arXiv:1605.03045v1 [cs.LO] 10 May 2016

Definability equals recognizability for graphs of bounded treewidth [*]

Mikołaj Bojańczyk   Michał Pilipczuk
University of Warsaw
{bojan,michal.pilipczuk}@mimuw.edu.pl

Abstract

We prove a conjecture of Courcelle, which states that a graph property is definable in MSO with modular counting predicates on graphs of constant treewidth if, and only if it is recognizable in the following sense: constant-width tree decompositions of graphs satisfying the property can be recognized by tree automata. While the forward implication is a classic fact known as Courcelle's theorem, the converse direction remained open.

Categories and Subject Descriptors F.4 [*Theory of Computation*]: Mathematical Logic and Formal Languages

Keywords treewidth, tree decomposition, Monadic Second-Order Logic, recognizability, Simon's factorization forest

1. Introduction

Classical results of Büchi, Trakhtenbrot and Elgot say that for finite words, languages recognised by finite automata are exactly those definable in Monadic Second-Order logic MSO. Courcelle's theorem shows the right-to-left inclusion holds for graphs of bounded treewidth: if a property of graphs can be defined in MSO with quantification over edge subsets and modular counting predicates — henceforth called counting MSO on graphs — then for every k there is a tree automaton recognising tree decompositions of width k of graphs satisfying the property. A corollary is that model checking counting MSO on graphs of constant treewidth can be done in linear time, which is one of the foundational results in parameterized complexity. For more details, we refer the reader to the monograph of Courcelle and Engelfriet [7] devoted to MSO on graphs.

Courcelle's theorem, stated as above, generalizes only one direction of the equivalence between MSO and automata. The converse question is whether one can use counting MSO to define any property of graphs that is recognizable for constant treewidth, where recognizability is defined by, say, the finiteness of the index of an appropriate Myhill-Nerode relation. In the case of words (or trees), this is the 'easy' direction: a formula of MSO can guess an accepting run of an automaton, by labelling nodes with states. Surprisingly, the generalization of this implication to graphs of constant treewidth remained open for the last 25 years.

This problem, known as the *Courcelle's conjecture*, was initially formulated by Courcelle in the very first paper of his monumental series *The monadic second-order logic of graphs* [4]. The fifth paper of the series [5] is entirely devoted to its investigation and contains a proof for graphs of treewidth 2. Since then, the conjecture has been confirmed for graphs of treewidth 3 [12], for k-connected graphs of treewidth k [12], for graphs of constant treewidth and chordality [1], and for k-outerplanar graphs [10]. There were also two claims of a significant progress on the general case. First, Kabanets [11] claimed a proof for graphs of bounded pathwidth, and then a resolution of the full conjecture was claimed by Lapoire [14]. Unfortunately, both these works are published only as extended conference abstracts, and no verified journal version has appeared. The proofs of Kabanets and Lapoire are widely regarded as unsatisfactory; cf. [1, 7, 8, 10]. In particular, the problem is stated as open both in the monograph of Courcelle and Engelfriet [7] and of Downey and Fellows [8].

The issue at the heart of Courcelle's conjecture is that an MSO formula is applied to the graph alone, without access to any predefined tree decomposition. Hence, one cannot simply guess a run of a tree automaton, because there is no tree. As Courcelle puts it in [5], *It is not clear at all how an automaton should traverse a graph. A "general" graph has no evident structure, whereas a word or a tree is (roughly speaking) its own algebraic structure*. Hence, the natural approach to proving the conjecture is to find, using MSO and the graph structure only, some tree decomposition of bounded width. This strategy, proposed by Courcelle [5], was used in all the previous work on Courcelle's conjecture. We also use this strategy.

Our main result is that there exists an MSO *transduction* that, given a graph of treewidth k encoded as a relational structure, outputs an encoding of a tree decomposition of width $f(k)$, for some function f. Informally speaking, an MSO transduction guesses existentially a number of vertex and edge subsets, and based on them defines a tree decomposition. Different guesses may lead to different decompositions, but provided the input graph has treewidth at most k, the output for at least one guess will be a tree decomposition of width bounded by $f(k)$. The precise statement is in Section 2.2.

Acknowledgment. We would like to thank Christoph Dittmann and Stephan Kreutzer for many helpful discussions.

2. Overview

The main technical result of this paper, stated in Theorem 2.4, is that using MSO one can compute a tree decomposition of a graph. This section describes the proof plan. We explain what it means to compute something in MSO, and divide Theorem 2.4 into lemmas.

2.1 Tree decompositions

In this section we define tree decompositions and show how we represent them for the purposes of MSO. Similar formalisms were used in the previous works, cf. [5, 10, 12], but we choose to introduce our own language for the sake of being self-contained.

Logical terminology. Define a *vocabulary* to be a set of relation names with associated arities (we do not use functions or constants). A *logical structure* over a vocabulary consists of a universe supplied

[*] M. Bojańczyk is supported by the ERC Consolidator Grant LIPA. This work was partially done while Mi. Pilipczuk held a post-doc position at Warsaw Centre of Mathematics and Computer Science. Mi. Pilipczuk is also supported by the Foundation for Polish Science (FNP) via the START stipend programme.

with interpretations of relations in the vocabulary. We use logical structures to model things like graphs and tree decompositions.

Graphs as logical structures. We model (undirected) graphs as logical structures, where the universe consists of both vertices and edges, and there is a binary *incidence* relation $\mathsf{incident}(v,e)$ which says when a vertex v is incident with an edge e. The edges can be recovered as those elements of the universe that are second arguments of the incidence relation, and the vertices can be recovered as those elements of the universe that are not edges. We do not allow multiple edges connecting the same pair of vertices, i.e., all graphs considered in this work are simple, unless explicitly stated. We choose this encoding so that set quantification in MSO can capture sets of edges as well.

Tree decompositions. We begin by defining tree decompositions. Define an *in-forest* to be an acyclic directed graph where every node has outdegree at most one. We use the usual tree terminology: root, parent, and child. Every connected component of an in-forest is a tree with exactly one root (vertex of outdegree zero).

Definition 2.1. A *tree decomposition* of a graph G is an in-forest t whose vertices called *nodes*, and a labelling of the nodes by sets of vertices in G, called *bags*, subject to the following conditions:

- for every edge e of G, some bag contains both endpoints of e;
- for every vertex v of G, the set of nodes in t that are labelled by bags containing v is nonempty and connected in t.

Note two minor changes with respect to the classic definition: tree decompositions are rooted, and we allow them to be forests, instead of just trees. Both changes are for convenience only and bear no significance for our results.

We write x, y, z for nodes and u, v, w for vertices. Below we introduce some terminology for tree decompositions, inspired by Grohe and Marx [9].

Definition 2.2. Let x be a node in a tree decomposition t. The *adhesion of* x is the intersection of the bags of x and its parent; if x is a root the adhesion is empty. The *margin of* x is its bag minus its adhesion. The *cone of* x is the union of the bags of the descendants of x, including x. The *component of* x is its cone minus its adhesion. If the decomposition t is not clear from the context, we write t-cone, t-adhesion, etc.

Note that the margins of the nodes of a tree decomposition form a partition of the vertex set of the underlying graph.

A *path decomposition* is the special case when the forest is a set of paths. The *width* of a tree or path decomposition is the maximum size of its bags, minus 1. The *treewidth* of a graph is the minimum possible width of its tree decomposition, likewise for *pathwidth*. The treewidth and pathwidth of a graph G are denoted by $\mathtt{tw}(G)$ and $\mathtt{pw}(G)$, respectively.

Tree decompositions as logical structures. A tree decomposition is represented as a logical structure as follows. The universe of the logical structure consists of the vertices and edges of the underlying graph, plus the nodes of the tree decomposition. The vocabulary, which we call the *vocabulary of tree decompositions*, consists of:

$$\underbrace{\mathsf{node}}_{\text{a unary predicate}} \qquad \underbrace{\mathsf{incident} \quad \mathsf{bag} \quad \mathsf{parent}}_{\text{binary predicates}}$$

The predicate $\mathsf{incident}$ describes the incidence relation in the underlying graph. The predicate $\mathsf{node}(x)$ says that x is a decomposition node, $\mathsf{bag}(v,x)$ says that vertex v is in the bag of node x, and $\mathsf{parent}(x,y)$ says that node x is the parent of node y.

MSO ***interpretations.*** We now define what it means to produce a tree decomposition (or some other structure) using MSO. We do this by using three types of basic operations on logical structures defined below, called *copying*, *coloring*, and *interpreting*. All three types describe binary relations on logical structures: copying is a function, coloring is a relation, and interpreting is a partial function.

1. *Copying.* Define the k*-copy* of a logical structure $\mathfrak{A}$ to be k disjoint copies of $\mathfrak{A}$, with the following fresh predicates added to the vocabulary:
$$\mathsf{copy}(a,b)\ ,\ \mathsf{layer}_1(a)\ ,\ \dots\ ,\ \mathsf{layer}_k(a).$$
The binary predicate copy checks whether two elements are copies of the same element of the original structure, whereas the unary predicate layer_i checks whether an element belongs to the i-th copy (called also the i-th *layer*).
2. *Coloring.* Define an k*-coloring* of a structure $\mathfrak{A}$ to be any structure obtained from $\mathfrak{A}$ by adding new unary predicates $X_1, \dots, X_k$ to the vocabulary and interpreting them as any subsets of the universe.
3. *Interpreting.* The syntax of an interpretation consists of an *input vocabulary* Σ, an *output vocabulary* Γ and a family of MSO formulas
$$\{\varphi_{\mathsf{dom}}, \varphi_{\mathsf{univ}}\} \cup \{\varphi_R\}_{R\in\Gamma},$$
over the input vocabulary Σ. The formula φ_{dom} has no free variables, the formula φ_{univ} has one free variable, and each formula φ_R has as many free variables as the arity of R. The free variables in all of these formulas range over elements, not sets of elements. If $\mathfrak{A}$ is a logical structure over the input vocabulary Σ that satisfies φ_{dom}, we define the output logical structure, which is over the output vocabulary Γ, as follows. The universe is the universe of $\mathfrak{A}$ restricted to elements satisfying φ_{univ} and each relation R of the output vocabulary is interpreted as those tuples in the universe which make φ_R true. If φ_{dom} is not satisfied in $\mathfrak{A}$, then the output of the interpretation is undefined.

Definition 2.3. An MSO *transduction* is a finite composition of the three types of operation defined above (treated as relations between logical structures), together with prescribed input and output vocabularies. If coloring is not used, we talk about a *deterministic* MSO transduction. If $\mathcal{I}$ is an MSO transduction, and $\mathfrak{A}$ is a structure over the input vocabulary, then by $\mathcal{I}(\mathfrak{A})$ we denote the *output* of $\mathcal{I}$, defined as the set of all structures over the output vocabulary that are in relation defined by $\mathcal{I}$ with $\mathfrak{A}$.

The definition above is equivalent to the one in [6]; this equivalence follows from [6, Theorem 1.39]. The crucial property of MSO transductions is the Backwards Translation Theorem [6, Theorem 1.40], which says that if $\mathcal{I}$ a MSO transduction and ψ is an MSO sentence over the output vocabulary, then

$$\{\mathfrak{A} : \text{at least on structure in } \mathcal{I}(\mathfrak{A}) \text{ satisfies } \psi\}$$

is a set of structures over the input vocabulary that is definable in MSO (for completeness, we give a proof sketch adjusted to our notation as Lemma B.1 in Appendix B). Using this result, we may apply MSO transductions to enrich the input structure with MSO-definable objects, and any property that can be defined in MSO afterwards, can be also defined directly in the input structure.

2.2 The main result

We are now ready to state our main technical result, which says that an MSO transduction can compute tree decompositions for graphs of bounded treewidth. We use the name *transduction from graphs to tree decompositions* if the input vocabulary is the vocabulary of graphs $\{\mathsf{incident}(x,y)\}$ and the output vocabulary is the vocabulary of tree decompositions defined previously.

Theorem 2.4. *There is a function $f : \mathbb{N} \to \mathbb{N}$ such that for every $k \in \mathbb{N}$ there exists an* MSO *transduction $\mathcal{I}$ from graphs to tree decompositions, such that every graph G satisfies:*

(1) Every output $\mathcal{I}(G)$ represents a tree decomposition of G of width at most $f(k)$.
(2) If G has treewidth at most k, then the output $\mathcal{I}(G)$ is nonempty.

We actually believe that a stronger variant of the above theorem holds, with f being the identity. In other words, we believe that there is an MSO transduction which inputs a graph of treewidth k, and produces a tree decomposition of width k. In order to prove the stronger version, it would be sufficient to show that for every $k \leq k'$, there is an MSO transduction which realizes the following task: given a tree decomposition of width k', produce, if possible, a tree decomposition of width k for the same graph. Our idea for a proof of this statement is to take a closer look at the algorithm of Bodlaender and Kloks [2] that solves exactly this task, and try to simulate it using an MSO transduction (even a deterministic one). We expand this topic in the concluding section (Section 6).

The proof of Theorem 2.4 consists of two steps, described below.

The special case of bounded pathwidth. The first step is to prove a weaker variant of the theorem. This variant has exactly the same statement, except that in condition (2) the assumptions are strengthened to requiring that the pathwidth of the graph is at most k. This weaker variant, Lemma 2.5 below, is proved in Section 4.

Lemma 2.5. *There is a function $f : \mathbb{N} \to \mathbb{N}$ such that for every $k \in \mathbb{N}$ there exists an* MSO *transduction $\mathcal{I}$ from graphs to tree decompositions, such that every graph G satisfies:*

(1) Every output $\mathcal{I}(G)$ represents a tree decomposition of G of width at most $f(k)$.
(2) If G has pathwidth at most k, then the output $\mathcal{I}(G)$ is nonempty.

There are two crucial ingredients in the proof of Lemma 2.5.

The first ingredient is that for path decompositions, we can use semigroup theory. Specifically, we use Factorisation Forest Theorem of Imre Simon [15]. The application of this result is the cornerstone of our approach, and it enables us to recursively decompose any graph of pathwidth k into constant-size pieces using only $f(k)$ levels of recursion — a number that depends on k alone, and not on the size of the graph. Lemma 2.5 then follows by verifying that each level incurs a fixed blow-up of the width of tree decompositions that we are able to describe in MSO.

The second ingredient is the definition of a *guidance system*, which is a combinatorial object used to describe additional structure in a graph, e.g., a tree decomposition, in a way that can be guessed by a MSO transduction. Guidance systems are introduced in Section 3, and are used throughout the whole paper to describe "MSO-guessable" tree decompositions.

Tree decompositions with bags of bounded pathwidth. In the second step, presented in Section 5, we show that there is an MSO transduction which inputs a graph of treewidth at most k, and outputs a tree decomposition of the graph where the bags are maybe arbitrarily large, but have pathwidth bounded by $2k + 1$, in the following sense. For a node x in a tree decomposition t, define its *marginal graph* as follows: take the subgraph of the underlying graph induced by the margin of x, and add an edge uv for every pair $\{u, v\}$ that appears together in the adhesion of some child node of x, provided this edge is not already included in the graph.

Lemma 2.6. *For every $k \in \mathbb{N}$, there exists an* MSO *transduction $\mathcal{B}$ from graphs to tree decompositions such that for every graph G the following holds:*

(1) Every output $\mathcal{B}(G)$ represents a sane tree decomposition of G.
(2) If G has treewidth at most k, then $\mathcal{B}(G)$ contains at least one tree decomposition where all marginal graphs have pathwidth at most $2k + 1$.

In Lemma 2.6 we use a technical notion of a *sane tree decomposition*, which is defined below.

Definition 2.7. A tree decomposition t of a graph G is called *sane* if the following conditions are satisfied for every node x:

(a) the margin of x is nonempty;
(b) the subgraphs induced in G by the cone of x and by the component of x are connected;
(c) every vertex of the adhesion of x has a neighbor in the component of x.

Intuitively, saneness means that the decomposition respects the connectivity of subgraphs corresponding to its subtrees. Indeed, it is straightforwards to see from the definition, that all the marginal graphs of a sane decomposition are nonempty and connected (connectivity follows from property (b) of saneness). A similar notion of *internal connectivity* was used by Lapoire [14]. The following lemma, which may be considered folklore, shows that any tree decomposition can be sanitized.

Lemma 2.8. *Suppose t is a tree decomposition of a graph G. Then there exists a sane tree decomposition s of G where every bag in s is a subset of some bag in t. In particular, if $\mathtt{tw}(G) \leq k$, then G admits a sane tree decomposition of width at most k.*

The above lemma is colored red because its proof can be found in the appendix. Throughout the paper, we use this convention to mark statements whose proofs are straightforward or not important for the main narrative; their formal verification is deferred to the appendix in order not to spoil the natural flow of argumentation.

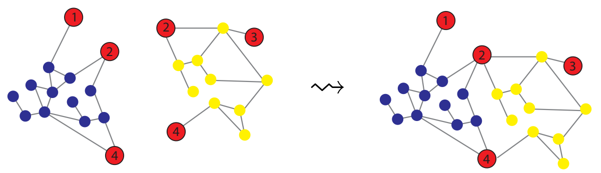
Proof of Theorem 2.4. The proof of Theorem 2.4 is a combination of Lemmas 2.5 and 2.6. This requires some technical care, but does not involve any substantially new ideas. We give a full exposition of this proof in Appendix A.

2.3 Courcelle's conjecture

In this section we use Theorem 2.4 to prove the conjecture of Courcelle mentioned in the introduction. We use a syntax slightly different than Courcelle.

Definition 2.9. Define an *interface graph* to consist of an *arity* $k \in \mathbb{N}$, an *underlying graph* G called the *underlying graph* and an *interface mapping*, which is an injective partial function from $\{1, 2, \ldots, k\}$ to vertices of the underlying graph. If image of $i \in \{1, 2, \ldots, k\}$ under the interface mapping is defined, it is called the *i-th interface vertex*. Then i is the *name* of this interface vertex.

Interface graphs of arity k are called k-interface graphs. If $\mathbb{G}$ and $\mathbb{H}$ are k-interface graphs, then their *gluing* $\mathbb{G} \oplus \mathbb{H}$ is the k-interface graph defined as follows. The underlying graph is the disjoint union of the two underlying graphs, with the interface vertices having the same names in $\mathbb{G}$ and $\mathbb{H}$ being fused. In other words, for any name $i \in \{1, 2, \ldots, k\}$ that is used both in $\mathbb{G}$ and in $\mathbb{H}$ (i.e. is in the intersection of the domains of the interface mappings), we fuse the corresponding i-th interface vertices in $\mathbb{G} \oplus \mathbb{H}$. If this process creates any parallel edges, we remove the duplicates. The names of the interface vertices in $\mathbb{G} \oplus \mathbb{H}$ are inherited from the arguments. We illustrate this definition with the following example:



In Courcelle's syntax from [5], the gluing essentially corresponds to substituting $\mathbb{H}$ for the hyperedge consisting of the interface vertices inside $\mathbb{G}$.

Recognisability. Let Π be a property of graphs. We say that two k-interface graphs $\mathbb{G}_1$ and $\mathbb{G}_2$ are Π-equivalent if

$$\mathbb{G}_1 \oplus \mathbb{H} \text{ satisfies } \Pi \qquad \text{iff} \qquad \mathbb{G}_2 \oplus \mathbb{H} \text{ satisfies } \Pi.$$

holds for every k-interface graph $\mathbb{H}$. This is an equivalence relation on k-interface graphs. If there are finitely many equivalence classes, then we say that Π is *k-recognisable*. Finally, we say that Π is *recognisable* if it is k-recognisable for every k. This is equivalent to Definition 1.12 in [5].

Courcelle's conjecture. In what follows, we consider the logic *counting* MSO which is the extension of MSO on graphs by predicates of the form "the size of set X is divisible by m" for every constant m. The following result was stated as Conjecture 1 in [5].

Theorem 2.10. *If a property of graphs that have treewidth bounded by a constant is recognisable, then it is definable in counting* MSO.

As mentioned in the introduction, the converse implication was proved by Courcelle in [4]. In his later work [5], Courcelle proposed the following approach to proving Theorem 2.10. Call a property of graphs Π *strongly context-free* if (informally), given any graph G from Π, some constant-width decomposition of G can be nondeterministically defined in MSO. If we prove that the class of graphs of treewidth k is strongly context-free (which is stated as Conjecture 2 in [4]), then Theorem 2.10 would follow from the following lemma.

Lemma 2.11. *Let $k \in \mathbb{N}$ and let Π be a k-recognisable property of graphs. There is a formula of counting* MSO *over the vocabulary of tree decompositions which is true in exactly those structures which represent a tree decomposition of width k where the underlying graph satisfies Π.*

Lemma 2.11 is essentially proved in [5] (cf. Theorem 4.8 therein), but for the sake of completeness we give a proof adjusted to our notation in the appendix. The statement that the class of graphs of treewidth at most k is strongly context-free is, up to insignificant differences in definitions, equivalent to our Theorem 2.4. Hence, we can complete the proof of Theorem 2.10 as Courcelle suggested.

Proof of Theorem 2.10. Let Π be a property of graphs of treewidth at most k. Apply Theorem 2.4 to k, yielding an MSO transduction, which maps graphs of treewidth at most k to tree decompositions of width at most $f(k)$. Apply Lemma 2.11 to $f(k)$ and the property Π, yielding a formula of counting MSO which tests Π on tree decompositions of width at most $f(k)$. The result follows by using the Backwards Translation Theorem. □

We believe that a stronger statement holds, namely: if Π is a property of graphs of treewidth k, then already being k-recognisable implies definability in counting MSO. This claim would follow from the stronger version of Theorem 2.4 described after its statement.

3. Guidance systems

In this section, we introduce guidance systems. The definition is a variant of the guidance systems defined in [3]. Guidance systems are used in the proofs of Lemmas 2.5 and 2.6, which are found in Sections 4 and 5.

Definition 3.1. A *guidance system* Λ over a graph G is a family of in-trees (i.e. connected in-forests), where each in-tree is obtained by orienting the edges of some subgraph of G. For a vertex u, define

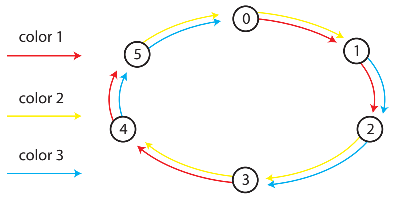
$$\Lambda(u) = \{v : \text{some tree from } \Lambda \text{ contains } u \text{ and has root } v\}.$$

A *coloring* of a guidance system is an assignment of trees to colors so that trees with the same color have disjoint vertex sets.

Example 3.2. Let G be an undirected cycle of length six, with vertices called $\{0, 1, \ldots, 5\}$, and edges being neighbor modulo 6. Consider the following guidance system

$$\Lambda = \{u \to u+1 \to u+2 : u \in \{0, \ldots, 5\}\}$$

where addition is modulo 6. This guidance system is 3-colorable:



For this guidance system,

$$\Lambda(u) = \{u, u+1, u+2 \mod 6\} \qquad \text{for } u \in \{0, \ldots, 6\}.$$

Guidance system are used to recognize sets of vertices in a graph, in the sense defined below.

Definition 3.3. Let G be a graph. A set of vertices X is said to be *captured* by a vertex u in a guidance system Λ if $X \subseteq \Lambda(u)$ holds. A family of sets of vertices is said to be captured by Λ if each set is captured by some vertex.

From adhesions to a tree decomposition. In Lemma 3.4 below, we show that in order to produce a sane tree decomposition with an MSO interpretation, it suffices to capture all its adhesions with a bounded number of colors. Note that in the lemma below we do not restrict the sizes of bags in the tree decompositions; e.g., a decomposition with all vertices in one bag has no adhesions and therefore falls into the scope of the lemma.

Lemma 3.4. *For every $k \in \mathbb{N}$ there is an* MSO *transduction from graphs to tree decompositions which maps every graph G to all sane tree decompositions of G whose family of adhesions can be captured by a k-colorable guidance system over G.*

The proof of Lemma 3.4, which can be found in Appendix C.2, is actually quite non-trivial. We use the connectivity conditions given by saneness in order to be able to guess the bags of a sane tree decomposition. Also, instead of the original graph, we need to work with the graph obtained by turning all adhesions into cliques. This structure can be constructed by an MSO transduction which guesses a guidance system that captures all the adhesions. The fact that the guidance system can be colored using few colors is necessary for it to be guessable in MSO.

Stability under small modifications. In Lemma 3.5 below, we show that the number of colors needed to capture a family of sets by a guidance system is stable under removing or adding vertices. If G is a graph, we write $G - u$ for the subgraph induced by removing u from the vertices.

Lemma 3.5. *Let G be a graph, let $\mathcal{X}$ be a family of subsets of vertices, and let u be a vertex.*

(1) If every set in $\mathcal{X}$ is contained in some connected component of G and

$$\mathcal{X} - u \overset{def}{=} \{X - \{u\} : X \in \mathcal{X}\}$$

is captured by a k-colorable guidance system over G, then $\mathcal{X}$ is captured by a $(k+1)$-colorable guidance system over G.

(2) If every set from $\mathcal{X}$ is contained in some connected component of $G - u$ and $\mathcal{X}$ is captured by a k-colorable guidance system over G, then $\mathcal{X}$ is captured by a $2k$-colorable guidance system over $G - u$.

4. Graphs of bounded pathwidth

In this section we prove Lemma 2.5, which says that an MSO transduction can transform a graph of bounded pathwidth into a tree decomposition. Our proof relies on the guidance systems defined in the previous section. We first outline the plan. In Section 4.1, we define a graph parameter called guided treewidth. In Section 4.2 we show that bounded pathwidth implies bounded guided treewidth. A combination of this result with the fact that guidance system can be expressed in MSO (Lemma 3.4) yields Lemma 2.5.

4.1 Guided treewidth

The following definition can be seen as a new graph parameter.

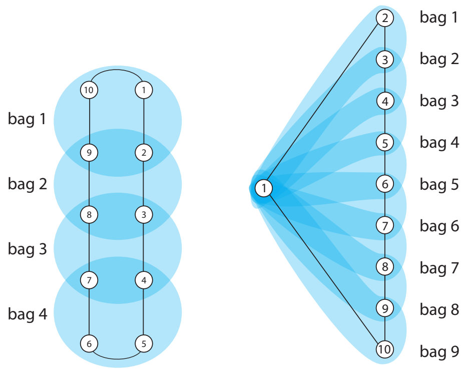
Definition 4.1. Define the *guided treewidth* of a graph G, denoted by $\mathtt{gtw}(G)$, to be the smallest k such that there exists a tree decomposition of G where all bags are captured by some k-colorable guidance system over G.

Note that if a bag is captured by a k-colorable guidance system, then it has size at most k. Hence, the guided treewidth of a graph is an upper bound on its treewidth. Since adhesions are contained in bags, tree decompositions whose bags are captured by a k-colorable guidance system fall into the scope of Lemma 3.4, and can be produced by an MSO transduction.

The goal of this section is to show that bounded pathwidth implies bounded guided treewidth, and therefore, by the above discussion, tree decompositions of graphs of bounded pathwidth can be produced by an MSO transduction.

To illustrate guided treewidth, we show an example where a poorly chosen tree decomposition needs a large number of colors to be captured.

Example 4.2. Consider a cycle with vertices $\{1, \ldots, 2n\}$. For this cycle, consider a path decomposition with $n-1$ bags, where the i-th bag contains vertices $i, i+1, 2n-i$, and $2n-i+1$. A picture for $n = 5$ is in the left panel of the figure below:



The path decomposition above has suboptimal width, but we could make it optimal by extending the graph by a disjoint clique of size 4.

It is now not hard to verify that any guidance system Λ that captures this tree decomposition, requires a number of colors that is linear in n. Indeed, for $i \in \{1, \ldots, n-1\}$, let u_i be the vertex that captures the bag $\{i, i+1, 2n-i, 2n-i+1\}$ in Λ. Hence, there are four in-trees, respectively with roots i, $i+1$, $2n-i$, and $2n-i+1$, such that each of them contains u_i. Consequently, wherever u_i lies, it must hold that one of these trees contains one of the following four arcs: (v_1, v_{2n}), (v_{2n}, v_1), (v_n, v_{n+1}), or (v_{n+1}, v_n). If we now restrict our attention only to odd indices i, the corresponding bags of the decomposition are disjoint, and hence one of arcs above belongs to at least $\frac{n-1}{8}$ different in-trees of Λ. These trees must receive pairwise different colors in any coloring of Λ.

To fix the problem we use a different path decomposition, with $2n-2$ bags, such that the u-th bag contains $\{1, u, u+1\}$. This decomposition is depicted in the right panel of the figure above. To capture its bags, we use a 3-colorable guidance system colorable. The first color is used to describe a directed path

$$2 \to 3 \to \cdots \to 2n \to 1.$$

The remaining two colors are used alternately to connect each vertex with its successor, similarly as in Example 3.2. Concluding, each cycle admits a tree (even path) decomposition that can be captured by a 3-colorable guidance system.

In the next section, we strengthen the result from the above example, and show that bounded pathwidth implies bounded guided treewidth. Before passing to the next section, we show that guided treewidth is robust with respect to graph operations like disjoint union (denoted $\uplus$) or adding/removing a single vertex. The proof is a simple application of Lemma 3.5.

Lemma 4.3. *Let G, G' be graphs and let u be a vertex in G. Then*

$$\begin{aligned} \mathtt{gtw}(G \uplus G') &= \max(\mathtt{gtw}(G), \mathtt{gtw}(G')) && (1)\\ \mathtt{gtw}(G) &\leq \mathtt{gtw}(G-u)+1 && (2)\\ \mathtt{gtw}(G-u) &\leq 2 \cdot \mathtt{gtw}(G) && (3) \end{aligned}$$

4.2 Guided treewidth is bounded by pathwidth

We now state and prove the main result of Section 4, which is that bounded pathwidth implies bounded guided treewidth.

Lemma 4.4. *There exists a function $f(k) \in 2^{2^{\mathcal{O}(k^2)}}$ such that*

$$\mathtt{gtw}(G) \leq f(\mathtt{pw}(G)) \qquad \textit{for every graph } G.$$

Note the asymmetry in the lemma: we assume bounded pathwidth, but produce a tree decomposition. It can be easily seen that Lemma 2.5 follows by combining Lemma 4.4 with Lemmas 2.8 and 3.4. A full proof of this implication is in Appendix D.

The rest of Section 4 is devoted to proving Lemma 4.4. In Section 4.2.1 we define bi-interface graphs, which give an alternative algebraic definition of pathwidth. Then, in Section 4.2.2, we use the Factorization Forest Theorem to prove Lemma 4.4.

4.2.1 Bi-interface graphs

Recall the interface graphs as defined in Section 2.3. In our approach to pathwidth, we use such an enriched version of this definition, where a graph is supplied with two sets of interfaces: left and right. Here is the formal definition.

Definition 4.5. A *bi-interface graph* consists of an *arity* $k \in \mathbb{N}$, an *underlying graph* G, and two partial injective functions $\mathsf{left}, \mathsf{right}$ from $\{1, \ldots, k\}$ to the vertices of G. We use the name *i-th left interface* for $\mathsf{left}(i)$, likewise for the i-th right interface. Moreover, we require that if a vertex is simultaneously an i-th left interface and a j-th right interface, then $i = j$.

Thus, we assume that the interface names of a bi-interface graph of arity k are numbers between 1 and k, however not all of them need to be used.

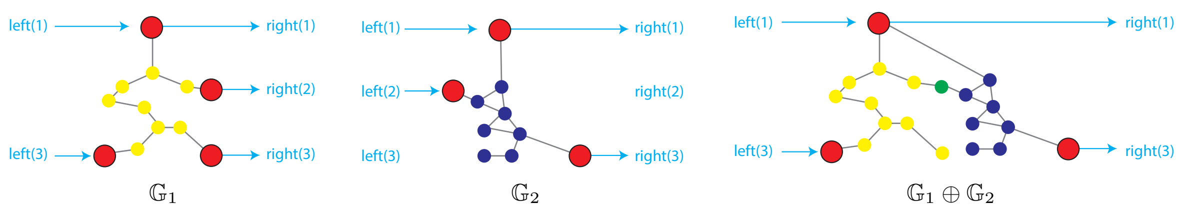


Figure 1: Two bi-interface graphs and their gluing. The interface nodes are the red ones, the incoming arrows indicate left interfaces and the outgoing arrows indicate right interfaces. Note how some of the left or right interfaces are undefined, e.g. the second left interface in $\mathbb{G}_1$, and how the first left and right interfaces are equal.

In Section 2.3 we showed how to glue interface graphs. For bi-interface graphs we use a similar notion, which is probably best seen in a picture, see Figure 1. Here is the formal definition. Let $\mathbb{G}_1, \mathbb{G}_2$ be two bi-interface graphs of the same arity k. Define their gluing $\mathbb{G}_1 \oplus \mathbb{G}_2$ as follows. Take the disjoint union of the underlying graphs, and fuse the i-th right interface of $\mathbb{G}_1$ with the i-th left interface of $\mathbb{G}_2$, whenever both are defined. As before, remove the duplicates whenever any parallel edge is created in this operation. As the left interface function take the left interface function of $\mathbb{G}_1$, and as the right interface function take the right interface function of $\mathbb{G}_2$. It is easy to verify that if $\mathbb{G}_1$ and $\mathbb{G}_2$ were both bi-interface graphs of arity k, then $\mathbb{G}_1 \oplus \mathbb{G}_2$ is also a bi-interface graph of arity k. Note that in $\mathbb{G}_1 \oplus \mathbb{G}_2$ we forget the information about the right interfaces of $\mathbb{G}_1$ and the left interfaces of $\mathbb{G}_2$.

The gluing operation defined above is associative, turning the set of bi-interface graphs of arity k into a semigroup. A product

$$\mathbb{G}_1 \oplus \ldots \oplus \mathbb{G}_n$$

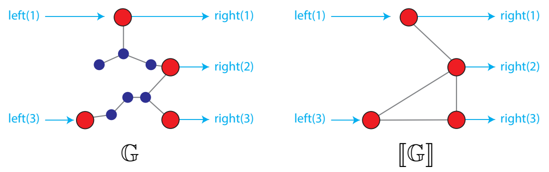
in this semigroup is essentially the same thing as a path decomposition, where the bags are the bi-interface graphs $\mathbb{G}_1, \ldots, \mathbb{G}_n$, and the interface functions say how the bags are connected. Hence the following lemma, whose straightforward proof is omitted.

Lemma 4.6. *A graph has pathwidth at most k if, and only if it is the underlying graph of a bi-interface graph of the form*

$$\mathbb{G}_1 \oplus \ldots \oplus \mathbb{G}_n$$

where each $\mathbb{G}_i$ has arity k and at most $k+1$ vertices.

Abstraction. Call two bi-interface graphs *isomorphic* if there is a bijection between their vertex sets that respects graph edges and the name of each interface. For a graph G and a subset of vertices X, by the *torso* of G with respect to X we mean a graph on vertex set X where two vertices are adjacent if they can be connected in G by a path whose internal vertices do not belong to X. Define the *abstraction* $[\![\mathbb{G}]\!]$ of a bi-interface graph $\mathbb{G}$ to be the isomorphism type of the bi-interface graph obtained by taking the torso of the underlying graph with respect to the interface vertices, and preserving the interface functions. Here is a picture of a bi-interface graph and its abstraction:



Define $\mathbb{A}_k$ to be the possible abstractions of bi-interface graphs of arity k. This is a finite set of size $2^{\mathcal{O}(k^2)}$. The abstraction function $\mathbb{G} \mapsto [\![\mathbb{G}]\!]$ is compositional in the sense that $[\![\mathbb{G}_1 \oplus \mathbb{G}_2]\!]$ is uniquely determined by $[\![\mathbb{G}_1]\!]$ and $[\![\mathbb{G}_2]\!]$. This means that $\mathbb{A}_k$ can be uniquely endowed with a multiplication operation which makes the abstraction function a semigroup homomorphism from bi-interface graphs of arity k to $\mathbb{A}_k$. From now on we will treat $\mathbb{A}_k$ as a semigroup, and the abstraction function as a homomorphism.

4.2.2 Simon's Factorization Forest Theorem

We now recall Simon's Factorization Forest Theorem from semigroup theory, whose application is the cornerstone of our proof of Lemma 2.5. Intuitively, the theorem says that if h is a homomorphism from words into a finite semigroup S, then every word can be recursively factorised until reaching single letters, with the depth of the factorisation depending only on the size of S, and each factorisation step respecting h in some way.

We write Σ^+ for the semigroup of nonempty finite words over alphabet Σ with concatenation. Let S be a finite semigroup and let

$$h : \Sigma^+ \to S$$

be a semigroup homomorphism. We define two types of factorizations for a word $u \in \Sigma^+$.

- **Binary**: a factorization into two factors
$$u = u_1 u_2.$$
- **Unranked**: a factorization into an arbitrary number of factors
$$u = u_1 \ldots u_n,$$
such that all factors $u_1, \ldots, u_n$ have the same image under h.

The *h-rank* of a word $u \in \Sigma^+$ is the natural number defined by induction on the length of u as follows. Single letters have rank 1. For a longer word u, its h-rank is

$$1 + \min_{u=u_1\ldots u_n} \max_{i\in\{1,\ldots,n\}} h\text{-rank of } u_i$$

where the minimum ranges over factorizations (either binary or unranked) of u into nonempty factors. Using binary factorizations only, it is easy to see that every word has h-rank that is at most logarithmic in its length. Imre Simon proved that thanks to the unranked factorisations, one can achieve constant rank.

Theorem 4.7 (Factorisation Forest Theorem [15, 13])**.** *If $h\colon \Sigma^+ \to S$ is a semigroup homomorphism with S finite, then every word in Σ^+ has h-rank at most $3|S|$.*

The original result there is actually slightly stronger: in the unranked factorisations only idempotent values in the semigroup can be used. We will not use idempotence. The bound $3|S|$ is from [13], improving on the original exponential bound of Simon [15]. The word *forest* in the theorem's name is because the definition of rank implicitly uses a tree structure of factorisations.

We now return to the proof of Lemma 4.4, which says that guided treewidth is bounded by a function of the pathwidth. Consider the semigroup homomorphism

$$h : \Sigma^+ \to \mathbb{A}_k,$$

where Σ is the set of arity-k bi-interface graphs with at most $k+1$ vertices, and h is the homomorphism that glues a word to a single bi-interface graph and takes its abstraction. We will show that for every

$$\mathbb{G}_1 \cdots \mathbb{G}_n \in \Sigma^+$$

the guided treewidth of the corresponding glued graph is at most exponential in the h-rank of the word. More precisely:

$$\mathtt{gtw}(\mathbb{G}_1 \oplus \cdots \oplus \mathbb{G}_n) \leq 2^{\mathcal{O}(k \cdot (h\text{-rank}(\mathbb{G}_1 \cdots \mathbb{G}_n)))}. \tag{4}$$

By Lemma 4.6 and the Factorisation Forest Theorem, every graph of pathwidth k can be obtained from some word in Σ^+ with h-rank bounded by $3|\mathbb{A}_k| = 2^{\mathcal{O}(k^2)}$, thus proving Lemma 4.4. It remains to show (4). For this, we use induction on the h-rank. The induction base follows from the observation that the guided treewidth of a graph is at most the number of its vertices, which follows directly from claim (2) of Lemma 4.3. For the induction step, we use the Binary and Unranked lemmas stated below. The Binary lemma, used for binary factorizations, follows immediately from Lemma 4.3.

Lemma 4.8 (Binary)**.** *If $\mathbb{G}_1, \mathbb{G}_2$ are arity-k bi-interface graphs, then*

$$\mathtt{gtw}(\mathbb{G}_1 \oplus \mathbb{G}_2) \leq k + 2^k \cdot \max(\mathtt{gtw}(\mathbb{G}_1), \mathtt{gtw}(\mathbb{G}_2)).$$

Proof. Observe that $\mathbb{G}_1 \oplus \mathbb{G}_2$ can be obtained from $\mathbb{G}_1$ and $\mathbb{G}_2$ by (i) removing from both graphs those interface vertices that get fused during gluing, (ii) taking the disjoint union of the obtained graphs, and (iii) reintroducing the removed interface vertices to the result. By claim (3) of Lemma 4.3, operation (i) can increase the guided treewidth of each of the graphs by a multiplicative factor of at most 2^k. By claim (1) of Lemma 4.3, in operation (ii) we obtain a graph with guided treewidth not exceeding the maximum of the guided treewidth of the factors. Finally, by claim (2) of Lemma 4.3, in operation (iii) the guided treewidth increases by an additive factor of at most k. Hence the bound follows. □

For unranked factorisations, we use the following lemma.

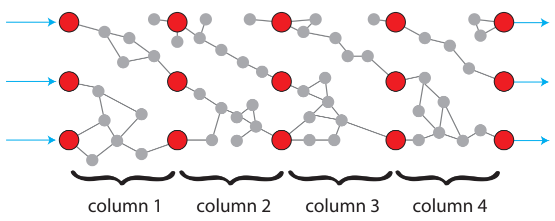
Lemma 4.9 (Unranked)**.** *Let $\mathbb{G}_1, \mathbb{G}_2, \ldots, \mathbb{G}_n$ be bi-interface graphs of arity k which all have the same abstraction. Then*

$$\mathtt{gtw}(\mathbb{G}_1 \oplus \cdots \oplus \mathbb{G}_n) \leq k(4k^2+5) + 8^k \cdot \max_{i \in \{1,\ldots,n\}} \mathtt{gtw}(\mathbb{G}_i).$$

The proof of the Unranked Lemma is more involved and we present it in the following section.

4.2.3 Proof of the Unranked Lemma

We begin by introducing some notation. If $\mathbb{G}_1, \ldots, \mathbb{G}_n$ are bi-interface graphs of the same arity and $i \in \{1, \ldots, n\}$, then there is a natural injective mapping column_i that associates the vertices of $\mathbb{G}_i$ with the corresponding vertices of its copy in $\mathbb{G}_1 \oplus \cdots \oplus \mathbb{G}_n$. Let us call this function the *i-th column function*, and its image the *i-th column*. Here is a picture that illustrates columns:



We begin by observing that if we glue many copies of the same bi-interface graph, then vertices in the same column can either be connected by a path visiting few columns, or not at all. Here, by a path *staying within* a set of columns we mean that the vertex set of the path is contained in the union of these columns.

Lemma 4.10. *Let $\mathbb{G}$ be a bi-interface graph. If two vertices of*

$$\mathbb{G}^n \overset{def}{=} \overbrace{\mathbb{G} \oplus \cdots \oplus \mathbb{G}}^{n \text{ times}}.$$

are in the same column $i \in \{1, \ldots, n\}$ and can be connected by a path, then they can also be connected by a path that stays only in columns j satisfying $|j - i| \leq k^2$, where k is the arity of $\mathbb{G}$.

Proof. Consider a path in $\mathbb{G}^n$ which connects two vertices from the i-th column. Cut this path whenever it visits an interface vertex of the i-th column, thus decomposing it into subpaths of three different types:

(i) stays within column i;
(ii) connects a pair of left interfaces of column i, and stays within columns $< i$;
(iii) connects a pair of right interfaces of column i, and stays within columns $> i$.

We show that a path of each type can be modified so that it stays only within columns with indices differing by at most k^2 from i. For type (i), this is already true. By symmetry, we only treat type (ii). We assume that $i > k^2 + 1$, because otherwise we are already done.

For $m \in \mathbb{N}$, define

$$\mathrm{Left}_m \subseteq \{1, \ldots, k\} \times \{1, \ldots, k\}$$

to be the set of pairs (x, y) such that there is a path in $\mathbb{G}^n$ which connects the x-th left interface in column m with the y-th left interface in column m, and stays within columns $< m$. Observe that set Left_{m+1} is defined only in terms of Left_m; formally, there is a function Φ that depends on $\mathbb{G}$ only such that $\mathrm{Left}_{m+1} = \Phi(\mathrm{Left}_m)$. This is because any path certifying that some pair belongs to Left_{m+1} can be decomposed into subpaths staying within column m and connecting some its interfaces, and subpaths contributing to the definition of Left_m, i.e., certifying that some pair belongs to Left_m. Hence, whether some pair belongs to Left_{m+1} depends only on the existence of connections between interfaces of $\mathbb{G}$ within this graph, and the information encoded in Left_m. Moreover, we have that $\mathrm{Left}_{m+1} \supseteq \mathrm{Left}_m$, because any path certifying that some pair belongs to Left_m can be shifted one column to the right (due to all columns being isomorphic), and then it certifies that the same pair belongs to Left_{m+1}. Therefore we have that

$$\mathrm{Left}_1 \subseteq \mathrm{Left}_2 \subseteq \mathrm{Left}_3 \subseteq \ldots$$

Since each of these sets has size at most k^2, the sequence must contain two equal consecutive sets after at most $k^2 + 1$ steps. Moreover, since Left_{m+1} is defined only in terms of Left_m, the sequence must stabilize from this point on. We conclude that $\mathrm{Left}_j = \mathrm{Left}_{k^2+1}$ for all $j > k^2$, and in particular $\mathrm{Left}_i = \mathrm{Left}_{k^2+1}$. This means that any path of type (ii), say connecting the x-th and y-th interface of column i, can be replaced by taking a path certifying that pair (x, y) belongs also to Left_{k^2+1}, and shifting it $i - (k^2 + 1)$ columns to the right. Then the replaced path stays within the k^2 columns to the right of column i. □

We now show that if we glue many bi-interface graphs which are not necessarily equal but have the same abstraction, then there is a guidance system, colorable by a small number of colors, which takes each vertex to the reachable interfaces from its own column.

Lemma 4.11. *Let $\mathbb{G}_1, \mathbb{G}_2, \ldots, \mathbb{G}_n$ be bi-interface graphs with the same arity k and the same abstraction, and such that the left and right interfaces are disjoint. Consider the gluing*

$$\mathbb{G} = \mathbb{G}_1 \oplus \cdots \oplus \mathbb{G}_n.$$

For $i \in \{1, \ldots, n\}$ and a vertex u in the i-th column, define $I_i(u)$ to be those interface vertices of the i-th column which are reachable *from u by a path in $\mathbb{G}$. Then there is a guidance system Λ over $\mathbb{G}$, which can be colored by at most $4k(k^2+1)$ colors, such that*

$$I_i(u) \subseteq \Lambda(u) \quad \textit{for each } i \in \{1,\dots,n\} \textit{ and } u \textit{ in the } i\textit{-th column.}$$

Proof. Using Lemma 4.10 applied to the common abstraction of graphs $\mathbb{G}_i$, and the definition of the abstraction operation, we get the following claim.

Claim 1. *For every $v \in I_i(u)$ there is a path from u to v which stays only within columns j satisfying $|i-j| \le k^2$.*

Let $i \in \{1,\dots,n\}$ and let v be an interface vertex in the i-th column. Consider the subgraph of $\mathbb{G}$ induced by all vertices that can be reached from v via paths that stay within columns j with $|i-j| \le k^2$; this subgraph is connected by definition. Choose an arbitrary spanning tree of this subgraph, and call it $t_{i,v}$. Choose v to be the root of this tree, and direct all edges of the tree toward the root. Define Λ to be the family

$$\{t_{i,v} : i \in \{1,\dots,n\} \text{ and } v \text{ is an interface in the } i\text{-th column}\}$$

By Claim 1, we know that $v \in I_i(u)$ implies that u belongs to the tree $t_{i,v}$, and therefore $I_i(u) \subseteq \Lambda(u)$. By the assumption on the left and right interfaces being disjoint, columns of $\mathbb{G}$ are disjoint unless they are consecutive. Therefore, the trees $t_{i,v}$ and $t_{j,w}$ are disjoint whenever $|i-j|$ is greater than $2k^2$. Finally, in each column there are at most $2k$ interface vertices. This means that Λ can be colored using $(2k^2+1)\cdot 2k$ colors as follows: each tree $t_{i,v}$ receives a color depending on the name of interface v (left or right, and a number between 1 and k) and the remainder of i modulo $2k^2+1$. ☐

The next step is to prove the Unranked Lemma in the special case where the bi-interface graphs have disjoint left and right interfaces. The proof is an application of Lemma 4.11, which ensures us that connections that have to be realized by the sought guidance system have a local character.

Lemma 4.12. *Let $\mathbb{G}_1, \mathbb{G}_2, \dots, \mathbb{G}_n$ be bi-interface graphs of arity k which all have the same abstraction, and such that the left and right interfaces are disjoint. Then*

$$\mathtt{gtw}(\mathbb{G}_1 \oplus \cdots \oplus \mathbb{G}_n) \le 4k(k^2+1) + 4^k \cdot \max_{i \in \{1,\dots,n\}} \mathtt{gtw}(\mathbb{G}_i).$$

Proof. By claim (1) of Lemma 4.3, it suffices to show that the guided treewidth is small for every connected component of

$$\mathbb{G} \overset{\text{def}}{=} \mathbb{G}_1 \oplus \cdots \oplus \mathbb{G}_n.$$

Let X be the vertex set of a connected component of $\mathbb{G}$. For a column $i \in \{1,\dots,n\}$ in $\mathbb{G}$, define X_i to be the intersection of X with the i-th column, and define $Y_i \subseteq X_i$ to be the interfaces of the i-th column that are inside X. Iteratively apply claim (3) of Lemma 4.3 to the subgraph induced by X_i, removing from it the at most $2k$ vertices of Y_i. This yields a tree decomposition, call it t_i, of the graph induced by $X_i - Y_i$, such that t_i is captured by a guidance system, call it Λ_i, that uses at most

$$4^k \cdot \mathtt{gtw}(\mathbb{G}_i)$$

colors. Define s_i to be the tree decomposition obtained from t_i by adding the interface vertices Y_i to every bag. Define a tree decomposition t as follows. First, for each index i, create a node y_i with Y_i being the associated bag, and attach all the roots of decomposition s_i as children of the node y_i. Then connect nodes y_i into a path: for each index $i < n$, attach y_{i+1} as a child of y_i. It is not hard to see that t is a tree decomposition of the connected component of $\mathbb{G}$ induced by X.

We now define a guidance system that captures all bags of t. Apply Lemma 4.11 to $\mathbb{G}_1,\dots,\mathbb{G}_n$ yielding a guidance system; call it $\Lambda_\oplus$. Add this guidance system to the guidance systems $\Lambda_1,\dots,\Lambda_n$ defined above, defining a guidance system

$$\Lambda \overset{\text{def}}{=} \Lambda_\oplus \cup \Lambda_1 \cup \cdots \cup \Lambda_n.$$

Since columns can only intersect on interfaces, it follows that the guidance systems $\Lambda_1,\dots,\Lambda_n$ are disjoint. Therefore $\Lambda_1 \cup \cdots \Lambda_n$ can be colored by the maximum number of colors needed for each $\Lambda_1,\dots,\Lambda_n$. To this, we add the number of colors needed for $\Lambda_\oplus$ from Lemma 4.11, thus getting the bound in the statement of the lemma. To finish the proof, we show that Λ captures every bag in the tree decomposition t. Let us first consider the bags of the form Y_i. By Lemma 4.11, we know that

$$Y_i \subseteq \Lambda_\oplus(u) \qquad \text{for every } u \in X_i. \tag{5}$$

In particular, Y_i is captured by any vertex from itself. Consider now a bag B in one of the tree decompositions s_i. By assumption on Λ_i, there is some vertex $u \in X_i$ such that

$$B - Y_i \subseteq \Lambda_i(u).$$

Combining the above with (5), we get that B is captured by u in the guidance system Λ. ☐

We now finish the proof of the Unranked Lemma. Let

$$\mathbb{G} \overset{\text{def}}{=} \mathbb{G}_1 \oplus \cdots \oplus \mathbb{G}_n$$

be a gluing of bi-interface graphs of the same arity k and the same abstraction. Define U_i to be the vertices in $\mathbb{G}_i$ that are both left and right interfaces at the same time and define $\mathbb{H}_i$ to be $\mathbb{G}_i$ with U_i removed. Consider the gluing

$$\mathbb{H} \overset{\text{def}}{=} \mathbb{H}_1 \oplus \cdots \oplus \mathbb{H}_n.$$

By the assumption that the abstractions of all of $\mathbb{G}_1,\dots,\mathbb{G}_n$ are the same, it follows that the image of U_i under the i-th column function is the same set, independent of i, and this set has at most k vertices. Therefore, $\mathbb{H}$ is equal to $\mathbb{G}$ with at most k vertices removed. By claim (2) of Lemma 3.5 we infer that

$$\mathtt{gtw}(\mathbb{G}) \le \mathtt{gtw}(\mathbb{H}) + k. \tag{6}$$

The left and right interfaces are disjoint in each $\mathbb{H}_i$, and these graphs also have the same abstraction, so we can use Lemma 4.12 to get:

$$\mathtt{gtw}(\mathbb{H}) \le 4k(k^2+1) + 4^k \cdot \max_{i \in \{1,\dots,n\}} \mathtt{gtw}(\mathbb{H}_i). \tag{7}$$

Finally, each $\mathbb{H}_i$ is obtained from $\mathbb{G}_i$ by removing at most k vertices, and hence we can apply claim (3) of Lemma 3.5 to infer that

$$\mathtt{gtw}(\mathbb{H}_i) \le 2^k \cdot \mathtt{gtw}(\mathbb{G}_i). \tag{8}$$

By combining (6), (7), and (8), we obtain the desired upper bound on the guided treewidth of $\mathbb{G}$. This finishes the proof of the Unranked Lemma and, as argued before, also the proof of Lemma 4.4 is thus complete.

5. Graphs of bounded treewidth

In this section we prove Lemma 2.6. Our strategy is to show the following result on guidance systems.

Lemma 5.1. *Let G be a graph of treewidth at most k. Then G admits a tree decomposition t° with the following properties:*

(a) every marginal graph has pathwidth at most $2k+1$;
(b) the family of adhesions of t° can be captured by a guidance system colorable with $4k^3+2k$ colors.

From the lemma above, we can deduce Lemma 2.6 as follows. Take interpretation $\mathcal{S}_{4k^3+2k}$ from Lemma 3.4. Lemma 5.1 implies that if the treewidth of the input graph is at most k, then at least one output of $\mathcal{S}_{4k^3+2k}$ satisfies the conditions of Lemma 2.6.

The rest of Section 5 is devoted to proving Lemma 5.1. In Section 5.1, we prove a "local" variant of the lemma, which provides one step of the construction. In Section 5.2, we iterate the local variant to construct a global decomposition, thus proving Lemma 5.1.

5.1 A local variant of Lemma 2.6

In this section, we state and prove Lemma 5.2, which can be seen as a local variant of Lemma 5.1. We begin with some hypergraph terminology, which is used in its statement and proof.

Hypergraphs. A *hypergraph* consists of a set of vertices together with a multiset of nonempty subsets of the vertices, called the *hyperedges*. Note the use of multisets: hyperedges can appear multiple times. If H is a hypergraph, the hypergraph *induced* by a subset of vertices X, denoted by $H[X]$, is the hypergraph where the vertices are X and the hyperedges are intersections of the original hyperedges with X, with the empty ones removed. A *path* in a hypergraph is a sequence

$$(u_1, e_1, u_2, \ldots, u_p, e_p, u_{p+1}),$$

where u_i are pairwise different vertices, e_i are pairwise different edges, and vertices u_i, u_{i+1} are contained in hyperedge e_i for each $i = 1, 2, \ldots, p$. Vertices u_1 and u_{p+1} are the *endpoints*, and the path is said to *go* from u_1 to u_{p+1}. Each vertex u_i and hyperedge e_i is said to be *traversed* by the path. Connected components in a hypergraph are defined by path-connectedness in a natural manner. Tree decompositions of hypergraphs are defined as for graphs, except that every hyperedge must be contained in some bag.

Prefixes and their torsos. Let t be a sane tree decomposition of a graph G. A *prefix* of t is a set of nodes Z in t that is closed under taking ancestors. If Z is a prefix, define ∂Z to be the nodes of t that are not in Z, but their parent is in Z. For a prefix Z, define

$$\mathsf{hypertorso}(t, Z)$$

to be the hypergraph obtained by taking the subgraph of G induced by the union of the bags in Z, and then for every $z \in \partial Z$ a hyperedge for the adhesion of z. When adding hyperedges, we respect multiplicities, i.e. if n nodes in ∂Z have the same adhesion, then this adhesion is used n times in $\mathsf{hypertorso}(t, Z)$.

We are now ready to state the local version of Lemma 5.1.

Lemma 5.2. *Let t be a width k sane decomposition of a connected graph G. Let u, v be vertices in the root bag (note that there is a unique root due to connectedness). Then there exists a nonempty prefix Z of t with the following properties:*

(a) the pathwidth of $\mathsf{hypertorso}(t, Z)$ *is at most $2k + 1$, and*

(b) the vertices u and v can be connected by two paths in $\mathsf{hypertorso}(t, Z)$ *such that if a hyperedge is traversed by both paths, then it is an edge of G.*

The rest of Section 5.1 is devoted to proving the above lemma. The proof uses a Menger style argument, so we begin by defining terminology for hypergraph networks.

Networks. Define a *network* to be a connected hypergraph together with two distinguished different vertices, called the *source* and the *sink*. We extend all notation from hypergraphs to networks in a natural manner. A *cutedge* in a network is a hyperedge which appears in every path from the source to the sink; equivalently, the removal of a cutedge makes the source and the sink fall into different connected components. The following claim is straightforward.

Lemma 5.3. *In every network one can order all of the cutedges into a sequence $(e_1, \ldots, e_p)$ such that every path from the source to the sink visits the cutedges in this order.*

The sequence $(e_1, \ldots, e_p)$ yielded by Lemma 5.3 will be called the *cutedge sequence* of a network. Define a *cutedge component* in a network to be a connected component of the hypergraph obtained by removing the cutedges. The following claim is straightforward.

Lemma 5.4. *Consider a network. Let $(e_1, \ldots, e_p)$ be its cutedge sequence, and define e_0, e_{p+1} be the singletons of the source and the sink, respectively. Then every cutedge component intersects exactly one or exactly two among elements $e_0, e_1, \ldots, e_p, e_{p+1}$. In the former case, the intersected e_i is a cutedge; i.e. i is not equal to 0 or $p + 1$. In the latter case, the two intersected elements must be consecutive in the sequence.*

The above lemma motivates the following terminology for a cutedge component. If it intersects two consecutive elements e_i and e_{i+1} in the cutedge sequence extended by the singletons of the source and the sink, then it is called an (e_i, e_{i+1})*-bridge*. If a cutedge component is not a bridge of any kind, and therefore it intersects exactly one cutedge e_i, then it is called an e_i*-appendix*. This terminology is illustrated in Figure 2.

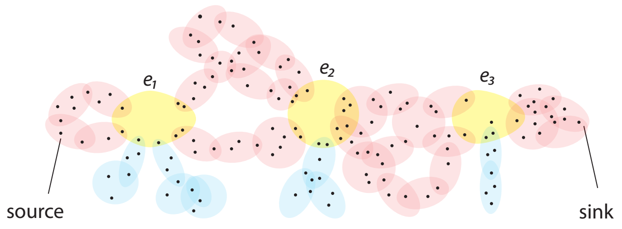


Figure 2: A network. The yellow hyperedges are the cutedges. The red hyperedges are those that contain bridges, the blue ones are those that contain appendices. Note the vertex in e_3 which participates in no blue or red hyperedges, this vertex is a singleton e_3-appendix.

The following lemma is proved by applying Menger's theorem, for every $i \in \{0, 1, \ldots, p\}$, to the union of all (e_i, e_{i+1})-bridges.

Lemma 5.5. *In every network one can find two paths from the source to the sink, such that a hyperedge is traversed by both paths if, and only if it is a cutedge.*

The invariant. To prove Lemma 5.2, we will start with a prefix of the decomposition that contains only the root, and keep on extending the prefix until it satisfies condition (b). While extending the prefix, we will preserve an invariant which implies condition (a). We now describe the invariant. Let t, u, v be as in Lemma 5.2. For a prefix Z of t, consider the network obtained from $\mathsf{hypertorso}(t, Z)$ by choosing u as the source and v as the sink. We will maintain the invariant that this network is k-thin in the sense defined below.

Definition 5.6. Consider a network with hypergraph H and cutedge sequence $(e_1, \ldots, e_p)$; we follow the convention that e_0, e_{p+1} denote the singletons of the source and the sink, respectively. Define V_i to be the union of the vertex sets of all (e_i, e_{i+1})-bridges, for $i = 0, 1, \ldots, p$, and W_i to be the union of the vertex sets of all e_i-appendices, for $i = 1, \ldots, p$. The network is called k-thin if:

(a) for every $i \in \{0, 1, \ldots, p\}$, the induced hypergraph $H[V_i]$ admits a path decomposition of width at most $2k + 1$ where the first bag contains $e_i \cap V_i$ and the last bag contains $e_{i+1} \cap V_i$;

(b) for every $i \in \{1, 2, \ldots, p\}$, the induced hypergraph $H[W_i]$ admits a path decomposition of width at most k where the first bag contains $e_i \cap W_i$.

The following lemma shows that our invariant implies condition (a) in Lemma 5.2. The proof is a simple surgery on decompositions certifying thinness.

Lemma 5.7. *A k-thin network has pathwidth at most $2k+1$.*

Before finishing the proof of Lemma 5.2, we show that thinness is preserved under a certain kind of replacements. Let H, K be hypergraphs such that the intersection of their vertex sets is equal to a hyperedge e of H. Define $H[e \to K]$ to be the following hypergraph. The vertex set is the union of the vertex sets in H, K. The hyperedges are the multiset union of the hyperedges in H, K, with the hyperedge e removed. The following lemma shows that the above defined replacement preserves k-thinness of networks, assuming that e is a cutedge and K is small. The proof is a technical, though conceptually simple analysis of the relationship between cutedges before and after the replacement.

Lemma 5.8. *Consider a k-thin network with hypergraph H. Let K be a connected hypergraph with at most $k+1$ vertices, with no hyperedge larger than k, and such that the intersection of the vertices of H and K is equal to some cutedge e of H. Then the hypergraph $H[e \to K]$, with the same source and sink as in H, is also a k-thin network.*

We are now ready to prove Lemma 5.2.

Proof. For a prefix Z of the tree decomposition t, define the *network of Z* to be the network obtained from $\mathsf{hypertorso}(t, Z)$ by choosing the source to be u and the sink to be v. This is indeed a network: the underlying hypergraph is connected because G itself is connected.

Initially, choose Z to be the prefix that contains only the root node of t. We will maintain the invariant that the network of Z is k-thin. The invariant is clearly satisfied by the initial choice, because the root bag has size at most $k+1$, and adhesions have sizes at most k (due to the saneness of t).

By Lemma 5.7, the invariant implies condition (a). We show below that if Z is a prefix satisfying the invariant, then either it satisfies condition (b), in which case we are done, or one can add a node to the prefix while maintaining the invariant. Since the tree decomposition t has a finite number of nodes, this process has to stop at some moment, thus proving the lemma.

Let then Z be a prefix such that the network of Z is k-thin. Apply Lemma 5.5, yielding two paths from the source to the sink in the network of Z, such that the only hyperedges traversed by both paths are the cutedges of the network of Z. If all these cutedges are original edges of G, then we are done, because Z satisfies condition (b). Otherwise, there is some cutedge e in the network of Z that is not an edge of G. By definition of the network of Z, the cutedge e corresponds to the adhesion of some $z \in \partial Z$. Again by definition, the network of $Z \cup \{z\}$ is obtained from the network of Z by: adding the bag of z to the vertices, removing the hyperedge e, and adding a hyperedge for every adhesion of a child of z. We now verify that this process is a special case of the replacement in Lemma 5.8. Indeed, if we define hypergraph $K = \mathsf{hypertorso}(t_z, \{z\})$, where t_z is the subtree of t rooted at z, then the network of $Z \cup \{z\}$ is obtained from the network of Z by replacing e by K. Observe that K has at most $k+1$ vertices, has no hyperedge larger than k due to the saneness of t, and is connected, again due to the saneness of t. Hence Lemma 5.8 ensures us that the network of $Z \cup \{z\}$ is also k-thin. □

5.2 Decomposition into low pathwidth parts

In this section we finish the proof of Lemma 5.1. We heavily use the notation from Definition 2.2.

Consider a tree decomposition t. For a distinguished set X of nodes in the decomposition t, which is required to include all the roots of t, we define a new tree decomposition $t/_X$ of the same underlying graph as follows. The nodes of $t/_X$ are X. For any node of t that is not in X, assign it to its closest ancestor that belongs to X, i.e., the ancestor from X for which there is no other node from X on the unique path between the node and the ancestor. Then the $t/_X$-bag of a node $x \in X$ is the union of the t-bags of the nodes assigned to x, plus the t-bag of x itself.

Lemma 5.1 will be obtained by taking any sane tree decomposition, and applying the following lemma to every connected component. Condition (a) of Lemma 5.1 will follow from Lemma 5.9(a), while condition (b) of Lemma 5.1 will follow from Lemma 5.10 proved at the end of this section.

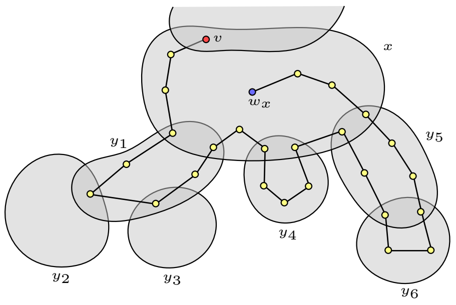


Figure 3: Example path in $\mathcal{P}_x$. This path contributes to the loads of nodes y_1, y_4, y_5, and y_6, but not of x, y_2, and y_3.

Lemma 5.9. *Let t be a width k sane tree decomposition of a connected graph G. Then one can find a set of nodes X in t, which includes the root of t, and families of paths $\{\mathcal{P}_x\}_{x\in X}$, such that every $x \in X$ satisfies:*

(a) The $t/_X$-marginal graph of x has pathwidth at most $2k+1$.
(b) Every element of $\mathcal{P}_x$ is a path in G that satisfies:
   (i) except for its endpoints, the path visits only vertices from the t-component of x;
   (ii) if $y \in X$ is a strict descendant of x, then restricting the path to the t-component of y yields an interval in the path.
(c) All paths in $\mathcal{P}_x$ have the same source, which belongs to the t-margin of x. Conversely, each vertex of the t-adhesion of x is a target of some path from $\mathcal{P}_x$.
(d) The following set of paths has size at most $2k^3$:

$$\mathrm{load}_x \overset{def}{=} \{P \in \mathcal{P}_y : y \in X \text{ is a strict ancestor of } x \text{ and } P \text{ intersects the } t\text{-component of } x\}.$$

Proof. Figure 3 illustrates the notions used in the lemma. We prove the following strengthening of the lemma, with sufficient parameters to be proved by induction.

($\star$) Let x_0 be a node of t, let I be a set of size at most $2k^3$, and let

$$\{(u_i, v_i)\}_{i\in I}$$

be a set of node pairs from the adhesion in x_0, with possible repetitions. One can find a set $X \ni x_0$ of nodes in the subtree of x_0, sets $\{\mathcal{P}_x\}_{x\in X}$ and a set of paths $\{Q_i\}_{i\in I}$ such that

(A) for every $i \in I$, the path Q_i goes from u_i to v_i and satisfies conditions (b:i) and (b:ii) in the lemma with respect to x_0;

(B) every $x \in X$ satisfies conditions (a)-(c) in the lemma and the following variant of (d): the size of load_x is at most

$$2k^3 - |\{i \in I : \text{path } Q_i \text{ intersects the } t\text{-component of } x\}|.$$

The lemma is a special case of ($\star$), by choosing x_0 to be the root of the tree decomposition t, and choosing I to be empty. It remains to prove ($\star$), which is done by induction on the number of nodes in the subtree of x_0.

Let x_0 and $\{(u_i, v_i)\}_{i\in I}$ be as in ($\star$). Choose (u, v) so that

$$I_0 = \{i \in I : (u_i, v_i) = (u, v)\}$$

has maximum size. If I is empty, choose I_0 to be empty and leave (u,v) undefined, this corner case will be considered separately. Since each candidate for (u,v) is in the adhesion of x_0, and adhesions have size bounded by k (due to the saneness of t), it follows that I_0 has size at least $|I|/k^2$.

Define t_0 to be the subtree of t rooted in x_0. Let t_0' be obtained from t_0 by removing from each bag those vertices of the adhesion of x_0 that are neither u nor v. It is easy to see that both t_0 and t_0' are sane tree decompositions. If u, v are defined, apply Lemma 5.2 to t_0' with distinguished vertices u, v, yielding a prefix Z and two paths in the hypergraph $\mathsf{hypertorso}(t_0', Z)$; call these paths P^1 and P^2. Since t_0 and t_0' have the same nodes, and these nodes are a subset of the nodes of t, we can view Z as a connected set of nodes in each of these tree decompositions. In case I is empty and (u,v) is undefined, we choose Z to be the singleton of the root of t.

Let J be the disjoint union of I and the t-adhesion of x_0. For $i \in J$, define a path R_i in $\mathsf{hypertorso}(t_0, Z)$ as follows:

- For $i \in I - I_0$, define R_i to be a path from u_i to v_i, which does not visit the t-adhesion of x_0 except for its endpoints. Such a path exists by the saneness of t.
- Split the set I_0 into two parts, with the first part having half the size (rounded up). For i in the first part, define R_i to be P^1, and for i in the second part, define R_i to be P^2.
- For the remaining i, i.e. those from the t-adhesion of x_0, do the following. Independently of i, choose some vertex w in the t-margin of x_0, which is the same as the t_0-margin of x_0. This is possible because margins are nonempty in a sane decomposition. For each i in the adhesion of x_0, define R_i to be a path from w to i, which does not visit the adhesion of x_0 except for its endpoints. Such a path exists by the saneness of t.

By definition, every hyperedge in $\mathsf{hypertorso}(t_0, Z)$ is an edge of G or corresponds to an adhesion of some $z \in \partial Z$. Let $z \in \partial Z$. Define I^z to be those $i \in J$ for which path R_i uses the hyperedge corresponding to the adhesion of z. The following claim is the key step in this lemma, allowing us to apply the induction assumption.

Claim 2. *For every $z \in \partial Z$, the set I^z has size at most $2k^3$.*

Proof. The claim is trivial in the case when $I = \emptyset$ and (u,v) is undefined, I_z is a subset of J, which in this case has size at most k. In the following, we focus on the general case when $I \neq \emptyset$.

Let e be the hyperedge of $\mathsf{hypertorso}(t_0, Z)$ that corresponds to the adhesion of node z. To bound the size of I_z, we count the number of elements $i \in J$ for which the path R_i traverses the hyperedge e.

First, there are at most k elements of J that originate from the t-adhesion of x_0. The remainder of J, being simply I, can be partitioned into I_0 and $I - I_0$. Among paths R_i of $i \in I_0$, at most half (rounded up) can traverse e. This is because e cannot be traversed simultaneously by P^1 and by P^2, because it is not an edge of the underlying graph G. We conclude that the size of I^z can be bounded as follows:

$$\begin{aligned} |I_z| &\le k + |I - I_0| + \lceil |I_0|/2 \rceil \\ &= k + |I| - \lfloor |I_0|/2 \rfloor \\ &\le k + |I| - \lfloor |I|/2k^2 \rfloor, \end{aligned}$$

where the last inequality follows from $|I_0| \ge |I|/k^2$. We now use the fact that $|I| \le 2k^3$ and that the function $a \mapsto a - \lfloor a/2k^2 \rfloor$ is non-decreasing to conclude that

$$|I_z| \le k + 2k^3 - k = 2k^3.$$

⌟

The informal reason for the claim is that we have used the two paths P^1 and P^2, and at most one of them visits the hyperedge corresponding to the adhesion of z. Since I_0 constitutes a $1/k^2$-fraction of I, and I_0 is split in halves with respect to using P^1 or P^2, we see that around $1/2k^2$-fraction of paths R_i for $i \in I$ avoid the adhesion of z. This is enough to amortize for the new paths R_i for i from the t-adhesion of x_0.

For $i \in I^z$, let u_i^z be the vertex used by the path R_i immediately before the hyperedge corresponding to the adhesion of z, and let v_i^z be the vertex used immediately after. Take the vertex z, which is a proper descendant of x_0, and the family

$$\{(u_i^z, v_i^z)\}_{i \in I^z}.$$

and apply to it the induction assumption of $(\star)$, yielding

$$X^z \qquad \{\mathcal{P}_x^z\}_{x \in X^z} \qquad \{Q_i^z\}_{i \in I^z}.$$

For $i \in J$, define Q_i to be following path in G. Take the path R_i, which might use hyperedges corresponding to adhesions from ∂Z, and for every $z \in \partial Z$ replace the hyperedge corresponding to the adhesion of z, if it is used, by the path Q_i^z. The following claim follows directly from the construction and the induction assumption.

Claim 3. *For every $i \in J$, the path Q_i has the same source and target as R_i and satisfies conditions (b:i) and (b:ii) from the lemma.*

We now complete the proof of $(\star)$. Define

$$X = \{x_0\} \cup \bigcup_{z \in \partial Z} X^z.$$

Note how the above union is actually a partition. Define

$$\mathcal{P}_x = \begin{cases} \{Q_i : i \text{ in the } t\text{-adhesion of } x_0\} & \text{when } x = x_0 \\ \mathcal{P}_x^z & \text{when } x \in X^z. \end{cases}$$

Finally, define $\{Q_i\}_{i \in I}$ to be the restriction of the previously defined family $\{Q_i\}_{i \in J}$ to the smaller indexing set $I \subseteq J$.

Let us check that conditions (A) and (B) in the conclusion of $(\star)$ are satisfied by the above choices. Condition (A) is satisfied thanks to Claim 3. For $x \in X - \{x_0\}$, we check condition (B) in the following claim, which follows from the induction assumption.

Claim 4. *Condition (B) in $(\star)$ holds for each $x \in X - \{x_0\}$.*

Proof. Let z be the node of ∂Z such that x belongs to the subtree rooted at some z. From the induction assumption we infer that load_x, when computed with respect the family $\{\mathcal{P}_x^z\}_{x \in X^z}$, has size at most

$$2k^3 - |\{i \in I^z : \text{path } Q_i^z \text{ intersects the } t\text{-component of } x\}|.$$

Let us denote the set whose cardinality is subtracted above by I_x^z.

When computing load_x with respect to the whole family $\{\mathcal{P}_x\}_{x \in X}$, we need to also take into account the contribution from paths Q_i for i in the adhesion of x_0. More precisely, this contribution is equal to the cardinality of the set I' of all vertices i from the t-adhesion of x_0, for which the path Q_i intersects the t-component of x.

Let us denote by $I_x^{x_0}$ the set of all indices $i \in I$, for which path Q_i intersects the t-component of x. Since J is the disjoint union of I and the adhesion of x_0, and the elements of I^z are in one-to-one correspondence with the elements $i \in J$ for which path Q_i intersects the t-component of z (which in particular happens if the path intersects the t-component of x), we infer that

$$|I_x^z| = |I'| + |I_x^{x_0}|.$$

Hence, load_x computed with respect to the whole family $\{\mathcal{P}_x\}_{x \in X}$ is upper bounded by

$$2k^3 - |I_x^z| + |I'| = 2k^3 - |I_x^{x_0}|,$$

as requested in condition (B) in the conclusion of $(\star)$. ⌟

It remains to check condition (B) for x_0, i.e. to check that x_0 satisfies conditions (a)-(c) and the variant of (d). For (a), we observe that every path decomposition for $\mathsf{hypertorso}(t'_0, Z)$ is also a path decomposition of $t/_X$-margin of x_0, and therefore the latter graph has pathwidth at most $2k+1$, by the conclusions of Lemma 5.2. (In the corner case when I was empty, the $t/_X$-margin of x_0 is the same as the t-margin, and therefore has size at most k.) For (b), we use Claim 3. Condition (c) follows by construction. Finally, condition (b) holds vacuously, because x_0 has no strict ancestors in X. ☐

To finish the proof of Lemma 5.1, we take $t^\circ = t/_X$ for X yielded by Lemma 5.9, and we are left with verifying that the adhesions of t° can be captured by a guidance system that uses few colors. This verification is given in the following lemma.

Lemma 5.10. *Let t and X be as in Lemma 5.9. The adhesions of the tree decomposition $t/_X$ can be captured by a guidance system that is colorable with $4k^3 + 2k$ colors.*

Proof. In the proof we will use the notation α, β, and σ explained in the beginning of Appendix C. These operators respectively denote the component, the bag, and the adhesion at some node, and they will always be applied to the decomposition t°.

Let M be the set of all pairs (x, v) such that x is a node of t° and v is in the adhesion of x. For each node x of t°, let w_x be the common start of all the paths from $\mathcal{P}_x$; recall that w_x belongs to the t°-margin of x. For each $(x, v) \in M$, let $P_{x,v}$ be a path from $\mathcal{P}_x$ that goes from w_x to v.

We will say that two pairs $(x_1, v_1), (x_2, v_2) \in M$ are *in conflict* if the following conditions are satisfied:

$$v_1 \neq v_2 \qquad \text{and} \qquad P_{x_1,v_1} \text{ and } P_{x_2,v_2} \text{ intersect.}$$

Intuitively, the conflict relation encodes that paths P_{x_1,v_1} and P_{x_2,v_2} have to be realized using different colors in a guidance system that captures the adhesions of t°. Note that two pairs with the same target vertex v are never in conflict, even if they share some other vertices. This detail will be crucial for the proof of the claim.

Let us define *conflict graph* D with vertex set M where the edge set encodes the relation of being in conflict. We now prove that D admits a proper coloring using at most $4k^3 + 2k$ colors. For this, we show that pairs from M admit an ordering $\prec$ such that every pair (x, v) has only at most $4k^3 + 2k - 1$ conflicting pairs that are smaller in $\prec$. Then, a greedy coloring procedure, which scans M in the $\prec$ order and assigns an arbitrary free color to each element of M, yields a coloring of D with at most $4k^3 + 2k$ colors[1].

As $\prec$ we take an arbitrary ordering of M that respects the top-down order in t°. That is, it has the following property: whenever x is a descendant of x', then $(x', v') \prec (x, v)$ for each $v' \in \sigma(x')$ and $v \in \sigma(x)$. Fix any pair $(x, v) \in M$; we now examine how many pairs $(x', v') \prec (x, v)$ can be in conflict with (x, v).

First, there are at most $k - 1$ other pairs of M where $x' = x$. Keeping these pairs in mind, from now on we consider only pairs where $x' \neq x$. Since (x', v') is in conflict with (x, v), we have that $v \neq v'$ and $P_{x,v}$ and $P_{x',v'}$ intersect.

Second, from condition (b:i) of Lemma 5.9 we infer that $P_{x,v}$ is entirely contained in $\alpha(x)$ apart from its endpoint v. Similarly, $P_{x',v'}$ is entirely contained in $\alpha(x')$ apart from its endpoint v'. Since $v \neq v'$, it must hold that x and x' are in ancestor-descendant relation, because otherwise $P_{x,v}$ and $P_{x',v'}$ could not intersect. Since $(x', v') \prec (x, v)$, we infer that x' is an ancestor of x.

Suppose now that $P_{x',v'}$ intersects $\alpha(x)$. Since x' is an ancestor of x, it follows that $P_{x',v'}$ belongs to load_x. Since the size of load_x is at most $2k^3$, due to condition (d) of Lemma 5.9, the number of such pairs (x', v') is upper bounded by $2k^3$. Keeping these pairs in mind, from now on we investigate only paths $P_{x',v'}$ that are disjoint with $\alpha(x)$. Since $P_{x,v}$ is entirely contained in $\alpha(x)$ apart from its endpoint v, the only remaining way (x, v) and (x', v') can be in conflict is when $P_{x',v'}$ traverses v but $v \neq v'$.

Let y be the topmost node in t° such that $v \in \beta(y)$. Clearly y is also a (strict) ancestor of x, because $v \in \sigma(x)$. Thus, y and x' are also in ancestor-descendant relation. Suppose for a moment that x' is a strict descendant of y. As $v \in \beta(x) \cap \beta(y)$, we have that $v \in \sigma(x')$. However, this is a contradiction with condition (b:i) of Lemma 5.9 for the path $P_{x',v'}$, as this path traverses v but avoids $\sigma(x')$ apart from its endpoint v', which is different from v.

This proves that either $x' = y$ or x' is an ancestor of y. The number of pairs (x', v') with $x' = y$ is at most k. Keeping these pairs in mind, from now on we concentrate on the remaining case when $x' \neq y$. Since y was chosen to be the top-most bag containing v, we have that $v \in \alpha(y)$. As v is traversed by $P_{x',v'}$, we infer that this path belongs to load_y. However, by condition (d) of Lemma 5.9 the size of load_y is at most $2k^3$, so the number of such pairs (x', v') is at most $2k^3$.

Thus, in total we have found at most $(k-1) + 2k^3 + k + 2k^3 = 4k^3 + 2k - 1$ pairs (x', v') that are smaller than (x, v) in the ordering $\prec$ and are in conflict with (x, v). As discussed before, this implies that D admits a proper coloring ϕ with $4k^3 + 2k$ colors, so that any conflicting pairs receive different colors.

We now define a guidance system Λ over G that captures the family of adhesions of t°. For each color c, let G_c be the union of paths $P_{x,v}$ for which $\phi(x, v) = c$. As ϕ is a proper coloring of D, pairs (x, v) mapped to the same color c are pairwise not in conflict. Hence, all paths $P_{x,v}$ that are in the same connected component of G_c actually need to have exactly the same target vertex v, as otherwise some two of them would be in conflict. For each connected component C of G_c, let v_C be this common target vertex. Construct Λ as follows: for each color c and each component C of G_c, arbitrarily select any its arbitrary spanning tree, orient it toward v_C, and add it to Λ. Assigning color c to each tree originating in G_c yields a coloring of Λ with $4k^3 + 2k$ colors.

It is now easy to verify that Λ captures all the adhesions of t°. Fix some node x of t°; we claim that $\sigma(x) \subseteq \Lambda(w_x)$. Take any $v \in \sigma(x)$, and let $c = \phi(x, v)$. Then path $P_{x,v}$ has been included in G_c, and in particular from the construction it follows that $v \in \Lambda(w_x)$. ☐

6. Conclusions

There are two concrete open questions that arise from our work. Firstly, we believe that our main result, Theorem 2.4, can be strengthened to the following statement: the transduction yields a tree decomposition of the optimum width k, instead of approximate $f(k)$. This would immediately lead to a stronger statement of Theorem 2.10 (Courcelle's conjecture): the assumption of k-recognizability alone, instead of k'-recognizability for all k', would imply that a property of graphs of treewidth k is definable in counting MSO.

Our idea for proving the stronger statement is to take a closer look on the work of Bodlaender and Kloks [2], who gave a dynamic programming algorithm that, given a graph together with a tree decomposition of width $k' \geq k$, constructs, if possible, a tree decomposition of width k. A preliminary inspection of the proof gives hope that the algorithm can be translated into a deterministic MSO transduction that, given a tree decomposition of width k', outputs a tree decomposition of width k. Such a transduction could be then combined with Theorem 2.4 to yield the strengthening.

Secondly, in Section 4.4 we have proved that the guided treewidth of a graph is bounded in terms of its pathwidth. It is natural to ask whether the same holds also for treewidth: does there exist a function

[1] This argument is equivalent to saying that D is $(4k^3 + 2k - 1)$-degenerate, and hence $(4k^3 + 2k)$-colorable.

f such that

$$\mathtt{gtw}(G) \leq f(\mathtt{tw}(G)) \qquad \text{for every graph } G?$$

Our current approach falls short of proving this conjecture. Intuitively, the main problem is that the combinatorial results of Lemmas 4.4 and 5.1 are combined at the (less restrictive) level of MSO transductions in the proof of Theorem 2.4, and we do not know how to combine them at the level of guidance systems. If true, the conjecture would give a somewhat conceptually easier way to prove our main result.

References

[1] H. L. Bodlaender, P. Heggernes, and J. A. Telle. Recognizability equals definability for graphs of bounded treewidth and bounded chordality. *Electronic Notes in Discrete Mathematics*, 49:559–568, 2015.

[2] H. L. Bodlaender and T. Kloks. Efficient and constructive algorithms for the pathwidth and treewidth of graphs. *J. Algorithms*, 21(2):358–402, 1996.

[3] M. Bojańczyk and S. Lasota. An extension of data automata that captures XPath. *Logical Methods in Computer Science*, 8(1), 2012.

[4] B. Courcelle. The Monadic Second-Order logic of graphs. I. Recognizable sets of finite graphs. *Inf. Comput.*, 85(1):12–75, 1990.

[5] B. Courcelle. The Monadic Second-Order logic of graphs V: On closing the gap between definability and recognizability. *Theor. Comput. Sci.*, 80(2):153–202, 1991.

[6] B. Courcelle and J. Engelfriet. *Graph Structure and Monadic Second-Order Logic — A Language-Theoretic Approach*, volume 138 of *Encyclopedia of mathematics and its applications*. Cambridge University Press, 2012.

[7] B. Courcelle and J. Engelfriet. *Graph Structure and Monadic Second-Order Logic - A Language-Theoretic Approach*, volume 138 of *Encyclopedia of mathematics and its applications*. Cambridge University Press, 2012.

[8] R. G. Downey and M. R. Fellows. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer, 2013.

[9] M. Grohe and D. Marx. Structure theorem and isomorphism test for graphs with excluded topological subgraphs. *SIAM J. Comput.*, 44(1):114–159, 2015.

[10] L. Jaffke and H. L. Bodlaender. Definability equals recognizability for k-outerplanar graphs. In *IPEC'15*, volume 43 of *LIPIcs*, pages 175–186, 2015.

[11] V. Kabanets. Recognizability equals definability for partial k-paths. In *ICALP'97*, volume 1256 of *LNCS*, pages 805–815. Springer, 1997.

[12] D. Kaller. Definability equals recognizability of partial 3-trees and k-connected partial k-trees. *Algorithmica*, 27(3):348–381, 2000.

[13] M. Kufleitner. The height of factorization forests. In *MFCS 2008*, volume 5162 of *LNCS*, pages 443–454. Springer, 2008.

[14] D. Lapoire. Recognizability equals Monadic Second-Order definability for sets of graphs of bounded tree-width. In *STACS'98*, volume 1373 of *LNCS*, pages 618–628. Springer, 1998.

[15] I. Simon. Factorization forests of finite height. *Theor. Comput. Sci.*, 72(1):65–94, 1990.

A. Proof of the main result (Theorem 2.4)

In this section, we combine Lemmas 2.5 and 2.6 to get the main technical Theorem 2.4. The proof strategy is to take a graph, apply Lemma 2.6 to get a tree decomposition with marginal graphs of bounded pathwidth, then apply Lemma 2.5 to get a tree decomposition of each marginal graph, and then to combine these tree decompositions into a single tree decomposition.

In the proof we will need the following simple technical statement about sane tree decompositions. Its proof can be found in Appendix C.2 (see Claim 10 therein).

Claim 5. *Suppose t is a sane tree decomposition of a graph G, y is a node of t, and x is the parent of y. Then there exists a vertex of G that is simultaneously in the adhesion of y and in the margin of its parent x.*

Nested decompositions. A *nested tree decomposition* consists of a sane tree decomposition t plus a sane tree decomposition t_x of the t-marginal graph of x for every node x of t. We use the name *main decomposition* for t, and the name *marginal decomposition* for any of the t_x. Note that by the saneness of t, the marginal graph of each node of t is nonempty and connected, and therefore each marginal decomposition is a tree and not a forest.

We model a nested tree decomposition as a logical structure as follows. The structure is the disjoint union of the structures encoding the main decomposition and all the marginal decompositions. Recall that each of these structures contains both the decomposition (as nodes connected by the parent relation) and the underlying graph (using the standard encoding for graphs), which are linked together by the bag relation encoding the contents of the bags. Thus, the structure encoding a nested tree decomposition contains a copy of the whole graph, linked to the main decomposition t, as well as, for each node x of t, a copy of the t-marginal graph of x, linked to its marginal decomposition t_x. These will be called the *underlying graphs* of t and t_x, respectively.

In the encoding of a nested tree decomposition, we assume that there is an additional binary predicate same that selects pairs (v, v') such that v and v' are copies of the same vertex: one in the underlying graph of t, and the second in the underlying graph of t_x, where the vertex belongs to the t-margin of x. We also assume that there is a binary predicate $\mathsf{marginal}(y, x)$ that selects pairs (y, x) such that y is a node of the marginal decomposition t_x. Note that marginal is a function on the union of the node sets of the marginal decompositions.

Define the *width* of a nested tree decomposition to be equal to

$$\text{(maximum width among marginal decompositions } t_x) \\ + \\ \text{(maximum size of an adhesion of } t).$$

The following lemma shows that a deterministic MSO transduction can flatten a nested tree decomposition.

Lemma A.1. *There exists a deterministic* MSO *transduction, whose domain is the set of nested tree decompositions, which transforms a nested tree decomposition into a tree decomposition with the same underlying graph. If the input has width at most k, then the output also has width at most k.*

Note that in the above lemma, the transduction does not fix the width k, i.e. there is a single transduction that works for all k.

Proof. Consider a nested tree decomposition of a graph G, consisting of the main decomposition t and marginal decompositions t_x, for nodes x of t. Below we define its *flattening*, which is going to be the output, to be the following tree decomposition s of G.

- *Nodes.* Nodes of the flattening s are pairs (x, y) such that x is a node of the main decomposition and y is a node in t_x.
- *Bags.* Let (x, y) be a node of the flattening s. The s-bag of (x, y) is the union of the t-adhesion of x and the t_x-bag of y.
- *Parents.* Let (x, y) be a node in the flattening s. Let x', y' be the parents of x, y in their respective tree decompositions, both of which might be undefined. If y' is defined, then the parent of (x, y) is (x, y'). If x' is undefined, then (x, y) is a root of the flattening s. Otherwise, if y' is undefined but x' is defined, we examine set A defined as the intersection of the t-adhesion of x with the t-margin of x'. By Claim 5, A is nonempty, and

moreover it is a clique in the t-marginal graph of x'. Since $t_{x'}$ is a tree decomposition of this t-marginal graph, the set of nodes of $t_{x'}$ whose bags contain A is nonempty and connected in $t_{x'}$. Let z be the lowest among these nodes; in other words, z is the lowest common ancestor of all the nodes whose bags contain A, which is also a node with this property. Then the parent of (x, y) is defined to be (x', z).

It is not difficult to check that the flattening described above is indeed a tree decomposition. Note that the definition of the width of a nested tree decomposition is chosen in such a way that, provided the input nested tree decomposition has width at most k, then the width of the output tree decomposition is also at most k.

We now shortly argue that the construction above can be implemented by means of a deterministic MSO transduction. A node (x, y) of the output t is interpreted in the node y of t_x on the input; note that the node x can be uniquely recovered from y using predicate $\mathsf{marginal}$. Then it is straightforward to implement all the definitions above using interpretation. $\square$

Theorem 2.4 is an immediate corollary of Lemma A.1 and the following lemma, which is where we use Lemmas 2.5 and 2.6.

Lemma A.2. *For every $k \in \mathbb{N}$ there is some $k' \in \mathbb{N}$ and an* MSO *transduction, whose domain is graphs, such that:*

- *every output is a nested tree decomposition of the input;*
- *if the input has treewidth at most k, then some output has width at most k'.*

Proof. Suppose we are given some graph G together with a coloring ϕ of its vertices using some finite set of colors. We will say that a tree decomposition t of G is *colorful under ϕ* if every adhesion of t has all its vertices colored differently in ϕ. For some $\ell \in \mathbb{N}$, by an *ℓ-colored* tree decomposition we mean a tree decomposition given as a logical structure (i.e. with the underlying graph encoded), where the underlying graph G is additionally supplied with some vertex coloring which uses ℓ colors and under which t is colorful. The coloring is given as ℓ unary predicates that partition the vertex set. Obviously, an ℓ-colored tree decomposition has no adhesion larger than ℓ.

Let t be a tree decomposition, given as a logical structure (i.e. with the underlying graph G encoded). Define the logical structure $\mathfrak{T}$ to be the disjoint union of t and all of its marginal graphs, together with a binary predicate same that selects all pairs (v, v') such that v and v' are copies of the same vertex: v belongs to the graph G (present in the logical structure encoding t), and v' is its copy in the unique marginal graph that contains v.

Claim 6. *Let $\ell \in \mathbb{N}$. There is an* MSO *transduction that implements the transformation $t \mapsto \mathfrak{T}$ for ℓ-colored tree decompositions t.*

Proof. We first create $2 + \binom{\ell}{2}$ copies of t (called further layers), and then shape these layers into the output structure $\mathfrak{T}$ using interpretation in a manner described as follows. The first layer is simply the copy of t in the output. The second layer is used to define the vertices of the marginal graphs, and the edges in the marginal graphs that correspond to the edges in G. The MSO-definability of the construction that carves out the marginal graphs from a copy of t is straightforward. Here, we can also define predicate same.

Let us index the remaining $\binom{\ell}{2}$ layers with 2-element subsets of $\{1, \ldots, \ell\}$; these layers will be used to produce the additional edges that are added in the construction of the marginal graphs. More precisely, for every node x in t and every its child y, we need to add the at most $\binom{\ell}{2}$ edges between vertices of the intersection of the adhesion of y and the margin of x. Let us call this intersection $A_{x,y}$. For the copy of y from the layer indexed with $\{i, j\}$, we check whether $A_{x,y}$ contains both a vertex u colored with color i, and a vertex v colored with color j, and moreover vertices u and v are not adjacent in G. If this is not the case, we remove the copy from the structure. Otherwise, we turn the copy into an edge between u and v in the marginal graph of x. Note that the copies of u and v in this marginal graph can be retrieved using predicate same, which we have already defined. Observe that, in this manner, an edge uv could have been added multiple times to the structure, in case it is contained in multiple adhesions of t. However, we can remove the multiplicities by guessing one duplicate of each edge to preserve, and removing all the others. $\square$

We now complete the proof of the lemma. Suppose that we are given on input a graph G. For k, apply the MSO transduction from Lemma 2.6, yielding a tree decomposition t. If the input graph G had treewidth at most k, then there is at least one output t for which all marginal graphs in t have pathwidth at most $2k + 1$, and the family of adhesions of t can be captured by a $(4k^3 + 2k)$-colorable guidance system.

Claim 7. *There exists a coloring of the vertex set of G using at most $4k^3 + 4k + 2$ colors such that every adhesion of t has all vertices colored differently.*

Proof. Since the adhesions of t can be captured by a guidance system that can be colored using $4k^3 + 2k$ colors, this means that all adhesions of t have at most this size. This, together with the bound on the pathwidth of each marginal graph, shows that G admits a nested tree decomposition of width at most $4k^3 + 4k + 1$. This nested tree decomposition remains a valid nested tree decomposition of the same width even if we consider a graph G' obtained from G by turning every adhesion of t into a clique. By applying the combinatorial construction of Lemma A.1 to this nested tree decomposition with the underlying graph G', we infer G' admits a tree decomposition of width at most $4k^3 + 4k + 1$, i.e.,

$$\mathtt{tw}(G') \leq 4k^3 + 4k + 1.$$

The result follows from a known fact that any graph of treewidth ℓ admits a proper coloring using $(\ell + 1)$ colors, and the property that in G' all adhesions of t are cliques. $\lrcorner$

Setting $\ell = 4k^3 + 4k + 2$, we guess a coloring of the vertex set of G with ℓ colors that has the property stated in Claim 7. Then we can enrich the tree decomposition t with the guessed coloring, thus obtaining an ℓ-colored tree decomposition. To this ℓ-colored tree decomposition we can apply the MSO transduction from Claim 6, which constructs the marginal graphs.

For now leave the tree decomposition t alone, and to the union of the marginal graphs apply the MSO transduction from Lemma 2.5. Since the pathwidth of the union of these graphs is bounded by $2k + 1$, this yields a tree decomposition of the marginal graphs with width bounded by a function of k.

So far, the result is two tree decompositions: the tree decomposition t and a tree decomposition s of the union of the marginal graphs. Tree decomposition s can be actually assumed to be sane; see the proof of Lemma 2.5 in Appendix D. Since a sane tree decomposition of a graph contains one tree per each connected component, we see that in s there is one tree for each marginal graph, because all marginal graphs are connected. This tree is a sane tree decomposition of this marginal graph. Hence, s together with the marginal graphs actually *is* a disjoint union of logical structures representing sane tree decompositions of the marginal graphs.

To fully define a nested tree decomposition, we are left with defining the predicate $\mathsf{marginal}(y, x)$ that associates each node y of the tree decomposition t_x of the marginal graph of x, with the node x. This is, however, straightforward: if u is any vertex of the margin of y in t_x (which exists by the saneness of t_x), then x is equal to the unique node of t whose margin contains u. Note that here we use the already defined predicate same to relate the vertices of the whole graph G with their copies in the marginal graphs.

Since the adhesions of t have sizes at most $4k^3 + 2k$, because they can be captured by a guidance system using this many colors, it follows that the width of the obtained nested tree decompositions can be bounded by a function of k. $\square$

B. Omitted proofs from Section 2

Lemma B.1 (Backwards Translation Theorem). *Suppose $\mathcal{I}$ is an* MSO *transduction with input vocabulary Σ and output Γ, and let ψ be an* MSO *sentence over vocabulary Γ. Then there exists an* MSO *sentence φ over vocabulary Σ such that φ is true in exactly those structure $\mathfrak{A}$ over Σ, for which $\mathcal{I}(\mathfrak{A})$ contains at least one structure satisfying ψ.*

Proof sketch. It suffices to verify the statement for the three basic types of operations.

For copying, we replace every quantification in ψ of some subset X with quantification of n sets $X_1, X_2, \ldots, X_k$, which represent the intersections of X with consecutive layers of the copied universe. Similarly, individual quantification in ψ of an element e can be replaced by quantification of this element and the layer it belongs to, where the latter is implemented as a constant-size conjunction or disjunction. It is then straightforward to adapt atomic formulas in ψ.

For coloring, formula φ simply existentially quantifies predicates $X_1, \ldots, X_k$, and then uses ψ.

For interpretation, formula φ is the conjunction of formulas φ_{dom} and ψ', where ψ' is derived from ψ as follows: every predicate R is replaced with the corresponding formula φ_R, and quantifications are relativized to the restricted universe using formula φ_{univ}. $\square$

Proof of Lemma 2.11. The following proof is an adaptation of the proof of Theorem 4.8, claim (2), from [5]. Our strategy is to reduce the case of graphs supplied with tree decompositions to the case of forests labelled by some finite alphabet. Then we use a result of Courcelle [4], which proves an analogous claim for labelled forests.

Let t be the tree decomposition of the graph G given on the input. Recall that the input structure encodes both the graph G and its tree decomposition t, and the width of t is bounded by k.

Without loss of generality, we can assume that no neighboring nodes in t have equal bags, since otherwise the edge between these two nodes could be contracted. Indeed, it is straightforward to see that such a contraction can be simultaneously applied to all such pairs of neighboring nodes by means of an MSO transduction; more precisely, the contracted decomposition can be interpreted in the input one. Hence, from now on we assume that t has the aforementioned property, and in particular all the adhesions in t (i.e. intersections of neighboring bags) have sizes at most k.

We now apply an MSO transduction $\mathcal{C}$ that does the following: guess a coloring of the vertex set of G using $k+1$ colors such that every two vertices that appear together in some bag of t receive different colors. It is a known fact about colorings of graphs of bounded treewidth that, provided the width of t is at most k, such a coloring exists. Thus, from now on we can assume that the input graph is supplied with a $(k+1)$-coloring having the property stated above.

Define Σ to be the set of pairs (H, X) with the following property: H is a graph whose vertex set is a subset of $\{1, \ldots, k+1\}$, and X is a subset of the vertices of H that has size at most k. Clearly, Σ is a finite set of size that is a function of k only. We now apply an MSO transduction $\mathcal{D}$ that labels each node of the tree decomposition t with the subgraph induced by its bag, together with the adhesion of the node. Formally, to each node x of t we associate the label $(H, X) \in \Sigma$, where H is the subgraph induced by the bag of x in G, with vertices renamed according to their colors in the guessed $(k+1)$-coloring, and X is the image of the adhesion of x under the same renaming. As usual, labels are encoded by adding a fresh unary predicates, one per label in Σ; such a predicate is true for exactly those nodes that are labelled with the element. It is clear that such labelling can be computed by an MSO transduction.

The intuition now is that the labelling of the decomposition uniquely encodes the underlying graph G, and the encoding can be reversed provided there is no "mismatch" between the encodings of two neighboring bags. We now formalize this intuition.

A tree s with nodes labelled by Σ is called *well-formed* if the following assertion holds for every node y with parent x: if x and y are labelled with (H_x, X_x) and (H_y, X_y), respectively, then X_y is a subset of the vertex set of H_x and the graphs induced by X_y in H_x and H_y are equal. It is clear that the output of the interpretation $\mathcal{D}$ above is always well-formed.

A well-formed tree s labelled with Σ may be uniquely associated with a $(k+1)$-interface graph as follows. If r is the root of t and it is labelled by (H, X), then the $(k+1)$-interface graph associated with s will use X for the names of the interfaces. This interface graph is recursively defined as follows:

- Define $\mathbb{H}_r$ to be the interface graph obtained from H by taking the identity on the vertex set of H as the interface mapping.
- Compute the $(k+1)$-interface graphs $\mathbb{H}_1, \mathbb{H}_2, \ldots, \mathbb{H}_p$ associated with the subtrees rooted at the children of the root r.
- Take the gluing of all the interface graphs defined above.
- Forget (i.e. remove from the interface mapping) all the interfaces whose names do not belong to X.

The $(k+1)$-interface graph associated with a well-formed tree s will be called the *decoding* of s. It is clear the input graph G is isomorphic to the graph obtained by considering any labelled forest output by the interpretation $\mathcal{C} \circ \mathcal{D}$, and taking the disjoint union of the decodings of the trees of this forest (each of these decodings has no interfaces).

We can define recognizability for properties of forests labelled with some finite alphabet as follows. First, by a *context* we mean a forest over the alphabet with one specified node called the *hole*, which is a leaf and has no label. If c is a context and s is a tree, then we can obtain a forest by replacing the hole in c with tree s in a natural manner: we take the disjoint union of c and s, remove the hole, and attach the root of s as a child of the parent of the hole, in case it exists. This operation, denoted by $c \circ t$, naturally leads to the definition of recognizability for properties of forests. Namely, for some property Π of labelled forests, we will say that two trees s_1, s_2 are Π-*equivalent* if, and only if for any context c we have that

$$c \circ s_1 \text{ satisfies } \Pi \qquad \text{iff} \qquad c \circ s_2 \text{ satisfies } \Pi.$$

We shall say that Π is *recognizable* if the equivalence relation defined above has finitely many equivalence classes.

Now, given the property Π of graphs, we define property Π' of rooted forests labelled with Σ as follows. A forest t satisfies Π' if, and only if:

- Each tree of s is well-formed, and the label of its root has the empty set on the second coordinate; and
- The graph obtained by taking the disjoint union of the decodings of the trees of s satisfies Π.

We now deduce that the recognizability of Π implies the recognizability of Π'.

Claim 8. *If Π is k-recognizable as a property of graphs, then Π' is recognizable as a property of rooted forests labelled with Σ.*

Proof. Observe that all the malformed trees over Σ are Π'-equivalent. Moreover, suppose s_1, s_2 are two well-formed trees whose decodings use the same set of interface names. Recall that this set of names is some subset I of $\{1, \dots, k+1\}$ of size at most k. Suppose further that, after renaming I to a subset of $\{1, \dots, k\}$ in both graphs in the same manner, the decodings turn out to be Π-equivalent as k-interface graphs. Then it follows directly from the definitions that s_1 and s_2 are Π'-equivalent. As the number of different sets of interface names over $\{1, \dots, k+1\}$ is finite, this implies that if Π-equivalence has finitely many classes of abstraction, then so does Π'-equivalence. ⌟

Now, we can use the result of Courcelle [4, Theorem 5.3] which says that recognizability of properties of forests is equivalent to their definability in counting MSO. More precisely, from [4] the following claim is immediate.

Claim 9 ([4]). *If a property* Π *of rooted forests over some finite alphabet is recognizable, then there is a formula* φ_Π *of counting* MSO *that is true in exactly those forests that satisfy* Π.

We remark that Courcelle [4] works over edge-labelled trees instead of node-labelled forests, but we can easily reduce our setting to his as follows: add an artificial root with all the previous roots attached as children, and then push all labels from nodes to the edges connecting them with their parents. Also, Courcelle [4] defines recognizability by the existence of a homomorphism into a finite algebra, which is easily equivalent to our definition via a Myhill-Nerode equivalence relation.

We conclude the proof by applying Claim 9 to property Π', which is recognizable due to Claim 8, and then using the Backwards Translation Theorem to translate the obtained formula $\varphi_{\Pi'}$ back through the composition of transductions $\mathcal{C}$ and $\mathcal{L}$. □

C. Omitted proofs from Section 3

In the proofs from this section, as well as some proofs in other parts of the appendix, we will use the following notation borrowed from Grohe and Marx [9]. Suppose t is a tree decomposition of a graph G, and x is some its node. Then we use the following notation:

- $\beta(x)$ is the bag of x;
- $\sigma(x)$ is the adhesion of x;
- $\gamma(x)$ is the cone of x;
- $\alpha(x)$ is the component of x;
- $\mu(x)$ is the margin of x.

It will be always clear from the context, to which tree decomposition t these operators are referring.

C.1 Shorter proofs: Lemmas 2.8 and 3.5

Proof of Lemma 2.8. In the following, for a tree decomposition s, by $V(s)$ and $E(s)$ we denote the node set and the edge set of s, respectively. We define the *potential* of a tree decomposition s as follows:

$$\Phi(s) = \sum_{x \in V(s)} |\alpha(x)|^2.$$

Let s be a tree decomposition of the input graph G that is chosen as follows:

(i) Each bag of s is a subset of some bag of t.

(ii) Among decompositions satisfying (i), s minimizes the potential $\Phi(s)$.

(iii) Among decompositions satisfying (i) and (ii), s has the minimum total size measured as

$$|V(s)| + |E(s)| + \sum_{x \in V(s)} |\beta(x)|.$$

Decomposition s is well defined due to t satisfying property (i). We now verify that s is sane.

First, suppose that condition (a) is not satisfied for some node x. If x has some parent y, then contracting edge xy in s and placing $\beta(y)$ at the node resulting from the contraction yields a tree decomposition s' of G. This tree decomposition has either strictly lower potential $\Phi(\cdot)$ (in case $\alpha(x) \neq \emptyset$), or the same potential $\Phi(\cdot)$ but strictly smaller size. As the bags of s' are a subfamily of the bags of s, this contradicts the choice of s. Also, if x is the root in s and $\mu(x) = \emptyset$, then in fact $\beta(x) = \emptyset$. Then we can remove x from the decomposition and make its children roots of respective subtrees. This yields a tree decomposition of G with the same bags, not larger potential $\Phi(\cdot)$, and strictly smaller size.

For condition (b), we only verify the connectivity of $G[\alpha(x)]$, because the connectivity of $G[\gamma(x)]$ follows from the connectivity of $G[\alpha(x)]$ and condition (c). Suppose then that $G[\alpha(x)]$ is not connected for some node x. This means that $\alpha(x)$ can be partitioned into nonempty sets A and B with no edge between them in G. Obtain a tree decomposition s' of G as follows. Let p be the subtree of s rooted at x. Replace p in s by two copies p_A and p_B of p, where in p_A every bag is replaced with its intersection with $A \cup \sigma(x)$, and likewise for p_B. Since there is no edge in G between A and B, it is easy to verify that s' obtained in this manner is indeed a tree decomposition of G. Moreover, if y is a node of p, then the components of the copies of y in p_A and in p_B form a partition of the component of y in p. By the convexity of function $a \mapsto a^2$ we infer that $\Phi(s')$ is not larger than $\Phi(s)$. Moreover, the equality can hold only if for every node y of p, the component of y is being partitioned trivially: it is either entirely contained in A or entirely contained in B. However, this is not the case for $y = x$, as both A and B are non-empty. This implies that in fact $\Phi(s') < \Phi(s)$. As every bag of s' is contained in a bag of s, which in turn is contained in a bag of t, this is a contradiction with the choice of s.

Finally, suppose that condition (c) is not satisfied for some node x and some vertex u of $\sigma(x)$. Since $\sigma(x) \neq \emptyset$, x has some parent y and $u \in \beta(y)$. Obtain decomposition s' by removing u from every bag of every descendant of x, including x itself. Since u is still contained in $\beta(y)$, which in turn contains whole $\sigma(x)$, and u has no neighbors in $\alpha(x)$, it follows that s' is still a tree decomposition of G. Moreover, it is easy to see that during the construction of s', the component of each node could have only shrunk; hence, $\Phi(s') \leq \Phi(s)$. Also s' is obtained from s by removing some vertices from some bags, so in particular the total size of s' is strictly smaller than that of s. This is a contradiction with the choice of s. □

Proof of Lemma 3.5. Ad (1). Let Λ be a guidance system over G that captures $\mathcal{X} - u$. Consider the connected component of u inside G. Choose a spanning tree of this component, and direct its edges toward u so that it becomes the root; call the resulting tree t. We claim that the guidance system $\Lambda \cup \{t\}$ captures $\mathcal{X}$, thus proving (1), because at most one more color is needed to color the added tree t. Consider a set $X \in \mathcal{X}$. If X is $\{u\}$, then it is captured by u using the added tree t. Otherwise, $X - u$ is a nonempty set that is contained in $\Lambda(v)$ for some v. If X does not contain u, then we are done. Otherwise, X is contained in the connected component of u, and therefore the added tree t takes care of u.

Ad (2). Let Λ be a guidance system in G that captures $\mathcal{X}$. Let

$$t_1, \dots, t_n$$

be the trees from Λ that contain vertex u. By assumption on Λ being k-colorable, there are at most k such trees. For each $i \in \{1, \dots, n\}$ choose a spanning tree s_i of the component in $G - u$ that contains the root of t_i, and direct all edges toward this root. Whenever this root is equal to u, by slightly abusing the notation take s_i to be an empty tree; we will note use them in the final guidance system. Define a new guidance system

$$\Lambda' = \Lambda - \{t_1, \dots, t_n\} \cup \{s_1, \dots, s_n\}.$$

This is a guidance system over $G - u$, and at most $n \leq k$ additional colors are needed to color trees $s_1, \dots, s_n$. In order to verify that Λ' captures $\mathcal{X}$, take any set X of $\mathcal{X}$. Then X is contained in some component C of $G - u$, and is captured by some vertex v in Λ.

Suppose first that v belongs to C. We claim that then X is also captured by v in Λ'. Indeed, for every element x of X, either the tree that certifies that $x \in \Lambda(v)$ survives in Λ', or it is one of $\{t_1, \dots, t_n\}$. In the latter case, the corresponding tree s_i certifies that $x \in \Lambda'(v)$, due to v and x being in the same connected component of $G - u$ and the way s_i is defined.

Suppose now that v belongs to some other component of $G - u$. Then, every tree of Λ that simultaneously contains v and some element of X, must also contain u. Consequently, every tree that certifies that X is captured by v has to be among $\{t_1, \dots, t_n\}$. By the construction of trees s_i, it follows that X is captured in Λ' by any vertex of component C. □

C.2 Proof of Lemma 3.4

This section is entirely devoted to the proof of Lemma 3.4. We first show how guidance systems can be guessed in MSO, and then apply this understanding to prove the lemma.

***Describing a guidance system in* MSO.** The point of guidance systems is that they can be encoded by an MSO transduction, as stated in Lemma C.1 below. We use the name *k-decorated graph* for a graph G with a distinguished family $\mathcal{X}$ of subsets of vertices, where each subset from $\mathcal{X}$ has size at most k. A k-decorated graph is represented as a logical structure $G^{\mathcal{X}}$ as follows. The universe is the vertices and edges of G, plus a new element for each set from $\mathcal{X}$. The vocabulary consists of two binary relations incident, element and one unary relation decoration, where incident describes the incidence between vertices and edges in G, decoration selects elements representing sets $X \in \mathcal{X}$, and element selects pairs (v, X) with $v \in X \in \mathcal{X}$.

We also define the *enriched* representation of a k-decorated graph $G^{\mathcal{X}}$ as follows. In addition to the predicates defined above, the structure contains also predicate $\mathsf{element}_i$ for each $i \in \{1, 2, \dots, k\}$. We require the following two conditions:

- Relation element is the disjoint union of all relations $\mathsf{element}_i$; that is, $\mathsf{element}(u, X)$ holds if and only if $\mathsf{element}_i(u, X)$ holds for some i, and $\mathsf{element}_i(u, X)$ and $\mathsf{element}_j(u, X)$ cannot hold simultaneously for $i \neq j$.
- For each set $X \in \mathcal{X}$ and each $i \in \{1, \dots, k\}$, there are no two different vertices u, u' for which $\mathsf{element}_i(u, X)$ and $\mathsf{element}_i(u', X)$ hold simultaneously.

Intuitively, the enriched representation is also supplied with some coloring of relation element, using which every set $X \in \mathcal{X}$ can distinguish between its elements using indices from 1 to k. Note that every $G^{\mathcal{X}}$ has a unique representation, but multiple enriched representations: the relation element may be partitioned in different ways into relations $\mathsf{element}_i$ to satisfy the conditions above.

Lemma C.1. *Let $k \in \mathbb{N}$. There exists an* MSO *transduction from graphs to enriched representations of decorated graphs with the following property: when given a graph G on the input, the transduction outputs some enriched representation of every k-decorated graph $G^{\mathcal{X}}$ for which $\mathcal{X}$ can be captured by some k-colorable guidance system in G.*

Proof. We construct an MSO transduction by composing the following seven steps. The MSO-definability of the stated properties will be always straightforward.

(1) Using coloring, guess $3k$ subsets
$$F_1, \dots, F_k \quad \text{and} \quad X_1, \dots, X_k \quad \text{and} \quad R_1, \dots, R_k.$$
(2) Using interpretation that uses only the domain check φ_{dom} and preserves the structure intact, verify the following for each $i = 1, 2, \dots, k$:
   - Sets X_i and R_i are subsets of vertices, and $R_i \subseteq X_i$.
   - Set F_i is an acyclic subset of edges, and every edge of F_i has both endpoints in X_i.
   - Each connected component of the forest (X_i, F_i) (i.e. forest with vertices X_i and edges F_i) contains exactly one vertex from R_i.

(3) Using interpretation, for each $i = 1, \dots, k$ introduce a predicate
$$\mathsf{root}_i(u, v)$$
with the following semantics: $\mathsf{root}_i(u, v)$ is true if, and only if u belongs to X_i and v is the unique vertex of R_i which belongs to the connected component of the forest (X_i, F_i) that contains u.

(4) Copy the universe $1 + 2^k$ times. Let us index the layers of the copied universe as
$$L_0 \cup \{L_A \colon A \subseteq \{1, \dots, k\}\}.$$
Let L' be equal to the union of all layers apart from L_0.

(5) Using interpretation, define predicate $\mathsf{element}_i(v, e)$ as follows: $\mathsf{element}_i(v, e)$ is true if, and only if v is a vertex from L_0 and e is a copy of a vertex contained in some L_A, for which $i \in A$ and $\mathsf{root}_i(e, v)$ holds. Define element as the union of relations $\mathsf{element}_i$.

(6) Using coloring and interpretation, guess any unary predicate decoration that is true only in some elements of L' that are copies of vertices.

(7) Using interpretation again, clean the structure using the following steps:
   - Trim the universe to the elements belonging to L_0 or satisfying decoration.
   - Restrict all the predicates to L_0, with the exception of element, $\mathsf{element}_i$, and decoration.
   - Remove predicates root_i.
   - Verify in the domain check φ_{dom} that the output is an enriched representation of a k-decorated graph.

Clearly, the composition of the steps above is an MSO transduction. From the construction it should be clear that for every k-decorated graph $G^{\mathcal{X}}$, for which $\mathcal{X}$ can be captured by some k-colorable guidance system in G, some enriched representation of $G^{\mathcal{X}}$ will be output by the interpretation. Indeed, in steps (1)–(3) we guess a k-colorable guidance system Λ by guessing the forest corresponding to each color separately, and we encode the function $\Lambda(\cdot)$ using predicates root_i. In steps (4)–(7) we guess and construct the decorated graph $G^{\mathcal{X}}$: each decoration $X \in \mathcal{X}$ is encoded using a vertex of the graph which captures X in Λ, plus the set of colors of trees that lead from this vertex to the vertices of X. The partition of relation element into relations $\mathsf{element}_i$ is defined according to the colors of trees in the guessed guidance system. □

In order to prove Lemma 3.4, it suffices to combine Lemma C.1 with the following lemma, which says that a transduction can recover a sane tree decomposition from its family of adhesions, assuming that the adhesions are not larger than k.

Lemma C.2. *Let $k \in \mathbb{N}$. There is an* MSO *transduction from enriched representations of k-decorated graphs to tree decompositions, which maps an enriched representation of a k-decorated graph $G^{\mathcal{X}}$ to all sane tree decompositions of G for which the family of adhesions is $\mathcal{X}$.*

Proof. We first explain how any sane tree decomposition t can be encoded using a very simple object in a graph, essentially a forest colored with two colors. For this, we first need some simple claims about sane tree decompositions.

Claim 10. *Suppose t is a sane tree decomposition of a graph G, y is a node of t, and x is the parent of y. Then*

$$\sigma(y) \cap \mu(x) \neq \emptyset,$$

that is, there exists a vertex of G that is simultaneously in the adhesion of y and in the margin of its parent x.

Proof. By condition (a) of Definition 2.7, both margins $\mu(x)$ and $\mu(y)$ are not empty. They are also disjoint and contained in $\alpha(x)$, which, by condition (b) of Definition 2.7, is connected in G. Hence, in G there is a path from a vertex of $\mu(y)$ to a vertex of $\mu(x)$ that traverses only vertices contained within $\alpha(x)$. Since $\mu(y)$ is a subset of $\alpha(y)$ and $\mu(x)$ is disjoint with $\alpha(y)$, this path needs to traverse some vertex u of $\sigma(y)$. Since the path is entirely contained in $\alpha(x)$, we have that $u \in \sigma(y) \cap \alpha(x) = \sigma(y) \cap \mu(x)$. ⌟

Suppose we are given a sane tree decomposition t of a graph G. Define $G\langle t\rangle$ to be the graph obtained from G by adding an edge between every pair of nonadjacent vertices of G that appear together within some adhesion of t; thus, $G\langle t\rangle$ is obtained from G by turning every adhesion into a clique. Obviously, t is still a sane tree decomposition of $G\langle t\rangle$. Note that the subgraphs of $G\langle t\rangle$ induced by the margins of the nodes of t are exactly the marginal graphs of these nodes. Hence, they are nonempty and connected due to the saneness of t. We now show that the marginal graphs also have connections to the adhesion of the corresponding nodes.

Claim 11. *Suppose t is a sane tree decomposition of a graph G, and x is some its node. Then every vertex of $\sigma(x)$ has a neighbor in $\mu(x)$ in the graph $G\langle t\rangle$.*

Proof. Fix a vertex $v \in \sigma(x)$. Take any vertex $u \in \mu(x)$. By the saneness of t, there is a path P in G that goes from u to v and which, apart from the endpoint v, is entirely contained in $\alpha(x)$. Let u' be the last vertex on this path that belongs to $\mu(x)$; then all the internal vertices on the suffix of P between u' and v need to be outside of $\beta(x)$. If there are no such internal vertices, i.e. this suffix consists only of u' and v, then we are done because u' belongs to $\mu(x)$ and is a neighbor of v in G. Otherwise, all these internal vertices must be contained in the component of some child y of x, and in particular both u' and v belong to the adhesion of y. Then from the construction of $G\langle t\rangle$ it follows that v and u' are adjacent in this graph. ⌟

Claims 10 and 11 suggest the following representation of a sane tree decomposition t. The representation consists of sets M, K, R, where

- M is a subset of edges of $G\langle t\rangle$ obtained as follows: for each node x of t, take an arbitrary spanning tree of the marginal graph of x, and as M take the union of the edge sets of these spanning trees.
- K is a subset of edges of $G\langle t\rangle$ obtained as follows: for each node y of t with parent x, take any vertex $v \in \sigma(y) \cap \mu(x)$ (which exists by Claim 10), and add to K an arbitrary edge of $G\langle t\rangle$ connecting it to some vertex of $\mu(y)$ (which exists by Claim 11).
- R is a subset of vertices of G that has exactly one vertex in every margin of a root of t, and no other vertices.

Any such triple (M, K, R) will be called an *encoding* of t. It is not hard to see that, given graph $G\langle t\rangle$ together with a triple (M, K, R) that is an encoding of t, one can uniquely decode the whole decomposition t. We do it in the following claim using an MSO transduction.

Claim 12. *There is an* MSO *transduction, whose domain are graphs supplied with a triple of universe subsets, that has the following property: if the input is $G\langle t\rangle$ together with an encoding (M, K, R) of t, for some sane tree decomposition t of a graph G, then the output of the transduction contains t with $G\langle t\rangle$ as the underlying graph.*

Proof. The MSO transduction takes the following steps (all of them are trivially definable in MSO):

1. Using an interpretation that keeps the structure intact and just makes some domain check, verify the following:
   - M and K are disjoint subsets of edges of $G\langle t\rangle$,
   - $M \cup K$ is a spanning forest of $G\langle t\rangle$, and
   - R contains exactly one vertex in each connected component of $G\langle t\rangle$.
2. Using coloring, copying, and interpretation, guess one vertex u_C from each connected component C of the subgraph spanned by M, and create its copy x_C. This copy will be used as the node in the output tree decomposition whose margin is the vertex set of C.
3. For each node x_C, define the parent of x_C to be the node $x_{C'}$ with the following property: If P is the unique path in $M \cup K$ leading from u_C to a vertex of R, then C' is the next connected component of the subgraph spanned by M that is visited by P. From now on we can talk about ancestor-descendant relation between nodes x_C. The goal now is, having the tree structure and the set of margins, to decode the bags of t
4. For each node x_C, define the component of x_C to be the union of the vertex sets of all C' for which either $x_{C'} = x_C$, or $x_{C'}$ is a descendant of x_C.
5. Finally, for each node x_C define the bag of x_C as the vertex set of C, plus all the neighbors of vertices in the component of x that do not belong to this component.
6. At the end, clean the structure from unnecessary relations and verify that the output is a sane tree decomposition whose encoding is (M, K, R).

It is straightforward to see that if (M, K, R) is an encoding of a sane tree decomposition t, then t will be the unique (up to isomorphism) decomposition output by the transduction. ⌟

Therefore, it remains to argue how to construct a suitable graph $G\langle t\rangle$ from the input graph G, because then we will be able to guess sets M, K, R using coloring, run the transduction of Claim 12, drop the added edges from the underlying graph, and at the end verify that the output is a sane tree decomposition of G whose family of adhesions is $\mathcal{X}$. This is, however, very easy. Namely, for all sane tree decompositions t for which $\mathcal{X}$ is the family of adhesions, the graphs $G\langle t\rangle$ are isomorphic: they are obtained from G by making each set of $\mathcal{X}$ into a clique. Since all the sets of $\mathcal{X}$ are of size at most k and we are working with an enriched representation, the construction of this graph can be easily done using an MSO transduction as follows.

Create $\binom{k}{2}$ copies of each decoration $X \in \mathcal{X}$, and index them using 2-element subsets of $\{1, 2, \dots, k\}$. For a copy corresponding to a pair $\{i, j\} \subseteq \{1, \dots, k\}$, verify whether there is a pair of vertices u, v in X for which $\mathsf{element}_i(u, X)$ and $\mathsf{element}_j(u, X)$ hold, and moreover u and v are nonadjacent in G. By the definition of an enriched representation, there is at most one such pair u, v. If the pair u, v does not exist, then remove the copy from the universe, and otherwise turn the copy into an edge between u and v. Observe that, in this manner, an edge uv could have been added multiple times to the structure, in case $\{u, v\}$ is a subset of multiple sets from $\mathcal{X}$. However, we can remove the multiplicities by guessing one duplicate of each edge to preserve, and removing all the others. $\square$

D. Omitted proofs from Section 4

Proof of Lemma 2.5, assuming Lemma 4.4. Let f be the function from Lemma 4.4, and let $k \in \mathbb{N}$. Applying Lemma 3.4, there is an MSO transduction $\mathcal{S}_{f(k)}$ that, given a graph, outputs all its sane tree decompositions for which the family of adhesions can be captured by an $f(k)$-colorable guidance system. Thanks to Lemma 4.4, we know that if the input graph has pathwidth at most k, then some its tree decomposition t is captured by an $f(k)$-colorable guidance system. By applying Lemma 2.8, we can further assume that t is sane; indeed, if a guidance system captures some bag, then it also captures all its subsets. Since adhesions of t are subsets of bags of t, we infer that the same guidance system also captures the family of adhesions of t, and hence t is among the outputs of $\mathcal{S}_{f(k)}$. To conclude the proof it remains to take $\mathcal{S}_{f(k)}$ and add a verification (in the domain check φ_{dom}) that the output decomposition has width at most $f(k)$, which clearly can be expressed in MSO for a constant k. $\square$

Proof of Lemma 4.3. Ad (1). We prove inequalities in both directions. First, if t is a tree decomposition of $G \uplus G'$ captured by some guidance system Λ, then restricting Λ only to trees contained in G yields a guidance system that captures a tree decomposition of G — the one derived from t by intersecting every bag with the vertex set of G. By performing the same reasoning for G', we conclude that $\mathtt{gtw}(G \uplus G') \geq \max(\mathtt{gtw}(G), \mathtt{gtw}(G'))$. Second, if t and t' are tree decompositions of G and G', respectively, then their disjoint union is a tree decomposition of $G \uplus G'$. This is because in Definition 2.1, we allow a tree decomposition to be a forest. If t and t' are captured by k-colorable guidance systems, over disjoint graphs, then the union of these guidance systems is also k-colorable and captures $t \uplus t'$. This proves that $\mathtt{gtw}(G \uplus G') \leq \max(\mathtt{gtw}(G), \mathtt{gtw}(G'))$.

Ad (2). Using (1), we assume that G is connected. Let t be a tree decomposition of $G - u$ which can be captured by a k-colorable guidance system over $G - u$, and therefore also over G. Add the vertex u to every bag, creating a new decomposition. Since G is connected, every bag of the new tree decomposition is contained in some connected component of G. By claim (1) of Lemma 3.5, every bag of the new decomposition can be captured by a guidance system with at most $k + 1$ colors over G.

Ad (3). Let t be a tree decomposition of G which is captured by a k-colorable guidance system over G. Define a tree decomposition of $G - u$ as follows: for each connected component of $G - u$, create a tree decomposition by restricting all bags of t to that component, and then take the union of all of these tree decompositions. Every bag of the new tree decomposition is contained in the intersection of some bag of t with some connected component of $G - u$. Therefore, by claim (2) of Lemma 3.5, the new tree decomposition can be captured by a $2k$-colorable guidance system over $G - u$. $\square$

E. Omitted proofs from Section 5

Proof of Lemma 5.3. Take any path P from the source to the sink, and let $e_1, e_2, \dots, e_p$ be the order in which the cutedges appear on this path. Suppose there is a path P' for which the i-th cutedge appearing on P' is different than e_i, and let i be the smallest index with this property for P'. Then the i-th cutedge appearing on P' is e_j for some $j > i$. Construct a path Q, leading from the source to the sink, by first traversing a prefix of P' up to the cutedge e_j, and then traversing a suffix of P from the cutedge e_j. Since neither this prefix nor this suffix traverses e_i, we conclude that Q is a path from the source to the sink that avoids e_i. This is a contradiction with e_i being a cutedge. $\square$

Proof of Lemma 5.4. Let H be the underlying hypergraph of the network. By a *source-sink path* we will mean a path going from the source of the network to its sink. By the connectedness of H, every cutedge component intersects at least one cutedge, i.e., one of elements $e_1, \dots, e_p$.

Let C be a cutedge component of H. For the sake of contradiction, suppose that C intersects some two elements that are not consecutive in the sequence; say e_i and e_j, where there exists some ℓ with $i < \ell < j$. Observe that excluding this situation already gives us all the claims stated in the lemma.

We first consider the case when $1 \leq i < j \leq p$, i.e., e_i and e_j are cutedges of the network, and not the singletons of the source or sink. Take any source-sink path P; by Lemma 5.3 we know that P visits the cutedges in the order given by the cutedge sequence. Construct a source-sink path as follows: take the prefix of P from the source to cutedge e_i, then travel inside the component C from a vertex belonging to e_i to a vertex belonging to e_j, and finally follow the suffix of P from the cutedge e_j to the sink. Thus, the constructed path leads from the source to the sink and avoids e_ℓ. This is a contradiction with e_ℓ being a cutedge.

By symmetry, we are left with considering the case when $i = 0$, i.e., e_i is the singleton of the source. We can perform almost the same construction as before: If P is any source-sink, then we can obtain a source-sink path that avoids e_ℓ by concatenating any path that leads from the source to a vertex of e_j within component C, and the suffix of P from e_j to the sink. Again, this contradicts the fact that e_ℓ is a cutedge. $\square$

Proof of Lemma 5.5. Let $(e_1, \dots, e_p)$ be the cutedge sequence of the network. Let H' be the hypergraph of the network with all the cutedges removed; the connected components of H' are bridges and appendices. Let V_i be the union of (e_i, e_{i+1})-bridges, for $i = 0, 1, \dots, p$, and let W_i be the union of e_i-appendices, for $i = 1, 2, \dots, p$. For any $i = 0, 1, \dots, p$, consider the hypergraph $H'[V_i]$. By applying Menger's theorem in its incidence graph, i.e., the bipartite graph formed by its vertices and hyperedges, where edge relation is defined by containment, we infer that in $H'[V_i]$ there are two hyperedge-disjoint paths P_i^1 and P_i^2 leading from $e_i \cap V_i$ to $e_{i+1} \cap V_i$. Indeed, otherwise $H'[V_i]$ would contain a hyperedge whose removal would disconnect $e_i \cap V_i$ from $e_{i+1} \cap V_i$; this hyperedge would be also a cutedge of the network that would not be among $e_1, e_2, \dots, e_p$. Now construct P^1 by concatenating alternately paths $P_0^1, P_1^1, \dots, P_p^1$ and cutedges $e_1, e_2, \dots, e_p$, and similarly construct P^2. It follows that both P^1 and P^2 lead from the source to the sink, and the only hyperedges shared by them are $e_1, e_2, \dots, e_p$. $\square$

Proof of Lemma 5.7. We adopt the notation of sets V_i and W_i from the definition of k-thinness. For $i = 0, 1, \dots, p$, let t_i be a path decomposition of $H[V_i]$ that certifies k-thinness. Similarly, let s_i be a path decomposition of $H[W_i]$ that certifies k-thinness. Construct s_i' from s_i by adding all the vertices of e_i to all the bags. As $|e_i| \leq k$, the width of s_i' does not exceed $2k$.

Construct a path decomposition of the given network by concatenating decompositions

$$t_0, s'_1, t_1, s'_2, t_2, \ldots, t_{p-1}, s'_p, t_p$$

in this order. It is easy to verify that this is indeed a path decomposition of the given network, and its width is at most $2k+1$. □

Proof of Lemma 5.8. Let $\mathbb{H}$ be the given network, let u and v be the source and the sink of $\mathbb{H}$, and let H be its hypergraph. Let $(e_1, \ldots, e_p)$ be the cutedge sequence of $\mathbb{H}$, and let us adopt the notation for sets V_i and W_i from the definition of k-thinness. As e is the cutedge of $\mathbb{H}$, we have that $e = e_\ell$ for some $\ell \in \{1, \ldots, p\}$.

Let $\widehat{H} = H[e \to K]$. Since K is connected, by the definition of $\widehat{H}$ it follows that $\widehat{H}$ is connected as well. Hence, $\widehat{H}$ with source u and sink v is a network, which we shall call $\widehat{\mathbb{H}}$. In the following, by a *source-sink* path we mean a path going from u to v. Our first goal is to understand the structure of the cutedges of $\widehat{\mathbb{H}}$.

First, note that each e_i with $i \neq \ell$ is still a cutedge in $\widehat{\mathbb{H}}$. Moreover, these cutedges must be ordered in the same manner in the cutedge sequence of $\widehat{\mathbb{H}}$ as in the cutedge sequence of $\mathbb{H}$. This is because any source-sink path in $\mathbb{H}$ can be lifted to a source-sink path in $\widehat{\mathbb{H}}$ by replacing the usage of e_ℓ with a path within K. Then the order in which the cutedges of $\mathbb{H}$ appear in the lifted path is the same as in the original one. The same argument shows that in the cutedge sequence of $\widehat{\mathbb{H}}$, all cutedges that originate from the hypergraph K have to be after all cutedges e_j for $j < \ell$, and before all cutedges e_j for $j > \ell$.

Next, we observe that every hyperedge f of H that is not a cutedge in $\mathbb{H}$, does not becomes a cutedge in $\widehat{\mathbb{H}}$. Indeed, Lemma 5.5 implies that in $\mathbb{H}$ there is a source-sink path that avoids f. By lifting this path in the same manner as in the previous paragraph, we obtain a source-sink path in $\widehat{\mathbb{H}}$ that avoids f; this certifies that f is not a cutedge in $\widehat{\mathbb{H}}$.

We conclude that the cutedge sequence of $\widehat{\mathbb{H}}$ has the following form:

$$(e_1, \ldots, e_{\ell-1}, f_1, \ldots, f_q, e_{\ell+1}, \ldots, e_p),$$

where $f_1, f_2, \ldots, f_q$ are the cutedges of $\widehat{\mathbb{H}}$ that originate in K. In what follows we assume for simplicity that $q > 0$; at the end we shortly discuss how the proof needs to be adjusted in the case when no new cutedge arises in the network.

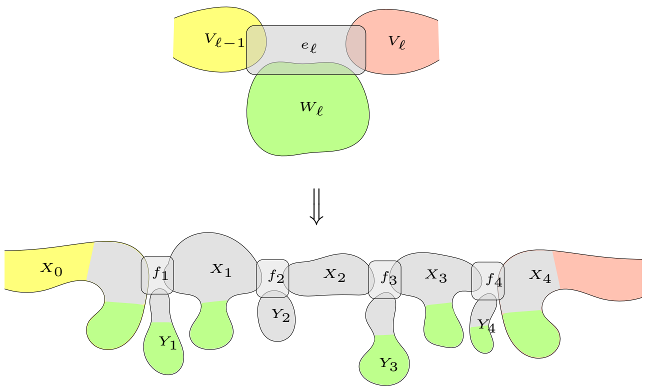


Figure 4: Part of the cutedge decomposition affected by replacement of a cutedge e; here, $q = 4$.

From Lemma 5.4 it straightforward to see that the bridges and appendices in $\widehat{\mathbb{H}}$ have the following structure (see Figure 4):

- For each $i \in \{0, \ldots, p\} - \{\ell-1, \ell\}$, every (e_i, e_{i+1})-bridge in $\mathbb{H}$ remains an (e_i, e_{i+1})-bridge in $\widehat{\mathbb{H}}$.
- For each $i \in \{1, \ldots, p\} - \{\ell\}$, every e_i-appendix in $\mathbb{H}$ remains an e_i-appendix in $\widehat{\mathbb{H}}$.
- For each $j \in \{1, \ldots, q-1\}$, every (f_j, f_{j+1})-bridge consists only of vertices originating in K or W_ℓ.
- For each $j \in \{1, \ldots, q\}$, every f_j-appendix consists only of vertices originating in K or W_ℓ.
- Each $(e_{\ell-1}, f_1)$-bridge consists only of vertices originating in $V_{\ell-1}$, K, or W_ℓ.
- Each $(f_q, e_{\ell+1})$-bridge consists only of vertices originating in V_ℓ, K, or W_ℓ.

In particular, the vertices originating in K and W_ℓ are partition between bridges and appendices that intersect cutedges f_j. Also, only the $(e_{\ell-1}, f_1)$- and $(f_q, e_{\ell+1})$-bridges may contain some vertices not originating in K or W_ℓ: each $(e_{\ell-1}, e_\ell)$-bridge of the original network $\mathbb{H}$ is contained in some $(e_{\ell-1}, f_1)$-bridge of $\widehat{\mathbb{H}}$, and likewise for $(e_\ell, e_{\ell+1})$-bridges of $\mathbb{H}$.

We now verify that $\widehat{\mathbb{H}}$ is k-thin by exposing appropriate path decompositions for all the relevant parts. Due to the first two items above, for the unions of (e_i, e_{i+1})-bridges, for $i \in \{0, \ldots, p\} - \{\ell-1, \ell\}$, and the unions of e_i-appendices, for $i \in \{1, \ldots, p\} - \{\ell\}$, we can use the same decompositions as the ones that certify that the original network $\mathbb{H}$ is k-thin. Hence we are left only with the relevant unions of bridges and appendices that intersect the new cutedges f_j. For $j = 1, \ldots, q+1$, let X_j be the union of the vertex sets of (f_{j-1}, f_j)-bridges in $\widehat{\mathbb{H}}$, where $f_0 = e_{\ell-1}$ and $f_{q+1} = e_{\ell+1}$. Similarly, for $j = 1, \ldots, q$, let Y_j be the union of the vertex sets of f_j-bridges in $\widehat{\mathbb{H}}$.

Consider first the part X_j for some $j \in \{1, 2, \ldots, q-1\}$. The vertices of this part originate in $W_\ell \cup V(K)$, where $V(K)$ denotes the vertex set of K. Recall that $H[W_\ell]$ admits some path decomposition s of width at most k with $W_\ell \cap e_\ell$ contained in the first bag. If W_ℓ is empty, then we take a decomposition consisting of one node with an empty bag; this applies also to similar situations in the next paragraphs. Consequently, by intersecting each bag of s with X_j we infer that $H[X_j \cap W_\ell]$ admits a path decomposition s' of width at most k with $(W_\ell \cap X_j) \cap e_\ell$ contained in the first bag. Recall also that K has at most $k+1$ vertices, so $|X_j \cap V(K)| \leq k+1$. Hence, a suitable path decomposition of $\widehat{H}[X_j]$ can be obtained by taking s' and adding $X_j \cap V(K)$ to every its bag; thus, the obtained decomposition has width at most $2k+1$. Observe that $X_j \cap f_{j-1} \subseteq X_j \cap V(K)$ and $X_j \cap f_{j-1} \subseteq X_j \cap V(K)$. Hence $X_j \cap f_{j-1}$ is contained in the first bag of this path decomposition, whereas $X_j \cap f_j$ is contained in the last bag, as required.

Consider now the part X_0. Since $V_{\ell-1} \subseteq X_0$, we can partition X_0 into $V_{\ell-1}$ and $X_0 \setminus V_{\ell-1}$, where the latter set will be denoted by M. By the assumption that $\mathbb{H}$ is k-thin, $H[V_{\ell-1}]$ admits a path decomposition t of width at most $2k+1$ where $V_{\ell-1} \cap e_{\ell-1}$ is contained in the first bag and $V_{\ell-1} \cap e_\ell$ is contained in the last bag. On the other hand, $M \subseteq W_\ell \cup V(K)$, and hence it can be further partitioned into $M \cap W_\ell$ and $M \setminus W_\ell \subseteq V(K) \cap X_0$. As in the previous paragraph, by the assumption of k-thinness we infer that $H[M \cap W_\ell]$ admits a path decomposition s of width at most k with $(M \cap W_\ell) \cap e_\ell$ contained in the first bag. Obtain a path decomposition s' by adding all the vertices of $V(K) \cap X_0$ to every bag of s; since $|V(K)| \leq k+1$, we have that the width of s' does not exceed $2k+1$. Finally, obtain a path decomposition of $\widehat{H}[X_0]$ by concatenating decompositions t and s'. It is easy to see that this is indeed a path decomposition of $\widehat{H}[X_0]$ and its width obviously is at most $2k+1$. Moreover, $X_0 \cap e_{\ell-1} = V_{\ell-1} \cap e_{\ell-1}$ is contained in its first bag (by the assumption about t) and $X_0 \cap f_1 \subseteq V(K) \cap X_0$ is contained in the last bag (by the construction).

A symmetric reasoning can be performed for the part X_q.

We are left with considering part Y_j, for any $j \in \{1, \ldots, q\}$. Again, the vertices of this part originate in $W_\ell \cup V(K)$. As before, by the assumption about k-thinness of $\mathbb{H}$, we have that $H[W_\ell \cap Y_j]$ admits a path decomposition s of width at most k with $(W_\ell \cap Y_j) \cap e_\ell$ contained in the first bag. Let s' be a path decomposition obtained from s by prepending (i.e., adding at the front) bag $V(K) \cap Y_j$. It is easy to verify that s' is indeed a path decomposition of $\widehat{H}[Y_j]$; this follows from the fact that $(V(K) \cap Y_j) \cap (W_\ell \cap Y_j) = (W_\ell \cap Y_j) \cap e_\ell$, and this set is contained in the first bag of s. Moreover, s' has width at most k due to $|V(K)| \leq k + 1$, and its first bag contains $Y_j \cap f_j \subseteq V(K) \cap Y_j$, as required.

Having considered all the parts, we conclude that $\widehat{\mathbb{H}}$ is k-thin in the case when $q > 0$. To see that case $q = 0$ can be analyzed in the same manner, it suffices to combine the arguments for X_0 and X_q. Precisely, a path decomposition of $\widehat{H}[X_0]$ is formed by concatenating: (i) the path decomposition of $H[V_{\ell-1}]$ certifying k-thinness of $\mathbb{H}$, (ii) the path decomposition of $H[W_\ell]$ certifying k-thinness of $\mathbb{H}$, with all the vertices of K added to each bag, and (iii) the path decomposition of $H[V_\ell]$ certifying k-thinness of $\mathbb{H}$. $\square$