Slim Tree-Cut Width

Robert Ganian ✉ 🅾

Algorithms and Complexity Group, TU Wien, Austria

Viktoriia Korchemna ✉ 🅾

Algorithms and Complexity Group, TU Wien, Austria

──── **Abstract** ────────────────────────────────

Tree-cut width is a parameter that has been introduced as an attempt to obtain an analogue of treewidth for edge cuts. Unfortunately, in spite of its desirable structural properties, it turned out that tree-cut width falls short as an edge-cut based alternative to treewidth in algorithmic aspects. This has led to the very recent introduction of a simple edge-based parameter called edge-cut width [WG 2022], which has precisely the algorithmic applications one would expect from an analogue of treewidth for edge cuts, but does not have the desired structural properties.

In this paper, we study a variant of tree-cut width obtained by changing the threshold for so-called thin nodes in tree-cut decompositions from 2 to 1. We show that this "slim tree-cut width" satisfies all the requirements of an edge-cut based analogue of treewidth, both structural and algorithmic, while being less restrictive than edge-cut width. Our results also include an alternative characterization of slim tree-cut width via an easy-to-use spanning-tree decomposition akin to the one used for edge-cut width, a characterization of slim tree-cut width in terms of forbidden immersions as well as an approximation algorithm for computing the parameter.

1 Introduction

Understanding which structural properties of inputs allow us to overcome the inherent intractability of problems of interest is a fundamental research area in computer science. In the context of parameterized complexity, one typically approaches this by asking which structural parameters of the input (or its graph representation) give rise to a fixed-parameter algorithm for a targeted problem. Treewidth [35] is the most prominent example of such a structural parameter, and can be viewed as a guarantee that a graph is iteratively decomposable along small vertex separators. Many problems are known to be fixed-parameter tractable when parameterized by treewidth – and for those that are not, there is a well-studied hierarchy of more restrictive[1] parameters based on vertex separators or vertex deletion that can sometimes be used instead (see, e.g., Figure 1 in [3]). Examples of such parameters include the vertex cover number [11, 14], the feedback vertex number [2, 27] and treedepth [19, 26, 31, 32].

However, such vertex based parameters seem ill suited for handling some problems. Consider, for instance, the classical EDGE DISJOINT PATHS problem (EDP): unlike VERTEX DISJOINT PATHS, EDP remains NP-hard not only on graphs of bounded treewidth, but

---

[1] We view parameter α as being more restrictive than parameter β if every graph class where α is bounded also has bounded β, but the opposite does not hold.

even on graphs with a vertex cover number of at most 3 [13]. While this effectively rules out the use of all parameters based on vertex separators, there is an intuitive expectation that EDP should be fixed-parameter tractable w.r.t. parameters that can guarantee an iterative decomposition of the graph along small edge cuts. Indeed, EDP is known to be fixed-parameter tractable w.r.t. two basic parameterizations which provide such a guarantee: the feedback edge number [20] and treewidth plus maximum degree [21].
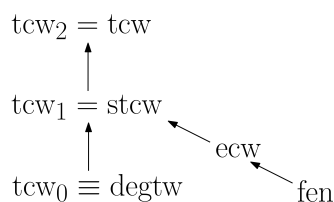
An ideal solution for handling such problems on more general inputs would be to use an alternative to treewidth that would be designed around edge cuts rather than vertex separators, one which would provide a unified justification for tractability w.r.t. the two basic "edge-cut restricting" parameterizations mentioned above. A candidate for such a parameter was proposed by Wollan, who defined *tree-cut width* along with *tree-cut decompositions* and described these as a variation of tree decompositions based on edge cuts instead of vertex separators [37]. But while it is true that "tree-cut decompositions share many of the natural properties of tree decompositions" [30], from the perspective of algorithmic design tree-cut width seems to behave differently than an edge-cut based alternative to treewidth. Indeed, not only does it fall short of yielding a fixed-parameter algorithm for EDP [20], it also fails to provide such algorithms for other problems one would expect to be fixed-parameter tractable w.r.t. an edge-cut based analogue to treewidth. In fact, out of twelve such problems where a tree-cut width parameterization has been pursued so far, only four are fixed-parameter tractable [16, 17] while eight turn out to be W[1]-hard [5, 16, 18, 20, 24] (see the Related Work at the end of the Introduction for details).

Very recently, Brand, Ceylan, Ganian, Hatschka and Korchemna [4] introduced a parameter called *edge-cut width* which aimed at filling this gap in our understanding of edge-cut based graph parameters. On the algorithmic side, edge-cut width has precisely the properties one could hope to see in an edge-based analogue to treewidth: not only does it yield fixed-parameter algorithms for all twelve "candidate" problems [4], but it is also based on a very simple type of decomposition that is much easier to use than tree-cut decompositions. That being said, already the authors of that paper noted that the structural properties of edge-cut width are far from ideal – for instance, it is the only algorithmically used parameter we are aware of that is not closed under vertex deletion. Moreover, while edge-cut width is less restrictive than the feedback edge number, unlike tree-cut width it is incomparable to treewidth plus maximum degree (even in an asymptotic sense). Because of this, it cannot act as a common generalization that would capture both of these basic approaches of enforcing decomposability along small edge cuts.

Contribution.   In this paper, we identify a graph parameter which combines the advantages of tree-cut width and edge-cut width while avoiding all of the shortcomings listed above. However, before we introduce it, it will be useful to establish at least some intuitive understanding of tree-cut width[2].

A graph G has tree-cut width at most k if it admits a tree-cut decomposition T of width k, whereas T is a rooted tree and its nodes act as bags that form a partitioning of $V(G)$. A non-root node t of T defines an edge cut between all vertices in the subtree rooted at t, and the rest of the graph. The definition of tree-cut width then restricts, for each node t, the number of its children defining an edge cut of size greater than 2. The constant "2" here arises from the structural properties Wollan aimed for when defining tree-cut width [37]; however, let us now pose the following question: How would the parameter change if we used a different constant c here instead?

---

[2] Formal definitions are provided in Section 2.

$$\text{tcw}_2 = \text{tcw}$$
$$\uparrow$$
$$\text{tcw}_1 = \text{stcw}$$
$$\text{ecw}$$
$$\uparrow \qquad \downarrow$$
$$\text{tcw}_0 \equiv \text{degtw} \qquad \text{fen}$$

Figure 1 Hierarchy of graph parameters based on edge cuts. Here ecw denotes edge-cut width and degtw denotes treewidth plus maximum degree. tcw_i denotes the parameter obtained from tree-cut width by setting the constant c described above to i. An arrow from p to q represents the fact that p is more restrictive than q, while asymptotic equivalence is depicted by $\equiv$.

On one hand, it is not difficult to observe that values of $c > 2$ would immediately lead to parameters without the properties we are aiming for, since these would be constant for, e.g., all 3-regular graphs. On the other hand, we show that for $c = 0$, one obtains an asymptotically equivalent characterization of one of the previously mentioned basic edge-cut restricting parameterizations: treewidth plus maximum degree. Our parameter of interest is then the outcome of setting $c = 1$; since this can be viewed as a variant of tree-cut width where all but a few children of each node need to have "even slimmer" edge-cuts, we refer to it as *slim tree-cut width* (stcw).

On the structural side, we show that stcw inherits the desirable properties of its "non-slim" namesake. In particular, unlike edge-cut width [4], stcw is closed under edge sums, vertex and edge deletion, as well as under the graph immersion operation. Similarly as Wollan did for tree-cut width [37], we also provide a set of forbidden immersions asymptotically characterizing stcw. Furthermore, we show that stcw is a common generalization of edge-cut width (and hence the feedback edge number), and treewidth plus maximum degree (see Figure 1).

Next, as one of our arguably most surprising results, we show that stcw is asymptotically equivalent to a slight generalization of edge-cut width: instead of measuring the width over the input graph G, we ask for the minimum edge-cut width of any supergraph of G. The transformation between these parameters is constructive and has interesting algorithmic implications. First of all, when designing algorithms it allows us to avoid the use of often cumbersome tree-cut decompositions, and instead opt for the simpler decompositions used for edge-cut width – which are nothing else than spanning trees (in this case of a supergraph). Second, all of the fixed-parameter algorithms recently designed for edge-cut width [4] rely on a dynamic programming traversal of the spanning tree, and can be straightforwardly adapted to work on spanning trees of supergraphs instead. This means that one can essentially reuse the same proofs to establish fixed-parameter tractability of all considered "candidate" problems w.r.t. stcw.

Naturally, a crucial prerequisite for algorithmically applying stcw is that we can actually compute it, or more precisely compute a suitable decomposition for graphs of small stcw. While the problem of computing an optimal decomposition remains open even for tree-cut width, a fixed-parameter approximation algorithm was obtained by Kim, Oum, Paul, Sau and Thilikos [28] and this suffices for the purposes of establishing fixed-parameter tractability. We obtain a similar outcome here and also provide a fixed-parameter approximation algorithm for stcw, albeit with a worse approximation factor than for tree-cut width.

■ **Table 1** The twelve candidate problems and their complexity w.r.t. edge-cut based parameters, where degtw denotes the maximum degree plus treewidth. Slim tree-cut width provides a unified explanation for why these problems are FPT w.r.t. both edge-cut width and degtw, and lifts these results to more general inputs.

Problem	tree-cut width	edge-cut width	degtw	stcw
CAPACITATED VERTEX COVER	FPT [16]	FPT	FPT	FPT
CAPACITATED DOMINATING SET	FPT [16]	FPT	FPT	FPT
IMBALANCE	FPT [16]	FPT	FPT	FPT
BOUNDED DEGREE DELETION	FPT [17]	FPT	FPT	FPT
EDGE DISJOINT PATHS	W[1]-hard [20]	FPT [4]	FPT [21]	FPT
LIST COLORING	W[1]-hard [16]	FPT [4]	FPT [16]	FPT
PRECOLORING EXTENSION	W[1]-hard [16]	FPT [4]	FPT [16]	FPT
BOOLEAN CONSTRAINT SATISFACTION	W[1]-hard [16]	FPT [4]	FPT [36]	FPT
BAYESIAN NETWORK STRUCTURE LEARNING	W[1]-hard [18]	FPT [4,18]	FPT [33]	FPT
POLYTREE LEARNING	W[1]-hard [18]	FPT [4,18]	FPT [18]	FPT
MIN. CHANGEOVER COST ARBORESCENCE	W[1]-hard [24]	FPT [4]	FPT [25]	FPT
MSRTIL[3]	W[1]-hard [5]	FPT [4]	FPT [1,5]	FPT

Related Work.   Tree-cut width parameterizations were typically considered for problems which are not fixed-parameter tractable (FPT) w.r.t. treewidth, but are FPT w.r.t. feedback edge number and also FPT w.r.t. treewidth plus maximum degree. The twelve candidate problems where tree-cut width parameterizations have been considered are shown in Table 1.

The structural properties of tree-cut width have also been studied in a number of recent papers [22, 23]. Last but not least, we note that a preprint exploring a different parameter that is aimed at providing an edge-based alternative to treewidth was recently authored by Magne, Paul, Sharma and Thilikos [29]; the parameter is based on different ideas and is incomparable to both tree-cut width and slim tree-cut width.

2   Preliminaries

We use standard terminology for graph theory [9] and assume basic familiarity with the parameterized complexity paradigm including, in particular, the notions of *fixed-parameter tractability* and W[1]-*hardness* [8, 10]. Let $\mathbb{N}$ denote the set of natural numbers including zero. We use $[i]$ to denote the set $\{0, 1, \ldots, i\}$.

The *(open) neighborhood* of a vertex $x \in V(G)$ is the set $\{y \in V(G) \mid xy \in E(G)\}$ and is denoted by $N_G(x)$. For a vertex subset X, the neighborhood of X is defined as $\bigcup_{x \in X} N_G(x) \setminus X$ and denoted by $N_G(X)$; we drop the subscript if the graph is clear from the context. If H is a subgraph of G, we denote it by $H \subseteq G$. *Contracting* an edge $\{a, b\}$ is the operation of replacing vertices a, b by a new vertex whose neighborhood is $(N(a) \cup N(b)) \setminus \{a, b\}$. For a vertex set A (or edge set B), we use $G - A$ ($G - B$) to denote the graph obtained from G by deleting all vertices in A (edges in B), and we use $G[A]$ to denote the *subgraph induced on* A, i.e., $G - (V(G) \setminus A)$.

---

[3] Maximum Stable Roommates with Ties and Incomplete Lists. For completeness, we note that the authors who showed W[1]-hardness w.r.t. tree-cut width also identified two additional restrictions which, when combined with tree-cut width, suffice for fixed-parameter tractability [5].

Let G be a graph and let x, y and z be three distinct vertices of G such that $(x,y),(y,z) \in E(G)$. To *lift* the pair of edges $(x,y),(y,z)$ means to delete the edges (x,y) and (y,z) from G and add (if it doesn't exist yet) a new edge (x,z). We say that G contains H as a *weak immersion* (denoted $H \leq_I G$) if and only if H can be obtained from G by a sequence of edge deletion, vertex deletion, and lifting operations.

For a natural number k, we say that a graph G is a *k-edge sum* of vertex-disjoint graphs G_1 and G_2 if there exist vertices $v_i \in V(G_i)$ of degree k for $i = 1, 2$ and a bijection $\pi : N_{G_1}(v_1) \to N_{G_2}(v_2)$ such that G is obtained from $(G_1 - \{v_1\}) \cup (G_2 - \{v_2\})$ by adding an edge $(v, \pi(v))$ for every $v \in N_{G_1}(v_1)$. In this case we write $G = G_1 \oplus_k G_2$. Observe that the same pair of graphs may produce different k-edge sums.

Given two graph parameters $\alpha, \beta : G \mapsto \mathbb{N}$, we say that α *dominates* β if there exists a function p such that for each graph G, $\alpha(G) \leq p(\beta(G))$. If α dominates β but β does not dominate α, we often say that β is more restrictive than α; as an example, treewidth dominates the vertex cover number. Two parameters that dominate each other are called asymptotically equivalent.

Tree-cut Width. The notion of tree-cut decompositions was introduced by Wollan [37], see also subsequent work by Marx and Wollan [30]. A family of subsets $X_1, \ldots, X_k$ of X is a *near-partition* of X if they are pairwise disjoint and $\bigcup_{i=1}^{k} X_i = X$, allowing the possibility of $X_i = \emptyset$.

▶ **Definition 1.** *A* tree-cut decomposition *of G is a pair $(T, \mathcal{X})$ which consists of a rooted tree T and a near-partition $\mathcal{X} = \{X_t \subseteq V(G) \mid t \in V(T)\}$ of $V(G)$. A set in the family $\mathcal{X}$ is called a* bag *of the tree-cut decomposition.*

For any node t of T other than the root r, let $e(t) = ut$ be the unique edge incident to t on the path to r. Let T_u and T_t be the two connected components in $T - e(t)$ which contain u and t, respectively. Note that $(\bigcup_{q \in T_u} X_q, \bigcup_{q \in T_t} X_q)$ is a near-partition of $V(G)$, and we use E_t to denote the set of edges with one endpoint in each part. We define the *adhesion* of t (adh(t)) as $|E_t|$; we explicitly set adh$(r) = 0$ and $E(r) = \emptyset$. The adhesion of $(T, \mathcal{X})$ is then adh$(T, \mathcal{X}) = \max_{t \in V(T)} \text{adh}(t)$.

The *torso* of a tree-cut decomposition $(T, \mathcal{X})$ at a node t, written as H_t, is the graph obtained from G as follows. If T consists of a single node t, then the torso of $(T, \mathcal{X})$ at t is G. Otherwise, let $T_1, \ldots, T_\ell$ be the connected components of $T - t$. For each $i = 1, \ldots, \ell$, the vertex set $Z_i \subseteq V(G)$ is defined as the set $\bigcup_{b \in V(T_i)} X_b$. The torso H_t at t is obtained from G by *consolidating* each vertex set Z_i into a single vertex z_i (this is also called *shrinking* in the literature). Here, the operation of consolidating a vertex set Z into z is to substitute Z by z in G, and for each edge e between Z and $v \in V(G) \setminus Z$, adding an edge zv in the new graph. We note that this may create parallel edges.

The operation of *suppressing* (also called *dissolving* in the literature) a vertex v of degree at most 2 consists of deleting v, and when the degree is two, adding an edge between the neighbors of v. Given a connected graph G and $X \subseteq V(G)$, let the *3-center* of (G, X) be the unique graph obtained from G by exhaustively suppressing vertices in $V(G) \setminus X$ of degree at most two. Finally, for a node t of T, we denote by $\tilde{H}_t$ the 3-center of (H_t, X_t), where H_t is the torso of $(T, \mathcal{X})$ at t. Let the *torso-size* tor(t) denote $|\tilde{H}_t|$.

▶ **Definition 2.** *The width of a tree-cut decomposition $(T, \mathcal{X})$ of G is $\max_{t \in V(T)}\{\text{adh}(t), \text{tor}(t)\}$. The tree-cut width of G, or tcw(G) in short, is the minimum width of $(T, \mathcal{X})$ over all tree-cut decompositions $(T, \mathcal{X})$ of G.*
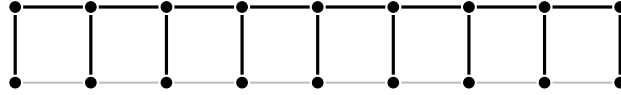
Figure 2 Example of a graph G with a spanning tree T (thick black) such that $\mathrm{ecw}(G) = \mathrm{ecw}(G, T) = 3$. The feedback edge number of G can be made arbitrarily large in this fashion.

Without loss of generality, we shall assume that $X_r = \emptyset$. We conclude this subsection with some notation related to tree-cut decompositions. Given a tree node t, let T_t be the subtree of T rooted at t. Let $Y_t = \bigcup_{b \in V(T_t)} X_b$, and let G_t denote the induced subgraph $G[Y_t]$. A node $t \neq r$ in a rooted tree-cut decomposition is *thin* if $\mathrm{adh}(t) \leq 2$ and *bold* otherwise.

A tree-cut decomposition $(T, \mathcal{X})$ is *nice* if it satisfies the following condition for every thin node $t \in V(T)$: $N(Y_t) \cap \left(\bigcup_{b \text{ is a sibling of } t} Y_b \right) = \emptyset$. The intuition behind nice tree-cut decompositions is that we restrict the neighborhood of thin nodes in a way which facilitates dynamic programming. Every tree-cut decomposition of width k can be transformed into a nice tree-cut decomposition of the same width in cubic time [16]. Moreover, the resulting nice decomposition has the following property. For a node t, let $B_t = \{b \text{ is a child of } t \mid |N(Y_b)| \leq 2 \wedge N(Y_b) \subseteq X_t\}$ denote the set of thin children of t whose neighborhood is a subset of X_t, and let $A_t = \{a \text{ is a child of } t \mid a \notin B_t\}$ be the set of all other children of t. Then $|A_t| \leq 2k + 1$ for every node t [16].

We refer to previous work [16, 28, 30, 37] for a detailed comparison of tree-cut width to other parameters. Here, we mention only that tree-cut width is dominated by treewidth and dominates treewidth plus maximum degree, which we denote $\mathrm{degtw}(G)$. It also dominates the feedback edge number (the size of a minimum feedback edge set), denoted $\mathrm{fen}(G)$.

▶ **Lemma 3** ([16, 30, 37]). *For every graph G, $\mathrm{tw}(G) \leq 2\,\mathrm{tcw}(G)^2 + 3\,\mathrm{tcw}(G)$ and $\mathrm{tcw}(G) \leq \mathrm{fen}(G) + 1$ and $\mathrm{tcw}(G) \leq 4\,\mathrm{degtw}(G)^2$.*

Edge-Cut Width.   The notion of edge-cut width was introduced by Brand at al. [4]. For a graph G and a maximal spanning forest T of G, let the *local feedback edge set* at $v \in V$ be

$$E_{\mathrm{loc}}^{G,T}(v) = \{uw \in E(G) \setminus E(T) \mid \text{ the unique path between } u \text{ and } w \text{ in } T \text{ contains } v\}.$$

▶ **Definition 4.** *The edge-cut width of the pair (G, T) is $\mathrm{ecw}(G, T) = 1 + \max_{v \in V} |E_{\mathrm{loc}}^{G,T}(v)|$, and the edge-cut width of G (denoted $\mathrm{ecw}(G)$) is the smallest edge-cut width among all possible maximal spanning forests T of G.*

▶ **Proposition 5** ([4]). *For every graph G, $\mathrm{tcw}(G) \leq \mathrm{ecw}(G) \leq \mathrm{fen}(G) + 1$.*

In fact, it was shown in [4] that the gaps in both inequalities can be arbitrary large, see Figure 2 for a simple example of the second one.

Edge-cut width is not closed under vertex or edge deletions and is incomparable to degtw [4]. However, the fact that its decomposition is simply a spanning tree makes it easier to work with in dynamic programming applications than, e.g., tree-cut decompositions [4].

3 Refined Measures for Tree-Cut Decompositions

3.1 Definitions and Comparison

Let us now define our parameter of interest, obtained by altering the threshold for when a vertex is suppressed (dissolved) in the definition of tree-cut width. Formally, let $(T, \mathcal{X})$ be some tree-cut decomposition of G. Given a connected graph Q and $X \subseteq V(Q)$, let the

2-center of (Q, X) be the unique graph obtained from Q by exhaustively deleting vertices in $V(Q) \setminus X$ of degree at most one. For a node t of T, we denote by $\bar{H}_t^2$ the 2-center of (H_t, X_t), where H_t is the torso of $(T, \mathcal{X})$ at t. Let us denote $|\bar{H}_t^2|$ by $\text{tor}_2(t)$.

▶ **Definition 6.** *The slim width of a tree-cut decomposition $(T, \mathcal{X})$ of a graph G is* $\text{stcw}(T, \mathcal{X}) = \max_{t \in V(T)}\{\text{adh}(t), \text{tor}_2(t)\}$. *The slim tree-cut width of G, or* $\text{stcw}(G)$ *in short, is the minimum slim width of $(T, \mathcal{X})$ over all tree-cut decompositions $(T, \mathcal{X})$ of G.*

Observe that the difference in definitions of $\text{tcw}(G)$ and $\text{stcw}(G)$ is whether we dissolve the vertices of degree at most two or at most one in the torso in each node. At this point, it would be reasonable to ask what happens if we dissolve only isolated vertices (i.e., vertices of degree 0) from the torso. Naturally extending the notions of 2- and 3-center for a connected graph Q and $X \subseteq V(Q)$, we define the *1-center* of (Q, X) as the graph obtained from Q by deleting isolated vertices in $V(Q) \setminus X$. For a node t of T, we denote by $\bar{H}_t^1$ the 1-center of (H_t, X_t), where H_t is the torso of $(T, \mathcal{X})$ at t. Let us denote $|\bar{H}_t^1|$ by $\text{tor}_1(t)$.

▶ **Definition 7.** *The 0-width of a tree-cut decomposition $(T, \mathcal{X})$ of G is $\max_{t \in V(T)}\{\text{adh}(t), \text{tor}_1(t)\}$. The 0-tree-cut width of G, or $\text{tcw}_0(G)$ in short, is the minimum 0-width of $(T, \mathcal{X})$ over all tree-cut decompositions $(T, \mathcal{X})$ of G.*

It follows from the definitions that for any tree-cut decomposition $(T, \mathcal{X})$ of G, for each node t of T, $\text{tor}(t) \leq \text{tor}_2(t) \leq \text{tor}_1(t)$. In particular, the width of $(T, \mathcal{X})$ is upper-bounded by its slim width, while the latter does not exceed the 0-width of $(T, \mathcal{X})$.

▶ **Corollary 8.** *For any graph G, $\text{tcw}(G) \leq \text{stcw}(G) \leq \text{tcw}_0(G)$.*

The gaps in these inequalites can be arbitrarily large – and, more strongly, tcw_0 is a more restrictive parameter than stcw, which is in turn more restrictive than tcw. Indeed, for the comparison of tcw_0 and stcw consider the class of stars which have slim tree-cut width 1. Let S_r denote the star with r leaves (i.e., the complete bipartite graph $K_{1,r}$).

▶ **Lemma 9.** *For every positive integer $r \geq 1$, $\text{tcw}_0(S_{r^2}) \geq r$.*

Proof. Let $(T, \mathcal{X})$ be a tree-cut decomposition of S_{r^2} of 0-width k where the bags of leaves are non-empty. Let t be the node of T such that X_t contains the vertex of degree r^2. Observe that t has at most $\text{tor}_1(t) - |X_t| \leq k - |X_t|$ children. For every child t' of t, $Y_{t'}$ contains at most $\text{adh}(t') \leq k$ vertices of S_{r^2}. In total, Y_t contains at most $|X_t| + k \cdot (k - |X_t|) \leq k^2$ vertices of S_{r^2}. Together with at most $\text{adh}(t) \leq k$ vertices outside of Y_t, S_{r^2} has at most $k \cdot (k + 1)$ vertices and hence $k \geq r$. ◀

To show the gap between stcw and tcw, let us denote by W_r the graph on $2r + 1$ vertices consisting of r triangles sharing one vertex; here we call such graphs windmills, and refer to Figure 3 later for an illustration. The class of windmills has tree-cut width 2 but, as the following lemma shows, unbounded slim tree-cut width.

▶ **Lemma 10.** *For every positive integer $r \geq 1$, $\text{stcw}(W_{r^2}) \geq r$.*

Proof. The case $r = 1$ is straightforward. For $r \geq 2$, assume, to the contrary, that there exists a tree-cut decomposition $(T, \mathcal{X})$ of W_{r^2} of slim width at most $r - 1$. Let t be the node of T such that X_t contains the vertex of degree $2r^2$. Without loss of generality, we assume that all the leaves of T have non-empty bags. Then the adhesion of any child t' of t is at least two, as $Y_{t'}$ contains some vertex v of W_{r^2} and the two edge-disjoint paths from v to the high-degree vertex in t each contribute to $\text{adh}(t')$. Hence, t has at most $\text{tor}_2(t) \leq r - 1$

children. Moreover, for every child t' of t, $Y_{t'}$ intersects at most $\frac{r-1}{2}$ distinct triangles of W_{r^2}, since each such triangle contributes 2 to $\mathrm{adh}(t')$. Hence, for every child t' of t, $Y_{t'}$ contains at most $r-1$ vertices of W_{r^2}. In total, $Y_t \setminus X_t$ contains at most $(r-1)^2$ vertices of W_{r^2}. Since both $\mathrm{adh}(t)$ and $|X_t|$ are upper-bounded by $r-1$ and the former bounds the number of vertices outside of Y_t by $r-1$, this would mean that W_{r^2} has at most $(r-1)^2 + 2r - 2$ vertices, a contradiction with the definition of W_{r^2}. ◀

Given a graph G and its nice tree-cut decomposition $(T, \mathcal{X})$ of width at most k, let us denote by $B_t^{(2)}$ the set of children of t from B_t with adhesion precisely two; notice that $B_t^{(2)}$ does not necessarily contain all children of t with adhesion precisely two, since some may lie in A_t. Observe that for every fixed vertex t of T, if x is an element of 2-center of the torso at t and $x \notin X_t$, then x corresponds either to the parent of t in T or to some child of t from $A_t \cup B_t^{(2)}$. Hence $\mathrm{tor}_2(t) \leq 1 + |X_t| + |A_t| + |B_t^{(2)}| \leq 3k + 2 + |B_t^{(2)}|$.

▶ **Corollary 11.** *Let G be a graph with tree-cut decomposition $(T, \mathcal{X})$ of width at most k. Then for each node t of T it holds that $|B_t^{(2)}| \geq \mathrm{tor}_2(t) - 3k - 2$.*

3.2   Weak Immersions

Naturally extending the result of Wollan for tree-cut width [37], we show that both slim and 0-tree-cut width are closed under weak immersions.

▶ **Theorem 12.** *If G and H are graphs such that $H \leq_I G$ then $\mathrm{stcw}(H) \leq \mathrm{stcw}(G)$ and $\mathrm{tcw}_0(H) \leq \mathrm{tcw}_0(G)$.*

Proof. It is sufficient to proof the statement when H is obtained from G by precisely one edge deletion, isolated vertex deletion or lifting a pair of edges. Let $(T, \mathcal{X})$ be a tree-cut decomposition of G of minimum slim (or 0-) width. Then $(T, \mathcal{X})$ is also a tree-cut decomposition of $G \setminus e$ for any edge e of G with the same or smaller slim (0-) width. Similarly for the isolated vertex deletion: we just need to delete the vertex from the corresponding bag. It remains to consider the case $H = G \setminus \{(x, y), (y, z)\} \cup (x, z)$ for some $(x, y), (y, z) \in E(G)$.

Notice that the lifting operation doesn't increase adhesion of any node t of T: if the edge (x, z) has endpoints in different connected components of $T \setminus e(t)$ then so does at least one of the edges (x, y) or (y, z). To see that $\mathrm{tor}_2(t)$ and $\mathrm{tor}_1(t)$ do not increase either, denote by Q_G and Q_H the torsos at t in $(T, \mathcal{X})$ for graphs G and H correspondingly. Every vertex of Q_G corresponds to a non-empty subset of the vertices of G. Depending on how the vertices x, y and z are split among these subsets, it holds that either $E(Q_H) \subseteq E(Q_G)$ (which yields the same or smaller 1-center and 2-center) or Q_H is obtained from Q_G by splitting a pair of edges. For the latter, observe that $v \in V(Q_G) \setminus X_t$ is not in the 2-center of (Q_G, X_t) if and only if v belongs to some induced subtree of Q_G connected to the rest of Q_G by at most one edge. It is not hard to see that lifting the pair of edges preserves the property. For the 1-center the situation is even simpler: isolated vertices of Q_G remain isolated. ◀

Recall that the weak immersion relation $\leq_I$ is a transitive, reflexive and antisymmetric relation on the set of finite graphs, i.e., a partial order. The previous theorem showed that stcw is monotone with respect to $\leq_I$. Our next goal is to find graphs of simple structure but large slim (or 0-) tree-cut width, such that forbidding them as weak immersions bounds the corresponding width of a graph. Wollan in [37] characterized such graphs for tree-cut width. Namely, he established the following dichotomy:

▶ **Theorem 13.**
(a) *If G is a graph such that $H_{2r^2} \leq_I G$ for some $r \geq 3$, then $\mathrm{tcw}(G) \geq r$.*
(b) *There exists a function $f : \mathbb{N} \to \mathbb{N}$ such that if $\mathrm{tcw}(G) \geq f(r)$, then $H_r \leq_I G$, $r \in \mathbb{N}$.*

Here H_r denotes the r-wall, the graph which can be obtained from the $r \times r$ grid by deleting every second vertical edge in each row, see [37] for the definition and Figure 3 for an illustration. We are going to complete the family of excluded immersions to provide similar characterizations for 0-tree-cut width and slim tree-cut width. Recall that the families of stars S_r and windmils W_r have unbounded 0- and slim tree-cut width, respectively (Lemmas 9 and 10). Combining this with Theorem 12, we immediately obtain:

▶ **Lemma 14.** *For every positive integer r, if* $\mathrm{stcw}(G) < r$ *(*$\mathrm{tcw}_0(G) < r$*), then G does not admit W_{r^2} (S_{r^2}, respectively) as a weak immersion.*

As we will show in the remainder of this subsection, excluding W_r (S_r) as a weak immersion along with H_r is actually sufficient to bound slim tree-cut width (0-tree-cut width).

▶ **Theorem 15.** *If G is a graph such that $H_{2r^2} \leq_I G$ for some $r \geq 3$ or $S_{r^2} \leq_I G$ for some $r \geq 1$, then $\mathrm{tcw}_0(G) \geq r$. Moreover, there exists a function $h : \mathbb{N} \to \mathbb{N}$ such that if $\mathrm{tcw}_0(G) \geq h(r)$, then $H_r \leq_I G$ or $S_r \leq_I G$.*

Proof. If $H_{2r^2} \leq_I G$ for some $r \geq 3$, we have that $\mathrm{tcw}(G) \geq r$ by Theorem 13 and hence $\mathrm{tcw}_0(G) \geq r$. In case $S_{r^2} \leq_I G$, the lower bound follows from Lemma 14.

Let f be the function given by Theorem 13. We define h by setting $h(r) = r \cdot f(r) + 3 \cdot f(r) + 2$. Assume that G is a graph such that $\mathrm{tcw}_0(G) \geq h(r)$. If $\mathrm{tcw}(G) \geq f(r)$, we immediatedly conclude that $H_r \leq_I G$ by Theorem 13. Otherwise, let $(T, \mathcal{X})$ be a nice tree-cut decomposition of G of width at most $f(r)$ with leaves having non-empty bags. There exists a node t of T such that $\mathrm{tor}_1(t) \geq h(r)$, in particular, $B_t \geq r \cdot f(r)$. As the size of X_t is at most $f(r)$, some vertex of X_t has degree of at least r and hence $S_r \leq_I G$. ◀

Before providing similar characterization for slim tree-cut width, we introduce a simple technical modification of tree-cut decompositions, which will also be used later for establishing the connection between slim tree-cut width and edge-cut width. The aim is, roughly speaking, to avoid the situation where a thin child has adhesion 2, even though it consists of two completely independent components each of which could be a thin child of adhesion 1. Formally, let $(T, \mathcal{X})$ be a nice tree-cut decomposition of G. We say that a node t with parent t' in T is *decomposable* if the following conditions hold:

- $t \in B_{t'}$ and there exist two edges e_1 and e_2 between G_t and $G \setminus G_t$ in G;
- the endpoints of e_1 and e_2 in G_t belong to different connected components of G_t.

▶ **Lemma 16.** *Any nice tree-cut decomposition of G can be transformed into a nice tree-cut decomposition of the same tree-cut width with no decomposable nodes.*
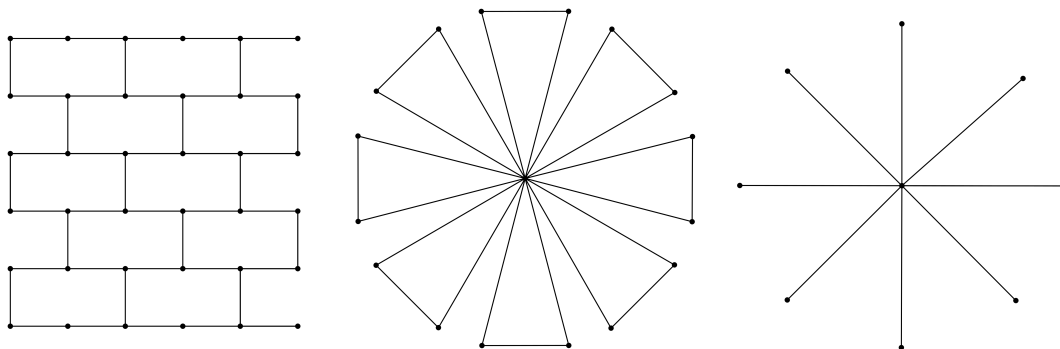
Figure 3 Illustrations of forbidden weak immersions for the graphs with bounded standard, slim or 0-tree-cut width. Left: 6-wall H_6, Middle: windmill W_8, Right: star S_8.

Proof. Let $(T', \mathcal{X}')$ be a nice tree-cut decomposition of G with at least one decomposable node. Let t be a decomposable node of T' with minimum distance to the root, and let e_1 and e_2 be the edges between G_t and $G \setminus G_t$ in G. We create a copy T'_{t^*} of the rooted subtree T'_t where the copy of $s \in T'_t$ is $s^* \in T'_{t^*}$. We then connect t^* to the parent of t. Let G_1 be the connected component of G_t containing an endpoint of e_1. For every $s \in V(T'_t)$ we set $X_s = X'_s \cap V(G_1)$ and $X_{s^*} = X'_s \setminus X_s$. For the rest of nodes s of T' we set $X_s = X'_s$. Finally, we exhaustively remove empty bags which are leaves and denote the obtained tree by T. Observe that the resulting decomposition $(T, \mathcal{X})$ is nice and its width is not greater than the width of $(T', \mathcal{X}')$. Moreover, our transformation doesn't create any decomposable nodes outside of subtrees rooted in t and t^*; both t and t^* have an adhesion of one and hence are not decomposable. Therefore, after a finite number of such steps we obtain some nice tree-cut decomposition of G of the same width but with no decomposable nodes. ◀

Further, as a technical term, we will refer to nice decompositions with no decomposable nodes as *very nice* decompositions.

▶ **Corollary 17.** *Every tree-cut decomposition can be transformed into a very nice tree-cut decomposition in quartic time, without increasing the width.*

Proof. Let $(T'', \mathcal{X}'')$ be a tree-cut decomposition of G of width k. We transform $(T'', \mathcal{X}'')$ into a nice tree-cut decomposition $(T', \mathcal{X}')$ of width at most k (this can be done in cubic time, see [16] for details). Further, we apply Lemma 16 on $(T', \mathcal{X}')$. This requires at most quartic time, since every node of T' is decomposed at most once and every such decomposition can be performed in cubic time. Then the resulting decomposition $(T, \mathcal{X})$ is very nice and has width of at most k. ◀

With this transformation in hand, we are now ready to fully characterize forbidden weak immersions for graphs of bounded slim tree-cut width.

▶ **Theorem 18.** *If G is a graph such that $H_{2r^2} \leq_I G$ for some $r \geq 3$ or $W_{r^2} \leq_I G$ for some $r \geq 1$, then $\mathrm{stcw}(G) \geq r$. Moreover, there exists a function $g : \mathbb{N} \to \mathbb{N}$ such that if $\mathrm{stcw}(G) \geq g(r)$, then $H_r \leq_I G$ or $W_r \leq_I G$.*

Proof. If $H_{2r^2} \leq_I G$ for some $r \geq 3$, we have that $\mathrm{tcw}_2(G) \geq r$ by Theorem 13 and hence $\mathrm{stcw}(G) \geq r$. In case $W_{r^2} \leq_I G$, the lower bound follows from Lemma 14.

Let f be the function given by Theorem 13. We define g by setting $g(r) = 2r \cdot f^2(r) + 3 \cdot f(r) + 2$. Assume that G is a graph such that $\mathrm{stcw}(G) \geq g(r)$. If $\mathrm{tcw}(G) \geq f(r)$, we immediately conclude that $H_r \leq_I G$ by Theorem 13. Otherwise, by Corollary 17 there exists a very nice tree-cut decomposition $(T, \mathcal{X})$ of G of width at most $f(r)$. Let us pick a node t of T such that $\mathrm{tor}_2(t) \geq g(r)$. By Corollary 11 we have that $|B_t^{(2)}| \geq g(r) - 3 \cdot f(r) - 2 = 2r \cdot f^2(r)$. Since $(T, \mathcal{X})$ is very nice, all the children of t in $B_t^{(2)}$ are non-decomposable. Recall that for every $t' \in B_t^{(2)}$, the neighbourhood of $Y_{t'}$ in G is a one- or two-element subset of X_t, and hence $Y_{t'}$ provides a path between some (possibly equal) vertices of X_t. As the size of X_t is at most $f(r)$, G contains either $2r$ cycles intersecting in one vertex of X_t or $2r$ paths between two vertices of X_t. Since every such pair of paths can be transformed into a cycle by lifting the pair of their first edges, in both cases we have $W_r \leq_I G$. ◀

3.3 k-Edge Sums

Another natural property Wollan [37] established for tree-cut width is that the parameter is closed under the operation of taking k-edge sum for small k. Specifically, he proved:

▶ **Lemma 19** ([37])**.** *Let G, G_1, and G_2 be graphs such that $G = G_1 \oplus_k G_2$. If G_j has a tree-cut decomposition $(T_j, \mathcal{X}_j)$ for $j = 1, 2$, then G has a tree-cut decomposition $(T, \mathcal{X})$ such that $\mathrm{adh}(T, \mathcal{X}) = \max\{k, \mathrm{adh}(T_1, \mathcal{X}_1), \mathrm{adh}(T_2, \mathcal{X}_2)\}$. Moreover, for every $t \in V(T)$, the torso H_t of t in $(T, \mathcal{X})$ is isomorphic to the torso of some vertex of $(T_1, \mathcal{X}_1)$ or $(T_2, \mathcal{X}_2)$.*

Based on this result for optimal decompositions $(T_1, \mathcal{X}_1)$ and $(T_2, \mathcal{X}_2)$, we immediatedly obtain the upper bound on 0- and slim tree-cut width for k-edge sums:

▶ **Corollary 20.** *Let G, G_1 and G_2 be graphs such that $G = G_1 \oplus_k G_2$. Then it holds that $\mathrm{stcw}(G) \leq \max\{k, \mathrm{stcw}(G_1), \mathrm{stcw}(G_2)\}$ and $\mathrm{tcw}_0(G) \leq \max\{k, \mathrm{tcw}_0(G_1), \mathrm{tcw}_0(G_2)\}$.*

In particular, if both G_1 and G_2 have 0-, slim or standard width of at most ω and $k \leq \omega$, we may conclude that the corresponding width of G is at most ω.

4 Alternative Characterizations

In this section, we study alternative characterizations of slim tree-cut width and 0-tree-cut width. In particular, we observe that the latter is asymptotically equivalent to maximum degree plus treewidth. This provides an interesting connection between tree decompositions and tree-cut decompositions, but essentially rules out its study as a means of establishing novel tractability results. For slim tree-cut width, however, we obtain a characterization that ties it to the previously studied edge-cut width and has algorithmic implications.

4.1 Characterization of 0-Tree-Cut Width

Wollan [37] showed that a bound on the treewidth and maximum degree implies a bound on the tree-cut width of a graph:

▶ **Proposition 21.** *Let G be a graph with maximal degree d and treewidth w. Then there exists a tree-cut decomposition of adhesion at most $(2w + 2)d$ such that every torso has at most $(d + 1)(w + 1)$ vertices.*

In particular, as $\mathrm{tor}_1(t) \leq |H_t| \leq (d + 1)(w + 1) \leq (2w + 2)d$ for every node t of T, we have $\mathrm{tcw}_0(G) \leq (2w + 2)d$. In the following proposition, we show that the converse is true as well: bounded tcw_0 implies bounded treewidth and maximum degree of a graph.

▶ **Proposition 22.** *Let G be a graph with $\mathrm{tcw}_0(G) = k$. Then every vertex of G has degree of at most $k^2 + 2k$ and $\mathrm{tw}(G) \leq 2k^2 + 3k$.*

Proof. By Lemma 3 and Corollary 8 we have that $\mathrm{tw}(G) \leq 2\,\mathrm{tcw}(G)^2 + 3\,\mathrm{tcw}(G) \leq 2k^2 + 3k$. Since $\mathrm{tcw}_0(G) = k$, Lemma 9 implies that G does not contain $S_{(k+1)^2}$ as a weak immersion, in particular, degree of any vertex of G is at most $k^2 + 2k$. ◀

▶ **Corollary 23.** *0-tree-cut width is asymptotically equivalent to maximum degree plus treewidth.*

4.2 Characterization of Slim Tree-Cut Width

Recall that edge-cut width is a parameter that is defined over spanning trees in the input graph G, which serve as the corresponding decompositions. Let us now consider a slight generalization of this where we consider not only spanning trees over G, but of any supergraph

of G. Such a generalization would – unlike edge-cut width itself – trivially be closed under both vertex and edge deletion. For our considerations, let us denote this parameter *super edge-cut width* $(\sec(G))$:

$$\sec(G) = \min\{\mathrm{ecw}(H,T) \mid H \supseteq G \text{ and } T \text{ is a spanning forest of } H\}.$$

If $H \supseteq G$ is a supergraph of G and T is a spanning forest of H such that $\mathrm{ecw}(H,T) \leq k$, we say that T *witnesses* $\sec(G) \leq k$. Observe that there always exists a connected witness, i.e., a tree. Indeed, if H consists of $m > 1$ connected components, we can arbitrarily extend it to a connected graph H^* by adding $m - 1$ edges. The addition of these edges to T then results in the tree T^* witnessing $\sec(G) \leq k$. Moreover, notice that any witness of $\mathrm{ecw}(G) \leq k$ is also a witness of $\sec(G) \leq k$.

▶ **Corollary 24.** *For every graph G, $\sec(G) \leq \mathrm{ecw}(G)$.*

However, graphs of constant super edge-cut width can have arbitrarily large edge-cut width, as will become clear at the end of the section. A slight modification of the proof of Proposition 5 yields:

▶ **Proposition 25.** *For every graph G, $\mathrm{tcw}(G) \leq \sec(G)$.*

Proof. Let Q be the supergraph of G and let T be the spanning tree of Q such that $\mathrm{ecw}(Q,T) = \sec(G)$. We construct a tree-cut decomposition $(T,\mathcal{X})$ of G where each bag contains at most one vertex, notably by setting $X_t = \{t\}$ for each $t \in V(G)$ and $X_t = \emptyset$ for each $t \in V(Q) \setminus V(G)$. Fix any node t in T other than the root, let u be the parent of t in T. All the edges of $G \setminus ut$ with one endpoint in the rooted subtree T_t and another outside of T_t belong to $E_{loc}^{Q,T}(t)$, so $\mathrm{adh}_T(t) \leq |E_{loc}^{Q,T}(t)| + 1 \leq \sec(G)$.

Let H_t be the torso of $(T,\mathcal{X})$ in t, then $V(H_t) = X_t \cup \{z_1...z_l\}$ where z_i correspond to connected components of $T \setminus t$, $i \in [l]$. In $\tilde{H}_t$, only z_i with degree at least 3 are preserved. But all such z_i are the endpoints of at least two edges in $|E_{loc}^{Q,T}(t)|$, so $\mathrm{tor}(t) = |V(\tilde{H}_t)| \leq 1 + |E_{loc}^{Q,T}(t)| \leq \sec(G)$. Thus $\mathrm{tcw}(G) \leq \sec(G)$. ◀

To represent a deeper connection between tree-cut decompositions and super edge-cut width, it will be convenient to work with very nice decompositions introduced in subsection 3.2.

▶ **Proposition 26.** *Let $(T,\mathcal{X})$ be a very nice tree-cut decomposition of G of width at most k. Then for each node t of T, $|B_t^{(2)}| \leq k \cdot \sec(G)$. In particular, $\mathrm{stcw}(G) \leq \sec(G)^2 + 4 \cdot \sec(G)$.*

Proof. Assume that T^* is a spanning tree of $H \supseteq G$ such that $\sec(G) = \mathrm{ecw}(H,T^*)$. For any node t of T and $b \in B_t^{(2)}$, b has one of three types (see Figure 4):
1. $N(Y_b) = \{x\}$ for some $x \in X_t$, x is connected to distinct x_b^1 and x_b^2 from Y_b;
2. $N(Y_b) = \{x_1, x_2\}$ for $x_1 \neq x_2$, x_1 and x_2 are connected to the same $x_b \in Y_b$;
3. $N(Y_b) = \{x_1, x_2\}$ for $x_1 \neq x_2$, x_1 and x_2 are connected to distinct x_b^1 and x_b^2 from Y_b correspondingly;

Let us start with the first type. If $x_b^i x$ doesn't belong to T^* for $i = 1$ or $i = 2$, then $x_b^i x \in E_{loc}^{H,T^*}(x)$. Otherwise, x_b^1 and x_b^2 are connected via x in T^*. Then $T^*[Y_b]$ has precisely two connected components. As b is not decomposable, there exists a path p between x_b^1 and x_b^2 in G_b containing precisely one edge outside of T^*. This edge contributes to $E_{loc}^{H,T^*}(x)$.

As T^* is a tree, there can be at most $|X_t| - 1 \leq k - 1$ thin children b of the second type such that x_b is adjacent to two elements of X_t in T^*. For the rest of b of the second type, there exists $x \in X_t$ such that $xx_b \in G \setminus T^* \subseteq H \setminus T^*$ and therefore $xx_b \in E_{loc}^{H,T^*}(x)$.

Let b be a thin node of the third type. If x_1^b and x_2^b are connected via a path in $T^*[Y_b]$, we can apply the same argument as for the second type. Otherwise, $T^*[Y_b]$ has precisely two connected components and, analogously to the first type, there exists an edge in $G_b \cup \{x_1 x_1^b, x_2 x_2^b\}$ that belongs to $E_{loc}^{H,T^*}(x_1)$.

To conclude, any node of $B_t^{(2)}$ either increases $E_{loc}^{H,T^*}(x)$ for some $x \in X_t$ or creates a path in T^* between two vertices of X_t. Since T^* is a tree, $|X_t| \leq k$ and $|E_{loc}^{H,T^*}(x)| \leq \sec(G) - 1$ for every $x \in X_t$, the size of $B_t^{(2)}$ is at most $(k-1) + \sum_{x \in X_t} |E_{loc}^{H,T^*}(x)| \leq k \cdot \sec(G) - 1$. Then $\mathrm{tor}_2(t) \leq |A_t| + |X_t| + 1 + |B_t^{(2)}| \leq 3k + 1 + k \cdot \sec(G) \leq k \cdot (\sec(G) + 4)$. Since the bound holds for every node t of T, we may conclude that the slim width of $(T, \mathcal{X})$ is at most $k \cdot (\sec(G) + 4)$. By Proposition 25 and Corollary 17, there exists a very nice tree-cut decomposition of G of width $k \leq \sec(G)$, therefore $\mathrm{stcw}(G) \leq \sec(G)^2 + 4 \cdot \sec(G)$. ◄

Hence, slim tree-cut with of any graph is upper-bounded by a quadratic function of its super edge-cut width. Next, we show that the converse statement holds as well:

▶ **Proposition 27.** *For every graph G, $\sec(G) \leq 3 \cdot (\mathrm{stcw}(G)+1)^2$. Moreover, given a tree-cut decomposition of G of slim width k, it is possible to compute a supergraph $Q \supseteq G$ and its spanning tree T witnessing $\sec(G) \leq 3(k+1)^2$ in cubic time.*

Proof. Let $(T_0, \mathcal{X}_0)$ be a tree-cut decomposition of G of slim width k. We start by transforming it into a nice tree-cut decomposition $(T, \mathcal{X})$ in cubic time as in [16]. The transformation procedure acts on the 2-centers of torsos only by contracting some edges. Recall that $v \in V(H_t) \setminus X_t$ is not in the 2-center of (H_t, X_t) if and only if v belongs to some induced subtree of H_t connected to the rest of H_t by at most one edge. Since contracting an edge either preserves the property or merges v with some other vertex, it doesn't increase $\mathrm{tor}_2(t)$ for any node t of T. In particular, the slim width of $(T, \mathcal{X})$ is at most k.

Let $\Omega \subseteq \mathcal{X}$ be the set of empty bags of $(T, \mathcal{X})$, we construct $Q \supseteq G$ along with its tree-cut decomposition $(T, \mathcal{X}')$ as follows. Firstly, we add to G vertices v_t for every $t \in \Omega$. We define $X_t' = \{v_t\}$ if $X_t = \emptyset$ and $X_t' = X_t$ otherwise. For every node $t \in T$, construct an arbitrary tree T_t^* over X_t' and add its edges to Q. Further, we process every edge $e = pt \in E(T)$ such that p is the parent of t in T and either $N(Y_t) \not\subseteq X_t$ or $\mathrm{adh}(t) > 1$ as follows. If G doesn't contain an edge between X_t' and X_p', we add to $E(Q)$ arbitrary edge with endpoints in X_t' and X_p'. This increases the adhesion of e by at most one.

Now we proceed to the choice of the spanning tree T^* in Q. For every $t \in T$ other then the root, let p be the parent of t in T. If $\mathrm{adh}(t) = 1$ and $N(Y_t) \subseteq X_t$, we denote by e_t the unique edge between Y_t' and X_p' in Q. Otherwise, let e_t be arbitrary edge of Q with endpoints in X_t' and X_p'. We then construct T^* by gluing together all T_t^* via edges e_t: $T^* = (\cup_{t \in V(T)} T_t^*) \bigcup (\cup_{t \in V(T) \setminus r} \{e_t\})$. Obviously the construction can be performed in cubic time; we will show that $\sec(Q, T^*) \leq 3(k+1)^2$.
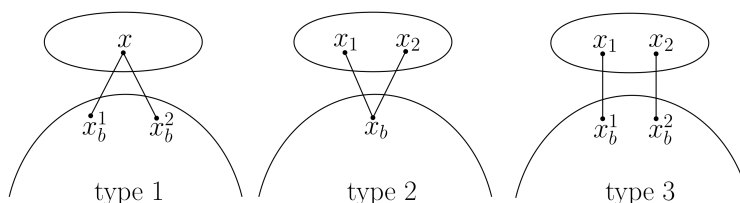
Figure 4 Possible configurations of edges between thin child $b \in B_t^{(2)}$ and its parent t.

To this end, fix any node t of T and $x \in X_t'$ and denote $E_{loc}(x) = E_{loc}^{Q,T^*}(x)$. If T^* contains more than one edge between Y_t' and rest of T^*, then all but one of them are the unique edges connecting Q_q' to the rest of Q for some descendants q of t in T. Hence, they don't belong to any path in T^* between the endpoints of some feedback edge $e \in E(Q) \setminus E(T^*)$. Therefore, every edge of $E_{loc}(x)$ has at least one endpoint in Y_t'. The number of edges in $E_{loc}(x)$ with both endpoints in X_t' is at most $|X_t'| \cdot (|X_t'| - 1) \leq k \cdot (k - 1)$. Every edge with one endpoint in X_t' and another outside of Y_t' contributes to the adhesion of t in $(T, \mathcal{X}')$, so their number is bounded by $k + 1$.

Finally, if $e = yz \in E_{loc}(x)$ contains an endpoint y in $Y_t' \setminus X_t'$, then $y \in Y_q'$ for some child q of t. Then Q contains a cycle intersecting Y_q' and $x \in X_t$. In particular, by construction of Q we may conclude $q \in A_t \cup B_t^{(2)}$ w.r.t. the decomposition $(T, \mathcal{X})$. By the same arguments as for the node t, we conclude that at most one edge between Y_q' and the rest of T^* belongs to any path in T^* between the endpoints of some feedback edge $e \in E(Q) \setminus E(T^*)$, so $z \notin Y_q'$ and e contributes to the adhesion of q in $(T, \mathcal{X}')$. In particular, $E_{loc}(x)$ contains at most $\text{adh}(q) + 1$ edges with an endpoint in Y_q'. In total, at most $\max_{q \in A_t}(\text{adh}(q) + 1) \cdot |A_t| + \max_{q \in B_t^{(2)}}(\text{adh}(q) + 1) \cdot |B_t^{(2)}| \leq (k+1)(2k+1) + 3k = 2k^2 + 6k + 1$ edges in $E_{loc}(x)$ have an endpoint in $Y_t' \setminus X_t'$, so $|E_{loc}(x)| \leq k \cdot (k-1) + (k+1) + 2k^2 + 6k + 1 = 3k^2 + 6k + 2$ and hence $\text{sec}(Q, T^*) \leq 3k^2 + 6k + 3 = 3(k+1)^2$. ◀

▶ **Corollary 28.** sec *and* stcw *are asymptotically equivalent.*

The results of this section are summarized in Figure 5. In particular, the graph family provided in [4, Lemma 2] shows that graphs of constant super edge-cut width may have arbitrarily large edge-cut width.

5   Approximating Slim Tree-Cut Width

In this section we show how to efficiently construct a tree-cut decomposition of a graph G with slim width bounded by a cubic function of its optimal value $\text{stcw}(G)$. As a starting point for our approximation, we use the following result of Kim, Oum, Paul, Sau and Thilikos:

▶ **Theorem 29** ([28])**.** *There exists an algorithm that, given a graph G and $\omega \in \mathbb{N}$, either outputs a tree-cut decomposition of G with width at most 2ω or correctly reports that no tree-cut decomposition of G with width at most ω exists in $2^{\mathcal{O}(\omega^2 \cdot \log \omega)} \cdot n^2$ steps.*

As an observant reader might have already noticed, if G has bounded slim tree-cut width, it imposes some restrictions on the structure of possible decompositions of G of small (standard) tree-cut width. This fact enables us to construct an efficient approximation for $\text{stcw}(G)$.
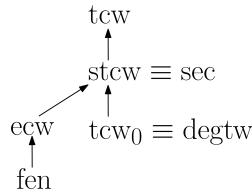
Figure 5 Position of slim and 0-tree-cut width in the hierarchy of edge-cut based parameters. An arrow from p to q represents the fact that p is more restrictive than q, while asymptotic equivalence is depicted by $\equiv$.

▶ **Theorem 30.** *There exists an algorithm that, given a graph G and $\omega \in \mathbb{N}$, either outputs a tree-cut decomposition of G with slim width at most $6(\omega + 1)^3$ or correctly reports that no tree-cut decomposition of G with slim width at most ω exists in $2^{\mathcal{O}(\omega^2 \cdot log\omega)} \cdot n^4$ steps.*

Proof. Given a graph G and $\omega \in \mathbb{N}$, let us run the algorithm from Theorem 29. If it reports that $\mathrm{tcw}(G) > \omega$, we may conclude that $\mathrm{stcw}(G) > \omega$ by Corollary 8. In case the algorithm returns a tree-cut decomposition $(T', \mathcal{X}')$ of width at most 2ω, we invoke Corollary 17 to transform this decomposition into a very nice decomposition $(T, \mathcal{X})$ of the same width in at most quartic time. By Proposition 26, we have that $|B_t^{(2)}| \leq 2\omega \cdot \mathrm{sec}(G)$ for each node t of T. If for some node t the size of $B_t^{(2)}$ exceeds $6\omega \cdot (\omega + 1)^2$, then $\mathrm{sec}(G) > 3(\omega + 1)^2$ and by Proposition 27 we may correctly report that $\mathrm{stcw}(G) > \omega$. Otherwise, $\mathrm{tor}_2(t) \leq 1 + |X_t| + |A_t| + |B_t^{(2)}| \leq 1 + 2\omega + (4\omega + 1) + 6\omega \cdot (\omega + 1)^2 \leq 6(\omega + 1)^3$ for any node t of T. Hence, the slim width of $(T, \mathcal{X})$ is at most $6(\omega + 1)^3$. ◄

6 Discussion of Algorithmic Applications

Having established its structural properties, we now turn to the algorithmic aspects of slim tree-cut width. Here, Corollary 28 shows that instead of using a tree-cut decomposition of the input graph G to design fixed-parameter algorithms – as was done in past dynamic programming algorithms that utilized tree-cut width – we can perform dynamic programming along a spanning tree T of a supergraph Q of G. Both Q and T can be computed from G in a pre-processing stage by using Proposition 27, and using a spanning tree instead of a tree-cut decomposition typically leads to significantly more concise (and conceptually cleaner) algorithms.

The cost for this simplification is the quadratic gap between the widths of these decompositions. We note that this situation is somewhat analogous to how one still typically uses clique-width [7] as a general and easy-to-use parameterization for various problems (especially when aiming for instances with higher edge-densities), even though rank-width [34] and Boolean-width [6] are asymptotically equivalent parameterizations which have been shown to yield more efficient algorithms [15] – there, the gap is even exponential.

Recall that a number of problems which remain W[1]-hard w.r.t. tree-cut width have recently been shown to be fixed-parameter tractable when parameterized by edge-cut width [4, 18], via explicit dynamic programming algorithms which proceed along the spanning tree of the input graph. While the functional gap between edge-cut width and super edge-cut width (and, analogously, slim tree-cut width) may be arbitrarily large, it is not difficult to see that each of the algorithms provided in those papers can be straightforwardly lifted to fixed-parameter algorithms w.r.t. super edge-cut width. Indeed, the only amendment one needs to make is to deal with the presence of "ghost" edges and vertices which occur in the spanning tree but not in the graph, and the computation of the records in these algorithms can easily deal with such vertices and edges.

To provide a concrete illustration of how this can be done, let us revisit the dynamic programming algorithm for the EDGE DISJOINT PATHS problem parameterized by edge-cut width [4, Theorem 2]. No change is needed to the records. When the algorithm attempts to compute the set of "valid records" for a vertex v from the sets of valid records for some of its children $v_1, \ldots, v_\psi$ in the spanning tree, the algorithm performs a branching step in which it considers all possible ways the paths can be routed between the subtrees rooted at these children (See the "If v is an internal node" paragraph in the proof). At this branching step, we simply discard all routings which use edges that are not present in G. The situation is no more complicated for the other considered problems – in essentially all cases, the change simply boils down to ignoring the vertices and edges which do not exist in G.

Hence, we obtain:

▶ **Corollary 31** (Theorems 2-6 in [4], Theorems 6 and 14 in [18]). LIST COLORING, PRECOLOR-ING EXTENSION, BOOLEAN CONSTRAINT SATISFACTION, EDGE DISJOINT PATHS, BAYESIAN NETWORK STRUCTURE LEARNING, POLYTREE LEARNING, MINIMUM CHANGEOVER COST ARBORESCENCE, and MAXIMUM STABLE ROOMMATES WITH TIES AND INCOMPLETE LISTS *are fixed-parameter tractable w.r.t. slim tree-cut width.*

Last but not least, given the ease with transferring dynamic programming algorithms from edge-cut width to slim tree-cut width, an inquisitive reader might be wondering whether it is not possible to formally prove that *every* problem which is FPT w.r.t. former is also FPT w.r.t. the latter. That is, however, not true in general: one can construct entirely artificial problems which do not behave in this way.

To illustrate this on a high level, let us consider an arbitrary graph problem P which remains NP-hard even on trees (as an example, the FIREFIGHTER problem [12]) and can be solved on general n-vertex graphs in time $\tau(n)$. Moreover, let $\iota(n)$ denote the time required to compute the slim tree-cut width of a graph G via an exhaustive brute force search, and let ψ be a function which dominates both τ and ι. We now define an artificial new problem P' as follows:

- every n-vertex graph G such that $\psi(\text{ecw}(G)) \leq n$ is a YES-instance, and otherwise
- G is a YES-instance if and only if G is a YES-instance of FIREFIGHTER.

Then P' is FPT parameterized by edge-cut width. Indeed, given an instance (G, k) of P', one can attempt to run a brute-force search to determine the edge-cut width (which is promised to be at most k) with a time-out of $\psi(\psi(k))$. If the algorithm times out, this implies that $\psi(\text{ecw}(G)) \leq n$ and we correctly output "Yes". If not, we proceed by calling a brute-force algorithm to solve FIREFIGHTER on G, and this must once again complete in time at most $\psi(\psi(k))$. On the other hand, P' remains NP-hard even on graph classes with constant $\text{stcw}(G)$ – consider, for instance, the class of all graphs with two connected components, one of which (C_1) is a tree and the other (C_2) a graph from the class with constant slim tree-cut width but unbounded edge-cut width (one such class is depicted in Figure 2 of [4]). On some inputs from this class, P' will ask for a solution to the FIREFIGHTER problem (which is NP-hard on trees) but the parameter $\text{stcw}(G)$ will remain constant.

7   Conclusion

The contribution of this work is mainly conceptual: it provides a possible resolution to the search for an alternative to treewidth for edge cuts which is both structurally sound and exhibits the expected (and desired) algorithmic properties. Slim tree-cut width can be viewed as the "missing link" which explains why the problems depicted in Table 1 admit fixed-parameter algorithms that exploit dynamic programming along small edge cuts w.r.t. both edge-cut width (as a generalization of the feedback edge number) and treewidth plus maximum degree. We firmly believe that there are many more problems of interest where edge-cut based parameters may help push the frontiers of tractability. On this front, the alternative characterization via the edge-cut width of a supergraph provides decompositions which are better suited for dynamic programming than tree-cut decompositions.

The problem of computing optimal decompositions for slim tree-cut width remains, similarly as in the case of tree-cut width [28], as a prominent open question. Moreover, we believe that the ideas used to obtain a 2-approximation algorithm for tree-cut width could also be used to obtain an improved constant-factor approximation for slim tree-cut width.

References

1   Deeksha Adil, Sushmita Gupta, Sanjukta Roy, Saket Saurabh, and Meirav Zehavi. Parameterized algorithms for stable matching with ties and incomplete lists. *Theor. Comput. Sci.*, 723:1–10, 2018. `doi:10.1016/j.tcs.2018.03.015`.

2   Benjamin Bergougnoux, Eduard Eiben, Robert Ganian, Sebastian Ordyniak, and M. S. Ramanujan. Towards a polynomial kernel for directed feedback vertex set. *Algorithmica*, 83(5):1201–1221, 2021. `doi:10.1007/s00453-020-00777-5`.

3   Hans L. Bodlaender, Bart M. P. Jansen, and Stefan Kratsch. Preprocessing for treewidth: A combinatorial analysis through kernelization. *SIAM J. Discret. Math.*, 27(4):2108–2142, 2013.

4   Cornelius Brand, Esra Ceylan, Christian Hatschka, Robert Ganian, and Viktoriia Korchemna. Edge-cut width: An algorithmically driven analogue of treewidth based on edge cuts. In *Graph-Theoretic Concepts in Computer Science – 48th International Workshop, WG 2022*, Lecture Notes in Computer Science. Springer, 2022. to appear. `arXiv:2202.13661`.

5   Robert Bredereck, Klaus Heeger, Dusan Knop, and Rolf Niedermeier. Parameterized complexity of stable roommates with ties and incomplete lists through the lens of graph parameters. In Pinyan Lu and Guochuan Zhang, editors, *30th International Symposium on Algorithms and Computation, ISAAC 2019, December 8-11, 2019, Shanghai University of Finance and Economics, Shanghai, China*, volume 149 of *LIPIcs*, pages 44:1–44:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019.

6   Binh-Minh Bui-Xuan, Jan Arne Telle, and Martin Vatshelle. Boolean-width of graphs. *Theor. Comput. Sci.*, 412(39):5187–5204, 2011. `doi:10.1016/j.tcs.2011.05.022`.

7   B. Courcelle, J. A. Makowsky, and U. Rotics. Linear time solvable optimization problems on graphs of bounded clique-width. *Theory Comput. Syst.*, 33(2):125–150, 2000.

8   Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.

9   Reinhard Diestel. *Graph Theory, 4th Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2012.

10 Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer Verlag, 2013. `doi:10.1007/978-1-4471-5559-1`.

11 Michael R. Fellows, Daniel Lokshtanov, Neeldhara Misra, Frances A. Rosamond, and Saket Saurabh. Graph layout problems parameterized by vertex cover. In *ISAAC*, Lecture Notes in Computer Science, pages 294–305. Springer, 2008.

12 Stephen Finbow, Andrew D. King, Gary MacGillivray, and Romeo Rizzi. The firefighter problem for graphs of maximum degree three. *Discret. Math.*, 307(16):2094–2105, 2007. `doi:10.1016/j.disc.2005.12.053`.

13 Krzysztof Fleszar, Matthias Mnich, and Joachim Spoerhase. New algorithms for maximum disjoint paths based on tree-likeness. *Math. Program.*, 171(1-2):433–461, 2018.

14 Robert Ganian. Improving vertex cover as a graph parameter. *Discret. Math. Theor. Comput. Sci.*, 17(2):77–100, 2015. URL: `http://dmtcs.episciences.org/2136`.

15 Robert Ganian and Petr Hliněný. On parse trees and Myhill-Nerode-type tools for handling graphs of bounded rank-width. *Discr. Appl. Math.*, 158(7):851–867, 2010.

16 Robert Ganian, Eun Jung Kim, and Stefan Szeider. Algorithmic applications of tree-cut width. In Giuseppe F. Italiano, Giovanni Pighizzini, and Donald Sannella, editors, *Mathematical Foundations of Computer Science 2015 – 40th International Symposium, MFCS 2015, Milan, Italy, August 24-28, 2015, Proceedings, Part II*, volume 9235 of *Lecture Notes in Computer Science*, pages 348–360. Springer, 2015. to appear in the Siam Journal on Discrete Mathematics. `arXiv:2206.00752`.

17 Robert Ganian, Fabian Klute, and Sebastian Ordyniak. On structural parameterizations of the bounded-degree vertex deletion problem. *Algorithmica*, 83(1):297–336, 2021.

18 Robert Ganian and Viktoriia Korchemna. The complexity of bayesian network learning: Revisiting the superstructure. In *Proceedings of NeurIPS 2021, the Thirty-fifth Conference on Neural Information Processing Systems*, 2021. to appear.

19 Robert Ganian and Sebastian Ordyniak. The complexity landscape of decompositional parameters for ILP. *Artif. Intell.*, 257:61–71, 2018.

20 Robert Ganian and Sebastian Ordyniak. The power of cut-based parameters for computing edge-disjoint paths. *Algorithmica*, 83(2):726–752, 2021.

21 Robert Ganian, Sebastian Ordyniak, and M. S. Ramanujan. On structural parameterizations of the edge disjoint paths problem. *Algorithmica*, 83(6):1605–1637, 2021.

22 Archontia C. Giannopoulou, O-joung Kwon, Jean-Florent Raymond, and Dimitrios M. Thilikos. Lean tree-cut decompositions: Obstructions and algorithms. In Rolf Niedermeier and Christophe Paul, editors, *36th International Symposium on Theoretical Aspects of Computer Science, STACS 2019, March 13-16, 2019, Berlin, Germany*, volume 126 of *LIPIcs*, pages 32:1–32:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019.

23 Archontia C. Giannopoulou, O-joung Kwon, Jean-Florent Raymond, and Dimitrios M. Thilikos. A menger-like property of tree-cut width. *J. Comb. Theory, Ser. B*, 148:1–22, 2021. `doi:10.1016/j.jctb.2020.12.005`.

24 Didem Gözüpek, Sibel Özkan, Christophe Paul, Ignasi Sau, and Mordechai Shalom. Parameterized complexity of the MINCCA problem on graphs of bounded decomposability. *Theor. Comput. Sci.*, 690:91–103, 2017.

25 Didem Gözüpek, Hadas Shachnai, Mordechai Shalom, and Shmuel Zaks. Constructing minimum changeover cost arborescenses in bounded treewidth graphs. *Theor. Comput. Sci.*, 621:22–36, 2016. `doi:10.1016/j.tcs.2016.01.022`.

26 Gregory Z. Gutin, Mark Jones, and Magnus Wahlström. The mixed chinese postman problem parameterized by pathwidth and treedepth. *SIAM J. Discret. Math.*, 30(4):2177–2205, 2016.

27 Bart M. P. Jansen and Hans L. Bodlaender. Vertex cover kernelization revisited – upper and lower bounds for a refined parameter. *Theory Comput. Syst.*, 53(2):263–299, 2013. `doi:10.1007/s00224-012-9393-4`.

28 Eun Jung Kim, Sang-il Oum, Christophe Paul, Ignasi Sau, and Dimitrios M. Thilikos. An FPT 2-approximation for tree-cut decomposition. *Algorithmica*, 80(1):116–135, 2018.

29 Loïc Magne, Christophe Paul, Abhijat Sharma, and Dimitrios M. Thilikos. Edge-treewidth: Algorithmic and combinatorial properties. *CoRR*, abs/2112.07524, 2021. `arXiv:2112.07524`.

30 Dániel Marx and Paul Wollan. Immersions in highly edge connected graphs. *SIAM J. Discrete Math.*, 28(1):503–520, 2014.

31 Jesper Nederlof, Michal Pilipczuk, Céline M. F. Swennenhuis, and Karol Wegrzycki. Hamiltonian cycle parameterized by treedepth in single exponential time and polynomial space. In Isolde Adler and Haiko Müller, editors, *Graph-Theoretic Concepts in Computer Science – 46th International Workshop, WG 2020, Leeds, UK, June 24-26, 2020, Revised Selected Papers*, volume 12301 of *Lecture Notes in Computer Science*, pages 27–39. Springer, 2020.

32 Jaroslav Nesetril and Patrice Ossona de Mendez. *Sparsity – Graphs, Structures, and Algorithms*, volume 28 of *Algorithms and Combinatorics*. Springer, 2012.

33 Sebastian Ordyniak and Stefan Szeider. Parameterized complexity results for exact bayesian network structure learning. *J. Artif. Intell. Res.*, 46:263–302, 2013. `doi:10.1613/jair.3744`.

34 Sang-il Oum. Approximating rank-width and clique-width quickly. In *Graph-Theoretic Concepts in Computer Science, 31st International Workshop, WG 2005, Metz, France, June 23-25, 2005, Revised Selected Papers*, volume 3787 of *Lecture Notes in Computer Science*, pages 49–58. Springer Verlag, 2005.

35 Neil Robertson and P. D. Seymour. Graph minors. II. Algorithmic aspects of tree-width. *J. Algorithms*, 7(3):309–322, 1986.

36 Marko Samer and Stefan Szeider. Constraint satisfaction with bounded treewidth revisited. *J. of Computer and System Sciences*, 76(2):103–114, 2010.

37 Paul Wollan. The structure of graphs not admitting a fixed immersion. *J. Comb. Theory, Ser. B*, 110:47–66, 2015.