On the Generality of Phylogenies from Incomplete Directed Characters

Itsik Pe'er[1], Ron Shamir[1], and Roded Sharan[1]

School of Computer Science, Tel-Aviv University, Tel-Aviv, 69978 Israel,
{izik,rshamir,roded}@post.tau.ac.il

Abstract. We study a problem that arises in computational biology, when wishing to reconstruct the phylogeny of a set of species. In Incomplete Directed Perfect Phylogeny (IDP), the characters are binary and directed (i.e., species can only gain characters), and the states of some characters are unknown. The goal is to complete the missing states in a way consistent with a perfect phylogenetic tree. This problem arises in classical phylogenetic studies, when some states are missing or undetermined, and in recent phylogenetic studies based on repeat elements in DNA. The problem was recently shown to be polynomial. As different completions induce different trees, it is desirable to find a *general* solution tree. Such a solution is consistent with the data, and every other consistent solution can be obtained from it by node splitting. Unlike the situation for complete datasets, a general solution may not exist for IDP instances. We provide a polynomial algorithm to find a general solution for an IDP instance, or determine that none exists.

1 Introduction

A *phylogenetic tree* describes the divergence patterns leading from a single ancestor species to its contemporary descendants. The task of *phylogenetic reconstruction* is to infer a phylogenetic tree from information regarding the contemporary species (see, e.g., [2]).

The character-based approach to tree reconstruction describes contemporary species by their attributes or *characters*. Each character takes on one of several possible discrete *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 phylogenetic tree along with the suggested character vectors of the internal nodes. This output must satisfy properties specified by the problem variant.

In this paper, we discuss a phylogenetic reconstruction problem called *Incomplete Directed Perfect Phylogeny (IDP)* [6]. It is assumed that each character is binary, where its absence or presence is denoted by 0 or 1, respectively. A character may be gained at most once (across the phylogenetic tree), but may never be lost. The input is a matrix of character vectors, where some character states are missing. The question is whether one can complete the missing states so that the resulting matrix admits a tree satisfying the above assumptions.

The problem of handling incomplete phylogenetic data arises whenever some of the data is missing or ambiguous. Quite recently, a novel kind of genomic data has given rise to the same problem: Nikaido et al. [5] use inserted repetitive genomic elements, particularly SINEs (Short Interspersed Nuclear Elements), as a source of evolutionary information. SINEs are short DNA sequences that were copied and randomly reinserted into various genomic loci during evolution. The distinct insertion loci are identifiable by the flanking sequences on both sides of the insertion site. These insertions are assumed to be unique events in evolution. 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 inserted segment along with its flanking sequences may be lost when a large genomic region, which includes them, is deleted. In that case we do not know whether a SINE insertion had occurred in the missing site prior to its deletion. 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.

Previous studies of related problems implied $\Omega(n^2m)$-time algorithms for IDP [1, 3], where n and m denote the number of species and characters, respectively. In a recent work [6] we provided near optimal $O(nm \cdot polylog(n+m))$-time algorithms for the problem.

In this paper we tackle a different aspect of IDP. Often there is more than one tree that is consistent with the data. When the input matrix is complete, there is always a tree $\mathcal{T}^*$ that is *general*, i.e., it is a solution, and every other tree consistent with the data can be obtained from $\mathcal{T}^*$ by node splitting. In other words, $\mathcal{T}^*$ describes all the definite information in the data, and ambiguities (nodes with three or more children) can be resolved by additional information. This is not always the case if the data matrix is incomplete: There may or may not be a general solution tree. In the latter case, any particular solution tree we choose may be ruled out by additional information, while this information is still consistent with an alternative solution tree. It is thus desirable to decide if a general solution exists and to generate such a solution if the answer is positive.

In this study we provide answers to both questions. We prove that an algorithm from [6], which we call Solve_IDP, provides the general solution of a problem instance, if such exists. We also give an algorithm which determines if the solution $\mathcal{T}$ produced by Solve_IDP is indeed general. The complexity of the latter algorithm is $O(nm + kd)$, where k is the number of 1-entries in the input matrix, and d denotes the maximum degree of $\mathcal{T}$.

The paper is organized as follows: In Section 2 we provide some preliminaries and background on the IDP problem. In Section 3 we analyze the generality of the solution produced by Solve_IDP and the complexity of testing generality. For lack of space some proofs are sketched or omitted.

2 Preliminaries

We first specify some terminology and notation. Matrices are denoted by an upper-case letter, while their elements are denoted by the corresponding lower-case letter. Let $G = (V, E)$ be an undirected graph. We denote its set of vertices V also by $V(G)$. A nonempty set $W \subseteq V$ is *connected* in G, if there is a path in G between every pair of vertices in W.

Let $\mathcal{T}$ be a rooted tree over a leaf set S. The *out-degree* of a node x in $\mathcal{T}$ is its number of children, and is denoted by $d(x)$. 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*.

Observation 1 *(cf. [4]) A collection $\mathcal{C}$ of subsets of a set S is the set of clades of some tree over S if and only if $\mathcal{C}$ contains the trivial clades and for every intersecting pair of its subsets, one contains the other.*

A tree $\mathcal{T}$ is uniquely characterized by its set of clades. The transformation between a branch-node representation of a tree and a list of its clades is straightforward. Thus, we hereafter identify a tree with the set of its clades.

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 $S(K) \equiv S \cap V(K)$. Let $\mathcal{B}_{n\times m}$ be a binary matrix whose rows and columns correspond to species and characters, respectively, and $b_{ij} = 1$ if and only if 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 is associated with a clade of $\mathcal{T}$, and the following properties are satisfied:
(1) If c_j is associated with a clade S' then $s_i \in S'$ if and only if $b_{ij} = 1$.
(2) Every non-trivial clade of $\mathcal{T}$ is associated with at least one character.

A $\{0, 1, ?\}$ matrix is called *incomplete*. For convenience, we also consider binary matrices as incomplete. Let $\mathcal{A}_{n\times m}$ be an incomplete 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. For two subsets $S' \subseteq S$ and $C' \subseteq C$ we denote by $\mathcal{A}|_{S',C'}$ the submatrix of $\mathcal{A}$ induced on $S' \times C'$. For a character c_j and a state $x \in \{0, 1, ?\}$, the *x-set* of c_j in $\mathcal{A}$ is the set of species $\{s_i \in S : a_{ij} = x\}$. c_j is called a *null character* if its 1-set is empty. We denote $E_x^{\mathcal{A}} = \{(s_i, c_j) : a_{ij} = x\}$, for $x = 0, 1, ?$. (In the sequel, we omit the superscript $\mathcal{A}$ when it is clear from the context.)

A binary matrix $\mathcal{B}$ is called a *completion* of $\mathcal{A}$ if $E_1^{\mathcal{A}} \subseteq E_1^{\mathcal{B}} \subseteq E_1^{\mathcal{A}} \cup E_?^{\mathcal{A}}$. 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* $\mathcal{B}$. An example of these definitions is given in Figure 1.

The problem of Incomplete Directed Perfect Phylogeny is defined as follows:
Incomplete Directed Perfect Phylogeny (IDP):
Instance: An incomplete matrix $\mathcal{A}$.
Goal: Find a phylogenetic tree for $\mathcal{A}$, or determine that no such tree exists.

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 $E = \{(s_i, c_j) : b_{ij} = 1\}$. An induced path

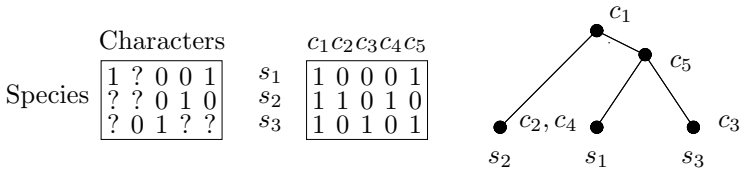

Fig. 1. Left to right: An incomplete matrix $\mathcal{A}$, a completion $\mathcal{B}$ of $\mathcal{A}$, and a phylogenetic tree that explains $\mathcal{A}$ via $\mathcal{B}$. Each character is written to the right of the root of its associated clade. (Example taken from [6].)

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. A bipartite graph with no induced Σ subgraph is said to be Σ*-free*.

We now recite several characterizations of IDP.

Theorem 2 ([6]). *$\mathcal{B}$ has a phylogenetic tree if and only if $G(\mathcal{B})$ is Σ-free.*

In [6] we used Theorem 2 to reformulate IDP as a graph sandwich problem. Finding a completion of an input matrix $\mathcal{A}$ was shown to be equivalent to finding a Σ-free supergraph of $G(\mathcal{A})$ whose set of edges does not intersect $E_0^{\mathcal{A}}$.

Corollary 1. *Let $\hat{S} \subseteq S$ and $\hat{C} \subseteq C$ be subsets of the species and characters, respectively. If $\mathcal{A}$ has a phylogenetic tree, then so does $\mathcal{A}|_{\hat{S},\hat{C}}$.*

Corollary 2. *Let $\mathcal{A}$ be a matrix explained by a tree $\mathcal{T}$ and let $\hat{S} = L(x)$ be a clade in $\mathcal{T}$. Then $\mathcal{A}|_{\hat{S}\times C}$ is explained by the subtree of $\mathcal{T}$ rooted at x.*

For a subset $S' \subseteq S$ of species, we say that a character c is S'*-universal* in $\mathcal{B}$, if its 1-set contains S' (i.e., every species in S' has that character).

Proposition 1 ([6]). *If $G(\mathcal{B})$ is connected and Σ-free, then there exists a character which is S-universal in $\mathcal{B}$.*

2.1 An Algorithm for IDP

In this section we briefly describe an algorithm for IDP, given in [6]. Let $\mathcal{A}$ be the input matrix. We denote by $B(\mathcal{A})$ the binary matrix of $\mathcal{A}$'s dimension with ?-s replaced by zeros. Define $G(\mathcal{A}) \equiv G(B(\mathcal{A})) = (S, C, E_1^{\mathcal{A}})$. For a nonempty subset $S' \subseteq S$, we say that a character is S'*-semi-universal* in $\mathcal{A}$ if its 0-set does not intersect S'.

The algorithm for solving IDP is described in Figure 2. It outputs the set of non-empty clades of a tree explaining $\mathcal{A}$, or *False* if no such tree exists. The algorithm is recursive and is initially called with Solve_IDP($\mathcal{A}$). It was shown in [6] that Solve_IDP has a deterministic implementation which takes $O(nm+|E_1|\log^2 l)$ time, and a randomized implementation which takes $O(nm+|E_1|\log(l^2/|E_1|)+l(\log l)^3\log\log l)$ expected time, where $l = n + m$.

Solve_IDP($\mathcal{A}$)**:**

1. **If** $|S| > 1$ then **do**:
   (a) Remove all S-semi-universal characters and all null characters from $G(\mathcal{A})$.
   (b) If the resulting graph G' is connected then output *False* and **halt**.
   (c) Otherwise, let $K_1, \ldots, K_r$ be the connected components of G', and let $\mathcal{A}_1, \ldots, \mathcal{A}_r$ be the corresponding sub-matrices of $\mathcal{A}$.
   (d) **For** $i = 1, \ldots, r$ **do**: Solve_IDP($\mathcal{A}_i$).
2. Output S.

Fig. 2. An algorithm for solving IDP [6].

3 Determining the Generality of the Solution

A 'yes' instance of IDP may have several distinct phylogenetic trees as solutions. These trees may be related in the following way: We say that a tree $\mathcal{T}$ *generalizes* a tree $\mathcal{T}'$, and write $\mathcal{T} \subseteq \mathcal{T}'$, if every clade of $\mathcal{T}$ is a clade of $\mathcal{T}'$, i.e., the evolutionary hypothesis expressed by $\mathcal{T}'$ includes all the details of the hypothesis expressed by $\mathcal{T}$, and possibly more. Therefore, $\mathcal{T}'$ represents a more specific hypothesis, and $\mathcal{T}$ represents a more general one. We say that a tree $\mathcal{T}$ is the *general solution* of an instance $\mathcal{A}$, if $\mathcal{T}$ explains $\mathcal{A}$, and generalizes every other tree which explains $\mathcal{A}$. Figure 3 demonstrates the definitions, and also gives an example of an instance which has no general solution.

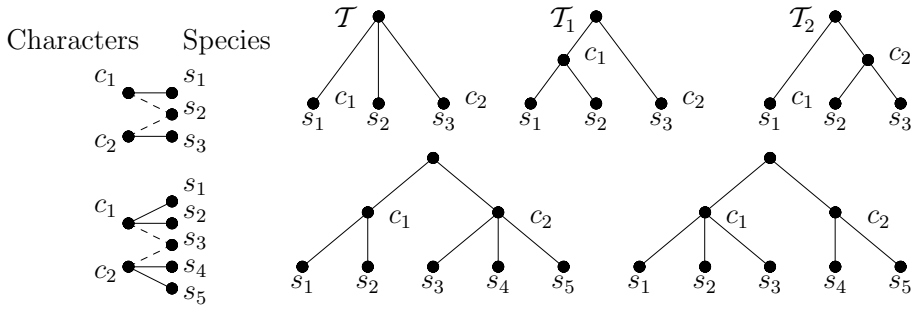

Fig. 3. Top left: An IDP instance which has a general solution. Dashed lines denote $E_?$-edges, while solid lines denote E_1-edges. Top-right: $\mathcal{T}, \mathcal{T}_1$ and $\mathcal{T}_2$ are the possible solutions. $\mathcal{T}$ generalizes $\mathcal{T}_1$ and $\mathcal{T}_2$ (which are obtained by splitting the root node of $\mathcal{T}$), and is the general solution. Bottom left: An IDP instance which has no general solution. Bottom middle and bottom right: Two possible solutions. The only tree which generalizes both solutions is the tree comprised of the trivial clades only, but this tree is not a solution.

3.1 Finding a General Solution

We prove in this section that whenever a general solution exists, Solve_IDP finds it. We use the following notation: Let $\mathcal{A}$ be an incomplete matrix and let $\hat{S} \subseteq S$. We denote by $W_{\mathcal{A}}(\hat{S})$ the set of $\hat{S}$-semi-universal characters in $\mathcal{A}$. Note that if $\mathcal{A}$ is binary, then $W_{\mathcal{A}}(\hat{S})$ is its set of $\hat{S}$-universal characters. We define the operator $\widetilde{\ }$ on incomplete matrices: We denote by $\widetilde{\mathcal{A}}$ the submatrix $\mathcal{A}|_{S,C\setminus W_{\mathcal{A}}(S)}$ of $\mathcal{A}$. In particular, $G(\widetilde{\mathcal{A}})$ is the graph produced from $G(\mathcal{A})$ by removing its set of S-semi-universal vertices.

Lemma 1. *Let $\mathcal{T}$ be the general solution for an instance $\mathcal{A}$ of IDP. Let $S' = L(x)$ be a clade of $\mathcal{T}$, corresponding to some node x. Let $\mathcal{T}'$ be the subtree of $\mathcal{T}$ rooted at x, and let $\mathcal{A}'$ be the instance induced on $S' \cup C$. Then $\mathcal{T}'$ is the general solution for $\mathcal{A}'$.*

Proof. By Corollary 2, $\mathcal{T}'$ explains $\mathcal{A}'$. Suppose that $\mathcal{T}''$ also explains $\mathcal{A}'$ and $\mathcal{T}' \not\subseteq \mathcal{T}''$. Then $\hat{\mathcal{T}} = (\mathcal{T} \setminus \mathcal{T}') \cup \mathcal{T}''$ explains $\mathcal{A}$, and $\mathcal{T} \not\subseteq \hat{\mathcal{T}}$, a contradiction. □

A nonempty clade of a tree is called *maximal* if the only clade that properly contains it is S.

Lemma 2. *Let $\mathcal{T}$ be a phylogenetic tree for a binary matrix $\mathcal{B}$. A non-empty clade S' of $\mathcal{T}$ is maximal if and only if S' is the species set of some connected component of $G(\widetilde{\mathcal{B}})$.*

Proof. Suppose that S' is a maximal clade of $\mathcal{T}$. We claim that S' is contained in some connected component K of $G(\widetilde{\mathcal{B}})$. If $|S'| = 1$ this trivially holds. Otherwise, the character c associated with S' connects all its species, and $c \notin W_{\mathcal{B}}(S)$, proving the claim. Proposition 1 implies that S is disconnected in $G(\widetilde{\mathcal{B}})$ and, therefore, $S' \subseteq S(K) \subset S$. Suppose to the contrary that $S(K)$ properly contains S'. In particular, $|S(K)| > 1$. By Proposition 1, there exists a character c' in $G(\widetilde{\mathcal{B}})$ whose 1-set is $S(K)$. Hence, $S(K)$ must be a clade of $\mathcal{T}$ which is associated with c', contradicting the maximality of S'.

To prove the converse, let S' be the species set of some connected component K of $G(\widetilde{\mathcal{B}})$. We first claim that S' is a clade. If $|S'| = 1$, S' is a trivial clade. Otherwise, by Proposition 1, there exists an S'-universal character c' in $G(\widetilde{\mathcal{B}})$. Since K is a connected component, c' has no neighbors in $S \setminus S'$. Hence, S' must be a clade in $\mathcal{T}$. Suppose to the contrary that S' is not maximal, then it is properly contained in a maximal clade S'', which by the previous direction is the species set of K, a contradiction. □

Theorem 3. *Solve_IDP produces the general solution for every IDP instance that has one.*

Proof. Let $\mathcal{A}$ be an instance of IDP for which there exists a general solution $\mathcal{T}^*$. Let $\mathcal{T}_{alg}$ be the solution tree produced by Solve_IDP. By definition $\mathcal{T}^* \subseteq \mathcal{T}_{alg}$. Suppose to the contrary that $\mathcal{T}^* \neq \mathcal{T}_{alg}$. Let S' be the largest clade in $\mathcal{T}_{alg} \setminus \mathcal{T}^*$

(S' must be non-trivial), and let S'' be the smallest clade in $\mathcal{T}_{alg}$ which properly contains it. Let $\mathcal{A}'$ be the instance induced on $S'' \cup C$. By Corollary 2, $\mathcal{A}'$ is explained by the corresponding subtrees $\mathcal{T}'_{alg}$ of $\mathcal{T}_{alg}$ and $\mathcal{T}'^*$ of $\mathcal{T}^*$. By Lemma 1, $\mathcal{T}'^*$ is the general solution of $\mathcal{A}'$. Due to the recursive nature of Solve_IDP, it produces $\mathcal{T}'_{alg}$ when invoked with input $\mathcal{A}'$. Thus, w.l.o.g., one can assume that $S'' = S$ and S' is a maximal clade of $\mathcal{T}_{alg}$.

Suppose that $\mathcal{T}^*$ explains $\mathcal{A}$ via a completion $\mathcal{B}^*$, and let $G^* = G(\mathcal{B}^*)$. Since S' is a maximal clade, it is reported during a second level call of Solve_IDP$(\cdot)$ (the call at the first level reports the trivial clade S). Hence, it must be the species set of some connected component K in $G(\widetilde{\mathcal{A}})$. Since every S-universal character in G^* is S-semi-universal in $\mathcal{A}$, S' is contained in some connected component K^* of $G(\widetilde{B^*})$. Denote $S^* \equiv S(K^*)$. By Lemma 2, S^* is a maximal clade of $\mathcal{T}^*$. Since $S' \notin \mathcal{T}^*$, we have $S' \neq S^*$, and therefore, $S^* \supset S'$. But $\mathcal{T}^* \subseteq \mathcal{T}_{alg}$, implying that S^* is also a non-trivial clade of $\mathcal{T}_{alg}$, in contradiction to the maximality of S'. □

3.2 Determining the Existence of a General Solution

We give in this section a characterization of IDP instances that admit a general solution. We also provide an algorithm to determine whether the solution tree $\mathcal{T}$ returned by Solve_IDP is general. The complexity of the latter algorithm is shown to be $O(mn + |E_1|d)$, where d is the maximum out-degree of $\mathcal{T}$.

Let $\mathcal{A}$ be a 'yes' instance of IDP. Consider a recursive call Solve_IDP$(\mathcal{A}')$ nested within Solve_IDP$(\mathcal{A})$, where $\mathcal{A}' = \mathcal{A}|_{C',S'}$. Let $K_1, \ldots, K_r$ be the connected components of $G(\widetilde{\mathcal{A}'})$ computed in Step 1c. Observe that $S(K_1), \ldots, S(K_r)$ are clades to be reported by recursive calls launched during Solve_IDP$(\mathcal{A}')$. A set U of characters is said to be (K_i, K_j)-*critical* if: (1) Characters in U are both $S(K_i)$-semi-universal and $S(K_j)$-semi-universal in $\mathcal{A}'$; and (2) removing U from $G(\widetilde{\mathcal{A}'})$ disconnects $S(K_i)$. Note that by definition of U, $U \subseteq W_{\mathcal{A}'}(S(K_i))$, and $a'_{sc} =?$ for all $c \in U, s \in S(K_j)$. A clade $S(K_i)$ is called *optional* with respect to $\mathcal{A}$, if $r \geq 3$ and there exists a (K_i, K_j)-critical set for some index $j \neq i$. Otherwise, we say that $S(K_i)$ is *supported*. In the example of Figure 3 (bottom), $K_1 = \{s_1, s_2, c_1\}, K_2 = \{s_3\}, K_3 = \{s_4, s_5, c_2\}$. The set $U = \{c_1\}$ is (K_1, K_2)-critical, so $S(K_1) = \{s_1, s_2\}$ is optional.

Theorem 4. *Let $\mathcal{T}_{alg}$ be a tree produced by Solve_IDP on instance $\mathcal{A}$. Then $\mathcal{T}_{alg}$ is the general solution for $\mathcal{A}$ if and only if all its clades are supported.*

Proof. $\Rightarrow$ Suppose to the contrary that $\mathcal{T}_{alg}$ contains an optional clade with respect to $\mathcal{A}$. W.l.o.g., assume it is maximal, i.e., during the recursive call Solve_IDP$(\mathcal{A})$, $G' = G(\widetilde{\mathcal{A}})$ has $r \geq 3$ connected components, $K_1, \ldots, K_r$, and there exists a (K_i, K_j)-critical set U (for some $1 \leq i \neq j \leq r$). Let $\mathcal{A}_i, \mathcal{A}_j$ and $\mathcal{A}_{ij}$ be the sub-instances induced on K_i, K_j and $K_i \cup K_j$, respectively. Consider the tree $\mathcal{T}'$ which is produced by a small modification to the execution of Solve_IDP$(\mathcal{A})$: Instead of recursively invoking Solve_IDP$(\mathcal{A}_i)$ and

Solve_IDP($\mathcal{A}_j$), call Solve_IDP($\mathcal{A}_{ij}$). Then $\mathcal{T}'$ is a phylogenetic tree which explains $\mathcal{A}$, but $\mathcal{T}'$ includes the clade $S(K_i \cup K_j)$. Since $r \geq 3$, $S(K_i \cup K_j)$ is non-trivial and is not a clade of $\mathcal{T}_{alg}$, a contradiction.

$\Leftarrow$ Suppose that $\mathcal{T}_{alg}$ is not the general solution for $\mathcal{A}$, i.e., there exists a solution $\mathcal{T}^*$ of $\mathcal{A}$ such that $\mathcal{T}_{alg} \not\subseteq \mathcal{T}^*$. We shall prove the existence of an optional clade with respect to $\mathcal{A}$. (The reader is referred to the example in Figure 4 for notation and intuition. The example follows the steps of the proof, leading to the identification of an optional clade.) Let $\mathcal{B}^*$ be a completion of $\mathcal{A}$ which is explained by $\mathcal{T}^*$, and denote $G^* = G(\mathcal{B}^*)$. Let $S' \in \mathcal{T}_{alg} \setminus \mathcal{T}^*$ be the largest clade reported by Solve_IDP which is not a clade of $\mathcal{T}^*$. W.l.o.g. (as argued in the proof of Theorem 3), S' is a maximal clade of $\mathcal{T}_{alg}$, and $S' = S(K_1)$, where $K_1, \ldots, K_r$ are the connected components of $G(\widetilde{\mathcal{A}})$.

Let $\{S_i^*\}_{i=1}^t$ be the nested set of clades in $\mathcal{T}^*$ that contain S': $S = S_1^* \supset \cdots \supset S_t^* \supset S'$. For each $i = 1, \ldots, t$, let C_i^* be the set of characters in $\mathcal{B}^*$ whose 1-set is non-empty and is properly contained in S_i^*. Denote $\mathcal{B}_i^* = \mathcal{B}^*|_{S_i^*, C_i^*}$ and let $H_i^* = G(\mathcal{B}_i^*)$, i.e., H_i^* is the subgraph of G^* induced on $S_i^* \cup C_i^*$. Let H_i be the subgraph of $G(\mathcal{A})$ induced on the same vertex set. Since G^* is a supergraph of $G(\mathcal{A})$, each H_i^* is a supergraph of H_i.

Claim. S' is disconnected in H_t^* and, therefore, also in H_t.

Proof. Suppose to the contrary that S' is contained in some connected component K^* of H_t^*. $S(K^*)$ is thus a clade of the (unique) phylogenetic tree for $\mathcal{B}_t^*$ and, therefore, also a clade of $\mathcal{T}^*$. It follows that $S_t^* \supset S(K^*) \supset S'$, where the first containment follows from the fact that H_t^* is disconnected, and the second from the assumption that S' is not a clade of $\mathcal{T}^*$. This contradicts the minimality of S_t^*. □

We now return to the proof of Theorem 4. Recall that S' is connected in $H_1 = G(\widetilde{A})$. Thus, the previous claim implies that $t > 1$. Let K_p be a connected component of $G(\widetilde{\mathcal{A}})$ such that $S(K_p) \subseteq S \setminus S_2^*$. Such a component exists since $G(\widetilde{\mathcal{A}})$ is not connected and S_2^* is the species set of one of its components. Let l be the minimal index such that there exists some connected component K_i of $G(\widetilde{\mathcal{A}})$ for which $S(K_i)$ is disconnected in H_l. l is properly defined as $S(K_1) = S'$ is disconnected in H_t. $l > 1$, since otherwise some K_i is disconnected in H_1 and, therefore, also in its subgraph $G(\widetilde{\mathcal{A}})$, in contradiction to the definition of $K_1, \ldots, K_r$. By minimality of l, $S_l^* \supseteq S(K_i)$. Also, $S_l^* \supseteq S_t^* \supset S' = S(K_1)$, so $S_l^* \neq S(K_i)$. We now claim that there exists some connected component K_j of $G(\widetilde{\mathcal{A}})$, $j \neq i$, such that $S(K_j) \subseteq S_l^*$. Indeed, if $i \neq 1$ then $j = 1$. If $i = 1$ then $l = t$ (by an argument similar to that in the proof of Claim 3.2), and since $S_l^* \setminus S'$ is non-empty, it intersects $S(K_j)$ for some $j \neq i$. By minimality of l, $S(K_j)$ is properly contained in $S_l^* \setminus S'$.

Define $U \equiv W_{G^*}(S_l^*)$. By definition all characters in U are S_l^*-universal in G^*, and are thus both K_i-semi-universal and K_j-semi-universal in $\mathcal{A}$. $S(K_i)$ is disconnected in $H_l = G(\mathcal{A}|_{C_l^*, S_l^*})$. Since K_i is a connected component of $G(\widetilde{\mathcal{A}})$, $S(K_i)$ is disconnected in $G(\mathcal{A}|_{C_l^*, S})$, implying that U is a (K_i, K_j)-critical set.

Also, K_i, K_j and K_p are distinct, implying that $r \geq 3$. Hence, U demonstrates that $S(K_i)$ is optional. □

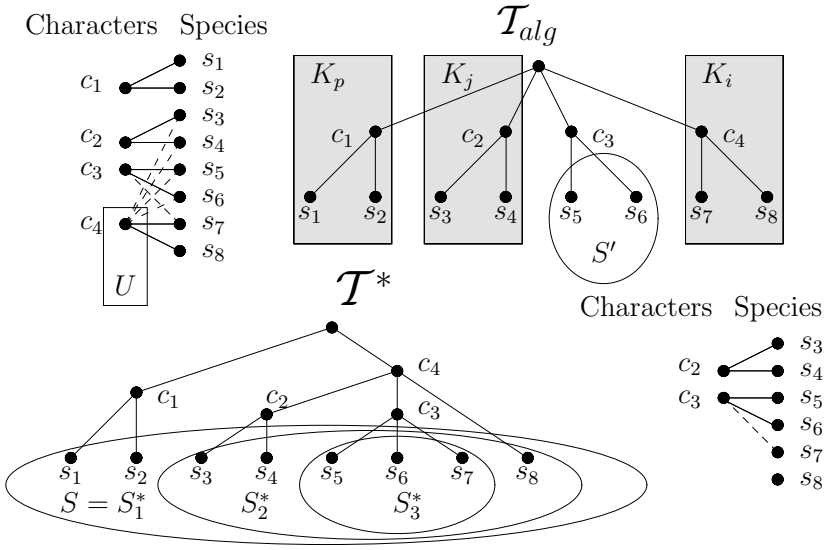

Fig. 4. An example demonstrating the proof of the 'if' part of Theorem 4, using the same notation. Left: A graphical representation of an input instance $\mathcal{A}$. Dashed lines denote $E_?$-edges, while solid lines denote E_1-edges. Top right: The tree $\mathcal{T}_{alg}$ produced by Solve_IDP. Bottom middle: A tree $\mathcal{T}^*$ corresponding to a completion $\mathcal{B}^*$ that uses all the edges in $E_?$. Bottom right: The graphs H_2 (solid edges) and H_2^* (solid and dashed edges). $\mathcal{T}_{alg} \not\subseteq \mathcal{T}^*$, and $S' = \{s_5, s_6\}$. There are $t = 3$ clades of $\mathcal{T}^*$ which contain S': $S_1^* = \{s_1, \ldots, s_8\}$, $S_2^* = \{s_3, \ldots, s_8\}$, and $S_3^* = \{s_5, s_6, s_7\}$. The component $K_p = \{c_1, s_1, s_2\}$ has its species in $S \setminus S_2^*$. Since $W_{\mathcal{A}}(S) = W_{\mathcal{B}^*}(S) = \emptyset$, $H_1 = G(\mathcal{A})$. Since $W_{\mathcal{B}^*}(S_2^*) = \{c_4\}$, the species set of the connected component $K_i = \{s_7, s_8, c_4\}$ is disconnected in H_2, implying that $l = 2$. For a choice of $K_j = \{s_3, s_4, c_2\}$, the set $U = \{c_4\}$ is (K_i, K_j)-critical, demonstrating that S' is optional.

The characterization of Theorem 4 leads to an efficient algorithm for determining whether a solution $\mathcal{T}_{alg}$ produced by Solve_IDP is general.

Theorem 5. *There is an $O(nm + |E_1|d)$-time algorithm to determine if a given solution $\mathcal{T}_{alg}$ is general, where d is the maximum out-degree in $\mathcal{T}_{alg}$.*

Proof. The algorithm simply traverses $\mathcal{T}_{alg}$ bottom-up, searching for optional clades. For each internal node x visited, whose children are $y_1, \ldots, y_{d(x)}$, the algorithm checks whether any of the clades $L(y_1), \ldots, L(y_{d(x)})$ is optional. If an optional clade is found the algorithm outputs *False*. Correctness follows from Theorem 4.

For analyzing the complexity, it suffices to show how to check whether a clade $L(y_i)$ is optional. If $d(x) = 2$, or y_i is a leaf, then certainly $L(y_i)$ is supported.

Otherwise, let U_i be the set of characters whose associated clade (in $\mathcal{T}_{alg}$) is $L(y_i)$. Let U_j^i denote the set of characters in U_i which are $L(y_j)$-semi-universal, for $j \neq i$. The computation of U_j^i for all i and j takes in total $O(nm)$ time, since for each character c and species s we check at most once whether $(s, c) \in E_?^{\mathcal{A}}$, for an input instance $\mathcal{A}$.

It remains to show how to efficiently check whether for some j, U_j^i disconnects $L(y_i)$ in the appropriate subgraph encountered during the execution of Solve_IDP. To this end, we define an auxiliary bipartite graph H^i whose set of vertices is $W_i \cup U_i$, where $W_i = \{w_1, \ldots, w_{d(y_i)}\}$ is the set of children of y_i in $\mathcal{T}_{alg}$. We include the edge (w_r, c_p) in H^i, for $w_r \in W_i, c_p \in U_i$, if $(c_p, s) \in E_1^{\mathcal{A}}$ for some species $s \in L(w_r)$. We construct for each $j \neq i$ a subgraph H_j^i of H^i induced on $W_i \cup (U_i \setminus U_j^i)$. All we need to report is whether H_j^i is connected. It can be shown that the overall complexity of the algorithm is $O(mn + |E_1^{\mathcal{A}}| \cdot \max_{v \in \mathcal{T}_{alg}} d(v))$. □

Acknowledgments

The first author was supported by the Clore foundation scholarship. The second author was supported in part by the Israel Science Foundation (grant number 565/99). The third author was supported by an Eshkol fellowship from the Ministry of Science, Israel.

References

1. 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.
2. Dan Gusfield. *Algorithms on Strings, Trees, and Sequences.* Cambridge University Press, 1997.
3. 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.
4. C. A. Meecham and G. F. Estabrook. Compatibility methods in systematics. *Annual Review of Ecology and Systematics*, 16:431–446, 1985.
5. 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.
6. I. Pe'er, R. Shamir, and R. Sharan. Incomplete directed perfect phylogeny. In *Eleventh Annual Symposium on Combinatorial Pattern Matching (CPM'00)*, pages 143–153, 2000.