Incomplete Directed Perfect Phylogeny

I. Pe'er[1], R. Shamir[1], and R. Sharan[1]

Department of Computer Science,
Tel-Aviv University, Tel-Aviv, Israel
{izik,shamir,roded}@math.tau.ac.il

Abstract. Perfect phylogeny is one of the fundamental models for studying evolution. We investigate the following generalization of the problem: The input is a species-characters matrix. The characters are binary and directed, i.e., a species can only gain characters. The difference from standard perfect phylogeny is that for some species the state of some characters is unknown. The question is whether one can complete the missing states in a way admitting a perfect phylogeny. The problem arises in classical phylogenetic studies, when some states are missing or undetermined. Quite recently, studies that infer phylogenies using inserted repeat elements in DNA gave rise to the same problem. The best known algorithm for the problem requires $O(n^2m)$ time for m characters and n species. We provide a near optimal $\tilde{O}(nm)$-time algorithm for the problem.

1 Introduction

When studying evolution, the divergence patterns leading from a single ancestor species to its contemporary descendants are usually modeled by a tree structure. Extant species correspond to the tree leaves, while their common progenitor corresponds to the root of this *phylogenetic tree*. Internal nodes correspond to hypothetical ancient species, which putatively split up and evolved into distinct species. Tree branches model changes through time of the hypothetical ancestor species. The common case is that one has information regarding the leaves, from which the phylogenetic tree is to be inferred. This task, called *phylogenetic reconstruction* (cf. [7]), was one of the first algorithmic challenges posed by biology, and the computational community has been dealing with problems of this flavor for over three decades (see, e.g., [12]).

In the character-based approach to tree reconstruction, contemporary species are described by their attributes or *characters*. Each character takes on one of several possible *states*. The input is represented by a matrix $\mathcal{A}$ where a_{ij} is the state of character j in species i, and the i-th row is the *character vector* of species i. The output sought is a hypothesis regarding evolution, i.e., a phylogenetic tree along with the suggested character-vectors of the internal nodes. This output must satisfy properties specified by the problem variant.

One important variant of the phylogenetic reconstruction problem is finding a *perfect phylogeny*. In this variant, the phylogenetic tree is required to have the property that for each state of a character, the set of all nodes that have that state induces a connected subtree. The general perfect phylogeny problem is NP-hard [4, 20]. When considering the number of possible states per character as a parameter, the problem is fixed parameter tractable [1, 15]. For *binary characters*, having only two states, perfect phylogeny is linear-time solvable [11].

Another common variant of phylogenetic reconstruction is the *parsimony* problem, which calls for a solution with fewest state changes altogether. A change is counted whenever the state of a character changes between a species and an offspring species. This problem is known to be NP-hard [8]. A special case introduced by Camin and Sokal [5] assumes that characters are binary and *directed*, namely, only $0 \rightarrow 1$ changes may occur. Noting by 1 and 0 the presence and absence, respectively, of the character, this means that characters can only be gained during evolution. Another related binary variant is Dollo parsimony [6, 19], which assumes that the change $0 \rightarrow 1$ may happen only once, i.e., a character can be gained once, but it can be lost several times. Both of these variants are polynomially solvable (cf. [7]). When no perfect phylogeny is possible, one can seek a largest subset of the characters which admits a perfect phylogeny. Characters in such a subset are said to be *compatible*. Compatibility problems have been studied extensively (see, e.g., [17]).

In this paper, we discuss a generalization of binary perfect phylogeny which combines the assumptions of both Camin-Sokal parsimony and Dollo parsimony. The setup is as follows: The characters are binary and directed. As in perfect phylogeny, the input is a matrix of character vectors, with the difference that some character states are missing. The question is whether one can complete the missing states in a way admitting a perfect phylogeny. We call this problem *Incomplete Directed Perfect phylogeny (IDP)*.

The problem of handling incomplete phylogenetic data arises whenever some of the data is missing. It is also encountered in the context of morphological characters, where for some species it may be impossible to reliably assign a state to a character. The popular PAUP software package [21] provides an exponential solution to the problem by exhaustively searching the space of missing states. Indeed, the problem of determining whether a set of incomplete *undirected* characters is compatible was shown to be NP-complete, even in the case of binary characters [20].

Quite recently, a novel kind of genomic data has given rise to the same problem: Nikaido et al. [18] use inserted repetitive genomic elements, particularly SINEs, as a source of evolutionary information. SINEs are short DNA sequences that were copied and randomly reinserted into various genomic loci during evolution. The specific insertion events are identifiable by the flanking sequences on both sides of the insertion site. These insertions are assumed to be unique events in evolution, because the odds of having separate insertion events at the very same locus are negligible. Furthermore, a SINE insertion is assumed to be irreversible, i.e., once a SINE sequence has been inserted somewhere along the genome, it is practically impossible for the exact, complete SINE to leave that specific locus. However, the site and its flanking sequences may be lost when a large genomic region, which includes them, is deleted. In that case we do not know whether an insertion had occurred in the missing site. One can model such data by assigning to each locus a character, whose state is '1' if the SINE occurred in that locus, '0' if the locus is present but does not contain the SINE, and '?' if the locus is missing. The resulting reconstruction problem is precisely Incomplete Directed Perfect phylogeny.

The incomplete perfect phylogeny problem becomes polynomial when the characters are directed: Benham et al. [3] studied the compatibility problem on generalized characters. Their work implies an $O(n^2m)$-time algorithm for IDP, where n and m denote the number of species and characters, respectively. A problem related to IDP is the consensus tree problem [2, 13]. This problem calls for constructing a consensus tree from homeomorphic binary subtrees, and is solvable in polynomial time. One can reduce IDP to the latter problem, but the reduction may take $\Omega(n^2m)$ time.

Our approach to the IDP problem is graph theoretic. We first provide several graph and matrix characterizations for solvable instances of binary directed perfect phylogeny. We then reformulate IDP as a *graph sandwich* problem: The input data is recast into two graphs, and solving IDP is shown to be equivalent to finding a graph of a particular type "sandwiched" between them. This formulation allows us to devise a polynomial algorithm for IDP. The deterministic complexity of the algorithm is shown to be $O(nm + k\log^2(n + m))$, for an instance with k 1-entries in the matrix. Alternatively, we give a randomized version of the algorithm which takes $O(nm+k\log(l^2/k)+l(\log l)^3\log\log l)$ expected time, where $l = n + m$. Since an $\Omega(nm)$ lower bound was shown by Gusfield for directed binary perfect phylogeny [11], our algorithm has near optimal time complexity.

The paper is organized as follows: In Section 2 we provide some preliminaries. Section 3 gives the characterizations and the graph sandwich formulation. In Section 4 we present the polynomial algorithm. For lack of space, some proofs are shortened or omitted.

2 Problem Formulation

We first specify some terminology and notation. We reserve the terms *nodes* and *branches* for trees, and will use the terms *vertices* and *edges* for other graphs. Matrices are denoted by an upper-case letter, while their elements are denoted by the corresponding lower-case letter.

Let $G = (V, E)$ be a graph. We denote its set of vertices also by $V(G)$, and its set of edges also by $E(G)$. For a vertex $v \in V$ we define its *degree to a subset* $R \subseteq V$ to be the number of edges connecting v to vertices in R. The *length* of a path in G is the number of edges along it.

Let $\mathcal{T}$ be a rooted tree over a leaf set S with branches directed from the root towards the leaves. For a node x in $\mathcal{T}$ we denote the leaf set of the subtree rooted at x by $L(x)$. $L(x)$ is called a *clade* of $\mathcal{T}$. For consistency, we consider $\emptyset$ to be a clade of $\mathcal{T}$ as well, and call it the *empty clade*. $S, \emptyset$ and all singletons are called *trivial clades*. We denote by $triv(S)$ the collection of all trivial clades. Two sets are said to be *compatible* if they are either disjoint, or one of them contains the other.

Observation 1. *A collection $\mathcal{S}$ of subsets of a set S is the set of clades of some tree over S iff $\mathcal{S}$ contains $triv(S)$ and its subsets are pairwise compatible.*

A tree is uniquely characterized by the set of its clades. The transformation between a branch-node representation of a tree and a list of its clades is straightforward. Hereafter, we identify a tree $\mathcal{T}$ with the set $\{S' : S' \text{ is a clade of } \mathcal{T}\}$. Let $\hat{S}$ be a subset of the leaves of $\mathcal{T}$. Then the subtree of $\mathcal{T}$ *induced* on $\hat{S}$ is the collection $\{\hat{S} \cap S' : S' \in \mathcal{T}\}$ (which defines a tree).

Throughout the paper we denote by $S = \{s_1, \ldots, s_n\}$ the set of all species and by $C = \{c_1, \ldots, c_m\}$ the set of all (binary) characters. For a graph K, we define $C(K) \equiv C \cap V(K)$ and $S(K) \equiv S \cap V(K)$. Let $\mathcal{B}_{n\times m}$ be a binary matrix whose rows correspond to species, each row being the character-vector of the corresponding species. That is, $b_{ij} = 1$ iff the species s_i has the character c_j. A *phylogenetic tree for $\mathcal{B}$* is a rooted tree $\mathcal{T}$ with n leaves corresponding to the n species of S, such that each character c_j is associated with a clade S' of $\mathcal{T}$, satisfying:
(1) $s_i \in S'$ iff $b_{ij} = 1$.
(2) Every non-trivial clade of $\mathcal{T}$ is associated with at least one character.
For a character c, the node x of $\mathcal{T}$, whose clade $L(x)$ is associated with c, is called the *origin* of c w.r.t. $\mathcal{T}$.

Let $\mathcal{A}_{n\times m}$ be a $\{0, 1, ?\}$ matrix in which $a_{ij} = 1$ if s_i has c_j, $a_{ij} = 0$ if s_i lacks c_j, and $a_{ij} =?$ if it is not known whether s_i has c_j. $\mathcal{A}$ is called *incomplete* if it contains at least one '?'. For a character c_j and a value $x \in \{0, ?, 1\}$, the *x-set* of c_j in $\mathcal{A}$ is the set of species $c_j(\mathcal{A}, x) \equiv \{s_i : a_{ij} = x\}$. c_j is called a *null character* if its 1-set is empty.

A binary matrix $\mathcal{B}$ is called a *completion* of $\mathcal{A}$ if $a_{ij} \in \{b_{ij}, ?\}$ for all i, j. Thus, a completion replaces all the ?-s in $\mathcal{A}$ by zeroes and ones. If $\mathcal{B}$ has a phylogenetic tree $\mathcal{T}$, we say that $\mathcal{T}$ is a *phylogenetic tree for $\mathcal{A}$* as well. We also say that *$\mathcal{T}$ explains $\mathcal{A}$ via $\mathcal{B}$*, and that $\mathcal{A}$ is *explainable*. An example of an incomplete matrix $\mathcal{A}$, a completion of $\mathcal{A}$, and a phylogenetic tree which explains $\mathcal{A}$, is given in Figure 1.

The following lemma, closely related to Observation 1, has been proven independently by several authors:

Lemma 1. *A binary matrix $\mathcal{B}$ has a phylogenetic tree iff the 1-sets of every two characters are compatible.*

An analogous lemma holds for undirected characters (cf. [11]). In contrast, for incomplete matrices, even if every pair of columns has a phylogenetic tree, the full matrix might not have one. Such an example was provided by [7] for undirected characters. We provide a simpler example for directed characters in Figure 2.

Characters

Species					
	1	?	0	0	1
	?	?	0	1	0
	?	0	1	?	?

	a	b	c	d	e
r	1	0	0	0	1
s	1	1	0	1	0
t	1	0	1	0	1

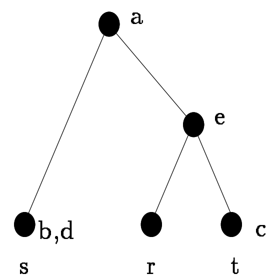

Fig. 1. Left to right: An incomplete matrix $\mathcal{A}$, a completion $\mathcal{B}$ of $\mathcal{A}$, and a phylogenetic tree that explains $\mathcal{A}$ via $\mathcal{B}$. A character x to the right of a vertex v means that v is the origin of x.

Characters

Species			
	1	0	0
	1	1	0
	?	1	1
	0	?	1

Fig. 2. An incomplete matrix which has no phylogenetic tree although every pair of its columns has one.

We are now ready to state the IDP problem:
Incomplete Directed Perfect Phylogeny (IDP):
Instance: An incomplete matrix $\mathcal{A}$.
Goal: Find a tree which explains $\mathcal{A}$, or determine that no such tree exists.

3 Characterizations of Explainable Binary Matrices

3.1 Forbidden Subgraph Characterization

Let $\mathcal{B}$ be a species-characters binary matrix of order $n \times m$. Construct the bipartite graph $G(\mathcal{B}) = (S, C, E)$ with $E = \{(s_i, c_j) : b_{ij} = 1\}$. For a subset $S' \subseteq S$ of species, we say that a character c is *S'-universal* in $\mathcal{B}$, if $S' \subseteq c(\mathcal{B}, 1)$.

An induced path of length four in $G(\mathcal{B})$ is called a *Σ subgraph* if it starts (and therefore ends) at a vertex corresponding to a species (see Figure 3). A graph with no induced Σ subgraph is said to be *Σ-free*.

Proposition 1. *If $G(\mathcal{B})$ is connected and Σ-free, then there exists a character which is S-universal in $\mathcal{B}$.*

Proof. Suppose to the contrary that $G(\mathcal{B})$ has no S-universal character. Consider the collection of all 1-sets of characters in $\mathcal{B}$. Let c be a character whose 1-set is maximal w.r.t. inclusion in this collection. Let s'' be a species which lacks c.

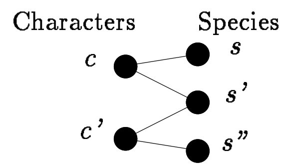

Fig. 3. The Σ subgraph.

Since c has a non-empty 1-set and $G(\mathcal{B})$ is connected, there exists a path from s'' to c in $G(\mathcal{B})$. Consider the shortest such path P. Since $G(\mathcal{B})$ is bipartite, the length of P is odd. P cannot be of length 1, by the choice of s''. Furthermore, if P is of length greater than 3, then its first 5 vertices induce a Σ subgraph, a contradiction. Thus $P = (s'', c', s', c)$ must be of length 3. By maximality of the 1-set of c, it is not contained in the 1-set of c'. Hence, there exists a species s which has the character c but lacks c'. Together with the vertices of P, s induces a Σ subgraph, as depicted in Figure 3, a contradiction.

The following theorem restates Lemma 1 in terms of graph theory.

Theorem 1. *$\mathcal{B}$ has a phylogenetic tree iff $G(\mathcal{B})$ is Σ-free.*

Corollary 1. *Let $\hat{\mathcal{A}}$ be a submatrix of $\mathcal{A}$. If $\mathcal{A}$ is explainable, then so is $\hat{\mathcal{A}}$. Furthermore, if $\mathcal{T}$ explains $\mathcal{A}$, then $\hat{\mathcal{A}}$ is explained by the subtree of $\mathcal{T}$ induced on its rows.*

Let Ψ be a graph property. In the *Ψ sandwich problem* one is given a vertex set V and a partition of $(V \times V) \setminus \{(v, v) : v \in V\}$ into three subsets: E_0 - the forbidden edges, E_1 - the mandatory edges, and $E_?$ - the optional edges. The objective is to find a supergraph of (V, E_1) which satisfies Ψ and contains no forbidden edges. For the property of having no induced Σ subgraphs, the problem is formally defined as follows:
Σ-free-sandwich:
Instance: A vertex set V, and two disjoint edge sets E_0, E_1 over V.
Question: Is there a set F of edges such that $F \supseteq E_1$, $F \cap E_0 = \emptyset$, and the graph (V, F) satisfies Ψ?

Proposition 2. *Σ-free-sandwich is equivalent to IDP.*

Hence, the required graph (V, F) must be "sandwiched" between (V, E_1) and $(V, E_1 \cup E_?)$. The reader is referred to [10, 9] for a discussion of various sandwich problems.

Theorem 1 motivates looking at the IDP problem with input $\mathcal{A}$ as an instance $((S, C), E_0^{\mathcal{A}}, E_?^{\mathcal{A}}, E_1^{\mathcal{A}})$ of the *Σ-free sandwich problem.* Here, $E_x^{\mathcal{A}} = \{(s_i, c_j) : a_{ij} = x\}$, for $x = 0, ?, 1$. In the sequel, we omit the superscript $\mathcal{A}$ when it is clear from the context.

Note, that there is an obvious 1-1 correspondence between completions of $\mathcal{A}$ and possible solutions of $((S, C), E_0, E_?, E_1)$. Hence, in the sequel we refer to matrices and their corresponding sandwich instances interchangeably.

3.2 Forbidden Submatrix Characterizations

A binary matrix $\mathcal{B}$ is called *canonical* if it can be decomposed as follows:
(1) Its $k_0 \geq 0$ left columns are all zero.
(2) Its next $k_1 \geq 0$ columns are all one.
(3) There exist canonical matrices $\mathcal{B}_1, \ldots, \mathcal{B}_l$, such that the rest (0 or more) of the columns of $\mathcal{B}$ form the block-structure illustrated in Figure 4.

We say that a matrix $\mathcal{B}$ *avoids* a matrix $\mathcal{X}$, if no submatrix of $\mathcal{B}$ is identical to $\mathcal{X}$.

Theorem 2. *Let $\mathcal{B}$ be a binary matrix. The following are equivalent:*

1. *$\mathcal{B}$ has a phylogenetic tree.*
2. *$G(\mathcal{B})$ is Σ-free.*
3. *Every matrix obtained by permuting the rows and columns of $\mathcal{B}$ avoids the following matrix:*
$$\mathcal{Z} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$
4. *There exists an ordering of the rows and columns of $\mathcal{B}$ which yields a canonical matrix.*
5. *There exists an ordering of the rows and columns of $\mathcal{B}$ so that the resulting matrix avoids the following matrices:*

$$\mathcal{X}_1 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \mathcal{X}_2 = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}, \mathcal{X}_3 = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, \mathcal{X}_4 = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

Proof.

1⇔2 Theorem 1.
2⇔3 Trivial.
1⇒4 Suppose $\mathcal{T}$ is a tree that explains $\mathcal{B}$. Assign to each node of $\mathcal{T}$ an index which equals its position in a preorder visit of $\mathcal{T}$. Sort all characters according to the preorder indices of their origin nodes (letting null characters come first). Sort all species according to the preorder indices of their corresponding leaves in $\mathcal{T}$. The result is a canonical matrix.
4⇒5 Easily verifiable from the definition of canonical matrices.
5⇒3 Suppose to the contrary that $\mathcal{B}$ has an ordering of its rows and columns, so that rows i_1, i_2, i_3 and columns j_1, j_2 of the resulting matrix compose the submatrix $\mathcal{Z}$. Consider the permutations $\theta_{row}, \theta_{col}$ of the rows and columns of $\mathcal{B}$, respectively, which yield a matrix avoiding $\mathcal{X}_1, \ldots, \mathcal{X}_4$. In this ordering, row $\theta_{row}(i_1)$ necessarily lies between rows $\theta_{row}(i_2)$ and $\theta_{row}(i_3)$, or else, the submatrix $\mathcal{X}_4$ occurs. Suppose that $\theta_{row}(i_2) < \theta_{row}(i_3)$ and $\theta_{col}(j_1) < \theta_{col}(j_2)$, then $\mathcal{X}_3$ occurs, a contradiction. The remaining cases are similar.

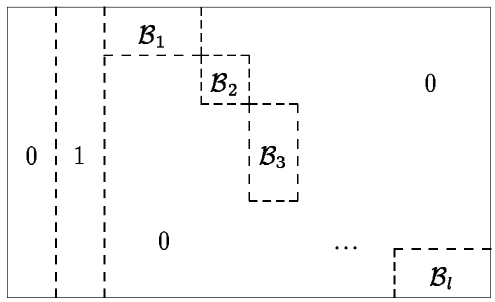

Fig. 4. Recursive definition of canonical matrices. Each $\mathcal{B}_i$ is constructed in the same manner.

Note, that a matrix which avoids $\mathcal{X}_4$ has the consecutive ones property in columns. Gusfield [11, Theorem 3] has proven that a matrix which has a phylogenetic tree can be reordered so as to satisfy that property. In fact, the reordering used by Gusfield's proof generates an essentially canonical matrix.

Klinz et al. [16] study and review numerous problems of permuting matrices to avoid forbidden submatrices.

4 The Algorithm

Let $\mathcal{A}$ be the input matrix. Define $G^x(\mathcal{A}) = (S, C, E_x^{\mathcal{A}})$ for $x = ?, 1$. For a subset $\emptyset \neq S' \subseteq S$, we say that a character is S'*-semi-universal* in $\mathcal{A}$ if its 0-set does not intersect S'.

We now present an algorithm for solving IDP. The algorithm outputs a tree $\mathcal{T}$ which explains $\mathcal{A}$, or outputs *False* if no such tree exists.

1. $G \leftarrow G^1(\mathcal{A}), \mathcal{T} \leftarrow triv(S)$.
2. Remove all S-semi-universal and all null characters from G.
3. **While** $E(G) \neq \emptyset$ **do**:
   (a) **For** each connected component K of G such that $|E(K)| \geq 1$ **do**:
      i. Let $\hat{S} \leftarrow S(K)$.
      ii. Compute the set U of all characters in K which are $\hat{S}$-semi-universal in $\mathcal{A}$.
      iii. **If** $U = \emptyset$ **then** output *False* and **halt**.
      iv. **Otherwise**, remove U from G and update $\mathcal{T} \leftarrow \mathcal{T} \cup \{\hat{S}\}$.
4. Output $\mathcal{T}$.

Theorem 3. *The above algorithm correctly solves IDP.*

Proof. Suppose that the algorithm returns *False*. Then there exists an iteration of the 'while' loop at which some connected component K contained no $S(K)$-semi-universal character. Suppose to the contrary that some F^* solves $\mathcal{A}$, and let $G^* = (S, C, F^*)$. Let H^* be the subgraph of G^* induced by $V(K)$. By definition,

H^* is connected and by Theorem 1, it is also Σ-free. Therefore, by Proposition 1 it has an $S(K)$-universal character. That character must be $S(K)$-semi-universal in $\mathcal{A}$, a contradiction.

On the other hand, suppose that the algorithm returns a collection $\mathcal{T}$ of sets. We shall prove that $\mathcal{T}$ is a tree which explains $\mathcal{A}$. We first prove that the collection $\mathcal{T}$ of sets is pairwise compatible, implying, by Observation 1, that $\mathcal{T}$ is a tree. Let S_1, S_2 be two subsets in $\mathcal{T}$. Let t_i denote the iteration of the 'while' loop at which S_i was added to $\mathcal{T}$, for $i = 1, 2$. If $t_1 = t_2$ then S_1 and S_2 are clearly disjoint. Otherwise, suppose w.l.o.g., that $t_1 < t_2$ and $S_1 \cap S_2 \neq \emptyset$. Let K_1 denote the connected component containing S_1 at iteration t_1 of the algorithm. The edge set of G at iteration t_1 contains the one at iteration t_2. Therefore, K_1 contains the vertices in S_2. It follows that $S_1 \supseteq S_2$.

It remains to show that the resulting tree is a phylogenetic tree for $\mathcal{A}$. Suppose that a character c was removed from G as a part of some set U in Step 3(a)iv. Associate with c the clade $\hat{S}$, which was added to $\mathcal{T}$ at that same step. Observe, that each non-trivial clade $\hat{S} \in \mathcal{T}$ is associated with at least one character. Associate with each S-semi-universal character the clade S. Associate with each null character the empty clade. Finally, define a binary matrix $\mathcal{B}_{n\times m}$ with $b_{sc} = 1$ iff s belongs to the clade S_c associated with c. Since $a_{sc} \neq 1$ for all $s \notin S_c$ and $a_{sc} \neq 0$ for all $s \in S_c$, $\mathcal{B}$ is a completion of $\mathcal{A}$. The claim follows.

The algorithm can be naively implemented in $O(hnm)$ time, where $h \leq \min\{m, n\}$ denotes the height of the reconstructed tree. This can be seen by noting that each iteration of the 'while' loop increases the height of the output tree by one. We give a better bound in the following theorem.

Theorem 4. *The complexity of the algorithm is $O(nm+|E_1|\log^2(n+m))$ deterministic time. Alternatively, there exists a randomized algorithm that solves IDP in $O(nm + |E_1|\log(l^2/|E_1|) + l(\log l)^3 \log\log l)$ expected time, where $l = n + m$.*

Proof. Each iteration of the 'while' loop splits the (potential) clades added in the previous one. Thus, the algorithm performs an iteration per level of the tree returned, and at most h iterations. The connected components of G can be initialized in $O(|E_1| + n + m)$ time, and maintained using a dynamic data structure for graph connectivity. Using the dynamic algorithm of [14] the connected components of G can be maintained during $|E_1|$ edge deletions at a cost of $O(|E_1|\log^2(n+m))$ time spent in Step 3(a)iv. Alternatively, using the Las-Vegas type randomized algorithm for decremental dynamic connectivity [22], the edge deletions can be supported in $O(|E_1|\log(l^2/|E_1|) + l(\log l)^3 \log\log l)$ expected time.

The connected components of G must be explicitly recomputed from the dynamic data structure for each iteration. This takes $O(h(m + n)) = O(nm)$ time in total. Since each set added to $\mathcal{T}$ in Step 3(a)iv corresponds to at least one character, and each character is associated with exactly one set, updating $\mathcal{T}$ requires in total $O(nm)$ time.

It remains to show how semi-universal characters can be efficiently found in Step 3(a)ii. Let $G(t)$ denote the graph G at iteration t of the 'while' loop. For a character c, denote by $d_c^1(t)$ its degree in $G(t)$, and by $d_c^?(t)$ the degree of c in $G^?(\mathcal{A})$ to its connected component K_c in $G(t)$. Given $d_c^1(t)$,$d_c^?(t)$, one can check in $O(1)$ time whether c is $S(K_c)$-semi-universal. $d_c^1(t)$ remains unchanged throughout, and equals $d_c^1(1)$. $d_c^?(t)$ can be maintained as follows. $d_c^?(1)$ is initialized in $O(|E_?|)$ time (given the connected components of $G(1)$). At the beginning of iteration t, $d_c^?(t+1)$ is initialized to $d_c^?(t)$. Each time a connected component K_c of $G(t)$ is split into sub-components $K_1, \ldots, K_l$ due to the removal of characters in Step 3(a)iv, we update $d_c^?(t+1)$ as follows: For $c \in C(K_j)$, we decrease $d_c^?(t+1)$ by $|\{(s,c) \in E_? : s \in S(K_p), p \neq j\}|$. This takes $O(|E_?|)$ time for all c, t. Finally, finding the semi-universal characters over all iterations costs $O(hm)$ time. The complexity follows.

We remark, that an $\Omega(mn)$ time lower bound for (undirected) binary perfect phylogeny was proven by Gusfield [11]. A closer look at Gusfield's proof reveals that it applies, as is, also to the directed case. As IDP generalizes directed binary perfect phylogeny, any algorithm for this problem would require $\Omega(mn)$ time.

5 Concluding Remarks

We have given a polynomial algorithm for reconstructing a phylogeny from incomplete binary directed data, using a graph theoretic reformulation of the problem. The algorithm is near optimal and takes $\tilde{O}(nm)$ time.

An interesting question regarding IDP is whether one can identify if there exists a "most general" solution, so that all others are obtained from it by refinements (additional clades). We have proven that in case a "most general" solution exists, the algorithm described here provides that solution. The full version of this manuscript will include this proof, along with a more general polynomial time algorithm, that also determines if such a solution exists.

Acknowledgments

We thank Dan Graur and Tal Pupko for drawing our attention to this phylogenetic problem, and for helpful discussions. We thank Joe Felsenstein, Dan Gusfield, Haim Kaplan, Mike Steel, and an anonymous CPM '00 referee for referring us to helpful literature.

The first author was supported by the Clore foundation scholarship. The second author was supported in part by the Israel Science Foundation formed by the Israel Academy of Sciences and Humanities. The third author was supported by an Eshkol fellowship from the Ministry of Science, Israel.

References

1. R. Agarwala and D. Fernández-Baca. A polynomial-time algorithm for the perfect phylogeny problem when the number of character states is fixed. *SIAM Journal on Computing*, 23(6):1216–1224, 1994.

2. A. V. Aho, Y. Sagiv, T. G. Szymanski, and J. D. Ullman. Inferring a tree from lowest common ancestors with an application to the optimization of relational expressions. *SIAM Journal on Computing*, 10(3):405–421, 1981.
3. C. Benham, S. Kannan, M. Paterson, and T.J. Warnow. Hen's teeth and whale's feet: generalized characters and their compatibility. *Journal of Computational Biology*, 2(4):515–525, 1995.
4. H. L. Bodlaender, M. R. Fellows, and T. J. Warnow. Two strikes against perfect phylogeny. In W. Kuich, editor, *Proc. 19th ICALP*, pages 273–283, Berlin, 1992. Springer. Lecture Notes in Computer Science, Vol. 623.
5. J. H. Camin and R. R. Sokal. A method for deducing branching sequences in phylogeny. *Evolution*, 19:409–414, 1965.
6. L. Dollo. Le lois de l'évolution. *Bulletin de la Societé Belge de Géologie de Paléontologie et d'Hydrologie*, 7:164–167, 1893.
7. J. Felsenstein. *Inferring Phylogenies*. Sinaur Associates, Sunderland, Massachusetts, 2000. In press.
8. L. R. Foulds and R. L. Graham. The Steiner problem in phylogeny is NP-complete. *Advances in Applied Mathematics*, 3:43–49, 1982.
9. M. C. Golumbic. Matrix sandwich problems. *Linear algebra and its applications*, 277:239–251, 1998.
10. M. C. Golumbic, H. Kaplan, and R. Shamir. Graph sandwich problems. *Journal of Algorithms*, 19:449–473, 1995.
11. D. Gusfield. Efficient algorithms for inferring evolutionary trees. *Networks*, 21:19–28, 1991.
12. D. Gusfield. *Algorithms on Strings, Trees, and Sequences*. Cambridge University Press, 1997.
13. M. Henzinger, V. King, and T.J. Warnow. Constructing a tree from homeomorphic subtrees, with applications to computational evolutionary biology. *Algorithmica*, 24:1–13, 1999.
14. J. Holm, K. de Lichtenberg, and M. Thorup. Poly-logarithmic deterministic fully-dynamic algorithms for connectivity, minimum spanning tree, 2-edge and biconnectivity. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing (STOC-98)*, pages 79–89, New York, May 23–26 1998. ACM Press.
15. S. Kannan and T. Warnow. A fast algorithm for the computation and enumeration of perfect phylogenies. *SIAM Journal on Computing*, 26(6):1749–1763, 1997.
16. B. Klinz, R. Rudolf, and G. J. Woeginger. Permuting matrices to avoid forbidden submatrices. *Discrete applied mathematics*, 60:223–248, 1995.
17. C. A. Meecham and G. F. Estabrook. Compatibility methods in systematics. *Annual Review of Ecology and Systematics*, 16:431–446, 1985.
18. M. Nikaido, A. P. Rooney, and N. Okada. Phylogenetic relationships among cetartiodactyls based on insertions of short and long interspersed elements: Hippopotamuses are the closest extant relatives of whales. *Proceedings of the National Academy of Science USA*, 96:10261–10266, 1999.
19. W. J. Le Quesne. The uniquely evolved character concept and its cladistic application. *Systematic Zoology*, 23:513–517, 1974.
20. M. A. Steel. The complexity of reconstructing trees from qualitative characters and subtrees. *Journal of Classification*, 9:91–116, 1992.
21. D. L. Swofford. *PAUP, Phylogenetic Analysis Using Parsimony (and Other Methods)*. Sinaur Associates, Sunderland, Massachusetts, 1998. Version 4.
22. M. Thorup. Decremental dynamic connectivity. *Journal of Algorithms*, 33:229–243, 1999.