node v. □

Definition 2.3 *Let C be a set of sites on a gall Q, and let the matrix $M(C)$ be matrix M restricted to the sites in C. Given a phylogenetic network for M, let $S_v(C)$ denote the sequence labeling node v, restricted to the sites in C.*

Lemma 2.1 implies the following

Corollary 2.1 *A sequence is in $M(C)$ if and only if it is the sequence $S_v(C)$ for some node v on Q. Stated differently, the node labels at nodes on Q, restricted to sites in C, are exactly the sequences in $M(C)$.*

Proof The sequences in $M(C)$ are the sequences labeling the leaves of N, restricted to C. If a leaf z is reachable from a node v in Q, not using an edge in Q, then by Lemma 2.1, $S_z(C)$ and $S_v(C)$ are the same. If leaf z is not reachable from any node v in Q, then it must have state 0 for every site i that mutates on Q. In that case $S_z(C)$ is all zeros, which is $S_w(C)$, where w is the coalescent node of Q. □

Corollary 2.1 is important because it says that information about the (interior) node labels on any gall is reflected in some sequences at the leaves, and hence that is contained in extant sequences. This is a property of galls that does not generalize to every non-gall recombination cycle, and is intuitively one of the reasons why problems concerning galls and galled-trees have efficient solutions.

Definition 2.4 *A node v on a recombination cycle Q is called a "branching node" if there is a directed edge (v, v') where v' is not on Q.*

The following theorem is the technical key to most of the analysis of the combinatorial structure of galled-trees. It is proven by case analysis, which we omit.

Theorem 2.2 *Let T be a galled-tree for matrix M. Two sites i and $j > i$ in M conflict if and only if the following conditions hold:*

a) i and j are together on the same gall (call it Q) in T, with recombination node x, and $i < r_x \leq j$.

b) Sites i and j are arrayed on Q in one of the following three ways (see Figure 3):

W1: Site i is on the P-side and j is on the S-side of Q, and there is a branching node between i and x, and a branching node between j and x. Note: In this case, the i, j state-pair in the recombinant sequence is 1,1.

W2: Sites i and j are both on the P-side with j above i (i.e., j mutates before i does), and there is a branching node between j and i, and a branching node between i and x. In this case the i, j state-pair in the recombinant sequence is 1,0.

W3: Sites i and j are both on the S-side with i above j, and there is a branching node between i and j, and a branching node between j and x. The state-pair in this case is 0,1.

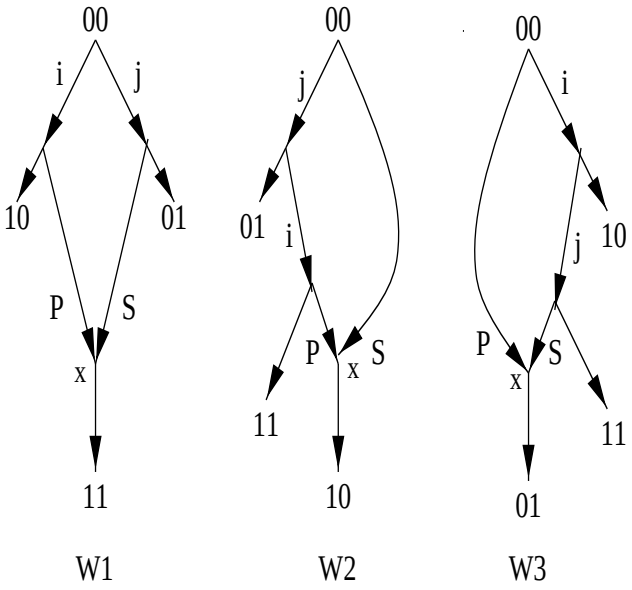

Figure 3. The three cases for Theorem 2.2. In each case, the recombination point x_r is between i and j.

The algorithm in [12] is only a sufficient test for the existence of a galled-tree that explains M, because it (implicitly) assumes that a pair of sites can conflict only due to arrangement W1. Equivalently, the algorithm in [12] correctly determines whether or not the input sequences can be generated on a galled-tree T having the added constraint: for each site i, if site i mutates on an edge e, then the state of i remains set at 1 at all nodes which are reachable from

the end of e. Hence once the state of i mutates from 0 to 1, it never returns to 0, even through the action of recombination. That is a severe restriction compared to what is allowed by the general definition of a galled-tree. In the galled-tree in Figure 2, the state of site 4 mutates from 0 to 1, but then is returned to 0 through recombination in the gall shown on the left.

We now state the theorems leading to the one-one correspondence between connected components in the conflict graph for M and the galls in a galled-tree for M.

Theorem 2.3 *For any non-trivial connected component C of the conflict graph, and any galled tree T for M, all the sites in C must occur together on a single gall in T.*

Proof This follows by transitivity from the necessary direction of part a) of Theorem 2.2, and the fact that for any pair of sites i and j in C, there must be a path connecting i to j in C. □

The following theorem is the complement to Theorem 2.3. It greatly constrains the structure of any galled tree for M and simplifies the development of an efficient algorithm to find a galled tree for M.

Theorem 2.4 *Let T be a galled-tree for M. If sites i and i' are on different non-trivial connected components of the conflict graph, then they must appear on different galls of T.*

We prove Theorem 2.4 by using the following lemma, which is of interest in its own right.

Lemma 2.2 *Let Q be a gall in T with recombination node x, and recombination point r, and let i, i', j, j' be sites on Q, where i conflicts with $j > i$ and i' conflicts with $j' > i'$. Then either i conflicts with j', or i' conflicts with j.*

Proof of Theorem 2.4: Let i and $j > i$ be conflicting sites on one non-trivial connected component, and i' and $j' > i'$ be conflicting sites on another non-trivial connected component. If these four sites are all together on a single gall Q, then with respect to the recombination point r of that gall, i and i' are below r, and j and j' are each equal to or above r. So by Lemma 2.2, either i conflicts with j' or i' conflicts with j. But that contradicts the assumption that i and j are on a different connected component of the conflict graph than are i' and j'. Hence i and j are on one gall and i' and j' are on another. But by Theorem 2.3, all sites on the same connected component are together on a single gall, so any two sites on two different connected components are on different galls. □

Theorems 2.3 and 2.4 together imply a one-one correspondence between the non-trivial connected components of the conflict graph and the galls in a galled-tree: more generally, every gall in any phylogenetic network contains all the sites of one (non-trivial) connected component, and contains no sites from another (non-trivial) connected component.

So far, we have only addressed conflicted sites on a gall. The next theorem addresses unconflicted sites.

Theorem 2.5 *Let N be a phylogenetic network with a gall Q that contains unconflicted sites, We can transform N to another phylogenetic network N' with the same number of recombinations, where the unconflicted sites of Q are moved to edges incident with Q (after which Q contains only conflicted sites or no sites), the internal arrangement of any conflicted sites on Q is the same in N and N', and all other details of N remain the same. If the remaining gall contains no sites, it can be contracted to a single node.*

Corollary 2.2 *If there is a galled-tree for M, then there is a galled-tree where the number of recombinations is exactly the number of connected components of the conflict graph, which is the minimum number of recombinations that any galled-tree can have.*

In the remainder of the paper, whenever we assume the existence of a galled-tree T for M, we assume, without stating it, that the galls of T only contain conflicted sites. Further, until Section 5, unless stated otherwise we assume that all sites are involved in some conflict, i.e., we completely ignore unconflicted sites.

3 Arranging the gall Q_C

The one-one correspondence between non-trivial connected component and galls in a galled-tree greatly simplifies the task of creating a galled-tree for M. We can focus independently on each non-trivial connected component C of the conflict graph, to determine how the sites on that component are arrayed on the gall Q_C, and how to select the recombination point for Q_C. In this section we show how to efficiently accomplish these tasks.

3.1 Selecting the recombination point r on Q_c

Lemma 3.1 *If there is a galled-tree for M, then every non-trivial connected component C of the conflict graph must be bipartite, and the bipartition is unique: the (indices of the) sites on one side of the bipartite graph must be strictly smaller than the sites on the other side.*

Lemma 3.1 gives a necessary condition that can be used to prove that certain sets of sequences cannot be derived on a galled-tree. For example, see Figures 4 and 5.

Proof All the sites on C must mutate on a single gall Q, and Q has only a single recombination point r. By Theorem

2.2a), $i < r \leq j$ for any conflicting pair i, j in C where $i < j$. Therefore, each edge in C connects one site whose index is below r and one site whose index is at or above r. □

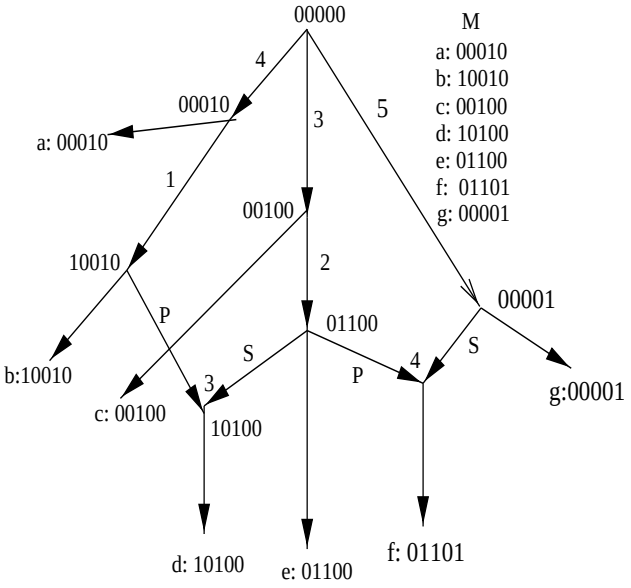

Figure 4. The phylogenetic network generates the sequences M shown to the right. Although this network is similar to the one shown in Figure 1, and only sequence g is different, M cannot be derived on a galled-tree. See Figure 5.

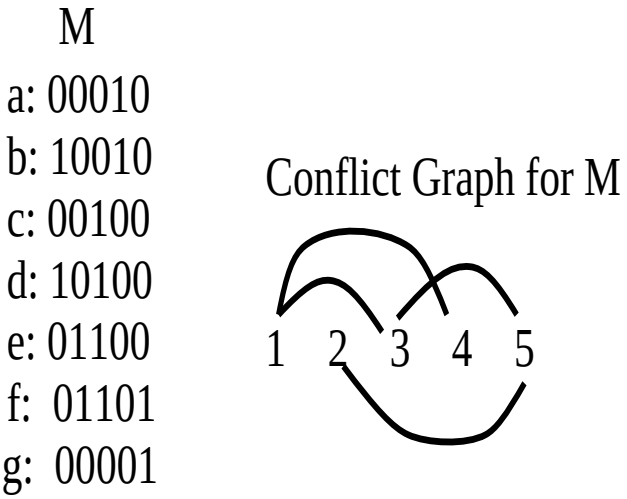

Figure 5. The conflict graph of the sequences M from Figure 4 is bipartite, and the bipartition is unique, but it does not have the required properties stated in Lemma 3.1.

Definition 3.1 *Given the bipartite graph for C, we call the side containing the smaller sites the L_C (left) side, and the other side the R_C (right) side.*

It is easy to find the bipartition and select r: let p be the largest node (site) in C which is connected only to larger nodes, and let q be the smallest node in C which is connected only to smaller nodes. Then r can be chosen to be any integer strictly larger than p and less or equal to q, and this defines L_C and R_C.

By a much more detailed analysis of the combinatorial structure of galls we can prove a stronger result than Lemma 3.1:

Theorem 3.1 *Let N be an arbitrary phylogenetic network for M. The sites in a connected component C can appear on a gall in N only if C is a bipartite graph with the bipartition described in Lemma 3.1, and C is a* **bi-convex** *graph. A bipartite graph is bi-convex if the nodes of the graph can be renumbered so that for any node v, the set of nodes that v is adjacent to form a contiguous interval in the new node numbers.*

This is a very useful, general theorem since it allows us to identify more connected components whose sites cannot appear on a gall in any phylogenetic network. One can determine if a graph is bi-convex in polynomial time and also find a minimum node cover of a bi-convex graph in linear time. It is easy to see that the minimum number of columns to remove from M so that no conflicts remain, is given by the minimum node cover of the conflict graph. This is called the "site consistency" problem, and it is NP-hard in general [1]. However, the node cover problem can be solved in polynomial time (by network flow) on any bipartite graph. So when there is a galled-tree for M, the site-consistency problem can be solved by network flow in polynomial time, and even faster by exploiting the fact that each connected component must be bi-convex.

3.2 Arranging the sites of C on Q_C

We now describe how to arrange the sites of C on a gall Q_C. Corollary 2.1 will be a central tool.

To understand the method for arranging the sites on a gall, consider a fixed galled-tree T for M, and focus on the arrangement of sites of C on gall Q_C in isolation of the rest of T. Now remove the recombination node x from Q_C, and the two edges entering x The resulting graph consists of one or two directed paths starting at the coalescent node of Q_C, and containing all the sites in C. If it only contains one path, then denote the coalescent node as w, and the single end node as u; otherwise let u and w be the two end nodes of the two paths. For each node v other than the coalescent

node, add an edge from v branching off of Q_C, and label its leaf end with $S_v(C)$. The result is a perfect phylogeny, denoted $T(C)$, that of course, derives the sequences labeling the leaves of $T(C)$. Further, $S_x(C)$ can be formed by a recombination of the sequences $S_u(C)$ and $S_w(C)$ at the recombination point r determined from C.

By Corollary 2.1, the leaf labels of $T(C)$ are exactly the sequences in $M(C)$, other than the sequence $S_x(C)$. That is, $T(C)$ is a perfect phylogeny for all the sequences in $M(C)$, other than $S_x(C)$. Hence, we have

Theorem 3.2 *There is a sequence X in $M(C)$, such that after removal of all copies of X, there is a perfect phylogeny for the resulting matrix; the labeled edges of that perfect phylogeny contain all sites in C organized into one or two paths; and the recombination of the two "end" sequences (either the root sequence and the single leaf sequence, or the two leaf sequences) at the recombination point r, creates sequence X.*

Now, by Theorem 2.1, any matrix which has a perfect phylogeny has a unique perfect phylogeny as long as no ordering is given to multiple sites on edges. Hence, given M and C, if we could guess X, we could create the correct, unique, perfect phylogeny and exactly recreate the arrangement of sites on Q_C as given in T.

However, since we do not know X, if we remove all copies of different sequence Y, and yet there is a (unique) perfect phylogeny for the resulting matrix, where all the sites in C are contained in one or two paths, and the recombination of the two end sequences at point r creates Y, then this new perfect phylogeny can also be used to arrange the site on Q_C. To see this, note that the node labels on both arrangements are exactly the same: restricted to C, both arrangements label the nodes with the sequences in $M(C)$, and only sites in C appear on Q_C. The state of each site not in C is identical at every node on Q. Hence, the two arrangements of sites on C induce a permutation of the nodes on Q_C and of the edges out of Q_C, but are indistinguishable outside of Q_C. Note that in all arrangements on Q_C, the sequence labeling the coalescent node contains only zeros at the sites in C. Hence, all arrangements of the sites on Q_C can be found by the following algorithm:

Site-Arrangement Algorithm

1) Let $M(C)$ be matrix M restricted to the sites in C.

2) For each distinct sequence X in $M(C)$ do: let $M'(C)$ be $M(C)$ after the removal of all rows with sequence X. Check if there is a (unique) perfect phylogeny for $M'(C)$, and if so, check if all sites on C are contained in one or two paths whose end sequences can be recombined at point r to create X. If the answer is "yes", then output an arrangement of the sites on Q_C consistent with this perfect phylogeny.

Specializing to galled-trees, we have established,

Theorem 3.3 *Assuming that there is a galled-tree for M, every arrangement of sites on Q_C that is used in some galled-tree, can be found as above. The set of sequences labeling nodes of Q_C, restricted to C, is invariant over all the arrangements, and all the galled-trees for M.*

Time Analysis: Given the matrix $M'(C)$ with n rows and at most $|C|$ columns, the perfect phylogeny algorithms in [2, 3] can determine if there is a perfect phylogeny for $M'(C)$, and construct it, in $O(n|C|)$ time. So, all the arrangements of Q_C that are used in any galled-tree for M can be found in $O(n^2|C|)$ time, and over all the galls, the time to find all the arrangements that appear on any galled-tree is $O(n^2m)$. It was established in [12] that m can be at most $2n$ if there is a galled-tree for M, so the total time is $O(n^3)$.

By a more detailed analysis of how the sites on a gall can be arranged, we have developed an alternative algorithm for arranging the sites on a gall, whose running time is only $O(n^2)$. However, the details require more space and are conceptually more involved than the approach discussed here.

We have claimed that the galled-tree for M is "essentially-unique". The one-one correspondence between connected-components and galls is the first part of that claim. We now show that the number of arrangements of the sites on a gall is very limited, further establishing the "essential-uniqueness".

Theorem 3.4 *Let C be a (non-trivial) connected component of the conflict graph whose sites can be arranged on a gall Q in a phylogenetic network for the input. The sites in C can be arranged on Q in at most three distinct ways.*

Proof We have already established that each distinct arrangement of sites on Q is associated with one distinct sequence X in $M(C)$, with the property that when all copies of X are removed from $M(C)$, the remaining sequences in $M(C)$ can be generated on a unique perfect phylogeny. Hence, when all copies of X are removed from $M(C)$, all conflicts between pairs in C are broken, and we can bound the number of distinct arrangements on Q by bounding the number of sequences in $M(C)$ whose removal breaks all conflicts among pairs in C.

Let i, j be a conflicting pair of columns in C. In order for the removal X to break the i, j conflict, the row for X in $M(C)$ must contain one of the three state-pairs 0,1 or 1,0 or 1,1 in columns i, j, and no other row in $M(C)$ can contain that state-pair in columns i, j. It follows that there can be at most three rows in $M(C)$ whose removal can break the i, j conflict, and hence there can be at most three distinct arrangements of the sites in C on Q. □

We can strengthen this result to show that the sites in C be arranged on Q in three ways only when C has two sites,

as in Theorem 2.2. Otherwise, Q can only be arranged in two ways. Moreover, if C has at least two sites that are below and two sites that are above the recombination point (any one for C will do), then the sites in C can be arranged on Q in only one way. We omit the proofs due to space limitations.

4 Connecting the galls in a galled-tree

Now we explain how to connect the galls together into a single galled-tree. Let T be a particular galled-tree for M and let Q and Q' be two galls in T. Gall Q is an "ancestor" of a gall Q' in T if there is a directed path in T from some node on Q to the coalescent node of Q'. If neither gall is an ancestor of the other, then we say that they are "incomparable". The algorithm to connect the galls will first deduce the ancestry relations between pairs of galls. We will see that the ancestry relations are invariant over all the galled-trees for M.

Since T is a particular galled-tree for M, the arrangement of sites on Q and Q' is determined. In that arrangement, let f_P and f_S be the first sites on the P and S sides respectively on Q. Define f'_P and f'_S similarly for Q'. Assume, without loss of generality, that f_S and f'_S exist. The analysis is symmetric for the other three combinations, one of which must exist. Also, let i, j be a pair of sites on Q that conflict with each other. Note that at the recombination node for Q, the state of at least one of i or j is set to 1, say i. Note that i might be f_P or f_S. Similarly, there is a site i' that has state 1 at the recombination node for Q'.

Now let Z' be any row of M with a 1 in column f'_S (there must be one since f'_S is involved in a conflict). If Q is an ancestor of Q' then Z' must have a 1 in at least one of the columns for f_S, f_P or i. Similarly, let Z be any row of M with a 1 in column f_S. If Q' is an ancestor of Q then Z must have a 1 in at least one of the columns for f'_S, f'_P or i'. So Q and Q' are incomparable if and only if neither of these conditions hold for rows Z' and Z.

With the proper data structure for M, rows Z' and Z can be found in constant time, and in constant time we can check those (up to six) entries in rows Z' and Z. So in constant time, we can determine whether Q and Q' are incomparable or not. If comparable, then we have found a row which has a 1 in a column q for a site that appears on Q and a 1 in a column q' for a site that appears on Q'. That means that there is a path from the root of T that passes through both the edges where sites q and q' mutate. We claim that Q is an ancestor of Q' if and only if site q has strictly more 1's in its column than does column q'. To see this, note first that by Lemma 2.1, site q appears before q' on the path if and only if site q has a 1 in every row where site q' has a 1. Moreover, for any conflicted site on a gall, there must be at least two nodes on that gall where that site has state 1, so a tie for the largest number of 1's in columns q and q' is not possible. Hence,

Theorem 4.1 *In constant time we can determine if Q and Q' are comparable, and if so, determine which is the ancestor of the other. There are at most $O(m) = O(n)$ galls, so over all the pairs of galls, we can determine the ancestry relations in $O(n^2)$ time.*

A gall Q is called the "immediate ancestor" of a gall Q' in T if Q is an ancestor of Q' and no descendent of Q is an ancestor of Q'. Every gall in T that has an ancestor in T, has a unique immediate ancestor, and the ancestor relation computed above is the transitive closure of the immediate-ancestor relation. Hence, given a fixed arrangement of the sites on each gall, to find the immediate-ancestor (if any) of each gall, we find the transitive-reduction of the ancestor relation. This can be done in $O(n^2)$ time because each gall has a unique immediate ancestor (we leave the details to the reader).

This identifies for each gall Q', its immediate ancestor Q in T, or determines that Q' has no ancestor. Since we are ignoring unconflicted sites, every site appears on some gall, so in T a gall is connected to its immediate ancestor by a single edge (rather than a path). If Q is the immediate ancestor of Q', we next want to determine the specific node, call it $v(Q, Q')$, on Q which is connected by a single edge to the coalescent node of Q' in T.

Let i, j be a conflicting pair on Q and let F be a sequence in M with a 1 for f'_S or f'_P. We claim that the 0/1 state of the i, j pair at recombination node x of Q is found at no other node on Q. Hence $v(Q, Q')$ is node x if and only if F has the same 0/1 state for i, j that is found at x. If that determination finds that $v(Q, Q')$ is not x, then F must have a 1 for exactly one of f_P or f_S, which identifies the side of Q that $v(Q, Q')$ is on. We then walk from the coalescent node of Q along that side until either encountering the edge e into the recombination node, or encountering the first edge e containing a site i such that F has state 0 for i. Node $v(Q, Q')$ is the node at the head of edge e. Since each of the $O(n)$ sites is on at most one gall, the total time to find all the these nodes is $O(n)$.

We let $\bar{T}$ denote the digraph determined to this point, i.e., consisting of all the arranged galls connected by the immediate ancestry edges.

Now the above exposition and determination of ancestry relations was based on assuming a particular arrangement of sites on each gall. But from Theorem 3.3, different arrangements of the sites on a gall merely permute the positions of the nodes, and the branching edges attached to them. This clearly does not change the ancestry and immediate ancestry relations. Therefore, we can use any permitted arrangement of the sites on the galls to determine $\bar{T}$, and

Theorem 4.2 *The digraph $\bar{T}$ is unique up to the different permitted arrangements of nodes inside the galls.*

Theorem 4.2 is a reflection of the "essential uniqueness" of the galled-trees that derive M.

Corollary 4.1 *Ignoring the issue of how to place the unconflicted sites, if there are k connected components of the conflict graph, and the sites on component C_q can be arranged in A_q different ways, then the number of galled-trees for M is exactly $\Pi_{q=1}^{q=k} A_q$.*

Algorithmically, Theorem 4.2 implies

Corollary 4.2 *Given the galls, and an arrangement of the sites on the galls, $\bar{T}$ can be determined in $O(n)$ time. Further, if there is a galled-tree for M, any $\bar{T}$ determined at this point can be extended to a galled-tree for M, by placing the unconflicted sites on edges of $\bar{T}$ between galls, and possibly adding new edges containing unconflicted sites, or new edges leading to leaves.*

5 Adding the leaf sequences and the unconflicted sites

To finish constructing the galled-tree for M, we must extend $\bar{T}$ by adding in any unconflicted sites, and place the sequences of M at specific leaves, possibly adding additional tree edges outside of any galls.

First, let Z be a sequence that has state 1 for a conflicted site i. Then in any galled-tree T for M, the leaf labeled with Z must be below the gall containing i. Conversely, any sequence labeling a leaf in T below a gall, must have a 1 for at least one site on that gall. Therefore, we can divide the sequences into those that have at least one 1 for a conflicted site, and those that don't. The sequences in the second set (if any) must be derivable on a unique perfect phylogeny that must be the upper part of any galled-tree for M. We can efficiently construct that perfect phylogeny, and determine where each gall in $\bar{T}$ resides relative to that perfect phylogeny (we leave the details to the reader).

Next, for each sequence Z in the first set of sequences, we will find the node v_Z in $\bar{T}$ such that in any galled-tree for M, v_Z is the last node in $\bar{T}$ on the path from the root to leaf Z. To do this, we do a bottom up traversal of $\bar{T}$, only traversing a gall after traversing all its descendents. At the start of the bottom up traversal, all sequences in M are unmarked. Let i, j be a conflicting pair that appears on a gall Q. We traverse gall Q as follows. Declare the recombination node x to be node v_Z for every unmarked sequence which has the same i, j states as in x; mark all of those sequences. Then traverse one side of Q, and for each site p encountered (just above a node v), declare node v to be v_Z for each unmarked sequence Z which has state 1 for site p; mark all of those sequences. Then traverse the other side of Q in a similar manner. The traversal takes $O(n^2)$ time and finds the node v_Z for each sequence Z in M. For each Z in the first set, we extend an edge from v_Z to a leaf labeled with Z. This places all the leaves for the sequences in M either in a perfect phylogeny above $\bar{T}$, or at a leaf connected by an edge to a node in $\bar{T}$.

Let $\bar{T}'$ denote the digraph constructed to this point. For exposition purposes we further modify $\bar{T}'$ as follows. For a node v on a gall Q, let V be the set of nodes not on Q that are immediate descendents of v. If $|V| > 1$, create a new edge from v to a new node v', and connect v' to every node in V. The effect is that every node v on a gall will have only one edge branching off the gall from v.

Next, with another bottom-up traversal, label each *edge* e with (the index of) one sequence $Z(e)$ such that node $v_{Z(e)}$ is below e. In case there is more than one to choose from, choose arbitrarily.

Now we turn to the issue of adding in the unconflicted sites. Sites that are part of the upper perfect phylogeny above $\bar{T}$ need no further attention. For any other unconflicted site i, do the following: Find a sequence Z in M which has a 1 for site i, and start a walk from the leaf labeled with Z towards the root of $\bar{T}'$. Let v be a node entered along an edge e' during this traversal, and let p be its parent node. If v is on a gall Q, then jump to the coalescent node of Q and continue the walk from there. If v is not on any gall, then examine the sequence $Z(e)$ for each edge e out of v other than e'. If none of these sequence have a 1 for site i, then place site i on edge e'. If every one of the sequences have a 1 for site i, then continue the upward walk from v. If some of the sequences have a 1 for site i, and some do not, then subdivide the edge (p, v) by creating a new node p' between p and v. Disconnect from v every edge e whose sequence $Z(e)$ does not have a 1 for site i, and reconnect e to node p'. Then place the site i on the (p', v) edge.

The time for placing the unconflicted sites is $O(n)$ per site, so $O(n^2)$ overall. We leave the proof of correctness to the reader.

6 Time bound and Correctness

All of the results given above assume the existence of a galled-tree for the input M. These results imply the correctness of the algorithm derived from them, when there is a galled-tree for M. When there isn't one, the algorithm either will not be able to execute a required step, or it will run to completion producing some galled-tree. At termination, we check whether or not the galled-tree derives M, and if not, correctly report that there is no galled-tree for M.

The overall time bound for the algorithm is the minimum of $O(nm+n^3)$ and $O(nm^2+n^2)$ time. If $m > 2n$, then we

first find and remove all duplicate columns in $O(nm)$ time. If the number of remaining columns is more than $2n$ then M has no galled tree [12]. Then we build the conflict graph in $O(n^3)$ time. Alternatively, if $m < 2n$ we directly build the conflict graph from M in $O(nm^2)$ time. Thereafter, all steps of the algorithm take $O(n^2)$ time.

7 Relation to the back-mutation model

Another deviation from the perfect phylogeny model that is of interest is to allow a limited number of back-mutations, but no recombinations. A back-mutation is a mutation from state 1 back to state 0 that occurs on an edge, i.e., it is not a change due to recombination.

Theorem 7.1 *Any set of sequences M that can be derived by a galled-tree, can be derived by a true tree (no recombinations and hence no underlying undirected cycles) with at most one mutation and one back-mutation per site.*

Proof We take a galled-tree T for M and transform each gall Q separately, so that no cycles remain, but all the node labels are preserved. The simplest case is that Q has one side, say S, which has no mutations (sites). Remove the S-side (which consists of just a single edge into x) from Q. Let p denote the P-side parent of x. Then for any site i which has state 1 at p, but has state 0 at x, write a back-mutation for i on the (p, x) edge. Hence Q no longer is a cycle, but all the node labels on Q remain unchanged. The more complex case is that both the S and P sides have at least one mutation. In this case, remove the first edge on Q out of the coalescent node, on either the P or the S-side, say the S-side, and reverse the direction of all the remaining edges on the S-side. Next, for every site i that has state 1 at p but state 0 at x, write a back-mutation for i on the (p, x) edge. For every site i that has state 0 at p but state 1 at x, write the mutation i on edge (p, x). Let s denote the parent of x on the S-side of Q. For every site i that has state 1 at s, but state 0 at x, write the mutation i on the (x, s) edge (which now runs from x to s). For every site i that has a state 1 at x, but state 0 at s, write the back-mutation for i on the (x, s) edge. Finally, convert each original mutation on a remaining edges of the S-side to a back mutation. The result is that Q is no longer a cycle, but all the node labels are preserved. Processing each gall in this way creates a true true that derives M using at most one back-mutation per site. See Figure 6 for an example. □

8 One Provable Lower Bound on m_M

Hudson and Kaplan [6] and Myers and Griffiths [9] give methods for computing lower-bounds on m_M. The methods in [9] seem very promising and may do well in practice, but

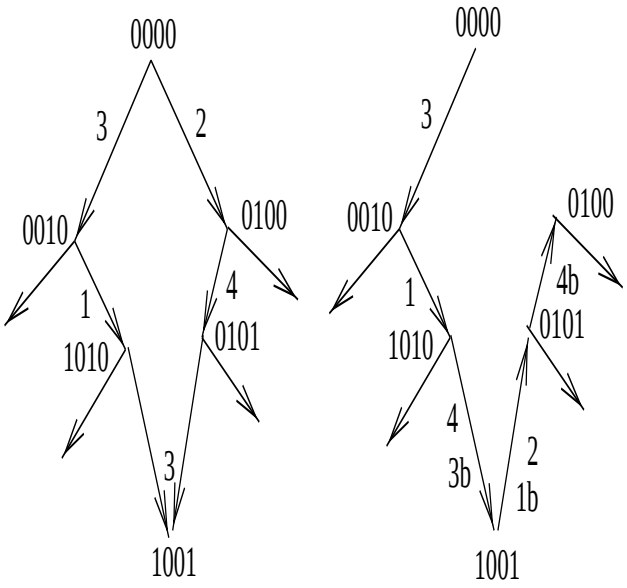

Figure 6. Gall Q is shown on the left and the result of the transformation is shown on the right. The recombination point for Q is 3, written above the recombination node. A number written on an edge is a mutation; a number followed by the letter b denotes a back-mutation.

what has been proven about lower bound methods is very limited. In particular, no existing (efficient) lower-bound method has even been proven to have the property that it can always correctly determine if $m_M > 1$, i.e., if more than one recombination is needed. While this is a modest result, the algorithm in this paper does provably have that property. A phylogenetic network using just one recombination is a galled-tree, and so the conflict graph must consist of a single component. In that case, the algorithm in this paper will construct a galled-tree with a single recombination. Conversely, if the algorithm cannot build a galled-tree for M, or cannot build one with just a single recombination, then m_M must be at least two.

9 Future Work and Open Questions

Future Work: The key ideas introduced in this paper are the one-one correspondence of connected components of the conflict graph and galls in a galled-tree, and the fact that the sites on a connected component C can appear on a gall in any phylogenetic network only if C is a bi-convex graph (with additional structure as detailed earlier). More generally, properties of constrained phylogenetic networks more complex than galled-trees can also be elucidated through

structural properties of the conflict graph. We are presently developing that viewpoint, and those results will be reported in a subsequent paper.

Open questions There are many open questions. The most immediate is: When M can be derived on a galled-tree, is the number of recombinations used m_M?

Acknowledgments Research partially supported by NSF grant EIA-0220154. We would like to thank Katherine St. John for LaTex advice.

References

[1] W. H. Day and D. Sankoff. Computational complexity of inferring phylogenies by compatibility. *Syst. Zool.*, 35:224–229, 1986.

[2] D. Gusfield. Efficient algorithms for inferring evolutionary history. *Networks*, 21:19–28, 1991.

[3] D. Gusfield. *Algorithms on Strings, Trees and Sequences: Computer Science and Computational Biology*. Cambridge University Press, 1997.

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

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

[6] R. Hudson and N. Kaplan. Statistical properties of the number of recombination events in the history of a sample of DNA sequences. *Genetics*, 111:147–164, 1985.

[7] J. D. Kececioglu and D. Gusfield. Reconstructing a history of recombinations from a set of sequences. *Discrete Applied Math.*, pages 239–260, 1998.

[8] J.Z. Lin, A. Brown, and M. T. Clegg. Heterogeneous geographic patterns of nucleotide sequence diversity between two alcohol dehydrogenase genes in wild barley (Hordeum vulgare subspecies spontaneum). *PNAS*, 98:531–536, 2001.

[9] S. R. Myers and R. C. Griffiths. Bounds on the minimum number of recombination events in a sample history. *Genetics*, 163:375–394, 2003.

[10] D. Posada and K. Crandall. Intraspecific gene genealogies: trees grafting into networks. *Trends in Ecology and Evolution*, 16:37–45, 2001.

[11] M. H. Schierup and J. Hein. Consequences of recombination on traditional phylogenetic analysis. *Genetics*, 156:879–891, 2000.

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