The Multi-State Perfect Phylogeny Problem with Missing and Removable Data: Solutions via Integer-Programming and Chordal Graph Theory

Dan Gusfield

Department of Computer Science, University of California, Davis
gusfield@cs.ucdavis.edu

Abstract. The *Multi-State Perfect Phylogeny Problem* is an extension of the *Binary* Perfect Phylogeny Problem, allowing characters to take on more than two states. In this paper we consider three problems that extend the utility of the multi-state perfect phylogeny model: **The Missing Data (MD) Problem** where some entries in the input are missing and the question is whether (bounded) values for the missing data can be imputed so that the resulting data has a multi-state perfect phylogeny; **The Character-Removal (CR) Problem** where we want to minimize the number of characters to remove from the data so that the resulting data has a multi-state perfect phylogeny; and **The Missing-Data Character-Removal (MDCR) Problem** where the input has missing data and we want to impute values for the missing data to *minimize* the solution to the resulting Character-Removal Problem.

We detail Integer Linear Programming (ILP) solutions to these problems for the special case of three permitted states per character and report on extensive empirical testing of these solutions. Then we develop a general theory to solve the MD problem for an *arbitrary* number of permitted states, using chordal graph theory and results on minimal triangulation of non-chordal graphs. This establishes new necessary and sufficient conditions for the existence of a perfect phylogeny with (or without) missing data. We implement the general theory using integer linear programming, although other optimization methods are possible. We extensively explore the empirical behavior of the general solution, showing that the methods are very practical for data of size and complexity that is characteristic of many current applications in phylogenetics. Some of the empirical results for the MD problem with an arbitrary number of permitted states are very surprising, suggesting the existence of additional combinatorial structure in multi-state perfect phylogenies.

Keywords: computational biology, phylogenetics, perfect phylogeny, integer programming, chordal graphs, graph triangulation.

1 Introduction and Background

In the **k-state Perfect Phylogeny Problem**, the input is an n by m matrix M whose values are integers from the set $Z(k) = \{1, 2, ..., k\}$. We refer to each

S. Batzoglou (Ed.): RECOMB 2009, LNCS 5541, pp. 294–310, 2009.

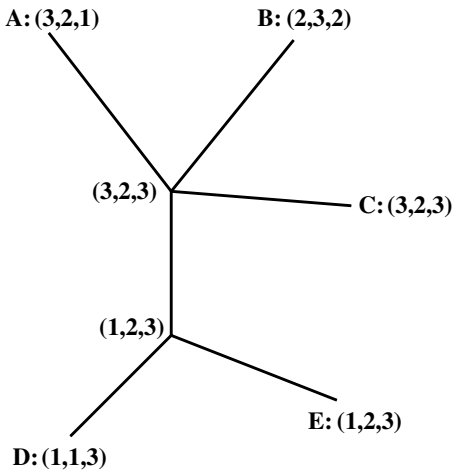

Fig. 1. A three-state perfect phylogeny with $n = 5, m = 3$. The input M consists of the five vectors that label the leaves of the tree. The subtree $T_3(3)$ contains the leaves labeled C,D,E, and the two interior nodes.

row of M as a *taxon* (plural *taxa*), to each column of M as a *character*, and to each value in a column c as a *state* of character c. A *k-state Perfect Phylogeny* for M is a tree T with n leaves, where each leaf is labeled by a distinct taxon of M, and each internal node of T is labeled by a vector in $Z(k)^m$ (which need not be in M), such that for every character c and every state i of c, the subgraph of T induced by the nodes labeled with state i for character c (which we denote by $T_c(i)$) must be a *connected subtree* of T. The requirement that subgraph $T_c(i)$ be a subtree is called the *convexity* requirement. Clearly, for any character c and states $i \neq j$, the subtrees $T_c(i)$ and $T_c(j)$ of perfect phylogeny T are node disjoint. An example is shown in Figure 1.

Another way to view convexity is to arbitrarily designate a node in T as the root; direct all the edges in T away from the root, and consider this directed tree as a giving a history of character mutations. The convexity requirement is then equivalent to saying that for any character/state pair (c, i), there is at most one edge in T where the state of character c mutates to i.

The **k-state Perfect Phylogeny Problem** is to determine, for input M, if there is a k-state perfect phylogeny for M, and if there is one, to construct one. The special case of $k = 2$ (the binary perfect phylogeny problem) has been extensively studied and is motivated by the "infinite sites" model from population genetics. The cases of $k > 2$ arise from non-binary integer data, and are further motivated by the related "infinite alleles" model from population genetics, rather than the infinite sites model. In the last decade, SNP data (which is binary) has been the dominant data type in population genomics, but increasingly *non-binary* integer population genomic data is becoming available and informative.

If neither k nor n nor m is fixed, so k can grow with n, then the k-state perfect phylogeny problem is NP-complete [3,25]. In contrast, if k is any *fixed* integer, independent of n and m, then the k-state Perfect Phylogeny Problem can be solved in time that is polynomial in n and m. This was first shown for $k = 2$ in [6] (and shown to be solvable in linear time in [13]), for $k = 3$ in [5], for $k = 3$ or 4 in [18], and for any fixed k in [1]. The later result was improved in [19], and a related method for the *near-perfect* phylogeny problem was developed in [9]. An excellent survey of most of these results appears in [8].

In this paper we consider the following three problems that extend the utility of the basic k-state Perfect Phylogeny model.

The Missing-Data (MD) Problem: If the input matrix M contains some cells with no values (and each cell with a value takes a value from $Z(k) = \{1..k\}$), can integers from $Z(k)$ be assigned to the cells with missing values so that the resulting matrix has a k-state perfect phylogeny?

Even for $k = 2$ this problem is NP-complete, although a directed version of it, when $k = 2$, can be solved in polynomial time [23]. The requirement that missing values be selected from $Z(k)$ is both biologically meaningful and computationally challenging. The MD problem has a very simple solution if integers up to n can be assigned [24].

The Character-Removal (CR) Problem: If there is no missing data in M, and M does not have a k-state perfect phylogeny, what is the *minimum* number of characters to remove so that the resulting matrix does have a k-state perfect phylogeny?

It is well known [24,7] that for binary data ($k = 2$), the CR problem reduces to the node-cover problem, but this is not true for $k > 2$, where the CR problem was previously unaddressed.

The Missing-Data Character-Removal (MDCR) Problem: If the input matrix M contains some cells with no values, and the answer to the MD problem is 'no', how should the missing values be set in order to minimize the solution to the resulting CR problem?

Problem MD is motivated by the reality of missing entries in biological datasets. For molecular genomic applications, missing data tends to be in the 1% to 5% range, and for phylogenetic applications, missing data in the 30% range is not uncommon. The CR and MDCR problems are motivated by the common practice in phylogenetics of removing characters when the existing data does not fit the perfect phylogeny model. This is most often done when the data is binary, but the problem and practice also arise for non-binary data [8,20]. The expectation in phylogenetics is that while there may be some characters that must be removed to make a perfect phylogeny, and hence some data are lost, if a perfect phylogeny can be constructed using a large percentage of the original characters then the resulting tree will give valid evolutionary information about the taxa. There are similar scenarios that motivate removal of sites in molecular data that arises in population genomics. Of course, in order to get the most informative

tree, we want to remove as few characters or sites as possible, motivating the Character-Removal problem.

In [14] we showed how to effectively solve, using *Integer Linear Programming*, the MD and the MDCR problems (and several other related problems) on *binary* data (i.e., $k = 2$) of size and complexity that is characteristic of many current applications in biology.

In this paper, we examine the MD, CR and MDCR problems for $k > 2$. We have developed Integer Linear Programming (ILP) solutions to these problems for the special cases of 3, 4 and 5 permitted states per character, but will only detail here the case of 3 states. We report on extensive empirical testing of these solutions. Then we develop a general theory to solve the MD problem for an *arbitrary* number of permitted states, using chordal graph theory and results on minimal triangulation of non-chordal graphs. This establishes new necessary and sufficient conditions for the existence of a perfect phylogeny with (or without) missing data. We implement the general theory using integer linear programming, although other optimization methods are possible. We extensively explore the empirical behavior of the general solution, showing that the methods are very practical for data of size and complexity that is characteristic of many current applications in phylogenetics. Some of the empirical results for the MD problem with an arbitrary number of permitted states are very surprising, suggesting the existence of additional combinatorial structure in multi-state perfect phylogenies.

When there is no missing data, these methods also establish alternative algorithms for the Perfect Phylogeny Problem. All of the software (other than the ILP solver) needed to replicate the results stated in this paper is available through the author's homepage: http://wwwcsif.cs.ucdavis.edu/~gusfield/.

2 The MD, CR and MDCR Problems for Three States

We have developed specialized methods for the cases of $k = 3, 4, 5$. For lack of space, we will only discuss the case of k = 3. For the MD problem, the general method for any k is more efficient on large problem instances than the specialized methods for $k = 3, 4, 5$.

First we state a needed definition and fact. In a binary matrix, two columns are called "incompatible" if they contain all of the binary pairs 0,0; 0,1; 1,0; and 1,1. Otherwise they are called "compatible". In our solution [14] to the MD and MDCR problems for *binary* data, we define a binary variable $Y(i, j)$ for every cell (i, j) that has a missing value in M, and a binary variable $I(p, q)$ for every pair of characters p, q. In [14] we developed compact linear inequalities that set $I(p, q)$ to 1 if and only if the Y variables are set in a way that makes characters p and q incompatible. These inequalities are also needed in our ILP formulations for the MD and MDCR problems with $k > 2$; in this paper we assume their existence, but refer the reader to [14] for details.

We next restate the main result from the paper by A. Dress and M. Steel [5] that establishes that the 3-state perfect phylogeny problem can be solved in

polynomial time. For this exposition, create another matrix $\overline{M}$ derived from M, with three characters $C_c(1), C_c(2), C_c(3)$, for each character c in M. All the taxa that have state i for c in M are given state 1 for character $C_c(i)$ in $\overline{M}$, and the other taxa are given state 0 for $C_c(i)$. So, the original input matrix M is recoded as a binary matrix $\overline{M}$ with three expanded characters for each character in M. Each expanded character defines a split of the taxa. The main structural result in [5], interpreted in terms of $\overline{M}$ is:

Theorem 1. *[5] Given matrix M with $k = 3$, there is a perfect phylogeny for M, if and only if there is a set of characters S of $\overline{M}$ which are pairwise compatible, and where for each character c in M, S contains at least two of the characters $C_c(1), C_c(2), C_c(3)$.*

In [5], a polynomial-time algorithm is given to find an appropriate set S, if one exits, but we will not need that here. In [15] we note that Theorem 1 can be used to reduce the three-state perfect phylogeny problem to the 2-SAT problem.

To explain our ILP solution to the MD problem for $k = 3$, we first describe how to solve the 3-state perfect phylogeny problem (without missing data) using integer linear programming. For each character c in M, let $S(c,1), S(c,2), S(c,3)$ be three binary variables associated with the characters $C_c(1), C_c(2), C_c(3)$ in $\overline{M}$, and create the inequality

(*) $S(c,1) + S(c,2) + S(c,3) \geq 2$.

Variable $S(c,z)$ (for z from {1,2,3}) is set to 1 to indicate that character $C_c(z)$ is selected in set S. Binary variable $I(C_c(z); C_{c'}(z'))$ is set to 1 if and only if characters $C_c(z)$ and $C_{c'}(z')$ are incompatible in $\overline{M}$. To implement the requirement that every pair of characters in S must be compatible, create the inequality: $S(c,z) + S(c',z') + I(C_c(z); C_{c'}(z')) \leq 2$. Then by Theorem 1, there is a 3-state perfect phylogeny for M if and only if there is a feasible solution to the ILP created by the above inequalities. Note that when there is no missing data, the value of $I(C_c(z); C_{c'}(z'))$ is determined from $\overline{M}$ and fixed, but when there is missing data, its value may be affected by the values given to missing data.

To solve the MD problem for 3-state data, we extend the above ILP for the 3-state perfect phylogeny problem as follows: If cell (i,j) of M is missing a value, we add the inequality $Y(i,j,1) + Y(i,j,2) + Y(i,j,3) = 1$, where $Y(i,j,1), Y(i,j,2), Y(i,j,3)$ are binary variables that select how the missing value in cell (i,j) should be set. We also add in the inequalities (introduced in Section 1) that set each binary variable $I(C_c(z); C_{c'}(z'))$ to 1 if and only if the Y variables are set in a way that makes the resulting columns $C_c(z)$ and $C_{c'}(z')$ in $\overline{M}$ incompatible.

We modify the above formulation for the MD problem to create an ILP that solves the MDCR problem. We remove the (*) inequalities, and create a binary variable $R(c)$ for each character c in M. Then for each c in M, we add the inequalities $S(c,1) + S(c,2) + S(c,3) - 2R(c) \geq 0$, which ensures that $R(c)$ will be set to 1 only if at least two of the variables $S(c,1), S(c,2), S(c,3)$ are set to 1. Finally, use the objective function of *maximizing* $\sum_c R(c)$. Problem CR is solved in the special case that there are no missing values in M.

3 The MD Problem for Arbitrary k

In this section we develop the conceptually deepest method, leading to the most surprising empirical results of this work. We consider a formulation to the MD problem that is correct for any allowed number of states, k. Note that although there is no restriction on what k can be in the formulation, k is known and fixed for any particular problem instance, and imputed values for missing entries must be from $Z(k)$. We show that an approach which at first might seem impractical, but in fact works extremely well in practice on data of size and complexity that are consistent with many (but not all) current applications in phylogenetics.

Our approach to the MD problem for arbitrary k is to use a classic, but not widely exploited, result about the Perfect Phylogeny Problem, together with newer combinatorial results on chordal graphs, clique-trees and minimal triangulations of non-chordal graphs. We assume throughout that the rows of M are distinct. We first introduce the classic theorem due to Buneman [4,24].

Given an n by m input matrix M define the PI (partition intersection) graph $G(M)$ for M as follows: $G(M)$ has one node for every character-state pair (c, i) that occurs in M; there is an edge between the nodes for character-state pairs (c, i) and (c', i') if and only if there is a taxon in M with state i for character c and with state i' for character c'. Thus $G(M)$ is formed by the superposition of n cliques (one for each taxon), each of size m, and $G(M)$ is an m-partite graph with m classes, one class for each character. Each class is sometimes refereed to as a *color*. A pair of nodes (c, i), (c', i') where $c \neq c'$, defines a *legal potential edge* (or 'legal edge' for short) if that edge is not already in $G(M)$. A pair of nodes (c, i), (c, i'), for some *single* character c, defines an *illegal potential edge* (or 'illegal edge' for short), and is not in $G(M)$. A pair of nodes in the same class of $G(M)$ are called a *mono-chromatic pair*.

A cycle C in a graph G is called *chordless* if G does not contain any edge e between two nodes in C unless e is an edge of C. A graph is *chordal* if and only if it contains no chordless cycles of size four or more. A *triangulation* or *chordal fill-in* $H(G)$ of a graph G is a chordal super-graph of G, on the same nodes as G. The edges in $H(G) - G$ are the *added edges*. Given M, a *legal triangulation* of $G(M)$ is a triangulation where all of the added edges are legal. Note that $G(M)$ might itself be chordal, but if not chordal, there might not be any legal triangulation of $G(M)$.

Buneman's Theorem [4,24]. M has a perfect phylogeny if and only if $G(M)$ has a legal triangulation $H(G(M))$.

We will also need the following definition and facts.

Definition. Let G be a graph and T be a tree containing one node for each maximal clique (a clique where no additional nodes can be added) in G. It is useful to label each node v in T with the nodes of G that form the maximal clique in G associated with node v in T. Then T is defined to be a *clique-tree* for G if and only if for each node w in G, the nodes in T which contain the label w form a *connected* subtree of T.

Theorem [11,4,12,20]. A graph G is chordal if and only if there is a clique-tree T for G.

When $G(M)$ is the PI graph for an input M, and a legal triangulation $H(G(M))$ exists for $G(M)$, each node in $G(M)$ represents a character-state pair, and each taxon in M induces a maximal clique in $G(M)$ and in $H(G(M))$. It is then easy to see that any clique tree for $H(G(M))$ is a perfect phylogeny for M. It is known [20] that a clique tree for a chordal graph can be found in time linear in the size of the chordal graph. Hence, when $G(M)$ has a legal triangulation, a perfect phylogeny for M can be found in linear time from $H(G(M))$.

3.1 The PI-Graph Approach to the MD Problem

If M contains missing data, let $G(M)$ now be created by only using cells in M where the value is *not* missing. If a taxon has q out of m cells with known values, then that taxon induces a clique in $G(M)$ of size q rather than a clique of size m; $G(M)$ is again an m-partite graph formed by the superposition of n cliques. The key point, which is easy to establish, is that Buneman's Theorem continues to hold for $G(M)$ created in this way, i.e., for input M with missing values, even if $G(M)$ now has fewer nodes. Indeed, even when M has no missing values, unless $G(M)$ is chordal without adding any edges, a legal triangulation can be thought of as adding new data (in fact, new taxa) to M which will be represented at internal nodes of the perfect phylogeny. When M has missing data, then a legal triangulation essentially adds new data to the taxa in M that are missing entries, along with adding new taxa to M. In more detail, let M be input where a taxon q has some missing entries, and suppose there is a legal triangulation $H(G(M))$ for $G(M)$; let T be a clique-tree for $H(G(M))$. Every node in T corresponds to a maximal clique in $H(G(M))$, and the original clique in $G(M)$ induced by taxon q must be completely contained in one or more maximal cliques in $H(G(M))$. Also each taxon in M that has no missing data induces a maximal clique in $H(G(M))$. Therefore there is one node in T for each taxon with no missing data in M, and at least one node v in T whose label contains all the nodes in the clique in $G(M)$ associated with taxon q. We then assign taxon q to node v of T, and if v is not a leaf, we add a new edge connecting v to a new leaf labeled q. The resulting tree T is now a perfect phylogeny for M.

We next use T to impute values for any missing data in M. Consider again taxon q as above. Recall that node v in T is labeled with nodes of $G(M)$, and hence is labeled with character/state pairs from M with at most one state for any character. Assign all the character/states that label v to taxon q. For any character c which does not have an associated label in v, do a breath-first search from v until reaching some node v' in T which is labeled by character/state pair (c, i), for some state i. Then label each node on the path from v to v' in T with (c, i), assigning state i for character c to taxon q. Successively repeat this until all taxa have values for all characters. The result is a filled matrix M which has a perfect phylogeny T.

Finding a Legal Triangulation

The PI-graph approach to the k-state Perfect Phylogeny problem is conceptually perfect to handle missing data. However, it is not clear how to efficiently find a legal triangulation in practice, or determine that there is none, so and it may seem that this would not be a productive approach. We next show, by exploiting more contemporary results about triangulation in general graphs, that this approach can be effectively implemented in practice on data of realistic size for many current application in biology. We first state some definitions and facts.

Definition. A triangulation of a non-chordal graph G is called *minimal* if no subset of the added edges defines a triangulation of G.

There is a large and continuing literature on the structure of minimal triangulations and on algorithms for finding them [16]. Clearly, for an input graph M, if there is a legal triangulation of $G(M)$ (and hence a perfect phylogeny for M), then there is a legal triangulation which is minimal, and we can apply appropriate results from the literature on minimal triangulations.

Definition. An *a,b separator* in a graph G is a set of nodes whose removal separates node a from node b. A *minimal a,b separator* is an *a,b* separator such that no subset of it is an *a,b* separator. A separator is called a *minimal separator* if it is a minimal *a,b* separator for some pair of nodes *a,b*. A minimal separator K is said to *cross* a minimal separator K' if K is a separator for some pair of nodes in K'.

It is easy to establish [22] that crossing is a symmetric relation for minimal separators; testing whether two minimal separators cross can be done in time proportional to the number of edges in G. Clearly, a minimal separator which is a clique cannot cross any other minimal separator, because crossing is symmetric and no separator can separate two nodes that are connected by an edge.

Definition. Two minimal separators that do not cross each other are said to be *parallel*.

Definition. A minimal separator K is *completed* by adding in all the missing edges to make K a clique.

It is also easy to establish [22] that when a minimal separator K is completed, every minimal separator that K crossed (before K was completed) now ceases to be a minimal separator.

Definition. A set Q of pairwise parallel minimal separators in G is said to be *maximal* if every other minimal separator in G crosses some separator in Q.

The capstone theorem in the study of minimal triangulations, due to A. Parra and P. Scheffler, is

Minimal Triangulation Theorem [21,22]: Every minimal triangulation of a non-chordal graph G can be created by finding a maximal set Q of pairwise parallel minimal separators in G and completing each separator in Q. Conversely, the completion of each separator in any maximal set of pairwise parallel minimal separators in G creates a minimal triangulation of G.

Now we can relate these general definitions and results about minimal triangulations to the multi-state Perfect Phylogeny Problem. Given input M, a minimal separator K in the partition intersection graph $G(M)$ is defined to be *legal* if, for each character c in M, K contains at most one node (c, i) associated with character c. That is, only legal edges will be added if K is completed. A minimal separator that is not legal is called *illegal*. With the above, we can establish the following new characterization of the existence of a multi-state Perfect Phylogeny, even when there is missing data.

Theorem 2 (MSP). *There is a perfect phylogeny for input M (even if M has missing data) if and only if there is a set Q of pairwise parallel legal minimal separators in $G(M)$ such that every illegal minimal separator in $G(M)$ is crossed by at least one separator in Q.*

Proof. If there is a perfect phylogeny for M then there is a legal triangulation of $G(M)$ (by Buneman's theorem and its extension to the case of missing data), and so there is a minimal triangulation $H(G(M))$ that is legal. By the Minimal Triangulation theorem, there is a maximal set Q of pairwise parallel minimal separators of $G(M)$ whose completion results in the graph $H(G(M))$. Set Q cannot contain an illegal minimal separator K, for if it did, the completion of K would add an illegal edge, contradicting the fact that the triangulation is legal. Therefore Q only contains legal minimal separators, and the fact that it is maximal (with respect to all separators, legal or not) means that every illegal minimal separator must be crossed by at least one separator in Q.

Conversely, suppose there is a set Q of pairwise parallel legal minimal separators in $G(M)$ such that every illegal minimal separator in $G(M)$ is crossed by at least one separator in Q. If Q is not maximal (because there are additional *legal* minimal separators that are not crossed by any member of Q), Q can be extended to become maximal by greedily adding in legal minimal separators until every remaining legal minimal separator crosses some separator in the extended Q. Then by the Minimal Triangulation theorem, the completion of each separator in Q results in a chordal graph. Since only legal edges were used in this triangulation, Buneman's theorem establishes that M has a perfect phylogeny. □

Theorem MSP leads to three corollaries which, when they apply, each solve the MD problem without a complex search for Q and without the need to solve an optimization problem.

Corollary 1 (MSP1). *If $G(M)$ has an illegal minimal separator (or mono-chromatic pair of nodes) that is not crossed by any legal minimal separator, then M has no perfect phylogeny.*

Corollary 2 (MSP2). *If $G(M)$ has no illegal minimal separators, then M has a perfect phylogeny.*

Corollary 3 (MSP3). *If no pair of legal minimal separators in $G(M)$ cross, and every illegal minimal separator (or mono-chromatic pair of nodes) is crossed by at least one legal minimal separator, then M has a perfect phylogeny.*

Each of these corollaries applies in some problem instances in our empirical tests. Strikingly, in our tests, Corollary MSP1 applies in *almost every* problem instance that has no perfect phylogeny, and Corollaries MSP2 and MSP3 apply frequently.

Theorem MSP can be extended in a way that leads to a simpler characterization of the existence of a perfect phylogeny, with practical consequences in some cases.

Theorem 3 (MSPN). *There is a perfect phylogeny for input M (even if M has missing data) if and only if there is a set Q of pairwise parallel legal minimal separators in $G(M)$ such that every mono-chromatic pair of nodes in $G(M)$ is separated by some separator in Q.*

Proof. We prove the 'if' direction first. Let i, j be a pair of states of a character (color) c in M, and let u_i, u_j be the mono-chromatic pair of nodes in $G(M)$ representing states i, j of c. Every (u_i, u_j) separator crosses every illegal minimal separator that contains (u_i, u_j). By definition, an illegal minimal separator must contain a monochromatic pair of nodes. So the existence of a set of pairwise-parallel legal minimal separators that separate every monochromatic pair of nodes in $G(M)$ demonstrates that there is a maximal set of pairwise-parallel minimal separators all of which are legal, and hence, by Theorem MSP there is a perfect phylogeny for M.

The 'only if' direction is more subtle. If M has a perfect phylogeny, then by the Buneman theorem there is a legal triangulation of the partition-intersection graph $G(M)$. As argued earlier, any clique-tree T derived from $G(M)$ is a perfect phylogeny for M. For clarity, we will use the term "nodes" for $G(M)$ and "vertices" for T. By convexity, for any pair of states i, j of a character c in M there must be an edge $e(i, j)$ in T that separates all vertices in T labeled with state i for c from all vertices labeled with state j for c. Let u_i, u_j be the nodes of $G(M)$ associated with states i, j of c. To prove this direction of the theorem, we first want to identify a legal, minimal separator in $G(M)$ that separates u_i and u_j. To do this, we the fact that T is a clique-tree, and two facts about clique-trees.

Each vertex v in T is labeled with a a subset of nodes of $G(M)$ that form a maximal clique in $G(M)$. Let S and S' be the two maximal cliques in $G(M)$ associated with two end-vertices of an edge $e = (v, v')$ in T. The removal of e from T creates two connected subtrees T_v and $T_{v'}$; let N_v and $N_{v'}$ be the two sets of nodes in G_M that label the vertices in these subtrees. We now state two central facts about chordal graphs (in general) and their clique-trees, specialized to $G(M)$ and T: a) the nodes in $S \cap S'$ form a minimal separator, K, in $G(M)$, which is an a, b minimal separator for every pair of nodes $a \in (T_v - K)$ and $b \in (T_{v'} - K)$; b) the set of all such minimal separators, obtained from the edges in T, are pairwise parallel. See [20,22] for the general version of the first fact, and [22] for the general version of the second fact. As a side note, these facts can be used to speed up the k-state perfect phylogeny algorithms from [1,19] in practice, but we will not detail that in this paper.

Now we apply these facts to edge $e(i, j)$. By convexity, node u_i cannot be part of labels of vertices in both subtrees of $T - e(i, j)$, and the same holds for u_j.

Therefore, neither u_i nor u_j can be in the minimal separator $K(i,j)$ of $G(M)$ associated with $e(i,j)$, and hence $K(i,j)$ separates u_i and u_j. Further, $K(i,j)$ is legal because the nodes in $K(i,j)$ are the intersection of two maximal cliques in $G(M)$, and no clique in $G(M)$ has two nodes representing states of the same character (for then $G(M)$ would have an illegal edge). Hence, we have proved that if M has a perfect phylogeny, it has a perfect phylogeny, T, where the set of edges of T define a set of legal, pairwise-parallel minimal separators of $G(M)$ which separate every mono-chromatic pair of nodes. □

We can also establish a different characterization as follows. Suppose K is a minimal separator in $G(M)$, and x is a sequence in M. Recall that there is a clique in $G(M)$ associated with x. Note that if one of the nodes in that clique is in the connected component C of $G(M)$ - K, then all nodes of that clique must be in $C \cup K$. This follows because there can be no edge between nodes in different connected components of $G(M) - K$. Therefore, we say that K separates two taxa x and x' in M if and only if K separates a node in the clique for x from a node in the clique for x'.

Theorem 4 (MSPNN)). *There is a perfect phylogeny for input M (even if M has missing data) if and only if there is a set Q of pairwise parallel legal minimal separators in $G(M)$ such that every pair of taxa in M is separated by some separator in Q.*

3.2 An ILP Formulation of Problem MD for Arbitrary *k*

From theorems MSP, MSPN and MSPNN, it is now conceptually simple to formulate an integer linear program for the Perfect Phylogeny Problem and for problem MD. This ILP tries to select an appropriate subset Q of legal minimal separators in $G(M)$. Given input M, the approach based on Theorem MSP creates the ILP as follows: find all the legal and illegal minimal separators in $G(M)$; for each legal minimal separator K, determine which minimal separators cross K; let $\mathcal{K}$ be the set of legal minimal separators which each cross some illegal minimal separator; create a binary variable for each minimal separator in $\mathcal{K}$; a variable will be set to 1 to indicate that the associated legal minimal separator will be selected for Q; create inequalities that forbid the selection into Q of two legal minimal separators in $\mathcal{K}$ that cross each other; and create inequalities that require that each illegal minimal separator be crossed by at least one legal minimal separator in $\mathcal{K}$. Then, the MD problem has a solution (values for the missing data can be found so that the resulting data has a perfect phylogeny) if and only if these inequalities have a feasible solution. Further, if there is a feasible solution (identifying a set Q), and Q is extended to become maximal (as in the proof of Theorem MSP), then completing the minimal separators in Q creates a chordal graph from which a clique-tree and a perfect phylogeny T for M can then be constructed; imputed values for any missing data in M can be obtained from T.

Similarly, from Theorems MSPN and MSPNN, we can formulate an ILP solution for the MD problem *without* knowing the illegal minimal separators of $G(M)$; it is sufficient to know only the legal minimal separators of $G(M)$.

Although we have formulated and tested the problem of finding Q as an ILP, alternative (perhaps more direct) methods are possible and may be more practical, especially when the number of minimal separators found in $G(M)$ is small.

3.3 Finding the Minimal Separators

All the minimal separators in $G(M)$ can be found using the algorithm in [2], which has a worst-case running time proportional to the number of edges in $G(M)$ times the total number of nodes in all the minimal separators. In worst case, the number of nodes in $G(M)$ is km and the number of edges is bounded by $\min(n, k^2) \times \binom{m}{2}$. Note that $k \leq n$. However, we only need to find the minimal separators if the number of edges is less than $km(m-1)$. This follows from the fact that a pair of characters i, j cannot have a perfect phylogeny (or be in a perfect phylogeny) unless the subgraph of $G(M)$ induced by the set of state-nodes of characters i and j is acyclic [10,24], meaning that the induced subgraph can have at most $2k-1$ edges. This required bound continues to apply if edges are added to $G(M)$. When we build $G(M)$ from M, we check that the number of edges induced by each pair of characters is less than $km(m-1)$ (we call this the *graph-density test*) and then search for the minimal separators only on graphs that pass the test. Moreover, there is a faster alternative approach to finding the legal minimal separators, detailed next, in the case that there are no missing entries.

Theorem 5. *No minimal separator in Q can have m nodes, and so we only need to find the legal minimal separators in $G(M)$ of size $m-1$ or less. When M has no missing data, the number of legal minimal separators of size $m-1$ or less in $G(M)$ is bounded by the number of proper clusters in M (the main object in the algorithms of [1,19]) and all these legal minimal separators can be found in $O(2^k nm^2)$ time.*

Following Theorems MSPN and MSPNN, Theorem 5 can be used to more efficiently construct an ILP for the k-state perfect phylogeny problem, and this also puts a polynomial bound, for any fixed k, on the size of the ILP needed to solve the problem. That ILP will have $O(2^k m)$ variables and $O(2^{2k}m^2 + \min(n^2, mk^2))$ inequalities. Note that Theorems MSPN and MSPNN hold for problem MD, but Theorem 5 has only been established for the case of *no* missing data. Whether this can be generalized to the case of missing data is an open question. Another open question is to extend the PI-graph approach to the CR and MDCR problems.

4 Empirical Results

For biologically informative simulation results, we want data obtained from simulators that model appropriate biological processes. To generate biologically informative data that is guaranteed to have a perfect phylogeny, we use the

well-known coalescent-based program *ms* [17], with no recombination, to generate *binary* matrices that are guaranteed to have 2-state perfect phylogenies. The following theorem explains how we use those matrices to obtain data guaranteed to have a k-state perfect phylogeny, for any chosen k.

Theorem 6. *Suppose M is an n by m binary matrix that has a 2-state perfect phylogeny, and that $m = m_k \times (k-1)$. Let M' be an n by m' matrix obtained by partitioning each row of M into m_k groups of $k-1$ consecutive columns each, and converting to decimal the $(k-1)$-length binary number in each group. Then each column of M' has at most k distinct numbers, and if, in each column, we map those k numbers to $Z(k)$ (in any arbitrary way), the resulting matrix is guaranteed to have a k-state perfect phylogeny.*

To generate k-state data which might not have a k-state perfect phylogeny we set the *ms* recombination parameter r to a value greater than zero so that the binary data is not guaranteed to have a binary perfect phylogeny; we again group $k-1$ consecutive columns, creating binary numbers of length $k-1$. In each such group, we scan the rows from top to bottom to find the first k distinct binary numbers, and map these to the states 0 to $k-1$; any binary number outside that set of k binary numbers is randomly mapped to one of the k states. When r is small, this problem instance might have a k-state perfect phylogeny but is not guaranteed to. As r grows, the chance that this data has a perfect phylogeny falls. We used the ILP solver Cplex 11, running on a 2.5 GHz. iMac with 3 Gig. of memory.

4.1 Empirical Results for $k = 3$

The ILP formulations developed for problems MD, CR, and MDCR for the special case of $k = 3$ are conceptually simple, and so the main result of this section is the empirical observation that those ILP formulations generally solve very quickly for biologically meaningful ranges of data. We have done extensive testing with a wide range of parameters, with results that are consistent with the illustrative results shown in Table 1. The times reported are for solving the ILPs; these are the times of interest. The times to generate the ILPs ranged from under one second to a small number of seconds, but are not reported because the Perl programs that generate the ILPs are highly unoptimized.

4.2 Empirical Results for Solving Problem MD for Arbitrary k

While the approach to problem MD based on PI graphs is conceptually correct, one might expect (and we did initially expect) that it would not be practical. There were two potential bottlenecks, the time needed to find all the minimal separators in $G(M)$, and the time needed to solve the resulting ILPs. The main result of this section is the surprising empirical observation that this approach is very practical in data sizes that cover many current, large phylogenetic applications, although not genomic size applications. We detail illustrative empirical

Table 1. Average ILP execution times (in seconds unless noted otherwise) for problems MD, MDCR and CR (problem MDCR reduces to problem CR when the percent deletion is 0) with $k = 3$, on relatively large problem instances. Each average is taken over 100 instances (except for the last entry in rows b and c) where a 3-state perfect phylogeny exists. The time for the last entries in rows b and c are averaged over ten instances. The header shows the percentage of cells whose values were removed before solving the problems. Row a) is for problem MD on data of size 50 by 25, and row b) is for problem MD on data of size 100 by 50. Times for instances of problem MD where the data does not have a 3-state perfect phylogeny are similar, but smaller. Row c) is for problem MDCR on data of size 50 by 25. Surprisingly, the times are very similar to the times for problem MD, except for the case of 35% deletion where the variance in the times for the MDCR solution becomes large. In the ten executions of MDCR taking an average of 43 minutes, seven took under two seconds, one took two minutes, one took around two hours, and one took over five hours. Problem MDCR did not solve quickly enough on data of size 100 by 50 for in-depth exploration, except for zero deletions, where it solved in an average of 0.05 seconds. On data of size 30 by 100, and 1% deleted values, the MDCR problem solved in an average of 3.3 seconds; with 5% deleted values it solved in an average of 13 seconds.

	0%	1%	5%	10%	20%	35%
a	0.01	0.06	0.18	0.32	0.6	32
b	0.024	0.6	1.8	3.04	19.6	$1hr.20mins.$
c	0.018	0.06	0.2	0.38	0.81	43 mins.

results in Table 2. The programs implementing the PI-graph approach are written in Perl, except for the program to find all of the minimal separators and their crossing relations, which is written in C for greater efficiency.

We separate the tests with *ms* recombination parameter, r, set to zero (where a perfect phylogeny is guaranteed), from tests with $r > 0$ (where a perfect phylogeny is not guaranteed, although one might exist). The choice of r is critical to test how the PI-graph approach performs on data that does not have a perfect phylogeny, and to see how often such data acquires a perfect phylogeny as the amount of missing data increases. If r is set too high, most instances fail the graph-density test, establishing trivially that there is no perfect phylogeny without needing to try to find a legal triangulation of $G(M)$. If r is set too low, then then there will be a perfect phylogeny. Through experimentation, we have found values of r that provide a good balance; generally r must decrease as the size of the problem increases.

The most striking empirical results, and the key observed reasons for the efficiency of the PI-graph approach are: 1) In data where there is a perfect phylogeny we observe vastly fewer minimal separators than are possible in worst-case; the minimal separators are found surprisingly quickly in practice; most of the observed legal minimal separators do not cross any illegal minimal separator, and so do not enter into the ILPs; many of the problem instances are solved by the use of Corollaries MSP2 and MSP3; in cases where an ILP must be solved, the ILPs are tiny and most solve in the Cplex pre-solver; all the ILPs, whether solved

Table 2. Illustrative empirical results for the PI-graph approach. The results in each row are based on 100 instances (except for the case of n = 100, where 25 instances were used). When $r = 0$, each instance is guaranteed to have a perfect phylogeny. When $r > 0$, an instance might have a perfect phylogeny, but we are interested in the instances which do not (the number of instances in any row that have no perfect phylogeny equals $d + c1 + inf$). Details for the column labels: n, m = dimension of M; k = number of allowed states; $r = ms$ recombination parameter; % miss = expected percentage of missing entries; # v, e = average number of nodes and edges in $G(M)$; $L, Iseps$ = average number of legal and illegal minimal separators in $G(M)$; sep time = average time to find all the minimal separators and the needed crossing relations and build the ILP if needed; % $d, c1, c2, c3$ = percentage of problem instances failing the graph-density test, or solved using Corollaries MSP1, MSP2 and MSP3 respectively; % ilps = percentage of instances that must be solved as ILPs; # var, con = average number of variables and constraints in the generated ILPs; inf = number of the ILPs that are infeasible, establishing that the problem instance has no perfect phylogeny; % pre = percentage of the generated ILPs that were solved in the Cplex pre-solver. Note that the number of ILP variables is the number of legal minimal separators that cross at least one illegal minimal separator. Comparing that number to the number of legal minimal separators shows that most legal minimal separators do not cross any illegal minimal separator. All ILPs solved in 0.00 Cplex-reported seconds.

n, m	k	r	% miss	# v, e	# $L, Iseps$	sep time	% $d, c1, c2, c3$	% ilps	# var con	# inf	% pre
20, 20	4	0	0	70.9, 990	16, 0.47	0.07 s	0, 0, 72, 27	1	3, 2	0	100
20, 20	4	0	5	69.8, 952	16.4, 0.75	0.071 s	0, 0, 64, 33	3	3.6, 2.6	0	100
20, 20	4	0	20	66.8, 840	17.7, 1.6	0.066 s	0, 0, 41, 40	19	5.3, 4.3	0	100
20, 20	4	0	35	62.3, 696	20.6, 5.4	0.066 s	0, 0, 10, 15	75	8.4, 10.3	0	88
40, 40	10	0	0	292, 8554	49, 6	0.850 s	0, 0, 23, 33	44	9, 5	0	91
40, 40	10	0	5	287, 8221	51, 8	0.855 s	0, 0, 13, 30	57	9, 15	0	95
40, 40	10	0	20	272, 7141	57, 27	1.03 s	0, 0, 0, 11	89	19, 36	0	91
40, 40	10	0	35	255, 5874	69, 11	2.3 s	0, 0, 0, 0	100	36, 136	0	40
40, 40	20	0	0	447, 12562	56, 23	1.44 s	0, 0, 3, 21	76	13, 41	0	72
40, 40	20	0	5	439, 11954	60, 31	1.58 s	0, 0, 2, 19	79	15, 51	0	71
40, 40	20	0	20	410, 9983	69, 90	2.58 s	0, 0, 0, 1	99	28, 111	0	63
60, 60	15	0	0	601, 27072	83, 31	6.1 s	0, 0, 1, 24	75	13, 43	0	89
60, 60	15	0	10	581, 24904	89, 56	7.4 s	0, 0, 0, 9	91	22, 67	0	89
60, 60	15	0	20	559, 22553	94, 107	9.9 s	0, 0, 0, 1	99	37, 118	0	85
80, 80	10	0	0	613, 38290	96, 19	12 s	0, 0, 4, 44	52	11, 31	0	96
80, 80	10	0	5	606, 37093	99, 24	12.6 s	0, 0, 2, 39	59	13, 37	0	97
80, 80	10	0	10	597, 35841	102, 37	14 s	0, 0, 0, 21	79	18, 48	0	95
80, 80	10	0	20	584, 33863	109, 70	18 s	0, 0, 0, 7	93	32, 83	0	87
80, 80	20	0	0	1026, 62540	118, 85	32 s	0, 0, 0, 12	88	21, 106	0	83
80, 80	20	0	1	1023, 62081	118, 91	33 s	0, 0, 0, 12	88	22, 113	0	84
100, 100	10	0	0	791, 62733	122, 33	33 s	0, 0, 0, 36	64	13, 51	0	94
100, 100	10	0	5	783, 60924	123, 42	34 s	0, 0, 0, 36	64	17, 60	0	94
20, 20	10	1.5	0	124, 1669	21, 5	0.099 s	5, 26, 43, 11	12	6,8	0	100
20, 20	10	1.5	10	119, 1493	23, 6	0.096 s	3, 27, 27, 23	20	6,8	1	95
20, 20	10	1.5	35	103, 1015	29, 16	0.095 s	1, 12, 3, 3	81	12, 21	1	75
40, 40	10	1.25	0	289, 8738	48, 18	1.03 s	37, 28, 4, 13	18	8, 13	1	83
40, 40	10	1.25	20	267, 7217	55, 97	2.4 s	15, 46, 1, 4	34	20,56	1	85
40, 40	10	1.25	35	250, 5899	67, 334	7.3 s	3, 55, 0, 0	42	36,145	1	41
80, 80	10	0.75	0	618, 39621	93, 31	13 s	52, 26, 3, 4	15	10, 26	0	93
80, 80	10	0.75	5	610, 38359	97, 41	15 s	50, 28, 1, 3	18	13, 31	0	94
80, 80	10	0.75	20	584, 34353	105, 117	29 s	35, 42, 1, 2	20	36,86	0	90
80, 80	20	0.5	0	1023, 62911	112, 117	49 s	2, 48, 0, 5	45	20, 104	0	84
80, 80	20	0.5	5	1006, 60449	114, 159	55 s	2, 49, 0, 3	46	27, 128	0	83
80, 80	20	0.5	20	954, 52563	107, 321	82 s	0, 54, 0, 0	46	60, 264	0	65

in the pre-solver or not, are solved by Cplex in 0.00 Cplex-reported seconds. 2) In instances which do not have a perfect phylogeny (and do not immediately fail the graph-density test), the solution to the MD problem was almost always detected by the use of Corollary MSP1 without even constructing an ILP. The few ILPs that were constructed for instances without a perfect phylogeny also solved in 0.00 Cplex-reported seconds. Theoretical explanations of these observations will likely be probabilistic and be related to the way the datasets were generated.

Acknowledgments

I thank Rob Gysel for programming the algorithm in [2] which finds all the minimal separators in a graph and the crossing pairs involving legal minimal separators. I thank the reviewers for careful reading and helpful comments. This research was partially supported by NSF grants SEI-BIO 0513910, CCF-0515378, and IIS-0803564.

References

1. Agarwala, R., Fernandez-Baca, D.: A polynomial-time algorithm for the perfect phylogeny problem when the number of character states is fixed. SIAM J. on Computing 23, 1216–1224 (1994)
2. Berry, A., Bordat, J.-P., Cogis, O.: Generating All the Minimal Separators of a Graph. In: Widmayer, P., Neyer, G., Eidenbenz, S. (eds.) WG 1999. LNCS, vol. 1665, pp. 167–172. Springer, Heidelberg (1999)
3. Bodlaender, H., Fellows, M., Warnow, T.: Two strikes against perfect phylogeny. In: Proc. of the 19'th Inter. colloquium on Automata, Languages and Programming, pp. 273–283 (1992)
4. Buneman, P.: A characterization of rigid circuit graphs. Discrete Math. 9, 205–212 (1974)
5. Dress, A., Steel, M.: Convex tree realizations of partitions. Applied Math. Letters 5, 3–6 (1993)
6. Estabrook, G., Johnson, C., McMorris, F.: An idealized concept of the true cladistic character. Math. Bioscience 23, 263–272 (1975)
7. Felsenstein, J.: Inferring Phylogenies. Sinauer, Sunderland (2004)
8. Fernandez-Baca, D.: The perfect phylogeny problem. In: Du, D.Z., Cheng, X. (eds.) Steiner Trees in Industries. Kluwer Academic Publishers, Dordrecht (2000)
9. Fernandez-Baca, D., Lagergren, J.: A polynomial-time algorithm for near-perfect phylogeny. In: Meyer auf der Heide, F., Monien, B. (eds.) ICALP 1996. LNCS, vol. 1099, pp. 670–680. Springer, Heidelberg (1996)
10. Fitch, W.: Towards finding the tree of maximum parsimony. In: Estabrook, G.F. (ed.) Proceedings of the eighth international conference on numerical taxonomy, pp. 189–230. W.H. Freeman, New York (1975)
11. Gavril, F.: The intersection graphs of subtrees in trees are exactly the chordal graphs. J. Combinatorial Theory, B 16, 47–56 (1974)
12. Golumbic, M.C.: Algorithmic Graph Theory and Perfect Graphs. Academic Press, New York (1980)
13. Gusfield, D.: Efficient algorithms for inferring evolutionary history. Networks 21, 19–28 (1991)

14. Gusfield, D., Frid, Y., Brown, D.: Integer programming formulations and computations solving phylogenetic and population genetic problems with missing or genotypic data. In: Lin, G. (ed.) COCOON 2007. LNCS, vol. 4598, pp. 51–64. Springer, Heidelberg (2007)
15. Gusfield, D., Wu, Y.: The three-state perfect phylogeny problem reduces to 2-SAT (to appear)
16. Heggernes, P.: Minimal triangulations of graphs: A survey. Discrete Mathematics 306, 297–317 (2006)
17. Hudson, R.: Generating samples under the Wright-Fisher neutral model of genetic variation. Bioinformatics 18(2), 337–338 (2002)
18. Kannan, S., Warnow, T.: Inferring evolutionary history from DNA sequences. SIAM J. on Computing 23, 713–737 (1994)
19. Kannan, S., Warnow, T.: A fast algorithm for the computation and enumeration of perfect phylogenies when the number of character states is fixed. SIAM J. on Computing 26, 1749–1763 (1997)
20. McKee, T.A., McMorris, F.R.: Topics in Intersection Graph Theory. Siam Monographs on Discrete Mathematics (1999)
21. Parra, A., Scheffler, P.: How to use the minimal separators of a graph for its chordal triangulation. In: Fülöp, Z., Gecseg, F. (eds.) ICALP 1995. LNCS, vol. 944, pp. 123–134. Springer, Heidelberg (1995)
22. Parra, A., Scheffler, P.: Characterizations and algorithmic applications of chordal graph embeddings. Discrete Applied Mathematics 79, 171–188 (1997)
23. Pe'er, I., Pupko, T., Shamir, R., Sharan, R.: Incomplete directed perfect phylogeny. SIAM J. on Computing 33, 590–607 (2004)
24. Semple, C., Steel, M.: Phylogenetics. Oxford University Press, Oxford (2003)
25. Steel, M.: The complexity of reconstructing trees from qualitative characters and subtrees. J. of Classification 9, 91–116 (1992)