ADJACENCY LABELLING FOR PROPER MINOR-CLOSED GRAPH CLASSES

Vida Dujmović [♀] *Cyril Gavoille* [♁] *Gwenaël Joret* [♂]
Piotr Micek [♃] *Pat Morin* [♄] *David R. Wood* [♅]

Abstract. We show that every proper minor-closed class of graphs admits a $(1+o(1))\log_2 n$-bit adjacency labelling scheme. Equivalently, for every proper minor-closed class $\mathcal{G}$ and every positive integer n there exists an $n^{1+o(1)}$-vertex graph U such that every n-vertex graph in $\mathcal{G}$ is isomorphic to an induced subgraph of U. Both results are optimal up to the lower order term. They generalize the corresponding results for planar graphs and apex-minor-free classes (Dujmović et al., J. ACM 2021) to all proper minor-closed classes, answering the open question raised in that paper and anticipated earlier by Bonamy, Gavoille, and Pilipczuk (SODA 2020).

1 Introduction

Let $\mathcal{G}$ be a class of graphs and let $f : \mathbb{N} \to \mathbb{N}$ be a function. We say that $\mathcal{G}$ admits an *$f(n)$-bit adjacency labelling scheme* if there exists a function $A : (\{0,1\}^*)^2 \to \{0,1\}$ such that for all positive integers n, for every n-vertex graph $G \in \mathcal{G}$, there exists a function $\ell : V(G) \to \{0,1\}^*$ such that $|\ell(v)| \leqslant f(n)$ for each vertex v in G, and such that for every two vertices u, v in G,

$$A(\ell(u), \ell(v)) = \begin{cases} 0 & \text{if } uv \notin E(G), \\ 1 & \text{if } uv \in E(G). \end{cases}$$

A graph H is a *minor* of a graph G if a graph isomorphic to H can be obtained from a subgraph of G by contracting edges. A class of graphs $\mathcal{G}$ is *minor-closed* if every minor of every graph in $\mathcal{G}$ is also in $\mathcal{G}$, and it is *proper* if it is not the class of all graphs. Some examples of proper proper minor-closed include planar graphs, graphs of treewidth at

[♀] School of Computer Science and Electrical Engineering, University of Ottawa, Ottawa, Canada (vida.dujmovic@uottawa.ca). Research supported by NSERC and a University of Ottawa Research Chair.

[♁] LaBRI, University of Bordeaux, France (gavoille@labri.fr). This work is supported by the French ANR Projets ENEDISC (ANR-24-CE48-7768-01) and TEMPOGRAL (ANR-22-CE48-0001).

[♂] Département d'Informatique, Université libre de Bruxelles, Belgium (gwenael.joret@ulb.be). G. Joret is supported by the Belgian National Fund for Scientific Research (FNRS) and by the Australian Research Council.

[♃] Department of Theoretical Computer Science, Faculty of Mathematics and Computer Science, Jagiellonian University, Kraków, Poland (piotr.micek@uj.edu.pl). Research supported by the National Science Center of Poland under grant UMO-2023/05/Y/ST6/00079 within the WEAVE-UNISONO program.

[♄] School of Computer Science, Carleton University, Ottawa, Canada (morin@scs.carleton.ca). Research supported by NSERC.

[♅] School of Mathematics, Monash University, Melbourne, Australia (david.wood@monash.edu). Research supported by the Australian Research Council and by NSERC.

most k, graphs embeddable in a fixed surface, linklessly embeddable graphs, knotlessly embeddable graphs and, for every fixed graph X, the class of graphs with no X-minor. Note that every proper minor-closed class is contained in the class of K_t-minor-free graphs for some fixed t.

In this paper we prove the following result. (All logarithms are in base 2.)

Theorem 1. *Every proper minor-closed class of graphs admits a* $(1+o(1))\log n$*-bit adjacency labelling scheme.*

Note that all the dependence on the fixed proper minor-closed class in Theorem 1 is in the $o(\log n)$ term. Also, Theorem 1 is optimal up to the $o(\log n)$ term, which is $O\left((\log n)^{3/4}\right)$. The proof of Theorem 1 is constructive. For every fixed proper minor-closed class $\mathcal{G}$, there is a polynomial-time algorithm that takes a graph $G \in \mathcal{G}$ as input and constructs the labelling $\ell : V(G) \to \{0,1\}^*$.

A consequence[1] of Theorem 1 is the existence of induced-universal graphs with a near-linear number of vertices for n-vertex graphs belonging to a fixed proper minor-closed class of graphs.

Corollary 2. *For every proper minor-closed class* $\mathcal{G}$*, for every positive integer* n*, there exists a graph* U *with* $n^{1+o(1)}$ *vertices such that every* n*-vertex graph in* $\mathcal{G}$ *is isomorphic to an induced subgraph of* U*.*

1.1 State of the Art

Adjacency labelling schemes were introduced in the late 1980s by Kannan, Naor, and Rudich [30] and independently in the PhD thesis of Muller [37]. Since this initial work, labelling schemes have been developed for many other queries besides adjacency, including distances [21], ancestry and nearest common ancestors in trees [1, 4, 5], reachability in planar digraphs [29, 41], and flows and connectivity [31]. A review of this vast literature is beyond the scope of this paper. Here we review results most relevant to the current work, focusing mainly on trees, bounded treewidth graphs, and planar graphs.

Adjacency labelling schemes have also been studied for other general families of graph classes. At the dense end of the spectrum, Bonamy, Esperet, Groenland, and Scott [7] constructed asymptotically optimal adjacency labelling schemes for every hereditary class containing $2^{\Omega(n^2)}$ n-vertex graphs. In particular, they proved that comparability graphs admit a labelling with labels of length $n/4 + o(n)$, which is best possible up to the lower-order term. Proper minor-closed classes lie at the opposite, sparse end of the spectrum, where the situation is markedly different, as we now discuss.

We start with a simple example to better grasp the notion of labelling schemes. Given an n-vertex tree T, fix an arbitrary vertex to be the root of T, and assign each vertex of T a unique identifier, i.e., a number in $\{0,\dots,n-1\}$. For each non-root vertex v of T,

[1] In fact, the viewpoints of adjacency labelling schemes and induced-universal graphs are essentially equivalent. A minor detail is that the labelling schemes need to be injective for the equivalence to hold but this is the case for those developed in this paper. See [40, Section 2.1] for more details on the connection between the two notions.

let the label of v be the pair consisting of its identifier and the identifier of its parent. For the root vertex of T, let its label be the pair consisting of two copies of its identifier. Clearly, given two labels of vertices in T, a test of adjacency is simply to verify whether an identifier of one vertex is equal to a stored identifier of the parent of the other vertex. This constitutes a $\lceil \log(n^2) \rceil = \lceil 2 \log n \rceil$-bit adjacency labelling scheme for trees. There is a fascinating series of results improving on this simple scheme for trees. In 1990, a result of Chung [11] about induced-universal graphs implies that n-vertex trees admit an adjacency labelling scheme with labels of length $\log n + \log\log n + O(1)$. In 2002, Alstrup and Rauhe [6] devised an improved scheme with labels of length $\log n + O(\log^* n)$. Finally, in 2015, Alstrup, Dahlgaard, and Knudsen [3] gave a $(\log n + O(1))$-bit adjacency labelling scheme for trees, which is optimal up to the $O(1)$ term. Indeed, consider an n-vertex graph with no two vertices having the same neighborhood (e.g., a path for $n \geqslant 4$). In any adjacency labelling scheme, each vertex must be assigned a distinct label and therefore one label must be of length at least $\log(n+1) - 1$.

In 2007, Gavoille and Labourel [20] presented a $(1 + o(1)) \log n$-bit adjacency labelling scheme for graphs of bounded treewidth. This is particularly relevant to the topic of this paper because of the following famous application of the graph minor structure theorem: DeVos, Ding, Oporowski, Sanders, Reed, Seymour, and Vertigan [13] showed that every graph in a proper minor-closed class can be edge 2-coloured so that each monochromatic subgraph has bounded treewidth. Therefore, given a proper minor-closed class of graphs $\mathcal{G}$, for each G in $\mathcal{G}$, fix such a 2-colouring of the edges of G, say with red and blue, and label each vertex of G by the concatenation of the two labels given by the adjacency labelling schemes for the red subgraph of G and for the blue subgraph of G. Thus, all labels are of length $(2 + o(1)) \log n$. Given two labels, to test adjacency simply test whether there is a blue edge or whether there is a red edge. This gives a $(2+o(1)) \log n$-bit adjacency labelling scheme for a fixed proper minor-closed class of graphs, which remained the best-known scheme up to the present work.

Planar graphs are a prominent example of a proper minor-closed class of graphs and there is a long history of research on their adjacency labelling schemes. Since planar graphs are 5-degenerate, they admit a simple $\lceil 6 \log n \rceil$-bit adjacency labelling scheme, which was already observed by Muller [37]. Kannan et al. [30] use a similar approach that makes use of the fact that planar graphs have arboricity 3 (so their edges can be partitioned into three forests [38]) to devise an adjacency labelling scheme for planar graphs whose labels have length at most $\lceil 4 \log n \rceil$. This length was reduced to $3 \log n + O(\log^* n)$ by the improved scheme of Alstrup and Rauhe [6] for arboricity-k graphs. Of course, the later result of Gavoille and Labourel [20] also applies to planar graphs, and gives a scheme with labels of length $(2+o(1)) \log n$. The most recent breakthrough came with the application of the product structure theorem for planar graphs by Dujmović, Joret, Micek, Morin, Ueckerdt, and Wood [16]. In 2020, Bonamy, Gavoille, and Pilipczuk [8] recognized that the product structure is particularly useful for adjacency labelling, breaking the $(2+o(1)) \log n$ barrier by introducing a scheme with labels of length $(4/3 + o(1)) \log n$. Finally, in 2020 Dujmović, Esperet, Gavoille, Joret, Micek, and Morin [15] used the product structure theorem to give a $(1 + o(1)) \log n$-bit adjacency labelling scheme for planar graphs, which is optimal up to the lower order term. In Dujmović et al. [15], the lower order term is in

$O(\sqrt{\log n \log\log n})$. Gawrychowski and Janczewski [22] later introduced a simplification that reduces the lower order term to $O(\sqrt{\log n})$ and allows for the adjacency testing function to be implemented in constant time in a realistic model of computation.

The adjacency labelling scheme of Dujmović et al. [15] works for graphs embeddable in any fixed surface, and more generally for any minor-closed class excluding a fixed apex graph as a minor. Here a graph X is *apex* if X can be made planar by the removal of at most one vertex. Since K_5 is apex, this implies a $(1+o(1))\log n$-bit adjacency labelling scheme for K_5-minor-free graphs. Such apex-minor-free classes are the limit of this approach, since a minor-closed class $\mathcal{G}$ has 'product structure' if and only if some apex graph is not in $\mathcal{G}$. Indeed, the existence of a $(1+o(1))\log n$-bit adjacency labelling scheme for K_t-minor-free graphs is stated as an open problem by Dujmović et al. [15, Section 6, Problem 2]. Bonamy et al. [8] likewise anticipated that such labelling schemes might extend to all proper minor-closed classes via the graph minor structure theorem of Robertson and Seymour, and identified the difficulty: "we do not know how to combine labeling schemes along tree decompositions". Prior to the present work, the best result for K_t-minor-free graphs, $t>5$, was the $(2+o(1))\log n$-bit adjacency labelling scheme of Gavoille and Labourel [20], who also conjectured that labels of length $\log n + O(1)$ suffice for every proper minor-closed class of graphs.

Finally we discuss the relationship between our result and the Implicit Graph Conjecture. A hereditary graph class is *small* if it has $2^{O(n\log n)}$ labelled n-vertex graphs for all n. For example, every proper minor-closed graph class is small [39]. The Implicit Graph Conjecture [30] asserted that every small graph class admits an $O(\log n)$-bit adjacency labelling scheme. Proper minor-closed classes satisfy this conjecture, and indeed they admit labels of only $(1+o(1))\log n$ bits by Theorem 1. The conjecture in general was disproved by Hatami and Hatami [27], who in fact showed there exist small graph classes for which every adjacency labelling scheme needs $\Omega(n^{1/2-\epsilon})$ bits. See [2, 9, 10] for recent extensions. In light of the counterexamples of [27], it remains an interesting problem to determine which hereditary classes admit $O(\log n)$-bit, or even $(1+o(1))\log n$-bit, adjacency labelling schemes. Our result shows that excluding a fixed minor suffices for the strongest conclusion.

1.2 Setup and Proof Overview

In this paper, all graphs are finite, simple, and undirected. For a graph G, $V(G)$ denotes the vertex-set of G, $E(G)$ denotes the edge-set of G. A *clique* in a graph is a non-empty set of pairwise adjacent vertices. Let G be a graph and $X \subseteq V(G)$. By $G[X]$ we denote the subgraph of G induced on X. For other standard graph theory terms, we refer the reader to the textbook by Diestel [14].

A *tree-decomposition* of a graph G is a pair $(T,(B_x \mid x \in V(T)))$, where T is a tree and $B_x \subseteq V(G)$ for every $x \in V(T)$, with the following properties: (a) for each vertex u in G, the subgraph of T induced by $\{x \in V(T) \mid u \in B_x\}$ is non-empty and connected; and (b) for each edge uv in G, there exists $x \in V(T)$ such that $u,v \in B_x$. We call the sets B_x the *bags* of the tree-decomposition and, for each edge xy of T, $B_x \cap B_y$ is an *adhesion* of the tree-decomposition. When T is rooted and x is the parent of y, $B_x \cap B_y$ is the *parent adhesion*

of y. The *parent adhesion* of the root of T is the empty set. The *home* of a vertex v of G is the unique $x \in V(T)$ such that $v \in B_x$ and v is not in the parent-adhesion of x. The *adhesion-width* of a tree-decomposition is the maximum size of an adhesion. For a tree-decomposition $\mathcal{T} := (T, (B_x \mid x \in V(T)))$ of a graph G, for each $x \in V(T)$, the *torso* of x, denoted by $G\langle \mathcal{T}, B_x \rangle$, is the graph with vertex-set B_x in which two distinct vertices u and v are adjacent if $uv \in E(G)$, or there exists $y \in N_T(x)$ such that $u, v \in B_x \cap B_y$.

The *strong product* of graphs A and B, denoted by $A \boxtimes B$, is the graph with vertex-set $V(A) \times V(B)$, where distinct vertices $(v, x), (w, y) \in V(A) \times V(B)$ are adjacent if $v = w$ and $xy \in E(B)$, or $x = y$ and $vw \in E(A)$, or $vw \in E(A)$ and $xy \in E(B)$. For an integer $k \geqslant 0$, let $\mathcal{R}_k$ be the class of graphs isomorphic to a subgraph of $H \boxtimes P$ for some graph H with treewidth at most k and for some path P. The main result of Dujmović et al. [15] is a $(1+o(1))\log n$-bit adjacency labelling scheme for the class $\mathcal{R}_k$, for any fixed k.

For any graph class $\mathcal{G}$, let $\mathcal{G}^{+a}$ be the class of graphs G such that $G - X$ is in $\mathcal{G}$ for some $X \subseteq V(G)$ with $|X| \leqslant a$. It is a simple exercise to show that any $(1 + o(1))\log n$-bit adjacency labelling scheme for a graph class $\mathcal{G}$ implies the existence of a $(1 + o(1))\log n$-bit adjacency labelling scheme for $\mathcal{G}^{+a}$ (see Subsection 3.1 for a more general result). In particular, for any fixed $k, a \geqslant 0$, there exists a $(1 + o(1))\log n$-bit adjacency labelling scheme for $\mathcal{R}_k^{+a}$.

Our starting point is the following variant of the Graph Minor Structure Theorem:

Theorem 3 (Graph Minor Product Structure Theorem [16])**.** *For every proper minor-closed class $\mathcal{G}$ there exist integers $k, a \geqslant 0$ such that every graph in $\mathcal{G}$ has a tree-decomposition in which every torso is in $\mathcal{R}_k^{+a}$.*

We now have enough background to sketch the proof of our main result. We begin by describing natural approaches to the problem, where these approaches have issues, and then explain how we overcome these issues. By the above-mentioned results, it suffices to consider a graph class $\mathcal{G}$ with the following property: For every $G \in \mathcal{G}$, there exists a tree-decomposition $\mathcal{T} := (T, (B_x \mid x \in V(T)))$ of G such that each torso of $\mathcal{T}$ comes from a monotone graph class $\mathcal{G}'$ that has a $((1 + o(1))\log n)$-bit adjacency labelling scheme. Let A' be the adjacency tester for the labelling scheme on $\mathcal{G}'$.

Let G be an n-vertex graph in $\mathcal{G}$ and let $\mathcal{T} := (T, (B_x \mid x \in V(T)))$ be a tree-decomposition of G such that each torso of $\mathcal{T}$ is in the class $\mathcal{G}'$. Each edge of G is present in $G[B_x]$ for at least one $x \in V(T)$. Thus, a natural first approach is to compute an adjacency labelling of $G[B_x]$ for A', for each $x \in V(T)$. The problem with this approach is that a vertex v of G can appear in many bags of $\mathcal{T}$, so it can have many labels. Simply concatenating these labels would result in labels of length much greater than $\log n$.

Assigning homes: For each $x \in V(T)$, let A_x be the parent adhesion of x in $\mathcal{T}$ and let $B_x^- := B_x \setminus A_x$. (Thus x is the home of each $v \in B_x^-$.) Then $\{B_x^- \mid x \in V(T)\}$ is a partition of $V(G)$. A natural second attempt is to compute an adjacency labelling ℓ_x of $G[B_x^-]$ for A', for each $x \in V(T)$. This leaves two closely-related issues:

1. For two vertices $v \in B_x^-$ and $w \in B_y^-$ with $x \neq y$, the adjacency test $A'(\ell_x(v), \ell_y(w))$ does not produce a useful result, since ℓ_x and ℓ_y are labellings of different graphs. To avoid misusing A' this way, we need a way of determining that $x \neq y$ from the labels

of v and w.

2. In the case where $x \neq y$, we need an alternative way of testing if v and w are adjacent, since vw is not present in $\bigcup_{x \in V(T)} G[B_x^-]$ even when $vw \in E(G)$.

These two issues make precise the difficulty that was also identified by Bonamy et al. [8], namely that of combining labellings along a tree-decomposition. The tools we develop next are designed to overcome it.

2-Part labels: To deal with the first issue, we can assign each vertex v a 2-part label $\langle x(v), \ell_x(v) \rangle$, where $x(v)$ is a unique identifier for the node x of T such that $v \in B_x^-$. Doing this naïvely could easily lead to a label length of $2\log n + o(\log n)$. This can be avoided by using a variable-length code so that $|x(v)| = \log n - \log |B^-_{x(v)}| + O(1)$. This way, $|x(v)| + |\ell_x(v)| = |x(v)| + \log |B^-_{x(v)}| + o(\log n) = \log n + o(\log n)$. This solves the first of the two problems above, but still leaves us with the second problem. There is still no way of testing adjacency between vertices with different homes.

Mixed labelling schemes: An important contribution of the current work is to introduce a generalization of adjacency labelling schemes, called *weighted mixed labelling schemes*, that attempts to solve both of these problems simultaneously. The weights play an important but technical role. For the simplicity of this overview we omit the role of the weights for now, speaking only of *mixed labelling schemes*, and return to it at the end of the overview when stating our main technical result. The formal definition appears at the beginning of Section 3. Informally, a mixed labelling μ is defined for a pair of graphs Q^+ and Q, where Q is a spanning subgraph of Q^+, and μ defines labels for both vertices and cliques of Q^+. The vertex labels assigned by μ are used with an adjacency tester A for the spanning subgraph Q. That is, for any two vertices v and w of Q, $A(\mu(v), \mu(w)) = 1$ if and only if $vw \in E(Q)$. In addition to the vertex and clique labels, a mixed labelling assigns, for each clique K of Q^+ and each vertex $u \in K$, a short *local identifier* $\kappa(K, u)$ for u in K. These clique, vertex, and local identifiers are used with an *identity tester* I. For any clique K of Q^+, any vertex $u \in K$, and any vertex v of Q, $I(\mu(K), \kappa(K, u), \mu(v)) = 1$ if and only if $v = u$.

Loosely speaking, a class admits an *efficient* mixed labelling scheme if it admits a $(1 + o(1))\log n$-bit mixed labelling scheme. This notion of efficiency is defined formally in the introduction to Section 3. Appendix A shows that the adjacency labelling scheme for graphs in $\mathcal{R}_k$ given by Dujmović et al. [15] and its improvements by Gawrychowski and Janczewski [22] can be extended to give an efficient mixed labelling scheme for $\mathcal{R}_k$. Lemma 13 shows that the addition of apex vertices to obtain the class $\mathcal{R}_k^{+a}$ can be accommodated by any mixed labelling scheme. The resulting mixed labelling scheme has clique and vertex labels of length $\log n + O(\sqrt{\log n})$ (see Lemma 19).

Thus, for every proper minor-closed graph family $\mathcal{G}$, every graph G in $\mathcal{G}$ has a tree-decomposition $\mathcal{T}$ of bounded adhesion-width whose torsos come from a class that has an efficient mixed labelling scheme. This is the only property of proper minor-closed graph classes that we use.

A solution for short tree-decompositions: We now describe a first attempt to use mixed labelling schemes to simultaneously address the questions of determining if two vertices of G have the same home and testing adjacency between two vertices of G with different

homes. We begin by constructing a mixed labelling μ_x for the pair of graphs $G\langle \mathcal{T}, B_x\rangle - A_x$ and $G[B_x] - A_x$, for each $x \in V(T)$. Then, the label $\mu(v)$ for a vertex v in G has the form

$$\mu(v) := \langle \mu_{x_0}(K_1), \ldots, \mu_{x_{p-1}}(K_p), \mu_{x_p}(v), \alpha(v)\rangle$$

where $x_0, \ldots, x_p$ is the path from the root x_0 of T to the home $x = x_p$ of v, and $K_i = A_{x_i} \setminus A_{x_{i-1}}$ for each $i \in \{1, \ldots, p\}$ is a clique in $G\langle \mathcal{T}, B_{x_{i-1}}\rangle - A_{x_{i-1}}$. Here, $\alpha(v)$ is a form of adjacency list that stores, for each $u \in N_G(v) \cap A_{x_p}$, the index i of the clique K_i that contains u and the local identifier $\kappa_{x_{i-1}}(K_i, u)$.

Now, consider a vertex v whose home is x and a vertex w whose home is y. By examining the sequence of clique labels in $\mu(v)$ and $\mu(w)$, an adjacency tester can determine which of the following four cases applies:

- $x = y$: In this case, the tester can use $\mu_x(v)$ and $\mu_y(w) = \mu_x(w)$ to determine if $vw \in E(G[B_x] - A_x)$.
- x is an ancestor of y: In this case, the path $x_0, \ldots, x_q$ from the root of T to y contains a node $x_p = x$. The only way in which v and w can be adjacent is if $v \in K_{p+1}$. If v and w are adjacent, then $v \in N_G(w) \cap A_{x_q}$ and therefore the pair $(p+1, \kappa_{x_p}(K_{p+1}, v))$ is contained in $\alpha(w)$. The adjacency tester checks this by searching through the entries in $\alpha(w)$ looking for an entry $(p+1, \kappa_{x_p}(K_{p+1}, v'))$ such that $I(\mu_{x_p}(K_{p+1}), \kappa_{x_p}(K_{p+1}, v'), \mu(v)) = 1$.
- y is an ancestor of x: This is symmetric to the previous case.
- x and y are not in an ancestor-descendant relationship. In this case A_x separates v and w, so v and w are not adjacent.

With a careful application of variable-length codes, we can ensure that the total length of the label $\mu(v)$ for a vertex v whose home has depth[2] p is at most $\log n + p \cdot g_3(n) + o(\log n)$, where $g_3(n)$, with $1 \leqslant g_3(n) \in o(\log n)$, is some overhead that is incurred with each step $x_{i-1}x_i$ in the path $x_0, \ldots, x_p$. In particular, there is an additive overhead of $g_3(n)$ in the length of $\mu_{x_{i-1}}(K_i)$ for each $i \in \{1, \ldots, p\}$.

Unfortunately, mixed labelling schemes still do not completely solve the problem, because the length p of the path $x_0, \ldots, x_p$ can only be upper bounded by the height of T, which may be $\Omega(n)$. To prove Theorem 1 would require that $\text{height}(T) \cdot g_3(n) = o(\log n)$. Even when $g_3(n) = 1$, this would require a tree-decomposition of height $o(\log n)$, which no variant of Theorem 3 can guarantee.

Nevertheless, this strategy gives a $((1 + o(1))\log n)$-bit adjacency labelling scheme for the class of graphs whose n-vertex members have a tree-decomposition of bounded adhesion-width and height $o(\log n / g_3(n))$ whose torsos come from a graph class $\mathcal{G}$ that has an efficient mixed labelling scheme. (The function $g_3(n)$ is determined by the mixed labelling scheme for $\mathcal{G}$.) This result is Lemma 16 in Subsection 3.4.

Short partition into skinny subtrees: To deal with the case where the tree T in the tree-decomposition $\mathcal{T}$ does not have small height, we partition the vertices of T into a collection of subtrees such that each root to leaf path of T intersects at most $1 + \log_b n$

[2]The *depth* of a node x in a rooted tree is the number of edges on the path from x to the root of the tree. The *height* of the tree is the maximum depth of a node of the tree.

of these subtrees, for a carefully chosen value of b. This is done in such a way that each subtree in the decomposition, rooted at its node closest to the root of T, is *b-skinny*: it has at most b nodes at depth i (measured from its own root), for each integer $i \geqslant 0$. This partition is described in Corollary 10 in Subsection 2.6. By treating each subtree as a single giant bag, this gives a tree-decomposition $\mathcal{T}'$ of G whose height is at most $\log_b n$, and in which each torso of $\mathcal{T}'$ has a b-skinny tree-decomposition whose bags are bags of $\mathcal{T}$. In particular, the torsos of this b-skinny tree-decomposition belong to the original class admitting an efficient mixed labelling scheme. Note that at this point we do not yet know how to label the (typically large) bags of $\mathcal{T}'$ themselves. That is the purpose of the next step.

Skinny tree-decompositions: The last ingredient needed to complete the proof is a method of handling graphs that have b-skinny tree-decompositions whose torsos belong to a graph class that admits an efficient mixed labelling scheme. To use our solution for short tree-decompositions, we must show that such graphs have an efficient mixed labelling scheme. To do this, we group the nodes of the tree-decomposition T into layers $L_1, \ldots, L_p$, where L_i contains all nodes of depth $i-1$. For each $i \in \{1, \ldots, p\}$, a mixed labelling μ_i is computed for the pair of graphs $\bigcup_{x \in L_i}(G\langle \mathcal{T}, B_x\rangle - A_x)$ and $\bigcup_{x \in L_i}(G[B_x] - A_x)$. We then reuse ideas of Gavoille and Labourel [20] and Dujmović et al. [15] to handle adjacency testing between vertices in different homes and extend these to provide the remaining functionality (clique labels, local identifiers, and identity testing) required for a mixed labelling scheme. The resulting mixed labelling scheme has labels of length $\log n + O(\log b) + o(\log n)$.

Closing the loop: Putting the preceding pieces together with $g_3(n) = O(\sqrt{\log n})$ and setting $b = 2^{(\log n)^{3/4}}$ shows that any proper minor-closed graph class has an $f(n)$-bit adjacency labelling scheme where

$$f(n) := \log n + (\log_b n) \cdot O(\sqrt{\log n}) + O(\log b) = \log n + O((\log n)^{3/4}) \ .$$

Without much extra work, the adjacency-labelling solution for short tree-decompositions can be extended to provide an efficient mixed labelling scheme. Since this only relies on the underlying tree-decomposition having bounded adhesion-width and having torsos in a class $\mathcal{G}$ that has an efficient mixed labelling scheme, our main result is thus described concisely by the following theorem:

Theorem 4. *Let $\mathcal{G}$ be a hereditary graph class closed under taking disjoint union and admitting an efficient weighted mixed labelling scheme. Then, for any fixed integer $k \geqslant 0$, the class of graphs that have a tree-decomposition of adhesion-width at most k whose torsos belong to $\mathcal{G}$ also admits an efficient weighted mixed labelling scheme.*

We can now say what the weights are for. For the recursion behind this theorem, plain mixed labelling schemes are not enough. We need a refinement in which each vertex v carries a positive weight $\omega(v)$ and vertex labels satisfy the Kraft-type bound $|\mu(v)| \leqslant \log \omega(V(G)) - \log \omega(v) + o(\log n)$. Intuitively, weights let heavier vertices receive shorter labels; when recursing on a tree-decomposition, inflating the weight of a vertex according to the total weight hanging below it makes the label lengths telescope along root-to-leaf

paths. This notion of a *weighted mixed labelling scheme* (and the accompanying notion of *efficiency*) is defined formally in Section 3.

A quantitative version of Theorem 4 appears as Lemma 17 in Subsection 3.5.

1.3 Paper Organization

The remainder of the paper is organized as follows:

Section 2 reviews some basic background material and shows how to partition any tree-decomposition into b-skinny subtrees so that contracting each subtree yields a tree-decomposition of height at most $\log_b n$, as outlined above. Section 3 formally defines weighted mixed labelling schemes and proves Theorem 4. We conclude in Section 4 by explaining how the constructive nature of Theorem 1 leads to a polynomial-time labelling algorithm.

2 Building blocks

Let $\mathbb{N}$ be the set of all nonnegative integers, and let $\mathbb{N}^+$ be the set of all positive integers. For $p \in \mathbb{N}^+$, let $[p] := \{1, \dots, p\}$.

A *graph class* is a set of graphs closed under isomorphism. A graph class $\mathcal{G}$ is *hereditary* if for every G in $\mathcal{G}$ every induced subgraph of G is also in $\mathcal{G}$. A graph class $\mathcal{G}$ is *monotone* if for every $G \in \mathcal{G}$, every subgraph of G is in $\mathcal{G}$. A *disjoint union* of two graphs G and H is a graph obtained from two vertex-disjoint copies of G and H with no edge in between. A class of graphs $\mathcal{C}$ is *closed under taking disjoint union* if for all G, H in $\mathcal{C}$, the disjoint union of G and H is also in $\mathcal{C}$.

If G is a graph, P is a path and $(P, (B_x \mid x \in V(P)))$ is a tree-decomposition of G, then $(P, (B_x \mid x \in V(P)))$ is called a *path-decomposition* of G, which we usually denote as $(B_1, \dots, B_{|V(P)|})$.

2.1 Trees and Forests

For a rooted tree T with root $r \in V(T)$ and a node x of T, define $P_T(x)$ to be the path in T from x to r. A *forest* F is a graph (possibly disconnected) each of whose components is a tree. If each component of a forest F is a rooted tree, then F is a *rooted* forest. For a node x in a rooted forest F, let $P_F(x) := P_T(x)$ where T is the component of F that contains x.

Let F be a rooted forest. For each $x \in V(F)$, the *F-depth* of x, denoted $\operatorname{depth}_F(x)$, is the number of edges in $P_T(x)$. The *height* of F, denoted $\operatorname{height}(F) := \max_{x \in V(F)} \operatorname{depth}_F(x)$, is the maximum F-depth of a vertex in F. For each $y \in V(F)$ and each $x \in V(P_F(y))$ (including y), the vertex x is an *F-ancestor* of y and y is a *F-descendant* of x. For each $x \in V(F)$, F_x denotes the subtree of F induced by all F-descendants of x. For each edge xy of F where x is an F-ancestor of y, we say that x is the *F-parent* of y and y is an *F-child* of x. We say that a set $S \subseteq V(F)$ is an *F-antichain*, if there are no distinct $x, y \in S$ such that x is a F-ancestor of y. (When there is no danger of ambiguity, we may drop the leading F- from F-depth, F-ancestor, F-descendant, F-parent, and F-child.) We treat each subgraph $F' \subseteq F$, as a rooted forest in which each component T of F' is rooted at the node of T having minimum F-depth. For each $i \in \mathbb{N}^+$, the *i-th layer* of F is $L_i(F) := \{x \in V(F) : \operatorname{depth}_F(x) = i - 1\}$. Thus,

$L_1(F)$ is the set of roots of components of F.

For a rooted tree T and any set $S \subseteq V(T)$, the *lowest common T-ancestor* of S, denoted $\mathrm{lca}_T(S)$, is the node in $\bigcap_{x\in S} V(P_T(x))$ of maximum T-depth.

2.2 Tree-Decompositions and Forest-Decompositions

It will be convenient to work with forest-decompositions, which are the natural generalization of a tree-decomposition that allow the graph indexing the bags to be a forest. Specifically, $(F,(B_x \mid x \in V(F))$ is a *forest-decomposition* of a graph G if F is a forest, for each edge vw of G there exists $x \in V(F)$ with $v,w \in B_x$, and for each vertex v of G, $F[\{x \in V(F) \mid v \in B_x\}]$ is connected. A forest-decomposition is *rooted* if it is indexed by a rooted forest. A rooted forest-decomposition $(F,(B_x \mid x \in V(F))$ of a graph G is *tidy* if (1) $B_x \neq \emptyset$ for each $x \in V(F)$; (2) $B_y \not\subseteq B_x$ for each $y \in V(F)$ with an F-parent x; and (3) $B_z \cap B_y \not\subseteq B_y \cap B_x$ for each $z \in V(F)$ with F-parent y and F-grandparent x. Any rooted forest-decomposition $\mathcal{F} := (T,(B_x \mid x \in V(T)))$ can be made into a rooted tidy forest-decomposition by (1) removing every node x of T with $B_x = \emptyset$, (2) removing every edge xy of T with $B_x \cap B_y = \emptyset$, and (3) repeatedly replacing an edge yz with the edge xz if $B_z \cap B_y = B_x \cap B_y$ and x is the parent of y. This transformation has the properties expressed in the following observation:

Observation 5. *Let G be a graph and let $(F,(B_x \mid x \in V(F)))$ be a forest-decomposition of G. Then there exists a rooted forest F' with $V(F') \subseteq V(F)$, $\mathrm{height}(F') \leqslant \mathrm{height}(F)$, such that $\mathcal{F}' := (F',(B_x \mid x \in V(F')))$ is a rooted tidy forest-decomposition of G and every torso of $\mathcal{F}'$ is a torso of $\mathcal{F}$.*

We make use of the fact that rooted tidy forest-decompositions remain tidy when we remove the root bags and their contents. More precisely, let $\mathcal{F} := (F,(B_x \mid x \in V(F)))$ be a rooted tidy forest-decomposition of a graph G, let R be the set of roots in F, let $B_R := \bigcup_{r\in R} B_r$, and let $C_x := B_x \setminus B_R$ for each $x \in V(F)$. Then $(F-R,(C_x \mid x \in V(F-R)))$ is a rooted tidy rooted forest-decomposition of $G - B_R$.

2.3 Weight Functions

For a non-empty set S, a *weight function* over S is a function $\omega : S \to \mathbb{R}^+$. For $X \subseteq S$, we use the shorthand $\omega(X) := \sum_{x\in X} \omega(x)$. When $S := V(G)$ is the vertex-set of a graph G and H is a subgraph of G, we use the shorthand $\omega(H) := \omega(V(H))$.

Let $\omega : S \to \mathbb{R}^+$ be a weight function. For each $x \in S$, the *(Kraft) ideal codeword length* for x is $\log \omega(S) - \log \omega(x)$. It will be convenient to place some upper bound on the Kraft ideal codeword length or, equivalently, on the ratio of total weight, $\omega(S)$ to minimum weight $\min_{x\in S}(\omega(x))$. The following observation gives a way to do this without increasing the ideal codeword length of each $x \in S$ by more than a constant:

Observation 6. *For every weight function $\omega : S \to \mathbb{R}^+$, there exists a weight function $\omega'(S) \to \mathbb{N}^+$ such that $\log \omega'(S) - \log \omega'(x) \leqslant \min\{\log|S|,\ \log \omega(S) - \log \omega(x)\} + 2$, for each $x \in S$.*

Proof. By dividing every weight by $\min_{x\in S} \omega(x)$ we may assume, without loss of generality, that $\omega(x) \geqslant 1$ for each $x \in S$, since this does not change the value of the right-hand side

in the inequality. For each $x \in S$, define $\omega'(x) := \lceil \max\{\omega(S)/|S|,\ \omega(x)\} \rceil$. Then $\omega'(S) < \sum_{x \in S} (\omega(S)/|S| + \omega(x) + 1) = 2\omega(S) + |S| \leqslant 3\omega(S)$. For each $x \in S$, the inequality $\omega'(x) \geqslant \omega(x)$ implies that $\log \omega'(S) - \log \omega'(x) \leqslant \log \omega(S) - \log \omega(x) + \log 3$ and the inequality $\omega'(x) \geqslant \omega(S)/|S|$ implies that $\log \omega'(S) - \log \omega'(x) \leqslant \log \omega(S) - \log(\omega(S)/|S|) + \log 3 = \log |S| + \log 3$. □

2.4 Shannon and Shannon–Fano–Elias Codes

A *code* for a set S is an injective function $\lambda : S \to \{0,1\}^*$ that maps each element in S to a distinct binary string. A code is *prefix-free* if $\lambda(x)$ is not a prefix of $\lambda(y)$ for all distinct $x, y \in S$.

A rooted tree T is a *binary tree* if each node of T has at most two children among which at most one is the *left child* and at most one is the *right child*. A binary tree T is *full* if for every node x of T, either x has two children or x has no children. A full binary tree T is *complete* if all the leaves of T have the same depth in T. Let T be a binary tree. There is a natural ordering, called *left-to-right ordering*, of the leaves of T, such that for every node x, if x has two children then all the leaf descendants of the left child of x precede all the leaf descendants of the right child of x.

The following result is a variant of Alphabetic Binary Trees. Similar results (with the constant 3 replaced by 2) are well-known and there are several proofs [23, 34, 36]. We provide a proof of the weaker statement here because it illustrates a simple but powerful technique—*simulating weight by multiplicity*—that we will use again later.

Lemma 7. *Let S be a linearly ordered set and let $\omega : S \to \mathbb{N}^+$. Then there exists a binary tree T such that S is the set of leaves of T, the linear order on S agrees with the left-to-right-order of the leaves in T, and for each $x \in S$*

$$\mathrm{depth}_T(x) \leqslant \log \omega(S) - \log \omega(x) + 3 \ .$$

Proof. Let $h := \lceil \log \omega(S) \rceil$. Let Q be the complete binary tree of height h, so with 2^h leaves. For each $x \in S$, define the $\omega(x)$-element set

$$S_\omega(x) := \{(x,1), (x,2), \ldots, (x, \omega(x))\}$$

and let $S_\omega := \bigcup_{x \in S} S_\omega(x)$. Then $|S_\omega| = \omega(S)$. Treat S_ω as a linearly ordered set by using lexicographic order and rename the leftmost $\omega(S)$ leaves of Q, in order, with elements of S_ω.

Now for each $x \in S$, we are going to fix a node $r(x)$ in Q such that (1) all the leaves of $Q_{r(x)}$ lie in $S_\omega(x)$; and (2) $Q_{r(x)}$ contains at least $\omega(x)/4$ leaves.

Let $x \in S$. If $\omega(x) = 1$ then define $r(x) := (x,1)$, which is a leaf of Q. Clearly, (1) and (2) hold. Now suppose that $\omega(x) \geqslant 2$. Consider the lowest common Q-ancestor y of $S_\omega(x)$. Thus, y is an internal node of Q and both children of y contain in their subtree a leaf in $S_\omega(x)$. Since $S_\omega(x)$ is a set of consecutive leaves in Q, all leaves of the left child y' of y in Q that are in $S_\omega(x)$ are rightmost in $Q_{y'}$ and all leaves of the right child y'' of y in Q that are in $S_\omega(x)$ are leftmost in $Q_{y''}$. Let z be the child of y in Q such that Q_z has at least $\omega(x)/2$ leaves in $S_\omega(x)$ and let S_z be the set of all such leaves. Thus, $|S_z| \geqslant \omega(x)/2$. Since

the argument is completely symmetric in these cases, assume that z is the left child of y. Consider a path in Q_z starting from z and always taking an edge to the right child of the current node until we finish at a leaf. Let v be the first vertex of that path such that Q_v has all leaves in S_z. If $v = z$ then we set $r(x) := v$. Clearly (1) and (2) hold. Suppose that $v \neq z$. Let u be the parent of v. Since the set of leaves of Q_u form an interval of rightmost leaves of Q_z and since Q_u contains a leaf not in S_z, we conclude that Q_u contains all nodes in S_z as leaves. Therefore, Q_v contains at least $|S_z|/2$ vertices of S_z as leaves. We set $r(x) := v$ and again (1) and (2) hold.

Note that the set $\{r(x) \mid x \in S\}$ is a Q-antichain. Indeed, if $x, y \in S$, $x \neq y$ and $r(x)$ is a Q-ancestor of $r(y)$, then the leaves of $Q_{r(y)}$ are contained in the leaves of $Q_{r(x)}$ which by (1) implies that $Q_{r(x)}$ contains all nodes in $S_\omega(y)$ as leaves. However, again by (1) all the leaves of $Q_{r(x)}$ lie in $S_\omega(x)$ which is disjoint from $S_\omega(y)$, a contradiction.

Let $x \in S$. Since $Q_{r(x)}$ contains at least $\omega(x)/4$ leaves, $\text{height}(T_{r(x)}) \geqslant \log(\omega(x)/4) = \log\omega(x) - 2$. Finally,

$$\begin{aligned}\text{depth}_Q(r(x)) &\leqslant \text{height}(Q) - \text{height}(Q_{r(x)}) \\ &\leqslant \lceil \log\omega(S) \rceil - (\log\omega(x) - 2) \\ &\leqslant \log\omega(S) - \log\omega(x) + 3 \ .\end{aligned}$$

Let $T := \bigcup_{x \in S} P_Q(r(x))$. Since $\{r(x) \mid x \in S\}$ is a Q-antichain, we conclude that $\{r(x) \mid x \in S\}$ is the set of all leaves of T. Identifying each leaf r_x of T with x for each $x \in S$ gives the desired tree. □

In coding theory, Lemma 7 is the basis of the Shannon-Fano-Elias coding scheme, as we now explain. By assigning each edge e of a binary tree T a 0 or 1 depending on whether e joins a parent to its left or right child, we obtain an encoding of each root-to-leaf path in T as a binary string. Doing this with the tree from Lemma 7, we obtain the following.

Corollary 8. *Let S be a linearly ordered set and let $\omega : S \to \mathbb{N}^+$. Then there exists a code $\rho : S \to \{0,1\}^\star$ for S such that:*

(a) $|\rho(x)| \leqslant \log\omega(S) - \log\omega(x) + 3$ for each $x \in S$, and
(b) $\rho(x)$ is lexicographically less than $\rho(y)$ for each $x, y \in S$ with $x < y$ in S.

2.5 On Multipart Labels

In many cases, a label (a bitstring) will have a variable number of parts (also bitstrings), of varying lengths. We write this as $s := \langle s_1, \ldots, s_p \rangle$, where each of $s_1, \ldots, s_p$ is a (possibly empty) bitstring. The concatenation of $s_1, \ldots, s_p$, has length

$$|s| = |s_1, \ldots, s_p| = \textstyle\sum_{i=1}^{p} |s_p| \ .$$

This ignores the important issue that, in order for a decoder to extract the individual parts $s_1, \ldots, s_p$ it must know the number of parts, p, and (at least $p-1$ of) the lengths $|s_1|, \ldots, |s_p|$. We handle this by encoding the value of p and the lengths $|s_1|, \ldots, |s_p|$ and prepending these to s. This is most easily done using Elias' code $\gamma : \mathbb{N}^+ \to \{0,1\}^*$ for positive integers, in

which each $x \in \mathbb{N}^+$ is encoded by a self-delimiting binary string $\gamma(x)$ with $|\gamma(x)| \leqslant 2\lfloor \log x \rfloor + 1$ [19].[3] In our application s_i may have zero length and p may even be equal to zero, so that the codeword for each $x \in \mathbb{N}$ is $\gamma(x+1)$ and has length $2\lfloor \log(x+1) \rfloor + 1$. In this way, we obtain a self-delimiting encoding of $s_1, \ldots, s_p$ as

$$\langle s_1, \ldots, s_p \rangle := \gamma(p+1), \gamma(|s_1|+1), \ldots, \gamma(|s_p|+1), s_1, \ldots, s_p \ ,$$

which has length

$$|\langle s_1, \ldots, s_p \rangle| = \textstyle\sum_{i=1}^{p} |s_i| + O(\hat{\lg} p) + O(\sum_{i=1}^{p} \hat{\lg} |s_i|) \ , \tag{1}$$

where $\hat{\lg} x := \log \max\{2, x\}$ (just so that $\hat{\lg}(x)$ is defined and $\hat{\lg}(x) \geqslant 1$ for each $x \in \mathbb{N}$). The bits of $\langle s_1, \ldots, s_p \rangle$ that come from $s_1, \ldots, s_p$ are called the *payload* bits and the remaining bits are called the *bookkeeping* bits of $\langle s_1, \ldots, s_p \rangle$. In our setting it will always be the case that $p \in o(\log n)$ and we will use Observation 6 to ensure that $|s_i| \in O(\log n)$ for each $i \in [p]$. Thus, the number of bookkeeping bits in $\langle s_1, \ldots, s_p \rangle$ is $O(p \log \log n)$.

In some cases, we include one or more non-negative integers as part of a multipart label. When we include $d \in \mathbb{N}$ as part of a label this way, we are actually including the $\lceil \log(d+1) \rceil$-bit binary representation $\mathrm{bin}(d)$ of d. For the sake of readability, we write this as $\langle s_1, d, s_2, \ldots \rangle$ rather than $\langle s_1, \mathrm{bin}(d), s_2, \ldots \rangle$.

2.6 Separators with Low Pathwidth and Low Path-Adhesion-Width

Recall that for $b \in \mathbb{R}^+$, a rooted tree T is b-skinny if $|L_i(T)| \leqslant b$ for each $i \in \mathbb{N}^+$, and a rooted tree-decomposition $(T, (B_x \mid x \in V(T)))$ of a graph G is b-skinny if T is b-skinny. The following lemma shows how to construct a $(1/b)$-separator using the bags in a b-skinny subtree of a tree-decomposition.

Lemma 9. *Let $b > 1$, let T be a rooted tree, let $\omega : V(T) \to \mathbb{R}$, and let*

$$X := \{x \in V(T) \mid \omega(T_x) > \omega(T)/b\} \ .$$

Then:

- *(a) $T[X]$ is connected and contains the root of T;*
- *(b) $|X \cap L_i(T)| < b$, for each $i \in \mathbb{N}^+$; and*
- *(c) $\omega(C) \leqslant \omega(T)/b$ for each component C of $T - X$.*

Proof. Let r be the root of T. Since $b > 1$, we have that $\omega(T_r) = \omega(T) > \omega(T)/b$. For $x \in X \setminus \{r\}$, let y be the T-parent of x, then $\omega(T_y) \geqslant \omega(y) + \omega(T_x) > \omega(T_x) > \omega(T)/b$. Therefore $r \in X$ and, for each $x \in X \setminus \{r\}$, X contains the T-parent of x. Thus $T[X]$ is connected. This proves (a).

For each $i \in \mathbb{N}^+$, the subtrees in $\{T_x \mid x \in X \cap L_i(T)\}$ are pairwise vertex-disjoint, so

$$\omega(T) \geqslant \sum_{x \in X \cap L_i(T)} \omega(T_x) > \sum_{x \in X \cap L_i(T)} \omega(T)/b = |X \cap L_i(T)| \cdot \omega(T)/b.$$

Rewriting this inequality gives $|X \cap L_i(T)| < b$ for each $i \in \mathbb{N}^+$. This proves (b).

[3] For each $x \in \mathbb{N}^+$, $\gamma(x)$ consists of $\lfloor \log x \rfloor$ 0 bits, followed by the binary representation of x (which begins with a 1 bit and has length $\lfloor \log x \rfloor + 1$).

By (a), each component C of $T-X$ is a subtree of T that is rooted at some node y whose T-parent is in X. Since $y \notin X$, $\omega(C) = \omega(T_y) \leqslant \omega(T)/b$. This proves (c). □

Corollary 10. *For every $b > 1$, every $n \in \mathbb{N}^+$, every n-vertex graph G and every tree-decomposition $\mathcal{T} := (T, (B_x \mid x \in V(T)))$ of G, there exists a tree-decomposition $\mathcal{Q} := (Q, (D_x \mid x \in V(Q)))$ of G such that*

(a) *each adhesion of $\mathcal{Q}$ is an adhesion of $\mathcal{T}$;*
(b) *for each $q \in V(Q)$, $G\langle \mathcal{Q}, D_q \rangle$ has a b-skinny tree-decomposition $(T(q), (B_y \mid y \in V(T(q))))$ where $T(q)$ is a subtree of T; and*
(c) $\text{height}(Q) \leqslant \log_b n$.

Proof. Fix $b > 1$. Let G be an n-vertex graph, let $\mathcal{T} := (T, (B_x \mid x \in V(T)))$ be a rooted tree-decomposition of G, and let r be the root of T. We prove a slightly stronger statement that involves a set $A \subseteq B_r$. The proof is by induction on $|V(G-A)|$. To make the induction work we strengthen (c) to:

(c') $\text{height}(Q) \leqslant \max\{0, \log_b(|V(G-A)|)\}$

Define the function

$$\omega(y) := \begin{cases} |B_r \setminus A| & \text{if } y = r \\ |B_y \setminus B_x| & \text{if } y \in V(T) \setminus \{r\} \text{ has } T\text{-parent } x. \end{cases}$$

Observe that $\sum_{y \in V(T)} \omega(y) = |V(G-A)|$. Let $N := |V(G-A)|$.

First we consider the base case. If $N \leqslant b$, let Q be a tree that has only one node, s. Let $X = \{r\}$ if $N = 0$, and $X = \{x \in V(T) \mid \omega(T_x) > 0\}$ otherwise. Then $T[X]$ is a subtree of T containing r. For any $i \in \mathbb{N}^+$, the subtrees T_x for $x \in X \cap L_i(T)$ are disjoint, so $|X \cap L_i(T)| \leqslant \sum_{x \in X \cap L_i(T)} \omega(T_x) \leqslant \omega(T) = N \leqslant b$. Thus, $T[X]$ is b-skinny. Let $D_s := \bigcup_{x \in X} B_x$ and let $\mathcal{Q} := (Q, (D_q \mid q \in V(Q)))$. For each $y \in N_T(X)$, $\omega(T_y) = 0$, so $V(G_y - A_y) = \emptyset$, which implies $B_z \subseteq A_y \subseteq B_x$ for all $z \in V(T_y)$, where $x \in X$ is the T-parent of y. Hence $D_s = \bigcup_{y \in V(T)} B_y$, meaning $\mathcal{Q}$ is a tree-decomposition of G with no adhesions, so (a) holds vacuously. Since $G\langle \mathcal{Q}, D_s \rangle = G$, we take $T(s) = T[X]$ and then $(T(s), (B_y \mid y \in V(T(s))))$ is a b-skinny tree-decomposition of $G\langle \mathcal{Q}, D_s \rangle$, so (b) holds. Since $V(Q) = \{s\}$ and $\text{depth}_Q(s) = 0$, condition (c') holds because $\text{height}(Q) = 0 \leqslant \max\{0, \log_b(N)\}$.

Now assume $N > b$. Let $X := \{x \in V(T) \mid \omega(T_x) > N/b\}$. By Lemma 9, $T[X]$ is a b-skinny subtree of T that includes r. The root of the tree Q will be a node s and $D_s := \bigcup_{x \in X} B_x$.

For each $y \in N_T(X)$, let $G_y := G[\bigcup_{z \in V(T_y)} B_z]$ and let $A_y := B_y \cap B_x$ be the parent adhesion of y in $\mathcal{T}$, where x is the T-parent of y. Then $\mathcal{T}_y := (T_y, (B_z \mid z \in V(T_y)))$ is a tree-decomposition of G_y and every adhesion of $\mathcal{T}_y$ is an adhesion of $\mathcal{T}$. Observe that $|V(G_y - A_y)| = \omega(T_y) \leqslant N/b$. Apply the inductive hypothesis to G_y with tree-decomposition $\mathcal{T}_y$ and special set A_y to obtain a tree-decomposition $\mathcal{Q}_y = (Q_y, (D_q \mid q \in V(Q_y)))$ of G_y. By induction, each bag of $\mathcal{Q}_y$ has a tree-decomposition formed by a (skinny) subtree of T. Therefore, some bag D_{r_y} of $\mathcal{Q}_y$ contains all the vertices of B_y. Make r_y a child of s in the tree Q.

We now show that $\mathcal{Q}$ is a tree-decomposition of G that satisfies the requirements of the lemma.

(a): For each $y \in N_T(X)$, the adhesion between s and r_y is A_y, which is an adhesion of $\mathcal{T}$. Furthermore, each adhesion of $\mathcal{T}_y$ is an adhesion of $\mathcal{T}$, so each adhesion of $\mathcal{Q}_y$ is an adhesion of $\mathcal{T}$. Thus every adhesion of $\mathcal{Q}$ is an adhesion of $\mathcal{T}$.

(b): $\mathcal{T}_X := (T[X], (B_y \mid y \in V(T[X])))$ is a b-skinny tree-decomposition of $G\langle \mathcal{Q}, D_s \rangle$. For each $y \in N_T(X)$, $G\langle \mathcal{Q}_y, D_{r_y} \rangle$ has a tree-decomposition whose tree is a b-skinny subtree of T_y. The tree-decomposition of $G_y\langle \mathcal{Q}_y, D_{r_y} \rangle$ contains B_y, therefore this tree-decomposition is also a tree-decomposition of $G\langle \mathcal{Q}, D_{r_y} \rangle$. By induction, every other bag of $\mathcal{Q}_y$ has a tree-decomposition whose tree is a b-skinny subtree of T. Thus, each torso of $\mathcal{Q}$ has a b-skinny tree-decomposition whose tree is a subtree of T.

(c'): For each $y \in L_2(Q)$, the inductive hypothesis implies that $\text{height}(Q_y) \leqslant \max\{0, \log_b(|V(G_y - A_y)|)\}$. Since $|V(G_y - A_y)| = \omega(T_y) \leqslant N/b = |V(G-A)|/b$,

$$\text{height}(Q_y) \leqslant \max\{0, \log_b(|V(G-A)|/b)\} = \max\{0, \log_b |V(G-A)| - 1\}.$$

Thus, $\text{height}(Q) = 1 + \max\{\text{height}(Q_y) : y \in L_2(Q)\} \leqslant \max\{1, \log_b |V(G-A)|\} = \log_b |V(G-A)|$ (since $|V(G-A)| > b \geqslant 1$). □

3 Weighted Mixed Labelling Schemes – Proof of Theorem 4

This section defines weighted mixed labelling schemes and proves our main technical result, namely Theorem 4. Let $\mathcal{G}$ be a class of graphs. Let $g_i : \mathbb{N} \to \mathbb{R}^+$ for each $i \in [3]$. The class $\mathcal{G}$ admits a *weighted (g_1, g_2, g_3) mixed labelling scheme* of $\mathcal{G}$ if there exists a pair (A, I) with

- $A : (\{0,1\}^*)^2 \to \{0,1\}$ (the *adjacency tester*); and
- $I : (\{0,1\}^*)^3 \to \{0,1\}$ (the *identity tester*)

such that for every $n \in \mathbb{N}^+$, every n-vertex graph G^+ in $\mathcal{G}$, every spanning subgraph G of G^+, and every weight function $\omega : V(G) \to \mathbb{R}^+$, there exists a function $\mu : V(G) \cup \{K \mid K \text{ is a clique in } G^+\} \to \{0,1\}^*$ and a function $\kappa : \{(K,u) \mid K \text{ is a clique in } G^+, u \in K\} \to \{0,1\}^*$ such that:

(i) For all pairs of distinct cliques K and L in G^+, $\mu(K) \neq \mu(L)$;

(ii) For all $u, v \in V(G)$,

$$A(\mu(u), \mu(v)) = 1 \text{ if and only if } uv \in E(G);$$

(iii) For every clique K in G^+, $u \in K$, $v \in V(G)$,

$$I(\mu(K), \kappa(K,u), \mu(v)) = 1 \text{ if and only if } u = v;$$

(iv) for every $v \in V(G)$,

$$|\mu(v)| \leqslant \log \omega(G) - \log \omega(v) + g_1(n);$$

(v) for every clique K in G^+,

$$|\mu(K)| \leqslant \log \omega(G) - \log \min_{v \in K}(\omega(v)) + g_3(n);$$

(vi) for every clique K in G^+ and $u \in K$,

$$|\kappa(K,u)| \leqslant g_2(n).$$

A pair (μ,κ) that satisfies the first two conditions is a *mixed labelling* of (G^+,G) for (A,I). A pair (μ,κ) that satisfies all these conditions is a *weighted (g_1,g_2,g_3) mixed labelling* of (G^+,G,ω) for (A,I).

Here is a little intuition about these definitions. Function $g_1(n)$ measures how much the length of a vertex label exceeds the ideal value $\log\omega(G)-\log\omega(v)$ that appears in Kraft's Inequality [35]. Function $g_2(n)$ is the length of the longest local identifier. Function $g_3(n)$ measures how much the length of a clique label exceeds the Kraft-like quantity $\log\omega(G)-\log\min_{v\in K}(\omega(v))$.

The existence of a weighted $(g_1(n),\cdot,\cdot)$ mixed labelling scheme for a graph class $\mathcal{G}$ immediately implies the existence of a $(\log n+g_1(n))$-bit adjacency labelling scheme for $\mathcal{G}$. Thus, we are interested in the case where $g_1(n)\in o(\log n)$. We say that a graph class $\mathcal{G}$ admits an *efficient* weighted mixed labelling scheme if $\mathcal{G}$ admits a weighted (g_1,g_2,g_3) mixed labelling scheme with $g_i(n)\in o(\log n)$ for each $i\in[3]$. So any graph class that admits an efficient weighted mixed labelling scheme admits a $(1+o(1))\log n$-bit adjacency labelling scheme. The following lemma shows that any adjacency labelling scheme obtained this way is injective, which justifies Corollary 2.

Lemma 11. *Let (μ,κ) be a mixed labelling of some (G^+,G,ω) for some (A,I). Then the restriction $\mu_{|V(G)}$ of μ to $V(G)$ is injective.*

Proof. Suppose, for the sake of contradiction, that there exists distinct $v,w\in V(G)$ such that $\mu(v)=\mu(w)$. Since $K:=\{v\}$ is a clique in G^+, this immediately leads to the contradiction $1=I(\mu(K),\kappa(K,v),\mu(v))=I(K,\kappa(K,v),\mu(w))=0$. □

It will actually be convenient to consider a slightly weaker notion of weighted (g_1,g_2,g_3) mixed labelling schemes.

A weight function $\omega:S\to\mathbb{N}^+$ is *nice* if for every $x\in S$,

$$\log\omega(S)-\log\omega(x)\leqslant\log|S|+2\ .$$

(We emphasize that weights are required to be positive integers here, which will be important later on.) We then say that a class of graphs $\mathcal{G}$ admits a *weak weighted (g_1,g_2,g_3) mixed labelling scheme* if $\mathcal{G}$ satisfies the definition given above for weighted (g_1,g_2,g_3) mixed labelling schemes, with the exception that only nice weight functions ω are considered. The two notions are essentially equivalent, as shown by the following lemma. This will allow us to only consider nice weight functions in the proofs.

Lemma 12. *Let $\mathcal{G}$ be a class of graphs. Let $g_i:\mathbb{N}\to\mathbb{R}$ be functions for each $i\in[3]$. If $\mathcal{G}$ admits a weak weighted (g_1,g_2,g_3) mixed labelling scheme, then $\mathcal{G}$ admits a weighted (g_1+2,g_2,g_3+2) mixed labelling scheme.*

Proof. Suppose that $\mathcal{G}$ admits a weak weighted (g_1, g_2, g_3) mixed labelling scheme. Let A and I be the corresponding adjacency and identity tester, respectively. We will use the same pair (A, I) for our weighted $(g_1 + 2, g_2, g_3 + 2)$ mixed labelling scheme. Let $n \in \mathbb{N}^+$, let G^+ be an n-vertex graph in $\mathcal{G}$, let G be a spanning subgraph of G^+, and let $\omega : V(G) \to \mathbb{R}^+$ be an arbitrary weight function. Let ω' be the function obtained by applying Observation 6 to ω. Thus, for every $v \in V(G)$,

$$\log \omega'(V(G)) - \log \omega'(v) \leqslant \min\{\log n,\ \log \omega(G) - \log \omega(v)\} + 2 \ .$$

In particular, ω' is nice. Hence, by our assumption, there is a mixed labelling μ of (G^+, G, ω') for (A, I) and a local identifier κ of (G^+, G, ω') for (A, I). Hence, for every $v \in V(G)$,

$$|\mu(v)| \leqslant \log \omega'(G) - \log \omega'(v) + g_1(n) \leqslant \log \omega(G) - \log \omega(v) + g_1(n) + 2 \ ,$$

and for every clique K in G^+,

$$|\mu(K)| \leqslant \log \omega'(G) - \log \min_{v \in K}(\omega'(v)) + g_3(n) \leqslant \log \omega(G) - \log \min_{v \in K}(\omega(v)) + g_3(n) + 2 \ .$$

It follows that μ and κ are the desired mixed labelling and local identifier, respectively, of (G^+, G, ω) for (A, I). □

3.1 Adding Apex Vertices

The next lemma shows how a mixed labelling scheme can accommodate the addition of apex vertices. Recall that $\mathcal{G}^{+a}$ is the class of graphs G such that $G - X \in \mathcal{G}$ for some $X \subseteq V(G)$ with $|X| \leqslant a$.

Lemma 13. *Let $\mathcal{G}$ be a class of graphs that admits a weighted (g_1, g_2, g_3) mixed labelling scheme, where $g_i : \mathbb{N} \to \mathbb{R}$ is a non-decreasing function with $g_i(n) \in O(\log n)$ for each $i \in [3]$. Let $a \in \mathbb{N}^+$. Then $\mathcal{G}^{+a}$ admits a weighted (g_1', g_2', g_3') mixed labelling scheme, where*

$$\begin{aligned} g_1'(n) &= g_1(n) + O(a) + O(\log\log n) \ , \\ g_2'(n) &= g_2(n) + O(\log a) \ , \\ g_3'(n) &= g_3(n) + O(a) + O(\log\log n) \ . \end{aligned}$$

Proof. By Lemma 12 it is enough to show that $\mathcal{G}^{+a}$ admits a weak weighted (g_1', g_2', g_3') mixed labelling scheme. We will describe an adjacency tester A' and an identity tester I' for which we can construct a mixed labelling (μ, κ) for any (G^+, G, ω) where $G^+ \in \mathcal{G}^{+a}$, G is a spanning subgraph of G^+, and $\omega : V(G) \to \mathbb{N}^+$ is a nice weight function. Since it is not possible to describe A' and I' without knowing the contents of the labels, we first describe how to compute μ and κ for a particular (G^+, G, ω).

Let G^+ be an n-vertex graph in $\mathcal{G}^{+a}$. Thus, there is a subset $X \subseteq V(G^+)$ of at most a vertices such that $G^+ - X \in \mathcal{G}$. Let G be a spanning subgraph of G^+, and let $\omega : V(G) \to \mathbb{N}^+$ be a nice weight function. Thus, for each $v \in V(G)$,

$$\log \omega(G) - \log \omega(v) \leqslant \log n + 2 \ . \tag{2}$$

Let $n \in \mathbb{N}^+$, let G^+ be an n-vertex graph in $\mathcal{G}^{+a}$, let G be a spanning subgraph of G^+, and let $\omega : V(G) \to \mathbb{N}^+$ be a nice weight function.

The subgraph label μ^-: Let (A, I) be the pair of adjacency and identity testers for a weighted (g_1, g_2, g_3) mixed labelling scheme of $\mathcal{G}$. Let (μ^-, κ^-) be a weighted (g_1, g_2, g_3) mixed labelling of $(G^+ - X, G - X, \omega_{|V(G)})$ for (A, I). Then for each $v \in V(G - X)$,

$$\begin{aligned} |\mu^-(v)| &\leqslant \log \omega(G - X) - \log \omega(v) + g_1(|V(G - X)|) \\ &\leqslant \log \omega(G) - \log \omega(v) + g_1(n) \ , \end{aligned} \tag{3}$$

and for each clique K in $G - X$,

$$\begin{aligned} |\mu^-(K)| &\leqslant \log \omega(G - X) - \log \min_{v \in K}(\omega(v)) + g_3(|V(G - X)|) \\ &\leqslant \log \omega(G) - \log \min_{v \in K}(\omega(v)) + g_3(n) \ . \end{aligned} \tag{4}$$

Extend the domain of μ^- to include the vertices in X by defining $\mu^-(v) := \varepsilon$ (the empty string), for each $v \in X$.

The apex identifier $\sigma(u)$: Fix a linear ordering $u_1, \ldots, u_b$ of the vertices in X. By definition, $b \leqslant a$. For each $u \in V(G)$ define

$$\sigma(u) := \begin{cases} \lceil \log(b+1) \rceil \text{-bit binary representation of } i & \text{if } u \in X \text{ and } u = u_i, \\ \lceil \log(b+1) \rceil \text{ zero bits} & \text{if } u \in V(G) - X. \end{cases}$$

The vertex label $\mu(v)$: For each $v \in V(G)$, let $c(v)$ be a b-bit string whose i-th bit indicates if $v u_i \in E(G)$. For each $v \in V(G)$, define

$$\mu(v) := \langle \sigma(v), c(v), \mu^-(v) \rangle \ .$$

For each $v \in V(G)$,

$$\begin{aligned} |\sigma(v)| + b + |\mu^-(v)| &\leqslant \lceil \log(b+1) \rceil + b + \log \omega(G - X) - \log \omega(v) + g_1(n - |X|) && \text{by (3)} \\ &\leqslant \log \omega(G) - \log \omega(v) + g_1(n) + 2a \\ &\leqslant \log n + g_1(n) + 2a + 2 && \text{since } \omega \text{ is nice} \\ &\in O(a + \log n) \ , \end{aligned}$$

and therefore

$$\begin{aligned} |\mu(v)| &\leqslant |\sigma(v)| + a + |\mu^-(v)| + \mathcal{O}(\hat{\lg}|\sigma(v)| + \hat{\lg} a + \hat{\lg}|\mu^-(v)|) && \text{by (1)} \\ &\leqslant \log \omega(G) - \log \omega(v) + g_1(n) + \mathcal{O}(a + \log\log n) \ . \end{aligned}$$

Adjacency testing: We now describe the adjacency tester A'. Given the labels $\mu(v) = \langle \sigma(v), c(v), \mu^-(v) \rangle$ and $\mu(w) = \langle \sigma(w), c(w), \mu^-(w) \rangle$ for two vertices v and w of G, A' works as follows. First by checking the values of $\sigma(v)$ and $\sigma(w)$ we check if v and/or w lie in X. If at least one of them, say v, is in X, then $v = u_i$ for some $i \in [b]$ and $\sigma(v)$ encodes the value of i. In this case, we set $A'(\mu(v), \mu(w))$ to be the i-th bit of $c(w)$. If neither v nor w is in X, then $A'(\mu(v), \mu(w)) = A(\mu^-(v), \mu^-(w))$.

The clique label $\mu(K)$: For each clique K in G^+, let $c(K)$ be a b-bit string whose i-th bit indicates if $u_i \in K$. For each clique K in G^+, define

$$\mu(K) := \begin{cases} \langle c(K) \rangle & \text{if } K \subseteq X, \\ \langle c(K), \mu^-(K - X) \rangle & \text{otherwise.} \end{cases}$$

To see that μ is injective when restricted to cliques, consider two distinct cliques K and L in G^+. If $K \cap X \neq L \cap X$, then $c(K) \neq c(L)$, so $\mu(K) \neq \mu(L)$ because their first parts differ. Otherwise, $K \setminus X \neq L \setminus X$. If one is empty, say $K \subseteq X$, then $\mu(K)$ has one part while $\mu(L)$ has two parts, so $\mu(K) \neq \mu(L)$. If both are non-empty, then $\mu^-(K - X) \neq \mu^-(L - X)$ since μ^- is injective on cliques of $G^+ - X$, which implies $\mu(K) \neq \mu(L)$.

If $K \not\subseteq X$, then

$$\begin{aligned} b + |\mu^-(K - X)| &\leqslant \log \omega(G) - \log \min_{v \in K}(\omega(v)) + g_3(n - |X|) + a && \text{by (4)} \\ &\leqslant \log n + 2 + g_3(n) + a && \text{since } \omega \text{ is nice} \\ &\in O(a) + O(\log n) \ , \end{aligned}$$

and therefore

$$\begin{aligned} |\mu(K)| &\leqslant |\mu^-(K - X)| + b + O(\hat{\lg} |\mu^-(K - X)| + \hat{\lg} b) && \text{by (1)} \\ &\leqslant \log \omega(G) - \log \min_{v \in K}(\omega(v)) + g_3(n) + \mathcal{O}(a + \log\log n) \ . \end{aligned}$$

If $K \subseteq X$, then $|\mu(K)| \leqslant b + O(\hat{\lg} b)$, which easily satisfies the same bound.

The local identifier $\kappa(K, u)$: For each clique K in G^+ and each $u \in K$, define

$$\kappa(K, u) := \begin{cases} 0, \kappa^-(K - X, u) & \text{if } u \notin X, \\ 1, \sigma(u) & \text{if } u \in X. \end{cases}$$

Note that $\kappa(K, u)$ is a concatenation, so it does not incur the encoding cost from (1). Thus,

$$|\kappa(K, u)| \leqslant 1 + \max\{g_2(n), \lceil \log(a + 1) \rceil\} \leqslant g_2(n) + O(\log a) \ .$$

Identity testing: We now describe the identity tester I'. Given $\mu(K)$, $\kappa(K, u)$, and $\mu(v)$ for some clique K in G^+, $u \in K$, and $v \in V(G)$, we compute $I'(\mu(K), \kappa(K, u), \mu(v))$ as follows. First by examining $\sigma(v)$ and the first bit of $\kappa(K, u)$, the tester determines whether $v \in X$ and whether $u \in X$. If $u, v \in X$, then $I'(\mu(K), \kappa(K, u), \mu(v)) = 1$ if $\kappa(K, u) = 1, \sigma(v)$, and $I'(\mu(K), \kappa(K, u), \mu(v)) = 0$ otherwise. If $u, v \notin X$, then since $u \in K$ we have that $K \not\subseteq X$, which means $\mu(K)$ has a second part $\mu^-(K - X)$. The tester extracts this second part and sets $I'(\mu(K), \kappa(K, u), \mu(v)) = I(\mu^-(K - X), \kappa^-(K - X, u), \mu^-(v))$. Otherwise, one of u or v is in X and the other is not, so $u \neq v$ and we set $I'(\mu(K), \kappa(K, u), \mu(v)) = 0$. □

3.2 Disjoint Unions of Graphs

Lemma 14. *Let $\mathcal{G}$ be a class of graphs that admits a weighted (g_1, g_2, g_3) mixed labelling scheme, where $g_i : \mathbb{N} \to \mathbb{R}^+$ is a non-decreasing function with $g_i(n) \in O(\log n)$ for each $i \in [3]$. Let*

$\mathcal{G}'$ be the class of graphs that can be formed by taking the union of a finite number of pairwise vertex-disjoint graphs in $\mathcal{G}$. Then $\mathcal{G}'$ admits a weighted (g_1', g_2, g_3') mixed labelling scheme, where

$$g_1'(n) = g_1(n) + O(\log\log n) \ ,$$
$$g_3'(n) = g_3(n) + O(\log\log n) \ .$$

Proof. By Lemma 12 it is enough to show that $\mathcal{G}'$ admits a weak weighted (g_1', g_2, g_3') mixed labelling scheme. We will describe an adjacency tester A' and an identity tester I' for which we can construct a mixed labelling (μ, κ) for any (G^+, G, ω) where $G^+ \in \mathcal{G}'$, G is a spanning subgraph of G^+, and $\omega : V(G) \to \mathbb{N}^+$ is a nice weight function. Since it is not possible to describe A' and I' without knowing the contents of the labels, we first describe how to compute μ and κ for a particular (G^+, G, ω).

Let G^+ be an n-vertex graph in $\mathcal{G}'$, let G be a spanning subgraph of G^+, and let $\omega : V(G) \to \mathbb{N}^+$ be a nice weight function. Thus, for each $v \in V(G)$,

$$\log \omega(G) - \log \omega(v) \leqslant \log n + 2 \ . \tag{5}$$

By definition, $G^+ := \bigcup_{i=1}^{m} G_i^+$, where $G_1^+, \ldots, G_m^+$ are pairwise vertex-disjoint members of $\mathcal{G}$. Therefore, $G := \bigcup_{i=1}^{m} G_i$, where $G_i := G[V(G_i^+)]$ is a spanning subgraph of G_i^+ for each $i \in [m]$.

The subgraph label $\rho(x)$: Let $\psi : [m] \to \mathbb{N}^+$ be the weight function defined by $\psi(i) := \omega(G_i)$ and observe that $\psi([m]) = \omega(G)$ since $G_1, \ldots, G_m$ are pairwise vertex-disjoint. Apply Corollary 8 to the set $[m]$ with weight function ψ to obtain a prefix-free code $\rho : [m] \to \{0,1\}^*$ such that for each $i \in [m]$,

$$|\rho(i)| \leqslant \log \omega(G) - \log \omega(G_i) + 3 \ . \tag{6}$$

For each $i \in [m]$ and each vertex v of G_i, define $\rho(v) := \rho(i)$. For each $i \in [m]$ and each clique K of G_i^+, define $\rho(K) := \rho(i)$.

The sub-label $\mu_i(x)$: Let (A, I) be the pair of adjacency and identity testers for a weighted (g_1, g_2, g_3) mixed labelling scheme of $\mathcal{G}$. For each $i \in [m]$, let (μ_i, κ_i) be a mixed labelling of $(G_i^+, G_i, \omega_{|V(G_i)})$ for (A, I). Thus, for each $i \in [m]$ and each $v \in V(G_i)$,

$$\begin{aligned} |\mu_i(v)| &\leqslant \log \omega(G_i) - \log \omega(v) + g_1(|V(G_i)|) \\ &\leqslant \log \omega(G_i) - \log \omega(v) + g_1(n) \end{aligned} \tag{7}$$

and for each clique K in G_i^+,

$$\begin{aligned} |\mu_i(K)| &\leqslant \log \omega(G_i) - \log \min_{v \in K}(\omega(v)) + g_3(|V(G_i)|) \\ &\leqslant \log \omega(G_i) - \log \min_{v \in K}(\omega(v)) + g_3(n). \end{aligned} \tag{8}$$

For each $i \in [m]$, and each vertex v of G_i, define $\mu'(v) := \mu_i(v)$. For each $i \in [m]$, and each clique K of G_i^+, define $\mu'(K) := \mu_i(K)$.

The vertex label $\mu(v)$: Let A and I denote the adjacency and identity testers, respectively, for the weighted (g_1, g_2, g_3) mixed labelling scheme of $\mathcal{G}$. For each $i \in [m]$ and each $v \in V(G_i)$, define

$$\mu(v) := \langle \rho(v), \mu'(v) \rangle \ ;$$

thus

$$\begin{aligned} |\rho(v)| + |\mu_i(v)| &\leqslant \log \omega(G) - \log \omega(v) + 3 + g_1(n) && \text{by (6) and (7),} \\ &\leqslant \log n + 5 + g_1(n) = \mathcal{O}(\log n) && \text{by (5),} \end{aligned}$$

and therefore

$$\begin{aligned} |\mu(v)| &\leqslant |\rho(v)| + |\mu'(v)| + O(\hat{\lg}|\rho(v)| + \hat{\lg}|\mu'(v)|) && \text{by (1)} \\ &\leqslant \log \omega(G) - \log \omega(v) + g_1(n) + O(\log\log n) \ . \end{aligned}$$

Adjacency testing: We now define an adjacency tester A'. Given the labels $\mu(v) = \langle \rho(v), \mu'(v) \rangle$ and $\mu(w) = \langle \rho(w), \mu'(w) \rangle$ for two vertices $v, w \in V(G)$, we compute $A'(\mu(v), \mu(w))$ as follows. First A' verifies if $\rho(v) = \rho(w)$. If $\rho(v) \neq \rho(w)$, then v and w lie in different components of G, so they cannot be adjacent in G, and we set $A'(\mu(v), \mu(w)) := 0$. If $\rho(v) = \rho(w)$, then v and w are both contained in the graph G_i, for some $i \in [m]$. In this case $\mu(v) = \langle \rho(i), \mu'(v) \rangle$, and $\mu(w) = \langle \rho(i), \mu'(w) \rangle$. In this case, we simply set $A'(\mu(v), \mu(w)) := A(\mu'(v), \mu'(w))$.

The clique label $\mu(K)$: For each $i \in [m]$ and each clique K in G_i^+, define

$$\mu(K) := \langle \rho(K), \mu'(K) \rangle \ ;$$

To see that μ is injective when restricted to cliques, consider two distinct cliques K and L in G^+. Since G^+ is the disjoint union of $G_1^+, \ldots, G_m^+$, each clique is contained in exactly one component. If K and L are in different components G_i^+ and G_j^+, then $\rho(K) = \rho(i) \neq \rho(j) = \rho(L)$ since ρ is a prefix-free code, so $\mu(K) \neq \mu(L)$. If they are in the same component G_i^+, then $\mu'(K) = \mu_i(K) \neq \mu_i(L) = \mu'(L)$ since μ_i is injective on cliques of G_i^+, which implies $\mu(K) \neq \mu(L)$.

Thus,

$$\begin{aligned} |\rho(K)| + |\mu'(K)| &\leqslant \log \omega(G) - \log \min_{v \in K}(\omega(v)) + 3 + g_3(n) \\ &\leqslant \log n + 5 + g_3(n) = \mathcal{O}(\log n) \end{aligned}$$

and therefore

$$\begin{aligned} |\mu(K)| &\leqslant |\rho(K)| + |\mu'(K)| + O(\hat{\lg}|\rho(K)| + \hat{\lg}|\mu'(K)|) && \text{by (1)} \\ &\leqslant \log \omega(G) - \log \min_{v \in K}(\omega(v)) + g_3(n) + O(\log\log n) \ . \end{aligned}$$

The local identifier $\kappa(K, u)$: Define $\kappa(K, u) := \kappa_i(K, u)$ for each $i \in [m]$, each clique K in G_i^+, and each $u \in K$. In particular,

$$|\kappa(K, u)| \leqslant g_2(n).$$

Identity testing: Define an identity tester I' as follows. Given $\mu(K)$, $\kappa(K,u)$, and $\mu(v)$ for some clique K in G^+, some $u \in K$, and some $v \in V(G)$, we compute $I'(\mu(K),\kappa(K,u),\mu(v))$ as follows. If $\rho(K) \neq \rho(v)$, then K and v lie in different components of G, so $v \notin K$, and we set $I'(\mu(K),\kappa(K,u),\mu(v)) := 0$. If $\rho(K) = \rho(v)$, then K and v are contained in the same graph G_i for some $i \in [m]$. In this case, $\mu(K) = \langle \rho(i), \mu'(K) \rangle$, $\mu(v) = \langle \rho(i), \mu'(v) \rangle$, and $\kappa(K,u) = \kappa_i(K,u)$. In this case, we simply set $I'(\mu(K),\kappa(K,u),\mu(v)) := I(\mu'(K),\kappa(K,u),\mu'(v))$. □

3.3 Graphs with Skinny Tree-Decompositions

Lemma 15. *Let $\mathcal{G}$ be a hereditary class of graphs closed under taking disjoint union. Let $k : \mathbb{N} \to \mathbb{N}^+$ and $b : \mathbb{N} \to \mathbb{R}^+$ be functions with $k(n) \in O(\log n)$ and with $b(n) > 1$ for all n. Let $\mathcal{G}'$ be the class of graphs consisting of, for each $n \in \mathbb{N}$, every n-vertex graph G with a rooted $b(n)$-skinny tree-decomposition of adhesion-width at most $k(n)$ in which each torso is a member of $\mathcal{G}$. Let $g_i : \mathbb{N} \to \mathbb{R}$ be a non-decreasing function with $g_i(n) \in O(\log n)$ for each $i \in [3]$. Suppose that $\mathcal{G}$ admits a weighted (g_1, g_2, g_3) mixed labelling scheme. Then $\mathcal{G}'$ admits a weighted (g_1', g_2', g_3') mixed labelling scheme, where*

$$\begin{aligned} g_1'(n) &= g_1(n) + O(k(n)\log b(n) + k(n)\log\log n) \ , \\ g_2'(n) &= g_2(n) + O(\log b(n) + \log\log n) \ , \\ g_3'(n) &= g_3(n) + O(k(n) + \log\log n) \ . \end{aligned}$$

Proof. By Lemma 12 it is enough to show that $\mathcal{G}'$ admits a weak weighted (g_1', g_2', g_3') mixed labelling scheme. We will describe an adjacency tester A' and an identity tester I' for which we can construct a mixed labelling (μ, κ) for any (G^+, G, ω) where $G^+ \in \mathcal{G}'$, G is a spanning subgraph of G^+, and $\omega : V(G) \to \mathbb{N}^+$ is a nice weight function. Since it is not possible to describe A' and I' without knowing the contents of the labels, we first describe how to compute μ and κ for a particular (G^+, G, ω).

Let G^+ be an n-vertex graph in $\mathcal{G}'$, let G be a spanning subgraph of G^+, and let $\omega : V(G) \to \mathbb{N}^+$ be a nice weight function. By definition, G^+ has a b-skinny tree-decomposition $\mathcal{T} := (T, (B_x \mid x \in V(T)))$ in which each torso belongs to $\mathcal{G}$ and each adhesion has size at most k, for integers $b = \lfloor b(n) \rfloor$ and $k = k(n)$. Note that $b, k \geqslant 1$.

Let $p := \text{height}(T) + 1$. For each $i \in [p]$, let $B_i := \bigcup_{x \in L_i(T)} B_x$. Then $\mathcal{P} := (B_1, \ldots, B_p)$ is a path-decomposition of G in which each adhesion has size at most bk. Let $A_1 := \emptyset$ and, for each $i \in \{2, \ldots, p\}$, let $A_i := B_i \cap B_{i-1}$.

Let

$$G^\star := \cup \{ G^+\langle \mathcal{T}, B_x \rangle \mid x \in V(T) \} \ ,$$

and for each $i \in [p]$, let

$$\begin{aligned} G_i^\star &:= G^\star[B_i] - A_i \ , \\ G_i &:= G[B_i] - A_i \ . \end{aligned}$$

Note that for each $i \in [p]$ and each $x \in L_i(T)$, $G\langle \mathcal{T}, B_x \rangle \in \mathcal{G}$ (by definition), and also since $\mathcal{G}$ is hereditary, $G^\star[B_x] \in \mathcal{G}$. Since $\mathcal{G}$ is closed under disjoint union, $G_i^\star \in \mathcal{G}$.

The layer label ρ: Define the weight function $\psi : [p] \to \mathbb{N}^+$ by $\psi(i) := \omega(G_i)$. Since $\{B_i \setminus A_i \mid i \in [p]\}$ is a partition of $V(G)$, we have that $\psi([p]) = \sum_{i\in[p]} \psi(i) = \omega(G)$. Apply Lemma 7 and Corollary 8 to the set $[p]$ with weight function ψ, and let T_ψ and $\rho : [p] \to \{0,1\}^*$ be the resulting binary tree and code, respectively. For each $i \in [p]$ and each $v \in G_i$, define $\rho(v) := \rho(i)$, so

$$|\rho(v)| \leqslant \log \psi([p]) - \log \psi(i) + 3 = \log \omega(G) - \log \omega(G_i) + 3 \ . \tag{9}$$

Also

$$\begin{aligned} \text{height}(T_\psi) &\leqslant \log \omega(G) - \log \omega(G_i) + 3 \\ &\in O(\log n) \ . \end{aligned} \tag{10}$$

Let (A, I) be the pair of adjacency and identity testers for a weighted (g_1, g_2, g_3) mixed labelling scheme of $\mathcal{G}$. For each $i \in [p]$ let (μ_i, κ_i) be a mixed labelling of $(G_i^\star, G_i, \omega_{|V(G_i)})$ for (A, I). For each $i \in [p]$, each $v \in V(G_i)$, and each clique K in $G_i^\star$,

$$|\mu_i(v)| \leqslant \log \omega(G_i) - \log \omega(v) + g_1(n) \ , \tag{11}$$

$$|\mu_i(K)| \leqslant \log \omega(G_i) - \log \min_{v\in K}(\omega(v)) + g_3(n) \ , \tag{12}$$

and for each $u \in K$,

$$|\kappa_i(K, u)| \leqslant g_2(n) \ . \tag{13}$$

For each $i \in [p]$ and each $v \in B_i \setminus A_i$, let

$$\mu'(v) := \mu_i(v).$$

The adhesion identifier $\beta(v) = (d(v), \varphi(v))$: For each vertex v in G, let

$$\begin{aligned} a(v) &:= \min\{i \in [p] \mid v \in B_i\} \ , \\ b(v) &:= \max\{i \in [p] \mid v \in B_i\} \ , \\ \text{lca}(v) &:= \text{lca}(T_\psi, \{a(v), b(v)\}) \ , \\ d(v) &:= \text{depth}_{T_\psi}(\text{lca}(v)) \ . \end{aligned}$$

Let $C := \bigcup_{i\in[p]} A_i$ be the set of vertices of G that participate in adhesions. Let x be a node in T_ψ. Define $L_x := \{v \in C \mid \text{lca}(v) = x\}$. We claim that $|L_x| \leqslant bk$. In order to prove this, consider a vertex $v \in C$ such that $\text{lca}(v) = x$. Since $v \in C$, we have that $a(v) < b(v)$, so $\text{lca}(v) = x$ must have two children in T_ψ. Consider the largest leaf i in the subtree of the left child of x in T_ψ and the smallest leaf j in the subtree of the right child of x in T_ψ. Then, $j = i + 1$, and $a(v) \leqslant i < i + 1 \leqslant b(v)$. Therefore, $v \in B_i \cap B_{i+1}$. We conclude that $|L_x| \leqslant |B_i \cap B_{i+1}| \leqslant bk$, as desired.

For each node x in T_ψ, define $\varphi_x : L_x \to [bk]$ to be an arbitrary injective function (which exists because $|L_x| \leqslant bk$). Now, define $\varphi : C \to [bk]$ so that $\varphi(v) := \varphi_{\text{lca}(v)}(v)$ for each $v \in C$.

For each $v \in V(G)$, let

$$\beta(v) := \begin{cases} \langle d(v), \varphi(v) \rangle & \text{if } v \in C, \\ \langle \varepsilon \rangle & \text{if } v \notin C. \end{cases}$$

Clearly,

$$\begin{aligned} |\mathrm{bin}(d(v))| + |\mathrm{bin}(\varphi(v))| &\leqslant \lceil \log(\mathrm{height}(T_\psi) + 1) \rceil + \lceil \log(bk+1) \rceil \\ &\in O(\log\log n + \log(bk)) && \text{by (10)} \quad (14) \\ &= O(\log\log n + \log b), && \text{since } k(n) \in O(\log n) \quad (15) \end{aligned}$$

and

$$\begin{aligned} |\beta(v)| &\leqslant |\mathrm{bin}(d(v))| + |\mathrm{bin}(\varphi(v))| + O(1) + O(\hat{\lg}|\mathrm{bin}(d(v))| + \hat{\lg}|\mathrm{bin}(\varphi(v))|) && \text{by (1)} \\ &= O(\log\log n + \log b) \ . \end{aligned}$$

Finally, for each $i \in [p]$ and each $w \in V(G_i)$, let

$$\alpha(w) := \langle \beta(v) \mid v \in N_G(w) \cap A_i \rangle \ .$$

Note that $|N_G(w) \cap A_i| \leqslant k$; indeed, if $w \in B_x$ with $x \in V(T)$ and y is the parent of x in T, then $N_G(w) \cap A_i \subseteq B_x \cap B_y$, and $|B_x \cap B_y| \leqslant k$ since $\mathcal{T}$ has adhesion-width at most k. Therefore,

$$\begin{aligned} |\alpha(w)| &\leqslant \textstyle\sum_{v \in N_G(w) \cap A_i} |\beta(v)| + O(\hat{\lg}|N_G(w) \cap A_i|) + O(\sum_{v \in N_G(w) \cap A_i} \hat{\lg}|\beta(v)|) \\ &\in O(k\log\log n + k\log b) \ , \end{aligned} \tag{16}$$

which follows by (1) and (14).

The vertex label $\mu(v)$: For each $v \in V(G)$, define

$$\mu(v) := \langle \rho(v), \mu'(v), \alpha(v), \beta(v) \rangle \ .$$

For each $v \in V(G)$,

$$\begin{aligned} |\rho(v)| + |\mu'(v)| &\leqslant \log\omega(G) - \log\omega(v) + g_1(n) + 3 && \text{by (9) and (11)} \quad (17) \\ &\in O(\log n) \ , && \text{since } \omega \text{ is nice and } g_1 \in O(\log n). \quad (18) \end{aligned}$$

Therefore, for each $v \in V(G)$, (17), (16), and (14) imply

$$\begin{aligned} |\rho(v)| + |\mu'(v)| + |\alpha(v)| + |\beta(v)| &\leqslant \log\omega(G) - \log\omega(v) + g_1(n) + O(k\log\log n + k\log(bk)) \\ &\in O(\log n + k\log\log n + k\log b) \ , \end{aligned}$$

and

$$\begin{aligned} |\mu(v)| &\leqslant |\rho(v)| + |\mu'(v)| + |\alpha(v)| + |\beta(v)| + O(\hat{\lg}|\rho(v)| + \hat{\lg}|\mu'(v)| + \hat{\lg}|\alpha(v)| + \hat{\lg}|\beta(v)|) \\ &\leqslant \log\omega(G) - \log\omega(v) + g_1(n) + O(k\log\log n + k\log b) \ . \end{aligned}$$

Thus, μ satisfies the condition on $g_1'(n)$.

Adjacency Testing: Now that the vertex labels are defined, we can describe the adjacency testing function A'. Given the labels $\mu(v) = \langle \rho(v), \mu'(v), \alpha(v), \beta(v) \rangle$ and $\mu(w) = \langle \rho(w), \mu'(w), \alpha(w), \beta(w) \rangle$ for two vertices v and w of G, the adjacency tester A' works as follows. First, the tester compares $\rho(v)$ and $\rho(w)$:

- If $\rho(v) = \rho(w)$ then v and w are both vertices of G_i, for some $i \in [p]$, and $vw \in E(G)$ if and only if $vw \in E(G_i)$. In this case, we set $A'(\mu(v), \mu(w)) = A(\mu'(v), \mu'(w)) = A(\mu_i(v), \mu_i(w))$.

- If $\rho(v) \neq \rho(w)$ then assume without loss of generality that $\rho(v)$ is lexicographically less than $\rho(w)$. In this case $v \in V(G_i)$ and $w \in V(G_j)$ for $i = a(v) < a(w) = j$. The edge vw can only be present in G if $v \in A_j$. First the tester checks whether $\beta(v)$ is not empty to see if $v \in C$. If $\beta(v) = \langle \varepsilon \rangle$, then $v \notin C$, and $v \notin A_j$, so $A'(\mu(v), \mu(w)) = 0$.

  Otherwise, $\beta(v) = \langle d(v), \varphi(v) \rangle$. Then the length-$d(v)$ prefix of $\rho(v)$ defines the path from the root of T_ψ to $\mathrm{lca}(v)$. By definition, $\mathrm{lca}(v)$ is a T_ψ-ancestor of all integers in $[a(v), b(v)]$. In this case, the tester checks if $\rho(v)$ and $\rho(w)$ have the same prefix of length $d(v)$. If they do not, then $\mathrm{lca}(v)$ is not a T_ψ-ancestor of $j = a(w)$, so $j \notin [a(v), b(v)]$. Therefore, $v \notin B_j$, which implies $v \notin A_j$. Therefore $vw \notin E(G)$, so $A'(\mu(v), \mu(w)) = 0$.

  Suppose that $\rho(v)$ and $\rho(w)$ have the same prefix of length $d(v)$. Then $j \in [a(v), b(v)]$ and $v \in A_j$. In this case the tester inspects each item of $\alpha(w)$. Let (d_0, φ_0) be such an item and say that $(d(u_0), \varphi(u_0)) = (d_0, \varphi_0)$ for $u_0 \in N_G(w) \cap A_j$. Since $u_0 \in A_j$, we have that $j \in [a(u_0), b(u_0)]$. Therefore, both nodes $\mathrm{lca}(v)$ and $\mathrm{lca}(u_0)$ are T_ψ-ancestors of j. In other words, both $\mathrm{lca}(v)$ and $\mathrm{lca}(u_0)$ lie on the path from the root to j in T_ψ. Therefore, $\mathrm{lca}(v) = \mathrm{lca}(u_0)$ if and only if $d(v) = d(u_0)$. Furthermore, $v = u_0$ if and only if $(d(v), \varphi(v)) = (d(u_0), \varphi(u_0))$. Thus, if $(d_0, \varphi_0) = (d(v), \varphi(v))$ then the tester concludes that the corresponding neighbour $u_0 = v$, so $A'(\mu(v), \mu(w)) = 1$. Finally, if no item (d_0, φ_0) in $\alpha(w)$ satisfies $(d_0, \varphi_0) = (d(v), \varphi(v))$, then $v \notin N_G(w) \cap A_j$ so v and w are not adjacent in G, and we set $A'(\mu(v), \mu(w)) = 0$.

The clique labels $\mu(K)$ and local identifiers $\kappa(K, v)$: For each clique K in $G^\star$, let

$$j(K) := \min\{j \in [p] : K \subseteq B_1 \cup \cdots \cup B_j\} \ .$$

Fix some clique K in $G^\star$ and let $j = j(K)$. By the definition of $j(K)$, we have that $B_j \cap K \setminus A_j = K \setminus A_j$ is non-empty. Therefore $K \setminus A_j$ is assigned a label $\mu_j(K \setminus A_j)$ in the mixed labelling of $(G_j^\star, G_j, \omega_{|V(G_j)})$ for (A, I). Since $K \setminus A_j$ is non-empty, let v be an arbitrary vertex in $K \setminus A_j$. Let $x = \mathrm{lca}(v)$, so $x \in L_j(T)$ and $v \in B_x \setminus B_y$, where y is the parent of x in T (if x is the root, let $B_y = \emptyset$). Since the sets $B_z \setminus B_{z'}$ are disjoint for all nodes $z \in L_j(T)$ with parent z', the node x is uniquely determined by $K \setminus A_j$. Because K is a clique, $K \subseteq B_x$, and thus $K \cap A_j \subseteq B_x \cap B_y$. Since $\mathcal{T}$ has adhesion-width at most k, $|B_x \cap B_y| \leqslant k$. Let $c(K)$ be a k-bit string whose i-th bit indicates whether the i-th vertex of $B_x \cap B_y$ (ordered canonically, for example by their β values) belongs to K. Define the clique label

$$\mu(K) := \langle \rho(j), \mu_j(K \setminus A_j), c(K) \rangle \ .$$

To see that μ is injective when restricted to cliques, consider two distinct cliques K and L in $G^\star$. If $j(K) \neq j(L)$, their first parts differ since ρ is a prefix-free code. If $j(K) = j(L) = j$, then either $K \setminus A_j \neq L \setminus A_j$ or $K \cap A_j \neq L \cap A_j$. In the former case, $\mu_j(K \setminus A_j) \neq \mu_j(L \setminus A_j)$ since μ_j is injective on cliques of $G_j^\star$. In the latter case, $K \setminus A_j = L \setminus A_j$ implies they belong to the same bag B_x, so $c(K) \neq c(L)$. Thus $\mu(K) \neq \mu(L)$.

Then

$$|\rho(j)| + |\mu_j(K \setminus A_j)| + |c(K)| \leqslant \log \omega(G) - \log \min_{v \in K \setminus A_j} (\omega(v)) + g_3(n) + 3 + k \qquad \text{by (12)} \tag{19}$$

$$\leqslant \log \omega(G) - \log \min_{v \in K} (\omega(v)) + g_3(n) + k + 3 \tag{20}$$

$$\in O(\log n) \tag{21}$$

since ω is nice, $g_3(n) \in O(\log n)$, and $k \in O(\log n)$. Thus

$$\begin{aligned} |\mu(K)| &\leqslant |\rho(j)| + |\mu_j(K \setminus A_j)| + |c(K)| + O(\hat{\lg}|\rho(j)| + \hat{\lg}|\mu_j(K \setminus A_j)| + \hat{\lg}|c(K)|) \\ &\leqslant \log \omega(G) - \log \min_{v \in K}(\omega(v)) + g_3(n) + O(k + \log\log n) \qquad \text{by (20) and (21)} \quad . \end{aligned}$$

Thus, μ satisfies the condition on $g_3'(n)$.

Recall that the local identifier $\kappa_j(K \setminus A_j, u)$ is defined for each $u \in K \setminus A_j$. For each $u \in K$, define

$$\kappa(K, u) := \begin{cases} \langle 0, \kappa_j(K \setminus A_j, u)\rangle & \text{if } u \in B_j \setminus A_j \\ \langle 1, \beta(u)\rangle & \text{if } u \in A_j. \end{cases}$$

In the first case, $|\kappa(K, u)| \leqslant g_2(n) + O(\log\log n)$. In the second case, $|\kappa(K, u)| \in O(\log(b) + \log\log n)$. Thus, κ satisfies the requirements for $g_2'(n)$.

Identity Testing: We now describe the identity testing function I'. Suppose I' is given the labels $\mu(K)$, $\kappa(K, u)$, and $\mu(v) = \langle \rho(v), \mu'(v), \alpha(v), \beta(v)\rangle$ for some clique K in $G^\star$, some $u \in K$, and some $v \in V(G)$. The label $\mu(K)$ always has the form $\langle \rho(j), \mu_j(K \setminus A_j), c(K)\rangle$ where $j = j(K)$. The identity tester I' simply ignores the third part of $\mu(K)$ and extracts $\rho(j)$ and $\mu_j(K \setminus A_j)$. The identity tester I' first compares $\rho(j)$ and $\rho(v) = \rho(a(v))$.

- If $\rho(j)$ is lexicographically smaller than $\rho(a(v))$, then $j < a(v)$. Since $K \subseteq \bigcup_{i=1}^{j} B_i$ and $v \notin B_{j'}$ for any $j' < a(v)$, this implies that $v \notin K$. Therefore $v \neq u$, since $u \in K$. In this case $I'(\mu(K), \kappa(K, u), \mu(v)) := 0$.

- If $\rho(j) = \rho(a(v))$ then $K \setminus A_j$ is a clique in $G_j^\star$ and $v \in B_j \setminus A_j$. In this case, v can only be equal to u if $u \in K \setminus A_j$. The first part b of $\kappa(K, u)$ is a single bit that indicates if $u \in K \setminus A_j$. If $b = 1$ then $u \notin K \setminus A_j$, so $I'(\mu(K), \kappa(K, u), \mu(v)) := 0$. If $b = 0$ then $u \in K \setminus A_j$ and $\kappa(K, u) = \langle 0, \kappa_j(K \setminus A_j, u)\rangle$. In this case, $\mu'(v) = \mu_j(v)$ and we set $I'(\mu(K), \kappa(K, u), \mu(v)) := I(\mu_j(K \setminus A_j), \kappa_j(K \setminus A_j, u), \mu_j(v))$.

- If $\rho(j)$ is lexicographically larger than $\rho(a(v))$ then $K \setminus A_j$ is a clique in $G_j^\star$ and $v \in B_i \setminus A_i$ for some $i = a(v) < j$. The tester examines the first part b of $\kappa(K, u)$ to determine

if $u \in K \setminus A_j$. If $b = 0$ then $u \in B_j \setminus A_j$ and $v \in B_i \setminus A_i$, so $u \neq v$ and $I'(\mu(K), \kappa(K,u), \mu(v)) := 0$. If $b = 1$ then $u \in A_j$ and $\kappa(K,u) = \langle 1, \beta(u) \rangle$. Since $u \in A_j$, $a(u) < j \leqslant b(u)$.

In particular, $u \in A_j$, so $\mathrm{lca}(u)$ is a T_ψ-ancestor of j. Therefore, the first $d(u)$ bits of $\rho(j)$ are equal to the first $d(u)$ bits of $\rho(u)$. The first $d(u)$ bits of $\rho(u)$ describe the path from the root of T_ψ to $\mathrm{lca}(u)$. The tester then compares the first $d(u)$ bits of $\rho(j)$ to the first $d(u)$ bits of $\rho(v)$. If these are not equal, then $\mathrm{lca}(u) \neq \mathrm{lca}(v)$, so $u \neq v$ and $I'(\mu(K), \kappa(K,u), \mu(v)) := 0$.

Since $u \in A_j$, we have that $u \in C$. The tester examines $\beta(v)$ to determine if $v \in C$. If $\beta(v) = \langle \varepsilon \rangle$ then $v \notin C$, so $u \neq v$ and $I'(\mu(K), \kappa(K,u), \mu(v)) := 0$. Otherwise $v \in C$ and $\beta(v) = \langle d(v), \varphi(v) \rangle$. Since we have already verified that the first $d(u)$ bits of $\rho(j)$ equal the first $d(u)$ bits of $\rho(v)$, we have that $\mathrm{lca}(u) = \mathrm{lca}(v)$ if and only if $d(u) = d(v)$. Thus, $v = u$ if and only if $(d(u), \varphi(u)) = (d(v), \varphi(v))$. We set $I'(\mu(K), \kappa(K,u), \mu(v)) = 1$ if $(d(u), \varphi(u)) = (d(v), \varphi(v))$ and $I'(\mu(K), \kappa(K,u), \mu(v)) = 0$ otherwise. □

3.4 Graphs with Short Tree-Decompositions

Lemma 16. *Let $\mathcal{G}$ be a hereditary class of graphs closed under taking disjoint union. Let $k : \mathbb{N} \to \mathbb{N}^+$, $h : \mathbb{N} \to \mathbb{R}$ be functions with $k(n), h(n) \in O(\log n)$ and with $h(n) > 0$ for all n. Let $\mathcal{G}'$ be the class of graphs consisting of, for each $n \in \mathbb{N}$, every n-vertex graph G with a rooted tree-decomposition $\mathcal{T} := (T, (B_x \mid x \in V(T)))$ such that*

(i) *$\mathcal{T}$ has adhesion-width at most $k(n)$;*
(ii) *$\mathrm{height}(T) \leqslant h(n)$;*
(iii) *for each node x in T, the torso $G\langle \mathcal{T}, B_x \rangle$ belongs to $\mathcal{G}$.*

Let $g_i : \mathbb{N} \to \mathbb{R}^+$ be a non-decreasing function with $g_i \in O(\log n)$ for each $i \in [3]$. Suppose that $\mathcal{G}$ admits a weighted (g_1, g_2, g_3) mixed labelling scheme. Then $\mathcal{G}'$ admits a weighted (g_1', g_2', g_3') mixed labelling scheme, where

$$g_1'(n) = g_1(n) + k(n) \cdot (g_2(n) + O(\log\log n)) + h(n) \cdot (g_3(n) + O(\log\log n)) + O(\log\log n) \ ,$$
$$g_2'(n) = g_2(n) + O(\log\log n) \ ,$$
$$g_3'(n) = (h(n)+1) \cdot (g_3(n) + O(k(n) + \log\log n)) \ .$$

Proof. We will describe an adjacency tester A' and an identity tester I' for which we can construct a mixed labelling (μ, κ) for any (G^+, G, ω) where $G^+ \in \mathcal{G}'$, G is a spanning subgraph of G^+, and $\omega : V(G) \to \mathbb{R}^+$ is a weight function. Since it is not possible to describe A' and I' without knowing the contents of the labels, we first describe how to compute μ and κ for a particular input tuple.

The proof is by induction on a slightly more detailed statement. An *instance* is a tuple $(n, h, k, G^+, G, \omega, \mathcal{F})$ with the following properties: n is an integer, $h \leqslant h(n)$ is a non-negative integer, $k = k(n)$ is a positive integer, $G^+ \in \mathcal{G}'$ with $n \geqslant |V(G^+)|$, G is a spanning subgraph of G^+, $\omega : V(G) \to \mathbb{N}^+$ is a weight function with

$$\log \omega(G) - \log \omega(v) \leqslant n + 2, \tag{22}$$

for each $v \in V(G)$, and $\mathcal{F} := (F, (B_x \mid x \in V(F)))$ is a rooted tidy forest-decomposition of G^+ having adhesion-width at most $k = k(n)$ and with F of height at most h.

Let $(n, h, k, G^+, G, \omega, \mathcal{F})$ be an instance. For each $y \in V(F)$, let

$$A_y := \begin{cases} \emptyset & \text{if } y \text{ is a root in } F, \\ B_y \cap B_x & \text{if } y \text{ is not a root in } F \text{ and } x \text{ is the } F\text{-parent of } y. \end{cases}$$

We prove by an induction on h that there is a (g_1', g_2', g_3') mixed labelling (μ, κ) of (G^+, G, ω) for (A', I') satisfying the following additional properties: for each $z \in V(F)$ and each $w \in B_z \setminus A_z$,

$$\begin{aligned} |\mu(w)| \leqslant \log \omega(G) - \log \omega(w) + g_1(n) \\ + |A_z| \cdot (g_2(n) + O(\log\log n)) \\ + h \cdot (g_3(n) + O(\log\log n)) + O(\log\log n) \ , \end{aligned} \tag{23}$$

and for each clique K of G^+,

$$|\mu(K)| \leqslant \log \omega(G) - \log \min_{v \in K}(\omega(v)) + (h+1) \cdot (g_3(n) + O(k + \log\log n)) \ . \tag{24}$$

Given an integer n, an n-vertex graph $G^+ \in \mathcal{G}'$, a spanning subgraph G of G^+, and a weight function $\omega : V(G) \to \mathbb{R}^+$, let $\mathcal{T} = (T, (B_x \mid x \in V(T)))$ be a rooted tree-decomposition of G^+ of adhesion-width at most $k(n)$ and with height$(T) \leqslant h(n)$ and such that each torso of $\mathcal{T}$ is in $\mathcal{G}$. Let $\mathcal{F}_0 := (F, (B_x \mid x \in V(F)))$ be a rooted tidy forest-decomposition of G^+ (obtained by applying Observation 5 to $\mathcal{T}$). Let $(n_0, h_0, k_0, G^+, G, \omega_0, \mathcal{F}_0)$ be an instance where $n_0 := n = |V(G)|$, $h_0 := h(n)$, $k_0 := k(n)$, $\omega_0 : V(G) \to \mathbb{N}^+$ is a nice weight function obtained by applying Observation 6 to ω, $\mathcal{F}_0$ is a rooted tidy forest-decomposition obtained by applying Observation 5 to $\mathcal{T}$. (The definition of ω_0 satisfies (22), by Observation 6.) By our inductive statement we obtain labelling functions (μ, κ) for (A', I'). The labels lengths are bounded in terms of ω_0 which is enough because $\log \omega_0(G) - \log \omega_0(v) \leqslant \log \omega(G) - \log \omega(v) + 2$, which follows from Observation 6. This will complete the proof of the lemma.

We now begin the inductive proof. Let $(n, h, k, G^+, G, \omega, \mathcal{F})$ be an instance, where $\mathcal{F} := (F, (B_x \mid x \in V(F)))$. Let (A, I) be the pair of adjacency and identity testers for a weighted (g_1, g_2, g_3) mixed labelling scheme of $\mathcal{G}$. Let

$$G^\star := \cup\{G^+\langle \mathcal{F}, B_z\rangle \mid z \in V(F)\} \ .$$

By definition, $\mathcal{F}$ is a forest-decomposition of $G^\star$. By the assumptions of the lemma,

$$G^\star[B_z] = G^+\langle \mathcal{F}, B_z\rangle \in \mathcal{G} \ , \tag{25}$$

for each $z \in V(F)$. Our labelling μ will assign a label $\mu(v)$ to each vertex v of $G^\star$ and a label $\mu(K)$ to each clique K of $G^\star$. Since $G^\star$ is a supergraph of G^+, this ensures that μ assigns a label to each vertex and clique of G^+.

The root labelling μ_R: Let R be the set of roots of F. Let

$$B_R := \cup\{B_r \mid r \in R\} \ .$$

Since $\mathcal{F}$ is tidy, each bag of $\mathcal{F}$ is non-empty, so $B_R \neq \emptyset$. Let

$$G_R^\star := G^\star[B_R] \ ,$$
$$G_R := G[B_R] \ .$$

Observe that $G_R^\star \in \mathcal{G}$, since the sets in $\{B_r \mid r \in R\}$ are pairwise disjoint, $G^\star[B_r] \in \mathcal{G}$ (by (25)) for each $r \in R$, and $\mathcal{G}$ is closed under disjoint union. Let

$$\mathcal{K}_R := \{B_y \cap B_R \mid y \in L_2(F)\} \ .$$

Note that if $h = 0$ then $\mathcal{K}_R$ is empty. Since $\mathcal{F}$ is tidy, each K in $\mathcal{K}_R$ is nonempty. Note also that each K in $\mathcal{K}_R$ is a clique in $G_R^\star$. For each K in $\mathcal{K}_R$, let

$$F_K := \cup\{F_y \mid y \in L_2(F) \text{ and } B_y \cap B_R = K\} \ ,$$
$$G_K := \cup\{G[B_z] \mid z \in V(F_K)\} - B_R \ .$$

Note that for each $K \in \mathcal{K}_R$, since $\mathcal{F}$ is tidy, G_K has at least one vertex. Thus, the family of sets $\{\{B_R\}\} \cup \{V(G_K) \mid K \in \mathcal{K}_R\}$ is a partition of vertices of G.

For each $v \in B_R$, let

$$\delta(v) := k \cdot \omega(v) + \sum\{\omega(G_K) \mid K \in \mathcal{K}_R,\ v \in K\} \ .$$

Since $k \geqslant 1$, $\delta(v) \geqslant \omega(v) \geqslant 1$ for each $v \in B_R$. Clearly, for each $v \in B_R$,

$$\delta(v) \geqslant k \cdot \omega(v) \ . \tag{26}$$

Note also that for each $K \in \mathcal{K}_R$ and for each $v \in K$, we have that $\delta(v) \geqslant \omega(G_K)$, and therefore (recalling that K is non-empty)

$$\min_{v \in K} \delta(v) \geqslant \omega(G_K) \ . \tag{27}$$

Since $B_R \neq \emptyset$, we have that $\delta(B_R) \geqslant 1$. Since $\mathcal{F}$ has adhesion-width at most k, each clique in $\mathcal{K}_R$ has size at most k, so

$$\delta(B_R) = \sum_{v \in B_R} \delta(v) \leqslant k \cdot \omega(B_R) + k \sum_{K \in \mathcal{K}_R} \omega(G_K) = k \cdot \omega(G) \ . \tag{28}$$

The cliques of $G_R^\star$ in $\mathcal{K}_R$ play an important role in our labelling scheme. Each clique in $\mathcal{K}_R$ is defined by an adhesion $A_y = B_r \cap B_y$ for at least one $y \in L_2(F)$. The label $\mu_R(K)$ for a clique $K \in \mathcal{K}_R$ will be used a part of the label $\mu(v)$ for each vertex v in G_R and as part of the label $\mu(K')$ for each clique K' of $G^\star$ that contains vertices in G_R.

Altogether, $G_R^\star \in \mathcal{G}$, G_R is a spanning subgraph of $G_R^\star$, $\delta : V(G_R) \to \mathbb{N}^+$ is a weight function. Let (μ_R, κ_R) be a mixed labelling of $(G_R^\star, G_R, \delta)$ for (A, I). For each $v \in B_R$,

$$\begin{aligned} |\mu_R(v)| &\leqslant \log \delta(B_R) - \log \delta(v) + g_1(|B_R|) \\ &\leqslant \log(k \cdot \omega(G)) - \log(k \cdot \omega(v)) + g_1(n) \quad \text{by (28), (26), and since } g_1 \text{ is non-decreasing} \end{aligned}$$

$$= \log \omega(G) - \log \omega(v) + g_1(n) \tag{29}$$

$$\in O(\log n) \qquad \text{by (22) and since } g_1(n) \in O(\log n). \tag{30}$$

Let K be a clique in $G_R^\star$. We make use of two different bounds on the length of $\mu_R(K)$. If $K \in \mathcal{K}_R$:

$$|\mu_R(K)| \leqslant \log \delta(B_R) - \log \min_{v \in K}(\delta(v)) + g_3(|B_R|)$$

$$\leqslant \log \omega(G) - \log \omega(G_K) + g_3(n) + \log k \quad \text{by (28), (27) and since } g_3 \text{ is non-decreasing} \tag{31}$$

$$\in O(\log n) \qquad \text{by (22) and since } g_3(n), k(n) \in O(\log n). \tag{32}$$

For each $K \notin \mathcal{K}_R$,

$$|\mu_R(K)| \leqslant \log \delta(B_R) - \log \min_{v \in K}(\delta(v)) + g_3(|B_R|)$$

$$\leqslant \log(k \cdot \omega(G)) - \log \min_{v \in K}(k \cdot \omega(v)) + g_3(n) \quad \text{by (26), (28) and since } g_3 \text{ is non-decreasing}$$

$$= \log \omega(G) - \log \min_{v \in K}(\omega(v)) + g_3(n) \tag{33}$$

$$\in O(\log n) \qquad \text{by (22) and since } g_3(n) \in O(\log n). \tag{34}$$

Moreover, for each clique K in $G_R^\star$,

$$|\kappa_R(K, v)| \leqslant g_2(|B_R|) \leqslant g_2(n) \in O(\log n) \ , \tag{35}$$

since g_2 is non-decreasing.

The subforest labelling μ_K: Suppose that $h \geqslant 1$. Thus, $F - R$ is a non-empty rooted forest. We now describe the subproblems on which we apply induction. For each $z \in V(F - R)$, let $C_z := B_z \setminus B_R$ and note that $\bigcup_{z \in V(F_K)} C_z = V(G_K)$.

Let $K \in \mathcal{K}_R$. Define

$$G_K^\star := G^\star[\bigcup \{C_z \mid z \in V(F_K)\}] \ ,$$
$$\mathcal{F}_K := (F_K, (C_z \mid z \in V(F_K))) \ .$$

Thus, $\mathcal{F}_K$ is a rooted tidy forest-decomposition of $G_K^\star$ of adhesion-width at most k and the height of F_K is at most $h - 1$. Since $\mathcal{G}$ is hereditary and by (25),

$$G_K^\star \langle \mathcal{F}_K, C_z \rangle = G^\star[B_z] - B_R \subseteq G^\star[B_z] \in \mathcal{G} \ ,$$

for every $z \in V(F_K)$.

Altogether, $\mathcal{F}_K$ is a rooted tidy forest-decomposition of $G_K^\star$ of adhesion-width at most k and with height$(F_K) \leqslant h - 1$ and with each torso in $\mathcal{G}$, G_K is a spanning subgraph of $G_K^\star$, and the restriction $\omega_{|V(G_K)}$ of ω to the vertices of G_K is a weight function that satisfies (22). Thus, $(n, h-1, k, G_K^\star, G_K, \omega_{|V(G_K)}, \mathcal{F}_K)$ is an instance on which we can apply induction. Fix a

weighted $(g_1'(n), g_2'(n), g_3'(n))$ mixed labelling (μ_K, κ_K) of $(G_K^\star, G_K, \omega_{|V(G_K)})$ for (A', I') that is obtained from the inductive hypothesis. In addition to being a weighted $(g_1'(n), g_2'(n), g_3'(n))$ mixed labelling, (μ_K, κ_K) also satisfies (23). That is, for each $z \in V(F_K)$ and each $w \in C_z \setminus A_z$,

$$\begin{aligned} |\mu_K(w)| \leqslant \log \omega(G_K) - \log \omega(w) + g_1(n) \\ + |A_z \setminus B_R| \cdot (g_2(n) + O(\log\log n)) \\ + (h-1) \cdot (g_3(n) + O(\log\log n)) + O(\log\log n) \ . \end{aligned} \tag{36}$$

The vertex labelling $\mu : V(G) \to \{0,1\}^*$: The format of a vertex label depends on whether the vertex is in B_R or not. For each $v \in B_R$, define

$$\mu(v) := \langle \mu_R(v) \rangle \ .$$

Thus, $\mu(v)$ has just one part and by (30) this part is of length in $O(\log n)$. Therefore, for each $v \in B_R$

$$\begin{aligned} |\mu(v)| &\leqslant |\mu_R(v)| + O(1) + O(\hat{\lg}|\mu_R(v)|) && \text{by (1)} \\ &\leqslant \log \omega(G) - \log \omega(v) + g_1(n) + O(\log\log n) && \text{by (29) and (30).} \end{aligned}$$

Thus, the length of $\mu(v)$ satisfies (23) when $v \in B_R$.

Now, let $w \in V(G) \setminus B_R$. Then, $w \in B_z \setminus A_z$ for some $z \in V(F_K)$ and some $K \in \mathcal{K}_R$. Note that the existence of w implies that $h \geqslant 1$. Define

$$\begin{aligned} \alpha(w) &:= \langle \kappa_R(K, v) \mid v \in A_z \cap K, vw \in E(G) \rangle \ , \\ \mu(w) &:= \langle \mu_R(K), \mu_K(w), \alpha(w) \rangle \ . \end{aligned}$$

Thus,

$$\begin{aligned} |\alpha(w)| &\leqslant \textstyle\sum_{v \in A_z \cap K} |\kappa_R(K, v)| + O(\hat{\lg}|A_z \cap K|) + O(\sum_{v \in A_z \cap K} \hat{\lg}|\kappa_R(K, v)|) && \text{by (1)} \\ &\leqslant |A_z \cap K| \cdot g_2(n) + O(\hat{\lg} k) + |A_z \cap K| \cdot O(\hat{\lg} g_2(n)) && \text{by (35)} \\ &\leqslant |A_z \cap B_R| \cdot (g_2(n) + O(\log\log n)) + O(\log\log n) \ , \end{aligned}$$

where the last line follows from $A_z \cap K = A_z \cap B_R$ and $k(n), g_2(n) \in O(\log n)$. Also,

$$\begin{aligned} |\mu_R(K)| + |\mu_K(w)| &\leqslant \log \omega(G) - \log \omega(G_K) + g_3(n) + \log k && \text{by (31)} \\ &\quad + \log \omega(G_K) - \log \omega(w) + g_1(n) && \text{by (36)} \\ &\quad + |A_z \setminus B_R| \cdot (g_2(n) + O(\log\log n)) \\ &\quad + (h-1) \cdot g_3(n) + h \cdot O(\log\log n) \\ &\leqslant \log \omega(G) - \log \omega(w) + g_1(n) \\ &\quad + |A_z \setminus B_R| \cdot (g_2(n) + O(\log\log n)) \\ &\quad + h \cdot (g_3(n) + O(\log\log n)) + O(\log\log n) \ . \end{aligned}$$

Altogether

$$|\mu_R(K)| + |\mu_K(w)| + |\alpha(w)| \leqslant \log \omega(G) - \log \omega(w) + g_1(n)$$

$$+ |A_z| \cdot (g_2(n) + O(\log\log n))$$
$$+ h \cdot (g_3(n) + O(\log\log n)) + O(\log\log n) \ .$$

Therefore,

$$\begin{aligned}
|\mu(w)| &= |\mu_R(K)| + |\mu_K(w)| + |\alpha(w)| + O(\hat{\lg}|\mu_R(K)| + \hat{\lg}|\mu_K(w)| + \hat{\lg}|\alpha(w)|) && \text{by (1)} \\
&\leqslant \log\omega(G) - \log\omega(w) + g_1(n) \\
&\quad + |A_z| \cdot (g_2(n) + O(\log\log n)) \\
&\quad + h \cdot (g_3(n) + O(\log\log n)) + O(\log\log n) \ .
\end{aligned}$$

Thus, the length of $\mu(w)$ satisfies (23) for each $w \in V(G) \setminus B_R$.

Adjacency Testing: At this point, $\mu(v)$ is defined for each $v \in V(G)$, so we can describe the operation of the adjacency tester A'. Given the labels $\mu(v)$ and $\mu(w)$ for two vertices v and w of G, A' examines how many parts each label has. For an arbitrary vertex $u \in V(G)$, if $\mu(u)$ has one part, then $u \in B_R$, otherwise $u \in V(G_K)$ for some $K \in \mathcal{K}_R$.

- If v and w are both in B_R then $\mu(v) = \langle\mu_R(v)\rangle$ and $\mu(w) = \langle\mu_R(w)\rangle$. In this case, $vw \in E(G)$ if and only if $vw \in E(G_R)$, so the adjacency tester returns $A'(\mu(v), \mu(w)) := A(\mu_R(v), \mu_R(w))$.
- If exactly one of v or w is in B_R then, assume without loss of generality that $v \in B_R$ and $w \in V(G_K)$ for some $K \in \mathcal{K}_R$. So $\mu(v) = \langle\mu_R(v)\rangle$ and $\mu(w) = \langle\mu_R(K), \mu_K(w), \alpha(w)\rangle$. In this case, $vw \in E(G)$ if and only if $v \in K \cap N_G(w)$. Thus, $vw \in E(G)$ if and only if $v \in K$ and $\alpha(w)$ contains $\kappa_R(K, v)$. In this case, the adjacency tester iterates through each κ in $\alpha(w)$ checking if $I(\mu_R(K), \kappa, \mu_R(v)) = 1$. If so, then $\kappa = \kappa_R(K, v)$ and $vw \in E(G)$ so $A'(\mu(v), \mu(w)) := 1$. If $I(\mu_R(K), \kappa, \mu_R(v)) = 0$ for every κ in $\alpha(w)$ then $v \notin K \cap N_G(w)$, so $vw \notin E(G)$, and $A'(\mu(v), \mu(w)) := 0$.
- Otherwise, $v \in V(G_K)$ and $w \in V(G_L)$ for some $K, L \in \mathcal{K}_R$. In this case, $\mu(v) = \langle\mu_R(K), \mu_K(v), \alpha(v)\rangle$ and $\mu(w) = \langle\mu_R(L), \mu_L(w), \alpha(w)\rangle$. If $\mu_R(K) \neq \mu_R(L)$ then $K \neq L$. In this case, K (and L) separates v and w in G^+, so $vw \notin E(G)$ and $A'(\mu(v), \mu(w)) := 0$. If $\mu_R(K) = \mu_R(L)$ then, since μ is injective, $K = L$ and $v, w \in V(G_K)$. Therefore, $vw \in E(G)$ if and only if $vw \in E(G_K)$, so the adjacency tester returns $A'(\mu(v), \mu(w)) := A'(\mu_K(v), \mu_K(w))$.

The clique label $\mu(K)$: What remains is to define $\mu(K)$ and $\kappa(K, u)$ for each clique K of $G^\star$ and each $u \in K$. For each clique K of $G^\star$ such that $K \subseteq B_R$,

$$\mu(K) := \langle\mu_R(K)\rangle \ .$$

Then for each clique K in $G^\star$ such that $K \subseteq B_R$,

$$\begin{aligned}
|\mu(K)| &\leqslant |\mu_R(K)| + O(1) + O(\hat{\lg}|\mu_R(K)|) && \text{by (1)} \\
&\leqslant \log\omega(G) - \log\min_{v \in K}(\omega(v)) + g_3(n) + O(\log\log n) && \text{by (33) and (34).}
\end{aligned}$$

Now, let K be a clique of $G^\star$ that contains at least one vertex not in B_R. Then there exists exactly one $L \in \mathcal{K}_R$ such that all the vertices of $K \setminus B_R$ are contained in G_L. Then $K \setminus B_R$ is a clique in $G_L^\star$, since $G_L^\star[K \setminus B_R] = G^\star[K \setminus B_R]$. Thus, $\mu_L(K \setminus B_R)$ is defined. Since K is a clique in $G^\star$, $K \cap B_R \subseteq L$. Since L is the parent adhesion of some node in $L_2(F)$, $|L| \leqslant k$. Let $c(K)$ be a k-bit string whose i-th bit indicates whether the i-th vertex of L (ordered canonically, for example by their $\mu_R(v)$ labels) belongs to K. Define

$$\mu(K) := \langle\, \mu_R(L),\ \mu_L(K \setminus B_R),\ c(K) \,\rangle \ .$$

We now argue that μ is injective on the set of cliques of $G^\star$. Consider two distinct cliques K and P of $G^\star$. If $K, P \subseteq B_R$, then $\mu(K) = \langle \mu_R(K) \rangle \neq \langle \mu_R(P) \rangle = \mu(P)$. If $K \subseteq B_R$ and $P \nsubseteq B_R$, then $\mu(K)$ has one part and $\mu(P)$ has three parts, so $\mu(K) \neq \mu(P)$. Now suppose that $K \nsubseteq B_R$ and $P \nsubseteq B_R$. Thus K is contained in $G_L^\star$ and P is contained in $G_M^\star$ for some cliques L, M in $G_R^\star$. If $L \neq M$ then $\mu_R(L) \neq \mu_R(M)$, so $\mu(K)$ differ in their first parts. Now assume that $L = M$. If $K \cap L \neq P \cap L$ then $c(K) \neq c(P)$, so $\mu(K)$ and $\mu(P)$ differ in their third parts. Otherwise, $K \setminus L \neq P \setminus L$, so $\mu_L(K \setminus B_R)$ and $\mu_L(P \setminus B_R)$ differ by induction, so $\mu(K)$ and $\mu(P)$ differ in their second parts.

Then,

$$\begin{aligned}
|\mu(K)| &= |\mu_R(L)| + |\mu_L(K \setminus B_R)| + |c(K)| + O(1) + O(\hat{\lg}|\mu_R(L)| + \hat{\lg}|\mu_L(K \setminus B_R)| + \hat{\lg}|c(K)|) \quad \text{by (1)} \\
&\leqslant |\mu_R(L)| + |\mu_L(K \setminus B_R)| + k + O(\log\log n) \ ,
\end{aligned}$$

where the last line follows by (34) and by induction as $|\mu_L(K \setminus B_R)| \leqslant \log \omega(G_L) - \log \min_{v \in K \setminus B_R} \omega(v) + h \cdot (g_3(n) + O(k + \log\log n)) \in O(\log^2 n)$, since $h \leqslant h(n) \in O(\log n)$, $k \in O(\log n)$, and $g_3(n) \in O(\log n)$. Continuing,

$$\begin{aligned}
|\mu(K)| &\leqslant |\mu_R(L)| + |\mu_L(K \setminus B_R)| + k + O(\log\log n) \\
&\leqslant \log \omega(G) - \log \omega(G_L) + g_3(n) + |\mu_L(K \setminus B_R)| + k + O(\log\log n) \qquad \text{by (33)} \\
&\leqslant \log \omega(G) - \log \min_{v \in K \setminus B_R} (\omega(v)) + (h+1) \cdot g_3(n) + (h+1) \cdot k + (h+1) \cdot O(\log\log n) \\
&\leqslant \log \omega(G) - \log \min_{v \in K} (\omega(v)) + (h+1) \cdot (g_3(n) + O(k + \log\log n)) \ .
\end{aligned}$$

Thus, μ satisfies (24) and thus the requirement for $g_3'(n)$.

The local identifier $\kappa(K, u)$: Let K be a clique in $G^\star$, let $u \in K$, and let z be the vertex of F such that $u \in B_z \setminus A_z$. The local identifier $\kappa(K, u)$ for (A', I') has two parts,

$$\kappa(K, u) := \langle \mathrm{depth}_F(z), \zeta(K, u) \rangle \ .$$

The first part is (the binary encoding of) $\mathrm{depth}_F(z)$. We now describe the second part, $\zeta(K, u)$. If $K \subseteq B_R$, then $\zeta(K, u) := \kappa_R(K, u)$. If $K \nsubseteq B_R$ then there is exactly one clique $L \in \mathcal{K}_R$ such that $K \setminus B_R$ is contained in G_L. If $u \in B_R$ then $\zeta(K, u) := \kappa_R(L, u)$. If $u \in K \setminus B_R$ then $K \setminus B_R$ is non-empty, so $\mu_L(K \setminus B_R)$ and $\kappa_L(K \setminus B_R, u)$ are defined as part of the mixed labelling (μ_L, κ_L) of $(G_L^\star, G_L, \omega_{|V(G_L)})$ for (A', I'). Since this is a labelling for (A', I') obtained by induction,

$$\kappa_L(K \setminus B_R, u) = \langle \mathrm{depth}_{F_L}(z), \zeta_L(K \setminus B_R, u) \rangle = \langle \mathrm{depth}_F(z) - 1, \zeta_L(K \setminus B_R, u) \rangle \ . \tag{37}$$

In this case, define $\zeta(K,u) := \zeta_L(K \setminus B_R, u)$.

It follows from this definition that $\zeta(K,u)$ is a local identifier $\kappa'(K',u)$, with $K' \subseteq K$ for some mixed labelling (μ',κ') for (A,I) on a graph with at most n vertices. Therefore, $|\zeta(K,u)| \leqslant g_2(n)$. Therefore,

$$\begin{aligned}|\kappa(K,u)| &\leqslant \lceil \log(\mathrm{depth}_F(z)+1)\rceil + |\zeta(K,u)| + O(\hat{\lg}\lceil \log(\mathrm{depth}_F(z)+1)\rceil) + O(\hat{\lg}|\zeta(K,u)|) \quad \text{by (1)}\\ &\leqslant |\zeta(K,u)| + O(\log\log n)\\ &\leqslant g_2(n) + O(\log\log n)\ ,\end{aligned}$$

since $\mathrm{depth}_F(z) \leqslant h \leqslant h(n) \in O(\log n)$ and $|\zeta(K,u)| \leqslant g_2(n) \in O(\log n)$. Thus, the length of $\kappa(K,u)$ satisfies the requirement for $g_2'(n)$.

Identity Testing: We now describe the identity tester I' that takes $\mu(K)$, $\kappa(K,u)$, and $\mu(v)$ as input and outputs 1 if $u = v$ or 0 if $u \neq v$. To do this, I' first examines $\mu(v)$ to determine if $v \in B_R$. If $\mu(v)$ has exactly one part, then $v \in B_R$. Otherwise, $\mu(v)$ has three parts and $v \notin B_R$. In both cases, I' will make use of $\kappa(K,u) = \langle \mathrm{depth}_F(z), \zeta(K,u)\rangle$, where z is the unique node of F such that $u \in B_z \setminus A_z$.

- If $v \in B_R$, then $\mu(v) = \langle \mu_R(v)\rangle$ and I' examines $\mathrm{depth}_F(z)$ as follows.
  - If $\mathrm{depth}_F(z) > 0$, then $z \notin R$ and $u \in B_z \setminus A_z \subseteq B_z \setminus B_R$. Since $v \in B_R$ and $u \notin B_R$, $u \neq v$. Thus, $I'(\mu(K),\kappa(K,u),\mu(v)) := 0$.
  - If $\mathrm{depth}_F(z) = 0$ then $z \in R$ and $u \in B_R$. There are now two possibilities that I' can distinguish by looking at the number of parts of $\mu(K)$.
    - If $\mu(K)$ has one part then $K \subseteq B_R$, $\mu(K) = \langle \mu_R(K)\rangle$, and $\zeta(K,u) = \kappa_R(K,u)$. Therefore, $I'(\mu(K),\kappa(K,u),\mu(v)) := I(\mu_R(K),\kappa_R(K,u),\mu_R(v))$.
    - If $\mu(K)$ has three parts, then $K \not\subseteq B_R$, $\mu(K) = \langle \mu_R(L), \mu_L(K \setminus B_R), c(K)\rangle$, and $\zeta(K,u) = \kappa_R(L,u)$, where L is the unique clique in $\mathcal{K}_R$ such that G_L contains $K \setminus B_R$. Then $K \cap B_R = K \cap L$, so $u \in L$. Therefore, $I'(\mu(K),\kappa(K,u),\mu(v)) := I(\mu_R(L),\kappa_R(L,u),\mu_R(v))$.
- If $v \notin B_R$ then $\mu(v) = \langle \mu_R(L'), \mu_{L'}(v), \alpha(v)\rangle$, where L' is the unique clique in $\mathcal{K}_R$ such that $v \in V(G_{L'})$.
  - If $\mathrm{depth}_F(z) = 0$ then $z \in R$, $u \in B_R$ and $v \notin B_R$, so $I'(\mu(K),\kappa(K,u),\mu(v)) := 0$.
  - If $\mathrm{depth}_F(z) > 0$ then $u \notin B_R$. Since $u \in K$ and $u \notin B_R$, $K \not\subseteq B_R$, so $\mu(K) = \langle \mu_R(L), \mu_L(K \setminus B_R), c(K)\rangle$ where L is the unique clique in $\mathcal{K}_R$ such that G_L contains $K \setminus B_R$. If $\mu_R(L') \neq \mu_R(L)$ then $L \neq L'$ and $u \neq v$ since G_L and $G_{L'}$ are vertex-disjoint, so $I'(\mu(K),\kappa(K,u),\mu(v)) := 0$.

    If $\mu_R(L) = \mu_R(L')$ then $L = L'$ since μ_R is injective. Since $u \in K$ and $u \notin B_R$, we have that $u \in K \setminus B_R$, so $K \setminus B_R$ is non-empty. Thus, $\zeta(K,u) = \zeta_L(K \setminus B_R, u)$ and I' can compute the local identifier $\kappa_L(K \setminus B_R, u)$ since, by (37),

    $$\kappa_L(K \setminus B_R, u) = \langle \mathrm{depth}_F(z) - 1, \zeta_L(K \setminus B_R, u)\rangle\ .$$

    In this case $I'(\mu(K),\kappa(K,u),\mu(v)) := I'(\mu_L(K \setminus B_R), \kappa_L(K \setminus B_R, u), \mu_L(v))$. □

3.5 Finalizing the Proof of Theorem 4

The following lemma proves Theorem 4, with quantitative bounds.

Lemma 17. *Let $\mathcal{G}$ be a hereditary class of graphs closed under taking disjoint union. Let $k \in \mathbb{N}$. Let $\mathcal{G}'$ be the class of graphs consisting of, for each $n \in \mathbb{N}$, every n-vertex graph G that has a tree-decomposition of adhesion-width at most k in which each torso is a member of $\mathcal{G}$. Let $g_i : \mathbb{N} \to \mathbb{R}^+$ be a non-decreasing function with $g_i \in O(\log n)$ for each $i \in [3]$. Suppose that $\mathcal{G}$ admits a weighted (g_1, g_2, g_3) mixed labelling scheme. Then $\mathcal{G}'$ admits a weighted (g_1'', g_2'', g_3'') mixed labelling scheme, where*

$$g_1''(n) = g_1(n) + O(g_2(n)) + O\left(\sqrt{g_3(n)\log n}\right) + O\left(\sqrt{\frac{\log n}{g_3(n)}}\log\log n\right) ,$$

$$g_2''(n) = g_2(n) + O\left(\sqrt{g_3(n)\log n}\right) + O(\log\log n) ,$$

$$g_3''(n) = O\left(\sqrt{g_3(n)\log n}\right) + O\left(\sqrt{\frac{\log n}{g_3(n)}}\log\log n\right) .$$

Proof. Let $b, h : \mathbb{N} \to \mathbb{R}^+$ be functions defined by

$$b(n) := 2^{\sqrt{g_3(n)\log n}} ,$$

$$h(n) := \log_{b(n)} n = \sqrt{\frac{\log n}{g_3(n)}} .$$

Since $g_1''(n)$, $g_2''(n)$, and $g_3''(n)$ all include asymptotic terms that increase without bound as a function of n, we may assume that $n \geqslant 2$ and $g_3(2) \geqslant 1$. Therefore, $b(n) \geqslant 2$.

Let $\mathcal{B}$ be a class of graphs that contains, for each $n \in \mathbb{N}^+$, every n-vertex graph that has a rooted $b(n)$-skinny forest-decomposition of adhesion-width at most k in which each torso is a member of $\mathcal{G}$. By Lemma 15, each graph in $\mathcal{B}$ admits a weighted $(g_1'(n), g_2'(n), g_3'(n))$ mixed labelling scheme, where

$$\begin{aligned} g_1'(n) &= g_1(n) + O(k\log b(n) + k\log\log n) \\ &= g_1(n) + O\left(\sqrt{g_3(n)\log n}\right) + O(\log\log n) , \\ g_2'(n) &= g_2(n) + O(\log b(n) + \log\log n) \\ &= g_2(n) + O\left(\sqrt{g_3(n)\log n}\right) + O(\log\log n) , \\ g_3'(n) &= g_3(n) + O(k + \log\log n) \\ &= g_3(n) + O(\log\log n) . \end{aligned}$$

Let $\mathcal{C}$ be a class of graphs that contains, for each $n \in \mathbb{N}^+$, every n-vertex graph that has a rooted forest-decomposition $(F, (C_x)_{x\in V(F)})$ of adhesion-width at most k, with $\text{height}(F) \leqslant h(n)$, and in which each torso is a member of $\mathcal{B}$. By Lemma 16, $\mathcal{C}$ has a weighted $(g_1''(n), g_2''(n), g_3''(n))$ mixed labelling scheme, where

$$g_1''(n) = g_1'(n) + k \cdot (g_2'(n) + O(\log\log n)) + h(n) \cdot (g_3'(n) + O(\log\log n)) + O(\log\log n)$$

$$
\begin{aligned}
&= g_1(n) + O\Big(\sqrt{g_3(n)\log n}\Big) + O(g_2(n)) + O\left((1+h(n))\log\log n\right)\\
g_2''(n) &= g_2'(n) + O(\log\log n)\\
&= g_2(n) + O\Big(\sqrt{g_3(n)\log n}\Big) + O(\log\log n)\\
g_3''(n) &= (h(n)+1)\cdot(g_3'(n) + O(k+\log\log n))\\
&= O\Big(\sqrt{g_3(n)\log n}\Big) + g_3(n) + O\left((1+h(n))\log\log n\right)\\
&= O\Big(\sqrt{g_3(n)\log n}\Big) + O\left((1+h(n))\log\log n\right) \quad .
\end{aligned}
$$

Substituting $h(n) = \sqrt{\log n / g_3(n)}$ into the error terms yields the bounds in the lemma statement.

Let G be an n-vertex graph in $\mathcal{G}'$. Then G has a tree-decomposition $\mathcal{T} := (T, (B_x \mid x \in V(T)))$ of adhesion-width at most k and such that $G\langle \mathcal{T}, B_x\rangle \in \mathcal{G}$ for each $x \in V(T)$. By Corollary 10, there is a tree-decomposition $\mathcal{Q} := (Q, (D_x \mid x \in V(Q)))$ of G such that $\text{height}(Q) \leqslant \log_{b(n)} n = h(n)$, $\mathcal{Q}$ has adhesion-width at most k, and such that each torso of $\mathcal{Q}$ has a $b(n)$-skinny tree-decomposition whose torsos are in $\mathcal{G}$. Therefore, all the torsos of $\mathcal{Q}$ are in the class $\mathcal{B}$ and therefore G is in the class $\mathcal{C}$. This completes the proof of the lemma. □

3.6 Proof of Theorem 1 with Explicit Bounds

We now prove our main result, Theorem 1, with an explicit lower order term.

Lemma 18. *Every proper minor-closed graph class $\mathcal{C}$ admits a $\log n + O\Big((\log n)^{3/4}\Big)$-bit adjacency labelling scheme.*

Proof. We show that $\mathcal{C}$ admits a weighted (g_1, g_2, g_3) mixed labelling scheme with $g_1(n) \in O((\log n)^{3/4})$. By the definition of mixed labelling schemes, this implies that $\mathcal{C}$ admits an $f(n)$-bit adjacency labelling scheme with $f(n) \leqslant \log n + O((\log n)^{3/4})$.

Let $k, a \geqslant 0$ be integers given by Theorem 3 for the class $\mathcal{C}$. So $\mathcal{C} \subseteq \mathcal{R}_k^{+a}$. By Lemma 13 and Lemma 19, $\mathcal{R}_k^{+a}$ admits a weighted (g_1', g_2', g_3') mixed labelling scheme, where $g_1'(n), g_3'(n) \in O(\sqrt{\log n})$ and $g_2'(n) \in O(\log\log n)$.

Let $\mathcal{Q}$ consist of every graph that can be formed by the disjoint union of a finite number of graphs in $\mathcal{R}_k^{+a}$. By Lemma 14, $\mathcal{Q}$ admits a weighted (g_1'', g_2'', g_3'') mixed labelling scheme with $g_1''(n), g_3''(n) \in O(\sqrt{\log n})$ and $g_2''(n) \in O(\log\log n)$. By definition, $\mathcal{Q}$ is closed under disjoint union. Also, by definition $\mathcal{R}_k^{+a} \subseteq \mathcal{Q}$. Since $\mathcal{R}_k^{+a}$ is hereditary, $\mathcal{Q}$ is hereditary.

Let $\mathcal{Q}'$ be the class of graphs that have a tree-decomposition whose torsos are in $\mathcal{Q}$. The graphs in $\mathcal{R}_k^{+a}$ have maximum clique size at most $2k + a + 2$. So the graphs in $\mathcal{Q}$ also have maximum clique size at most $2k + a + 2$. Since each adhesion of a tree-decomposition is a clique in some torso, each graph in $\mathcal{Q}'$ has a tree-decomposition of adhesion-width at most $2k + a + 2$ whose torsos are in $\mathcal{Q}$.

By Lemma 17, each graph in $\mathcal{Q}'$ admits a weighted (g_1, g_2, g_3) mixed labelling scheme

with

$$g_1(n) \in g_1''(n) + O\left(g_2''(n) + \sqrt{g_3''(n)\log n} + \sqrt{\frac{\log n}{g_3''(n)}}\log\log n\right) = O((\log n)^{3/4}) \ .$$

Since $\mathcal{R}_k^{+a} \subseteq \mathcal{Q}$, Theorem 3 implies that $\mathcal{C} \subseteq \mathcal{Q}'$. Therefore, $\mathcal{C}$ admits a weighted (g_1, g_2, g_3) mixed labelling scheme with $g_1(n) \in O((\log n)^{3/4})$. □

4 Discussion

We conclude with a brief discussion about constants and computational complexity. The constant a in Theorem 3 is surprisingly small. Suppose that some graph X is not in $\mathcal{G}$, where $X - A$ is planar for some non-empty set $A \subseteq V(X)$. Then Theorem 42 in [16] (which employs the structure theorem of Dvořák and Thomas [18]) shows that Theorem 3 holds with $a = |A| - 1$. For example, if K_t is not in $\mathcal{G}$, then Theorem 3 holds with $a = \max\{t-5, 0\}$. Moreover, using the recent polynomial bounds in the Graph Minor Structure Theorem by Gorsky, Seweryn, and Wiederrecht [25], in Theorem 3 one can obtain polynomial bounds on k and a (as a function of the excluded minor). However, it is open whether one can simultaneously get the above optimal bound on a and a polynomial bound on k. Gorsky et al. [25, Conjecture 18.9] conjecture this is possible (also see the discussion in [26]).

We now explain why the proof of Theorem 1 is constructive and that there is a polynomial time encoder. The only difficult part of this process is finding the decomposition described by the Graph Minor Product Structure Theorem (Theorem 3). Dujmović et al. [16] show how to obtain this decomposition in polynomial time, given the structural decomposition in the original Robertson-Seymour Graph Minor Structure Theorem. Finding the Robertson-Seymour decomposition is complicated, but polynomial time algorithms exist [12, 25, 32, 33]. Together, these give the parameters k and a, the tree-decomposition $\mathcal{T} := (T, (B_x : x \in V(T)))$ of G and the embedding of each torso of $\mathcal{T}$ into a graph in $\mathcal{R}_k^{+a}$. The adjacency labelling scheme in [15] and its modification to weighted mixed labellings in Appendix A are easily implemented in polynomial time just from their definitions. Thus, for any torso of $\mathcal{T}$, the mixed labelling of any subgraph of the torso and any set of cliques in the torso can be computed in polynomial time. The partition of T into skinny subtrees and the corresponding contracted tree T' and tree-decomposition $\mathcal{T}'$ are easily obtained (even in linear time). The labelling of each torso of $\mathcal{T}'$ given in Lemma 15 requires only computing alphabetic codes (Lemma 7 and Corollary 8) as well as weighted mixed labellings on subgraphs of torsos of $\mathcal{T}$. The final vertex label $\mu(v)$ for a vertex of G is then obtained by following the path from the root of T' to the home of v in $\mathcal{T}'$ and concatenating a sequence of clique labels, the vertex label for v in its home torso, and at most k local identifiers.

Finally, we note that Theorem 4 applies beyond proper minor-closed classes. Recall that, for fixed k, $\mathcal{R}_k$ consists of the graphs isomorphic to subgraphs of $H \boxtimes P$ for some graph H of treewidth at most k and some path P. This class is hereditary, closed under taking disjoint union, and admits an efficient weighted mixed labelling scheme (Lemma 19), so Theorem 4 applies to it. Moreover, for suitable $k = k(p)$, $\mathcal{R}_k$ contains every p-planar graph (graphs that can be drawn in the plane with at most p crossings per edge) [17], and more

generally every p-matching-planar graph [28] (graphs that have a drawing in the plane such that, for every edge e, every matching among the edges crossing e has size at most p). Note that these classes are not minor-closed. Consequently, for all fixed p and q, the class of graphs admitting a tree-decomposition of adhesion-width at most q in which every torso is p-matching-planar (again a class that is not minor-closed) admits a $(1+o(1))\log n$-bit adjacency labelling scheme.

Acknowledgement

This work was initiated during the Ninth Annual Workshop on Geometry and Graphs (v2), held January 21–28, 2022, at the Bellairs Research Institute of McGill University. The authors are thankful to the other organizers and participants for providing a stimulating research environment.

References

[1] Serge Abiteboul, Haim Kaplan, and Tova Milo. Compact labeling schemes for ancestor queries. In *Proc. 12th Annual ACM-SIAM Symposium on Discrete Algorithms* (SODA '01), pages 547–556. ACM-SIAM, 2001.

[2] Noga Alon. Implicit representation of sparse hereditary families. *Discrete & Computational Geometry*, 72(2):476–482, 2024. doi:10.1007/s00454-023-00521-0.

[3] Stephen Alstrup, Søren Dahlgaard, and Mathias Bæk Tejs Knudsen. Optimal induced universal graphs and adjacency labeling for trees. *J. ACM*, 64(4):27:1–27:22, 2017. doi:10.1145/3088513. Preliminary version appeared in FOCS 2015, pages 1311–1326.

[4] Stephen Alstrup, Cyril Gavoille, Haim Kaplan, and Theis Rauhe. Nearest common ancestors: A survey and a new algorithm for a distributed environment. *Theory Comput. Syst.*, 37(3):441–456, 2004. doi:10.1007/S00224-004-1155-5.

[5] Stephen Alstrup and Theis Rauhe. Improved labeling scheme for ancestor queries. In David Eppstein, editor, *Proc. 13th Annual ACM-SIAM Symposium on Discrete Algorithms* (SODA '02), pages 947–953. ACM-SIAM, 2002.

[6] Stephen Alstrup and Theis Rauhe. Small induced-universal graphs and compact implicit graph representations. In *Proc. 43rd IEEE Symposium on Foundations of Computer Science* (FOCS 2002), pages 53–62. IEEE, 2002. doi:10.1109/SFCS.2002.1181882.

[7] Marthe Bonamy, Louis Esperet, Carla Groenland, and Alex D. Scott. Optimal labelling schemes for adjacency, comparability, and reachability. In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 1109–1117. ACM, 2021. doi:10.1145/3406325.3451102.

[8] Marthe Bonamy, Cyril Gavoille, and Michał Pilipczuk. Shorter labeling schemes for planar graphs. *SIAM J. Discrete Math.*, 36(3):2082–2099, 2022. doi:10.1137/20M1330464. Preliminary version appeared in SODA 2020, pages 446–462.

[9] Édouard Bonnet, Julien Duron, John Sylvester, Viktor Zamaraev, and Maksim Zhukovskii. Tight bounds on adjacency labels for monotone graph classes. In

Proc. 51st International Colloquium on Automata, Languages, and Programming (ICALP 2024), volume 297 of *LIPIcs*, pages 31:1–31:18. Schloss Dagstuhl, 2024. doi:10.4230/LIPIcs.ICALP.2024.31.

[10] Édouard Bonnet, Julien Duron, John Sylvester, Viktor Zamaraev, and Maksim Zhukovskii. Small but unwieldy: A lower bound on adjacency labels for small classes. *SIAM J. Computing*, 53(5):1578–1601, 2024. doi:10.1137/23M1618661.

[11] Fan R. K. Chung. Universal graphs and induced-universal graphs. *J. Graph Theory*, 14(4):443–454, 1990. doi:10.1002/jgt.3190140408.

[12] Erik D. Demaine, MohammadTaghi Hajiaghayi, and Ken ichi Kawarabayashi. Algorithmic graph minor theory: Decomposition, approximation, and coloring. In *Proc. 46th Annual IEEE Symposium on Foundations of Computer Science* (FOCS'05), pages 637–646. IEEE Computer Society, 2005. doi:10.1109/SFCS.2005.14.

[13] Matt DeVos, Guoli Ding, Bogdan Oporowski, Daniel P. Sanders, Bruce Reed, Paul Seymour, and Dirk Vertigan. Excluding any graph as a minor allows a low tree-width 2-coloring. *J. Combinatorial Theory, Series B*, 91(1):25–41, 2004. doi:https://doi.org/10.1016/j.jctb.2003.09.001.

[14] Reinhard Diestel. *Graph Theory, Fifth Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2017. doi:10.1007/978-3-662-53622-3.

[15] Vida Dujmović, Louis Esperet, Cyril Gavoille, Gwenaël Joret, Piotr Micek, and Pat Morin. Adjacency labelling for planar graphs (and beyond). *J. ACM*, 68(6):42:1–42:33, 2021. doi:10.1145/3477542. Preliminary version appeared in FOCS 2020, pages 577–588.

[16] Vida Dujmović, Gwenaël Joret, Piotr Micek, Pat Morin, Torsten Ueckerdt, and David R. Wood. Planar graphs have bounded queue-number. *J. ACM*, 67(4):22:1–22:38, 2020. doi:10.1145/3385731. Preliminary version appeared in FOCS 2019, pages 862–875.

[17] Vida Dujmović, Pat Morin, and David R. Wood. Graph product structure for non-minor-closed classes. *J. Combin. Theory, Ser B.*, 162:34–67, 2023.

[18] Zdeněk Dvořák and Robin Thomas. List-coloring apex-minor-free graphs. *arXiv:1401.1399*, 2014. doi:10.48550/1401.1399.

[19] Peter Elias. Universal codeword sets and representations of the integers. *IEEE Trans. Inf. Theory*, 21(2):194–203, 1975. doi:10.1109/TIT.1975.1055349.

[20] Cyril Gavoille and Arnaud Labourel. Shorter implicit representation for planar graphs and bounded treewidth graphs. In Lars Arge, Michael Hoffmann, and Emo Welzl, editors, *Proc. 15th Annual European Symposium on Algorithms* (ESA 2007), volume 4698 of *Lecture Notes in Comput. Sci.*, pages 582–593. Springer, 2007. doi:10.1007/978-3-540-75520-3_52.

[21] Cyril Gavoille, David Peleg, Stéphane Pérennes, and Ran Raz. Distance labeling in graphs. *Journal of Algorithms*, 53(1):85–112, 2004. doi:https://doi.org/10.1016/j.jalgor.2004.05.002.

[22] Pawel Gawrychowski and Wojciech Janczewski. Simpler adjacency labeling for planar graphs with B-trees. In Karl Bringmann and Timothy M. Chan, editors, *Proc. 5th Symposium on Simplicity in Algorithms* (SOSA@SODA 2022), pages 24–36. SIAM, 2022. doi:10.1137/1.9781611977066.3.

[23] Edgar N. Gilbert and Edward F. Moore. Variable-length binary encodings. *Bell System Technical Journal*, 38(4):933–967, 1959. doi:10.1002/j.1538-7305.1959.tb01583.x.

[24] Paul C. Gilmore and Alan J. Hoffman. A characterization of comparability graphs and of interval graphs. *Canadian J. Math.*, 16:539–548, 1964. doi:10.4153/CJM-1964-055-5.

[25] Maximilian Gorsky, Michal T. Seweryn, and Sebastian Wiederrecht. Polynomial bounds for the graph minor structure theorem. In *Proc. 66th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2025*, pages 1961–1978. IEEE, 2025. doi:10.1109/FOCS63196.2025.00104.

[26] Maximilian Gorsky, Michał T. Seweryn, and Sebastian Wiederrecht. The price of homogeneity is polynomial. *arXiv:2602.01882*, 2026. doi:10.48550/arXiv.2602.01882.

[27] Hamed Hatami and Pooya Hatami. The implicit graph conjecture is false. In *Proc. 63rd Annual IEEE Symposium on Foundations of Computer Science* (FOCS 2022), pages 1134–1137. IEEE, 2022. doi:10.1109/FOCS54457.2022.00109.

[28] Kevin Hendrey, Nikolai Karol, and David R. Wood. Structure of k-matching-planar graphs. *CoRR*, abs/2507.22395, 2025. doi:10.48550/ARXIV.2507.22395. 2507.22395.

[29] Jacob Holm, Eva Rotenberg, and Mikkel Thorup. Planar reachability in linear space and constant time. In Venkatesan Guruswami, editor, *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 370–389. IEEE Computer Society, 2015. doi:10.1109/FOCS.2015.30.

[30] Sampath Kannan, Moni Naor, and Steven Rudich. Implicit representation of graphs. *SIAM J. Discrete Math.*, 5(4):596–603, 1992. doi:10.1137/0405049. Preliminary version appeared in STOC 1988, pages 334–343.

[31] Michal Katz, Nir A. Katz, Amos Korman, and David Peleg. Labeling schemes for flow and connectivity. *SIAM J. Comput.*, 34(1):23–40, 2004. doi:10.1137/S0097539703433912.

[32] Ken ichi Kawarabayashi, Robin Thomas, and Paul Wollan. A new proof of the flat wall theorem. *J. Combinatorial Theory, Series B*, 129:158–203, 2018. doi:10.1016/j.jctb.2017.09.002.

[33] Ken-ichi Kawarabayashi, Robin Thomas, and Paul Wollan. Quickly excluding a non-planar graph. *arXiv:2010.12397*, 2020. doi:10.48550/arXiv.2010.12397.

[34] Donald E. Knuth. *The Art of Computer Programming, Volume 3: Sorting and Searching*. Addison-Wesley, 2nd edition, 1998.

[35] Leon G. Kraft. *A Device for Quantizing, Grouping, and Coding Amplitude Modulated Pulses*. M.S. thesis, Massachusetts Institute of Technology, Cambridge, MA, 1949. Hdl:1721.1/12390.

[36] Kurt Mehlhorn. Nearly optimal binary search trees. *Acta Informatica*, 5(4):287–295, 1975. doi:10.1007/BF00264563.

[37] John H. Muller. *Local Structure in Graph Classes*. Ph.D. thesis, School of Information and Computer Science, Georgia Institute of Technology, 1988.

[38] C. St. J. A. Nash-Williams. Edge-disjoint spanning trees of finite graphs. *J. London Math. Society*, 36(1):445–450, 1961. doi:10.1112/jlms/s1-36.1.445.

[39] Serguei Norine, Paul Seymour, Robin Thomas, and Paul Wollan. Proper minor-closed families are small. *J. Combin. Theory Ser. B*, 96(5):754–757, 2006. doi:10.1016/j.jctb.2006.01.006.

[40] Jeremy P. Spinrad. *Efficient graph representations*, volume 19 of *Fields Institute Monographs*. American Math. Society, 2003.

[41] Mikkel Thorup. Compact oracles for reachability and approximate distances in planar digraphs. *J. ACM*, 51(6):993–1024, 2004. doi:10.1145/1039488.1039493.

A Mixed Labelling Scheme for $\mathcal{R}_k$

In this appendix, we modify the labelling scheme of Dujmović et al. [15] and its improved version by Gawrychowski and Janczewski [22] in order to show the existence of the efficient mixed labelling scheme for $\mathcal{R}_k$ mentioned in Subsection 1.2. We prove the following result.

Lemma 19. *For each fixed integer $k \geqslant 0$, the graph class $\mathcal{R}_k$ admits an efficient weighted (g_1, g_2, g_3) mixed labelling scheme, where $g_1(n), g_3(n) \in O(\sqrt{\log n})$, and $g_2(n) \in O(1)$.*

Proof. We begin by reviewing the proofs in [15] and [22] and then describe the changes needed to obtain the statement of Lemma 19.

Let G be an n-vertex graph in $\mathcal{R}_k$, so G is a subgraph of $H \boxtimes P$, where H is a graph of treewidth k and P is a path. Without loss of generality, assume that $P = (1, 2, \ldots, h)$, with $h \leqslant n$, and H is an edge-maximal graph of treewidth k. This implies that H has an acyclic orientation in which the out-neighbourhood $N_H^+(v)$ is a clique in H, for each $v \in V(H)$. It follows that the out-degree $|N_H^+(v)|$ is at most k, for each vertex $v \in V(H)$.

Mapping $V(H)$ onto intervals: The first step in [15] is an idea used by Gavoille and Labourel [20] in their labelling scheme for bounded-treewidth graphs. Each vertex v of H is mapped to a real interval $f(v)$. This mapping has the following properties (for some fixed $c > 0$):

(I1) For any two vertices $v, w \in V(H)$, $f(v) \subseteq f(w)$, $f(w) \subseteq f(v)$ or $f(v) \cap f(w) = \emptyset$.
(I2) For each $vw \in E(H)$, $f(v) \cap f(w) \neq \emptyset$.
(I3) For each $x \in \mathbb{R}$, $|\{v \in V(H) : x \in f(v)\}| \leqslant ck \log n$.

The existence of such a mapping f is easily established by a recursive procedure that takes an open interval (a, b) and a subgraph H' of H. The procedure identifies a set S of at most $k+1$ vertices in H' such that the components of $H' - S$ can be partitioned into two graphs H_1' and H_2', each with at most $\frac{2}{3}|V(H')|$ vertices. For each $v \in S$, $f(v) := (a, b)$. The procedure is then applied inductively to map the vertices of H_1' onto subintervals of $(a, (a+b)/2)$ and then again to map the vertices of H_2' onto subintervals of $((a+b)/2, b)$.

Let I be the interval intersection graph with vertex-set $V(I) := V(H)$ and edge-set $E(I) := \{vw \in \binom{V(H)}{2} : f(v) \cap f(w) \neq \emptyset\}$. Property (I2) implies that H is a subgraph of I. Property (I3) implies that H has no clique larger than $ck \log n$, for some fixed constant c.

The graph I has a proper colouring $\varphi : V(H) \to [\lfloor ck \log n \rfloor]$ [24]. The label for each vertex (v,i) of G will include the $O(\log(k \log n))$-bit integer $\varphi(v)$.

Mapping $V(H)$ onto points: For each vertex v of H, let $x_f(v)$ be the midpoint of the interval $f(v)$.[4] Property (I1) then implies:

(I4) For each clique K in I, there exists $v \in K$ such that $x_f(v) \in f(w)$ for each $w \in K$.

Indeed, a vertex w of K that minimizes the length of $f(w)$ satisfies this condition.

Treating G as a sequence of graphs: Each vertex of G is a pair (v,i) with an *H-coordinate* $v \in V(H)$ and a *P-coordinate* $i \in V(P)$. We may assume, without loss of generality, that G contains at least one vertex with P-coordinate i, for each $i \in [h]$ (and therefore $h \leqslant n$). The next idea in [15] is to treat $V(G)$ as a sequence of vertex subsets of H. For each $i \in [h]$, let $V_i := \{v \in V(H) : \{(v,i),(v,i-1)\} \cap V(G) \neq \emptyset\}$. Each vertex (v,i) of G contributes a vertex to the two sets V_i and V_{i+1}, so $\sum_{i=1}^{h} |V_i| \leqslant 2n$. For each vertex (v,i) of G, define the set $X_{v,i} := \{v\} \cup (V_i \cap N_H^+(v))$.

Observation 20. *For any edge $(v,i)(w,j)$ of G with $w \in N_H^+(v)$, we have that $w \in X_{v,i}$ (if $j \in \{i-1,i\}$) or $w \in X_{v,i+1}$ (if $j \in \{i,i+1\}$).*

The label of any vertex (v,i) of G has two main parts, whose lengths sum to $\log n + o(\log n)$. The first part, $\lambda_1((v,i))$ is determined by the P-coordinate, i. The second part, $\lambda_2((v,i))$, is obtained from a labelling of $H[V_i]$, so it represents the H-coordinate, v.

Let $S_0 := S_{h+1} := \emptyset$ and, for each $i \in [h]$, let $S_i := \{x_f(v) : v \in V_i\}$. Note that S_i is a set of real numbers, for each $i \in [h]$. It is helpful to impose a relationship between consecutive sequences S_i and S_{i+1}, for each $i \in [h-1]$. Therefore, [15] defines a sequence of supersets $S_1^+, \ldots, S_h^+$. The exact definition of $S_1^+, \ldots, S_h^+$ is not necessary for the current discussion. The relevant properties here are:

(a) $S_i^+ \supseteq S_i$ for each $i \in [h]$;
(b) there is a constant $c > 0$ such that $\sum_{i=1}^{h} |S_i^+| \leqslant cn$.
(c) there is a constant $c > 0$ such that $1/c \leqslant |S_i^+|/|S_{i+1}^+| \leqslant c$, for each $i \in [h-1]$.

We remark that the contents of the sets $S_1^+, \ldots, S_h^+$ are entirely determined by the contents of the sets $S_1, \ldots, S_h$. (Formally, the sequence $S_1^+, \ldots, S_h^+$ is the result of applying a function $q : (2^{\mathbb{R}})^* \to (2^{\mathbb{R}})^*$ whose input is a sequence of sets of real numbers and whose output is a sequence of sets of real numbers. The function q is universal; it does not depend on H, P, or G.)

The row label λ_1: With the sets $S_1^+, \ldots, S_h^+$ described, we can now discuss λ_1. The first part of $\lambda_1(v,i)$ is obtained by applying Corollary 8 to the set $[h]$ using the weight function defined by $i \mapsto |S_i^+|$. This gives a prefix-free code $\rho : [h] \to \{0,1\}^*$ where, by Property (b) of Corollary 8, $|\rho(i)| \leqslant \log \rho([h]) - \log |S_i^+| + O(1) \leqslant \log n - \log |S_i^+| + O(1)$ for each $i \in [h]$. For each $i \in \{2, \ldots, h\}$, a string $\pi(i)$ of length $O(\log \log n)$ is computed so that $\rho(i-1)$ can be derived from $\rho(i)$ and $\pi(i)$. Formally, there exists a universal function $P : (\{0,1\}^*)^2 \to \{0,1\}^*$

[4] Any real number $x_f(v)$ in the interior of the interval $f(v)$ would also work.

such that $P(\rho(i), \pi(i)) = \rho(i-1)$, for each $i \in \{2, \ldots, h\}$. Then

$$\lambda_1(v,i) := \langle \rho(i), \pi(i) \rangle$$

and

$$|\lambda_1(v,i)| \leqslant \log n - \log |S_i^+| + O(\log\log n) \ . \tag{38}$$

for each vertex (v,i) of G.

Bulk trees: In [15], each set S_i^+ is stored in a binary search tree T_i of height $\log|S_i^+| + o(\log n)$. As in Corollary 8, each node x of T_i is assigned a *signature* $\sigma_{T_i}(x)$ that describes the root-to-x path $P_{T_i}(x)$ as a binary string of length $\text{depth}_{T_i}(x)$. Note that, for any $x \in V(T_i)$:

$$|\sigma_{T_i}(x)| \leqslant \text{height}(T_i) \leqslant \log|S_i^+| + o(\log n)$$

For each $i \in [h]$, each vertex $v \in V_i$ is mapped to the node $x_{T_i}(v) \in V(T_i)$ of minimum T_i-depth such that (the real number) $x_{T_i}(v)$ is contained in the (real interval) $f(v)$. For each $v \in V_i$, the node $x_{T_i}(v)$ always exists because $x_f(v) \in S_i \subseteq S_i^+$ and $x_f(v) \in f(v)$. The same reasoning implies that $x_{T_i}(w)$ is defined, for each $w \in X_{v,i}$.

For each $v \in V_i$, define the *signature* $\sigma(v,i) := \sigma_{T_i}(x_{T_i}(v))$. Note that, for two vertices (v,i) and (v',i') of G, $(v,i) = (v',i')$ if and only if

$$(i, \sigma(v,i), \varphi(v)) = (i', \sigma(v',i'), \varphi(v')) \ . \tag{39}$$

For each $(v,i) \in V(G)$, the *extended signature* $\sigma^+(v,i)$ is the signature in $\Sigma_{v,i} := \{\sigma(w,i) : w \in X_{v,i}\}$ of maximum length. Since $X_{v,i}$ is a clique in H, property (I4) implies that every signature in $\Sigma_{v,i}$ is a prefix of $\sigma^+(v,i)$ for each vertex (v,i) of G. In particular, for each $w \in X_{v,i}$, the signature $\sigma(w,i)$ can be derived easily from $\sigma^+(v,i)$ and $|\sigma(w,i)|$.

For each $(v,i) \in V(G)$, define

$$\begin{aligned}
\alpha^-(v,i) &= \langle\, \langle |\sigma(w,i)|, \varphi(w)\rangle \mid w \in \{v\} \cup N_H^+(v),\ (w, i-1) \in N_G((v,i))\, \rangle \\
\alpha^0(v,i) &= \langle\, \langle |\sigma(w,i)|, \varphi(w)\rangle \mid w \in N_H^+(v),\ (w, i) \in N_G((v,i))\, \rangle \\
\alpha^+(v,i) &= \langle\, \langle |\sigma(w,i+1)|, \varphi(w)\rangle \mid w \in \{v\} \cup N_H^+(v),\ (w, i+1) \in N_G((v,i))\, \rangle \\
\alpha(v,i) &= \langle\, \alpha^-(v,i), \alpha^0(v,i), \alpha^+(v,i)\, \rangle.
\end{aligned}$$

Since $|N_G^+(v)| \leqslant k$ for each $v \in V(H)$, $|\alpha(v,i)| \in O(k(\log k + \log\log n))$.

The last (and critical) piece of the labelling scheme in [15] is the notion of a *transition code*. The authors of [15] choose the sets $S_1^+, \ldots, S_h^+$ and design the binary search trees $T_1, \ldots, T_h$ so that, for any vertex (v,i) of G, the extended signature $\sigma^+(v,i+1)$ can be derived from $\sigma^+(v,i)$ and a transition code $\tau((v,i))$ of length $o(\log n)$. Formally, there is a universal function $J : (\{0,1\}^*)^2 \to \{0,1\}^*$ such that $J(\sigma^+(v,i), \tau(v,i)) = \sigma^+(v,i+1)$ for each $(v,i) \in V(G)$.

This gives all the pieces needed for the second part of the label for a vertex (v,i) of G:

$$\lambda_2(v,i) := \langle \sigma^+(v,i),\ |\sigma(v,i)|,\ \varphi(v),\ \alpha(v,i),\ \tau(v,i),\ |\sigma(v,i+1)| \rangle \ .$$

The complete vertex label for a vertex (v,i) of G is

$$\mu((v,i)) := \langle \lambda_1(v,i),\ \lambda_2(v,i) \rangle$$

which has length $\log n + O(\sqrt{\log n \log\log n})$, for fixed k. (For k that grows with n, $|\mu(v,i))| = \log n + O(\sqrt{\log n \log\log n} + k(\log k + \log\log n))$.)

Adjacency Testing We now describe a procedure that defines the output of the adjacency tester A. The operation of this procedure makes use of the fact that $\lambda_2(v,i)$ contains all the information in $\sigma^+(v,i)$, $\sigma(v,i)$, $\sigma^+(v,i+1) = J(\sigma^+(v,i),\tau(v,i))$, and $\sigma(v,i+1)$.

Given the labels $\mu((v,i))$ and $\mu((w,j))$ for two vertices (v,i) and (w,j) of G, the procedure first uses $\lambda_1(v,i)$ and $\lambda_1(w,j)$ (along with the function P) to test if $i = j$, $i = j+1$ or $i = j-1$. If none of these is the case, then the procedure can immediately conclude that $(v,i)(w,j)$ is not an edge of H, since $i \neq j$ and ij is not an edge of P. Since G is a subgraph of $H \boxtimes P$, this implies that $A(\mu((v,i)),\mu((w,j))) := 0$. Otherwise, the procedure deals with one of the preceding cases:

- If $i = j$, then the procedure computes $\sigma(w,i)$ using $\sigma^+(w,i)$ and $|\sigma(w,i)|$. Then the procedure uses $\sigma^+(v,i)$ and $\alpha^0(v,i)$ to compute $\sigma(w',i)$ and $\varphi(w')$ for each $w' \in N_H^+(v)$ such that $(w',i) \in N_G((v,i))$. For each of these the procedure checks if $(\sigma(w',i),\varphi(w')) = (\sigma(w,i),\varphi(w))$. If so, then $(w,i) = (w',i) \in N_G((v,i))$ so $A(\mu((v,i)),\mu((w,j))) := 1$.

  If this test fails for each of the entries in the second part of $\alpha(v,i)$, then the procedure concludes that $w \notin X_{v,i}$ or $(w,i) \notin N_G((v,i))$. This still leaves the possibility that $v \in X_{w,i}$ and $(v,i) \in N_G((w,j))$. The procedure checks this in exactly the same manner, but with the roles of $\mu(v,i)$ and $\mu(w,j)$ reversed. If this test also fails to return a result, then $A(\mu((v,i)),\mu((w,j))) := 0$.

- If $i = j+1$ then the procedure computes $\sigma^+(w,i) = J(\sigma^+(w,j),\tau(w,j))$. Then the procedure computes $\sigma(w,i)$ using $\sigma^+(w,i)$ and $|\sigma(w,i)|$. Then the procedure continues as in the previous case, except now using entries from $\alpha^-(v,i)$ to find (w',i) such that $(\sigma(w',i),\varphi(w')) = (\sigma(w,i),\varphi(w))$.

  If this search fails, then the procedure continues as in the previous case, except now using entries from $\alpha^+(w,j)$ to find a (v',i) such that $(\sigma(v',i),\varphi(v')) = (\sigma(v,i),\varphi(v))$. If this search also fails, then $A(\mu((v,i)),\mu((w,j))) := 0$.

- If $i = j-1$ then the procedure computes $\sigma^+(v,j) = J(\sigma^+(v,i),\tau(v,i))$. Then the procedure computes $\sigma(w,j)$ using $\sigma^+(w,j)$ and $|\sigma(w,j)|$. Then the procedure continues as in the previous cases, except now using entries from $\alpha^+(v,i)$ to find (w',j) such that $(\sigma(w',j),\varphi(w')) = (\sigma(w,j),\varphi(w))$.

  If this search fails, then the procedure continues by using entries in $\alpha^+(w,j)$ to find (v',j) such that $(\sigma(v',j),\varphi(v')) = (\sigma(v,j),\varphi(v))$. If this search also fails, then the procedure concludes that $A(\mu((v,i)),\mu((w,j))) := 0$.

B-trees: Gawrychowski and Janczewski [22] use weight-balanced B-trees to replace the (binary) bulk trees used in [15]. A *B-tree* stores a set of real numbers in the leaves of a tree B in which all leaves have the same depth. *Weight-balanced B-trees* are parameterized by a parameter $a \geqslant 6$. A weight-balanced B-tree B is *semi-balanced* if, for each node x of B, the number of leaves in the subtree B_x is at most $6a^{\text{height}(B_x)}$ and for each *non-root* node x of B, the number of leaves in B_x is at least $\frac{1}{2}a^{\text{height}(B_x)}$. Let B be a weight-balanced B-tree with $r \in [n]$ leaves. The weight-balance conditions guarantee that $\text{height}(B) \leqslant \log_a r + 1$. The weight-balance conditions also ensure that the number of children of any node x of B is at most $12a$. This implies that any root-to-leaf path in a B-tree can be encoded by a

sequence of $\log_a r + 1$ integers, each requiring $\lceil \log(12a) \rceil \leqslant 5 + \log a$ bits to represent, for a total of at most $\lceil \log(12a) \rceil \cdot (\log_a r + 1) = \log r + O(\log_a r + \log a)$ bits. The value of a used in [22] is approximately $2^{\sqrt{\log n}}$, so that any root-to-leaf path in B can be represented using $\log r + O(\sqrt{\log n})$ bits, for any $r \in O(n)$.

The sequence of binary search trees $T_1, \ldots, T_h$ used in [22] is replaced with a sequence of B-trees $B_1, \ldots, B_h$. For each $i \in [h]$, the leaves of B_i store the values in S_i^+. For each $i \in [h]$, each vertex $(v, j) \in V_i \times \{i-1, i\}$ is mapped to the lowest-common-B_i-ancestor $x_{B_i}(v)$ of $f(v) \cap S_i^+$. Then, for a vertex (v, i) of G, the signature $\sigma(v, i)$ is the encoding of $P_{B_i}(x_{B_i}(v))$. Then

$$|\sigma(v,i)| \leqslant \lceil \log(12a) \rceil \cdot (\text{depth}_{B_i}(x_{B_i}(v))) = \lceil \log(12a) \rceil (\text{depth}_{B_i}(\text{lca}_{B_i}(f(v) \cap S_i^+))) \ .$$

An upper bound on $|\sigma((v,i))|$ can be obtained by using the $\log_a |S_i^+| + 1$ upper bound on $\text{height}(B_i)$:

$$|\sigma(v,i)| \leqslant \lceil \log(12a) \rceil \text{height}(B_i) \leqslant \lceil \log(12a) \rceil (\log_a |S_i^+| + 1) = \log |S_i^+| + O(\sqrt{\log n})$$

The main advantage of using B-trees over the bulk trees in [15] is that in B-trees the rebalancing operations needed to maintain the height bound result in a much simpler transition code $\tau(w, i-1)$. This simplicity ultimately comes from the fact that, for any $p \in S_{i-1} \cap S_i$, the root-to-p path $P_{B_{i-1}}(p)$ in B_{i-1} and the root-to-p path $P_{B_i}(p)$ in B_i differ by at most a single vertex. More precisely, a subpath xyz in $P_{B_{i-1}}(p)$ can be replaced by a subpath $xy'z$ to give $P_{B_i}(x)$.[5] In terms of the signature function σ, this means that the difference between these two paths can be described by an $O(\log \log_a n)$-bit integer that gives the index of the vertex y being replaced and two $\lceil \log(12a) \rceil$ bit integers that give the index of y' in the list of children of x and the index of z in the list of children of y'.% [6] As a result of this, the transition code $\tau((w, i-1))$ used for the B-trees in [22] is shorter (it has length $O(\sqrt{\log n})$ rather than the $O(\sqrt{\log n \log \log n})$ required by bulk trees) and, on a computer with $O(\log n)$-bit words, the translation from $\sigma^+((w, i-1))$ to $\sigma^+((w,i))$ can be done in constant time, resulting in a constant-time adjacency testing procedure. (All of the other adjacency-testing operations for both schemes can be done in constant-time.)

Except for the new definition of the basic signature σ and the transition function τ, every other aspect of the resulting labelling scheme and adjacency testing procedure is as described in [15].

A weighted mixed labelling scheme: We now describe the changes to [15, 22] needed to prove Lemma 19. For this we will define an adjacency tester A and an identity tester I. As usual, we first describe the format of the vertex and clique labels before describing the operation of A and I.

[5] There are two exceptions. Occasionally, $P_{B_i}(x)$ is obtained by removing the first vertex of $P_{B_{i-1}}(x)$ or by adding a vertex at the beginning of $P_{B_{i-1}}(x)$. Since these are distracting and easily handled, we ignore them here.

[6] Determining $\sigma^+((w,i))$ from $\sigma^+((w,i-1))$ requires a bit more work, since it is possible that $x_{B_i}(w) \neq x_{B_{i-1}}(w)$. This is common to both labelling schemes and does not affect the upcoming discussion. The important thing for what comes next is the definition of $x_{B_i}(w) = \text{lca}_{B_i}(f(w) \cap S_i^+)$.

Let H be an edge-maximal graph of treewidth at most k, let $P = (1, \ldots, h)$ be a path, let G^+ be a subgraph of $H \boxtimes P$, let G be a spanning subgraph of G^+ and let $\omega : V(G) \to \mathbb{R}^+$ be a weight function. As in other proofs, we may assume that ω is nice, so that $\omega(v)$ is a positive integer, for each $v \in V(G)$, and that $\log \omega(G) \in O(\log n)$. We may also assume that G^+ is an induced subgraph of $H \boxtimes P$.

To help with clique labelling later, we will introduce some *dummy* vertices to G^+ and G. Since G^+ is an induced subgraph of $H \boxtimes P$, this also introduces edges and cliques in G^+. For each vertex (v,i) of G^+ with $i \in [h-1]$, if G^+ does not contain the vertex $(v,i+1)$ then we add the dummy vertex $(v,i+1)$ to G^+ and G and define $\omega(v,i+1) := \omega(v,i)$. The addition of dummy vertices does not increase the number of vertices or the total weight $\omega(G)$ by more than a factor of 2. Any vertex of G that is not a dummy vertex is called a *real* vertex.

To satisfy the requirement on $g_1(n)$, the label $\mu((v,i))$ of a vertex (v,i) of G should have length $\log \omega(G) - \log \omega((v,i)) + O(\sqrt{\log n})$. To achieve this, we require that the extended signature $\sigma^+(v,i)$ have length at most $\log |S_i^+| - \log \omega((v,i)) + O(\sqrt{\log n})$. To accomplish this, we define an intermediate weight function δ, as in the proof of Lemma 16. For each $(w,j) \in V(H \boxtimes P)$, define $X_{w,j}^{-1} := \{(v,i) \in V(G) \mid i \in \{j, j+1\},\ w \in X_{v,i}\}$ and define

$$\delta(w,j) := \sum_{(v,i) \in X_{w,j}^{-1}} \omega(v,i) + \omega((w,j-1)) \ ,$$

where we use the convention that $\omega((w,j-1)) = 0$ if $(w,j-1) \notin V(G)$. The intuition behind the sum in the definition $\delta(w,j)$ is that, for each vertex (v,i) of G, the extended signature $\sigma^+(v,i)$ must have length at most $\log |S_i^+| - \log \omega((v,i)) + O(\sqrt{\log n})$. However, $\sigma^+(v,i)$ should be long enough to contain $\sigma(w,i)$ for each $w \in X_{v,i}$. Thus, δ increases the weight of (w,j) to ensure this. Importantly, for each $(v,i) \in V(G)$ and each $w \in X_{v,i}$,

$$\delta(w,i) \geqslant \omega((v,i)) \ . \tag{40}$$

(The final term, $\omega(w,j-1)$ in the definition of $\delta(w,j)$ will be useful for clique labelling later.)

The total weight assigned by δ is

$$\sum_{(w,j) \in V(H \boxtimes P)} \delta(w,j) \leqslant \sum_{(v,i) \in V(G))} |X_{v,i}| \cdot \omega((v,i)) + \omega(G) \leqslant (2k+3)\omega(G) \ . \tag{41}$$

At this point, we proceed almost exactly as in the proof of Lemma 7. For each $i \in [h]$ and each $v \in V_i$, define

$$S_{\delta,i}(v) := \{(x_f(v), 1), \ldots, (x_f(v), \delta(v,i)\}$$

and define $S_{\delta,i} := \bigcup_{v \in V_i} S_{\delta,i}(v)$. For each $i \in [h]$, treat $S_{\delta,i}$ as a totally ordered set where order is determined by lexicographic comparison. Note that $|S_{\delta,i}| = \delta(V(H) \times \{i\})$ for each $i \in [h]$ and $\sum_{i=1}^{h} |S_{\delta,i}| = \sum_{(w,j) \in H \boxtimes P} \delta(w,j) \leqslant (2k+3)\omega(G)$, by (41).

In the following, we use the convention that a real interval (a,b) contains $(x,j) \in S_i^+$ if (the real interval) (a,b) contains (the real number) x. In particular, for a vertex $v \in V(H)$

and $i \in [h]$, $f(v) \cap S_i^+ := \{(x,j) \in S_i^+ : x \in f(v)\}$. Then, just as in the unweighted case $x_{B_i}(v) = \mathrm{lca}_{B_i}(f(v) \cap S_i^+)$.

At this point, we apply exactly the same machinery used in [15, 22] beginning with the sets $S_{\delta,1}, \ldots, S_{\delta,h}$ instead of the sets $S_1, \ldots, S_h$. This results in sets $S^+_{\delta,1}, \ldots, S^+_{\delta,h}$ of total size $\sum_{i=1}^h |S^+_{\delta,i}| \in O(k \cdot \omega(G)) \in O(kn)$, by (41), and since ω is nice. This has the desired effect because a B-tree of height h has at most $6a^h$ leaves. Therefore,

$$\begin{aligned}
\mathrm{depth}_{B_i}(x_{B_i}(v)) &= \mathrm{depth}_{B_i}(\mathrm{lca}_{B_i}(f(v) \cap S_{\delta,i})) \\
&\leqslant \mathrm{depth}_{B_i}(\mathrm{lca}_{B_i}(S_{\delta,i}(v))) \\
&\leqslant \mathrm{height}(B_i) - \log_a(\delta(v,i)/6) \\
&\leqslant \log_a |S^+_{\delta,i}| - \log_a \delta(v,i) + O(1)
\end{aligned}$$

.

Furthermore, for any $(w,j) \in X_{v,i}$, by (40),

$$\mathrm{depth}_{B_i}(x_{B_i}(w)) \leqslant \log_a |S^+_{\delta,i}| - \log_a \delta(w,i) + O(1) \leqslant \log_a |S^+_{\delta,i}| - \log_a \omega(v,i) + O(1) \ .$$

Therefore,

$$\begin{aligned}
|\sigma^+(v,i)| &= \max\{|\sigma((w,i))| \mid w \in X_{v,i}\} \\
&\leqslant \lceil \log(12a) \rceil \cdot \max\{\mathrm{depth}_{B_i}(x_{B_i}(w)) \mid w \in X_{v,i}\} \\
&\leqslant \lceil \log(12a) \rceil \cdot (\log_a |S^+_{\delta,i}| - \log_a \omega((v,i)) + O(1)) \\
&\leqslant (5 + \log a) \cdot (\log_a |S^+_{\delta,i}| - \log_a \omega((v,i)) + O(1)) \\
&\leqslant \log |S^+_{\delta,i}| - \log \omega((v,i)) + O(1 + \log_a |S^+_{\delta_i}| + \log a) \\
&\leqslant \log |S^+_{\delta,i}| - \log \omega((v,i)) + O(\sqrt{\log n}) \ .
\end{aligned}$$

Then

$$\begin{aligned}
|\lambda_2(v,i)| &\leqslant \log |S^+_{\delta,i}| - \log \omega((v,i)) + O(\sqrt{\log n} + k(\log k + \log\log n)) \\
&\leqslant \log |S^+_{\delta,i}| - \log \omega((v,i)) + O(\sqrt{\log n} + k(\log k + \log\log n)) \ .
\end{aligned}$$

By (41), we have that $|\lambda_1((v,i))| \leqslant \log \omega(G) - \log |S^+_{\delta,i}| + O(\log\log n)$, so

$$\begin{aligned}
|\mu((v,i))| &= |\lambda_1((v,i))| + |\lambda_2((v,i))| + O(\log\log n) \\
&\leqslant \log \omega(G) - \log |S^+_{\delta,i}| + \log |S^+_{\delta,i}| - \log \omega((v,i)) + O(\sqrt{\log n} + k(\log k + \log\log n)) \\
&= \log \omega(G) - \log \omega((v,i)) + O(\sqrt{\log n} + k(\log k + \log\log n)) \ .
\end{aligned}$$

For any fixed k, $|\mu((v,i))| = \log \omega(G) - \log \omega((v,i)) + O(\sqrt{\log n})$, satisfying the requirement $g_1(n) \in O(\sqrt{\log n})$. Since the vertex labels assigned by μ have the same format as those in [3, 15], the adjacency tester A proceeds exactly as described in [5, 15].

Clique labelling and identity testing: Thus far, we have shown that the lengths of the vertex labels promised in Lemma 19 are achievable. What remains is to describe the clique labels, the local identifiers, and the identity testing procedure.

Let K be a clique in G^+ that contains no dummy vertices and let $V_K := \{v \mid (v,j) \in K\}$.[7] Then $K \subseteq V(H) \times \{i-1, i\}$ and $V_i \supseteq V_K$ for some $i \in \{2, \ldots, h\}$. Since K is a clique in $G^+ \subseteq H \boxtimes P$, V_K is a clique in H. Therefore, there exists $v^* \in V_K$ such that $V_K \subseteq \{v^*\} \cup N_H^+(v^*)$.

If $(v^*, i) \in K$ then $\delta(v^*, i) \geqslant \omega((v^*, i))$. If $(v^*, i) \notin K$ then $(v^*, i-1) \in K$ and, by the definition of δ, $\delta(v^*, i) \geqslant \omega((v^*, i-1))$.[8] In either case, $\delta(v^*, i) \geqslant \min_{(v,j)\in K} \omega((v,j))$. Since K contains no dummy vertices, (v^*, i) is a (possibly dummy) vertex of G^+. Therefore, $\mu((v^*, i)) = \langle \lambda_1(v^*, i), \lambda_2(v^*, i) \rangle$ is defined. Furthermore, $X_{v^*,i}$ contains V_K. To summarize, $\mu((v^*, i))$ is defined, and has length at most $\log n - \log \min_{(v,j)\in K}(\omega((v,j))) + O(\sqrt{\log n})$. Also, $\mu((v^*, i)))$ contains $\sigma^+(v^*, i)$, which contains $\sigma(w, i)$ as a prefix, for each $w \in V_K$.

We can now mimic the adjacency labelling. Define

$$\begin{aligned}
\alpha^-(K) &:= \langle\, \langle |\sigma(w,i)|, \varphi(w) \rangle \mid (w, i-1) \in K \,\rangle \\
\alpha^0(K) &:= \langle\, \langle |\sigma(w,i)|, \varphi(w) \rangle \mid (w, i) \in K \,\rangle \\
\alpha(K) &:= \langle\, \alpha^-(K),\ \alpha^0(K) \,\rangle \ .
\end{aligned}$$

Since $|K| \leqslant 2k+2$, $|\alpha(K)| \in O(k \log\log n)$. Define

$$\mu(K) := \langle\, \lambda_1(v^*, i),\ \sigma^+(v^*, i),\ \alpha(K) \,\rangle \ .$$

To see that μ is injective when restricted to cliques in G^+, suppose $\mu(K) = \mu(L)$ for two distinct cliques K and L. Then their first parts match, which implies $i_K = i_L = i$ since the prefix-free code $\rho(i)$ inside λ_1 uniquely determines the P-coordinate i. They also have the same extended signature $\sigma^+ := \sigma^+(v_K^*, i) = \sigma^+(v_L^*, i)$ and the same third part $\alpha(K) = \alpha(L)$. The set of vertices in K is uniquely determined by i, σ^+, and $\alpha(K)$: for each entry $\langle \ell, \varphi \rangle$ in $\alpha^-(K)$ (respectively, $\alpha^0(K)$), there is a vertex $(w, i-1)$ (respectively, (w, i)) in K where $\varphi(w) = \varphi$ and $\sigma(w, i)$ is the prefix of σ^+ of length ℓ. By (39), the H-coordinate w is uniquely identified by i, $\sigma(w, i)$, and $\varphi(w)$, which in turn uniquely determines each vertex (w, j) of K. Since $\alpha(K) = \alpha(L)$, we conclude $K = L$. Thus μ is injective on cliques of G^+.

Then,

$$\begin{aligned}
|\mu(K)| &\leqslant |\lambda_1(v^*, i)| + |\sigma^+(v^*, i)| + O(k \log\log n) \\
&\leqslant \log \omega(G) - \log \delta(v^*, i) + O(\sqrt{\log n} + k(\log k + \log\log n)) \\
&\leqslant \log \omega(G) - \log \min_{(v,j)\in K} \omega((v,j)) + O(\sqrt{\log n} + k(\log k + \log\log n)).
\end{aligned}$$

For fixed k, $|\mu(K)| \leqslant \log \omega(G) - \log \min_{(v,j)\in K} \omega((v,j)) + O(\sqrt{\log n})$, satisfying the requirement $g_3(n) \in O(\sqrt{\log n})$.

[7] The requirement that K contains no dummy vertices is justified by the fact that we are only required to provide labels for cliques that exist in the original input graph G^+.

[8] This is precisely the reason for the $\omega((v, i-1))$ term in the definition of $\delta(v, i)$.

Local identifiers For each vertex $(u,j) \in K$, the local identifier for (u,j) is a single integer $\kappa(K,(u,j)) \in [2k+2]$ that indicates the position of the entry for (u,j) in $\alpha(K)$. This integer also implicitly encodes whether $j = i-1$ or $j = i$. Thus $|\kappa(K,(u,j))| = O(\log k)$. For fixed k, this satisfies the requirement $g_2(n) \in O(1)$.

Identity testing We now describe the identity tester I. Let K be a clique in G^+ containing only real vertices, let $(u,j) \in K$, and let (v,i') be a real vertex of G. Let (v^*,i) be as in the description of $\mu(K)$, so that $K \subseteq V(H) \times \{i-1,i\}$ and therefore $j \in \{i-1,i\}$. Given $\mu(K) = \langle \lambda_1(v^*,i), \sigma^+(v^*,i), \alpha(K) \rangle$, $\kappa(K,(u,j))$, and $\mu((v,i')) = \langle \lambda_1(v,i'), \lambda_2(v,i') \rangle$, the identity tester proceeds as follows. First, the tester examines $\lambda_1((v,i'))$ and $\lambda_1((v^*,i))$ to determine if $i' = i$ or $i' = i-1$. If neither of these is the case, then $(u,j) \neq (v,i')$ so $I(\mu(K), \kappa(K,(u,j)), \mu((v,i'))) := 0$ since $j \in \{i-1,i\}$. If $i' \in \{i-1,i\}$, then the tester locates the entry $\langle |\sigma(u,i)|, \varphi(u) \rangle$ in $\alpha(K)$ using $\kappa(K,(u,j))$. The location of this entry also determines whether $j = i-1$ or $j = i$. Since the tester now knows the values of j and i', it can immediately determine that $I(\mu(K), \kappa(K,(u,j)), \mu((v,i'))) := 0$ if $j \neq i'$. If $j = i'$, then the tester computes $\sigma(u,i)$ using $\sigma^+(v^*,i)$ and $|\sigma(u,i)|$. Recall that $\lambda_2(v,i') = \langle \sigma^+(v,i'), |\sigma(v,i')|, \varphi(v), \alpha(v,i'), \tau(v,i'), |\sigma(v,i'+1)| \rangle$. There are now two cases to consider:

- If $i' = i$ then the tester uses $\sigma^+(v,i')$ and $|\sigma(v,i')|$ to compute $\sigma(v,i') = \sigma(v,i)$. If $(\sigma(v,i), \varphi(v)) = (\sigma(u,i), \varphi(u))$ then $(v,i) = (u,i)$ and $I(\mu(K), \kappa(K,(u,j)), \mu((v,i'))) := 1$. Otherwise, $I(\mu(K), \kappa(K,(u,j)), \mu((v,i'))) := 0$.
- If $i' = i-1$ then the tester computes $\sigma^+(v,i) = J(\sigma^+(v,i'), \tau(v,i'))$ and uses this with $|\sigma(v,i'+1)| = |\sigma(v,i)|$ to compute $\sigma(v,i)$. Again, the tester concludes by checking if $(\sigma(v,i), \varphi(v)) = (\sigma(u,i), \varphi(u))$.

This concludes the proof. □