Finding Most Compatible Phylogenetic Trees over Multi-State Characters

Tuukka Korhonen, Matti Järvisalo
HIIT, Department of Computer Science, University of Helsinki, Finland

Abstract

The reconstruction of the evolutionary tree of a set of species based on qualitative attributes is a central problem in phylogenetics. In the NP-hard *perfect phylogeny* problem the input is a set of taxa (species) and characters (attributes) on them, and the task is to find an evolutionary tree that describes the evolution of the taxa so that each character state evolves only once. However, in practical situations a perfect phylogeny rarely exists, motivating the *maximum compatibility* problem of finding the largest subset of characters admitting a perfect phylogeny. Various declarative approaches, based on applying integer programming (IP), answer set programming (ASP) and pseudo-Boolean optimization (PBO) solvers, have been proposed for maximum compatibility. In this work we develop a new hybrid approach to solving maximum compatibility for multi-state characters, making use of both declarative optimization techniques (specifically maximum satisfiability, MaxSAT) and an adaptation of the Bouchitté-Todinca approach to triangulation-based graph optimization problems. Empirically our approach outperforms in scalability the earlier proposed approaches w.r.t. various parameters underlying the problem.

1 Introduction

Phylogenetics concerns the evolutionary history and relationships between individual or groups of individuals, e.g., species or populations in biology. Reconstruction of an evolutionary tree of a set of species based on qualitative attributes is a central problem in phylogenetics (Semple and Steel 2003; Bordewich, Huber, and Semple 2005), and is in many settings NP-hard (Steel 1992). In the *perfect phylogeny* problem the input is a set of taxa (species) and characters (attributes) on them. Each character is a partition of a subset of the taxa into character states, describing the properties of the taxa. The task is then to find an evolutionary tree that describes the evolution of the taxa so that each character state evolves only once. This problem has received a lot of attention in theoretical algorithmics (Bodlaender, Fellows, and Warnow 1992; Steel 1992).

However, in practical situations a perfect phylogeny rarely exists. This gives rise to *maximum compatibility*, the problem of finding the largest subset of characters admitting a perfect phylogeny. Maximum compatibility has found various applications in analyzing real-world datasets (Ringe, Warnow, and Taylor 2002; Brooks et al. 2007). However, developing efficient algorithms for the problem is challenging; in addition to NP-hardness, maximum compatibility cannot be approximated well, shown via a reduction from the inapproximable maximum clique problem (Day and Sankoff 1986; Arora et al. 1992).

Nevertheless, the quest for practical exact algorithms for maximum compatibility has received noticeable attention: various declarative approaches have been proposed, including an encoding as answer set programming (ASP) (Brooks et al. 2007); integer programming (IP) formulations based on minimal separators (Gusfield 2010; Gysel and Gusfield 2011) and by reducing the general problem to a specific special case (Gusfield, Frid, and Brown 2007; Stevens and Gusfield 2010) and a pseudo-Boolean optimization (PBO) approach (Miranda, Lynce, and Manquinho 2014). Beyond declarative approaches, the algorithmic BT framework of Bouchitté and Todinca (2001) which yields efficient exponential-time algorithms for various triangulation-based graph optimization problems (Bodlaender and Fomin 2005; Furuse and Yamazaki 2014; Fomin, Todinca, and Villanger 2015), can be adapted to solving maximum compatibility (Gysel 2014). However, this comes with the restriction that the number of states allowed in the characters—describing properties of taxa—is only two. In contrast, in applications to biology the number of states of characters is often naturally higher—for example, four for DNA and 20 for amino acid sequences (Wiley and Lieberman 2011).

In this work, we propose a new exact approach to maximum compatibility. We make use of the BT framework as a basis of a novel declarative view to the problem. Whereas BT is based on computing the so-called potential maximal cliques (PMCs) of the input graph and then performing recursive computation in the style of dynamic programming over the PMCs, we harness the declarative optimization paradigm of maximum satisfiability (MaxSAT) for the second part. We model as MaxSAT in one shot the recursive computations of BT (as hard clauses) and the task of optimally resolving a triangulation-related issue that obstructs extending the original BT algorithm to multi-state characters. We also give formal evidence for the fact that the restriction of BT to two-state characters in maximum compatibility is in itself a fundamental barrier, further motivating our hybrid approach. Our approach is also directly applicable to cases where characters are weighted as well that cases where parts of the input data is missing. We show that an implementation of the resulting hybrid approach outperforms in scalability the earlier proposed approaches in terms of various parameters underlying the maximum compatibility problem.

2 Preliminaries

We start with necessary preliminaries on graph-related concepts and compatibility of phylogenetic characters. In particular, our approach to the problem builds on Theorem 1.

2.1 Graph Theory

Let G be an undirected simple graph. The set of vertices and edges of G are $V(G)$ and $E(G)$, respectively. A complete graph has an edge between every pair of its vertices. We denote the edges of a complete graph having S as vertices by S^2. If S is a subset of vertices of G, $G[S]$ is an induced subgraph of G defined by $V(G[S]) = S$, $E(G[S]) = E(G) \cap S^2$. If $G[S]$ is complete, S is a clique of G. The neighbours of vertex v are $N(v) = \{u \mid (v, u) \in E(G)\}$ and the neighbourhood of a vertex set S is $N(S) = \cup_{v \in S} N(v) \setminus S$. For a vertex set S, let $G \setminus S = G[V(G) \setminus S]$. A subset of vertices of G is a connected component if there is a path between each pair of its vertices and it is subset-maximal. We denote the connected components of G by $\mathfrak{C}(G)$. S is a *minimal separator* of G if there are two distinct components $C_1, C_2 \in \mathfrak{C}(G \setminus S)$ such that $N(C_1) = N(C_2) = S$.

A graph is chordal if every cycle of length at least 4 has a chord, an edge that joins two non-adjacent vertices. Graph H is a *triangulation* of G if it is chordal, $V(G) = V(H)$ and $E(G) \subset E(H)$. The edges in $E(H) \setminus E(G)$ are called *fill edges*. A *minimal triangulation* is a triangulation such that no subset of its edges form another triangulation. We denote the set of minimal triangulations of G by $\mathrm{MT}(G)$.

Example 1. *Consider the graph G in Figure 1 (left). Let $S = \{b, e\}$. The graph $G \setminus S$ is in Figure 1 (middle). S is a minimal separator of G as the connected components of $G \setminus S$ are $\mathfrak{C}(G \setminus S) = \{\{d\}, \{a, c, f\}\}$ and $N(\{d\}) = N(\{a, c, f\}) = \{b, e\}$. The other minimal separators of G are $\{b, c\}$, $\{c, e\}$ and $\{c, d\}$. G is not chordal since it has chordless cycle $\{b, c, e, d\}$. The graph H in Figure 1 (right) is a minimal triangulation of G. The fill edges of H are $E(H) \setminus E(G) = \{(b, e)\}$.*

2.2 Character Compatibility

In phylogenetics *characters* describe properties of taxa (singular: taxon). A character on a set of taxa X is a function $\mathcal{X} : X' \to C$, where X' is a non-empty subset of X and C is the state set of $\mathcal{X}$. If $X' = X$, then $\mathcal{X}$ is a full character. If $|C| = r$, then $\mathcal{X}$ is an r-state character. Therefore an r-state character $\mathcal{X}$ partitions X' into r parts. We denote this partition with $\pi(\mathcal{X}) = \{\mathcal{X}^{-1}\{\alpha\} \mid \alpha \in C\}$.

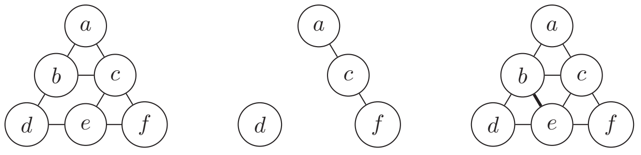


Figure 1: An example graph (left), one of its subgraphs (middle), and one of its minimal triangulations (right).

For taxa X, an X-tree is a pair $\mathcal{T} = (T, \phi)$, where T is a tree and ϕ is a function $\phi : X \to V(T)$ such that $v \in \phi(X)$ for each node $v \in V(T)$ with degree at most two. For a node set $S \subset V(T)$, let T_S be the minimal connected subtree of T containing S. An X-tree displays a character $\mathcal{X}$ if no two of the trees in $\{T_{\phi(A)} \mid A \in \pi(\mathcal{X})\}$ intersect each other.

Definition 1 (Semple and Steel 2003)**.** *A set of characters $\mathcal{C}$ on taxa X is compatible if there is an X-tree displaying $\mathcal{C}$.*

The perfect phylogeny problem is to determine if a set of characters is compatible and the **maximum compatibility problem** is to find a maximum size subset of characters that is compatible.

Example 2. *Consider a set of three-state characters $\mathcal{C} = \{A, B, C\}$ on a set of taxa $X = \{X_1, \ldots, X_5\}$ in Figure 2 (left). A is not a full character since it does not map X_2 to any state. B and C are full characters. $\mathcal{C}$ is compatible as the X-tree in Figure 2 (middle) displays all of its characters. The nodes with degree at most 2 are labeled with taxa in X to describe ϕ. The internal nodes of degree 3 are labeled with hypothetical taxa* [1 1 2] *and* [1 2 2] *which do not appear in X. Such a labeling is always possible in a way that preserves the displayed characters.*

Characters that are k-state for $k > 2$ are multi-state characters. Maximum compatibility instances in which not all characters are full are called instances with *missing data*.

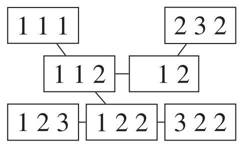
We will apply a characterization of compatibility based on triangulations of the *partition intersection graph* of the characters. This allows us to apply the BT approach for minimal triangulations to the maximum compatibility problem.

Definition 2 (Buneman 1974)**.** *Let $\mathcal{C}$ be a set of characters. The partition intersection graph (PI-graph) int($\mathcal{C}$) of $\mathcal{C}$ has*

$$V(\mathit{int}(\mathcal{C})) = \bigcup_{\mathcal{X} \in \mathcal{C}} \{(\mathcal{X}, A) \mid A \in \pi(\mathcal{X})\},$$

$$E(\mathit{int}(\mathcal{C})) = \{((\mathcal{X}, A), (\mathcal{X}', B)) \mid A \cap B \neq \emptyset\}.$$

	A	B	C
X_1	1	1	1
X_2		1	2
X_3	3	2	2
X_4	1	2	3
X_5	2	3	2



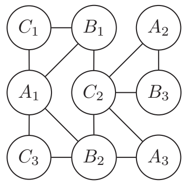


Figure 2: Three characters on five taxa. Table describing character states (left), X-tree that displays the characters (middle) and the PI-graph of the characters (right).

In words, each vertex of the PI-graph corresponds to a character state. Two vertices are connected if the corresponding character states have a common taxon, i.e., the correspondings parts of the partitions intersect. The PI-graph is used to characterize compatible sets of characters as follows. Let $((\mathcal{X}, A), (\mathcal{X}', B)) \in V(\text{int}(\mathcal{C}))^2$ be an edge that is added to the PI-graph. The *broken characters* of the edge are

$$BC(((\mathcal{X}, A), (\mathcal{X}', B))) = \begin{cases} \{\mathcal{X}\} & \text{if } \mathcal{X} = \mathcal{X}' \\ \emptyset & \text{otherwise} \end{cases} .$$

The broken characters of a triangulation H of int$(\mathcal{C})$ are $BC(H) = \bigcup_{e \in E(H)} BC(e)$.

Theorem 1 (Bordewich, Huber, and Semple 2005)**.** *Let int$(\mathcal{C})$ be a PI-graph of $\mathcal{C}$. A subset of characters $\mathcal{C}' \subset \mathcal{C}$ is compatible iff there is a triangulation H of int$(\mathcal{C})$ with broken characters $BC(H) \subset \mathcal{C} \setminus \mathcal{C}'$.*

Therefore maximum compatibility problem reduces to finding a triangulation H of int$(\mathcal{C})$ with minimum cardinality of $BC(H)$.

Example 3. *Consider characters $\mathcal{C}$ on taxa X as shown in Figure 2 (left). Figure 2 (right) is the PI-graph int$(\mathcal{C})$ of $\mathcal{C}$. The vertices are labeled with the corresponding characters and the states are denoted in subscripts. The PI-graph is not chordal because it contains the chordless cycle $\{A_1, B_1, C_2, B_2\}$. The PI-graph has two minimal triangulations $\mathrm{MT}(int(\mathcal{C})) = \{H_1, H_2\}$, each containing only one fill edge. Let H_1 be the triangulation with (A_1, C_2) as the fill edge and H_2 the triangulation with (B_1, B_2) as the fill edge. $BC(H_1) = \{\}$, and hence $\mathcal{C}$ is compatible. On the other hand, $BC(H_2) = \{B\}$.*

3 The Bouchitté-Todinca Approach

Our work builds on an instantiation of the Bouchitté-Todinca approach to the maximum compatibility problem. BT (Bouchitté and Todinca 2001; Fomin et al. 2008) was proposed originally for the treewidth and minimum fill-in problems. Later the algorithmic approach of BT has been extended to cover multiple other problems that are formulated as finding an optimal minimal triangulation w.r.t some cost function (Bodlaender and Fomin 2005; Furuse and Yamazaki 2014; Fomin, Todinca, and Villanger 2015). BT has also been extended—restricted to two-state characters—to the maximum compatibility problem via a reduction to the weighted minimum fill-in problem (Gysel 2014). We now first give a generic overview of the BT approach and then explain its instantiation to maximum compatibility over two-state characters. These details build ground for our BT-oriented MaxSAT approach to the general maximum compatibility problem over multi-state characters (Section 4).

3.1 A General View on BT

The BT approach characterizes minimal triangulations of a graph via recursion on the *blocks* of the graph. A pair (S, C) is a block of G if S is a minimal separator and C is a connected component of $G \setminus S$ such that $N(C) = S$. The realization of a block, $R(S, C)$, is the subgraph induced by $S \cup C$, where the separator is filled into a clique. It has

$$V(R(S, C)) = S \cup C,$$

$$E(R(S, C)) = E(G[S \cup C]) \cup S^2.$$

The BT approach defines the set of minimal triangulations of each realization of a block based on the set of minimal triangulations of smaller realizations of blocks. In dynamic programming algorithms that utilize the BT approach, the dynamic programming scheme computes an optimal minimal triangulation for each realization of a block.

In order to simplify representation, we define that $(\{\}, V(G))$ is also a block of G. We do this to represent the graph G itself as a realization of a block, $R(\{\}, V(G)) = G$.

The BT approach defines the transitions between blocks with potential maximal cliques (PMCs). A set of vertices Ω is a PMC of G if it is a maximal clique in some minimal triangulation of G. We denote the set of PMCs of a graph G with $\Pi(G)$. The minimal triangulations of a realization $R(S, C)$ are characterized via PMCs of G that are contained in $S \cup C$ and contain the minimal separator S. We call those PMCs the *relevant PMCs* of (S, C) and denote them by

$$\mathcal{R}(S, C) = \{\Omega \in \Pi(G) \mid S \subset \Omega \subset S \cup C\}.$$

The *associated blocks* of PMC Ω in component C are

$$\mathcal{A}(C, \Omega) = \{(S_i, C_i) \mid C_i \in \mathfrak{C}(G[C \setminus \Omega]), S_i = N(C_i)\}.$$

Theorem 2 formalizes the recursion that characterizes the minimal triangulations of G.

Theorem 2 (Bouchitté and Todinca 2001)**.** $H \in \mathrm{MT}(R(S, C))$ *if and only if* $\Omega \in \mathcal{R}(S, C)$ *and*

$$V(H) = S \cup C,$$

$$E(H) = \Omega^2 \cup \bigcup_{(S_i, C_i) \in \mathcal{A}(C, \Omega)} E(H_i),$$

where $H_i \in \mathrm{MT}(R(S_i, C_i))$.

Intuitively, for each block there is a choice of which of the relevant PMCs to fill into a clique. After filling the PMC into a clique, the computation is divided into the associated blocks in smaller components of the graph. The root of the recursion is given by the block $R(\{\}, V(G)) = G$.

Example 4. *Consider the graph G in Figure 3 (left). Let $S = \{b, c\}$ and $C = \{d, e, f\}$. The pair (S, C) is a block of G. There relevant PMCs of the block are $\mathcal{R}(S, C) = \{\{b, c, e\}, \{b, c, d\}\}$. The realizations of the two blocks associated to PMC $\{b, c, e\}$ and of the single block associated to PMC $\{b, c, d\}$ are shown in Figure 3 (middle) and (right), resp., with their minimal separators in gray.*

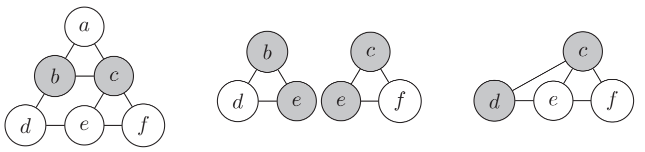


Figure 3: A graph G (left) and realizations $R(\{b, e\}, \{d\})$, $R(\{c, e\}, \{f\})$ (middle) and $R(\{c, d\}, \{e, f\})$ (right). Gray vertices correspond to minimal separators of G.

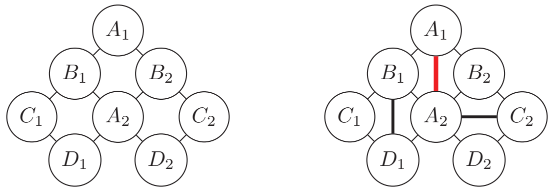


Figure 4: A PI-graph of a set of two-state characters (left) and a minimum-weight triangulation of the PI-graph (right).

Instantiations of the BT approach enumerate the potential maximal cliques of G and then adapt the recursion of Theorem 2 to find an optimal minimal triangulation. The blocks and relations $\mathcal{R}$ and $\mathcal{A}$ connecting them to PMCs are computed in time $O(|\Pi(G)|n^3)$. Best known time complexity for enumerating PMCs is $O(1.7347^n)$ (Fomin, Todinca, and Villanger 2015), but practical algorithms are observed to often be much faster (Korhonen, Berg, and Järvisalo 2019).

3.2 BT for Two-State Maximum Compatibility

Towards our hybrid BT-oriented MaxSAT approach to maximum compatibility, Gysel (2014) provides a reduction of the maximum compatibility problem over two-state characters to weighted minimum fill-in problem which can be solved directly via BT.

The idea of the reduction is that in the two-state case, for each character $\mathcal{X} \in \mathcal{C}$ there is only one edge $e \in V(\text{int}(\mathcal{C}))^2$ that has $BC(e) = \{\mathcal{X}\}$, i.e., that breaks the character $\mathcal{X}$. Therefore, in order to count the number of broken edges $|BC(H)|$ of triangulation H, it is sufficient to count the number of fill-edges $e \in E(H)$ that have $|BC(e)| = 1$.

The *minimum fill-in problem* is to find a triangulation H of a graph G which minimizes $|E(H) \setminus E(G)|$, the number of fill-edges. In the weighted version of the problem, each possible fill edge is assigned a non-negative weight by a weight function $w : V(G)^2 \to \mathbb{R}_{\geq 0}$, and the goal is to minimize the sum of the weights of the added edges, $w(H) = \sum_{e \in E(H) \setminus E(G)} w(e)$. Weighted minimum fill-in can be solved in $O(|\Pi(G)|n^3)$ time after listing the PMCs (Gysel 2014; Furuse and Yamazaki 2014).

Consider the weight function $w(e) = |BC(e)|$.

Theorem 3 (Gysel 2014)**.** *Let $\mathcal{C}$ be a set of two-state characters. A maximum compatible subset of characters $\mathcal{C}' \subset \mathcal{C}$ has size $|\mathcal{C}'|$ iff a minimum-weight triangulation H of int$(\mathcal{C})$ has weight $w(H) = |\mathcal{C}| - |\mathcal{C}'|$.*

Example 5. *Consider a set of two-state characters $\mathcal{C} = \{A, B, C, D\}$ with a PI-graph int$(\mathcal{C})$ in Figure 4 (left). The edge weight function w of Theorem 3 assigns weight 1 to fill edges (A_1, A_2), (B_1, B_2), (C_1, C_2) and (D_1, D_2). It assigns weight 0 to other edges. Figure 4 (right) displays a minimum-weight triangulation H of int$(\mathcal{C})$. It has fill edges (A_1, A_2), (B_1, D_1) and (A_2, C_2), so its weight is $w((A_1, A_2)) + w((B_1, D_1)) + w((A_2, C_2)) = 1 + 0 + 0 = 1$. Following Theorem 3, a maximum compatible subset of $\mathcal{C}$ has size 3. A maximum compatible subset is given by $\mathcal{C} \setminus BC(H) = \{A, B, C, D\} \setminus \{A\} = \{B, C, D\}$.*

However, the reduction of Gysel (2014) is not directly applicable to maximum compatibility over multi-state characters since broken characters may be counted multiple times. This turns out to be a fundamental limitation for BT.

4 BT / MaxSAT Approach to Maximum Compatibility over Multi-State Characters

We will provide evidence on why BT (as overviewed in Section 3) is not applicable for solving the maximum compatibility problem over characters with more than two states. This is a critical issue in many scenarios, as biologically interesting data most often has more character states than two—for example, DNA sequences are over four states. To circumvent this issue, we will introduce a novel approach to the maximum compatibility problem that uses the BT recursion to formulate the problem as MaxSAT.

4.1 Non-Generalizability of BT to Multi-State

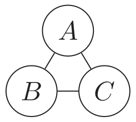
Theorem 3 does not directly generalize to characters with more than two states because there can be multiple possible fill edges within each character. Our following theorem provides strong corroboration for the fact that BT (as discussed so far) is not applicable to solving the maximum compatibility problem over multi-state characters in general.

Theorem 4. *The vertex cover problem has a polynomial reduction to maximum compatibility of $\mathcal{C}$ where each component of int$(\mathcal{C})$ has 4 vertices.*

Proof. Let a graph G be an instance of vertex cover problem. We create set of characters $\mathcal{C} = \{\mathcal{X}_v \mid v \in V(G)\}$ such that each character corresponds to a vertex of the graph. For each edge $(a, b) \in E(G)$ we create a 4-cycle in the PI-graph int$(\mathcal{C})$ by adding two new states for both $\mathcal{X}_a$ and $\mathcal{X}_b$ and four taxa to connect these character states. A minimal triangulation of this 4-cycle breaks either $\mathcal{X}_a$ or $\mathcal{X}_b$, representing how either a or b should be chosen to the vertex cover. $\square$

Each potential maximal clique is contained in exactly one connected component so Theorem 4 implies that vertex cover can be polynomially reduced into maximum compatibility of $\mathcal{C}$, where int$(\mathcal{C})$ has a linear number of potential maximal cliques. In fact, the PI-graph created in the proof of Theorem 4 has exactly $4m$ potential maximal cliques, where m is the number of edges of the vertex cover instance. Current adaptations of BT approach work in polynomial time w.r.t the number of potential maximal cliques and the size of the graph. By Theorem 4 and NP-hardness of vertex cover, this kind of adaptation is not possible for maximum compatibility assuming P $\neq$ NP. We conclude that, in order to solve multi-state maximum compatibility with BT approach, a different approach needs to be taken for solving NP-hard problems on top of BT.

Example 6. *Consider the vertex cover instance in Figure 5 (left). The reduction of Theorem 4 states that the maximum compatibility instance on characters $\mathcal{C} = \{A, B, C\}$ in the two tables in Figure 5 (middle) has an equivalent optimal solution to it. The PI-graph int$(\mathcal{C})$ of $\mathcal{C}$ in Figure 5 (right)*



A	B	C
1	1	
1	2	
2	1	
2	2	
	3	1
	3	2
	4	1
	4	2
3		3
3		4
4		3
4		4

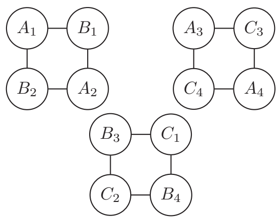


Figure 5: An instance of vertex cover (left), a corresponding maximum compatibility instance (middle) and the PI-graph of the maximum compatibility instance (right).

consists of connected components of size 4. From the PI-graph we see that broken characters cannot be counted in a local manner, which is typical for BT, as the character A could be broken by the edge (A_1, A_2) or (A_3, A_4) or both, which are in different connected components.

4.2 An BT-MaxSAT Hybrid for Maximum Compatibility over Multi-State Characters

To overcome the limitation implied by Theorem 4, we propose an alternative way of making use of BT in the general setting of multi-state characters. Whereas the original BT approach is based on computing the potential maximal cliques (PMCs) of the input graph and then performing recursive computation in the style of dynamic programming over the PMCs, we harness the declarative optimization paradigm of maximum satisfiability (MaxSAT) for the second part.

A partial MaxSAT instance (F_S, F_H) consists of two sets of clauses, soft clauses F_S and the hard clauses F_H, over Boolean variables. An optimal solution is an assignment of variables which satisfies all clauses in F_H and the maximum number of clauses in F_S.

The main issue in applying the BT algorithm to multi-state maximum compatibility is that, in the second phase of the algorithm (i.e., after enumerating the PMCs) the broken characters cannot be counted "locally"; we can add a fill edge that breaks a character in one connected component, and another fill edge that breaks the same character in a different connected component. Instead, one needs to reason in a globally optimal way about the number of simultaneously compatible characters. To address this, we model as MaxSAT in one shot the recursive computations of BT (as hard clauses) and the task of finding a globally optimal way of minimizing broken characters (as soft clauses).

The input to the MaxSAT encoding is a PI-graph of an instance of the maximum compatibility problem and the PMCs of the PI-graph underlying the instance. The blocks and their relations to PMCs ($\mathcal{R}$ and $\mathcal{A}$) are straightforward to compute given the PMCs. For each character $\mathcal{X}_i \in \mathcal{C}$, we use a variable X_i to represent inclusion of $\mathcal{X}_i$ to the subset of compatible characters. The assignment $X_i = 1$ corresponds to including the character to the subset. The soft clauses of the encoding are $F_S = \{(X_i) \mid \mathcal{X}_i \in \mathcal{C}\}$, so an optimal solution corresponds to the maximum number of compatible characters.

Next we describe the hard clauses F_H. The hard clauses ensure that there is a triangulation for which the broken characters are the characters $\mathcal{X}_i$ for which X_i is assigned to 0. To this end, for each PMC $\Omega \in \Pi(G)$ we use a variable P_Ω to denote the PMCs that are filled into cliques. We add constraints as hard clauses to ensure that characters broken by the fill edges of the PMCs are broken by the X_i variables

$$P_\Omega \rightarrow \bigwedge_{\mathcal{X}_i \in BC(\Omega^2)} \neg X_i.$$

For each block (S, C) we add a variable $B_{S,C}$ denoting if there is a triangulation of the realization of the block such that its broken characters are a subset of the characters indicated by $X_i = 0$. To express the BT recursion, we additionally add variable $B_{S,C,\Omega}$ for each relevant PMC Ω of the block (S, C) to indicate which of the PMCs are filled in the BT recursion, and encode the recursion via

$$B_{S,C} \rightarrow \bigvee_{\Omega \in \mathcal{R}(S,C)} B_{S,C,\Omega},$$

$$B_{S,C,\Omega} \rightarrow \bigwedge_{(S_i,C_i) \in \mathcal{A}(C,\Omega)} B_{S_i,C_i}$$

$$B_{S,C,\Omega} \rightarrow P_\Omega.$$

Finally, the base case is enforced by setting $B_{\{\},V(G)} = 1$.

Often, like in Example 6, the PI-graph consists of multiple connected components. In such cases, the hard clauses for each connected component are generated independently.

Proposition 1. *Given a PI-graph of any instance of the maximum compatibility problem (possibly over multi-state characters) and its PMCs, any optimal solution to the above MaxSAT encoding corresponds to an optimal solution of the maximum compatibility instance. Furthermore, the underlying optimal X-tree can be obtained in linear time from the optimal assignment of the variables.*

Finally, as a side-note, we observe that the MaxSAT encoding produces instances that are essentially Horn-MaxSAT (i.e., each clause is Horn modulo negating all literals except the X_i). While Horn-MaxSAT remains NP-hard, as observed in (Marques-Silva, Ignatiev, and Morgado 2017) Horn-MaxSAT can at least in theory be easier to solve for certain types of current state-of-the-art MaxSAT solvers.

4.3 Preprocessing

In addition to the MaxSAT approach, we also consider multiple polynomial-time preprocessing steps to simplify the instance and its PI-graph before the PMC enumeration phase of the approach.

1. Remove character states that correspond to only one taxon.
2. Remove characters with less than two distinct states – they are always compatible with other characters.
3. Divide the PI-graph into *atoms* – induced subgraphs that are separated by clique separators. Each atom will be considered as a separate connected component.
4. For each atom, compute a minimal triangulation. If the minimal triangulation does not break any characters, remove the atom.

Techniques 1–2 are used before computing the PI-graph of the instance, and techniques 3–4 are for simplifying the PI-graph. Technique 1 is correct due to the equivalence of missing data and states that correspond to only one taxon (Semple and Steel 2003). Technique 2 is the trivial character technique from (Stevens and Gusfield 2010). Technique 3 is correct since a minimal triangulation will never contain a fill-edge between two different atoms (Tarjan 1985). Technique 4 is based on the fact that triangulations of each atom can be considered independently in the case where they do not break any characters. To this end, we compute minimal triangulations with the maximum cardinality search algorithm (Berry et al. 2004). In addition to Technique 4, minimal triangulations are used in Technique 3 for computing the atoms (Tarjan 1985). We note that these 4 techniques cover the removal of simplicial vertices of the PI-graph proposed by Gysel and Gusfield (2011). In particular, splitting the graph into atoms (Technique 3) and removing the atoms that are already chordal (Technique 4) removes all simplicial vertices.

These preprocessing techniques can also be lifted for the approaches that do not consider the PI-graph (including ones we compare to in our experiments) by first employing all of the techniques on the instance, and then taking the subset of characters that occur in the final preprocessed PI-graph. The rest of the characters will always be compatible.

5 Experiments

We empirically evaluate the performance of our hybrid BT-MaxSAT approach. Our implementation, all test data and detailed results are available via https://github.com/Laakeri/phylogeny-aaai. We compare the performance of our implementation with the performance of earlier proposed approaches to maximum compatibility over multi-state characters, each of which is based on employing declarative optimization solvers.

PBO (Miranda, Lynce, and Manquinho 2014): Pseudo-Boolean approach based on encoding the structure of phylogenetic trees.

Minsep IP (Gusfield 2010; Gysel and Gusfield 2011): Integer programming formulation based on minimal separators.

Bin IP (Gusfield, Frid, and Brown 2007; Stevens and Gusfield 2010): Integer programming formulation based on reducing the problem to the two-state case.

The PBO approach was earlier shown (Miranda, Lynce, and Manquinho 2014) to be superior to an earlier declarative approach based on answer set programming (Brooks et al. 2007). For the PBO approach, we used the code provided by the authors, in particular the PBO-RR variant which includes additional constraints to strengthen the PBO formulation, which we observed to be the best-performing variant in preliminary experiments. For the experiments on PBO, we used Minisat+ (Eén and Sorensson 2006) as the PBO solver.

We re-implemented both of the IP-based approaches ourselves, as we were unable to obtain the original implementations. For more information on these approaches, the minimal separator IP (Minsep IP) approach characterizes triangulations of the PI-graph with minimal separators: A triangulation is obtained by completing a maximal set of non-crossing separators into cliques. The approach first enumerates the (exponential number of) minimal separators of the PI-graph, and encodes as an IP the maximality and non-crossing constraints, creating $\#\text{minseps}^2$ constraints in the worst case. The Bin IP approach employs first a reduction from multi-state to two-state perfect phylogeny with missing data (Stevens and Gusfield 2010), extended to maximum compatibility with additional constraints, and then uses an IP formulation of Gusfield, Frid, and Brown (2007) on the level of the character-state matrix. For both of the IP-based approaches, we used CPLEX 12.7.1 (IBM ILOG 2017) as the IP solver in the experiments.

We implemented our BT-MaxSAT approach based on the Triangulator implementation of BT (Korhonen, Berg, and Järvisalo 2019), and used MaxHS (Davies and Bacchus 2013) as the MaxSAT solver. The running times reported for our approach include both the PMC enumeration and the MaxSAT solving phases.

We used the preprocessing techniques as described in Section 4.3 with all of the approaches. For Minsep IP, the preprocessing on the PI-graph level was directly implemented, and for PBO and Bin IP it was lifted as described in Section 4.3. The experiments were run single-threaded on computers with 2.4-GHz Intel Xeon E5-2680-v4 processors with a per-instance 2-hour time limit and 32-GB memory limit.

Benchmarks. We generated a comprehensive set of benchmarks using the *ms* data generator (Hudson 2002) employed in various empirical evaluations of algorithms for maximum compatibility and perfect phylogeny (Gusfield, Frid, and Brown 2007; Gusfield 2010; Stevens and Gusfield 2010; Gysel and Gusfield 2011; Gysel, Gusfield, and Stevens 2013; Coulombe, Stevens, and Gusfield 2015) extended to multi-state instances as described in (Gusfield 2010). The generator generates instances based on four parameters: n, the number of taxa; m, the number of characters; k, the number of states for each character; and r, the recombination parameter that controls how far the data is from admitting a perfect phylogeny. Following Gusfield (2010) we set $n = m$. The value nm reflects the size of an instance. For the parameter k, the values 4 and 20 are biologically relevant (nucleotide and amino acid) (Wiley and Lieberman 2011). Setting $r = 0$ guarantees a perfect phylogeny; we let r vary from 0 to 4. For instance, we found that parameters $n = m = 200$, $k = 20$, $r = 2$ produce instances where the maximum compatible subset consists of $60 - 92\%$ of the characters.

Results. We look at the results in two parts. Firstly, we tested how the runtimes varied when two of the parameters n, k, r was fixed and one varied. The fixed values were always $n = 200$, $k = 20$ and $r = 2$, and the range of variation was from 20 to 400 for n, from 2 to 40 for k and from 0 to 3.8 for r. For each set of parameters 20 random instances were generated. Figure 6 shows for each parameter combination how many instances of the 20 generated were solved by the different approaches. Our MaxSAT approach exhibits best performance overall regardless of which of the three parameters is varied, with the Minsep IP and Bin IP approaches coming in second and third, respectively.

Secondly, we generated instances with all parameter combinations from the sets $n = \{50, 100, \ldots, 400\}$, $k =$

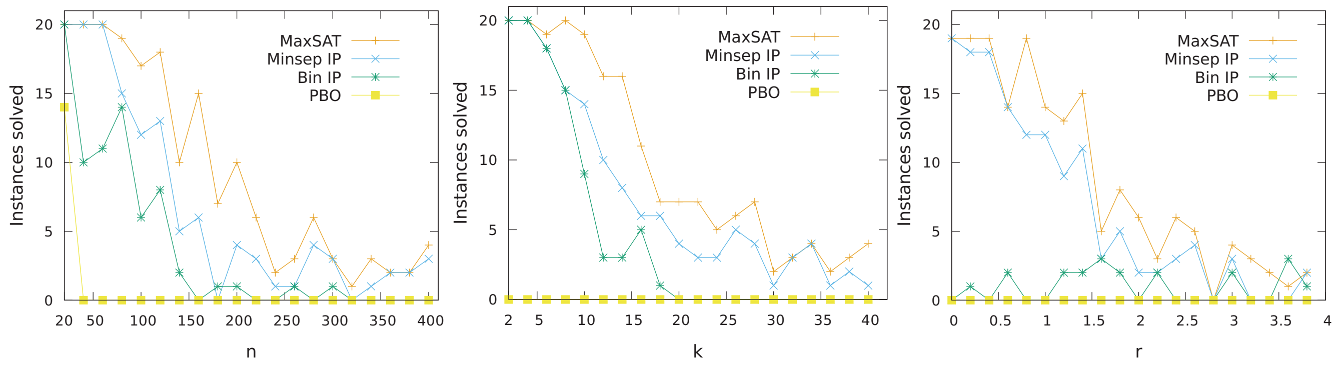


Figure 6: Number of solved instances out of 20 generated with *ms* with different parameters fixed.

$\{4, 10, 20, 40\}$ and $r = \{0, 1, 2, 4\}$. We generated 5 random instances for each of the combinations, resulting in a total of 640 instances. We compare the MaxSAT approach to the Bin IP and Minsep IP in Figure 7 and Figure 8, respectively. Our MaxSAT approach performs significantly better than Bin IP at larger values of k, with a significantly fewer timeouts, and, similarly, better than Minsep IP at larger values of r.

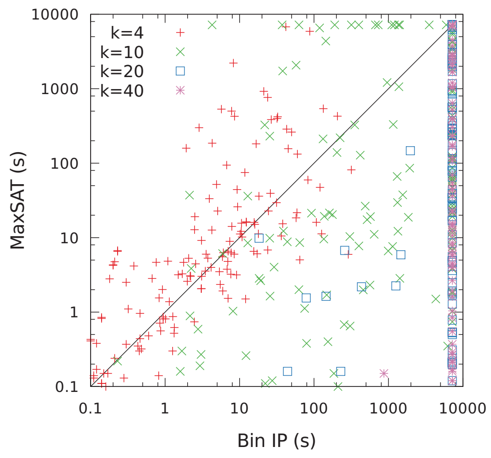


Figure 7: MaxSAT vs Bin IP for different values of k

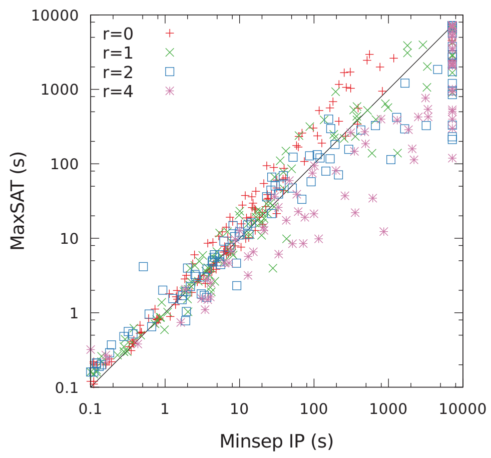


Figure 8: MaxSAT vs Minsep IP for different values of r

The MaxSAT solving phase caused a timeout on only 5 instances among all of the 1840 generated. Over solved instances, the average time spent in PMC enumeration was 669s and 51s for MaxSAT solving.

Finally, we note that we also tested the algorithms on the (unfortunately few) "real-world" benchmarks based on datasets used in (Miranda, Lynce, and Manquinho 2014) dealing with Chinese dialects (Minett and Wang 2003), Indo-European languages (Ringe, Warnow, and Taylor 2002), Mammal mitochondrial sequences (Hasegawa, Kishino, and Yano 1985) and Alcataenia (Hoberg 1992). These instances turned out to be mostly very fast to solve for all of the approaches, with running times < 5 seconds, apart from a non-preprocessed version of the Indo benchmark (on which both Minsep IP and PBO ran out of memory, MaxSAT solving the instance in hours) and Mammals (on which PBO took more than 20 seconds).

6 Conclusions

The NP-hard maximum compatibility problem finds various applications in the analysis of phylogenetic data. We proposed a novel practical approach to maximum compatibility, bridging together ideas from the BT algorithmic framework and declarative optimization, in particular maximum satisfiability. Empirically, we showed that this hybrid approach outperforms earlier proposed declarative approaches to maximum compatibility. Furthermore, we provided a rigorous argument for why the BT framework in itself cannot be generalized to cover maximum compatibility in the general setting we considered, i.e., the multi-state character setting motivated by real-world applications.

In this work we focused on generalizing the second phase of the BT approach via harnessing maximum satisfiability. Further improvements to our approach could be achieved by improving the first (PMC enumeration) phase of the hybrid approach. More generally, studying applications of the BT-with-MaxSAT approach proposed in this work to other problems that can be defined via minimal triangulations of graphs but not directly as BT and dynamic programming is another possible research direction.

Acknowledgements

This work has been financially supported by Academy of Finland (grants 276412, 312662, 322869).

References

Arora, S.; Lund, C.; Motwani, R.; Sudan, M.; and Szegedy, M. 1992. Proof verification and hardness of approximation problems. In *Proc. FOCS*, 14–23. IEEE.

Berry, A.; Blair, J. R.; Heggernes, P.; and Peyton, B. W. 2004. Maximum cardinality search for computing minimal triangulations of graphs. *Algorithmica* 39(4):287–298.

Bodlaender, H. L., and Fomin, F. V. 2005. Tree decompositions with small cost. *Discrete Applied Mathematics* 145(2):143–154.

Bodlaender, H. L.; Fellows, M. R.; and Warnow, T. J. 1992. Two strikes against perfect phylogeny. In *Proc. ICALP*, volume 623 of *LNCS*, 273–283. Springer.

Bordewich, M.; Huber, K. T.; and Semple, C. 2005. Identifying phylogenetic trees. *Discrete Mathematics* 300(1-3):30–43.

Bouchitté, V., and Todinca, I. 2001. Treewidth and minimum fill-in: Grouping the minimal separators. *SIAM Journal on Computing* 31(1):212–232.

Brooks, D. R.; Erdem, E.; Erdoğan, S. T.; Minett, J. W.; and Ringe, D. 2007. Inferring phylogenetic trees using answer set programming. *Journal of Automated Reasoning* 39(4):471.

Buneman, P. 1974. A characterisation of rigid circuit graphs. *Discrete Mathematics* 9(3):205–212.

Coulombe, M.; Stevens, K.; and Gusfield, D. 2015. Construction, enumeration, and optimization of perfect phylogenies on multi-state data. In *Proc. ICCABS*, 1–6. IEEE.

Davies, J., and Bacchus, F. 2013. Exploiting the power of MIP solvers in MaxSAT. In *Proc. SAT*, volume 7962 of *LNCS*, 166–181. Springer.

Day, W. H. E., and Sankoff, D. 1986. Computational complexity of inferring phylogenies by compatibility. *Systematic Zoology* 35(2):224–229.

Eén, N., and Sorensson, N. 2006. Translating pseudo-boolean constraints into SAT. *Journal on Satisfiability, Boolean Modeling and Computation* 2:1–26.

Fomin, F. V.; Kratsch, D.; Todinca, I.; and Villanger, Y. 2008. Exact algorithms for treewidth and minimum fill-in. *SIAM Journal on Computing* 38(3):1058–1079.

Fomin, F. V.; Todinca, I.; and Villanger, Y. 2015. Large induced subgraphs via triangulations and CMSO. *SIAM Journal on Computing* 44(1):54–87.

Furuse, M., and Yamazaki, K. 2014. A revisit of the scheme for computing treewidth and minimum fill-in. *Theoretical Computer Science* 531:66–76.

Gusfield, D.; Frid, Y.; and Brown, D. 2007. Integer programming formulations and computations solving phylogenetic and population genetic problems with missing or genotypic data. In *Proc. COCOON*, volume 4598 of *LNCS*, 51–64. Springer.

Gusfield, D. 2010. The multi-state perfect phylogeny problem with missing and removable data: solutions via integer-programming and chordal graph theory. *Journal of Computational Biology* 17(3):383–399.

Gysel, R., and Gusfield, D. 2011. Extensions and improvements to the chordal graph approach to the multistate perfect phylogeny problem. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 8(4):912–917.

Gysel, R.; Gusfield, D.; and Stevens, K. 2013. Triangulation heuristics for maximum character compatibility. In *Proc. ICCABS*, 1–2. IEEE.

Gysel, R. 2014. Minimal triangulation algorithms for perfect phylogeny problems. In *Proc. LATA*, volume 8370 of *LNCS*, 421–432. Springer.

Hasegawa, M.; Kishino, H.; and Yano, T.-a. 1985. Dating of the human-ape splitting by a molecular clock of mitochondrial dna. *Journal of Molecular Evolution* 22(2):160–174.

Hoberg, E. P. 1992. Congruent and synchronic patterns in biogeography and speciation among seabirds, pinnipeds, and cestodes. *Journal of Parasitology* 601–615.

Hudson, R. R. 2002. Generating samples under a wright–fisher neutral model of genetic variation. *Bioinformatics* 18(2):337–338.

IBM ILOG. 2017. Cplex optimizer 12.7.1.

Korhonen, T.; Berg, J.; and Järvisalo, M. 2019. Solving graph problems via potential maximal cliques: An experimental evaluation of the Bouchitté–Todinca algorithm. *Journal of Experimental Algorithmics* 24(1):1–9.

Marques-Silva, J.; Ignatiev, A.; and Morgado, A. 2017. Horn maximum satisfiability: Reductions, algorithms and applications. In *Proc. EPIA*, volume 10423 of *LNCS*, 681–694. Springer.

Minett, J. W., and Wang, W. S. 2003. On detecting borrowing: distance-based and character-based approaches. *Diachronica* 20(2):289–330.

Miranda, M.; Lynce, I.; and Manquinho, V. 2014. Inferring phylogenetic trees using pseudo-boolean optimization. *AI Communications* 27(3):229–243.

Ringe, D.; Warnow, T.; and Taylor, A. 2002. Indo-european and computational cladistics. *Transactions of the Philological Society* 100(1):59–129.

Semple, C., and Steel, M. 2003. *Phylogenetics*, volume 24. Oxford University Press.

Steel, M. 1992. The complexity of reconstructing trees from qualitative characters and subtrees. *Journal of Classification* 9(1):91–116.

Stevens, K., and Gusfield, D. 2010. Reducing multi-state to binary perfect phylogeny with applications to missing, removable, inserted, and deleted data. In *Proc. WABI*, volume 6293 of *LNCS*, 274–287. Springer.

Tarjan, R. E. 1985. Decomposition by clique separators. *Discrete Mathematics* 55(2):221–232.

Wiley, E. O., and Lieberman, B. S. 2011. *Phylogenetics: Theory and Practice of Phylogenetic Systematics*. John Wiley & Sons.