Sparse induced subgraphs in P_6-free graphs*

Maria Chudnovsky† Rose McCarty‡ Marcin Pilipczuk§ Michał Pilipczuk¶
Paweł Rzążewski‖

Abstract

We prove that a number of computational problems that ask for the largest sparse induced subgraph satisfying some property definable in CMSO_2 logic, most notably Feedback Vertex Set, are polynomial-time solvable in the class of P_6-free graphs. This generalizes the work of Grzesik, Klimošová, Pilipczuk, and Pilipczuk on the Maximum Weight Independent Set problem in P_6-free graphs [SODA 2019, TALG 2022], and of Abrishami, Chudnovsky, Pilipczuk, Rzążewski, and Seymour on problems in P_5-free graphs [SODA 2021].

The key step is a new generalization of the framework of *potential maximal cliques*. We show that instead of listing a large family of potential maximal cliques, it is sufficient to only list their *carvers*: vertex sets that contain the same vertices from the sought solution and have similar separation properties.

*This research is a part of a project that has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme Grant Agreement 714704 (Rose, Marcin) and 948057 (Michał, Paweł). Maria is supported by NSF-EPSRC Grant DMS-2120644 and by AFOSR grant FA9550-22-1-008. Rose is also supported by NSF Grant DMS-2202961. Marcin is also partially funded by BARC, supported by the VILLUM Foundation grant 16582, and by Polish National Science Centre SONATA BIS-12 grant number 2022/46/E/ST6/00143.

†Department of Mathematics, Princeton University, USA

‡Department of Mathematics, Princeton University, USA and Institute of Informatics, University of Warsaw, Poland

§Institute of Informatics, University of Warsaw, Poland and IT University of Copenhagen, Denmark

¶Institute of Informatics, University of Warsaw, Poland

‖Warsaw University of Technology, Poland and Institute of Informatics, University of Warsaw, Poland

1 Introduction

The landmark work of Bouchitté and Todinca [6] uncovered the pivotal role that *potential maximal cliques* (PMCs for short) play in tractability of the classic Maximum (Weight) Independent Set problem (MIS or MWIS for short). The MIS (MWIS) problem asks for a set of pairwise nonadjacent vertices (called an *independent* set or a *stable* set) in a given graph of maximum possible cardinality (or weight, in the weighted setting, where every vertex is given a positive integral weight). Without giving a precise definition, a potential maximal clique is a set of vertices of the graph that can be seen as a "reasonable" choice for a bag in a tree decomposition of the graph, which in turn can be seen as a "reasonable" choice of a separating set for a divide-and-conquer algorithm.

Bouchitté and Todinca [6] showed that MWIS is solvable in time polynomial in the size of the graph and *the number of PMCs* of the input graph. At the time, this result unified a number of earlier tractability results for MWIS in various hereditary graph classes, giving an elegant common explanation for tractability. Later, Fomin, Todinca, and Villanger [9] showed that the same result applies not only to MWIS, but to a wide range of combinatorial problems, captured via the following formalism.

For a fixed integer k and a CMSO_2 formula[1] φ with one free vertex set variable, consider the following problem. Given a graph G, find a pair $(\mathtt{Sol}, X)$ maximizing $|X|$ such that $X \subseteq \mathtt{Sol} \subseteq V(G)$, $G[\mathtt{Sol}]$ has treewidth at most k, and $\varphi(X)$ is satisfied in $G[\mathtt{Sol}]$. This problem can also be considered in the weighted setting, where vertices of G have positive integral weights and we look for $(\mathtt{Sol}, X)$ maximizing the weight of X. For fixed k and φ, we denote this weighted problem as $(\mathrm{tw} \leqslant k, \varphi)$-MWIS.[2] Fomin, Todinca, and Villanger showed that $(\mathrm{tw} \leqslant k, \varphi)$-MWIS is solvable in time polynomial in the size of the graph and the number of its PMCs. Clearly, MWIS can be expressed as a $(\mathrm{tw} \leqslant 0, \varphi)$-MWIS problem. Among the many problems captured by this formalism, we mention that Feedback Vertex Set can be expressed as a $(\mathrm{tw} \leqslant 1, \varphi)$-MWIS problem: indeed, the complement of a minimum (weight) feedback vertex set is a maximum (weight) induced forest.

Another application is as follows. Let $\mathcal{G}$ be a minor-closed graph class that does not contain all planar graphs. Thanks to the Graph Minor Theorem of Robertson and Seymour [23], there exists a finite set $\mathcal{F}$ of graphs such that $G \in \mathcal{G}$ if and only if G does not contain any graph of $\mathcal{F}$ as a minor. Consequently, the property of belonging to $\mathcal{G}$ can be expressed in CMSO_2. Furthermore, as $\mathcal{G}$ does not contain all planar graphs, $\mathcal{G}$ is of bounded treewidth [22]. Thus the problem of finding a largest induced subgraph that belongs to $\mathcal{G}$ is a special case of $(\mathrm{tw} \leqslant k, \varphi)$-MWIS.

In both applications above we have $\mathtt{Sol} = X$. To see an example where these sets are different, consider the problem of packing the maximum number of vertex-disjoint and pairwise non-adjacent induced cycles. To see that it is also a special case of $(\mathrm{tw} \leqslant k, \varphi)$-MWIS, let $k = 2$ and φ be the formula enforcing that $G[\mathtt{Sol}]$ is 2-regular (i.e., a collection of cycles) and no two vertices from X are in the same component of $G[\mathtt{Sol}]$.

Unfortunately, the results of [6] and [9] do not cover all cases where we expect even the original MWIS problem to be polynomial-time solvable. A key case arises from excluding an induced path. For a fixed graph H, the class of *H-free* graphs consists of all graphs that do not contain H as an induced subgraph. For an integer t, we denote the path on t vertices by P_t. While the class of P_4-free graphs has bounded clique-width, the class of P_t-free graphs does not seem to exhibit any apparent structure for $t \geqslant 5$. Still, as observed by Alekseev [2,3], MWIS is not known to be NP-hard in P_t-free graphs for any fixed t.

[1] CMSO_2 stands for monadic second-order logic in graphs with quantification over edge subsets and modular counting predicates. In this logic, one can quantify both over single vertices and edges and over their subsets, check membership and vertex-edge incidence, and apply modular counting predicates with fixed moduli to set variables. See Section 2.1 for a formal introduction of the syntax and semantics of CMSO_2.

[2] Here, MWIS stands for "maximum weight induced subgraph."

At first glance, the PMC framework of [6] does not seem applicable to P_t-free graphs for $t \geqslant 5$, as even co-bipartite graphs can have exponentially-many PMCs. (A graph is *co-bipartite* if its complement is bipartite; these graphs are P_5-free.) In 2014, Lokshtanov, Vatshelle, and Villanger [17] revisited the framework of Bouchitté and Todinca and showed that it is not necessary to use *all* PMCs of the input graph, but only some carefully selected subfamily of PMCs. They also showed that for P_5-free graphs, one can efficiently enumerate a suitable family of polynomial size, thus proving tractability of MWIS in P_5-free graphs. The arguments of [17] were then expanded to P_6-free graphs by Grzesik et al. [13]. The case of P_7-free graphs remains open.

A general belief is that the MWIS problem is actually tractable in P_t-free graphs for any constant t. This belief is supported by the existence of *quasi-polynomial-time* algorithms that work for every t [10,21]. Extending these results, Gartland et al. [11] proved that for every t, k, and φ, the $(\mathrm{tw} \leqslant k, \varphi)$-MWIS problem is solvable in quasi-polynomial time on P_t-free graphs via a relatively simple branching algorithm. Actually, their algorithm solves the $(\deg \leqslant k, \varphi)$-MWIS problem, where instead of a subgraph of bounded treewidth we ask for a subgraph of bounded *degeneracy*. We remark that $(\mathrm{tw} \leqslant k, \varphi)$-MWIS can be expressed as $(\deg \leqslant k', \varphi')$-MWIS. Indeed, degeneracy is always upper-bounded by treewidth and the property of being of bounded treewidth is expressible by a CMSO_2 formula. On the other hand, the language of $(\deg \leqslant k, \varphi)$-MWIS allows us to capture more problems. One well-known example is Vertex Planarization [15,20], which asks for a maximum (or maximum weight) induced planar subgraph. Indeed, planar graphs have degeneracy at most 5, but they might have unbounded treewidth, so Vertex Planarization is not a special case of $(\mathrm{tw} \leqslant k, \varphi)$-MWIS. However, Gartland et al. [11] showed that in (a superclass of) P_t-free graphs, treewidth and degeneracy are functionally equivalent. Consequently, even though $(\deg \leqslant k, \varphi)$-MWIS is more general than $(\mathrm{tw} \leqslant k, \varphi)$-MWIS, in P_t-free graphs both formalisms describe the same family of problems.

One of the obstructions towards extending the known polynomial-time algorithms for MWIS beyond P_5-free and P_6-free graphs is the technical complexity of the method. The algorithm for P_5-free graphs [17] is already fairly involved, and the generalization to P_6-free graphs [13] resulted in another significant increase in the amount of technical work. In particular, it is not clear how to apply such an approach to solve $(\mathrm{tw} \leqslant k, \varphi)$-MWIS (or, equivalently, $(\deg \leqslant k, \varphi)$-MWIS). Furthermore, in a recent note [12], the authors of [13] discuss limitations of applying the method to solving MWIS in graph classes excluding longer paths.

Both algorithms for P_5-free [17] and for P_6-free graphs [13] focused on restricting the family of needed PMCs, but the algorithms still listed the PMCs exactly. A major twist was made by Abrishami at al. [1] who showed that instead of determining a PMC exactly, it suffices to find only a *container* for it: a superset that does not contain any extra vertices from the solution. They also showed that with this container method, the arguments for P_5-free graphs from [17] greatly simplify to some elegant structural observations about P_5-free graphs. Moreover, the container method from [1] works with any $(\deg \leqslant k, \varphi)$-MWIS problem. In particular, the authors of [1] showed that Feedback Vertex Set is polynomial-time solvable in P_5-free graphs.

While the container method of [1] pushed the boundary of tractability, it does not seem to be easily applicable to P_t-free graphs for $t \geqslant 6$; in particular, we do not know how to significantly simplify the arguments of [13] using containers.

Our contribution

Our contribution is three-fold.

Identifying treedepth as the relevant width measure. Previous work on MWIS in P_5-free and P_6-free graphs [12,17] first fixed a sought solution I (which is an inclusion-wise maximal independent set), then observed that it suffices to focus on PMCs that contain at most one vertex from the fixed solution I. They also distinguished between PMCs that have one vertex in

common with I, and PMCs that have zero. The former ones turn out to be easy to handle, but the latter ones, called "I-free" or "I-safe", are trickier; to tackle them one has to rely on some additional properties stemming from the fact that I is maximal.

We introduce generalizations of these notions to induced subgraphs of bounded *treedepth*, a structural notion more restrictive than treewidth. It turns out that the correct analog of independent sets are induced subgraphs of bounded treedepth with a fixed elimination forest. Maximality corresponds to the inability to extend the subgraph by adding a leaf vertex to the elimination forest, while I-freeness corresponds to not containing any *leaf* of the fixed elimination forest of the sought solution.

Focusing on treedepth naturally leads us to the $(\mathrm{td} \leqslant k, \varphi)$-MWIS problem, where $G[\texttt{Sol}]$ is required to have treedepth at most d. Luckily, in P_t-free graphs treedepth is functionally equivalent to treewidth (and thus to degeneracy, too), so in this setting the $(\mathrm{td} \leqslant k, \varphi)$-MWIS, $(\mathrm{tw} \leqslant k, \varphi)$-MWIS, and $(\deg \leqslant k, \varphi)$-MWIS formalisms define the same class of problems.

While simple in their form and proofs, the above generalizations allow us to adapt many arguments of [12, 17] to all $(\mathrm{td} \leqslant k, \varphi)$-MWIS problems.

Generalizing containers to carvers. We introduce a notion of a *carver* that generalizes containers. Our inspiration comes from thinking of a PMC as a "reasonable" separation in a divide-and-conquer algorithm. Instead of determining a PMC exactly, we want to find an "approximation" that, on one hand, contains the same vertices from the sought solution, and, on the other hand, *splits the graph at least as well as the PMC*. The crux lies in properly defining this latter notion.

Note that a container should satisfy any reasonable definition of "splitting at least as well;" if X is a set of vertices which contains a PMC Ω, then each component of $G - X$ is a subset of a component of $G - \Omega$. However, if we allow that the approximation X of Ω does not contain some vertices of Ω, then we need to somehow restrict the way the vertices of $\Omega \setminus X$ connect the components of $G - (\Omega \cup X)$. The first natural idea, to ask that no component of $G - X$ intersects more than one component of $G - \Omega$, turns out to be not very useful. The actual definition allows $\Omega \setminus X$ to glue up some components of $G - (\Omega \cup X)$ as long as we can show that another carver, for a different PMC, will later separate them.

We prove that carvers are sufficient to solve all problems of our interest in P_t-free graphs.

Theorem 1.1 (informal statement of Theorem 3.2)**.** *Any* $(\deg \leqslant k, \varphi)$-MWIS *problem is solvable on P_t-free graphs in time polynomial in the size of the input graph and the size of the supplied carver family.*

Finding carvers in P_6-free graphs. We showcase the strength of Theorem 1.1 by lifting the approach of Grzesik et al. [13] from just MWIS to arbitrary $(\deg \leqslant k, \varphi)$-MWIS problems on P_6-free graphs. Formally, we prove the following.

Theorem 1.2. *For any choice of k and φ, the* $(\deg \leqslant k, \varphi)$-MWIS *problem is polynomial-time solvable on P_6-free graphs.*

Note that Theorem 1.2 in particular implies that Feedback Vertex Set is polynomial-time solvable on P_6-free graphs, which was a well-known open problem [5, 18, 19]. Apart from being applicable to a wider class of problems, our carver-based approach also significantly simplifies, or even makes obsolete, many of the technical parts of [13].

On high level, the proof of [13] consists of two parts. In the first part, PMCs that in some sense "have more than two principal components" are analysed. Here, the arguments are arguably neat and elegant in many places. The second part deals with PMCs with exactly two "principal components", that can chain up into long sequences. Here, a highly technical replacement argument is developed to "canonize" an I-free minimal chordal completion in such parts of the input graph.

Using the newly developed notions of treedepth structures, we lift the (more elegant part of the) arguments of [13] to (td $\leqslant k, \varphi$)-MWIS problems, showing that PMCs with "more than two principal components" admit containers, not only carvers. Furthermore, we use the power of the new notion of the carver to construct carvers for PMCs with two "prinicipal components", replacing the highly technical part of [13] with arguably shorter and more direct arguments.

Technical overview

Let us now have a closer look at the three aforementioned contributions.

To this end, we need to introduce some definitions regarding chordal completions and PMCs. Given a graph G, a set $\Omega \subseteq V(G)$ is a *potential maximal clique* (or a *PMC*) if there exists a minimal chordal completion of G in which Ω is a maximal clique. A *chordal completion* of G is a supergraph of G which is chordal and has the same vertex-set as G; it is *minimal* if it has no proper subgraph which is also a chordal completion of G. (Recall that a graph is *chordal* if it has no holes, where a *hole* is an induced cycle of length at least 4.) Since chordal completions are obtained by adding edges to G, it is convenient to write them as $G + F$, where F is a set of non-edges of G.

Chordal completions in a certain sense correspond to tree decompositions, and it is often more convenient to work with the latter. (The formal definition of a tree decomposition can be found in Section 2.) It is a folklore result that a graph H is chordal if and only if it has a tree decomposition whose bags are exactly the maximal cliques of H (meaning, in particular, that the number of nodes of the tree is equal to the number of maximal cliques of H). Such a tree decomposition is called a *clique tree* of H; note that while the set of bags of a clique tree is defined uniquely, the actual tree part of the tree decomposition is not necessarily unique.

In the other direction, observe that if we have a tree decomposition of a given graph G, then by completing every bag of this tree decomposition into a clique, we obtain a chordal supergraph. Hence, minimal chordal completions correspond to "the most refined" tree decompositions of G, and this supports the intuition that PMCs are "reasonable" choices of bags in a tree decomposition of G.

For a set $S \subseteq V(G)$ in a graph G, a *full component* of S is a connected component A of $G - S$ such that $N(A) = S$. A set S is a *minimal separator* if S has at least two full components. It is well-known (cf. [6]) that if Ω is a PMC in G, then for every component D of $G - \Omega$, $N(D)$ is a minimal separator with D as a full component and another full component containing $\Omega \setminus N(D)$. Furthermore, if st is an edge of T for a clique tree (T, β) of a minimal chordal completion $G + F$, then $\beta(s) \cap \beta(t)$ is a minimal separator with one full component containing $\beta(s) \setminus \beta(t)$ and one full component containing $\beta(t) \setminus \beta(s)$. Thus, in some sense, minimal separators are building blocks from which PMCs are constructed. While PMCs correspond to bags of tree decompositions of G, minimal separators correspond to *adhesions* (intersections of neighboring bags).

Treedepth structures. The starting insight of Lokshtanov, Vatshelle, and Villanger [17] is that if I is a maximal independent set in G, then by completing $V(G) \setminus I$ into a clique we obtain a chordal graph (even a split graph), and thus there exists a minimal chordal completion $G + F$ that does not add any edge incident to I; we call such a chordal completion *I-free*. In $G + F$, every maximal clique contains at most one vertex of I and, if $I \cap \Omega = \{v\}$ for a maximal clique Ω, then $\Omega \subseteq N_G[v]$ and $N_G[v]$ is a good container for Ω. They argue that it is sufficient to list a superset of all maximal cliques of $G + F$, and hence it suffices to focus on PMCs of G that are disjoint from the sought solution I. Such PMCs are henceforth called *I-free*.

Let Ω be an I-free PMC. Since I is maximal, every $v \in \Omega$ has a neighbor in I that is outside Ω, as Ω is I-free. The existence of such neighbors is pivotal to a number of proofs of [13, 17].

To discuss our generalization to induced subgraphs of bounded treedepth, we need a few standard definitions. A *rooted forest* is a forest $\mathcal{T}$ where each component has exactly one

specified vertex called its *root*. The *depth* of a vertex $v \in V(\mathcal{T})$ is the number of vertices in the unique path from v to a root (so roots have depth 1). The *height* of $\mathcal{T}$ is the maximum depth of any of its vertices. A path in $\mathcal{T}$ is *vertical* if one of its ends is an ancestor of the other. (We consider each vertex to be both an ancestor and a descendent of itself.) Two vertices are *$\mathcal{T}$-comparable* if they are connected by a vertical path; otherwise they are *$\mathcal{T}$-incomparable*. An *elimination forest* of a graph G is a rooted forest $\mathcal{T}$ such that $V(\mathcal{T}) = V(G)$ and the endpoints of each edge of G are $\mathcal{T}$-comparable. The *treedepth* of G is then the smallest integer d such that G has an elimination forest of height d.

Let us now move to the new definitions. Let G be a graph and d be a positive integer. A *treedepth-d structure in G* is a rooted forest $\mathcal{T}$ of height at most d such that $V(\mathcal{T})$ is a subset of $V(G)$ and $\mathcal{T}$ is an elimination forest of the subgraph of G induced by $V(\mathcal{T})$. We say that $\mathcal{T}$ is *maximal* if there is no treedepth-d structure $\mathcal{T}'$ in G such that $\mathcal{T}$ is a proper induced subgraph of $\mathcal{T}'$ and every root of $\mathcal{T}$ is a root of $\mathcal{T}'$. In other words, $\mathcal{T}$ is maximal if one cannot extend it by appending a leaf while preserving the bound on the height.

Note that if H is a maximal induced subgraph of G of treedepth at most d, and $\mathcal{T}$ is a height-d elimination forest of that subgraph, then $\mathcal{T}$ is a maximal treedepth-d structure in G. Consequently, in the context of $(\mathrm{td} \leqslant d, \varphi)$-MWIS, we can consider $\mathtt{Sol}$ as being in fact a maximal set inducing a subgraph of treedepth at most d in G; if $(\mathtt{Sol}, X)$ is an actual solution, then there exists a maximal treedepth-d structure $\mathtt{Sol}'$ that is a superset of $\mathtt{Sol}$, and we can extend φ by saying that there exists a set $\mathtt{Sol} \subseteq \mathtt{Sol}'$ with all the desired properties. Thus, most of the structural results in this work consider the set of all maximal treedepth-d-structures, which are more detailed versions of maximal sets inducing a subgraph of treedepth at most d.

Recall that for any independent set I, there is a minimal chordal completion of G that is I-free, that is, it does not add any edge incident to I. This statement generalizes to chordal completions *aligned* with a given treedepth-d structure $\mathcal{T}$; we say that a chordal completion $G + F$ is *$\mathcal{T}$-aligned* if F does not contain any pair uv such that

(i) u or v is a depth-d vertex of $\mathcal{T}$, or

(ii) u and v are vertices of $\mathcal{T}$ which are $\mathcal{T}$-incomparable.

The second condition equivalently says that $\mathcal{T}$ is a treedepth-d structure in $G + F$.

We show that there is always a $\mathcal{T}$-aligned minimal chordal completion (see Lemma 2.11) and argue that it suffices to focus on PMCs that come from an aligned minimal chordal completion.

The analog of the notion of "I-freeness" is as follows: A PMC Ω is *$\mathcal{T}$-avoiding* if it is a maximal clique of a minimal chordal completion that is $\mathcal{T}$-aligned, and it does not contain any depth-d vertex of $\mathcal{T}$. Similarly as in the case of PMCs that are not I-free, if Ω is $\mathcal{T}$-aligned but not $\mathcal{T}$-avoiding, it contains exactly one vertex of $\mathcal{T}$ of depth d and one can argue that the closed neighborhood of such vertex gives raise to a container for Ω (after excluding the vertices of $\mathcal{T} \setminus \Omega$ that accidentally got into it, but there are at most $d-1$ one of them, because they all are ancestors of the guessed vertex of $\mathcal{T} \cap \Omega$ of depth d in the rooted forest $\mathcal{T}$).

Thus, it remains to focus on $\mathcal{T}$-avoiding PMCs. In the I-free setting, the important property of an I-free PMC was that every $v \in \Omega$ has a neighbor in I. Here, one can argue that every $v \in \Omega \setminus \mathcal{T}$ in a $\mathcal{T}$-avoiding PMC Ω has a neighbor in $\mathcal{T} \setminus \Omega$, as otherwise it can be added to $\mathcal{T}$ without increasing the maximum depth of $\mathcal{T}$, contradicting the maximality of $\mathcal{T}$.

This concludes the overview of the adaptation of the notion of I-freeness to induced subgraphs of bounded treedepth.

Carvers. Let G be a graph and let I be an optimal solution to MWIS in G. Assume that we are given a polynomial-sized family $\mathcal{F}$ of PMCs in G that contains all maximal cliques of some I-free minimal chordal completion $G + F$ of G. The crucial insight of [17] is that this is enough to solve MWIS in G in polynomial time by a dynamic programming algorithm. The algorithm considers the following set of states: for every $\Omega \in \mathcal{F}$, every $J \subseteq \Omega$ of size at most 1,

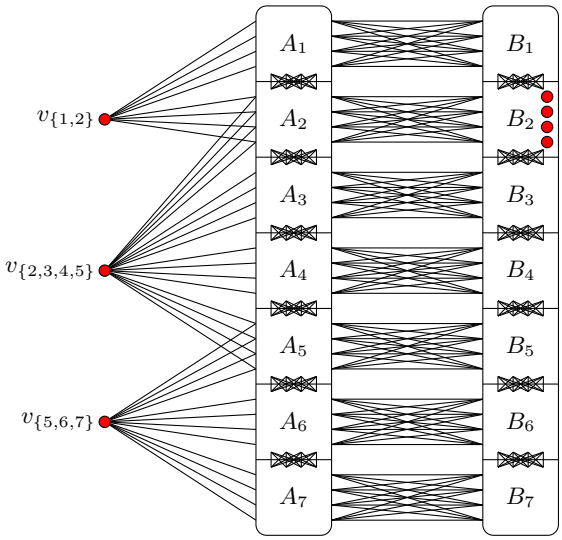

Figure 1: An example of a P_6-free graph with a maximal independent set where a weak container seems to be a too restrictive notion. The red vertices are the vertices of a maximal independent set. Here, $n = 7$, $i_0 = 2$, and $\mathcal{F} = \{\{1,2\},\{2,3,4,5\},\{5,6,7\}\}$.

and every component D of $G-\Omega$, it tries to compute the best possible independent set $I[\Omega, J, D]$ in $G[\Omega \cup D]$ with $I[\Omega, J, D] \cap \Omega = J$. The assumption that $\mathcal{F}$ contains all PMCs of $G+F$ allows one to argue that there is a computation path of this dynamic programming algorithm that finds an independent set that is at least as good as I (we may not find I itself).

The crucial insight of [1] is that for the dynamic programming algorithm to work, it is enough to know *containers* for the maximal cliques of $G+F$, that is, it is fine if the provided sets in $\mathcal{F}$ are larger, as long as they do not contain extra vertices from the sought solution. The intuition here is that the dynamic programming algorithm relies on the separation properties of PMCs as bags of a clique tree of $G+F$, and a superset is an even better separator than a PMC itself.

From the point of view of separation, the following relaxation of a container would suffice. A set X is a *weak container* of a PMC Ω if it contains the same vertices from the sought solution and every connected component of $G-X$ intersects at most one connected component of $G-\Omega$ (that is, the vertices of $\Omega \setminus X$ do not connect two components of $G-(\Omega \cup X)$).

However, in the context of P_6-free graphs, we are unable to provide even weak containers to some PMCs, and there seems to be a good reason for this failure. Namely, there are examples of P_6-free graphs G with an (I-free or $\mathcal{T}$-avoiding, depending on the problem we are solving) PMC Ω with a subset $\mathcal{D}$ of components of $G-\Omega$ such that some local modifications to the minimal chordal completion $G+F$ modify Ω slightly, but completely reshuffle the vertices of $\mathcal{D}$ into new components. The intuition is that the dynamic programming algorithm should not attempt to separate $\mathcal{D}$ into components while looking at a (weak) container of Ω, but while looking at another PMC Ω' that is "closer" to $\mathcal{D}$.

More precisely, consider the following example (cf. Figure 1). Let $A_1, \ldots, A_n$ and $B_1, \ldots, B_n$ be two sequences of P_6-free graphs and let $\mathcal{F}$ be a family of subsets of $[n]$ of size being a large polynomial in n with $\bigcup \mathcal{F} = [n]$; all subsets of $[n]$ of size at most C for a large constant C would do the job. Construct a graph G as follows. Start with a disjoint union of $A_1, \ldots, A_n$ and $B_1, \ldots, B_n$. For every $i, j \in [n]$, $i \neq j$, add all edges between A_i and A_j and all edges between B_i and B_j. For every $i \in [n]$, add all edges between A_i and B_i. Finally, for every $K \in \mathcal{F}$, introduce a vertex v_K and make it adjacent to $\bigcup_{i \in K} A_i$. A direct check shows that G is P_6-free, for every choice of $i_0 \in [n]$ and a maximal independent set I_0 in B_{i_0}, the set $I_{i_0,I_0} := I_0 \cup \{v_K \mid K \in \mathcal{F}\}$ is a maximal independent set in G, and, for every $\emptyset \neq J \subsetneq [n]$ that is not contained in any

set of $\mathcal{F}$, the set $S_J := \bigcup_{i\in J} A_i \cup \bigcup_{i\in[n]\setminus J} B_i$ is a minimal separator with one full component $B_J := \bigcup_{i\in J} B_i$ and a second full component $A_J := \bigcup_{i\in[n]\setminus J} A_i \cup \{v_K \mid K \in \mathcal{F}, K \not\subseteq J\}$, and a number of single-vertex components $\{v_K\}$ for $K \in \mathcal{F}, K \subseteq J$. Observe that a weak container for S_J should separate v_K for $K \subseteq J$ from those v_K for which $K \not\subseteq J$. The only way to make a small family of (weak) containers for all such separators S_J is to make containers containing whole $\bigcup_{i\in I} A_i$ but none of the vertices v_K; however, distinguishing $\bigcup_{i\in I} A_i$ and $\{v_K \mid K \in \mathcal{F}\}$ seems difficult using the toolbox used in [13, 17].

In the above example a chordal completion will turn every A_i for $i \in [n]$ and every B_i for $i \in [n] \setminus \{i_0\}$ into a clique, and take any permutation π of $[n]$ with $\pi(1) = i_0$ and add edges between $A_{\pi(i)}$ and $B_{\pi(j)}$ for every $1 \leqslant i < j \leqslant n$. This corresponds to turning S_J for $J = \pi(\{1, 2, \ldots, i\})$ for every $1 \leqslant i \leqslant n$ into a clique. Intuitively, the algorithm should not bother with the choice of π, which corresponds to ignoring how vertices v_K are separated while looking at intermediate separators S_J.

Recall that the correctness of the dynamic programming algorithm of [17] relies on the observation that a clique tree of $G + F$ provides a computation path in which the algorithm finds a solution at least as good as I. To provide an analogous proof in our setting, one needs to confine such problematic set $\mathcal{D}$ in one subtree of a clique tree of $G + F$. This consideration brings us to the final definition of a carver.

Definition 1.3. Let G be a graph and d and k be positive integers. A family $\mathcal{C} \subseteq 2^{V(G)}$ is a *tree-depth-d carver family* of *defect k* in G if for every treedepth-d structure $\mathcal{T}$ in G, there exists a tree decomposition (T, β) of G such that for each $t \in V(T)$ there exists $C \in \mathcal{C}$ such that

(i) $C \cap \mathcal{T}$ contains $\beta(t) \cap \mathcal{T}$ and has size at most k, and

(ii) each component of $G - C$ is contained in $\beta(t) \cup \bigcup_{s\in T'} \beta(s)$ for some component T' of $T - \{t\}$.

Such a set C as above is called a $(\mathcal{T}, (T, \beta))$-*carver* for $\beta(t)$ of defect k; it might not be unique. We use this definition independently of that of carver families.

We prove that this definition works as intended: a tree-depth-d carver family of small defect in G is enough to design a dynamic programming routine that solves the $(\mathrm{td} \leqslant d, \varphi)$-MWIS problem on G.

Theorem 1.4. *For any positive integers d and k and any CMSO_2 formula φ, there exists an algorithm that, given a vertex-weighted graph G and a tree-depth-d carver family $\mathcal{C} \subseteq 2^{V(G)}$ of defect k in G, runs in time polynomial in the input size and either outputs an optimal solution to the $(\mathrm{td} \leqslant d, \varphi)$-MWIS problem on G, or determines that no feasible solution exists.*

We remark that the proof of Theorem 3.2 is far from being just an involved verification of a natural approach. There is a significant technical hurdle coming from the fact that, with fixed $\mathcal{T}$ and (T, β), carvers for neighboring bags of (T, β) may greatly differ from each other in terms of the amount of non-solution vertices added to them. One needs to design careful tie-breaking schemes for choices in partial solutions in the dynamic programming algorithm in order to avoid conflicting tie-breaking decisions made while looking at different carvers.

Application to P_6-free graphs. The starting point of the work of [13] on MWIS in P_6-free graph is an analysis of minimal separators that identifies a crucial case distinction between full components of a minimal separator, into ones whose complement is disconnected (so-called *mesh components*) or connected (*non-mesh components*). The analysis splits minimal separators in a P_6-free graph G into three categories:

Simple, being a proper subset of another minimal separator, having more than two full components, or having two non-mesh full components. Here, one can enumerate a polynomial-sized family of candidates that contains all such separators.

Somewhat complicated, having exactly two full components, both being mesh. Here, one can enumerate a polynomial-sized family that contains a "weak container" for every such separator, which is equally good for our applications as just knowing the separator exactly.

Really complicated, having exactly two full components, one mesh and one non-mesh. Here, we can only enumerate a polynomial-sized family of "semi-carvers" that separate the mesh component from the other components, but such a semi-carver is not guaranteed to separate the non-mesh full component from some non-full components. (This weakness corresponds to examples mentioned earlier about inability to split some family $\mathcal{D}$ of components of $G - \Omega$ for a PMC Ω; note that all components S_J in the aforementioned examples are of the really complicated type.)

This analysis generalizes to our setting, using the new notions of treedepth structures.

We proceed to discussing the PMCs. Then, the following case distinction is identified in [13]. A potential maximal clique Ω in a graph G is *two-sided* if there exist two distinct connected components D_1, D_2 of $G - \Omega$ such that for every connected component D of $G - \Omega$, we have $N(D) \subseteq N(D_1)$ or $N(D) \subseteq N(D_2)$.

The following statement has been essentially proven in [13]. However, it has been proven only with the Max Weight Independent Set problem in mind, so we need to adjust the argumentation using the notion of treedepth structures.

Theorem 1.5. *For every positive integer d there exists a polynomial-time algorithm that, given a P_6-free graph G outputs a family $\mathcal{C} \subseteq 2^{V(G)}$ with the following guarantee: for every maximal treedepth-d structure $\mathcal{T}$ in G and every potential maximal clique Ω of G that is $\mathcal{T}$-avoiding and not two-sided, there exists $C \in \mathcal{C}$ that is a container for Ω, i.e., $\Omega \subseteq C$ and $C \cap V(\mathcal{T}) = \Omega \cap V(\mathcal{T})$.*

It remains to study two-sided PMCs, which were the main cause of technical hurdles in [13]. Here we depart from the approach of [13] and use the power of carvers instead.

To use carvers, we would like to choose not only a minimal chordal completion $G+F$ (which, following the developments in the first part of our work, would be any $\mathcal{T}$-aligned minimal chordal completion, where $\mathcal{T}$ is the sought solution) but also a clique tree (T, β) of $G + F$. Recall that adhesions in (T, β) correspond to minimal separators in G, and the really complicated minimal separators are the ones with one mesh and one non-mesh full component, in which case it is difficult to isolate the non-mesh component. So, we would like the clique tree (T, β) to be imbalanced in the following way: if $st \in E(T)$ is such that $\beta(s) \cap \beta(t)$ is a really complicated minimal separator with the non-mesh full component A_s containing $\beta(s) \setminus \beta(t)$ and the mesh full component A_t containing $\beta(t) \setminus \beta(s)$, then as much as possible of the decomposition (T, β) should be reattached to the component of $T - \{st\}$ that contains s.

More precisely, for a clique tree (T, β) of $G + F$, for every edge st as above, orient st from t to s (and keep all edges of T that do not correspond to really complicated minimal separators undirected). Consider now an edge st as above and assume that there exists $s' \in N_T(t)$, $s \neq s'$ such that

$$\beta(s') \cap \beta(t) \subseteq \beta(s) \cap \beta(t). \tag{1}$$

Then, the minimal separator $\beta(s') \cap \beta(t)$ is a simple one (it is contained in another minimal separator $\beta(s) \cap \beta(t)$), so the edge $s't$ is undirected. Observe that the assumption (1) allows the following modification of (T, β): replace the edge $s't$ with an edge $s's$. This modification corresponds to the intuition that while studying the really complicated minimal separator $\beta(s) \cap \beta(t)$, it is difficult to separate the component A_s from the full component of $G - (\beta(s') \cap \beta(t))$ that contains $\beta(s') \setminus \beta(t)$, and thus — from the point of view of the PMC $\beta(t)$ — both these components should be contained in bags of the same component of $T - \{t\}$.

A simple potential argument shows that such modifications cannot loop indefinitely and there exists a clique tree (T, β) where no modification is possible. This is the clique tree for which we are finally able to construct carvers using the aforementioned analysis of minimal separators, in

particular semi-carvers for the really difficult minimal separators. The actual construction is far from straightforward, but arguably simpler than the corresponding argumentation of [13] that handles two-sided PMCs.

Organization

After the preliminaries (Section 2), we introduce the notion of carvers and carver families and provide the main algorithmic engine in Section 3. The remaining sections are devoted to P_6-free graphs and the proof of Theorem 1.2. Sections 4 and 5 study approximate guessing of minimal separators. Section 6 recalls the main (and most elegant) structural results of P_6-free graphs of [13], essentially extracting from [13] a family of containers for all PMCs that in some sense have "more than two sides." Section 7 uses the results for minimal separators of Sections 4 and 5 to provide carvers for the remaining PMCs; this is the place where we crucially rely on the fact that we want to provide only carvers, not containers. Finally, Section 8 wraps up the proof of Theorem 1.2, and Section 9 gives a concluding remark about P_7-free graphs.

2 Preliminaries

We use standard graph-theoretic notation, and all graphs are simple, loopless, and finite. We consider the edge-set of a graph G to be a subset of $\binom{V(G)}{2}$, which is the set of all 2-element subsets of $V(G)$. We write uv for an element $\{u, v\}$ of $\binom{V(G)}{2}$. A *non-edge* of G is then a pair of vertices uv which is not in $E(G)$. Given a set $F \subseteq \binom{V(G)}{2}$, we write $G + F$ for the graph with vertex set $V(G)$ and edge set $E(G) \cup F$; so $G + F$ is obtained from G by adding all pairs from F as edges if they were not already present.

Given a graph G and a set of vertices $S \subseteq V(G)$, we write $N(S)$ and $N[S]$, respectively, for the open and closed neighborhood of S in G. That is, $N(S) \coloneqq \{u \in V(G) - S : uv \in E(G)$ for some $v \in S\}$ and $N[S] \coloneqq S \cup N(S)$. We do not distinguish between induced subgraphs and their vertex sets, except when it might cause confusion. So we typically use S and $G[S]$ interchangeably. Finally, if $v_0, v_1, \ldots, v_k$ are distinct vertices of G, then we write $N(v_0, v_1, \ldots, v_k)$ for $N(\{v_0, v_1, \ldots, v_k\})$ and $N[v_0, v_1, \ldots, v_k]$ for $N[\{v_0, v_1, \ldots, v_k\}]$.

We use the following notation to talk about paths. If $X_1, X_2, \ldots, X_k \subseteq V(G)$, then a P_k *of the form* $X_1X_2 \ldots X_k$ is an induced copy of P_k in G so that the first vertex is in X_1, the second vertex is in X_2, and so on. If $X_i = \{v\}$ for some vertex v, then we may put v instead of X_i in the sequence denoting the form. For instance, given a vertex v and a set $A \subseteq V(G)$, a P_4 of the form $vAAA$ is one that starts at a vertex v and has the rest of its vertices in A.

We say that two disjoint sets $X, Y \subseteq V(G)$ are *complete* if every vertex in X is adjacent to every vertex in Y. If $X = \{v\}$ for some vertex v, then we say that v and Y are *complete*. Similarly, we say that two disjoint sets, or a vertex and a set not containing that vertex, are *anticomplete* if they are complete in the complement of G. The complement of G is denoted by $\overline{G}$.

The following observation is straightforward and will be often used implicitly.

Observation 2.1. *Let G be a graph, X be a connected subset of $V(G)$, and $v \in V(G) \setminus X$ be neither complete nor anticomplete to X. Then there exists a P_3 of the form vXX.*

2.1 Logic

In this paper we use the logic CMSO_2, which stands for monadic second-order logic with quantification over edge subsets and modular counting predicates, as a language for expressing graph problems. In this logic we have variables of four sorts: for single vertices, for single edges, for vertex subsets, and for edge subsets. The latter two types are called *monadic variables*. Atomic formulas of CMSO_2 are as follows:

- equality $x = y$ for any two variables x, y of the same sort;
- membership $x \in X$, where X is a monadic variable and x is a single vertex/edge variable;
- modular counting predicates of the form $|X| \equiv a \bmod m$, where X is a monadic variable and a, m are integers, $m \neq 0$; and
- incidence $\mathsf{inc}(x, f)$, checking whether vertex x is incident to edge f.

Then CMSO_2 consists of all formulas that can be obtained from the atomic formulas by means of standard boolean connectives, negation, and universal and existential quantification (over all sorts of variables). This gives the syntax of CMSO_2, and the semantics is obvious.

Note that a formula may have *free variables*, which are variables not bound by any quantifier. A formula without free variables is called a *sentence*.

It will be sometimes useful to consider graphs with some vertex annotations. Formally, for a finite set Σ^0, a Σ^0-*annotated graph* G is a graph G together with a function $\lambda^0 : V(G) \to \Sigma^0$. A CMSO_2 formula φ over Σ^0-annotated graphs has additionally access to atomic formulas $\lambda^0(x) = \sigma$ for a single vertex variable x and $\sigma \in \Sigma^0$. The $(\mathrm{tw} \leqslant k, \varphi)$-MWIS problem naturally generalizes to Σ^0-annotated graphs G.

Logic CMSO_2 is usually associated with tree-like graphs through the following fundamental result of Courcelle [7]: given an n-vertex graph G of treewidth at most k and a sentence φ of CMSO_2, one can determine whether φ holds in G in time $f(k, \varphi) \cdot n$, for a computable function f. The proof of this result brings the notion of *tree automata* to the setting of tree-like graphs, which is a connection that will be also exploited in this work. For an introduction to this area, see the monograph of Courcelle and Engelfriet [8].

2.2 Treewidth and treedepth

We now introduce treedepth because it turns out to be a more natural width parameter than treewidth in the context of P_t-free graphs. It is convenient to begin with some definitions on forests.

A *rooted forest* is a forest $\mathcal{T}$ where each component has exactly one specified vertex called its *root*. The *depth* of a vertex $v \in V(\mathcal{T})$ is the number of vertices in the unique path from v to a root (so roots have depth 1). The *height* of $\mathcal{T}$ is the maximum depth of any of its vertices. A path in $\mathcal{T}$ is *vertical* if one of its ends is an ancestor of the other. (We consider each vertex to be both an ancestor and a descendent of itself.) Two vertices are $\mathcal{T}$-*comparable* if they are connected by a vertical path; otherwise they are $\mathcal{T}$-*incomparable*.

An *elimination forest* of a graph G is a rooted forest $\mathcal{T}$ such that $V(\mathcal{T}) = V(G)$ and the endpoints of each edge of G are $\mathcal{T}$-comparable. The *treedepth* of G is then the smallest integer d such that G has an elimination forest of height d. Finally, we define the problem $(\mathrm{td} \leqslant d, \varphi)$-MWIS analogously to $(\mathrm{tw} \leqslant k, \varphi)$-MWIS, where the only difference is that $G[\mathtt{Sol}]$ is required to have treedepth at most d (instead of treewidth at most k).

Luckily, in the context of P_t-free graphs, the parameters of treedepth, treewidth, and degeneracy are functionally equivalent due to the following theorem.

Theorem 2.2. *For any integers t and ℓ, there exists an integer d such that if G is a P_t-free graph with degeneracy at most ℓ, then the treedepth of G is at most d.*

Theorem 2.2 has been discussed in [11], but let us recall the reasoning. The first step is the following result of [11]. (A graph is $C_{>t}$-*free* if it does not contain a cycle longer than t as an induced subgraph; note that the class of $C_{>t}$-free graphs is a proper superclass of the class of P_t-free graphs.)

Theorem 2.3 ([11])**.** *For every pair of integers ℓ and t, there exists an integer $k \in (\ell t)^{\mathcal{O}(t)}$ such that every $C_{>t}$-free graph of degeneracy at most ℓ has treewidth at most k.*

Treewidth and treedepth are functionally equivalent on P_t-free graphs by the following result of [4].

Theorem 2.4 ([4, Lemma 29]). *For any integer t, if G is a P_t-free graph, then*

$$\text{treedepth}(G) \leqslant (\text{treewidth}(G)+1)^{t-1}.$$

Since the property of having treewidth at most k and the property of having treedepth at most d can be expressed in CMSO_2, we obtain that the $(\text{tw} \leqslant k, \varphi)$-MWIS and $(\text{td} \leqslant d, \varphi)$-MWIS formalisms describe the same class of problems in P_t-free graphs for any fixed t; every $(\text{tw} \leqslant k, \varphi)$-MWIS problem has an equivalent definition as a $(\text{td} \leqslant d, \varphi')$-MWIS for some d and φ' depending on k and φ, and vice-versa. Hence, in this paper we can focus on solving problems formulated in the $(\text{td} \leqslant d, \varphi)$-MWIS formalism.

2.3 Chordal completions and PMCs

Recall that our overall approach is based on potential maximal cliques. We introduce this approach now.

Given a graph G, a set $\Omega \subseteq V(G)$ is a *potential maximal clique* (or a *PMC*) if there exists a minimal chordal completion of G in which Ω is a maximal clique. A *chordal completion* of G is a supergraph of G which is chordal and has the same vertex-set as G; it is *minimal* if it has no proper subgraph which is also a chordal completion of G. (Recall that a graph is *chordal* if it has no holes, where a *hole* is an induced cycle of length at least 4.) Since chordal completions are obtained by adding edges to G, it is convenient to write them as $G+F$, where F is a set of non-edges of G.

The following classic result characterizes PMCs.

Proposition 2.5 ([6, Theorem 3.15]). *Given a graph G, a set $\Omega \subseteq V(G)$ is a PMC if and only if both of the following conditions hold.*

(i) For each component D of $G-\Omega$, $N(D)$ is a proper subset of Ω.

(ii) If uv is a non-edge of G with $u,v \in \Omega$, then there exists a component D of $G-\Omega$ such that $u,v \in N(D)$.

Chordal completions in a certain sense correspond to tree decompositions and it is often more convenient to work with the latter. So recall that a *tree decomposition* of a graph G is a pair (T,β) such that T is a tree, β is a function from $V(T)$ to $2^{V(G)}$, and the following conditions are satisfied:

(i) for each $u \in V(G)$, the set $\{t \in V(T) : u \in \beta(t)\}$ induces a non-empty and connected subtree of T, and

(ii) for each $uv \in E(G)$, there is a node t of T such that $u,v \subseteq \beta(t)$.

For a node t of T, the set $\beta(t)$ is called the *bag* of t, and for an edge $st \in E(T)$, the set $\beta(s) \cap \beta(t)$ is called the *adhesion* of st, and is denoted by $\sigma(st)$.

It is a folklore result that a graph H is chordal if and only if it has a tree decomposition whose bags are exactly the maximal cliques of H (meaning, in particular, that the number of nodes of the tree is equal to the number of maximal cliques of H). Such a tree decomposition is called a *clique tree* of H; note that while the set of bags of a clique tree is defined uniquely, the actual tree part of the tree decomposition is not necessarily unique. For example, if $H = K_{1,s}$, then there are s maximal cliques (corresponding to edges of H), but they can be arranged into a tree decomposition in essentially an arbitrary manner. We also remark that a chordal graph on n vertices has at most n maximal cliques, and hence its clique tree has at most n nodes.

We will need some additional facts about clique trees of minimal chordal completions. Let G be a graph. Given a set $S \subseteq V(G)$, a *full component* of S is a component A of $G - S$ such that $N(A) = S$. A *minimal separator* of G is then a set $S \subseteq V(G)$ which has at least two full components.

The next two lemmas were proven in [13] using the toolbox from [6]. The first one shows how to obtain minimal separators from adhesions.

Lemma 2.6 ([13, Proposition 2.7])**.** *Let G be a graph, $G + F$ be a minimal chordal completion of G, and (T, β) be a clique tree of $G + F$. Then for each edge $st \in E(T)$, the adhesion $\sigma(st)$ is a minimal separator of G, and it has full components A and B such that $\beta(s) \setminus \sigma(st) \subseteq A$ and $\beta(t) \setminus \sigma(st) \subseteq B$.*

Notice that the full component A which satisfies Lemma 2.6 is unique given the vertex s and the edge $st \in E(T)$. (This uses the fact that $\beta(s) \setminus \sigma(st)$ is non-empty, which holds since $\beta(s)$ and $\beta(t)$ are distinct maximal cliques of $G+F$.) When the graph, chordal completion, and clique tree are clear from context, we call A *the full component of $\sigma(st)$ on the s-side.*

Lemma 2.6 immediately implies also the following.

Lemma 2.7. *Let G be a graph, $G + F$ be a minimal chordal completion of G, and (T, β) be a clique tree of $G + F$. Then for every $st \in E(T)$, there exists a connected component D of $G - \beta(t)$ such that $N(D) = \sigma(st)$ and $D \subseteq \bigcup_{t' \in V(T_s)} \beta(t')$ where T_s is the component of $T - \{t\}$ that contains s.*

Proof. Use Lemma 2.6 and take D to be the full component of $\sigma(st)$ on the s-side. □

The next lemma shows how to obtain minimal separators from PMCs.

Lemma 2.8 ([13, Proposition 2.10])**.** *Let G be a graph, Ω be a PMC of G, and D be a component of $G - \Omega$. Then $N(D)$ is a minimal separator of G, and it has a full component $D^\Omega \neq D$ which contains $\Omega \setminus N(D)$.*

We will also need the following well-known facts about chordal completions.

Lemma 2.9. *Let G be a graph and $G+F$ be a minimal chordal completion of G. Let $S \subseteq V(G)$ be such that $(G + F)[S]$ is a clique. Then F contains no edges between different connected components of $G - S$.*

Proof. Let $\mathcal{D}$ be the family of connected components of $G - S$. For every $D \in \mathcal{D}$, $F \cap \binom{N[D]}{2}$ is a chordal completion of $G[N[D]]$ that turns $N(D)$ into a clique. Since $(G + F)[S]$ is a clique, $(F \cap \binom{S}{2}) \cup \bigcup_{D \in \mathcal{D}} F \cap \binom{N[D]}{2}$ is a chordal completion of G. The claim follows by the minimality of $G + F$. □

Lemma 2.10. *Let G be a graph, $G + F$ be a minimal chordal completion of G, and (T, β) be a clique tree of $G + F$. Let S be a minimal separator of G such that $(G + F)[S]$ is a clique and let A and B be two full sides of S. Then there exists an edge $t_A t_B \in E(T)$ such that $\sigma(t_A t_B) = S$, $A \subseteq \bigcup_{t \in V(T_A)} \beta(t)$, $B \subseteq \bigcup_{t \in V(T_B)} \beta(t)$, where T_A and T_B are the components of $T - \{t_A t_B\}$ that contain t_A and t_B, respectively.*

Proof. Let $Z_A = \{t \in V(T) \mid A \cap \beta(t) \neq \emptyset\}$ and similarly define Z_B. Since A and B are connected, Z_A and Z_B are connected in T. By Lemma 2.9, $Z_A \cap Z_B = \emptyset$. Let Q be the unique path in T that has one endpoint in Z_A, the second endpoint in Z_B, and all internal vertices outside $Z_A \cup Z_B$. Note that the length of Q is at least one. Let q_A and q_B be the endpoints of Q in Z_A and Z_B, respectively.

Since (T, β) is a tree decomposition of G, $N_G[A] \subseteq \bigcup_{t \in Z_A} \beta(t)$. Since (T, β) is a clique tree of the chordal graph $G + F$, we have $N_{G+F}[A] \supseteq \bigcup_{t \in Z_A} \beta(t)$. Lemma 2.9 implies that

$N_G[A] = N_{G+F}[A]$. Thus $N_G[A] = N_{G+F}[A] = \bigcup_{t \in Z_A} \beta(t)$ and, similarly, $N_G[B] = N_{G+F}[B] = \bigcup_{t \in Z_B} \beta(t)$.

Since $S = N_G[A] \cap N_G[B]$, $S \subseteq \beta(s)$ for every $s \in V(Q)$. By the definition of Z_A, we have $\beta(s) \cap N_G[A] \subseteq S$ for every $s \in V(Q) \setminus \{q_A\}$. Hence, if q is the unique neighbor of q_A on Q, then $\sigma(qq_A) = S$. The lemma follows with $t_A = q_A$ and $t_B = q$. □

2.4 Aligning chordal completions and treedepth structures

Throughout the paper we will try to find a maximal induced subgraph with treedepth at most d. We will do so by considering a fixed elimination forest of this induced subgraph, as well as a chordal completion which "aligns with" the elimination forest. We now formalize these ideas.

Let G be a graph and d be a positive integer. A *treedepth-d structure in G* is a rooted forest $\mathcal{T}$ of height at most d such that $V(\mathcal{T})$ is a subset of $V(G)$ and $\mathcal{T}$ is an elimination forest of the subgraph of G induced by $V(\mathcal{T})$. We sometimes write $\mathcal{T}$ instead of $V(\mathcal{T})$ when it is clear that we are working with a set of vertices; in particular, if X is a set of vertices of G, then we write $X \cap \mathcal{T}$ instead of $X \cap V(\mathcal{T})$. We say that $\mathcal{T}$ is *maximal* if there is no treedepth-d structure $\mathcal{T}'$ in G such that $\mathcal{T}$ is a proper induced subgraph of $\mathcal{T}'$ and every root of $\mathcal{T}$ is a root of $\mathcal{T}'$.

Note that if H is a maximal induced subgraph of G of treedepth at most d, and $\mathcal{T}$ is a height-d elimination forest of that subgraph, then $\mathcal{T}$ is a maximal treedepth-d structure in G. Consequently, in the context of $(\mathrm{td} \leqslant d, \varphi)$-MWIS, we can consider $\texttt{Sol}$ being in fact a maximal set inducing a subgraph of treedepth at most d in G: if $(\texttt{Sol}, X)$ is an actual solution, then there exists a maximal treedepth-d structure $\texttt{Sol}'$ that is a superset of $\texttt{Sol}$ and quantification over $\texttt{Sol}$ can be implemented inside φ. (This step is formally explained in Section 3.) Thus, most of the structural results in this work consider the set of all maximal treedepth-d-structures, which are more detailed versions of maximal sets inducing a subgraph of treedepth at most d.

We conclude this section by discussing "aligned" chordal completions and by proving some basic lemmas about them. Let G be a graph, d be a positive integer, and $\mathcal{T}$ be a treedepth-d structure in G. We say that a chordal completion $G + F$ is *$\mathcal{T}$-aligned* if F does not contain any pair uv so that

(i) u or v is a depth-d vertex of $\mathcal{T}$, or

(ii) u and v are vertices of $\mathcal{T}$ which are $\mathcal{T}$-incomparable.

The second condition equivalently says that $\mathcal{T}$ is a treedepth-d structure in $G + F$. First we show that there is always a $\mathcal{T}$-aligned minimal chordal completion.

Lemma 2.11. *For any positive integer d, graph G, and treedepth-d structure $\mathcal{T}$ in G, there exists a minimal chordal completion of G that is $\mathcal{T}$-aligned.*

Proof. Let F denote the set of all non-edges uv of G which are not incident to a depth-d vertex of $\mathcal{T}$, and are not between two vertices of $\mathcal{T}$ which are $\mathcal{T}$-incomparable. It suffices to prove that $G + F$ is chordal, since any chordal subgraph of $G + F$ is $\mathcal{T}$-aligned.

Going for a contradiction, suppose that C is a hole of $G + F$. As $(G + F) - \mathcal{T}$ is a clique, there is a vertex in $C \cap \mathcal{T}$; choose one, say u, which has maximum depth in $\mathcal{T}$. Consider the two neighbors of u in C; they are either outside of $\mathcal{T}$ or ancestors of u in $\mathcal{T}$. However, the set of all such vertices forms a clique in $G + F$, which contradicts the fact that C has length at least 4. □

Throughout the paper we consider PMCs and minimal separators which might come from an aligned chordal completion. So, to state these definitions, let G be a graph, d be a positive integer, and $\mathcal{T}$ be a treedepth-d structure in G. A PMC Ω is *$\mathcal{T}$-avoiding* if it is a maximal clique of a minimal chordal completion that is $\mathcal{T}$-aligned, and it does not contain any depth-d vertex of $\mathcal{T}$. We deal with the case that Ω does contain a depth-d vertex separately, in the next lemma.

Finally, a minimal separator S of G is *$\mathcal{T}$-avoiding* if $S \cap \mathcal{T}$ is contained in a vertical path of $\mathcal{T}$ and has no depth-d vertex. (So these are the separators that can come from $\mathcal{T}$-avoiding PMCs.)

For a fixed treedepth-d structure $\mathcal{T}$, a set $\widetilde{Y} \subseteq V(G)$ is a *container* for a set $Y \subseteq V(G)$ if $Y \subseteq \widetilde{Y}$ and $\widetilde{Y} \cap \mathcal{T} = Y \cap \mathcal{T}$, that is, $\widetilde{Y} \setminus Y$ is disjoint from $\mathcal{T}$.

Lemma 2.12. *For each positive integer d, there is a polynomial-time algorithm which takes in a graph G and returns a collection $\mathcal{L} \subseteq 2^{V(G)}$ such that for any maximal treedepth-d structure $\mathcal{T}$ in G, any $\mathcal{T}$-aligned minimal chordal completion $G + F$ of G, and any maximal clique Ω of $G + F$ which contains a depth-d vertex of $\mathcal{T}$, $\mathcal{L}$ contains a set $\widetilde{\Omega}$ that is a container for Ω, i.e., $\Omega \subseteq \widetilde{\Omega}$ and $\widetilde{\Omega} \cap \mathcal{T} = \Omega \cap \mathcal{T}$.*

Proof. We guess the vertex $v \in \Omega$ which is a depth-d vertex of $\mathcal{T}$ (this vertex is unique). Thus v is adjacent in G to every other vertex of Ω, because Ω is a clique in a $\mathcal{T}$-aligned chordal completion. Moreover, v has at most $d-1$ neighbors in $\mathcal{T}$. We then guess the set X of all neighbors of v which are in $\mathcal{T}$ but are not in Ω. Finally, for all guesses of v and X, we add the set $N[v] \setminus X$ to $\mathcal{L}$. This collection $\mathcal{L}$ is as desired. □

The final lemma is how we will use the maximality of a treedepth-d structure.

Lemma 2.13. *Let G be a graph, d be a positive integer, and $\mathcal{T}$ be a maximal treedepth-d structure in G. Then for any $\mathcal{T}$-avoiding potential maximal clique Ω of G, each vertex in $\Omega \setminus \mathcal{T}$ has a neighbor in $\mathcal{T} \setminus \Omega$.*

Proof. Recall that the set $\Omega \cap \mathcal{T}$ is contained in a vertical path of $\mathcal{T}$. Moreover, since Ω is $\mathcal{T}$-avoiding, $\Omega \cap \mathcal{T}$ does not contain any depth-d vertex of $\mathcal{T}$. So, if a vertex $u \in \Omega \setminus \mathcal{T}$ had no neighbor in $\mathcal{T} \setminus \Omega$, then we could find another treedepth-d structure $\mathcal{T}'$ in G where $\mathcal{T}'$ is obtained from $\mathcal{T}$ by adding u. □

3 Dynamic programming

Now let us recall the definition of the main object of study in this paper.

Definition 3.1. Let G be a graph and d and k be positive integers. A family $\mathcal{C} \subseteq 2^{V(G)}$ is a *tree-depth-d carver family* of *defect* k in G if for every treedepth-d structure $\mathcal{T}$ in G, there exists a tree decomposition (T, β) of G such that for each $t \in V(T)$ there exists $C \in \mathcal{C}$ such that

- *(i)* $C \cap \mathcal{T}$ contains $\beta(t) \cap \mathcal{T}$ and has size at most k, and
- *(ii)* each component of $G - C$ is contained in $\beta(t) \cup \bigcup_{s \in T'} \beta(s)$ for some component T' of $T - \{t\}$.

Such a set C as above is called a *$(\mathcal{T}, (T, \beta))$-carver* for $\beta(t)$ of defect k; it might not be unique. We use this definition independently of that of carver families.

It is important to compare the notion of a carver family with the notion of *containers* of [1]. There, instead of the properties above, we mandate that $|\beta(t) \cap \mathcal{T}| \leqslant k$, that $C \cap \mathcal{T} = \beta(t) \cap \mathcal{T}$, and that $\beta(t) \subseteq C$ (so that, in particular, the choice of the tree T in the tree decomposition (T, β) is irrelevant for the definition). These requirements imply that parts *(i)* and *(ii)* of the definition of a carver family hold; for the second part, observe that if $\beta(t) \subseteq C$, then any component of $G - C$ is contained in a component of $G - \beta(t)$ which, by the properties of a tree decomposition, lies in the union of bags of a single component of $T - \{t\}$. The main difference is that in the notion of a carver, we actually allow a carver C to miss some vertices of Ω, as long as this does not result in "gluing" connected components of $G - \Omega$ residing in different subtrees of $T - \{t\}$ within the same connected component of $G - C$.

The main result of this section is that a tree-depth-d carver family of small defect in G is enough to design a dynamic programming routine that solves the $(\mathrm{td} \leqslant d, \varphi)$-MWIS problem

on G. We state it in the slightly more general form that allows Σ^0-annotated graphs for fixed finite set Σ^0.

Theorem 3.2. *For any positive integers d and k, any finite set Σ^0 and any CMSO_2 formula φ over the signature of Σ^0-annotated graphs, there exists an algorithm that, given a vertex-weighted Σ^0-annotated graph G and a tree-depth-d carver family $\mathcal{C} \subseteq 2^{V(G)}$ of defect k in G, runs in time polynomial in the input size and either outputs an optimal solution to the $(\mathrm{td} \leqslant d, \varphi)$-MWIS problem on G, or determines that no feasible solution exists.*

The remainder of this section is devoted to the proof of Theorem 3.2.

3.1 Canonizing and extending partial solutions

Fix an integer d and let G be a graph. A *partial solution* in G is any tuple $(\mathcal{T}, X, \mathtt{Sol})$ such that $\mathcal{T}$ is a tree-depth-d structure in G and $X \subseteq \mathtt{Sol} \subseteq V(\mathcal{T})$.

Very roughly, the dynamic programming routine will have a table with some entries for each partial solution $(\mathcal{T}, X, \mathtt{Sol})$ such that $\mathcal{T}$ has at most k leaves. Each of these entries will contain a partial solution $(\mathcal{T}', X', \mathtt{Sol}')$ which "extends" $(\mathcal{T}, X, \mathtt{Sol})$ into a specified part of the graph. We will update this partial solution $(\mathcal{T}', X', \mathtt{Sol}')$ when we find a "better" one. Sometimes this choice is arbitrary. So, in order to have more control over arbitrary choices, we now introduce a consistent tie-breaking scheme over partial solutions. More formally, we introduce a quasi-order $\preceq$ over partial solutions.

First, fix an arbitrary enumeration of $V(G)$ as $v_1, v_2, \ldots, v_{|V(G)|}$. Second, define a total order $\preceq_1$ on subsets of $V(G)$ as follows: $X \prec_1 Y$ if $|X| > |Y|$ or if $|X| = |Y|$ and we have $v_i \in X$, where i is the minimum integer such that $v_i \in X \triangle Y$ (i.e., we use the lexicographic order). Third, define a quasi-order $\preceq_2$ on tree-depth-d structures in G as follows. Given a tree-depth-d structure $\mathcal{T}$, associate to $\mathcal{T}$ the following tuple of $d+1$ subsets of $V(G)$:

- $V(\mathcal{T})$,
- the set of all vertices of depth 1 in $\mathcal{T}$ (i.e., the roots),
- the set of all vertices of depth 2 in $\mathcal{T}$,
  ...
- the set of all vertices of depth d in $\mathcal{T}$.

When comparing two tree-depth-d structures with $\preceq_2$, we compare with $\preceq_1$ the first sets in the above tuple that differ.

For two distinct tree-depth-d structures $\mathcal{T}$ and $\mathcal{T}'$, we have $\mathcal{T} \preceq_2 \mathcal{T}'$ or $\mathcal{T}' \preceq_2 \mathcal{T}$. However, we may have both $\mathcal{T} \preceq_2 \mathcal{T}'$ or $\mathcal{T}' \preceq_2 \mathcal{T}$ (i.e., it is possible that, for two different tree-depth-d structures $\mathcal{T}$ and $\mathcal{T}'$, we have $V(\mathcal{T}) = V(\mathcal{T}')$ and every vertex of $V(\mathcal{T})$ has the same depth in $\mathcal{T}$ and in $\mathcal{T}'$). So $\preceq_2$ is only a quasi-order on the set of all tree-depth-d structures in G; it partitions tree-depth-d structures into equivalence classes, and between the equivalence classes it is a total order.

In order to avoid this problem, we will show that we can convert any tree-depth-d structure into one that is "neat", and that $\preceq_2$ is a total order on "neat" tree-depth-d structures. Formally, a tree-depth-d structure $\mathcal{T}$ of G is *neat* if for any non-root node v of $\mathcal{T}$, the graph G has at least one edge joining the parent of v in $\mathcal{T}$ with a descendant of v in $\mathcal{T}$ (possibly v itself). One can easily see that this is equivalent to the following condition: for every node v of $\mathcal{T}$, the subgraph of G induced by the descendants of v (including v) is connected.

The following lemma is standard when working with elimination forests: any tree-depth-d structure can be adjusted to a neat one without increasing the depth.

Lemma 3.3. *Given a graph G and a tree-depth-d structure $\mathcal{T}$ of G, one can in polynomial time compute a neat tree-depth-d structure $\mathcal{T}'$ of G such that $V(\mathcal{T}') = V(\mathcal{T})$ and for each $v \in V(\mathcal{T})$, the depth of v in $\mathcal{T}'$ is at most the depth of v in $\mathcal{T}$.*

Proof. While possible, perform the following improvement step. If $v \in V(\mathcal{T})$ is such that v is not a root of $\mathcal{T}$, but has a parent u, and the subtree $\mathcal{T}_v$ of $\mathcal{T}$ rooted at v does not contain a vertex of $N_G(u)$, then reattach $\mathcal{T}_v$ to the parent of u if u is not a root or detach it as a separate component of $\mathcal{T}$ otherwise. It is immediate that the new rooted forest is also a tree-depth-d structure of G and, furthermore, that the depths of the elements of $\mathcal{T}_v$ decreased by one. This in particular implies that there will be at most $|V(G)|^2$ improvement steps. Each of them can be executed in polynomial time. Once no more improvement steps are possible, the resulting tree-depth-d structure is neat, as desired. □

Next, we show that $\preceq_2$ is a total order on neat tree-depth-d structures. In fact we show something slightly stronger: that each neat tree-depth-d structure is in a singleton equivalence class.

Lemma 3.4. *If $\mathcal{T}$ and $\mathcal{T}'$ are tree-depth-d structures such that $\mathcal{T}$ is neat, $\mathcal{T} \preceq_2 \mathcal{T}'$, and $\mathcal{T}' \preceq_2 \mathcal{T}$, then $\mathcal{T} = \mathcal{T}'$.*

Proof. Since $\mathcal{T} \preceq_2 \mathcal{T}'$ and $\mathcal{T}' \preceq_2 \mathcal{T}$, we have that $V(\mathcal{T}) = V(\mathcal{T}')$ and every vertex has the same depth in $\mathcal{T}$ and $\mathcal{T}'$. We prove inductively on i that the set of all vertices of depth at least $d - i$ induces the same forest in $\mathcal{T}$ and $\mathcal{T}'$. The base case of $i = 0$ holds since the depth-d vertices are an independent set in both $\mathcal{T}$ and $\mathcal{T}'$. For the inductive step, it suffices to show that each depth-$(d-i)$ vertex v has the same parent in $\mathcal{T}$ and $\mathcal{T}'$. So let u and u' be the parent of v in $\mathcal{T}$ and $\mathcal{T}'$, respectively. From the inductive hypothesis, the subtrees of $\mathcal{T}$ and $\mathcal{T}'$ rooted at v are equal. Since $\mathcal{T}$ is neat, there is a vertex w in this subtree that is adjacent to u in G. Since $\mathcal{T}'$ is an elimination forest, u' and w are comparable in $\mathcal{T}'$. Since u' and u have the same depth in $\mathcal{T}$ and $\mathcal{T}'$, this is only possible if $u = u'$. □

Finally, given two partial solutions $(\mathcal{T}, X, \texttt{Sol})$ and $(\mathcal{T}', X', \texttt{Sol}')$ in a graph G, we say that $(\mathcal{T}, X, \texttt{Sol}) \preceq (\mathcal{T}', X', \texttt{Sol}')$ if:

- *(i)* the weight of X is larger than the weight of X', or
- *(ii)* the weights of X and X' are equal, but $X \prec_1 X'$;
- *(iii)* $X = X'$, but $\texttt{Sol} \prec_1 \texttt{Sol}'$;
- *(iv)* $X = X'$ and $\texttt{Sol} = \texttt{Sol}'$, but $\mathcal{T} \preceq_2 \mathcal{T}'$.

We say that $(\mathcal{T}, X, \texttt{Sol})$ is *better* than $(\mathcal{T}', X', \texttt{Sol}')$ (or that $(\mathcal{T}', X', \texttt{Sol}')$ is *worse* than $(\mathcal{T}, X, \texttt{Sol})$) if $(\mathcal{T}, X, \texttt{Sol}) \preceq (\mathcal{T}', X', \texttt{Sol}')$ and some comparison above is strict (or, equivalently, if it does not hold that $\mathcal{T}' \preceq_2 \mathcal{T}$).

Using this quasi-order, we can now look for a partial solution $(\mathcal{T}, X, \texttt{Sol})$ such that $\mathcal{T}$ is maximal and neat. This is based on the following observation.

Lemma 3.5. *For any $X \subseteq \texttt{Sol} \subseteq V(G)$ such that $G[\texttt{Sol}]$ has tree-depth at most d, there exists a tree-depth-d structure $\mathcal{T}$ such that $(\mathcal{T}, X, \texttt{Sol})$ is a partial solution. Moreover, if one chooses $\mathcal{T}$ so that $(\mathcal{T}, X, \texttt{Sol})$ is $\preceq$-minimal (among all choices of $\mathcal{T}$, for fixed X and $\texttt{Sol}$), then $\mathcal{T}$ is maximal and neat.*

Proof. For the first claim, any depth-d elimination forest of $G[\texttt{Sol}]$ can serve as $\mathcal{T}$. For the second claim, fix a $\preceq$-minimal partial solution $(\mathcal{T}, X, \texttt{Sol})$. Since the first comparison is on the sizes of $V(\mathcal{T})$, we have that $\mathcal{T}$ is maximal.

Now, by Lemma 3.3, there exists a neat tree-depth-d structure $\mathcal{T}'$ such that $V(\mathcal{T}') = V(\mathcal{T})$ and, for each $v \in V(\mathcal{T})$, the depth of v in $\mathcal{T}'$ is at most the depth of v in $\mathcal{T}$. Thus $\mathcal{T}' \preceq_2 \mathcal{T}$. So, since $(\mathcal{T}', X, \texttt{Sol})$ is not better than $(\mathcal{T}, X, \texttt{Sol})$, we also have that $\mathcal{T} \preceq_2 \mathcal{T}'$. Since $\mathcal{T}'$ is neat, Lemma 3.4 says that $\mathcal{T}' = \mathcal{T}$. So $\mathcal{T}$ is neat, as desired. □

It is convenient to conclude this subsection by defining extensions of partial solutions. Roughly, an "extension" of a partial solution $(\mathcal{T}, X, \texttt{Sol})$ in a graph G is any partial solution $(\mathcal{T}', X', \texttt{Sol}')$ that can be obtained from $(\mathcal{T}, X, \texttt{Sol})$ by adding new vertices which are not ancestors of any node of $\mathcal{T}$. More formally, $(\mathcal{T}', X', \texttt{Sol}')$ is an *extension* of $(\mathcal{T}, X, \texttt{Sol})$ if $\mathcal{T}$ is an induced subgraph of $\mathcal{T}'$, every root of $\mathcal{T}$ is a root of $\mathcal{T}'$, and $X' \cap \mathcal{T} = X$ and $\texttt{Sol}' \cap \mathcal{T} = \texttt{Sol}$. We define extensions of tree-depth-d structures analogously, omitting X and $\texttt{Sol}$.

We will use the following properties of extensions.

Lemma 3.6. *Let G be a graph, d be an integer, and $\mathcal{T}$ and $\mathcal{T}'$ be tree-depth-d structures in G such that $\mathcal{T}'$ is neat and extends $\mathcal{T}$. Then each connected component of $\mathcal{T}' - V(\mathcal{T})$ is neat and induces a connected subgraph of G.*

Proof. Each connected component of $\mathcal{T}' - V(\mathcal{T})$ is obtained by selecting a vertex $v \in V(\mathcal{T}')$ and then taking all descendants of v in $\mathcal{T}'$ (including v itself). Any such subtree of $\mathcal{T}'$ is neat. For the second part, we observe that any neat tree-depth-d structure with just one root induces a connected subgraph of G. □

3.2 Threshold automata

Next, we introduce *threshold automata*, which capture through an abstract notion of a computation device, the idea of processing a labelled forest in a bottom-up manner using a dynamic programming procedure. As we will comment on, the design of this automata model follows standard constructions that were developed in the 90s.

We need to introduce some notation before stating the main definitions. For a finite alphabet Σ, a Σ-*labelled forest* is a rooted forest F where every vertex $x \in V(F)$ is labelled with an element $\mathsf{label}(x) \in \Sigma$. Similarly, given an unlabelled rooted forest F, we call any function label from $V(F)$ to Σ a Σ-*labelling* of F.

We use the notation $\{\{\cdot\}\}$ for defining multisets. For a multiset X and an integer $\tau \in \mathbb{N}$, let $X \wedge \tau$ be the multiset obtained from X by the following operation: for every element e whose multiplicity k is larger than 2τ, we reduce its multiplicity to the unique integer in $\{\tau+1, \ldots, 2\tau\}$ with the same residue as k modulo τ (that is, we reduce it to $k - \tau\lfloor\frac{k-\tau-1}{\tau}\rfloor$). This definition lets us track at the same time the residue modulo τ of the multiplicity as well as whether the multiplicity is greater than τ or not. For a finite set Q and an integer $\tau \in \mathbb{N}$, we write $\mathsf{Multi}(Q, \tau)$ for the family of all multisets with elements from Q, where each element appears at most 2τ times. Note that $|\mathsf{Multi}(Q, \tau)| = (2\tau + 1)^{|Q|}$.

Informally, a threshold automaton is run bottom-up on a Σ-labelled forest F. As it runs, it assigns each vertex of F a state from a finite set Q. The state of the next vertex $v \in V(F)$ depends only on $\mathsf{label}(v)$ and the "reduced" multiset $X \wedge \tau$, where X denotes the multiset of the states of all children of v. The accepting condition is similarly determined by "reducing" the multiset of the states of the roots. The formal definition is as follows.

Definition 3.7. A *threshold automaton* is a tuple $\mathcal{A} = (Q, \Sigma, \tau, \delta, C)$, where:

- Q is a finite set of states;
- Σ is a finite alphabet;
- $\tau \in \mathbb{N}$ is a nonnegative integer called the *threshold*;
- $\delta\colon \Sigma \times \mathsf{Multi}(Q, \tau) \to Q$ is the transition function; and
- $C \subseteq \mathsf{Multi}(Q, \tau)$ is the accepting condition.

For a Σ-labelled forest F, the *run* of $\mathcal{A}$ on F is the unique labelling $\xi\colon V(F) \to Q$ satisfying the following property for each $x \in V(F)$:

$$\xi(x) = \delta(\mathsf{label}(x), \{\{\xi(y)\colon y \text{ is a child of } x\}\} \wedge \tau).$$

We say that $\mathcal{A}$ *accepts* F if

$$\{\{\xi(z)\colon z \text{ is a root of } F\}\} \wedge \tau \in C,$$

where ξ is the run of $\mathcal{A}$ on F.

It turns out that threshold automata precisely characterize the expressive power of CMSO over labelled forests. Here, we consider the standard encoding of Σ-labelled forests as relational structures using one binary parent relation and $|\Sigma|$ unary relations selecting nodes with corresponding labels. Consequently, by CMSO over Σ-labelled forests we mean the logic in which

- there are variables for single nodes and for node sets,
- in atomic formulas one can check equality, membership, modular counting predicates, parent relation, and labels of single nodes, and
- larger formulas can be obtained from atomic ones using standard boolean connectives, negation, and both universal and existential quantification over both sorts of variables.

The proof of the next statement is standard, see for instance [7, Theorem 5.3] for a proof in somewhat different terminology and [16, Section 7.6] for the closely related settings of binary trees and ordered, unranked trees (the proof techniques immediately lift to our setting). Hence, we only provide a sketch.

Lemma 3.8. *Let Σ be a finite alphabet. Then for every sentence φ of CMSO over Σ-labelled forests, there exists a threshold automaton $\mathcal{A}$ with alphabet Σ such that for any Σ-labelled forest F, we have $F \models \varphi$ if and only if $\mathcal{A}$ accepts F.*

Sketch of proof. Let the *rank* of φ be the product of the quantifier rank of φ (that is, the maximum number of nested quantifiers in φ) and the least common multiple of all moduli featured in modular predicates present in φ. It is well-known that there is only a finite number of pairwise non-equivalent CMSO sentences over Σ-labelled forests with rank at most q. Let then $\mathsf{Sentences}^q$ be the set containing one such sentence from each equivalence class. Then $\mathsf{Sentences}^q$ is finite, and we may assume that $\varphi \in \mathsf{Sentences}^q$.

Consider a Σ-labelled forest F. For a vertex $x \in V(F)$, let F_x be the subtree of F induced by x and all of its descendants. The *q-type* of F_x is the set of all sentences from $\mathsf{Sentences}^q$ which are satisfied in F_x, that is,

$$\mathrm{tp}^q(F_x) \coloneqq \{\, \psi \in \mathsf{Sentences}^q \;\mid\; F_x \models \psi \,\}.$$

A standard argument using Ehrenfeucht-Fraïsse games shows that $\mathrm{tp}^q(F_x)$ is uniquely determined by $\mathsf{label}(x)$ and the multiset $\{\{\mathrm{tp}^q(F_y) \colon y \text{ is a child of } x\}\} \wedge q$. Similarly, the type $\mathrm{tp}^q(F)$, defined analogously as above, is uniquely determined by the multiset $\{\{\mathrm{tp}^q(F_r) \colon r \text{ is a root of } F\}\} \wedge q$. This means that we may define a threshold automaton $\mathcal{A}$ with state set $\mathsf{Sentences}^q$ and threshold q so that $\mathcal{A}$ accepts F if and only if $\varphi \in \mathrm{tp}^q(F)$, which is equivalent to $F \models \varphi$. □

We would like to use Lemma 3.8 in order to verify that a given solution $(\mathtt{Sol}, X)$ to $(\mathrm{td} \leqslant d, \varphi)$-MWIS indeed is such that $G[\mathtt{Sol}]$ satisfies $\varphi(X)$. For this, our dynamic programming tables will be indexed not only by partial solutions of the form $(\mathcal{T}, X, \mathtt{Sol})$, but also by guesses on "partial evaluation" of φ that occurs outside of $V(\mathcal{T})$; or more formally, by an appropriate multiset of states of a threshold automaton associated with φ. For this, we need to understand how to run threshold automata on treedepth-d structures rather than just labelled forest. This will be done in a standard way: by labelling the forest underlying a treedepth-d structure $\mathcal{T}$ so to encode $\mathcal{T}$ through the labels. This idea is formalized in the next definition.

Definition 3.9. Let d be an integer and Σ^0, Σ be finite alphabets. Then a (d, Σ^0, Σ)-*labeller* is a polynomial-time algorithm Λ that, given an Σ^0-annotated graph (G, λ^0) with a partial solution $(\mathcal{T}, X, \mathtt{Sol})$ for the $(\mathrm{td} \leqslant d, \varphi)$-MWIS problem, computes a Σ-labelling of $\mathcal{T}$ such that for every $v \in V(\mathcal{T})$, the label of v depends only on:

- the label $\lambda^0(v)$,
- the integer $h \in \{1, 2, \ldots, d\}$ such that v has depth h in $\mathcal{T}$,
- the set of all indices $i \in \{1, 2, \ldots, h-1\}$ such that v is adjacent, in G, to the unique ancestor of v in $\mathcal{T}$ with depth i, and
- which of the sets X and $\mathtt{Sol}$ contain v.

That is, if we run Λ again on another G' and $(\mathcal{T}', X', \mathtt{Sol}')$, then any vertex $v' \in V(\mathcal{T}')$ with the same properties from above as v is labelled the same as v.

When Λ and (G, λ^0) are clear from context, we write $\mathsf{label}_{(\mathcal{T},X,\mathtt{Sol})}$ for the Σ-labelling on $\mathcal{T}$ which is returned by running Λ on (G, λ^0) and $(\mathcal{T}, X, \mathtt{Sol})$. A key aspect of this definition is that, if $(\mathcal{T}', X', \mathtt{Sol}')$ is a partial solution which extends $(\mathcal{T}, X, \mathtt{Sol})$, then each vertex $v \in V(\mathcal{T})$ satisfies $\mathsf{label}_{(\mathcal{T},X,\mathtt{Sol})}(v) = \mathsf{label}_{(\mathcal{T}',X',\mathtt{Sol}')}(v)$.

We are now ready to state the main proposition of this subsection.

Proposition 3.10. *Given a fixed* $(\mathrm{td} \leqslant d, \varphi)$-MWIS *problem, where* φ *is a* CMSO_2 *formula over the signature of* Σ^0*-annotated graphs, there exists a finite alphabet* Σ*, a* (d, Σ^0, Σ)*-labeller* Λ*, and a threshold automaton* $\mathcal{A}$ *with alphabet* Σ *such that for any partial solution* $(\mathcal{T}, X, \mathtt{Sol})$ *in any* Σ^0*-annotated graph* G*, we have that* $(\mathtt{Sol}, X)$ *is feasible for* $(\mathrm{td} \leqslant d, \varphi)$-MWIS *in* G *if and only if* $\mathcal{A}$ *accepts the* Σ*-labelled forest obtained from* $\mathcal{T}$ *by equipping it with* $\mathsf{label}_{(\mathcal{T},X,\mathtt{Sol})}$*.*

We first prove several lemmas, and then we prove Proposition 3.10 by combining them. It is straightforward to rewrite formulas to obtain the following lemma.

Lemma 3.11. *For any* $d \in \mathbb{N}$*, finite set* Σ^0*, and* CMSO_2 *formula* φ *over the signature of* Σ^0*-annotated graphs with one free vertex set variable, there exists a* CMSO_2 *formula* φ *over the signature of* Σ^0*-annotated graphs with two free vertex set variables such that for any partial solution* $(\mathcal{T}, X, \mathtt{Sol})$ *in any* Σ^0*-annotated graph* G*, we have that* $(\mathtt{Sol}, X)$ *is feasible for* $(\mathrm{td} \leqslant d, \varphi)$-MWIS *in* G *if and only if* $G[\mathcal{T}] \models \varphi(X, \mathtt{Sol})$*.*

We now show how to obtain an alphabet Σ and a (d, Σ^0, Σ)-labeller which lets us get rid of the graph entirely. That is, we will reduce the given sentence to a sentence in CMSO over a Σ-labelled forest.

Lemma 3.12. *For any* $d \in \mathbb{N}$ *and finite set* Σ^0*, there exist a finite alphabet* Σ *and a* (d, Σ^0, Σ)*-labeller* Λ *so that the following holds. For any formula* φ *of* CMSO_2 *over* Σ^0*-annotated graphs with two free vertex set variables, there exists a sentence* $\widehat{\varphi}$ *of* CMSO *over* Σ*-labelled forests such that for any partial solution* $(\mathcal{T}, X, \mathtt{Sol})$ *in any* Σ^0*-annotated graph* G*,*

$$G[\mathcal{T}] \models \varphi(X, \mathtt{Sol}) \qquad \text{if and only if} \qquad \widehat{\mathcal{T}} \models \widehat{\varphi},$$

where $\widehat{\mathcal{T}}$ *is the* Σ*-labelled forest obtained from* $\mathcal{T}$ *by equipping it with* $\mathsf{label}_{(\mathcal{T},X,\mathtt{Sol})}$*.*

Proof. We let $\Sigma = \Sigma^0 \times \{1, \ldots, d\} \times \{0,1\}^{\{1,\ldots,d\}} \times \{0,1\}^2$, where the third coordinate is treated as a function from $\{1, \ldots, d\}$ to $\{0, 1\}$; note that $|\Sigma| = |\Sigma^0| \cdot d \cdot 2^{d+2}$.

Consider a graph G and a partial solution $(\mathcal{T}, X, \mathtt{Sol})$ in G. We now define the (d, Σ^0, Σ)-labeller Λ. Consider any $x \in V(\mathcal{T})$. Let h be the depth of x in $\mathcal{T}$. Let f be the function from $\{1, \ldots, d\}$ to $\{0, 1\}$ defined as follows: for $i \geqslant h$ we set $f(i) = 0$, and for $i < h$ we set $f(i) = 1$ if and only if x is adjacent to the unique ancestor of x in $\mathcal{T}$ that has depth i. Let $\mathbb{1}_X$ and $\mathbb{1}_{\mathtt{Sol}}$ be equal the value 1 if v is in X or $\mathtt{Sol}$, respectively, and 0 otherwise. Then we set

$$\mathsf{label}(x) \coloneqq (\lambda^0(x), h, f, \mathbb{1}_X, \mathbb{1}_{\mathtt{Sol}}).$$

Note that this labelling function can be computed from $G[\mathcal{T}]$ and $(\mathcal{T}, X, \mathtt{Sol})$ in polynomial time. Moreover, this algorithm is a (d, Σ^0, Σ)-labeller. Let $\widehat{\mathcal{T}}$ denote the Σ-labelled forest obtained from $\mathcal{T}$ by equipping it with this labelling.

We now apply the following syntactic transformation to φ in order to obtain a sentence $\widehat{\varphi}$ of CMSO over Σ-labelled forests.

- For every quantification over an edge e, replace it with a quantification over the pair x, y of its endpoints, followed by a check that x and y are indeed adjacent. Since the depth of $\mathcal{T}$ is at most d, which is a constant, this check can be performed using a first-order formula as follows: verify that x and y are in the ancestor-descendant relation in $\widehat{\mathcal{T}}$, retrieve the depth of x and y in $\widehat{\mathcal{T}}$ from their labels, and check that the label of the deeper of those two nodes contains information that the shallower one is adjacent to it.
- Replace each atom expressing that a vertex z is incident to an edge e by a disjunction checking that z is one of the endpoints of e.
- For every quantification over an edge set, say $\exists Y$, replace it with quantification of the form $\exists Y_1\, \exists Y_2\, \ldots\, \exists Y_{d-1}$, where Y_i is interpreted as the set of all the deeper endpoints of those edges from Y whose shallower endpoint has depth i. This quantification is followed by checking that for each $x \in Y_i$, indeed x is adjacent to its unique ancestor at depth i; this information is encoded in the label of x.
- Replace each atom $e \in Y$, where e is an edge variable and Y is an edge set variable, with a disjunction over $i \in \{1, \ldots, d\}$ of the following checks: denoting the endpoints of e by x and y, either x is at depth i and $y \in Y_i$, or vice versa.
- Replace each check $\lambda^0(x) = \sigma$, $x \in X$, or $x \in \texttt{Sol}$ with the corresponding check of the first, fourth, or fifth coordinate of the label of x.

It is straightforward to see that the sentence $\widehat{\varphi}$ obtained in this manner satisfies the desired property. This completes the proof of Lemma 3.12. □

We complete this section by proving Proposition 3.10, which is restated below for convenience.

Proposition 3.10. *Given a fixed* $(\mathrm{td} \leqslant d, \varphi)$-MWIS *problem, where* φ *is a* CMSO_2 *formula over the signature of* Σ^0*-annotated graphs, there exists a finite alphabet* Σ*, a* (d, Σ^0, Σ)*-labeller* Λ*, and a threshold automaton* $\mathcal{A}$ *with alphabet* Σ *such that for any partial solution* $(\mathcal{T}, X, \texttt{Sol})$ *in any* Σ^0*-annotated graph* G*, we have that* $(\texttt{Sol}, X)$ *is feasible for* $(\mathrm{td} \leqslant d, \varphi)$-MWIS *in* G *if and only if* $\mathcal{A}$ *accepts the* Σ*-labelled forest obtained from* $\mathcal{T}$ *by equipping it with* $\mathsf{label}_{(\mathcal{T},X,\texttt{Sol})}$.

Proof. Fix d, Σ^0, and φ. By Lemma 3.11, there exists a CMSO_2 formula φ over the signature of Σ^0-annotated graphs with two free vertex set variables such that for any partial solution $(\mathcal{T}, X, \texttt{Sol})$ in any graph G, we have that $(\texttt{Sol}, X)$ is feasible for $(\mathrm{td} \leqslant d, \varphi)$-MWIS in G if and only if $G[\mathcal{T}] \models \varphi(X, \texttt{Sol})$. By Lemma 3.12, there exist a finite alphabet Σ, a (d, Σ^0, Σ)-labeller Λ, and a sentence $\widehat{\varphi}$ of CMSO over Σ-labelled forests such that for any partial solution $(\mathcal{T}, X, \texttt{Sol})$ in any Σ^0-annotated graph G,

$$G[\mathcal{T}] \models \varphi(X, \texttt{Sol}) \qquad \text{if and only if} \qquad \widehat{\mathcal{T}} \models \widehat{\varphi},$$

where $\widehat{\mathcal{T}}$ is the Σ-labelled forest obtained from $\mathcal{T}$ by equipping it with $\mathsf{label}_{(\mathcal{T},X,\texttt{Sol})}$.

Finally, by Lemma 3.8, there exists a threshold automaton $\mathcal{A}$ with alphabet Σ such that for any Σ-labelled forest F, we have $F \models \widehat{\varphi}$ if and only if $\mathcal{A}$ accepts F. Proposition 3.10 follows. □

3.3 The algorithm

Fix an integer d, a finite set Σ^0, and a CMSO_2 formula φ over the signature of Σ^0-annotated graphs. By Proposition 3.10, there exists a finite alphabet Σ, a (d, Σ^0, Σ)-labeller Λ, and a threshold automaton $\mathcal{A} = (Q_{\mathcal{A}}, \Sigma, \tau_{\mathcal{A}}, \delta_{\mathcal{A}}, C_{\mathcal{A}})$ such that for any partial solution $(\mathcal{T}, X, \texttt{Sol})$ in any Σ^0-annotated graph (G, λ^0), we have that $(\texttt{Sol}, X)$ is feasible for $(\mathrm{td} \leqslant d, \varphi)$-MWIS in (G, λ^0) if and only if $\mathcal{A}$ accepts the Σ-labelled forest obtained from $\mathcal{T}$ by equipping it with the labelling $\mathsf{label}_{(\mathcal{T},X,\texttt{Sol})}$. The algorithm will make use of Σ, Λ, and $\mathcal{A}$.

For convenience, we say that a *multistate assignment* of a rooted forest F is any function $\xi : \{\emptyset\} \cup V(F) \to \mathsf{Multi}(Q_{\mathcal{A}}, \tau_{\mathcal{A}})$. Consider a multistate assignment ξ of a tree-depth-d structure $\mathcal{T}$. Essentially, we use ξ to specify the desired behavior of an extension of a partial solution $(\mathcal{T}, X, \mathtt{Sol})$. In order to combine two extensions, sometimes we need to combine two multistate assignments ξ_1 and ξ_2 of a rooted forest F. So we write $\xi_1 \cup \xi_2$ for the multistate assignment of F defined by setting $(\xi_1 \cup \xi_2)(v) \coloneqq (\xi_1(v) \cup \xi_2(v)) \wedge \tau_{\mathcal{A}}$ for each $v \in \{\emptyset\} \cup V(F)$.

Let $\mathcal{C} \subseteq 2^{V(G)}$ be a tree-depth-d carver family of defect k in G. A *template* is a tuple $\sigma = (\mathcal{T}, X, \mathtt{Sol}, C, D, \xi)$ such that

(i) $(\mathcal{T}, X, \mathtt{Sol})$ is a partial solution in G,

(ii) $C \in \mathcal{C}$,

(iii) D is a subset of $V(G)$ which is a union of zero or more components of $G - C$, and

(iv) ξ is a multistate assignment of $\mathcal{T}$.

We say that σ is *simple* if $\mathcal{T}$ has at most k leaves and D is a component of $G - C$. A *(simple) pre-template* is a tuple $\alpha = (\mathcal{T}, X, \mathtt{Sol}, C)$ as in the definition of a (simple) template, except with D and ξ omitted. We say that a template $\sigma = (\mathcal{T}, X, \mathtt{Sol}, C, D, \xi)$ is *over* the pre-template $(\mathcal{T}, X, \mathtt{Sol}, C)$.

The dynamic programming algorithm stores a table M that has an entry $M[\sigma]$ for each simple template σ. We observe that the table has $\mathcal{O}(|\mathcal{C}| \cdot |V(G)|^{dk+1})$ entries, where the constant hidden in the big-$\mathcal{O}$ notation depends on d, $|\Sigma^0|$, k, and φ. We initiate the value of each entry $M[\sigma]$ to a symbol $\perp$. As the algorithm proceeds, $M[\sigma]$ will be updated to contain a partial solution $(\mathcal{T}', X', \mathtt{Sol}')$ which is a "valid extension" (defined formally in the next paragraph) of σ. We only update $M[\sigma]$ when we discover a new valid extension better than the old one according to $\preceq$; we use the convention that every valid extension is better than $\perp$.

Now, let $\sigma = (\mathcal{T}, X, \mathtt{Sol}, C, D, \xi)$ be a template (which may or may not be simple). Then a *valid extension* of σ is any extension $(\mathcal{T}', X', \mathtt{Sol}')$ of $(\mathcal{T}, X, \mathtt{Sol})$ such that $V(\mathcal{T}') \setminus V(\mathcal{T}) \subseteq D$ and, if $\xi' : V(\mathcal{T}') \to Q_{\mathcal{A}}$ denotes the run of $\mathcal{A}$ on the Σ-labelled forest obtained from $\mathcal{T}'$ by equipping it with $\mathsf{label}_{(\mathcal{T}', X', \mathtt{Sol}')}$, then

$$\{\{\xi'(z) \mid z \text{ is a root of } \mathcal{T}' \text{ but not of } \mathcal{T}\}\} \wedge \tau_{\mathcal{A}} = \xi(\emptyset),$$

and, for every $v \in V(\mathcal{T})$,

$$\{\{\xi'(z) \mid z \text{ is a child of } v \text{ in } \mathcal{T}' \text{ but not in } \mathcal{T}\}\} \wedge \tau_{\mathcal{A}} = \xi(v).$$

Note that if z is a child of v in $\mathcal{T}'$ but not in $\mathcal{T}$, then $z \notin V(\mathcal{T})$ since, if it was, then it would have the same parent in $\mathcal{T}'$ and $\mathcal{T}$ by the definition of extensions.

The following observation about combining extensions is the crucial building block of the algorithm. To state the lemma, we need to know when we can combine two tree-depth-d structures $\mathcal{T}$ and $\mathcal{T}'$ in a graph G. So we say that $\mathcal{T}$ and $\mathcal{T}'$ are *compatible* if the sets $V(\mathcal{T}) \setminus V(\mathcal{T}')$ and $V(\mathcal{T}') \setminus V(\mathcal{T})$ are anticomplete in G, and each vertex in $V(\mathcal{T}) \cap V(\mathcal{T}')$ has the same parent in $\mathcal{T}$ and $\mathcal{T}'$. (We think of the empty set as being the parent of a root; so in particular this means that every ancestor of a vertex in $V(\mathcal{T}) \cap V(\mathcal{T}')$ is also in $V(\mathcal{T}) \cap V(\mathcal{T}')$.) If $\mathcal{T}$ and $\mathcal{T}'$ are compatible, then there is a unique tree-depth-d structure, which we denote by $\mathcal{T} \cup \mathcal{T}'$, such that:

(i) the vertex set of $\mathcal{T} \cup \mathcal{T}'$ is $V(\mathcal{T}) \cup V(\mathcal{T}')$,

(ii) each vertex in $V(\mathcal{T})$ has the same parent in $\mathcal{T} \cup \mathcal{T}'$ and $\mathcal{T}$, and

(iii) each vertex in $V(\mathcal{T}')$ has the same parent in $\mathcal{T} \cup \mathcal{T}'$ and $\mathcal{T}'$.

Note that we can check if $\mathcal{T}$ and $\mathcal{T}'$ are compatible, and find $\mathcal{T} \cup \mathcal{T}'$ if they are, in polynomial time. Now we are ready to state the key lemma.

Lemma 3.13. *Let $\sigma_1 = (\mathcal{T}, X, \mathtt{Sol}, C, D_1, \xi_1)$ and $\sigma_2 = (\mathcal{T}, X, \mathtt{Sol}, C, D_2, \xi_2)$ be two templates over the same pre-template. Suppose that D_1 and D_2 are disjoint and that $(\mathcal{T}_i, X_i, \mathtt{Sol}_i)$ is a valid extension of σ_i for $i = 1, 2$. Then $\mathcal{T}_1$ and $\mathcal{T}_2$ are compatible and $(\mathcal{T}_1 \cup \mathcal{T}_2, X_1 \cup X_2, \mathtt{Sol}_1 \cup \mathtt{Sol}_2)$ is a valid extension of $(\mathcal{T}, X, \mathtt{Sol}, C, D_1 \cup D_2, \xi_1 \cup \xi_2)$.*

Moreover, if for $i = 1, 2$, $(\mathcal{T}'_i, X'_i, \mathtt{Sol}'_i)$ is a valid extension of σ_i which is not worse than $(\mathcal{T}_i, X_i, \mathtt{Sol}_i)$, then $(\mathcal{T}'_1 \cup \mathcal{T}'_2, X'_1 \cup X'_2, \mathtt{Sol}'_1 \cup \mathtt{Sol}'_2)$ is not worse than $(\mathcal{T}_1 \cup \mathcal{T}_2, X_1 \cup X_2, \mathtt{Sol}_1 \cup \mathtt{Sol}_2)$.

Proof. Observe that since D_1 and D_2 are disjoint and each of them is a union of components of $G - C$, they are also anticomplete. So the sets $V(\mathcal{T}_1) \setminus V(\mathcal{T})$ and $V(\mathcal{T}_2) \setminus V(\mathcal{T})$ are also disjoint and anticomplete. So $V(\mathcal{T}_1) \cap V(\mathcal{T}_2) = V(\mathcal{T})$ and, by the definition of extensions, it follows that $\mathcal{T}_1$ and $\mathcal{T}_2$ are compatible and that $(\mathcal{T}_1 \cup \mathcal{T}_2, X_1 \cup X_2, \mathtt{Sol}_1 \cup \mathtt{Sol}_2)$ is an extension of the partial solution $(\mathcal{T}, X, \mathtt{Sol})$. It is also clear that $V(\mathcal{T}_1 \cup \mathcal{T}_2) \setminus V(\mathcal{T})$ is a subset of $D_1 \cup D_2$.

Now it just remains to consider the run ξ' of $\mathcal{A}$ on the Σ-labelled forest obtained from $\mathcal{T}_1 \cup \mathcal{T}_2$ by equipping it with the labelling $\mathsf{label}_{(\mathcal{T}_1 \cup \mathcal{T}_2, X_1 \cup X_2, \mathtt{Sol}_1 \cup \mathtt{Sol}_2)}$. For this, observe that every component of the graph $\mathcal{T}_1 \cup \mathcal{T}_2 - V(\mathcal{T})$ is either a component of $\mathcal{T}_1 - V(\mathcal{T})$ or a component of $\mathcal{T}_2 - V(\mathcal{T})$. Hence, for $i = 1, 2$, the function ξ' gives the same state to each vertex in $V(\mathcal{T}_i) \setminus V(\mathcal{T})$ as does the run of $\mathcal{A}$ on $\mathcal{T}_i$ and $\mathsf{label}_{(\mathcal{T}_i, X_i, \mathtt{Sol}_i)}$. The first part of the lemma now follows from the fact that, for any disjoint multisets A and B whose elements are in $Q_{\mathcal{A}}$, we have that $(A \cup B) \wedge \tau_{\mathcal{A}} = ((A \wedge \tau_{\mathcal{A}}) \cup (B \wedge \tau_{\mathcal{A}})) \wedge \tau_{\mathcal{A}}$.

The second part of the lemma follows immediately from the used total ordering of partial solutions. □

3.3.1 Subroutine

Given as input a simple pre-template $(\mathcal{T}, X, \mathtt{Sol}, C)$ and a sequence $(D_i)_{i=1}^r$ of pairwise distinct components of $G - C$, we define the following subroutine. For each $j \in \{0, 1, \ldots, r\}$, we set $D_{\leqslant j} := \bigcup_{i=1}^{j} D_i$. (So $D_{\leqslant 0}$ is the empty set.) The subroutine creates an auxiliary table M' with an entry $M'[j, \xi]$ for every $j \in \{0, 1, \ldots, r\}$ and every multistate assignment ξ of $\mathcal{T}$. Each entry $M'[j, \xi]$ will be either the symbol $\perp$, or a valid extension of the template $(\mathcal{T}, X, \mathtt{Sol}, C, D_{\leqslant j}, \xi)$. Initially all cells are set to $\perp$.

For $j = 0$, there is only one multistate assignment ξ of $\mathcal{T}$ such that the template $(\mathcal{T}, X, \mathtt{Sol}, C, \emptyset, \xi)$ might have a valid extension, and that is the function $\xi \equiv \emptyset$. The unique valid extension is $(\mathcal{T}, X, \mathtt{Sol})$; so we set $M'[0, \xi \equiv \emptyset] := (\mathcal{T}, X, \mathtt{Sol})$. Then, for $j = 1, 2, \ldots, r$, we fill the cells $M'[j, \cdot]$ as follows. We iterate over all multistate assignments $\xi_<$ and $\xi_=$ of $\mathcal{T}$, and, if neither $M'[j-1, \xi_<]$ nor $M[(\mathcal{T}, X, \mathtt{Sol}, C, D_j, \xi_=)]$ is $\perp$, then we apply Lemma 3.13 to combine them into a valid extension $(\mathcal{T}', X', \mathtt{Sol}')$ of $(\mathcal{T}, X, \mathtt{Sol}, C, D_{\leqslant j}, \xi_< \cup \xi_=)$. If this extension is better than the previous value of $M'[j, \xi_< \cup \xi_=]$, then we set $M'[j, \xi_< \cup \xi_=] := (\mathcal{T}', X', \mathtt{Sol}')$. This finishes the description of the subroutine.

3.3.2 Outline

In a preliminary phase, the algorithm iterates over every simple template $\sigma = (\mathcal{T}, X, \mathtt{Sol}, C, D, \xi)$ such that $\xi \equiv \emptyset$. Then it sets $M[\sigma] := (\mathcal{T}, X, \mathtt{Sol})$; note that $(\mathcal{T}, X, \mathtt{Sol})$ is a valid extension of σ which is better than $\perp$.

In the main phase, the algorithm performs $2|V(G)|$ loops. In each loop, it iterates over every simple template $\sigma = (\mathcal{T}, X, \mathtt{Sol}, C, D, \xi)$ and simple pre-template $\alpha_0 = (\mathcal{T}_0, X_0, \mathtt{Sol}_0, C_0)$ such that $\mathcal{T}$ and $\mathcal{T}_0$ are compatible, $X \cap \mathcal{T} \cap \mathcal{T}_0 = X_0 \cap \mathcal{T} \cap \mathcal{T}_0$, and $\mathtt{Sol} \cap \mathcal{T} \cap \mathcal{T}_0 = \mathtt{Sol}_0 \cap \mathcal{T} \cap \mathcal{T}_0$. The algorithm will try to find a valid extension of σ which is better than $M[\sigma]$. The building blocks for constructing this valid extension of σ will be the valid extensions $M[\sigma_0]$ where σ_0 is a simple template over α_0. In fact we will be slightly more restrictive about which components of $G - C_0$ we are allowed to "extend α_0 into".

We call a component D_0 of $G - C_0$ *useless* if $\mathcal{T} \cup \mathcal{T}_0$ is a maximal tree-depth-d structure in the subgraph of G induced by $V(\mathcal{T}) \cup V(\mathcal{T}_0) \cup (D \cap D_0)$; we call D_0 *useful* otherwise. Note

that if D_0 is useful, then in particular $D \cap D_0$ is non-empty. We now execute the subroutine on the simple pre-template α_0 and the useful components of $G - C_0$, ordered arbitrarily. (If there are no useful components, then we still execute the subroutine on the empty sequence.) The subroutine returns an array M'. Write r for the number of useful components of $G - C_0$ and $U \subseteq V(G)$ for their union. Then iterate over all multistate functions ξ_0 of $\mathcal{T}_0$ such that $M'[r, \xi_0] \neq \bot$. Thus $M'[r, \xi_0]$ is a valid extension of $(\mathcal{T}_0, X_0, \mathtt{Sol}_0, U, \xi_0)$, which we denote by $(\mathcal{T}_0', X_0', \mathtt{Sol}_0')$.

Now, let A denote the set of all vertices of $\mathcal{T}_0'$ which are an ancestor, in $\mathcal{T}_0'$, of at least one vertex in D. If $\mathcal{T}_0'[A]$ and $\mathcal{T}$ are compatible, and if the tuple

$$(\mathcal{T}_0'[A] \cup \mathcal{T}, (X_0' \cap A) \cup X, (\mathtt{Sol}_0' \cap A) \cup \mathtt{Sol})$$

is a valid extension of σ, then update $M[\sigma]$ to the above if it is better than the previous value of $M[\sigma]$. This can be done in polynomial time.

After completing the main phase consisting of $2|V(G)|$ loops as above, the algorithm performs the following finalizing step, which is very similar to the above routine except without σ. So, for every simple pre-template $\alpha = (C, \mathcal{T}, X, \mathtt{Sol})$, we execute the subroutine on α and the components of $G - C$ in an arbitrary order. The subroutine returns an array M'. Then, writing r for the number of components of $G - C$, we iterate over all multistate functions ξ of $\mathcal{T}$ such that $M'[r, \xi] \neq \bot$. We then check if the valid extension $M'[r, \xi]$ is a feasible solution to the problem. That is, if $M'[r, \xi] = (\mathcal{T}', X', \mathtt{Sol}')$, we check whether $\mathcal{A}$ accepts the Σ-labelled forest obtained from $\mathcal{T}'$ by equipping it with the labelling $\mathsf{label}_{(\mathcal{T}', X', \mathtt{Sol}')}$. By Proposition 3.10, this is equivalent to $(\mathtt{Sol}', X')$ being feasible for $(\mathrm{td} \leqslant d, \varphi)$-MWIS in G. Finally, we return the best solution found, or that there is no solution if none was found.

This concludes the description of the algorithm. Clearly, it runs in $\mathcal{O}(|\mathcal{C}|^2 \cdot |V(G)|^{2dk+\mathcal{O}(1)})$ time. It remains to prove correctness.

3.4 Correctness

We may assume that the $(\mathrm{td} \leqslant d, \varphi)$-MWIS problem is feasible since the algorithm checks for feasibility before returning a solution. So there exists a partial solution $(\mathcal{T}, X, \mathtt{Sol})$ which is $\preceq$-minimal among all partial solutions $(\mathcal{T}', X', \mathtt{Sol}')$ such that $(\mathtt{Sol}', X')$ is feasible for $(\mathrm{td} \leqslant d, \varphi)$-MWIS in G. By Lemma 3.5, we have that $\mathcal{T}$ is maximal and neat, and X has maximum possible weight among all feasible solution for $(\mathrm{td} \leqslant d, \varphi)$-MWIS in G. By Lemma 3.4, there is no other partial solution in the same equivalence class of $\preceq$ as $(\mathcal{T}, X, \mathtt{Sol})$.

Since $\mathcal{C}$ is a tree-depth-d carver family of defect k in G, there exists a tree decomposition (T, β) of G as in Definition 3.1. That is, for each $t \in V(T)$, we can fix a set of vertices $C_t \in \mathcal{C}$ such that *(i)* $C_t \cap \mathcal{T}$ contains $\beta(t) \cap \mathcal{T}$ and has size at most k, and *(ii)* for each component D of $G - C_t$, there exists a component T' of $T - \{t\}$ such that D is contained in $\beta(t) \cup \bigcup_{s \in T'} \beta(s)$.

We root T in an arbitrary node. We argue that without loss of generality we can assume that

$$\mathrm{height}(T) \leqslant 2|V(G)|. \tag{2}$$

Indeed, it is straightforward to verify that one can perform the following operations on (T, β) exhaustively:

- *(i)* if there exists a leaf t of T with a parent t' such that $\beta(t) \subseteq \beta(t')$, delete t from (T, β);
- *(ii)* if there exists a node t of T with only one child t' such that $\beta(t) \subseteq \beta(t')$, contract the edge tt' in (T, β).

If none of this operation is applicable to (T, β), then T has at most $|V(G)|$ leaves and for every edge between a parent t and a child t', either t has more than one children, or $\beta(t) \setminus \beta(t') \neq \emptyset$. There are at most $|V(G)| - 1$ vertices t of the first type in T and on every root-to-leaf path there are at most $|V(G)|$ edges of the second type. The bound (2) follows.

Consider now a fixed node $t \in V(T)$. We say that a *child component of* t is any component of $G - C_t$ which is contained in the union of all bags $\beta(s)$ such that s is a descendant of t in T (including s itself). We define a partial solution $(\mathcal{T}_t, X_t, \mathtt{Sol}_t)$ corresponding to t as follows. Let $\mathcal{T}_t$ be the subgraph of $\mathcal{T}$ induced by all vertices which are an ancestor of at least one vertex in C_t. Set $X_t := X \cap \mathcal{T}_t$ and $\mathtt{Sol}_t := \mathtt{Sol} \cap \mathcal{T}_t$. It is convenient to write $\alpha_t := (\mathcal{T}_t, X_t, \mathtt{Sol}_t, C_t)$; so α_t is a simple pre-template. Finally, let h_t be the height of t in the subtree of T rooted at t; so the leaves of T have $h_t = 1$, for instance.

We will show that after h_t iterations of the algorithm, the following holds for each child component D of t: there exists a multistate function $\xi_{t,D}$ of $\mathcal{T}_t$ such that $M[(\alpha, D, \xi_{t,D})]$ is precisely the partial solution "induced by the ancestors of $D \cup C_t$ in $(\mathcal{T}, X, \mathtt{Sol})$." This lemma, which is stated as Lemma 3.15, will essentially complete the proof. (After $|V(G)|$ rounds, we will consider the child components of the root node of T.) However, it is convenient to give some more definitions before stating the lemma.

So consider a fixed node $t \in V(T)$ and a fixed set $D \subseteq V(G)$ which is the union of zero or more components of $G - C_t$. First we define a partial solution $(\mathcal{T}_{t,D}, X_{t,D}, \mathtt{Sol}_{t,D})$ as follows. Let $\mathcal{T}_{t,D}$ be the subgraph of $\mathcal{T}$ induced by all vertices which are an ancestor of at least one vertex in $D \cup C_t$. Set $X_{t,D} := X \cap \mathcal{T}_{t,D}$ and $\mathtt{Sol}_{t,D} := \mathtt{Sol} \cap \mathcal{T}_{t,D}$. We note that $V(\mathcal{T}_{t,D}) \setminus V(\mathcal{T}_t)$ is actually contained in D. To see this, observe that by Lemma 3.6, since $\mathcal{T}$ is neat and extends $\mathcal{T}_t$, each component of $\mathcal{T} - V(\mathcal{T}_t)$ induces a connected subgraph of G. Therefore each component of $\mathcal{T} - V(\mathcal{T}_t)$ is either disjoint from or contained in D.

Finally, let $\xi_{t,D}$ denote the multistate function of $\mathcal{T}_t$ defined as follows. If ξ is the run of $\mathcal{A}$ on $\mathcal{T}_{t,D}$ equipped with $\mathsf{label}_{(\mathcal{T}_{t,D}, X_{t,D}, \mathtt{Sol}_{t,D})}$, then we set:

$$\xi_{t,D}(\emptyset) := \{\{\xi(z) \mid z \text{ is a root of } \mathcal{T}_{t,D} \text{ but not of } \mathcal{T}_t\}\} \wedge \tau_{\mathcal{A}},$$

and, for every $v \in V(\mathcal{T}_t)$,

$$\xi_{t,D}(v) = \{\{\xi(z) \mid z \text{ is a child of } v \text{ in } \mathcal{T}_{t,D} \text{ but not in } \mathcal{T}_t\}\} \wedge \tau_{\mathcal{A}}.$$

Notice that $(\mathcal{T}_{t,D}, X_{t,D}, \mathtt{Sol}_{t,D})$ is a valid extension of $(\alpha_t, D, \xi_{t,D})$; denote the latter by $\sigma_{t,D}$. So, if D is a component of $G - C_t$, then $\sigma_{t,D}$ is a simple template.

Our tie-breaking quasi-order and the choice of $(\mathcal{T}, X, \mathtt{Sol})$ imply that, in fact, $(\mathcal{T}_{t,D}, X_{t,D}, \mathtt{Sol}_{t,D})$ is the unique $\preceq$-minimal valid extension of $\sigma_{t,D}$.

Lemma 3.14. *Let $t \in V(T)$ and let $D \subseteq V(G)$ be the union of zero or more components of $G - C_t$. Then $(\mathcal{T}_{t,D}, X_{t,D}, \mathtt{Sol}_{t,D})$ is the only valid extension of $\sigma_{t,D}$ which is not worse than $(\mathcal{T}_{t,D}, X_{t,D}, \mathtt{Sol}_{t,D})$.*

Proof. Let $(\mathcal{T}', X', \mathtt{Sol}')$ be a valid extension of $\sigma_{t,D}$ which is not worse than $(\mathcal{T}_{t,D}, X_{t,D}, \mathtt{Sol}_{t,D})$. Let D_0 denote the union of all components of $G - C_t$ which are not in D. We already observed that $(\mathcal{T}_{t,D_0}, X_{t,D_0}, \mathtt{Sol}_{t,D_0})$ is a valid extension of σ_{t,D_0}. So, since D and D_0 are disjoint, Lemma 3.13 tells us that $\mathcal{T}'$ and $\mathcal{T}_{t,D_0}$ are compatible, and that the component-wise union of $(\mathcal{T}', X', \mathtt{Sol}')$ and $(\mathcal{T}_{t,D_0}, X_{t,D_0}, \mathtt{Sol}_{t,D_0})$ is a valid extension of $(\alpha_t, D \cup D_0, \xi_{t,D} \cup \xi_{t,D_0})$. Also by Lemma 3.13, this valid extension is not worse than the component-wise union of $(\mathcal{T}_{t,D}, X_{t,D}, \mathtt{Sol}_{t,D})$ and $(\mathcal{T}_{t,D_0}, X_{t,D_0}, \mathtt{Sol}_{t,D_0})$. The latter equals $(\mathcal{T}, X, \mathtt{Sol})$ and is a valid extension of that same template $(\alpha_t, D \cup D_0, \xi_{t,D} \cup \xi_{t,D_0})$.

In general, the runs of $\mathcal{A}$ on any two valid extensions of the same template are the same. By Proposition 3.10, the run of $\mathcal{A}$ determines whether a partial solution yields a solution to $(\mathrm{td} \leqslant d, \varphi)$-MWIS on G. So, by the choice of $(\mathcal{T}, X, \mathtt{Sol})$ and by Lemma 3.4 applied to the tree $\mathcal{T}$, which is neat, we find that

$$(\mathcal{T}' \cup \mathcal{T}_{t,D_0}, X' \cup X_{t,D_0}, \mathtt{Sol}' \cup \mathtt{Sol}_{t,D_0}) = (\mathcal{T}, X, \mathtt{Sol}).$$

It follows that $(\mathcal{T}', X', \mathtt{Sol}') = (\mathcal{T}_{t,D}, X_{t,D}, \mathtt{Sol}_{t,D})$, as desired. □

We are now ready to prove the main lemma.

Lemma 3.15. *Let $t \in V(T)$, and assume that at least h_t iterations of the algorithm have been executed. Then for any child component D of t, we have $M[\sigma_{t,D}] = (\mathcal{T}_{t,D}, X_{t,D}, \texttt{Sol}_{t,D})$. Furthermore, if the subroutine is executed on α_t and any sequence of child components of t, then, where we write M' for the array which is returned, r for the number of child components under consideration, and U for their union, we have $M'[r, \xi_{t,U}] = (\mathcal{T}_{t,U}, X_{t,U}, \texttt{Sol}_{t,U})$.*

Proof. We may assume that the lemma holds for every child of t by induction on h_t. We will argue about the first claim of the lemma for the node t. Note that the second claim follows from the first claim and Lemmas 3.13 and 3.14 (using induction on r). So, fix a child component D of t. Note that we only have to show that $M[\sigma_{t,D}]$ is set to $(\mathcal{T}_{t,D}, X_{t,D}, \texttt{Sol}_{t,D})$ at some point; Lemma 3.14 implies that, once this occurs, $M[\sigma_{t,D}]$ is never changed.

For the base case of $h_t = 1$, we have that t is a leaf of T and $D \subseteq \beta(t)$. So, since $C_t \cap \mathcal{T}$ contains $\beta(t) \cap \mathcal{T}$ by the definition of a carver family, the set $D \cap \mathcal{T}$ is empty. Thus $\mathcal{T}_{t,D} = \mathcal{T}_t$ and $\xi_{t,D} \equiv \emptyset$. It follows that, in the preliminary phase, we set $M[\sigma_{t,D}] = (\mathcal{T}_t, X_t, \texttt{Sol}_t)$, as desired. So we may assume that $h_t > 1$.

Thus, using the definition of a carver family, there exists a child s of t in T such that we have $D \subseteq \beta(t) \cup \bigcup_{t' \in T'} \beta(t')$, where T' denotes the component of $T - \{t\}$ which contains s. (If $D \subseteq \beta(t)$, then there may be more than one such vertex s, and we choose s arbitrarily.) We focus on the h_t-th iteration of the algorithm and the moment when the algorithm considers the simple template $\sigma_{t,D}$ and the simple pre-template α_s. Note that $\mathcal{T}_t$ and $\mathcal{T}_s$ are compatible, and that $\mathcal{T}_t \cup \mathcal{T}_s$ is precisely the subgraph of $\mathcal{T}$ induced by the ancestors of $C_t \cup C_s$. Recall that a component D_s of $G - C_s$ is *useful* if $\mathcal{T}_t \cup \mathcal{T}_s$ is not a maximal tree-depth-d structure in the subgraph of G induced by $V(\mathcal{T}_t) \cup V(\mathcal{T}_s) \cup (D \cap D_s)$.

We need the following key observation.

Claim 3.15.1. *Every useful component of $G - C_s$ is a child component of s.*

Proof of Claim. Suppose towards a contradiction that D_s is a useful component of $G - C_s$ which is not a child component of s. Then the definition of a carver family tells us that $D_s \subseteq \beta(s) \cup \bigcup_{t' \in T'} \beta(t')$, where T' is the component of $T - \{s\}$ which contains t. Since D is a child component of t, we have that $D \cap D_s \subseteq \beta(t) \cup \beta(s)$.

Since $D \cap D_s$ is disjoint from $C_t \cup C_s$, and the latter contains all vertices of $(\beta(t) \cup \beta(s)) \cap \mathcal{T}$, we also have that $D \cap D_s$ is disjoint from $V(\mathcal{T})$. Furthermore, $D \cap D_s$ is the union of some subset of components of $G - (C_t \cup C_s)$. By Lemma 3.6, since $\mathcal{T}$ is neat, each component of $\mathcal{T} - (V(\mathcal{T}_t) \cup V(\mathcal{T}_s))$ induces a connected subgraph of G; so the vertex set of each such component is either contained in or disjoint from $D \cap D_s$. Hence, the maximality of $\mathcal{T}$ implies that $\mathcal{T}_t \cup \mathcal{T}_s$ is also a maximal tree-depth-d structure in the subgraph of G induced by $V(\mathcal{T}_t) \cup V(\mathcal{T}_s) \cup (D \cap D_s)$. This contradicts the fact that D_s is useful. ⌟

As in the outline of the algorithm, let $D_1, \ldots, D_r$ be the useful components of $G - C_s$, in an arbitrary order. Claim 3.15.1 implies that every D_j is a child component of s. Hence, from the inductive hypothesis, at the beginning of the h_t-th iteration we have, for every $1 \leqslant j \leqslant r$, that $M[\sigma_{s,D_j}] = (\mathcal{T}_{s,D_j}, X_{s,D_j}, \texttt{Sol}_{s,D_j})$. We now claim the following.

Claim 3.15.2. *In the run of the subroutine, we have for every $0 \leqslant j \leqslant r$, that*

$$M'[j, \xi_{s,D_{\leqslant j}}] = (\mathcal{T}_{s,D_{\leqslant j}}, X_{s,D_{\leqslant j}}, \texttt{Sol}_{s,D_{\leqslant j}}).$$

Proof of Claim. We prove the claim by induction on j. For $j = 0$ the claim holds since $\xi_{s,\emptyset} \equiv \emptyset$ and thus $M'[0, \xi_{s,\emptyset}] = (\mathcal{T}_s, X_s, \texttt{Sol}_s)$. For $j > 0$, from the inductive hypothesis on j we have $M'[j-1, \xi_{s,D_{\leqslant j-1}}] = (\mathcal{T}_{s,D_{\leqslant j-1}}, X_{s,D_{\leqslant j-1}}, \texttt{Sol}_{s,D_{\leqslant j-1}})$ and from before, we have $M[\sigma_{s,D_j}] = (\mathcal{T}_{s,D_j}, X_{s,D_j}, \texttt{Sol}_{s,D_j})$. Hence, the partial solution $(\mathcal{T}_{s,D_{\leqslant j}}, X_{s,D_{\leqslant j}}, \texttt{Sol}_{s,D_{\leqslant j}})$ is considered for

$M'[j, \xi_{s,D_{\leqslant j}}]$; Lemma 3.14 ensures that it is assigned there and stays till the end. This proves the claim. ⌟

After the subroutine is executed, the algorithm iterates over every multistate function ξ_s of $\mathcal{T}_s$ and attempts to use $M'[r, \xi_s]$ to find a better valid extension of $\sigma_{t,D}$ than $M[\sigma_{t,D}]$. By Lemma 3.14, it suffices to prove that when $\xi_{s,D\leqslant r}$ is considered, the resulting valid extension $(\mathcal{T}_{t,D}, X_{t,D}, \mathtt{Sol}_{t,D})$ of $\sigma_{t,D}$ is found.

By Claim 3.15.2 we have $M'[r, \xi_{s,D_{\leqslant r}}] = (\mathcal{T}_{s,D_{\leqslant r}}, X_{s,D_{\leqslant r}}, \mathtt{Sol}_{s,D_{\leqslant r}})$. As in the outline of the algorithm, let A denote the set of all vertices of $\mathcal{T}_{s,D_{\leqslant r}}$ which are an ancestor of at least one vertex in D. Note that $\mathcal{T}_{s,D_{\leqslant r}}[A] = \mathcal{T}[A]$, that $\mathcal{T}[A]$ and $\mathcal{T}_t$ are compatible, and that $\mathcal{T}[A] \cup \mathcal{T}_t$ is precisely the subtree of $\mathcal{T}$ induced by the ancestors of vertices in $D \cap (D_{\leqslant r} \cup C_s)$ and C_t. Thus, it just remains to show that this induced subtree is $\mathcal{T}_{t,D}$, or, equivalently, that every vertex in $(D \cap \mathcal{T}) \setminus (C_s \cup C_t)$ is in a useful component of $G - C_s$ (i.e., in $D_{\leqslant r}$). This holds by the definition of useful components, because such a vertex can be added to the tree-depth-d structure $\mathcal{T}_s \cup \mathcal{T}_t$. This finishes the proof of Lemma 3.15. □

By (2), after $2|V(G)|$ iterations, Lemma 3.15 can be applied to the root of T, which we denote by t. Consider now the finalizing step of the algorithm and the moment it considers the pre-template α_t. Let M' denote the computed array, r the number of components of $G - C_t$, and U the set $V(G) \setminus C_t$. Since every component of $G - C_t$ is a child component of t, Lemma 3.15 implies that $M'[r, \xi_{t,U}] = (\mathcal{T}_{t,U}, X_{t,U}, \mathtt{Sol}_{t,U})$. So, as $U = V(G) \setminus C_t$, we have $\mathcal{T}_{t,U} = \mathcal{T}$ and thus $M'[r, \xi_{t,U}] = (\mathcal{T}, X, \mathtt{Sol})$. As $(\mathcal{T}, X, \mathtt{Sol})$ is the unique $\preceq$-minimal partial solution such that $(\mathtt{Sol}, X)$ is feasible for $(\mathrm{td} \leqslant d, \varphi)$-MWIS in G, the algorithm returns $(\mathcal{T}, X, \mathtt{Sol})$.

This finishes the proof of Theorem 3.2.

4 Minimal separator carving

Given a graph G and a minimal separator S of G, we say that a set $\widetilde{S}$ *carves away* a component D of $G - S$ if no component of $G - \widetilde{S}$ intersects both D and another component of $G - S$. (We say that two sets *intersect* if their intersection is non-empty.) In this section we find "carvers" for minimal separators. We break up minimal separators into four different types based on which of their full components can be carved away.

First of all, a minimal separator S is *subordinate* if there exists a minimal separator S' and two full sides A' and B' of S' such that $S \subseteq S'$ and some full component of S is disjoint from $A' \cup S' \cup B'$. Notice that any minimal separator which is not subordinate has exactly two full sides; otherwise we could take $S' = S$ and A' and B' to be two full components of S.

The other three types of minimal separator are based on how many full components are "mesh". A graph H is *mesh* if its complement $\overline{H}$ is not connected. Otherwise $\overline{H}$ is connected, and we call H *non-mesh*. We say that a minimal separator S is *mesh*/*mixed*/*non-mesh* (respectively) if S is not subordinate and has exactly $2/1/0$ full components which are mesh.

Now we define carvers for minimal separators based on their type.

Definition 4.1. Let G be a graph, d be a positive integer, $\mathcal{T}$ be a treedepth-d structure in G, and let S be a $\mathcal{T}$-avoiding minimal separator of G. Then a *$\mathcal{T}$-carver for S* is a set $\widetilde{S} \subseteq V(G)$ such that $\widetilde{S} \cap \mathcal{T} = S \cap \mathcal{T}$ and

- *(i)* if S is subordinate or non-mesh, then $\widetilde{S} = S$;
- *(ii)* if S is mixed, then $\widetilde{S}$ carves away the mesh full component of S; and
- *(iii)* if S is mesh, then $\widetilde{S}$ carves away every component of $G - S$.

In this section we show how to find a subset of $2^{V(G)}$ which contains carvers for all appropriate $\mathcal{T}$ and S; see Proposition 4.9 for a precise statement. Our approach to proving this proposition is based on the theory of modular decompositions.

A *module* of a graph G is a set $X \subseteq V(G)$ such that every vertex in $V(G) \setminus X$ is adjacent to either all of X or none of X. A module is *strong* if it does not cross any other module, where two sets *cross* if they intersect and neither is contained in the other. A strong module is *maximal* if it is not $V(G)$ and it is not properly contained in any strong module besides $V(G)$. We do not need the full theory of modular decompositions, just the following fact.

Lemma 4.2 (see [14])**.** *The maximal strong modules of a graph G are disjoint and, if G is mesh, then they are the vertex sets of the components of $\overline{G}$.*

We typically guess two vertices which satisfy the following lemma.

Lemma 4.3. *Let G be a graph, d be a positive integer, $\mathcal{T}$ be a maximal treedepth-d structure in G, S be a $\mathcal{T}$-avoiding minimal separator, and A be a full component of S. Then $A \cap \mathcal{T}$ is non-empty, and there exists a vertex $p_A \in A$ which has at most $d-1$ neighbors in $\mathcal{T}$. Moreover, if $|A| > 1$, then there exists a vertex $q_A \in A$ which is adjacent to p_A and in a different maximal strong module of A than p_A.*

Proof. First notice that A contains a vertex in $\mathcal{T}$. Otherwise, each vertex $a \in A$ would satisfy $N(a) \cap \mathcal{T} \subseteq S$. As $\mathcal{T} \cap S$ is contained in a vertical path of $\mathcal{T}$ and does not contain any depth-d vertex of $\mathcal{T}$, we could add a to $\mathcal{T}$ as a leaf without increasing its height beyond d, thus contradicting the maximality of $\mathcal{T}$.

Now choose a vertex $p_A \in A \cap \mathcal{T}$ which has maximum depth among all vertices in $A \cap \mathcal{T}$. All vertices in $N(p_A) \cap A \cap \mathcal{T}$ are ancestors of p_A in $\mathcal{T}$. All vertices in $N(p_A) \cap \mathcal{T}$ that are descendants of p_A must be in S and thus they are contained in a vertical path of $\mathcal{T}$. This means that all vertices in $N(p_A) \cap \mathcal{T}$ are contained in a single vertical path in $\mathcal{T}$, which also contains p_A. Hence, $|N[p_A] \cap \mathcal{T}| \leqslant d$, thus $|N(p_A) \cap \mathcal{T}| \leqslant d-1$.

Finally, suppose that $|A| > 1$. Lemma 4.2 tells us that the maximal strong modules of A partition A. There is more than one part since $|A| > 1$. So, since A is connected, we can choose a neighbor q_A of p_A which is in a different part from p_A. □

We frequently apply the following lemmas from [13] to two vertices which come from Lemma 4.3.

Lemma 4.4 ([13, Lemma 4.2])**.** *Let G be a graph, let S be a minimal separator, let A be a full component of S, and let p_A and q_A be adjacent vertices which are in different maximal strong modules of A. Then for any $u \in S$, at least one of the following conditions holds:*

- *(i) there is an induced P_4 of the form $uAAA$,*
- *(ii) at least one of p_A and q_A is adjacent to u, or*
- *(iii) the graph A is mesh, and each of its maximal strong modules is either complete or anti-complete to u.*

We note that the outcomes in Lemma 4.4 are not exclusive.

The next lemma helps us to take care of minimal separators which are mesh.

Lemma 4.5 ([13, Lemma 4.4])**.** *Let G be a P_6-free graph, S be a minimal separator, A and B be full mesh components of S, and p_A and q_A (respectively, p_B and q_B) be adjacent vertices which are in different maximal strong modules of A (respectively, B). Then there exist $r_A \in A$ and $r_B \in B$ so that $S \subseteq N(p_A, q_A, r_A, p_B, q_B, r_B)$.*

Note that since A (resp., B) is mesh and p_A and q_A (resp., p_B and q_B) are in different maximal strong modules, we can equivalently say that $N[p_A, q_A, r_A, p_B, q_B, r_B] = A \cup S \cup B$.

We also need the following simplified version of Lemma 4.5 which applies to every type of minimal separator.

Lemma 4.6 ([13, Lemma 4.5])**.** *Let G be a P_6-free graph, S be a minimal separator, and A and B be two full components of S. Then there exist $A' \subseteq A$ and $B' \subseteq B$ such that $|A'| \leqslant 3$, $|B'| \leqslant 3$, and $S \subseteq N(A' \cup B')$.*

Note that Lemma 4.6 is sufficient to find all subordinate separators.

Corollary 4.7. *There is a polynomial-time algorithm which takes in a P_6-free graph G and returns a collection $\mathcal{S}_{sub} \subseteq 2^{V(G)}$ which contains each subordinate minimal separator.*

Proof. Let S be a subordinate minimal separator. Then there exists a minimal separator S' and two full sides A' and B' of S' so that $S \subseteq S'$ and some full component of S is disjoint from $A' \cup S' \cup B'$. By Lemma 4.6, there exist $A'' \subseteq A'$ and $B'' \subseteq B'$ so that $|A''| \leqslant 3$, $|B''| \leqslant 3$, and $S' \subseteq N(A'' \cup B'')$. We guess[3] A'' and B''. Then, for each component D of $G - N(A'' \cup B'')$, we insert $N(D)$ into $\mathcal{S}$. The full component of S which is disjoint from $A' \cup S' \cup B'$ is itself such a component D. So $\mathcal{S}$ contains S and $|\mathcal{S}| \leqslant |V(G)|^6$. □

Finally, some types of minimal separator can be taken care of very quickly using the following lemma. We state it in a slightly weaker fashion than in [13].

Lemma 4.8 ([13, Lemma 5.5])**.** *There is a polynomial-time algorithm which takes in a P_6-free graph G and returns a collection $\mathcal{F} \subseteq 2^{V(G)}$ which contains each full component of a non-mesh separator of G.*

Now we are ready to prove the main result of this section.

Proposition 4.9. *For each positive integer d, there exists a polynomial-time algorithm which takes in a P_6-free graph G and returns a collection $\mathcal{S} \subseteq 2^{V(G)}$ such that for any maximal treedepth-d structure $\mathcal{T}$ in G and any $\mathcal{T}$-avoiding minimal separator S, the collection $\mathcal{S}$ contains a $\mathcal{T}$-carver for S.*

Proof. Let d, G, $\mathcal{T}$, and S be as in the statement of the proposition. We will show how to construct $\mathcal{S}$ by making "guesses" among polynomially-many options. We will separately consider four cases, depending on the type of S. We output the collection $\mathcal{S}$ consisting of all sets $\widetilde{S}$ constructed as described below.

Case 1. S is subordinate. Recall that in Corollary 4.7 we constructed the family $\mathcal{S}_{\text{sub}}$ that contains all subordinate minimal separators S. As S is a $\mathcal{T}$-carver for S, it is sufficient to include $\mathcal{S}_{\text{sub}}$ in the output family $\mathcal{S}$.

Case 2. S is non-mesh. By Lemma 4.8 we can, in polynomial time, find a collection $\mathcal{F} \subseteq 2^{V(G)}$ which contains each full component of a non-mesh separator of G. For each $D \in \mathcal{F}$, we insert $N(D)$ into $\mathcal{S}$. So $\mathcal{S}$ contains S, which is a $\mathcal{T}$-carver for S.

From now on we may assume that S is either mixed or mesh. Thus S has exactly two full components, and at least one of them is mesh. Let A and B be the two full components of S. (We are not guessing A and B, we are just giving them names.) Next, guess the vertices in $S \cap \mathcal{T}$ (there are at most $d-1$ of them) and add them to a set $\widetilde{S}$. As we proceed throughout the proof, we will add more and more vertices of $V(G) \setminus \mathcal{T}$ to $\widetilde{S}$. Thus we will always have that $\widetilde{S} \cap \mathcal{T} = S \cap \mathcal{T}$, and we are trying to show that $\widetilde{S}$ eventually becomes a $\mathcal{T}$-carver for S.

By Lemma 4.3, there exists a vertex $p_A \in A$ (respectively, $p_B \in B$) which has at most $d-1$ neighbors in $\mathcal{T}$. For convenience, write $T_A \coloneqq N(p_A) \cap A \cap \mathcal{T}$ and $T_B \coloneqq N(p_B) \cap B \cap \mathcal{T}$. We

[3]Throughout this paper, by *guessing* we mean branching into polynomially many choices of fixing the object in question.

guess the vertices p_A and p_B and the sets T_A and T_B. We then add the vertices in $N(p_A) \setminus T_A$ and $N(p_B) \setminus T_B$ to $\widetilde{S}$.

If either A or B has size one, then this set $\widetilde{S}$ already contains S and is therefore a $\mathcal{T}$-carver for S. So we may assume that $|A| > 1$ and $|B| > 1$. Thus, by Lemma 4.3, there exists $q_A \in A$ (respectively, $q_B \in B$) so that p_A and q_A (respectively, p_B and q_B) are adjacent vertices which are in different maximal strong modules of A (respectively, B). We add every vertex which is in both $N(\{p_A, q_A\} \cup T_A)$ and $N(\{p_B, q_B\} \cup T_B)$ to $\widetilde{S}$; note that these newly added vertices are a subset of S.

It is helpful to state the following observation; note that it will hold even after we add more vertices to $\widetilde{S}$.

(1) Each vertex $u \in S \setminus \widetilde{S}$ is non-adjacent to p_A and p_B and therefore in a P_3 of the form uAA and in a P_3 of the form uBB.

In particular, note that when applying Lemma 4.4 for any $u \in S \setminus \widetilde{S}$ and A (resp., B) we never obtain the first outcome, as then we would get an induced P_6 of the form $BBuAAA$ (resp., $AAuBBB$).

Case 3. S **is mixed.** We claim that $\widetilde{S}$ is already a $\mathcal{T}$-carver for S. By symmetry between A and B, we may assume that A is mesh and B is non-mesh. So it just remains to show that A is carved away by $\widetilde{S}$; that is, that no component of $G - \widetilde{S}$ intersects both A and another component of $G - S$. We will do this by showing that $S \setminus \widetilde{S}$ and $A \setminus \widetilde{S}$ are anticomplete. So consider a vertex $u \in S \setminus \widetilde{S}$. By (1), the second outcome of Lemma 4.4 holds for B, and $u \in N(p_B, q_B)$. So u is anticomplete to $\{p_A, q_A\} \cup T_A$; otherwise we would have $u \in \widetilde{S}$. Now the third outcome of Lemma 4.4 holds for A; that is, each maximal strong module of A is either complete or anticomplete to u. As A is mesh and $p_A \notin N(u)$, each neighbor of u in A is in $N(p_A) \setminus T_A$. Since $N(p_A) \setminus T_A \subseteq \widetilde{S}$, this completes the proof that $\widetilde{S}$ is a $\mathcal{T}$-carver for S.

Case 4. S **is mesh.** Then by Lemma 4.5, there exist $r_A \in A$ and $r_B \in B$ so that $S \subseteq N(p_A, q_A, r_A, p_B, q_B, r_B)$, i.e., $N[p_A, q_A, r_A, p_B, q_B, r_B] = A \cup S \cup B$. Guess these vertices r_A and r_B. So for each component D of $G - N[p_A, q_A, r_A, p_B, q_B, r_B]$, add the vertices in $N(D)$ to $\widetilde{S}$. Furthermore, we add to $\widetilde{S}$ all vertices in $N[q_A, r_A] \cap N[q_B, r_B]$. Note that these newly added vertices are a subset of S.

We will show that now $\widetilde{S}$ is a $\mathcal{T}$-carver for S. It just remains to show that $\widetilde{S}$ carves away the components of $G - S$: that is, that each component of $G - \widetilde{S}$ intersects at most one component of $G - S$. Since $N(D)$ as explicitly added to $\widetilde{S}$ for each component D of $G - [p_A, q_A, r_A, p_B, q_B, r_B]$, the only possibility that needs to be checked is that some component of $G - \widetilde{S}$ intersects both A and B. So it suffices to show that $S \setminus \widetilde{S}$ has a partition into two parts, S_A and S_B, so that S_A and S_B are anticomplete, S_A and $B \setminus \widetilde{S}$ are anticomplete, and S_B and $A \setminus \widetilde{S}$ are anticomplete.

Let S_A (respectively, S_B) be the set of all vertices in $S \setminus \widetilde{S}$ which are in $N(q_A, r_A)$ (respectively, $N(q_B, r_B)$). The sets S_A and S_B partition $S \setminus \widetilde{S}$ by observation (1) and the definitions of r_A, r_B and $\widetilde{S}$. Now consider a vertex $u \in S_A$. Again using (1), the third outcome of Lemma 4.4 holds for B; each maximal strong module of B is either contained in or disjoint from the neighborhood of u. So each neighbor of u in B is in $N(p_B) \setminus T_B$, and therefore also in $\widetilde{S}$. By this and the symmetric argument for S_B, we have proven that S_A and $B \setminus \widetilde{S}$ are anticomplete, and S_B and that $A \setminus \widetilde{S}$ are anticomplete.

It just remains to show that S_A and S_B are anticomplete. For this we need to be slightly more careful about the argument above; notice that we actually have that if $u \in S_A$, then u is anticomplete to every component of $\overline{B}$ which intersects $\{p_B, q_B, r_B\}$. Let M_B denote the union of these components of $\overline{B}$. Note that M_B induces a connected subgraph of B since p_B and q_B are in different components of $\overline{B}$. Thus, if u was adjacent to a vertex $v \in S_B$, then we could find

a P_6 of the form $M_A M_A uv M_B M_B$, where, symmetrically, M_A is the union of the components of $\overline{A}$ which intersect $\{p_A, q_A, r_A\}$.

This completes all four cases and therefore the proof of Proposition 4.9. □

5 Improving carvers for mixed minimal separators

We need a more refined understanding of mixed minimal separators. So let G be a graph, S be a mixed minimal separator of G, and A and B be the mesh and non-mesh full sides of S, respectively. Given a set $\widetilde{S} \subseteq V(G)$, we say that a component $\widetilde{D}$ of $G - \widetilde{S}$ is *clarified* if it is disjoint from $A \cup B$. In this section we show how to "carve away" all of the clarified components; see Proposition 5.4.

To prove this proposition, we will use the following enumeration routine to obtain a "fuzzy" version of the mesh full component. Given a graph G, a *fuzzy version* of a set $A \subseteq V(G)$ is a set $A^+ \subseteq V(G)$ such that $A \subseteq A^+$ and every vertex of $A^+ \setminus A$ is complete to A.

Lemma 5.1 ([13, Lemma 5.6])**.** *There is a polynomial-time algorithm which takes in a P_6-free graph G and returns a collection $\mathcal{A} \subseteq 2^{V(G)}$ such that for every mixed minimal separator S in G with A as its full mesh component, there exists $A^+ \in \mathcal{A}$ that is a fuzzy version of A.*

We also use the following lemma about minimal elements in quasi-orders. A *quasi-order* is a pair $(X, \preceq)$ so that X is a set and $\preceq$ is a reflexive and transitive relation on X.

Lemma 5.2 ([13, Lemma 4.1])**.** *Let X be a non-empty finite set, and let $(X, \preceq_0)$ and $(X, \preceq_1)$ be quasi-orders such that each pair of elements of X is comparable either with respect to $\preceq_0$ or with respect to $\preceq_1$ (or both). Then there exists an element $x \in X$ such that for every $y \in X$, either $x \preceq_0 y$ or $x \preceq_1 y$ (or both).*

We use Lemma 5.2 to prove the following lemma, which will help us recognize an independent set which is contained in a mixed minimal separator.

Lemma 5.3. *Let G be a P_6-free graph, $S \subseteq V(G)$ be a set with a mesh full component A, and $I \subseteq S$ be a non-empty independent set. Then there exist a component M_I of $\overline{A}$ and a vertex $x \in I \cap N(M_I)$ so that every vertex $y \in I \setminus N(M_I)$ is a neighbor of every component D of $G - (A \cup S)$ so that $x \in N(D)$.*

Proof. For convenience, let $\mathcal{D}$ denote the collection of components of $G - (A \cup S)$, and let $\mathcal{M}$ denote the collection of components of $\overline{A}$; we will obtain one quasi-order from $\mathcal{D}$ and another from $\mathcal{M}$. Notice that if there are two vertices $u, v \in I$ such that there exists both a pair $D_u, D_v \in \mathcal{D}$ so that $N(D_u) \cap \{u, v\} = \{u\}$ and $N(D_v) \cap \{u, v\} = \{v\}$, and a pair $M_u, M_v \in \mathcal{M}$ so that $N(M_u) \cap \{u, v\} = \{u\}$ and $N(M_v) \cap \{u, v\} = \{v\}$, then there is a P_6 of the form $D_u u M_u M_v v D_v$.

Consider the quasi-orders $\preceq_0$ and $\preceq_1$ on I defined as follows:

$$u \preceq_0 v \iff \{D \in \mathcal{D} \mid u \in N(D)\} \subseteq \{D \in \mathcal{D} \mid v \in N(D)\}, \text{ and}$$
$$u \preceq_1 v \iff \{M \in \mathcal{M} \mid u \in N(M)\} \subseteq \{M \in \mathcal{M} \mid v \in N(M)\}.$$

Any two $u, v \in I$ are comparable in at least one of these orders. Hence, Lemma 5.2 asserts that there exist $x \in I$ such that for every $y \in I$ either $x \preceq_0 y$ or $x \preceq_1 y$. We pick any $M_I \in \mathcal{M}$ with x as a neighbor (it exists since $I \subseteq S = N(A)$). □

We are ready to prove the main proposition about improving carvers for mixed minimal separators.

Proposition 5.4. *For each positive integer d, there exists a polynomial-time algorithm which takes in a P_6-free graph G and a set $\widetilde{S} \subseteq V(G)$ and returns a collection $\mathcal{S}' \subseteq 2^{V(G)}$ so that for any maximal treedepth-d structure $\mathcal{T}$ in G and any $\mathcal{T}$-avoiding mixed minimal separator S of G, there exists $S' \in \mathcal{S}'$ so that*

(i) S' *contains* $\widetilde{S}$,

(ii) $S' \cap \mathcal{T} \subseteq S \cup \widetilde{S}$, *and*

(iii) for each clarified component $\widetilde{D}$ *of* $G - \widetilde{S}$, *no component of* $\widetilde{D} - S'$ *intersects more than one component of* $G - S$.

Proof. Let d, G, $\widetilde{S}$, S, and $\mathcal{T}$ be as in the lemma statement. Let A and B denote the mesh and non-mesh full sides of S, respectively. Additionally, let $\mathcal{D}$ denote the set of all vertices of $G - S$ which are in a clarified component of $G - \widetilde{S}$. So $\mathcal{D}$ is the union of some components of $G - (A \cup S \cup \widetilde{S} \cup B)$, and the graph $G - \widetilde{S}$ has no path between $\mathcal{D}$ and $A \cup B$. We will find a set S' which satisfies conditions *(i)* and *(ii)* of the proposition and includes $N(\mathcal{D}) \cap S$; this implies condition *(iii)*.

Notice that there are at most d components of $\overline{A}$ which intersect $\mathcal{T}$; let $M \subseteq V(G)$ denote the union of these components. We claim that we can guess M. By Lemma 5.1, we can, in polynomial-time, obtain a set $\mathcal{A} \subseteq 2^{V(G)}$ which includes a fuzzy version of A. That is, there exists $A^+ \in \mathcal{A}$ so that $A \subseteq A^+$ and $A^+ \setminus A$ is complete to A. Guess this set $A^+ \in \mathcal{A}$; there are polynomially-many choices. Now each component of $\overline{A}$ is also a component of the complement of A^+ and can thus be guessed. So indeed we can guess M, as it is the union of at most d components of $\overline{A^+}$. We will use the fact that M is non-empty, which follows from the fact that $A \cap \mathcal{T}$ is non-empty by Lemma 4.3.

Now we define an intermediate set $X \subseteq V(G)$ which contains $\widetilde{S}$ and is our current best guess at S'. To begin with we set $X \coloneqq \widetilde{S} \cup N(M)$; these vertices are safe to include since $N(M) \cap \mathcal{T} \subseteq S$. Next, by Lemma 4.3, there exists a vertex $p_B \in B$ which has at most $d - 1$ neighbors in $\mathcal{T}$. We guess this vertex, along with which of its neighbors are in $\mathcal{T} \cap B$, and then we add all of its other neighbors to X. This completes the definition of X. Notice that $X \cap \mathcal{T} \subseteq S \cup \widetilde{S}$, that $S \setminus X$ is anticomplete to $M \cup \{p_B\}$, and that $G - X$ has no path between $\mathcal{D}$ and $A \cup B$ (this follows from the fact that $\widetilde{S} \subseteq X$). We also remark that $X \subseteq A \cup B \cup S \cup \widetilde{S}$.

We claim that there exists a vertex $q_B \in B$ which is complete to $S \setminus X$. If $S \setminus X$ is empty, then this is trivially true, so assume that it is non-empty. Then $|B| > 1$ since $S \setminus X$ is anticomplete to p_B. So by Lemma 4.3, there is a vertex $q_B \in B$ so that p_B and q_B are adjacent and in different maximal strong modules of B. If $S \setminus X$ is not complete to q_B, then by Lemma 4.4 applied to the full component B of S, we obtain a vertex $u \in S \setminus X$ which is in a P_4 of the form $uBBB$. However, u is also in a P_3 of the form uAA since $S \setminus X$ is anticomplete to M (which is non-empty). But then we obtain a P_6 of the form $AAuBBB$, which contradicts the fact that G is P_6-free. Consequently, that $S \setminus X$ is complete to q_B. We guess such a vertex q_B.

Now form an independent set $I \subseteq S \setminus X$ as follows. For each component of $S \setminus X$ which has a neighbor in $\mathcal{D}$, choose one vertex with a neighbor in $\mathcal{D}$ and add that vertex to I. (We are not saying that we can guess I, just that it exists.) We may assume that I is non-empty since otherwise the proposition holds with $S' \coloneqq X$. Now apply Lemma 5.3 to the subgraph induced on $A \cup S \cup \mathcal{D}$. Thus, there exist a component M_I of $\overline{A}$ and a vertex $x \in I \cap N(M_I)$ so that every vertex $y \in I \setminus N(M_I)$ is a neighbor of every component D of $\mathcal{D}$ so that $x \in N(D)$. We can guess M_I for the same reason we were able to guess M (because M_I is a component of $\overline{A}$ and we can guess the fuzzy version A^+ of A).

We will prove that the following set S' satisfies the proposition. First we add X and $N(M_I) \cap N(q_B)$ to S'. These vertices are safe to add since $X \cap \mathcal{T} \subseteq S \cup \widetilde{S}$ and $N(M_I) \cap N(q_B) \subseteq S$. We observe that since $X \subseteq A \cup B \cup S \cup \widetilde{S}$, we have $S' \subseteq A \cup B \cup S \cup \widetilde{S}$ at this moment. Now consider each component D of $G - X - N(q_B)$ which has x as a neighbor. Clearly, D is disjoint from S as $S \setminus X \subseteq N(q_B)$. Let H be a component of $N(q_B) \setminus X$ that contains a neighbor of D. Since x has a neighbor in $\mathcal{D}$, there is a component of $G - \widetilde{S}$ that contains H, D, x, and a component of $\mathcal{D}$, hence, it is disjoint from $A \cup B$. In particular, D is disjoint with $A \cup B$, so $N(D) \subseteq S \cup \widetilde{S}$ as $X \subseteq A \cup B \cup S \cup \widetilde{S}$. Furthermore, we have $H \subseteq S$. Over all choices of D and H as above, we add the component H to S'.

We have already proved that conditions *(i)* and *(ii)* of the proposition hold for S'. Recall that, in order to obtain the final condition *(iii)*, it is enough to show that S' contains $N(\mathcal{D}) \cap S$. So, going for a contradiction, suppose that there exists a vertex $u \in \mathcal{D}$ which has a neighbor $v \in S \setminus S'$. Let H be the component of $S \setminus X$ which contains v. Then x is disjoint from and anticomplete to $H \cup \{u\}$, since otherwise we would have added v to S'. However, now there is a P_6 of the form $uvq_Bx M_I M$, which contradicts the fact that G is P_6-free. (To see that there is a P_6 of this form, recall that q_B is complete to $S \setminus X$, x has a neighbor in M_I while u and v do not, and $S \setminus X$ is anticomplete to M, which is non-empty; therefore M_I is a component of $\overline{A}$ which is not any of the components of $\overline{A}$ we used to define M.) This contradiction completes the proof of Proposition 5.4. □

6 Not-two-sided PMCs

A potential maximal clique Ω in a graph G is *two-sided* if there exist two distinct connected components D_1, D_2 of $G - \Omega$ such that for every connected component D of $G - \Omega$, we have $N(D) \subseteq N(D_1)$ or $N(D) \subseteq N(D_2)$.

The following statement has been essentially proven in [13]. However, it has been proven only with the Max Weight Independent Set problem in mind, so we need to slightly adjust the argumentation to fit the more general setting of this paper.

Theorem 6.1. *For every positive integer d there exists a polynomial-time algorithm that, given a P_6-free graph G outputs a family $\mathcal{C} \subseteq 2^{V(G)}$ with the following guarantee: for every maximal tree-depth-d structure $\mathcal{T}$ in G and every potential maximal clique Ω of G that is $\mathcal{T}$-avoiding and not two-sided, there exists $C \in \mathcal{C}$ that is a container for Ω, i.e., $\Omega \subseteq C$ and $C \cap V(\mathcal{T}) = \Omega \cap V(\mathcal{T})$.*

As mentioned, Theorem 6.1 is essentially proven in Section 5 of [13]. There, for a fixed maximal independent set I, a PMC Ω is *I-free* if it is disjoint with Ω. This assumption here is replaced with Ω being $\mathcal{T}$-avoiding for a fixed tree-depth-d structure $\mathcal{T}$. Informally speaking, to adjust it to our setting, we need to make three adjustments within the proof of [13].

(i) Often, when mesh component D is analyzed, it is argued that the independent set I intersects at most one maximal module M_p of D, and a vertex $p \in M_p \cap I$ is guessed. This step is usually followed by a guess of an arbitrary vertex q in a different maximal strong module of D.

In our case, the tree-depth-d structure $\mathcal{T}$ can intersect at most d modules of D, and the guess of p is replaced with a guess of a set P of at most d vertices of $\mathcal{T} \cap D$, one vertex from each maximal strong module of D that intersects $\mathcal{T}$. For q, it is enough to take an arbitrary vertex of D, unless $|P| = 1$ (i.e., $\mathcal{T}$ intersects only one maximal strong module of D) where we need to pick q from a different maximal strong module. In this manner, we maintain the property that $P \cup \{q\}$ contains vertices of at least two maximal strong modules of D, so in particular $D \subseteq N[P \cup \{q\}]$. Whenever later the proof of [13] considers $N[p]$ or $N[p, q]$, we consider here $N[P]$ or $N[P \cup \{q\}]$ instead.

In what follows, we call such a set P a *footprint of $\mathcal{T}$ in D* and the vertex q a *satellite of the footprint P*.

(ii) When a PMC Ω that is disjoint with the maximal independent set I is analyzed, and we often argue that the maximality of I implies that every $v \in \Omega$ has a neighbor in I that is outside Ω. In our case, Lemma 2.13 gives the same corollary, except for the vertices of $\mathcal{T} \cap \Omega$, but there are fewer than d of them and they can be guessed separately.

(iii) Finally, the notion of a *neighbor-maximal* component of Section 5.3 of [13] is a bit incompatible with our statement, as it considers two components D_1, D_2 of $G - \Omega$ with $N(D_1) = N(D_2)$ both *not* neighbor-maximal. This definition restricts the set of all PMCs

with more than two neighbor-maximal components. We observe that the assumption "more than two neighbor-maximal components" is used only once in the proof and can be easily replaced with the (slightly weaker) assumption of being not two-sided.

Let us now have a closer look at Section 5 of [13] and provide formal details. The toolbox in the earlier sections nor Lemmas 5.2 up to Lemma 5.7 use the notion of I-freeness, so they can be used in our setting without any modifications.

Lemma 5.8 of [13], the main result of Section 5 there, would now obtain the following form.

Lemma 6.2 (analog of Lemma 5.8 of [13])**.** *For every integer d there exists a polynomial-time algorithm that, given on input a P_6-free graph G, outputs two families $\mathcal{F}_9^1$ and $\mathcal{F}_9^2$ such that the following holds: for every maximal tree-depth-d structure $\mathcal{T}$ in G and every potential maximal clique Ω of G that is $\mathcal{T}$-avoiding and not two-sided, either $\mathcal{F}_9^1$ contains Ω or $\mathcal{F}_9^2$ contains a triple $(\Omega \cup D_1 \cup D_2, D_1^+, D_2^+)$ for some components D_1, D_2 of $G - \Omega$ that are mesh, where D_i^+ is a fuzzy version of D_i for $i \in \{1, 2\}$.*

Note that Theorem 6.1 follows easily from Lemma 6.2: we insert into $\mathcal{C}$ every element of $\mathcal{F}_9^1$ and, for every $(K, L_1, L_2) \in \mathcal{F}_9^2$, every choice of at most d maximal strong modules of L_1 and every choice of at most d maximal strong modules of L_2, we insert into $\mathcal{C}$ the set K minus the chosen modules. Thus, it remains to prove Lemma 6.2.

The proof of Lemma 5.8 of [13] splits into three lemmas: Lemma 5.9, Lemma 5.10, and Lemma 5.11. These statements have a fixed P_6-free graph G and a maximal independent set I in their context. In our setting, instead of I we fix an integer d and a maximal tree-depth-d structure $\mathcal{T}$ in G.

Lemma 5.9 of [13] takes the following form.

Lemma 6.3 (analog of Lemma 5.9 of [13])**.** *Suppose Ω is a $\mathcal{T}$-avoiding PMC in G and D is a component of $G-\Omega$ which is mesh. Let P be a footprint of $\mathcal{T}$ in D and let q be a satellite of P. Let $J \subseteq N(D)$ be an independent set with the following property: for every $v \in J$, the set $N(v) \cap D$ consists of some maximal strong modules of D and is disjoint with $\mathcal{T} \cap D$. Then there exists $w \in D$ and a component D' of $G - \Omega$, distinct from D, such that $J \subseteq (\mathcal{T} \cap \Omega) \cup N(w) \cup N(D')$.*

Proof sketch. The proof of Lemma 5.9 of [13] uses I-freeness of Ω in only one place: to argue that if $v \in J$ is anti-complete to all vertices of I in D, then it needs to be adjacent to a vertex of I in another component D' of $G - \Omega$, so in particular it is adjacent to some other component of $G - \Omega$. In our case, J is anti-complete to $D \cap \mathcal{T}$, and Lemma 2.13 gives the same corollary, except for the vertices of $\mathcal{T} \cap \Omega$ that need to be added there separately. The rest of the proof is the same. □

Similarly we adjust Lemma 5.10 of [13].

Lemma 6.4 (analog of Lemma 5.10 of [13])**.** *Given a family $\mathcal{X} \subseteq 2^{V(G)}$, one can in time polynomial in the size of G and the size of $\mathcal{X}$ compute a family $\mathcal{F}_7(\mathcal{X}) \subseteq 2^{V(G)}$ with the following properties: for every $\mathcal{T}$-avoiding PMC Ω and every component D of $G-\Omega$, if all components of $G - \Omega$, except for possibly D, belong to $\mathcal{X}$, then all components of $G - \Omega$ belong to $\mathcal{F}_7(\mathcal{X})$.*

Proof sketch. The assumption on I-freeness of Lemma 5.10 of [13] comes into play in the proof only in the last case, namely Case 3, where in particular D is mesh.

First, after Claim 1, we guess a vertex $p \in I \cap M_p$ for the unique maximal strong module M_p of D that intersects I, and a vertex q in another maximal strong module. Here, we perform the standard adjustment, guessing instead a footprint of $\mathcal{T}$ in D and its satellite.

Second, in the definition of Y, we also want to exclude the vertices of $\mathcal{T} \cap D$ from it (there are fewer than d of them, so we just try all possibilities).

Third, after Claim 7 we invoke Lemma 5.9. Because of the previous adjustment, the set J here is disjoint with $\mathcal{T} \cap \Omega$. Hence, we can invoke the adjusted Lemma 6.3 instead. □

We now move to Lemma 5.11 of [13].

Lemma 6.5 (analog of Lemma 5.11 of [13])**.** *One can in polynomial time compute a family $\mathcal{F}_8$ such that the following holds: Take any $\mathcal{T}$-avoiding PMC Ω and assume there are different components D_1, D_2 of $G-\Omega$ that are meshes. Then $\mathcal{F}_8$ contains either D_1, or D_2, or $\Omega \cup D_1 \cup D_2$.*

Proof sketch. Again, the proof in [13] starts by selecting, for every $i \in \{1,2\}$, a vertex $p_i \in I$ in the unique maximal strong module of D_i that intersects I. We adjust it in the standard way by selecting a footprint P_i of $\mathcal{T}$ and its satellite q_i.

Then, when defining X and Z, we need to also include $\mathcal{T} \cap \Omega$ into X, so Z is disjoint with $\mathcal{T}$. Since $\mathcal{T} \cap \Omega$ is of size less than d, we just try all possibilities.

Finally, Claim 11 relies on I-freeness. It argues that a vertex $z \in Z \subseteq N(D_2)$ that does not have a neighbor in $I \cap D_2$, needs to have a neighbor in I in another component of $G - \Omega$, in particular it is adjacent to another component of $G - \Omega$. In our case, z is not in $\mathcal{T}$ (as it is in Z) and z has no neighbor in $\mathcal{T} \cap D_2$, so Lemma 2.13 gives the same corollary. □

With the above three lemmas in hand, we can now adjust the proof of Lemma 5.8 of [13] to show Lemma 6.2. The crucial insight is that if there are two components D_1, D_2 of $G - \Omega$ with $N(D_1) = N(D_2)$, then $N(D_1)$ is subordinate (because $N(D_1)$ has three full sides, D_1, D_2, and a component containing $\Omega \setminus N(D_1)$) and hence it belongs to the family $\mathcal{S}_{\mathrm{sub}}$ provided by Corollary 4.7.

Therefore, by adding all full components of subordinate separators to a constructed set $\mathcal{G}$, we obtain the same properties as in the proof of Lemma 5.8 of [13] under the weaker assumption that Ω is not two-sided: $\mathcal{G}$ contains either all components of $G-\Omega$, or all except for at most two mesh components D_1 and D_2. The first outcome allows us to recover Ω exactly. In the second outcome we use Lemma 6.5: we either get exactly Ω or the set $\Omega \cup D_1 \cup D_2$. In the latter case, it remains to get, for every $i \in \{1,2\}$, a fuzzy version of D_i.

Since $D_i \notin \mathcal{G}$, $N(D_i)$ is not subordinate. Since Ω is not two-sided, there is another component D' of $G - \Omega$, distinct from D_1 and D_2, such that $N(D') \not\subseteq N(D_i)$. This component is in $\mathcal{G}$. Then, Lemma 5.7 of [13] gives a polynomial number of candidates for a fuzzy version of D_i.

This completes the proof sketch of Lemma 6.2 and thus concludes the proof of Theorem 6.1.

7 Analysis of two-sided aligned PMCs

In this section we deal with the last remaining type of PMCs: two-sided aligned PMCs. Contrary to the previous sections, we need to make some delicate surgery on the clique tree in order to adjust it before generating a small family of carvers. More precisely, we will need the following special property of a clique tree (T, β) of a chordal completion $G + F$. (Recall here that full components of adhesions were defined following Lemma 2.6.)

(♠) There are no two distinct edges $st, tu \in E(T)$ such that

- *(i)* $\sigma(st) \subseteq \sigma(tu)$;
- *(ii)* $\sigma(tu)$ is a mixed minimal separator; and
- *(iii)* the full component of $\sigma(tu)$ on the u-side is non-mesh.

The next lemma verifies that property (♠) can be always achieved, even without changing the completion set F.

Lemma 7.1. *For any graph G and minimal chordal completion $G+F$ of G, there exists a clique tree (T, β) of $G + F$ with property (♠).*

Proof. We already know that $G + F$ has some clique tree. We will choose a clique tree which maximizes a certain count; for this definition we need to orient some edges of the tree. So, given a clique-tree (T, β) of $G + F$, orient each edge of T whose adhesion is a mixed minimal separator "towards the non-mesh side". That is, if $tu \in E(T)$ is such that $\sigma(tu)$ is a mixed minimal separator and the full component of $\sigma(tu)$ on the u-side is non-mesh, then orient tu as (t, u).

Now, choose a clique tree (T, β) of $G+F$ which maximizes the sum, over all nodes $u \in V(T)$, of the number of undirected edges which are incident to a node of T that can be reached from u via a directed path (that is, a path which does not use any undirected edge and which follows the directed edges according to their direction). Such a choice exists since all clique trees have the same number of nodes. We will prove that (T, β) satisfies the conditions of the lemma. So, going for a contradiction, suppose that there exist distinct edges $st, tu \in E(T)$ so that *(i)*, *(ii)*, and *(iii)* of property (♠) hold. By conditions *(ii)* and *(iii)*, tu is oriented as (t, u).

For convenience, set $S \coloneqq \sigma(tu)$, and let A (respectively, B) denote the full component of S on the t-side (respectively, u-side). Since S is mixed, it has exactly two full components: A that contains $\beta(t) \setminus S$ and B that contains $\beta(u) \setminus S$. Since $\sigma(st) \subseteq S$ and $\sigma(st)$ separates $\beta(s) \setminus \sigma(st)$ from both $\beta(t) \setminus S$ and $\beta(u) \setminus S$, it follows that the full component of $\sigma(st)$ on the s-side is disjoint from $A \cup S \cup B$. In particular, this component cannot be a full component of S (which has only two full sides, A and B), hence $\sigma(st) \subsetneq S$. Therefore $\sigma(st)$ is subordinate and the edge st of T is undirected.

Now we define a new clique tree (T', β') of $G + F$ as follows. Replace the edge st of T with the edge su; that is, reattach the component of $T - \{st\}$ that contains s to be connected via an edge su instead of the edge st. Since $\sigma(st) \subseteq S = \sigma(tu)$, the resulting tree is in fact a clique tree of $G + F$. Furthermore, the orientations of the edges do not change; su is an undirected edge as S is a subordinate separator, while for every other edge of T the full sides considered in the orientation remain the same. Moreover, the relevant count of (T', β') is strictly larger than that of (T, β): the count for u increases by one, while no other count decreases. This contradicts the choice of (T, β) and completes the proof of Lemma 7.1. □

Now we are ready to prove the main result of this section.

Proposition 7.2. *For each positive integer d, there exists a polynomial-time algorithm which takes in a P_6-free graph G and returns a collection $\mathcal{C}_1 \subseteq 2^{V(G)}$ so that for any maximal treedepth-d structure $\mathcal{T}$ in G and any $\mathcal{T}$-aligned minimal chordal completion $G + F$ of G, there exists a clique tree (T, β) of $G + F$ such that for each node t of T, if $\beta(t)$ is two-sided and $\mathcal{T}$-avoiding, then the set $\mathcal{C}_1$ contains a $(\mathcal{T}, (T, \beta))$-carver for $\beta(t)$.*

Proof. Let d, G, $\mathcal{T}$, F be as in the lemma statement. Let (T, β) be a clique tree of $G + F$ which satisfies property (♠), its existence if guaranteed by Lemma 7.1. We orient some of the edges of T as in the proof of Lemma 7.1. That is, for each edge tu of T so that $\sigma(tu)$ is a mixed minimal separator, we orient tu as (t, u) if the full component of $\sigma(tu)$ on the u-side is non-mesh, and as (u, t) otherwise. In this language, property (♠) becomes the following.

(♣) There do not exist distinct edges $st, tu \in E(T)$ such that $\sigma(st) \subseteq \sigma(tu)$ and tu is oriented towards u.

Now fix $t \in V(T)$ such that $\beta(t)$ is two-sided and $\mathcal{T}$-avoiding. We will argue how to construct a $(\mathcal{T}, (T, \beta))$-carver for $\beta(t)$ using guesswork with only polynomially-many options. To this end, set $\Omega \coloneqq \beta(t)$, and let D_0 and D_1 be the components of $G - \Omega$ which witness that Ω is two-sided. Throughout the rest of this proof we write indices on subscripts modulo 2.

First of all, for each $v \in V(G)$, we add the set $N[v]$ to $\mathcal{C}_1$. Note that this takes care of all PMCs Ω which contain a vertex v that does not have a neighbor outside Ω. Indeed, by the characterization of PMCs in Proposition 2.5, we would have $N[v] = \Omega$ and thus $\Omega \in \mathcal{C}_1$.

Thus from now on we may assume that each vertex from Ω has a neighbor outside of Ω. We now use the characterization of PMCs in Proposition 2.5 to infer the following claim.

Claim 7.2.1. *The following properties hold:*

(i) $N(D_0) \cup N(D_1) = \Omega$,

(ii) the sets $N(D_0) \setminus N(D_1)$ *and* $N(D_1) \setminus N(D_0)$ *are nonempty and complete to each other, and*

(iii) there exists $j \in \{0,1\}$ *such that* D_j *is complete to* $N(D_j) \setminus N(D_{j+1})$.

Proof of Claim. By assumption, each vertex in Ω has a neighbor outside of Ω. Since Ω is two-sided, we infer that $N(D_0) \cup N(D_1) = \Omega$. Since $N(D_0)$ and $N(D_1)$ are proper subsets of Ω (see Proposition 2.5), we have that both $N(D_0) \setminus N(D_1)$ and $N(D_1) \setminus N(D_0)$ are nonempty. From Proposition 2.5, we infer that $N(D_0) \setminus N(D_1)$ is complete to $N(D_1) \setminus N(D_0)$, as there is no connected component of $G - \Omega$ that is adjacent to some vertices in both those sets.

Finally, suppose towards a contradiction that for every $i \in \{0,1\}$, there exists $v_i \in N(D_i) \setminus N(D_{i+1})$ that is not complete to D_i. Then there is a P_6 of the form $D_0 D_0 v_0 v_1 D_1 D_1$. This contradiction completes the proof of Claim 7.2.1. ⌟

By Lemma 2.8, for $i \in \{0,1\}$, the set $N(D_i)$ is a minimal separator of G which has a full side $D_i^\Omega \neq D_i$ that contains $\Omega \setminus N(D_i)$. Since Ω is two-sided and $N(D_0) \cup N(D_1) = \Omega$ by part *(i)* of Claim 7.2.1, it follows that D_i^Ω is precisely the union of $\Omega \setminus N(D_i)$ and the components of $G - \Omega$ which have a neighbor in $\Omega \setminus N(D_i)$. Thus, in particular, $D_0^\Omega \cap D_1^\Omega = \emptyset$ since Ω is two-sided.

We now show how the adhesions relate to the components of $G - \Omega$. For each component D of $G - \Omega$, we write T_D for the component of $T - \{t\}$ so that $D \subseteq \bigcup_{s \in T_D} \beta(s)$; this component exists and is unique. We write t_D for the node of T_D which is a neighbor of t in T. We also write t_0 and t_1 as shorthand for t_{D_0} and t_{D_1}, respectively, and similarly for T_0 and T_1.

We now attempt to "capture" the minimal separators $N(D_0)$ and $N(D_1)$. By Proposition 4.9, we can, in polynomial-time, obtain a collection $\mathcal{S} \subseteq 2^{V(G)}$ which contains a $\mathcal{T}$-carver for each $\mathcal{T}$-avoiding minimal separator. So in particular, $\mathcal{S}$ contains $\mathcal{T}$-carvers S_0 and S_1 for $N(D_0)$ and $N(D_1)$, respectively. We can guess these sets S_0 and S_1 since $\mathcal{S}$ also has polynomial size.

We will use the following observation twice.

Claim 7.2.2. *Let* $k \in \{0,1\}$ *be such that* tt_k *is not oriented towards* t. *Then no component of* $G - S_k$ *intersects both* $N(D_k) \setminus N(D_{k+1})$ *and* D_k^Ω.

Proof of Claim. Let D be a component of $G - S_k$ that intersects D_k^Ω. Since tt_k is not oriented towards t, by the properties of S_k we have that S_k carves away D_k^Ω, hence $D \subseteq N(D_k) \cup D_k^\Omega$.

Assume there exists $v \in D \cap (N(D_k) \setminus N(D_{k+1}))$. Since $v \in N(D_k) \setminus S_k$ while $S_k \cap \mathcal{T} = N(D_k) \cap \mathcal{T}$, we have $v \notin \mathcal{T}$. By Lemma 2.13, there exists $w \in \mathcal{T} \setminus \Omega$ that is a neighbor of v. Since $w \in \mathcal{T} \setminus \Omega$ while $S_k \cap \mathcal{T} = N(D_k) \cap \mathcal{T}$, we have $w \notin S_k$, thus $w \in D \setminus \Omega$. As $D \subseteq N(D_k) \cap D_k^\Omega$, every component D' of $G - \Omega$ that intersects D satisfies $N(D') \subseteq N(D_{k+1})$. This is a contradiction with $w \in N(D_k) \setminus N(D_{k+1})$. ⌟

A *precarver* is a set $\widetilde{S} \subseteq V(G)$ such that $\widetilde{S} \cap \mathcal{T} = \Omega \cap \mathcal{T}$ and there exists $k \in \{0,1\}$ such that for every component $\widetilde{D}$ of $G - \widetilde{S}$ at least one of the following conditions holds:

- there exists a component T' of $T - \{t\}$ with $\widetilde{D} \subseteq \beta(t) \cup \bigcup_{t' \in V(T')} \beta(t')$, or
- tt_k is oriented and $\widetilde{D}$ is clarified with regards to the mixed separator $N(D_k)$ (i.e., $\widetilde{D}$ is disjoint with $D_k \cup D_k^\Omega$).

If we are able to guess a precarver $\widetilde{S}$, then we apply Proposition 5.4 for the minimal separator $N(D_k)$ to guess a superset C of $\widetilde{S}$ with $C \cap \mathcal{T} = \widetilde{S} \cap \mathcal{T}$. Then the properties of the precarver

together with Proposition 5.4 imply that C will be a $(\mathcal{T}, (T, \beta))$-carver for Ω. Hence, in the remainder of the proof we focus on guessing a precarver.

We observe that the first bullet of the definition of a precarver holds immediately for a component $\widetilde{D}$ if $\widetilde{D} \subseteq \Omega$ or there exists a component D of $G - \Omega$ such that $\widetilde{D} \subseteq D \cup \Omega$. The latter applies in to the case $\widetilde{D} \cap \Omega = \emptyset$.

We perform now case distinction on how the edges tt_0 and tt_1 are oriented in (T, β), which is in fact a case distinction on the types of separators $N(D_0)$ and $N(D_1)$.

Case 1. There exists $k \in \{0, 1\}$ such that tt_k is undirected. We claim that then $\widetilde{S} = S_0 \cup S_1$ is a precarver. To this end, let $\widetilde{D}$ be a component of $G - \widetilde{S}$.

If $\widetilde{D} \subseteq \Omega$, there is nothing to prove, so assume otherwise. Let D be a component of $G - \Omega$ that intersects $\widetilde{D}$. If $N(D) \subseteq N(D_k)$, then D is a component of $G - N(D_k)$ and thus, as tt_k is undirected and S_k is a carver for tt_k, we have $\widetilde{D} \subseteq D \cup N(D_k) \subseteq D \cup \Omega$.

If tt_{k+1} is undirected too, then a symmetric argument resolves the case $N(D) \subseteq N(D_{k+1})$. Since Ω is two-sided, this completes the proof in this case.

Otherwise, tt_{k+1} is directed; without loss of generality assume $k = 1$. Recall that we are left with analysing a component $\widetilde{D}$ of $G - \widetilde{S}$ that satisfies the following: for every component D of $G - \Omega$ that intersects $\widetilde{D}$, we have $N(D) \not\subseteq N(D_1)$ (so $N(D) \subseteq N(D_0)$ and $N(D) \cap (N(D_0) \setminus N(D_1)) \neq \emptyset$, as Ω is two-sided). This implies that $\widetilde{D} \subseteq \Omega \cup D_1^{\Omega}$.

Case 1.1. tt_0 **is oriented towards** t**.** As $\widetilde{D}$ intersects D_1^{Ω}, from Claim 7.2.2 for $k = 1$ we infer that $\widetilde{D}$ is disjoint with $N(D_1) \setminus N(D_0)$. Recall that $\widetilde{D}$ is also disjoint with every component D of $G - \Omega$ with $N(D) \subseteq N(D_1)$. Thus, $\widetilde{D}$ is disjoint with D_0^{Ω}, as D_0^{Ω} consists of $\Omega \setminus N(D_0) = N(D_1) \setminus N(D_0)$ and every component D of $G - \Omega$ with $N(D) \subseteq N(D_1)$ and $N(D) \cap (N(D_1) \setminus N(D_0)) \neq \emptyset$.

If $\widetilde{D}$ intersects D_0, then, by the properties of the carver S_0, $\widetilde{D} \subseteq D_0 \cup N(D_0)$ and we are done. Otherwise, $\widetilde{D}$ is clarified with regards to the mixed separator $N(D_0)$, because it is disjoint with both full sides: D_0 and D_0^{Ω}. Hence, $\widetilde{S}$ is a precarver.

Case 1.2. tt_0 **is oriented towards** t_0**.** Since $N(D) \not\subseteq N(D_1)$, we have $t_D = t_0$, as otherwise the edge tt_D is an undirected edge with $\sigma(tt_D) \subseteq \sigma(tt_0)$, violating property (♣). As the above holds for every component D of $G - \Omega$ that intersects $\widetilde{D}$, we have $\widetilde{D} \subseteq \Omega \cup \bigcup_{t' \in V(T_0)} \beta(t')$ and we are done.

Case 2. Both tt_0 and tt_1 are oriented towards t. Then both D_0 and D_1 are mesh. Note that $N(D_0) \cap N(D_1)$ is a minimal separator with full sides D_0 and D_1 in a induced subgraph of G. So by Lemma 4.5 applied to this induced subgraph, we can pick at most three elements of D_0 and at most three elements of D_1 so that every vertex in $N(D_0) \cap N(D_1)$ is a neighbor of one of these six (or fewer) vertices. By adding at most one more vertex from a different component of $\overline{D_0}$, and similarly for $\overline{D_1}$, we obtain sets $D_0' \subseteq D_0$ and $D_1' \subseteq D_1$ so that $|D_0'| \leqslant 4$, $|D_1'| \leqslant 4$, and every vertex in D_0, D_1, and $N(D_0) \cap N(D_1)$ is in $N[D_0' \cup D_1']$. Guess these sets D_0' and D_1'.

For every $i \in \{0, 1\}$, recall that there are at most d components of $\overline{D_i}$ which intersect $\mathcal{T}$; let $M_i \subseteq V(G)$ denote the union of these components. Since Lemma 5.1 allows us to guess a fuzzy version of D_i, we can guess M_i, as every component of $\overline{D_i}$ is a component of the complement of a fuzzy version of D_i. We set

$$\widetilde{S} := S_0 \cup S_1 \cup (N[D_0' \cup D_1'] \setminus (M_0 \cup M_1)).$$

We claim that $\widetilde{S}$ is a precarver. As $N(D_0) \cup N(D_1) = \Omega$, it is immediate that $\widetilde{S} \cap \mathcal{T} = \Omega \cap \mathcal{T}$.

Recall from part *(iii)* of Claim 7.2.1 that there exists $j \in \{0, 1\}$ such that D_j is complete to $N(D_j) \setminus N(D_{j+1})$. By symmetry, we can assume that D_1 is complete to $N(D_1) \setminus N(D_0)$. Hence, $N(D_1) \setminus N(D_0) \subseteq \widetilde{S}$. Since also $N(D_0) \cap N(D_1) \subseteq \widetilde{S}$ due to the inclusion of $N[D_0' \cup D_1'] \setminus (M_0 \cup M_1)$, we have $N(D_1) \subseteq \widetilde{S}$.

Consider now a component $\widetilde{D}$ of $G - \widetilde{S}$. We claim that either $\widetilde{D}$ is contained in $D \cup \Omega$ for a single component D of $G - \Omega$ or $\widetilde{D}$ is clarified with regards to the minimal separator $N(D_0)$

(whose full sides are D_0 and D_0^Ω). The claim is trivial if $\widetilde{D} \subseteq \Omega$. If there exists $i \in \{0,1\}$ such that $\widetilde{D}$ intersects D_i, then $\widetilde{D} \subseteq \Omega \cup D_i$ due to the inclusion of the carvers S_0 and S_1 in $\widetilde{S}$. If $\widetilde{D}$ intersects a component $D \notin \{D_0, D_1\}$ of $G - \Omega$ such that $N(D) \subseteq N(D_1)$, then $\widetilde{D} \subseteq D$ as $N(D_1) \subseteq \widetilde{S}$. In the remaining case, $\widetilde{D}$ intersects a component $D \notin \{D_0, D_1\}$ with $N(D) \subseteq N(D_0)$, $N(D) \cap (N(D_0) \setminus N(D_1)) \neq \emptyset$. Furthermore, due to the exclusion of the previous cases, $\widetilde{D}$ is disjoint both with D_0 and with D_0^Ω, as the latter consists of D_1, $N(D_1) \setminus N(D_0)$ (which is a subset of $\widetilde{S}$) and all components D' of $G - \Omega$ with $N(D') \subseteq N(D_1)$, $N(D') \cap (N(D_1) \setminus N(D_0)) \neq \emptyset$. Hence, $\widetilde{D}$ is clarified with regards to $N(D_0)$. This finishes the proof that $\widetilde{S}$ is a precarver.

Case 3. Both tt_0 and tt_1 are oriented away from t. By Lemma 2.7, for every $s \in N_T(t)$ we have $\sigma(st) \subseteq N(D_0)$ or $\sigma(st) \subseteq N(D_1)$. Hence, by property (♣), t_0 and t_1 are the only two neighbors of t in G.

We claim that $\widetilde{S} = S_0 \cup S_1$ is a precarver in this case. Consider a component $\widetilde{D}$ of $G - \widetilde{S}$.

If there exists $k \in \{0,1\}$ such that $\widetilde{D}$ intersects D_k^Ω, then, by the properties of the carver S_k, we have $\widetilde{D} \subseteq D_k^\Omega \cup N(D_k)$. Consequently, for every component D of $G - \Omega$ that intersects $\widetilde{D}$, it holds that $N(D) \subseteq N(D_{k+1})$, $N(D) \cap (N(D_{k+1}) \setminus N(D_k)) \neq \emptyset$. We infer $t_D = t_{k+1}$ for every such component D. Hence, $\widetilde{D} \subseteq \Omega \cup \bigcup_{t' \in T_{k+1}} \beta(t')$.

If $\widetilde{D}$ is disjoint with $D_0^\Omega \cup D_1^\Omega$, then it is disjoint also with $D_0 \cup D_1$ as $D_{k+1} \subseteq D_k^\Omega$ for every $k \in \{0,1\}$. Hence, $\widetilde{D}$ is clarified with regards to both $N(D_0)$ and $N(D_1)$. This finishes the proof that $\widetilde{S}$ is a precarver.

Case 4. One of the edges tt_0 and tt_1 is oriented towards t and one is oriented away from t. Without loss of generality, assume tt_0 is oriented towards t_0 and tt_1 is oriented towards t.

We distinguish the following two subcases.

Case 4.1. There exists a component D of $G-\Omega$, $D \neq D_1$, with $N(D) \cap (N(D_1) \setminus N(D_0)) \neq \emptyset$. Let D be such a component and let $v \in N(D) \cap (N(D_1) \setminus N(D_0))$. We argue that

$$\begin{aligned} &\text{For every } u \in (N(D_0) \cap N(D_1)) \setminus N(D), \text{ there is no } P_4 \text{ of the form} \\ &uD_0D_0D_0, \text{and if additionally } uv \notin E(G), \text{ then } u \text{ is complete to } D_0. \end{aligned} \tag{3}$$

Let $u \in (N(D_0) \cap N(D_1)) \setminus N(D)$. Let Q be an induced path consisting of a shortest path from u to v possibly via D_1 if $uv \notin E(G)$ and then a neighbor of v in D. Observe that Q has three vertices if $uv \in E(G)$ and at least four vertices if $uv \notin E(G)$.

If there exists an induced P_4 of the form $uD_0D_0D_0$, then the concatenation of this P_4 with Q yields a P_6, a contradiction. Similarly, if there exists an induced P_3 of the form uD_0D_0 (which is equivalent to u not being complete to D_0), then the concatenation of this P_3 with Q yields a P_6 if $uv \notin E(G)$. This proves (3).

For every $k \in \{0,1\}$, apply Lemma 4.3 to the separator $N(D_k)$ with full component D_k, obtaining a vertex $p_k \in D_k \cap \mathcal{T}$ with $A_k := \mathcal{T} \cap D_k \cap N(p_k)$ of size at most $d-1$ and, if $|D_k| > 1$, a vertex $q_k \in D_k \cap N(p_k)$ in a different maximal strong module of D_k than p_k. We set $q_k = p_k$ if $|D_k| = 1$.

Let

$$\widetilde{S} = S_0 \cup S_1 \cup \left(\bigcup_{k \in \{0,1\}} (N(p_k) \setminus A_k) \right) \cup (N(q_0) \cap N(\{v, q_1\})) \cup N(D).$$

Note that $\widetilde{S}$ can be guessed with polynomial number of options, as $N(D)$ is a subordinate separator and hence can be guessed using Corollary 4.7.

We claim that $\widetilde{S}$ is a precarver. Since $N(q_0) \cap N(\{v, q_1\}) \subseteq N(D_0) \subseteq \Omega$, we have $\widetilde{S} \cap \mathcal{T} = \Omega \cap \mathcal{T}$.

We now show that

$$N(D_0) \cap N(D_1) \subseteq \widetilde{S}. \tag{4}$$

Let $u \in N(D_0) \cap N(D_1)$. If $u \in N(D)$ or $u \in N(p_0)$, then $u \in \widetilde{S}$. Otherwise, u is not complete to D_0, so by (3) we have $u \in N(v)$ and there is no P_4 of the form $uD_0D_0D_0$. Lemma 4.4 implies that $u \in N(q_0)$. Hence, $u \in \widetilde{S}$. This proves (4).

Consider now a component $\widetilde{D}$ of $G - \widetilde{S}$. We distinguish two cases, depending on whether $\widetilde{D}$ intersects D_0^Ω.

If $\widetilde{D}$ intersects D_0^Ω, then by the properties of the carver S_0 we have $\widetilde{D} \subseteq N(D_0) \cup D_0^\Omega$. By Claim 7.2.2 for $k = 0$, $\widetilde{D}$ is disjoint with $N(D_0) \setminus N(D_1)$. By (4), $\widetilde{D}$ is disjoint with $N(D_0)$, that is, $\widetilde{D} \subseteq D_0^\Omega$. In particular, $\widetilde{D}$ is disjoint with D_1^Ω.

If $\widetilde{D}$ intersects D_1 then, by the properties of the carver S_1, we have $\widetilde{D} \subseteq D_1 \cup N(D_1)$. Otherwise, $\widetilde{D}$ is disjoint with both D_1 and D_1^Ω and thus is clarified with regards to the separator $N(D_1)$.

In the other case, the component $\widetilde{D}$ is disjoint with D_0^Ω. So $N(D) \subseteq N(D_0)$ for every component D of $G - \Omega$ that intersects $\widetilde{D}$. If there exists a component D of $G - \Omega$ with $N(D) \subseteq N(D_0) \cap N(D_1)$ that intersects $\widetilde{D}$, then $\widetilde{D} \subseteq D$ thanks to (4). Otherwise, for every component D of $G - \Omega$ that intersects $\widetilde{D}$ we have $N(D) \not\subseteq N(D_1)$. By Lemma 2.7 and property (♣), for every such component we have $t_D = t_0$. Thus, $\widetilde{D} \subseteq \beta(t) \cup \bigcup_{t' \in V(T_0)} \beta(t')$.

This finishes the proof that $\widetilde{S}$ is a precarver in this case.

Case 4.2. For every component D of $G-\Omega$, either $D = D_1$ or $N(D) \subseteq N(D_0)$. Lemma 2.7 and property (♣) imply that t is of degree 2 in T, that is, t_0 and t_1 are the only two neighbors of t in T.

For every $k \in \{0, 1\}$, proceed as follows. Call a node t of T considered in this case *special*; since this is the last case, we may assume that for all non-special nodes of T, we already have constructed carvers for their bags. Let t'_k be the closest to t node of T_k that is not special. Note that t'_k exists and is unique, as every node of T that is special has degree 2 in T. (It may happen that $t'_k = t_k$). Let Q_k be the path in T between t and t'_k.

As this is the last case, we can guess a $(\mathcal{T}, (T, \beta))$-carver C_1 for $\beta(t'_1)$. (Note that this guesswork may involve Lemma 2.12 if $\beta(t'_1)$ is not $\mathcal{T}$-avoiding or Theorem 6.1 if $\beta(t'_1)$ is $\mathcal{T}$-avoiding but not two-sided.) Let $A_1 = \mathcal{T} \cap (C_1 \setminus \Omega) = \mathcal{T} \cap (\beta(t'_1) \setminus \Omega)$; as $|A_1| \leqslant d$, we can guess A_1.

We now perform an analysis of components of $G - \Omega$.

Claim 7.2.3. *Let D be a component of $G - \Omega$ distinct from D_0 and D_1 and let $k^D \in \{0, 1\}$ be such that $t_D = t_{k^D}$. Then there exists an edge $t_A^D t_B^D$ of T such that:*

- *$\sigma(t_A^D t_B^D) = N(D)$.*
- *If T_A^D is the component of $T - \{t_A^D t_B^D\}$ that contains t_A^D, then $D \subseteq \bigcup_{t' \in V(T_A^D)} \beta(t')$.*
- *The nodes t_A^D, t_B^D, t'_{k^D}, t_{k^D}, and t lie on the unique path between t_A^D and t in T in this order, with possibly $t_B^D = t'_{k^D}$ and/or $t'_{k^D} = t_{k^D}$.*

In particular, if T^D is the unique component of $T - \{t'_{k^D}\}$ that contains t_A^D, then $t \notin V(T^D)$ and $D \subseteq \bigcup_{t' \in V(T^D)} \beta(t')$.

Proof of Claim. Let D be as in the statement. By Lemma 2.8, $N(D)$ is a minimal separator with full sides D and D^Ω, where D^Ω contains $\Omega \setminus N(D)$. By the assumptions of the current case, $N(D) \subseteq N(D_0)$. Furthermore, as $N(D_0)$ is mixed, $N(D_0)$ has only two full sides: D_0 and D_0^Ω that contains $\Omega \setminus N(D_0)$. As both of them are disjoint with D, it follows that D is not a full component of $G - N(D_0)$, that is, $N(D)$ is a proper subset of $N(D_0)$. Hence, D^Ω contains not only $\Omega \setminus N(D)$, but also both D_0 and D_0^Ω, which in turn contains D_1.

Since $N(D) \subseteq \Omega$, $N(D)$ is a clique in $G+F$. We apply Lemma 2.10 for $S = N(D)$, $A = D$, and $B = D^\Omega$, obtaining the edge $t_A^D t_B^D$. The first two promised properties are immediate by Lemma 2.10.

For the third property, since $N(D)$ is a subordinate separator, $t_A^D t_B^D$ is an undirected edge of T. Thus $t_A^D t_B^D$ lies in the component of $T - E(Q_0 \cup Q_1)$ that contains t'_{k^D} and, furthermore, as $\beta(t) \cap D^\Omega \neq \emptyset$, both t'_{k^D} and t_B^D lie on the unique path from t_A^D to t in T. The claim follows. ⌟

With the above claim in hand, we now prove that

$$A_1 \subseteq D_1. \tag{5}$$

By contradiction, assume that A_1 intersects a component $D \neq D_1$ of $G - \Omega$. As $A_1 \subseteq \beta(t'_1) \subseteq \bigcup_{t' \in V(T_1)} \beta(t')$, we have $N(D) \subseteq N(D_1)$, $t_D = t_1$, and thus $D \neq D_0$ and $k^D = 1$. By Claim 7.2.3, t'_1 lies on the unique path from t_B^D to t in T (possibly $t'_1 = t_B^D$). Hence, $A_1 \cap D \subseteq \beta(t'_1) \cap D \subseteq \beta(t_A^D) \cap \beta(t_B^D) = N(D) \subseteq \Omega$, a contradiction. This proves (5).

Define

$$C' := S_0 \cup S_1 \cup C_1 \quad \text{and} \quad C := C' \setminus A_1.$$

We claim that C is a $(\mathcal{T}, (T, \beta))$-carver for Ω. Clearly, $C \cap \mathcal{T} = \Omega \cap \mathcal{T}$. (We would like to use C' as the carver, but unfortunately C' may contain vertices of $\mathcal{T}$ in $\beta(t'_1) \setminus \Omega$, that is, A_1. Therefore we need to exclude them manually.) Let $\widetilde{D}$ be a component of $G - C$; we want to show that there exists $k \in \{0, 1\}$ such that $\widetilde{D} \subseteq \beta(t) \cup \bigcup_{t' \in V(T_k)} \beta(t')$.

If $\widetilde{D}$ intersects D_1, then by the properties of the carver S_1 we have $\widetilde{D} \subseteq N(D_1) \cup D_1$ and we are done with $k = 1$. Otherwise, $\widetilde{D} \cap A_1 = \emptyset$ by (5). Hence, $\widetilde{D}$ is also a component of $G - C'$.

Assume now that $\widetilde{D}$ intersects a component D of $G - \Omega$ such that $D \notin \{D_0, D_1\}$ and $k^D = 1$. Then, as $\widetilde{D}$ is a component of $G - C'$ and $C_1 \subseteq C'$, by the properties of the carver C_1 and Claim 7.2.3 we have

$$\widetilde{D} \subseteq \beta(t'_1) \cup \bigcup_{t' \in V(T^D)} \beta(t') \subseteq \Omega \cup \bigcup_{t' \in V(T_1)} \beta(t').$$

In the remaining case, for every component D of $G - \Omega$ that intersects $\widetilde{D}$ we have $t_D = t_0$. Hence, $\widetilde{D} \subseteq \Omega \cup \bigcup_{t' \in V(T_0)} \beta(t')$. This finishes the proof in this case.

This completes the case analysis and thus the proof of Proposition 7.2. □

8 Wrap up

We are now ready to conclude the construction of a treedepth-d carver family for P_6-free graphs.

Theorem 8.1. *For each positive integer d, there exists a polynomial-time algorithm that takes in a P_6-free graph G and outputs a family $\mathcal{F} \subseteq 2^{V(G)}$ that is a treedepth-d carver family for G.*

Proof. Fix any maximal treedepth-d structure $\mathcal{T}$ in G, any $\mathcal{T}$-aligned minimal chordal completion $G+F$ of G, and any maximal clique Ω of $G+F$. The crucial observation is that any container for Ω is a $(\mathcal{T}, (T, \beta))$-carver for Ω regardless of the clique tree (T, β) of $G+F$. Hence, Proposition 7.2 gives a family of carvers handling two-sided maximal cliques of $G + F$ for a particular choice of the clique tree, while Theorem 6.1 and Lemma 2.12 handle the remaining maximal cliques of $G + F$ regardless of the choice of the clique tree. □

Theorem 1.2 follows by a direct combination of Theorem 8.1, Theorem 3.2, and Theorem 2.2.

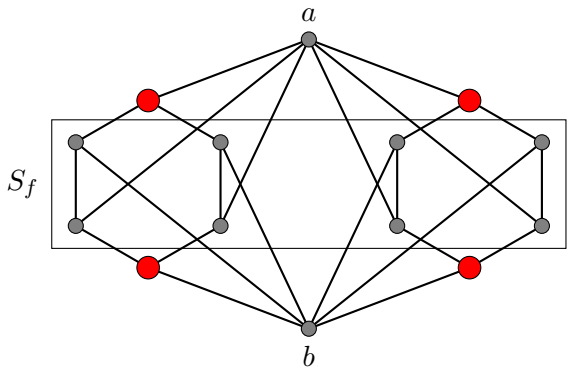

Figure 2: The graph G_2 with an independent set I_f depicted as large red vertices and the corresponding separator S_f boxed. (The definition of f is not needed due to automorphisms of the graph.)

9 Conclusions

In this paper, we introduced the notion of carvers, a relaxation of the notion of containers, and showed its applicability by proving that any $(\deg \leqslant k, \varphi)$-MWIS problem is solvable in polynomial time on P_6-free graphs.

While in Definition 3.1 and Theorem 3.2 we only require that there exists a tree decomposition (T, β) that is represented in a carver family, our proof in fact provides a carver family that works for some clique tree of every $\mathcal{T}$-aligned chordal completion $G+F$, where $\mathcal{T}$ is any maximal treedepth-d structure containing the solution. (Note that in the context of MWIS, $d = 1$ and $\mathcal{T}$ is just the sought solution, since it is a maximal independent set.) We now present an example showing that if one aims for the ultimate goal of proving the tractability of $(\deg \leqslant k, \varphi)$-MWIS in P_t-free graphs for any fixed t, in particular for $t = 7$, one needs to either really use the flexibility of the choice of (T, β), or further adjust the notion of a carver. See Figure 2 for a depiction of the example.

For an integer $n \geqslant 1$, construct a graph G_n as follows; take n copies of the 6-vertex cycle, let the vertices of the i-th cycle be $v_{i,0}, \ldots, v_{i,5}$, $1 \leqslant i \leqslant n$, and add two vertices a and b; a is adjacent to all vertices $v_{i,0}, v_{i,2}, v_{i,4}$ and b is adjacent to all vertices $v_{i,1}, v_{i,3}, v_{i,5}$, $1 \leqslant i \leqslant n$. The graph G_n is P_7-free. For every $f : \{1, \ldots, n\} \to \{0, 2, 4\}$, the graph G_n contains a maximal independent set

$$I_f = \{v_{i,f(i)}, v_{i,f(i)+3} \mid 1 \leqslant i \leqslant n\}$$

and a minimal separator

$$S_f = \{v_{i,f(i)+1}, v_{i,f(i)+2}, v_{i,f(i)+4}, v_{i,f(i)+5} \mid 1 \leqslant i \leqslant n\}$$

with full mesh sides

$$A_f = \{a\} \cup \{v_{i,f(i)} \mid 1 \leqslant i \leqslant n\},$$
$$B_f = \{b\} \cup \{v_{i,f(i)+3} \mid 1 \leqslant i \leqslant n\}.$$

Here, the addition in the second index is performed modulo 6. In this example, if one wants to provide for every f a carver for (an I_f-aligned PMC containing) the separator S_f that separates $I_f \cap A_f$ from $I_f \cap B_f$, one needs an exponential number of carvers. However, the minimal separator $\{a, b\}$ instead of S_f seems like a much better choice for the algorithm.

References

[1] Tara Abrishami, Maria Chudnovsky, Marcin Pilipczuk, Paweł Rzążewski, and Paul Seymour, *Induced subgraphs of bounded treewidth and the container method*, 32nd Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, SIAM, 2021, pp. 1948–1964.

[2] Vladimir E Alekseev, *The effect of local constraints on the complexity of determination of the graph independence number*, Combinatorial-algebraic methods in applied mathematics (1982), 3–13.

[3] Vladimir E. Alekseev, *On easy and hard hereditary classes of graphs with respect to the independent set problem*, Discret. Appl. Math. **132** (2003), no. 1-3, 17–26.

[4] Marthe Bonamy, Nicolas Bousquet, Michał Pilipczuk, Paweł Rzążewski, Stéphan Thomassé, and Bartosz Walczak, *Degeneracy of P_t-free and $C_{>t}$-free graphs with no large complete bipartite subgraphs*, Journal of Combinatorial Theory, Series B **152** (2022), 353–378.

[5] Marthe Bonamy, Konrad K. Dabrowski, Carl Feghali, Matthew Johnson, and Daniël Paulusma, *Independent Feedback Vertex Set for P_5-free graphs*, Algorithmica **81** (2019), no. 4, 1342–1369.

[6] Vincent Bouchitté and Ioan Todinca, *Treewidth and minimum fill-in: Grouping the minimal separators*, SIAM J. Comput. **31** (2001), no. 1, 212–232.

[7] Bruno Courcelle, *The Monadic Second-Order logic of graphs. I. Recognizable sets of finite graphs*, Inf. Comput. **85** (1990), no. 1, 12–75.

[8] Bruno Courcelle and Joost Engelfriet, *Graph structure and monadic second-order logic - A language-theoretic approach*, Encyclopedia of mathematics and its applications, vol. 138, Cambridge University Press, 2012.

[9] Fedor V. Fomin, Ioan Todinca, and Yngve Villanger, *Large induced subgraphs via triangulations and CMSO*, SIAM J. Comput. **44** (2015), no. 1, 54–87.

[10] Peter Gartland and Daniel Lokshtanov, *Independent set on P_k-free graphs in quasi-polynomial time*, 61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, IEEE, 2020, pp. 613–624.

[11] Peter Gartland, Daniel Lokshtanov, Marcin Pilipczuk, Michał Pilipczuk, and Paweł Rzążewski, *Finding large induced sparse subgraphs in $C_{>t}$-free graphs in quasipolynomial time*, 53rd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2021, ACM, 2021, pp. 330–341.

[12] Andrzej Grzesik, Tereza Klimošová, Marcin Pilipczuk, and Michal Pilipczuk, *Covering minimal separators and potential maximal cliques in P_t-free graphs*, Electron. J. Comb. **28** (2021), no. 1.

[13] Andrzej Grzesik, Tereza Klimošová, Marcin Pilipczuk, and Michał Pilipczuk, *Polynomial-time algorithm for Maximum Weight Independent Set on P_6-free graphs*, ACM Trans. Algorithms **18** (2022), no. 1, 4:1–4:57.

[14] Michel Habib and Christophe Paul, *A survey of the algorithmic aspects of modular decomposition*, Computer Science Review **4** (2010), no. 1, 41–59.

[15] Bart M. P. Jansen, Daniel Lokshtanov, and Saket Saurabh, *A near-optimal planarization algorithm*, 25th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, SIAM, 2014, pp. 1802–1811.

[16] Leonid Libkin, *Elements of finite model theory*, Texts in Theoretical Computer Science. An EATCS Series, Springer, 2004.

[17] Daniel Lokshantov, Martin Vatshelle, and Yngve Villanger, *Independent set in P_5-free graphs in polynomial time*, 25th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, SIAM, 2014, pp. 570–581.

[18] Giacomo Paesani, Daniël Paulusma, and Paweł Rzążewski, *Classifying Subset Feedback Vertex Set for H-free graphs*, 48th International Workshop on Graph-Theoretic Concepts

in Computer Science, WG 2022, Lecture Notes in Computer Science, vol. 13453, Springer, 2022, pp. 412–424.

[19] ———, *Feedback Vertex Set and Even Cycle Transversal for H-free graphs: Finding large block graphs*, SIAM J. Discret. Math. **36** (2022), no. 4, 2453–2472.

[20] Marcin Pilipczuk, *A tight lower bound for Vertex Planarization on graphs of bounded treewidth*, Discret. Appl. Math. **231** (2017), 211–216.

[21] Marcin Pilipczuk, Michał Pilipczuk, and Paweł Rzążewski, *Quasi-polynomial-time algorithm for Independent Set in P_t-free graphs via shrinking the space of induced paths*, 4th Symposium on Simplicity in Algorithms, SOSA 2021, SIAM, 2021, pp. 204–209.

[22] Neil Robertson and P. D. Seymour, *Graph minors. V. Excluding a planar graph*, Journal of Combinatorial Theory, Series B **41** (1986), no. 1, 92–114.

[23] ———, *Graph minors. XX. Wagner's conjecture*, Journal of Combinatorial Theory, Series B **92** (2004), no. 2, 325–357, Special Issue Dedicated to Professor W.T. Tutte.