Finding Optimal Triangulations Parameterized by Edge Clique Cover

Tuukka Korhonen
Department of Computer Science, University of Helsinki, Finland
`https://tuukkakorhonen.com`
tuukka.m.korhonen@helsinki.fi

Abstract

Many graph problems can be formulated as a task of finding an optimal triangulation of a given graph with respect to some notion of optimality. In this paper we give algorithms to such problems parameterized by the size of a minimum edge clique cover ($\texttt{cc}$) of the graph. The parameter $\texttt{cc}$ is both natural and well-motivated in many problems on this setting. For example, in the perfect phylogeny problem $\texttt{cc}$ is at most the number of taxa, in fractional hypertreewidth $\texttt{cc}$ is at most the number of hyperedges, and in treewidth of Bayesian networks $\texttt{cc}$ is at most the number of non-root nodes of the Bayesian network.

Our results are based on the framework of potential maximal cliques. We show that the number of minimal separators of graphs is at most $2^{\texttt{cc}}$ and the number of potential maximal cliques is at most $3^{\texttt{cc}}$. Furthermore, these objects can be listed in times $O^*(2^{\texttt{cc}})$ and $O^*(3^{\texttt{cc}})$, respectively, even when no edge clique cover is given as input; the $O^*(\cdot)$ notation omits factors polynomial in the input size. Using these enumeration algorithms we obtain $O^*(3^{\texttt{cc}})$ time algorithms for problems in the potential maximal clique framework, including for example treewidth, minimum fill-in, and feedback vertex set. We also obtain an $O^*(3^m)$ time algorithm for fractional hypertreewidth, where m is the number of hyperedges. In the case when an edge clique cover of size $\texttt{cc}'$ is given as an input we further improve the time complexity to $O^*(2^{\texttt{cc}'})$ for treewidth, minimum fill-in, and chordal sandwich. This implies an $O^*(2^n)$ time algorithm for perfect phylogeny, where n is the number of taxa. We also give polynomial space algorithms with time complexities $O^*(9^{\texttt{cc}'})$ and $O^*(9^{\texttt{cc}+O(\log^2 \texttt{cc})})$ for problems in this framework.

2012 ACM Subject Classification Theory of computation → Parameterized complexity and exact algorithms; Theory of computation → Graph algorithms analysis

Keywords and phrases Treewidth, Minimum fill-in, Perfect phylogeny, Fractional hypertreewidth, Potential maximal cliques, Edge clique cover

Digital Object Identifier 10.4230/LIPIcs.IPEC.2020.22

Related Version A full version of the paper is available at `https://arxiv.org/abs/1912.10989`.

Funding This work has been financially supported by Academy of Finland (grant 322869).

Acknowledgements I wish to thank Matti Järvisalo, Mikko Koivisto, Andreas Niskanen, and Juha Harviainen for helpful comments.

1 Introduction

In this paper, we give algorithms to problems that can be formulated as a task of finding an optimal triangulation of a given graph with respect to some notion of optimality. A triangulation of a graph G is a chordal graph H with $E(G) \subseteq E(H)$. For example, computing the graph parameter treewidth corresponds to finding a triangulation with the minimum possible size of a maximum clique, and minimum fill-in corresponds to finding a triangulation with the least number of edges. In particular, we consider optimal triangulation problems parameterized by the size of a minimum edge clique cover of the input graph, denoted by $\texttt{cc}$. An edge clique cover of a graph is a set of cliques of the graph that covers all edges of the graph. Our algorithms are based on the framework of potential maximal cliques [5, 14].

1.1 Motivation

While in general the parameter $\texttt{cc}$ could be considered non-standard, it has natural interpretations in at least three settings on which algorithms for finding optimal triangulations are applied: hypergraph parameters, phylogenetics, and probabilistic inference. The reason that $\texttt{cc}$ is a natural choice in these settings is that the input graph is constructed as a union of cliques, with the goal of each clique W representing a constraint of type "the triangulation must contain a maximal clique Ω with $W \subseteq \Omega$". Maximal cliques of a triangulation in turn correspond to bags of a tree decomposition. Next we discuss the three settings in more detail.

The hypergraph parameter fractional hypertreewidth is a central structural parameter of constraint satisfaction problems (CSPs) [18]. The computation of fractional hypertreewidth can be formulated as a task of finding a triangulation H of the primal graph of the hypergraph, minimizing the quantity $\max_{\Omega \in \mathrm{MC}(H)} \mathrm{FCOV}(\Omega)$, where $\mathrm{MC}(H)$ denotes the set of maximal cliques of H and $\mathrm{FCOV}(\Omega)$ denotes the minimum size of a so-called fractional edge cover of the maximal clique Ω [30]. The primal graph of a hypergraph is constructed by inducing a clique on each hyperedge, and therefore the size of its minimum edge clique cover is at most m, the number of hyperedges.

In phylogenetics a central problem is to construct an evolutionary tree of a set of n taxa (i.e. species) based on k characters (i.e. attributes) describing them [34]. For example, when the character data is drawn from molecular sequences, the number of characters k can be much larger than the number of taxa n [24]. Deciding if the taxa admit a perfect phylogeny can be reduced to the chordal sandwich problem on the partition intersection graph of the characters [19, 34]. Moreover, the optimization version of perfect phylogeny, called the maximum compatibility problem of phylogenetic characters, can be reduced to the weighted minimum fill-in problem on the partition intersection graph if the characters are binary [4, 19]. The partition intersection graph has a vertex for each character-state pair, and its edges are constructed by inducing a clique corresponding to each taxon [34]. Hence the size of its minimum edge clique cover is at most n, the number of taxa.

A third setting in which parameterization by $\texttt{cc}$ is motivated is probabilistic inference. Given a Bayesian network, the first step of efficient probabilistic inference algorithms is to compute a tree decomposition of small width of the moral graph of the Bayesian network [22, 23]. The moral graph is constructed as a union of n' cliques, where n' is the number of non-root nodes of the Bayesian network [23], and thus the size of its minimum edge clique cover is at most n'.

This paper is also directly motivated by observations in practice. Starting from the Second Parameterized Algorithms and Computational Experiments challenge (PACE 2017) [10], algorithm implementations based on potential maximal cliques have been observed to outperform other exact algorithm implementations on problems formulated as finding optimal triangulations [25, 26, 27, 31, 36, 37]. In particular, this paper is motivated by experimental observations of the usefulness of potential maximal cliques in computing hypergraph parameters [25, 26] and in phylogenetics [25, 27].

Our parameterization can be justified by real-world instances with small edge clique covers. In the context of fractional hypertreewidth, 708 of the 3072 hypergraphs in the standard HyperBench library [11] have $m < n/2$, where n is the number of vertices and m is the number of hyperedges. In the context of phylogenetics, an instance describing mammal mitochondrial sequences [20] has 7 taxa while its partition intersection graph has 245 vertices and an instance describing Indo-European languages [32] has 24 taxa while its partition intersection graph has 864 vertices. In the context of Bayesian networks, the Bayesian network "Andes" [8], accessed from the standard BNlearn repository [33], has 223 nodes of which 134 are non-root.

1.2 Techniques

The algorithms that we design in this paper are based on the framework of potential maximal cliques (PMCs) [5, 14]. Algorithms in this framework typically consist of two phases. In the first phase the set $\Pi(G)$ of PMCs of the input graph G is enumerated, and in the second phase dynamic programming over the PMCs is performed in time $O^*(|\Pi(G)|)$. The second phase of the PMC framework has already been formulated for all of the problems that we consider [14, 16, 17, 19, 30], so our $O^*(3^{\mathsf{cc}})$ time algorithms follow from an $O^*(3^{\mathsf{cc}})$ time PMC enumeration algorithm that we give. This algorithm is based on the Bouchitté–Todinca algorithm [6]. We achieve the $O^*(3^{\mathsf{cc}})$ bound by novel characterizations of minimal separators and PMCs with respect to an edge clique cover. In particular, we show that minimal separators correspond to bipartitions of an edge clique cover and almost all potential maximal cliques correspond to tripartitions of an edge clique cover.

On some of the problems we improve the time complexity to $O^*(2^{\mathsf{cc}'})$, where cc' is the size of an edge clique cover given as an input. The $O^*(2^{\mathsf{cc}'})$ time algorithms use the same dynamic programming states as the standard PMC framework, but instead of using PMCs for transitions we use fast subset convolution [2]. The application of fast subset convolution requires ad-hoc techniques for each problem to take into account the cost caused by the PMC implicitly selected by the convolution.

We also give algorithms that work in polynomial space and in times $O^*(9^{\mathsf{cc}'})$ and $O^*(9^{\mathsf{cc}+O(\log^2 \mathsf{cc})})$. These algorithms are based on a polynomial space and $O^*(9^{\mathsf{cc}})$ time algorithm for enumerating PMCs and on a lemma asserting that every minimal triangulation of a graph G has a maximal clique that is in a sense a balanced separator with respect to an edge clique cover of G.

1.3 Contributions

We start by giving bounds for the numbers of minimal separators and PMCs.

▶ **Theorem 1.** *If G is a graph with an edge clique cover of size cc, then the number of minimal separators of G is at most 2^{cc} and the number of potential maximal cliques of G is at most 3^{cc}.*

There are $O^*(|\Delta(G)|)$ time algorithms for enumerating the minimal separators $\Delta(G)$ of a graph G [1, 35], so it follows that the minimal separators of a graph can be enumerated in $O^*(2^{\mathsf{cc}})$ time. For enumerating PMCs no algorithms that are linear in the size of the output and polynomial in the size of the input are known[1]. Despite that, we are able to design an efficient algorithm for enumerating PMCs parameterized by cc, even when no edge clique cover is given as input. The fact that the algorithm works even when no edge clique cover is given as input is crucial because there is no $O^*(2^{2^{o(\mathsf{cc})}})$ time parameterized algorithm for minimum edge clique cover assuming the exponential time hypothesis [9].

▶ **Theorem 2.** *There is an algorithm that given a graph G whose minimum edge clique cover has size cc enumerates the potential maximal cliques of G in $O^*(3^{\mathsf{cc}})$ time.*

It follows that all problems that can be solved in $O^*(|\Pi(G)|)$ time when the set $\Pi(G)$ of PMCs of the input graph G is given can be solved in $O^*(3^{\mathsf{cc}})$ time, even when no edge clique cover is given as input. We remark that in addition to the problems mentioned earlier, the set

[1] Obtaining an output-linear input-polynomial time algorithm for enumerating PMCs has been explicitly stated as an open problem in 2006 [3] and to the best of our knowledge is still open.

of such problems includes all problems in the framework called *maximum induced subgraph of bounded treewidth* [16], including for example the problems maximum independent set, minimum feedback vertex set, and longest induced path [16].

▶ **Corollary 3.** *Treewidth, weighted minimum fill-in, and all instances of maximum induced subgraph of bounded treewidth can be solved in $O^*(3^{\mathsf{cc}})$ time, where cc is the size of a minimum edge clique cover of the input graph. Fractional hypertreewidth can be solved in $O^*(3^m)$ time, where m is the number of hyperedges. The maximum compatibility problem of binary phylogenetic characters can be solved in $O^*(3^n)$ time, where n is the number of taxa.*

When an edge clique cover of size cc' is given as an input, some of the algorithms can be optimized to $O^*(2^{\mathsf{cc}'})$ time.

▶ **Theorem 4.** *Treewidth, minimum fill-in, and chordal sandwich can be solved in $O^*(2^{\mathsf{cc}'})$ time, where cc' is the size of an edge clique cover given as an input.*

▶ **Corollary 5.** *The perfect phylogeny problem can be solved in $O^*(2^n)$ time, where n is the number of taxa.*

A previous parameterized algorithm for perfect phylogeny works in time $O^*(4^r)$, where $r \leq n$ is the arity of characters [24]. Our $O^*(2^n)$ time algorithm improves over it in the case when $r > n/2$. This case is motivated by the fact that the $O^*(4^r)$ algorithm works for partial characters only via a reduction that sets $r \geq nf$ for a fraction f of missing data [34].

We also give polynomial space algorithms for some of the problems. The algorithms work in $O^*(9^{\mathsf{cc}'})$ time when an edge clique cover of size cc' is given as an input and in $O^*(9^{\mathsf{cc}+O(\log^2 \mathsf{cc})})$ time when the parameter cc is given as an input.

▶ **Theorem 6.** *Treewidth and weighted minimum fill-in can be solved in polynomial space and $O^*(9^{\mathsf{cc}'})$ time, where cc' is the size of an edge clique cover given as an input. Furthermore, fractional hypertreewidth can be solved in polynomial space and $O^*(9^m)$ time.*

▶ **Corollary 7.** *The perfect phylogeny problem and the maximum compatibility problem of binary phylogenetic characters can be solved in polynomial space and $O^*(9^n)$ time, where n is the number of taxa.*

▶ **Theorem 8.** *Treewidth and weighted minimum fill-in can be solved in polynomial space and $O^*(9^{\mathsf{cc}+O(\log^2 \mathsf{cc})})$ time, where cc is an integer given as an input that is at least the size of a minimum edge clique cover of the input graph.*

Finally, we demonstrate the tightness of the bounds on the numbers of minimal separators and potential maximal cliques.

▶ **Theorem 9.** *There is a family of graphs, containing for each positive integer cc a graph with edge clique cover of size cc, $O(\mathsf{cc}^2)$ vertices, $\Theta(2^{\mathsf{cc}})$ minimal separators, and $\Theta(3^{\mathsf{cc}})$ potential maximal cliques.*

Furthermore, the parameter edge clique cover cannot be relaxed to vertex clique cover while retaining FPT, or even XP, bounds.

▶ **Theorem 10.** *There is a family of graphs, containing for each positive integer n a graph with n vertices, vertex clique cover of size 2, and $\Theta(2^{n/2})$ minimal separators.*

1.4 Related Work

The prior FPT algorithms for enumerating PMCs include an $O^*(4^{\texttt{vc}})$ time algorithm, where $\texttt{vc}$ is the size of a minimum vertex cover, and an $O^*(1.7347^{\texttt{mw}})$ time algorithm, where $\texttt{mw}$ is the modular width [15], extending the $O(1.7347^n)$ time algorithm, where n is the number of vertices [16]. One can see that edge clique cover and vertex cover are orthogonal parameters by considering complete graphs and star graphs. For modular width, the relation $\texttt{mw} \leq 2^{\texttt{cc}}$ holds, but there are graphs with $\texttt{mw} = 2^{\texttt{cc}} - 2$. In the conclusion of [15] the authors mentioned that they are not aware of other FPT parameters than $\texttt{vc}$ and $\texttt{mw}$ for PMCs and asked whether more parameterizations could be obtained.

Other parameterized approaches on the PMC framework include an FPT modulator parameter [28] and an XP parameterization for minimal separators in H-graphs [13]. The modulator parameter is orthogonal to edge clique cover. On H-graphs, the graphs with edge clique cover of size $\texttt{cc}$ are $K_{\texttt{cc}}$-graphs, so the H-graph parameterization implies an $n^{O(\texttt{cc}^2)}$ time algorithm for enumerating PMCs.

In addition to the already discussed $O^*(4^r)$ time algorithm for perfect phylogeny [24], we are not aware of prior single-exponential FPT algorithms with the same parameters as our algorithms. For fractional hypertreewidth there are parameterized algorithms whose parameters depend on the sizes of intersections of hyperedges [12]. For treewidth and chordal sandwich, different techniques have been used to obtain an $O^*(3^{\texttt{vc}})$ time algorithm for treewidth [7] and an $O^*(2^{\texttt{vc}'})$ time algorithm for chordal sandwich, where $\texttt{vc}'$ is the size of a minimum vertex cover of the admissible edge set [21].

1.5 Organization of the Paper

The proofs of lemmas marked with $\star$ are omitted and can be found in the full version of the paper. In Section 2 we give necessary definitions and background on minimal triangulations and PMCs. In Section 3 we characterize minimal separators and PMCs based on edge clique cover, proving Theorem 1. In Section 4 we give enumeration algorithms for PMCs, proving Theorem 2. In Section 5 we give faster algorithms for the case when an edge clique cover is given as an input, proving Theorem 4. In Section 6 we give polynomial space algorithms, proving Theorems 6 and 8. In Section 7 we demonstrate the tightness of our results, proving Theorems 9 and 10. Proofs for the relation of modular width and edge clique cover claimed in Section 1.4 can be found in the full version of the paper. We conclude in Section 8.

2 Preliminaries

We recall the standard graph notation that we use and preliminaries on minimal triangulations. We also give formal definitions of the problems that we consider and introduce our notation related to edge clique cover.

2.1 Notation on Graphs

We consider graphs that are finite, simple, and undirected. We assume that the graphs given as input are connected. For graphs with multiple connected components, the algorithms can be applied to each connected component independently. The sets of vertices and edges of a graph G are denoted by $V(G)$ and $E(G)$, respectively. The set of edges of a complete graph with vertex set X is X^2. The subgraph $G[X]$ induced by $X \subseteq V(G)$ has $V(G[X]) = X$ and $E(G[X]) = E(G) \cap X^2$. We also use the notation $G \setminus X = G[V(G) \setminus X]$. The vertex sets of connected components of a graph G are $\mathcal{C}(G)$. The set of neighbors of a vertex v is denoted by $N(v)$ and the set of neighbors of a vertex set X by $N(X) = \bigcup_{v \in X} N(v) \setminus X$. The closed neighborhood of a vertex v is $N[v] = N(v) \cup \{v\}$ and the closed neighborhood of a vertex set X is $N[X] = N(X) \cup X$. A clique of a graph G is a vertex set X such that $G[X]$ is complete. The set of inclusion maximal cliques of G is denoted by $\mathrm{MC}(G)$.

2.2 Minimal Triangulations

A graph is *chordal* if it has no induced cycle of four or more vertices. A chordal graph H is a *triangulation* of a graph G if $V(G) = V(H)$ and $E(G) \subseteq E(H)$. A triangulation H of G is a *minimal triangulation* of G if there is no triangulation H' of G with $E(H') \subsetneq E(H)$. The edges in $E(H) \setminus E(G)$ are called *fill-edges*. A vertex set $\Omega \subseteq V(G)$ is a *potential maximal clique* (PMC) of G if there is a minimal triangulation H of G such that $\Omega \in \mathrm{MC}(H)$. The set of PMCs of G is denoted by $\Pi(G)$.

A vertex set S is a *minimal a,b-separator* of graph G if the vertices a and b are in different components of $G \setminus S$, and S is inclusion minimal in this regard. A *full component* of a vertex set X is a component $C \in \mathcal{C}(G \setminus X)$ with $N(C) = X$. We note that S is a minimal a,b-separator if and only if S has distinct full components containing a and b. A vertex set S is a *minimal separator* if it is a minimal a,b-separator for some pair a,b, i.e., it has at least two full components. We denote the set of minimal separators of G with $\Delta(G)$.

A *block* of a graph G is a vertex set $C \subseteq V(G)$ such that $N(C) \in \Delta(G)$. We remark that a common notation is to call such a pair $(N(C), C)$ a full block [5]. In modern formulations of the PMC framework the concept of non-full blocks is not needed [16], so we simplify the notation by identifying the block with only the vertex set C.

Next we recall a couple of required propositions on the structure of PMCs.

▶ **Proposition 11** ([5]). *A vertex set $\Omega \subseteq V(G)$ is a PMC of a graph G if and only if*

1. *$N(C) \subsetneq \Omega$ for all $C \in \mathcal{C}(G \setminus \Omega)$, i.e., no component of Ω is full, and*
2. *for all pairs of distinct vertices $u, v \in \Omega$, either $\{u, v\} \in E(G)$ or there is a component $C \in \mathcal{C}(G \setminus \Omega)$ with $\{u, v\} \subseteq N(C)$.*

We will refer to condition 1 of Proposition 11 as the *no full component condition* and to condition 2 as the *cliquish condition.* Note that Proposition 11 implies an $O(nm)$ time algorithm for testing if a vertex set is a PMC [5].

▶ **Proposition 12** ([5]). *If Ω is a PMC of a graph G then all components $C \in \mathcal{C}(G \setminus \Omega)$ are blocks of G.*

We call the components $C \in \mathcal{C}(G \setminus \Omega)$ the blocks of Ω. Note that $N(C) \subsetneq \Omega$, i.e., the minimal separators of the blocks $C \in \mathcal{C}(G \setminus \Omega)$ are strict subsets of the PMC. We also need the following proposition connecting PMCs and blocks.

▶ **Proposition 13** ([5]). *If Ω is a PMC of a graph G and C is a block of Ω, then there is a full component C' of $N(C)$ such that $\Omega \subseteq N[C']$.*

The following lemma, which follows from Proposition 11, simplifies some of our proofs.

▶ **Lemma 14** ($\star$). *If Ω is a PMC of a graph G and contains a vertex $v \in \Omega$ such that no block $C \in \mathcal{C}(G \setminus \Omega)$ has $v \in N(C)$, then $\Omega = N[v]$.*

2.3 Definitions of Problems

Let $\mathrm{Tr}(G)$ denote the set of triangulations of a graph G. The treewidth of a graph G is $\min_{H \in \mathrm{Tr}(G)} \max_{\Omega \in \mathrm{MC}(H)} |\Omega| - 1$. The minimum fill-in of G is $\min_{H \in \mathrm{Tr}(G)} |E(H) \setminus E(G)|$. Given a weight function $w : V(G)^2 \to \mathbb{R}_{\geq 0}$, the weighted minimum fill-in of G with respect to w is $\min_{H \in \mathrm{Tr}(G)} \sum_{e \in (E(H) \setminus E(G))} w(e)$. Given a graph G and a set of admissible edges $F \subseteq V(G)^2 \setminus E(G)$, the chordal sandwich problem is to determine if there is a triangulation H of G with $E(H) \subseteq E(G) \cup F$. Note that chordal sandwich can be reduced to weighted minimum fill-in on the same graph G.

A hypergraph $\mathcal{G}$ has a set of vertices $V(\mathcal{G})$ and a set of hyperedges $E(\mathcal{G})$ that are arbitrary subsets of vertices. The primal graph $P(\mathcal{G})$ of $\mathcal{G}$ has vertices $V(P(\mathcal{G})) = V(\mathcal{G})$ and edges $E(P(\mathcal{G})) = \bigcup_{e \in E(\mathcal{G})} e^2$. A fractional edge cover of a set $X \subseteq V(\mathcal{G})$ is an assignment $c : E(\mathcal{G}) \to \mathbb{R}_{\geq 0}$ so that for each $v \in X$ it holds that $\sum_{v \in e \in E(\mathcal{G})} c(e) \geq 1$. The size of a fractional edge cover c is $\sum_{e \in E(\mathcal{G})} c(e)$. The minimum size of a fractional edge cover of a set $X \subseteq V(\mathcal{G})$ is denoted by $\mathrm{FCOV}(X)$. Note that $\mathrm{FCOV}(X)$ can be computed in polynomial time by linear programming [18]. The fractional hypertreewidth of a hypergraph $\mathcal{G}$ is $\min_{H \in \mathrm{Tr}(P(\mathcal{G}))} \max_{\Omega \in \mathrm{MC}(H)} \mathrm{FCOV}(\Omega)$ [18, 30].

For all of the aforementioned problems there is an optimal solution corresponding to a minimal triangulation. Furthermore, all of the problems can be solved in $O^*(|\Pi(G)|)$ time when $\Pi(G)$ is given as an input [14, 17, 19, 29, 30].

2.4 Notation on Edge Clique Cover

An edge clique cover of a graph G is a collection $\mathcal{W}$ of cliques of G so that $\bigcup_{W \in \mathcal{W}} W^2 = E(G)$. We often manipulate vertex sets based on an edge clique cover $\mathcal{W}$. For a vertex $v \in V(G)$, we denote by $\mathcal{W}[v] = \{W \in \mathcal{W} \mid v \in W\}$ the set of cliques in $\mathcal{W}$ that contain v. Similarly, for a vertex set X we denote by $\mathcal{W}[X] = \bigcup_{v \in X} \mathcal{W}[v]$ the set of cliques in $\mathcal{W}$ that intersect X.

A non-empty subset $\mathcal{W}' \subseteq \mathcal{W}$ of an edge clique cover $\mathcal{W}$ is called a part of the edge clique cover. The vertices in $V(G, \mathcal{W}') = \{v \in V(G) \mid \mathcal{W}[v] \subseteq \mathcal{W}'\}$ are called the vertices of the part $\mathcal{W}'$. We use a shorthand $V(G, \mathcal{W}_1, \ldots, \mathcal{W}_p) = V(G, \mathcal{W}_1) \cup \ldots \cup V(G, \mathcal{W}_p)$ to denote the union of vertices of multiple parts. The components of a part are $\mathcal{C}(\mathcal{W}') = \mathcal{C}(G[V(G, \mathcal{W}')])$. A part is called good if all of its components are blocks, in which case the components of the part may be called the blocks of the part. Note that a part $\mathcal{W}'$ with $V(G, \mathcal{W}') = \emptyset$ is good. Two disjoint parts $\mathcal{W}_1, \mathcal{W}_2$ are called compatible if $V(G, \mathcal{W}_1, \mathcal{W}_2) = V(G, \mathcal{W}_1 \cup \mathcal{W}_2)$.

3 Characterization of the Central Combinatorial Objects

In this section we show that if a graph has an edge clique cover of size cc, then the number of blocks and minimal separators of the graph is at most 2^{cc} and the number of potential maximal cliques is at most 3^{cc}. The characterizations of these objects will be later used in the design of the algorithms.

We start by showing that blocks correspond to parts of an edge clique cover.

▶ **Lemma 15.** *Let G be a graph and $\mathcal{W}$ an edge clique cover of G. If C is a block of G then $V(G, \mathcal{W}[C]) = C$.*

Proof. Clearly $C \subseteq V(G, \mathcal{W}[C])$. Note that $V(G, \mathcal{W}[C]) \subseteq N[C]$ because any vertex v intersecting a common clique with a vertex $u \in C$ must be a neighbor of u. Suppose there is a vertex $v \in (V(G, \mathcal{W}[C]) \cap N(C))$. Let C' be a full component of $N(C)$ distinct from C, implying that $v \in N(C')$. All the cliques that v intersects also intersect with C, and therefore there must be a vertex in C' that is in a clique intersecting with C which is a contradiction to the fact that $N(C)$ separates C and C'. ◀

By Lemma 15, any block C of G can be uniquely identified with a set $\mathcal{W}[C] \subseteq \mathcal{W}$. Therefore, the number of blocks is at most 2^{cc}. Any minimal separator is identified as $N(C)$ of at least two blocks C, so the number of minimal separators is at most $2^{\mathsf{cc}-1}$.

Next we show that each PMC is either a closed neighborhood of a vertex or can be represented by a tripartition of edge clique cover.

▶ **Lemma 16.** *Let G be a graph and $\mathcal{W}$ an edge clique cover of G. If Ω is a potential maximal clique of G, then either (1) $\Omega = N[v]$ for a vertex $v \in V(G)$ or (2) $\mathcal{C}(G \setminus \Omega) = \mathcal{C}(\mathcal{W}_1) \cup \mathcal{C}(\mathcal{W}_2) \cup \mathcal{C}(\mathcal{W}_3)$, where $\{\mathcal{W}_1, \mathcal{W}_2, \mathcal{W}_3\}$ is a partition of $\mathcal{W}$ into good parts.*

Proof. Suppose that case 1 does not apply, i.e., Ω is not equal to $N[v]$ for any $v \in V(G)$. By Proposition 12 the components $C \in \mathcal{C}(G \setminus \Omega)$ are blocks and therefore by Lemma 15 they define a collection $P = \{\mathcal{W}[C] \mid C \in \mathcal{C}(G \setminus \Omega)\}$ of disjoint good parts of $\mathcal{W}$. If the collection P is not a partition of $\mathcal{W}$, add an additional part $\mathcal{W}' = \mathcal{W} \setminus (\bigcup_{\mathcal{W}_i \in P} \mathcal{W}_i)$ to the collection to make it a partition of $\mathcal{W}$. We have that $V(G, \mathcal{W}') = \emptyset$ because if there would be a vertex v with $\mathcal{W}[v] \subseteq \mathcal{W}'$, then v would be in Ω because it is not in any block, and also v would not be in the neighborhood of any block, so by Lemma 14 we would have $\Omega = N[v]$. Now we have a partition P of $\mathcal{W}$ into good parts with $\bigcup_{\mathcal{W}_i \in P} \mathcal{C}(\mathcal{W}_i) = \mathcal{C}(G \setminus \Omega)$.

The partition P has at least two parts because otherwise Ω would be empty. If the number of parts is two, i.e. $P = \{\mathcal{W}_1, \mathcal{W}_2\}$, let $\mathcal{W}_1$ be a part such that $V(G, \mathcal{W}_1)$ is not empty. Such a part exists because otherwise $\Omega = V(G)$ and case 1 would apply. No vertex of Ω is in $V(G, \mathcal{W}_2)$, so $V(G, \mathcal{W}_1)$ is a full component of Ω, which is a contradiction to the no full component condition.

If the number of parts is at least three, merge arbitrary compatible pairs of parts until the number of parts is three or no pairs of parts can be merged anymore. If we end up with three parts, we are done. If we end up with more than three parts, i.e. $P = \{\mathcal{W}_1, \mathcal{W}_2, \mathcal{W}_3, \mathcal{W}_4, \ldots\}$, then take a vertex $u \in (V(G, \mathcal{W}_1 \cup \mathcal{W}_2) \setminus V(G, \mathcal{W}_1, \mathcal{W}_2))$ and a vertex $v \in (V(G, \mathcal{W}_3 \cup \mathcal{W}_4) \setminus V(G, \mathcal{W}_3, \mathcal{W}_4))$. Because of our assumption that we cannot continue the merging process anymore both of these vertices exist and are in Ω. However, there is no edge between u and v and there is no common component in whose neighborhood u and v are, which is a contradiction to the cliquish condition. ◀

We call the PMCs corresponding to case 1 of Lemma 16 type 1 PMCs and the PMCs corresponding to case 2 type 2 PMCs. The number of type 1 PMCs is at most n, the number of vertices. Another upper bound for the number of type 1 PMCs is 2^{cc}, because if $\mathcal{W}[v] = \mathcal{W}[u]$ for vertices v and u then $N[v] = N[u]$. The number of type 2 PMCs is at most $S(\mathsf{cc}, 3)$, where S denotes the Stirling numbers of the second kind. One can verify that $S(\mathsf{cc}, 3) + 2^{\mathsf{cc}} \leq 3^{\mathsf{cc}}$ and the bound for PMCs follows.

4 Enumeration Algorithms

In this section we modify the Bouchitté–Todinca algorithm [6] for enumerating PMCs to give an $O^*(3^{\mathsf{cc}})$ time PMC enumeration algorithm and a polynomial space $O^*(9^{\mathsf{cc}})$ time PMC enumeration algorithm.

The Bouchitté–Todinca algorithm is based on theorems characterizing PMCs based on minimal separators. We summarize the theorems in the following proposition.

▶ **Proposition 17** ([6]). *Let G be a connected graph with $|V(G)| > 1$ and v any vertex of G. If Ω is a PMC of G, one of the following holds.*

1. *$\Omega \setminus \{v\} \in \Pi(G \setminus \{v\})$.*
2. *$\Omega \setminus \{v\} \in \Delta(G)$.*
3. *$\Omega = S \cup T$, where $S \in \Delta(G)$ and $T \in \Delta(G[C \cup \{x, y\}])$, where C is a full component of S and x and y are non-adjacent vertices in S.*

The algorithm uses case 1 to generate n induced subgraphs of the input graph, from which PMCs are generated by cases 2 and 3. Note that the size of a minimum edge clique cover is monotone with respect to induced subgraphs. We need the following lemma, which follows from Proposition 11, to ensure that each PMC of each induced subgraph corresponds to at most one PMC of the original graph.

▶ **Lemma 18** ([6]). *Let G be a graph and $v \in V(G)$. If $\Omega \in \Pi(G \setminus \{v\})$, then at most one of Ω and $\Omega \cup \{v\}$ is a PMC of G.*

Now, we can just enumerate PMCs from cases 2 and 3 in each of the n induced subgraphs, and each time a PMC is found we use Lemma 18 to generate at most one PMC of the original graph in polynomial time.

Next we complete the description of the algorithm by showing that PMCs from cases 2 and 3 can be enumerated in $O^*(3^{\mathsf{cc}})$ time. The proof is based on Lemma 15 on the structure and the number of blocks.

▶ **Lemma 19.** *There is an algorithm that given a graph G whose minimum edge clique cover has size cc enumerates the PMCs of G, possibly with duplicates, in polynomial space and $O^*(3^{\mathsf{cc}})$ time.*

Proof. By Lemma 18, it is sufficient to enumerate PMCs corresponding to cases 2 and 3 of Proposition 17. For case 2, the bound follows from the 2^{cc} bound on minimal separators and a polynomial space $O^*(|\Delta(G)|)$ time minimal separator enumeration algorithm [35]. For case 3, we do polynomial space enumeration of minimal separators in the graph G, and every time we output a minimal separator S, we do polynomial space enumeration of minimal separators in the graph $G[C \cup \{x, y\}]$ for all full components C of S and all non-adjacent pairs $x, y \in S$.

We do the inner iteration $O(n^2)$ times for each block C of G. The complexity of the inner iteration depends on the size of a minimum edge clique cover of $G[C \cup \{x, y\}]$. Let $\mathcal{W}$ be a minimum edge clique cover of G. By Lemma 15, the block C corresponds to an unique subset $\mathcal{W}[C]$ of $\mathcal{W}$. The subset $\mathcal{W}[C]$ is an edge clique cover of $G[C \cup \{x, y\}]$ because all edges in it are adjacent to C because x and y are non-adjacent. Therefore, the time complexity of the inner iteration is $O^*(2^{|\mathcal{W}[C]|})$, and therefore, the time complexity of the algorithm is at most $\sum_{\mathcal{W}' \subseteq \mathcal{W}} O^*(2^{|\mathcal{W}'|}) = O^*(3^{\mathsf{cc}})$. ◀

Using for example sorting we can deduplicate the output of the algorithm of Lemma 19 and an $O^*(3^{\mathsf{cc}})$ time exponential space algorithm for enumerating PMCs without duplicates follows. For deduplication in polynomial space, we use a simple trick that is efficient enough for our purposes.

▶ **Lemma 20.** *There is an algorithm that given a graph G whose minimum edge clique cover has size cc enumerates the PMCs of G in polynomial space and $O^*(9^{\mathsf{cc}})$ time.*

Proof. Run the algorithm of Lemma 19 multiple times in succession, each time outputting the lexicographically smallest PMC that is lexicographically larger than the previous PMC outputted, until no such PMC is found. Now, using the algorithm at most 3^{cc} times we have outputted the PMCs of G in lexicographically strictly increasing order. ◀

5 Faster Algorithms When Edge Clique Cover is Given

We use fast subset convolution [2] to design $O^*(2^{\mathtt{cc}'})$ time algorithms for treewidth, minimum fill-in, and chordal sandwich, where $\mathtt{cc}'$ is the size of an edge clique cover given as an input. In particular, we make use of the following result.

▶ **Proposition 21** ([2]). *Let X be a set, $f : 2^X \to [M]$ and $g : 2^X \to [M]$ functions from the set 2^X of all subsets of X to the set of integers up to M. The function $(f * g)$ defined as $(f * g)(Y) = \min_{Y' \subseteq Y} f(Y') + g(Y \setminus Y')$ can be computed for all $Y \subseteq X$ in $O^*(2^{|X|}M)$ time.*

The algorithms we introduce are modifications of the dynamic programming phase of the PMC framework. In most of this section our presentation is general in the sense that it applies to each of the three problems. We use the term "optimal triangulation" to refer to a triangulation with the minimum size of a maximum clique in the context of treewidth, a triangulation with the least number of edges in the context of minimum fill-in, and to a triangulation that has no non-admissible fill-edges in the context of chordal sandwich (or to information that no such triangulation exists).

We start by recalling the dynamic programming phase of the PMC framework. The states of the dynamic programming correspond to *realizations* of blocks.

▶ **Definition 22** ([5]). *Let G be a graph and C a block of G. A realization $R(C)$ of C is a graph with $V(R(C)) = N[C]$ and $E(R(C)) = E(G[N[C]]) \cup N(C)^2$.*

The following proposition characterizes minimal triangulations of a realization of a block.

▶ **Proposition 23** ([5]). *Let G be a graph and C a block of G. The graph H is a minimal triangulation of $R(C)$ if and only if (i) $V(H) = N[C]$ and (ii) there is a PMC $\Omega \in \Pi(G)$ with $N(C) \subseteq \Omega \subseteq N[C]$ and*

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

where H_i is any minimal triangulation of $R(C_i)$. It also holds that each C_i is a block of G.

Proposition 23 implies dynamic programming formulas for computing optimal triangulations of realizations of all blocks [5, 14, 29].

We use the following proposition for a base case.

▶ **Proposition 24** ([5]). *Let G be a graph that is not complete. The graph H is a minimal triangulation of G if and only if (i) $V(H) = V(G)$ and (ii) there is a minimal separator $S \in \Delta(G)$ with*

$$E(H) = \bigcup_{C_i \in \mathcal{C}(G \setminus S)} E(H_i),$$

where H_i is any minimal triangulation of $R(C_i)$. It also holds that each C_i is a block of G.

Once we have computed optimal triangulations of realizations of all blocks, we can compute an optimal triangulation of the graph via Proposition 24 in time $O^*(2^{\mathtt{cc}})$. For computing optimal triangulations of realizations, the bottleneck in implementing the recursion of Proposition 23 is in iterating over the PMCs.

The high-level idea of our algorithm is that we use Proposition 23 directly only with type 1 PMCs, i.e., PMCs $\Omega = N[v]$ for some vertex $v \in V(G)$. For type 2 PMCs, we simulate the iteration over PMCs with fast subset convolution. In particular, we show that each PMC

Ω of type 2 with $N(C) \subseteq \Omega \subseteq N[C]$ can be expressed in terms of two disjoint good parts $\mathcal{W}_1$ and $\mathcal{W}_2$ of $\mathcal{W}[C]$, where $\mathcal{W}$ is an edge clique cover of G. In the case of treewidth, minimum fill-in, and chordal sandwich, an optimal partition of every subset $\mathcal{W}' \subseteq \mathcal{W}$ into two good parts $\mathcal{W}_1$ and $\mathcal{W}_2$ can be computed with fast subset convolution, provided that we have first computed optimal triangulations of realizations of all blocks in $\mathcal{C}(\mathcal{W}_1)$ and $\mathcal{C}(\mathcal{W}_2)$.

We first show the direction that each PMC can be expressed in terms of $\mathcal{W}_1$ and $\mathcal{W}_2$.

▶ **Lemma 25.** *Let G be a graph, $\mathcal{W}$ an edge clique cover of G, and C a block of G. If a graph H is a minimal triangulation of $R(C)$, then either (1) there is a vertex $v \in V(G)$ and*

$$E(H) = N[v]^2 \cup \bigcup_{C_i \in \mathcal{C}(R(C) \setminus N[v])} E(H_i),$$

where $N[v] \in \Pi(G)$, $N(C) \subseteq N[v] \subseteq N[C]$, and H_i is a minimal triangulation of $R(C_i)$, or (2) there is a partition $\{\mathcal{W}_1, \mathcal{W}_2, \mathcal{W}_o\}$ of $\mathcal{W}$ into good parts with $\mathcal{W}_1 \cup \mathcal{W}_2 \subseteq \mathcal{W}[C]$, a block $C' \in \mathcal{C}(\mathcal{W}_o)$ with $N(C) = N(C')$, and

$$E(H) = \Omega^2 \cup \bigcup_{C_i \in \mathcal{C}(\mathcal{W}_1) \cup \mathcal{C}(\mathcal{W}_2) \cup (\mathcal{C}(\mathcal{W}_o) \setminus \mathcal{C}(G \setminus N(C)))} E(H_i),$$

where $\Omega = V(G) \setminus V(G, \mathcal{W}_1, \mathcal{W}_2, \mathcal{W}_o)$ and H_i is a minimal triangulation of $R(C_i)$.

Proof. Case 1 corresponds to Proposition 23 with PMCs of type 1. Next we prove that case 2 covers all PMCs of type 2.

Let Ω be any PMC of G with $N(C) \subseteq \Omega \subseteq N[C]$ such that there is no vertex v with $\Omega = N[v]$. Consider the part $\mathcal{W}'_o = \mathcal{W} \setminus \mathcal{W}[C]$ and let us prove that $\mathcal{W}'_o$ is a good part and $\mathcal{C}(\mathcal{W}'_o) = \mathcal{C}(G \setminus N(C)) \setminus \{C\}$. Observe that $\{C, N(C), V(G, \mathcal{W}'_o)\}$ is a partition of $V(G)$, and moreover there are no edges between C and $V(G, \mathcal{W}'_o)$. Now, for any component $C' \in \mathcal{C}(\mathcal{W}'_o)$, it must hold that $N(C') \subseteq N(C)$, and therefore $N(C')$ is a minimal separator and therefore $\mathcal{W}'_o$ is a good part whose blocks are the components of $G \setminus N(C)$ except C.

Similarly as in the proof of Lemma 16, consider the collection of disjoint good parts $P = \{\mathcal{W}[C_i] \mid C_i \in \mathcal{C}(G \setminus \Omega)\}$. All of the parts that intersect $\mathcal{W}'_o$ are subsets of $\mathcal{W}'_o$ because they do not intersect $\mathcal{W}[C]$, and therefore we replace the parts that intersect $\mathcal{W}'_o$ by the part $\mathcal{W}'_o$. Now by similar arguments as in Lemma 16 we can add one additional part to the collection to make it a partition of $\mathcal{W}$. Now we have a partition P of $\mathcal{W}$ with $|P| \geq 2$ and $\mathcal{W}'_o \in P$. Moreover, $\bigcup_{\mathcal{W}_i \in P} \mathcal{C}(\mathcal{W}_i) = \mathcal{C}(G \setminus \Omega)$. If $|P| = 2$, then it would hold that $P = \{\mathcal{W}'_o, \mathcal{W}[C]\}$, in which case $\Omega = N(C)$ would hold, which is a contradiction. If there are at least three parts, then merge compatible parts until we have three parts or cannot merge parts anymore. By the proof of Lemma 16, we will end up with three parts. Now let $\mathcal{W}_o$ be the part that contains $\mathcal{W}'_o$ and $\mathcal{W}_1$ and $\mathcal{W}_2$ the other two parts. Because $\mathcal{W}'_o = \mathcal{W} \setminus \mathcal{W}[C]$, we have that $\mathcal{W}_1 \cup \mathcal{W}_2 \subseteq \mathcal{W}[C]$. Moreover, because $N(C)$ has at least two full components, and all components of $N(C)$ except C are components of $\mathcal{W}'_o$, we have that there is a block $C' \in \mathcal{C}(\mathcal{W}_o)$ with $N(C') = N(C)$.

Finally, we need to show that $\mathcal{C}(R(C) \setminus \Omega) = \mathcal{C}(\mathcal{W}_1) \cup \mathcal{C}(\mathcal{W}_2) \cup (\mathcal{C}(\mathcal{W}_o) \setminus \mathcal{C}(G \setminus N(C)))$. By Proposition 13 we have that $\mathcal{C}(R(C) \setminus \Omega) = \mathcal{C}(G \setminus \Omega) \setminus \mathcal{C}(G \setminus N(C))$ and therefore $\mathcal{C}(R(C) \setminus \Omega) = \mathcal{C}(\mathcal{W}_1) \cup \mathcal{C}(\mathcal{W}_2) \cup \mathcal{C}(\mathcal{W}_o) \setminus \mathcal{C}(G \setminus N(C))$. All blocks of $\mathcal{W}_1$ and $\mathcal{W}_2$ are subsets of C because $\mathcal{W}_1 \cup \mathcal{W}_2 \subseteq \mathcal{W}[C]$. ◀

The following lemma guarantees that the characterization of PMCs of type 2 in Lemma 25 is sound in the sense that all graphs H that it defines are (not necessarily minimal) triangulations of G.

▶ **Lemma 26** (⋆). *Let G be a graph, $\mathcal{W}$ an edge clique cover of G, and C a block of G. Furthermore, let $\{\mathcal{W}_1, \mathcal{W}_2, \mathcal{W}_o\}$ be a partition of $\mathcal{W}$ into good parts with $\mathcal{W}_1 \cup \mathcal{W}_2 \subseteq \mathcal{W}[C]$ and C' a block of G with $C' \in \mathcal{C}(\mathcal{W}_o)$ and $N(C) = N(C')$. Let H be any graph with (i) $V(H) = N[C]$ and (ii)*

$$E(H) = \Omega^2 \cup \bigcup_{C_i \in \mathcal{C}(\mathcal{W}_1) \cup \mathcal{C}(\mathcal{W}_2) \cup (\mathcal{C}(\mathcal{W}_o) \setminus \mathcal{C}(G \setminus N(C)))} E(H_i),$$

where $\Omega = V(G) \setminus V(G, \mathcal{W}_1, \mathcal{W}_2, \mathcal{W}_o)$ and H_i is any triangulation of $R(C_i)$. The graph H is a triangulation of $R(C)$.

In Lemmas 25 and 26 we formulated a recursion that characterizes all minimal triangulations of a realization $R(C)$ of a block C in terms of minimal triangulations of realizations $R(C')$ of blocks $C' \subsetneq C$. What remains is to integrate the computation of the optimal cost of the triangulation into this characterization. The following lemma is used for treewidth and minimum fill-in. It is simple, but we state it as a warmup for what follows.

▶ **Lemma 27** (⋆). *Let G be a graph, $\mathcal{W}$ an edge clique cover of G, $\{\mathcal{W}_1, \mathcal{W}_2, \mathcal{W}_o\}$ a partition of $\mathcal{W}$, and $\Omega = V(G) \setminus V(G, \mathcal{W}_1, \mathcal{W}_2, \mathcal{W}_o)$. It holds that $|\Omega| = |V(G)| - |V(G, \mathcal{W}_1)| - |V(G, \mathcal{W}_2)| - |V(G, \mathcal{W}_o)|$.*

Therefore, the size of Ω can be computed as a sum that considers $\mathcal{W}_1$, $\mathcal{W}_2$, and $\mathcal{W}_o$ independently, and therefore we can integrate the computation of it into fast subset convolution. For treewidth, we only have to make sure that $|\Omega| \leq k + 1$, where k is the upper bound for treewidth in the decision problem. For minimum fill-in, we can compute the number of edges in the triangulation of $R(C)$ as $\binom{|\Omega|}{2} + \sum_{C_i \in \mathcal{C}(R(C) \setminus \Omega)} (|E(H_i)| - \binom{|N(C_i)|}{2})$, where H_i is an optimal triangulation of the realization $R(C_i)$.

A similar lemma is used for chordal sandwich.

▶ **Lemma 28** (⋆). *Let G be a graph, $\mathcal{W}$ an edge clique cover of G, $\{\mathcal{W}_1, \mathcal{W}_2, \mathcal{W}_o\}$ a partition of $\mathcal{W}$ into good parts, and $\Omega = V(G) \setminus V(G, \mathcal{W}_1, \mathcal{W}_2, \mathcal{W}_o)$. It holds that $\Omega^2 = \bigcup_{\mathcal{W}_i \in \{\mathcal{W}_1, \mathcal{W}_2, \mathcal{W}_o\}} (V(G) \setminus V(G, \mathcal{W}_i, \mathcal{W} \setminus \mathcal{W}_i))^2$.*

Lemma 28 guarantees that each fill-edge caused by the PMC Ω can be "seen" from at least one of the parts $\mathcal{W}_1$, $\mathcal{W}_2$, $\mathcal{W}_o$, implying that it is sufficient to check each part independently to guarantee that Ω does not add any forbidden fill-edges. We remark that Lemma 28 appears to be difficult to generalize to count the exact number of fill-edges, which is the barrier why we are not able to give an $O^*(2^{\mathsf{cc}'})$ time algorithm for weighted minimum fill-in.

Algorithm 1 presents the full $O^*(2^{\mathsf{cc}'})$ time algorithm for treewidth. The algorithms for minimum fill-in and chordal sandwich are similar. The algorithm maintains a collection $\mathcal{B}$ of blocks C for which it is known that the treewidth of $R(C)$ is at most k. The invariant of the main loop of lines 3 to 19 is that after ith iteration, all blocks of size at most i and treewidth at most k have been added to $\mathcal{B}$. In each iteration of the main loop, the algorithm iterates over all good parts $\mathcal{W}'$ on lines 5 to 7, and if all realizations of blocks of the part have treewidth at most k adds the part to a collection $F_{|V(G,\mathcal{W}')|}$. These parts $\mathcal{W}'$ correspond to parts $\mathcal{W}_1$ and $\mathcal{W}_2$ of our lemmas. Then, fast subset convolution is applied on line 8 on the collections F to find for all combinations $(\mathcal{W}_1 \cup \mathcal{W}_2)$ of disjoint parts $\mathcal{W}_1$ and $\mathcal{W}_2$ the maximum number of vertices in $V(G, \mathcal{W}_1, \mathcal{W}_2)$. Then on lines 9 to 15 the algorithm iterates through all good parts $\mathcal{W}_o$, thus determining $(\mathcal{W}_1 \cup \mathcal{W}_2)$ and all other variables that need to be taken into account. In particular, note that each part $\mathcal{W}_o$ determines only polynomially many blocks C such that there is $C' \in \mathcal{C}(\mathcal{W}_o)$ with $N(C') = N(C)$.

Algorithm 1 Treewidth in $O^*(2^{\mathsf{cc}'})$ time.

```
Input  : Connected graph $G$, an edge clique cover $\mathcal{W}$ of $G$, and an integer $k$
Output : Whether the treewidth of $G$ is at most $k$
1  if $G$ is complete then return $|V(G)| \leq k+1$
2  Let $\mathcal{B} \leftarrow \emptyset$ be a collection of blocks of $G$
3  for $i \leftarrow 1$ to $n$ do
4      For each $0 \leq j \leq n$ let $F_j \leftarrow \emptyset$ be a collection of subsets of $\mathcal{W}$
5      for each good part $\mathcal{W}' \subseteq \mathcal{W}$ do
6          if $\mathcal{C}(\mathcal{W}') \subseteq \mathcal{B}$ then
7              $F_{|V(G,\mathcal{W}')|} \leftarrow F_{|V(G,\mathcal{W}')|} \cup \{\mathcal{W}'\}$
8      Use fast subset convolution to compute for each subset $\mathcal{W}' \subseteq \mathcal{W}$ the maximum
        value of $j+k$ such that there is $\mathcal{W}'' \subseteq \mathcal{W}'$ with $\mathcal{W}'' \in P_j$ and $(\mathcal{W}' \setminus \mathcal{W}'') \in P_k$
9      for each good part $\mathcal{W}_o \subseteq \mathcal{W}$ do
10         Let $t$ be the value on $\mathcal{W} \setminus \mathcal{W}_o$ computed in line 8
11         if $t$ exists and $n - t - |V(G, \mathcal{W}_o)| \leq k+1$ then
12             for each minimal separator $N(C')$ with $C' \in \mathcal{C}(\mathcal{W}_o)$ do
13                 if exists $C \in \mathcal{C}(G \setminus N(C'))$ with $(\mathcal{W} \setminus \mathcal{W}_o) \subseteq \mathcal{W}[C]$ then
14                     if $(\mathcal{C}(\mathcal{W}_o) \setminus \mathcal{C}(N(C))) \subseteq \mathcal{B}$ then
15                         $\mathcal{B} \leftarrow \mathcal{B} \cup \{C\}$
16     for each block $C$ of $G$ do
17         for $v \in V(G) \mid N[v] \in \Pi(G)$ and $|N[v]| \leq k+1$ and $N(C) \subseteq N[v] \subseteq N[C]$ do
18             if $\mathcal{C}(R(C) \setminus N[v]) \subseteq \mathcal{B}$ then
19                 $\mathcal{B} \leftarrow \mathcal{B} \cup \{C\}$
20 for $S \in \Delta(G)$ do
21     if $\mathcal{C}(G \setminus S) \subseteq \mathcal{B}$ then
22         return True
23 return False
```

The analysis of the algorithm focuses on transitions via PMCs of type 2 and proceeds by induction on the main loop invariant. The time complexity follows simply from fast subset convolution, the bound 2^{cc} on the number of blocks, and the fact that each iteration of the loop of the lines 9 to 15 takes polynomial time. The correctness is shown by combining the lemmas introduced in this section.

▶ **Lemma 29** ($\star$). *There is an algorithm that given a graph G with an edge clique cover of size cc' determines the treewidth and minimum fill-in of G in time $O^*(2^{\mathsf{cc}'})$. Furthermore, if also a set $F \subseteq V(G)^2 \setminus E(G)$ is given the algorithm determines if there is a triangulation H of G with $E(H) \subseteq E(G) \cup F$, i.e., solves the chordal sandwich problem.*

6 Polynomial Space Algorithms

We give polynomial space algorithms for treewidth, weighted minimum fill-in, and fractional hypertreewidth. The algorithms are based on the following characterization of minimal triangulations.

▶ **Proposition 30** ([5]). *Let G be a graph, H a minimal triangulation of G, and Ω a maximal clique of H. For each $C_i \in \mathcal{C}(G \setminus \Omega)$ there exists a minimal triangulation H_i of $R(C_i)$ such that*

$$E(H) = \Omega^2 \cup \bigcup_{C_i \in \mathcal{C}(G \setminus \Omega)} E(H_i).$$

Note that by iterating over all $\Omega \in \Pi(G)$ in Proposition 30 we can indeed construct all minimal triangulations of G. Furthermore, all graphs H constructed in this manner are minimal triangulations [5].

The idea of the algorithm is to use the recursion of Proposition 30 directly, without dynamic programming. The following "balanced PMC" lemma guarantees that we can expect the size of an edge clique cover to roughly halve in each level of the recursion.

▶ **Lemma 31.** *Let G be a graph with an edge clique cover $\mathcal{W}$. Any minimal triangulation H of G has a maximal clique Ω so that all blocks $C \in \mathcal{C}(G \setminus \Omega)$ have $|\mathcal{W}[C]| \leq |\mathcal{W}|/2$.*

Proof. Note that for any PMC Ω there can be at most one component $C \in \mathcal{C}(G \setminus \Omega)$ so that $|\mathcal{W}[C]| > |\mathcal{W}|/2$ because the sets $\mathcal{W}[C_i]$ over $C_i \in \mathcal{C}(G \setminus \Omega)$ correspond to disjoint subsets of $\mathcal{W}$. Let H be any minimal triangulation of G and pick arbitrary maximal clique Ω of H. While there is a component $C \in \mathcal{C}(G \setminus \Omega)$ such that $|\mathcal{W}[C]| > |\mathcal{W}|/2$, pick a maximal clique Ω of H such that $N(C) \subseteq \Omega \subseteq N[C]$. If this process stops, we have found the desired maximal clique Ω. Suppose the process does not stop. It considers an infinite sequence of blocks $C_1, C_2, \ldots$ with an associated infinite sequence of PMCs $\Omega_1, \Omega_2, \ldots$ with $C_i \in \mathcal{C}(G \setminus \Omega_i)$. Consider two consecutive blocks C_i and C_{i+1} in this sequence such that C_{i+1} is not a subset of C_i, which exist because G is finite. Recall that $N(C_i) \subseteq \Omega_{i+1} \subseteq N[C_i]$. Because C_{i+1} is not a subset of C_i, we have that $N(C_{i+1}) \subseteq N(C_i)$, implying that C_i and C_{i+1} are two distinct components of $N(C_i)$. Therefore the sets $\mathcal{W}[C_i]$ and $\mathcal{W}[C_{i+1}]$ are disjoint, implying that either of them has to be of size at most $|\mathcal{W}|/2$, which is a contradiction. ◀

We combine Proposition 30 and Lemma 31 into the following lemma.

▶ **Lemma 32** ($\star$). *Let G be a graph and $\mathcal{W}$ an edge clique cover of G. A graph H is a minimal triangulation of G if and only if (1) $V(H) = V(G)$ and (2) there is a PMC $\Omega \in \Pi(G)$ with $|\mathcal{W}[C_i]| \leq |\mathcal{W}|/2$ for all $C_i \in \mathcal{C}(G \setminus \Omega)$ and*

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

where H_i is a minimal triangulation of $R(C_i)$.

Algorithm 2 presents a polynomial space $O^*(9^{\mathsf{cc}'})$ time algorithm for treewidth. The algorithms for other problems are similar. The algorithm implements the characterization of Lemma 32, with the observation that $\mathcal{W}[C] \cup \{N(C)\}$ is an edge clique cover of $R(C)$.

The time complexity analysis of the algorithm reduces to a recursion equation resembling $t(\mathsf{cc}') = 9^{\mathsf{cc}'} + 3^{\mathsf{cc}'} 2t(\mathsf{cc}'/2)$, with some polynomial factors that get cleaned up at the end by the $O^*(\cdot)$ notation.

▶ **Lemma 33** ($\star$). *There is an algorithm that given a graph G with an edge clique cover of size cc' determines the treewidth of G in polynomial space and $O^*(9^{\mathsf{cc}'})$ time. If also a weight function $w : V(G)^2 \to \mathbb{R}_{\geq 0}$ is given, the algorithm determines the weighted minimum fill-in of G with respect to w. There is also algorithm that given a hypergraph $\mathcal{G}$ with m hyperedges determines its fractional hypertreewidth in polynomial space and $O^*(9^m)$ time.*

Algorithm 2 Treewidth in polynomial space and $O^*(9^{\mathsf{cc}'})$ time.

```
Input  : Connected graph $G$, an edge clique cover $\mathcal{W}$ of $G$, and an integer $k$
Output : Whether the treewidth of $G$ is at most $k$
1 for $\Omega \in \Pi(G)$ do
2     if $|\Omega| \leq k+1$ and $|\mathcal{W}[C_i]| \leq |\mathcal{W}|/2$ for all $C_i \in \mathcal{C}(G \setminus \Omega)$ then
3         ok ← True
4         for $C \in \mathcal{C}(G \setminus \Omega)$ do
5             if $\mathit{Treewidth}(R(C), \mathcal{W}[C] \cup \{N(C)\}, k) = \mathit{False}$ then
6                 ok ← False
7         if ok then return True
8 return False
```

The main idea to make the algorithm work when we do not know the edge clique cover is to just check if there are at most 3^{cc} PMCs. The reason why the time complexity becomes a bit higher than in Lemma 33 is that we cannot assume that the worst case of branching from a PMC Ω results in only two subproblems. In particular, we cannot assume anything better than cc subproblems each with a minimum edge clique cover of size $\mathsf{cc}/2+1$.

▶ **Lemma 34** ($\star$). *There is an algorithm that given a graph G and an integer cc uses polynomial space and $O^*(9^{\mathsf{cc}+O(\log^2 \mathsf{cc})})$ time and returns an integer t that is at least the treewidth of G. If also a weight function $w : V(G)^2 \to \mathbb{R}_{\geq 0}$ is given, the algorithm also returns a number f that is at least the weighted minimum fill-in of G with respect to w. If cc is at least the size of a minimum edge clique cover of G, then t is the treewidth of G and f is the weighted minimum fill-in of G with respect to w.*

7 Tightness

We show that the bounds $O(2^{\mathsf{cc}})$ and $O(3^{\mathsf{cc}})$ for the numbers of minimal separators and PMCs are tight, and that the parameter edge clique cover cannot be relaxed to vertex clique cover.

Let us construct a graph K_2^{cc}. Let $\mathcal{W}$ be a collection of size cc initially containing disjoint sets of size 1, i.e., $\mathcal{W} = \{W_1, \ldots, W_{\mathsf{cc}}\}$ with $W_i = \{v_i\}$. For each pair $1 \leq i < j \leq \mathsf{cc}$ we insert an element $v_{i,j}$ into the sets W_i and W_j. Now, the vertex set of K_2^{cc} is $V(K_2^{\mathsf{cc}}) = \bigcup_{W \in \mathcal{W}} W$ and the edge set of K_2^{cc} is $E(K_2^{\mathsf{cc}}) = \bigcup_{W \in \mathcal{W}} W^2$. The collection $\mathcal{W}$ is therefore an edge clique cover of K_2^{cc}.

▶ **Lemma 35.** *The graph K_2^{cc} has $\Theta(2^{\mathsf{cc}})$ minimal separators and $\Theta(3^{\mathsf{cc}})$ PMCs.*

Proof. Upper bounds follow from Theorem 1. For distinct subsets $\mathcal{W}' \subseteq \mathcal{W}$ the vertex sets $V(K_2^{\mathsf{cc}}, \mathcal{W}')$ are distinct because they can be identified by the inclusion of the v_i vertices. Let $\mathcal{W}'$ be any non-empty strict subset of $\mathcal{W}$. Note that $K_2^{\mathsf{cc}}[V(K_2^{\mathsf{cc}}, \mathcal{W}')]$ is connected, and therefore also $K_2^{\mathsf{cc}}[V(K_2^{\mathsf{cc}}, \mathcal{W} \setminus \mathcal{W}')]$ is connected. Any vertex in $V(K_2^{\mathsf{cc}}) \setminus V(K_2^{\mathsf{cc}}, \mathcal{W}', \mathcal{W} \setminus \mathcal{W}')$ is of type $v_{i,j}$ and has a neighbor in both $V(K_2^{\mathsf{cc}}, \mathcal{W}')$ and $V(K_2^{\mathsf{cc}}, \mathcal{W} \setminus \mathcal{W}')$. Therefore, $V(K_2^{\mathsf{cc}}) \setminus V(K_2^{\mathsf{cc}}, \mathcal{W}', \mathcal{W} \setminus \mathcal{W}')$ is a minimal separator and $V(K_2^{\mathsf{cc}}, \mathcal{W}')$ and $V(K_2^{\mathsf{cc}}, \mathcal{W} \setminus \mathcal{W}')$ are blocks of it. Therefore, the number of minimal separators of K_2^{cc} is at least the number of bipartitions of $\mathcal{W}$, i.e., $\Omega(2^{\mathsf{cc}})$.

Take any tripartition $\{\mathcal{W}_1, \mathcal{W}_2, \mathcal{W}_3\}$ of $\mathcal{W}$. Note that distinct tripartitions define distinct sets $V(K_2^{\mathsf{cc}}, \mathcal{W}_1, \mathcal{W}_2, \mathcal{W}_3)$. We show that $\Omega = V(K_2^{\mathsf{cc}}) \setminus (K_2^{\mathsf{cc}}, \mathcal{W}_1, \mathcal{W}_2, \mathcal{W}_3)$ is a PMC of K_2^{cc}. Any vertex in Ω is of type $v_{i,j}$, with vertices v_i and v_j in different blocks $V(K_2^{\mathsf{cc}}, \mathcal{W}_p)$.

Therefore, for any pair of vertices in Ω there is a common block that they are adjacent to, and therefore Ω satisfies the cliquish condition. A component $V(K_2^{\mathtt{cc}}, \mathcal{W}_1)$ cannot be full because there is a vertex $v_{i,j}$ that intersects cliques $W_i \in \mathcal{W}_2$ and $W_j \in \mathcal{W}_3$ and therefore is in the PMC but not in the neighborhood of $V(K_2^{\mathtt{cc}}, \mathcal{W}_1)$. Therefore, Ω satisfies the no full component condition. Therefore, the number of PMCs of $K_2^{\mathtt{cc}}$ is at least the number of tripartitions of $\mathcal{W}$, i.e., $\Omega(3^{\mathtt{cc}})$. ◀

We use a simpler construction to show that there cannot be FPT, or even XP, bounds for minimal separators and potential maximal cliques parameterized by the size of a minimum vertex clique cover, i.e., the number of cliques to cover all vertices of a graph.

▶ **Lemma 36.** *A graph that is constructed as a disjoint union of two cliques of size $n/2$ connected by a matching of $n/2$ edges has $\Omega(2^{n/2})$ minimal separators.*

Proof. All ways of selecting one endpoint of each edge of the matching result in selecting a minimal separator, expect the two ways that select one of the cliques. ◀

Note that $|\Pi(G)| \geq |\Delta(G)|/n$ [6], so the graphs of Lemma 36 have also an exponential number of PMCs.

8 Conclusion

We bounded the number of minimal separators and PMCs by the size of a minimum edge clique cover, obtaining new FPT algorithms for problems in the PMC framework. The parameterization by edge clique cover is motivated by real applications of optimal triangulations, and our results provide theoretical corroboration on the observations of the efficiency of the PMC framework in practice. Prior to our work, only the recent paper of Fomin et al. [15] considers FPT bounds for PMCs. Our work answers to their proposal for finding further FPT parameterizations for PMCs.

We omitted polynomial factors in our analysis, but the author believes that the polynomial factors are reasonably low and all of the algorithms that we gave can be implemented in a practical manner when $\mathtt{cc}$ is small. We also remark that the techniques introduced in this paper can be used to obtain $O^*(2^{O(m)})$ time algorithms for generalized hypertreewidth via the results of [30]. We focused on fractional hypertreewidth because it is a more general parameter [18], and its computation is simpler in the sense that $\text{FCOV}(\Omega)$ can be computed in polynomial time, unlike its counterpart in generalized hypertreewidth.

Our bounds and enumeration algorithms for minimal separators and PMCs are tight with respect to $\mathtt{cc}$ up to polynomial factors. Improving the algorithms for individual problems or proving (conditional) lower bounds remains as a problem for future work. Also, the question of finding further useful parameterizations of PMCs still remains for future work. Because of the naturality of $\mathtt{cc}$ in multiple settings related to optimal triangulations, we expect that more applications of our results could arise in future.

References

1 Anne Berry, Jean Paul Bordat, and Olivier Cogis. Generating all the minimal separators of a graph. *International Journal of Foundations of Computer Science*, 11(3):397–403, 2000. doi:10.1142/S0129054100000211.

2 Andreas Björklund, Thore Husfeldt, Petteri Kaski, and Mikko Koivisto. Fourier meets Möbius: fast subset convolution. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing*, pages 67–74. ACM, 2007. doi:10.1145/1250790.1250801.

3 Hans L Bodlaender, Leizhen Cai, Jianer Chen, Michael R. Fellows, Jan Arne Telle, and Dániel Marx. Open problems in parameterized and exact computation – IWPEC 2006. Technical Report UU-CS-2006-052, Department of Information and Computing Sciences, Utrecht University, 2006. URL: `https://dspace.library.uu.nl/handle/1874/22186`.

4 Magnus Bordewich, Katharina T. Huber, and Charles Semple. Identifying phylogenetic trees. *Discrete Mathematics*, 300(1-3):30–43, 2005. `doi:10.1016/j.disc.2005.06.015`.

5 Vincent Bouchitté and Ioan Todinca. Treewidth and minimum fill-in: Grouping the minimal separators. *SIAM Journal on Computing*, 31(1):212–232, 2001. `doi:10.1137/S0097539799359683`.

6 Vincent Bouchitté and Ioan Todinca. Listing all potential maximal cliques of a graph. *Theoretical Computer Science*, 276(1-2):17–32, 2002. `doi:10.1016/S0304-3975(01)00007-X`.

7 Mathieu Chapelle, Mathieu Liedloff, Ioan Todinca, and Yngve Villanger. Treewidth and pathwidth parameterized by the vertex cover number. *Discrete Applied Mathematics*, 216:114–129, 2017. `doi:10.1016/j.dam.2014.12.012`.

8 Cristina Conati, Abigail S. Gertner, Kurt VanLehn, and Marek J. Druzdzel. On-line student modeling for coached problem solving using Bayesian networks. In *User Modeling*, volume 383 of *CISM*, pages 231–242. Springer, 1997. `doi:10.1007/978-3-7091-2670-7_24`.

9 Marek Cygan, Marcin Pilipczuk, and Michal Pilipczuk. Known algorithms for edge clique cover are probably optimal. *SIAM Journal on Computing*, 45(1):67–83, 2016. `doi:10.1137/130947076`.

10 Holger Dell, Christian Komusiewicz, Nimrod Talmon, and Mathias Weller. The PACE 2017 parameterized algorithms and computational experiments challenge: The second iteration. In *12th International Symposium on Parameterized and Exact Computation*, volume 89 of *LIPIcs*, pages 30:1–30:12. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. `doi:10.4230/LIPIcs.IPEC.2017.30`.

11 Wolfgang Fischl, Georg Gottlob, Davide Mario Longo, and Reinhard Pichler. HyperBench: A benchmark and tool for hypergraphs and empirical findings. In *Proceedings of the 38th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, pages 464–480, 2019. `doi:10.1145/3294052.3319683`.

12 Wolfgang Fischl, Georg Gottlob, and Reinhard Pichler. General and fractional hypertree decompositions: Hard and easy cases. In *Proceedings of the 37th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, pages 17–32, 2018. `doi:10.1145/3196959.3196962`.

13 Fedor V. Fomin, Petr A. Golovach, and Jean-Florent Raymond. On the tractability of optimization problems on H-graphs. *Algorithmica*, 2020. `doi:10.1007/s00453-020-00692-9`.

14 Fedor V. Fomin, Dieter Kratsch, Ioan Todinca, and Yngve Villanger. Exact algorithms for treewidth and minimum fill-in. *SIAM Journal on Computing*, 38(3):1058–1079, 2008. `doi:10.1137/050643350`.

15 Fedor V. Fomin, Mathieu Liedloff, Pedro Montealegre, and Ioan Todinca. Algorithms parameterized by vertex cover and modular width, through potential maximal cliques. *Algorithmica*, 80(4):1146–1169, 2018. `doi:10.1007/s00453-017-0297-1`.

16 Fedor V. Fomin, Ioan Todinca, and Yngve Villanger. Large induced subgraphs via triangulations and CMSO. *SIAM Journal on Computing*, 44(1):54–87, 2015. `doi:10.1137/140964801`.

17 Masanobu Furuse and Koichi Yamazaki. A revisit of the scheme for computing treewidth and minimum fill-in. *Theoretical Computer Science*, 531:66–76, 2014. `doi:10.1016/j.tcs.2014.03.013`.

18 Martin Grohe and Dániel Marx. Constraint solving via fractional edge covers. *ACM Transactions on Algorithms*, 11(1):4:1–4:20, 2014. `doi:10.1145/2636918`.

19 Rob Gysel. Minimal triangulation algorithms for perfect phylogeny problems. In *Language and Automata Theory and Applications - 8th International Conference*, volume 8370 of *LNCS*, pages 421–432. Springer, 2014. `doi:10.1007/978-3-319-04921-2_34`.

20 Masami Hasegawa, Hirohisa Kishino, and Taka-aki Yano. Dating of the human-ape splitting by a molecular clock of mitochondrial dna. *Journal of Molecular Evolution*, 22(2):160–174, 1985.

21 Pinar Heggernes, Federico Mancini, Jesper Nederlof, and Yngve Villanger. A parameterized algorithm for chordal sandwich. In *Proceedings of 7th International Conference on Algorithms and Complexity*, volume 6078 of *LNCS*, pages 120–130. Springer, 2010. doi:10.1007/978-3-642-13073-1_12.

22 Finn V. Jensen and Frank Jensen. Optimal junction trees. In *Proceedings of the 10th Annual Conference on Uncertainty in Artificial Intelligence*, pages 360–366. Morgan Kaufmann, 1994. URL: https://dslpitt.org/uai/displayArticleDetails.jsp?mmnu=1&smnu=2&article_id=524&proceeding_id=10.

23 Finn V. Jensen and Thomas D. Nielsen. *Bayesian networks and decision graphs*. Springer, 2007. doi:10.1007/978-0-387-68282-2.

24 Sampath Kannan and Tandy Warnow. A fast algorithm for the computation and enumeration of perfect phylogenies. *SIAM Journal on Computing*, 26(6):1749–1763, 1997. doi:10.1137/S0097539794279067.

25 Tuukka Korhonen. Finding optimal tree decompositions. Master's thesis, University of Helsinki, 2020. URL: http://urn.fi/URN:NBN:fi:hulib-202006173010.

26 Tuukka Korhonen, Jeremias Berg, and Matti Järvisalo. Solving graph problems via potential maximal cliques: An experimental evaluation of the Bouchitté-Todinca algorithm. *ACM Journal of Experimental Algorithmics*, 24(1):1.9:1–1.9:19, 2019. doi:10.1145/3301297.

27 Tuukka Korhonen and Matti Järvisalo. Finding most compatible phylogenetic trees over multi-state characters. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, pages 1544–1551. AAAI Press, 2020. doi:10.1609/aaai.v34i02.5514.

28 Mathieu Liedloff, Pedro Montealegre, and Ioan Todinca. Beyond classes of graphs with "few" minimal separators: FPT results through potential maximal cliques. *Algorithmica*, 81(3):986–1005, 2019. doi:10.1007/s00453-018-0453-2.

29 Daniel Lokshtanov. On the complexity of computing treelength. *Discrete Applied Mathematics*, 158(7):820–827, 2010. doi:10.1016/j.dam.2009.10.007.

30 Lukas Moll, Siamak Tazari, and Marc Thurley. Computing hypergraph width measures exactly. *Information Processing Letters*, 112(6):238–242, 2012. doi:10.1016/j.ipl.2011.12.002.

31 Noam Ravid, Dori Medini, and Benny Kimelfeld. Ranked enumeration of minimal triangulations. In *Proceedings of the 38th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, pages 74–88. ACM, 2019. doi:10.1145/3294052.3319678.

32 Don Ringe, Tandy Warnow, and Ann Taylor. Indo-european and computational cladistics. *Transactions of the Philological Society*, 100(1):59–129, 2002. doi:10.1111/1467-968X.00091.

33 Marco Scutari. Learning Bayesian networks with the bnlearn R package. *Journal of Statistical Software*, 35(3):1–22, 2010. doi:10.18637/jss.v035.i03.

34 Charles Semple and Mike Steel. *Phylogenetics*. Oxford University Press, 2003.

35 Ken Takata. Space-optimal, backtracking algorithms to list the minimal vertex separators of a graph. *Discrete Applied Mathematics*, 158(15):1660–1667, 2010. doi:10.1016/j.dam.2010.05.013.

36 Hisao Tamaki. Computing treewidth via exact and heuristic lists of minimal separators. In *Proceedings of the Special Event on Analysis of Experimental Algorithms*, volume 11544 of *LNCS*, pages 219–236. Springer, 2019. doi:10.1007/978-3-030-34029-2_15.

37 Hisao Tamaki. Positive-instance driven dynamic programming for treewidth. *Journal of Combinatorial Optimization*, 37(4):1283–1311, 2019. doi:10.1007/s10878-018-0353-z.