Approximation algorithms for classes of graphs excluding single-crossing graphs as minors*

Erik D. Demaine[1], MohammadTaghi Hajiaghayi[1], Naomi Nishimura[2], Prabhakar Ragde[2], and Dimitrios M. Thilikos[3]

[1] Laboratory for Computer Science, Massachusetts Institute of Technology, 200 Technology Square, Cambridge, MA 02139, U.S.A.

[2] School of Computer Science, University of Waterloo, 200 University Ave. West, Waterloo, Ontario, N2L 3G1, Canada.

[3] Departament de Llenguatges i Sistemes Informàtics, Universitat Politècnica de Catalunya, Campus Nord – Mòdul C5, Desp. 211b, c/Jordi Girona Salgado, 1-3. E-08034, Barcelona, Spain

Abstract. Many problems that are intractable for general graphs allow polynomial-time solutions for structured classes of graphs, such as planar graphs and graphs of bounded treewidth. In this paper, we demonstrate structural properties of larger classes of graphs and show how to exploit the properties to obtain algorithms. The classes considered are those formed by excluding as a minor a graph that can be embedded in the plane with at most one crossing. We show that graphs in these classes can be decomposed into planar graphs and graphs of small treewidth; we use the decomposition to show that all such graphs have locally bounded treewidth (all subgraphs of a certain form are graphs of bounded treewidth). Finally, we make use of the structural properties to derive polynomial-time algorithms for approximating treewidth within a factor of 1.5 and branchwidth within a factor of 2.25 as well as polynomial-time approximation schemes for both minimization and maximization problems and fixed-parameter algorithms for problems such as vertex cover, edge-dominating set, feedback vertex set, and others.

1 Introduction

The development of algorithms for NP-complete problems on restricted classes of graphs has resulted in structural characterizations of algorithmic utility. For example, algorithms for graphs of bounded treewidth rely on techniques using separator properties resulting from tree decompositions. In this paper we focus on graph classes obtained by excluding a single-crossing graph as a minor. We present a polynomial-time algorithm that determines a *clique-sum decomposition* of such a graph, a representation of the graph as a collection of planar graphs and graphs of small treewidth (defined formally in Section 3). Our result generalizes previous decomposition results for graphs excluding special single-crossing graphs such as $K_{3,3}$ [Asa85] and K_5 [KM92].

* Parts of this paper appeared in the Proceedings of the Fifth International Workshop on Approximation Algorithms for Combinatorial Optimization (APPROX 2002) and in the Proceedings of the Euroconference on Combinatorics, Graph Theory, and Applications (COMB 2001). The research of the third and fourth authors was supported by the Natural Sciences and Engineering Research Council of Canada and Communication and Information Technology Ontario. The work of the fifth author was supported by the IST Programme of the EU under contract number IST-1999-14186 (ALCOM-FT), the Spanish CICYT project TIC2000-1970-CE, and the Ministry of Education and Culture of Spain (Resolución 31/7/00 – BOE 16/8/00). E-mail addresses: {hajiagha,edemaine}@theory.lcs.mit.edu, {nishi,plragde}@uwaterloo.ca, and sedthilk@lsi.upc.es .

A second structural property is that of *locally bounded treewidth* which allows a *layer decomposition*, enabling us to represent a graph as a collection of subgraphs, each of bounded treewidth (formal definitions follow in Section 4). In order to take advantage of the bound on treewidth of the subgraphs, we give a tree decomposition algorithm for this special case, adding to the toolkit of tree decomposition algorithms for small bounds on treewidth [AP86,MT92,San96] and r-outerplanar graphs [ABF$^+$02] that reduce the prohibitively high constant factors found in algorithms for more general graphs [Lag96,BK96,Bod96].

Included among the results in this paper are several applications of our structural characterizations and algorithms. Using clique-sum decompositions, we obtain the first constant-factor approximation algorithm for treewidth of nonplanar graphs. Furthermore, we use properties of layer decompositions to form polynomial-time approximation schemes for a range of maximization and minimization problems, as well as fixed-parameter algorithms for dominating set and related problems.

We present decomposition results followed by algorithms for NP-complete problems. Sections 2 through 4 present general results from which applications are derived in Sections 5 through 7. First, in Section 2, we introduce the terminology used throughout the paper. Next, in Section 3, we introduce the concept of clique sums and demonstrate how graphs excluding single-crossing graphs as minors can be characterized using a clique-sum decomposition. The fact that such graphs have bounded local treewidth is established in Section 4, which also gives an algorithm for computing the tree decomposition of a local neighborhood in a graph. The rest of the paper contains results that follow from these properties: an approximation algorithm for treewidth (Section 5), polynomial-time approximation schemes for optimization problems (Section 6), and fixed-parameter algorithms for dominating set and its variants (Section 7). Finally, in Section 8, we summarize our results and present directions for further research.

2 Background

2.1 Graph terminology

We assume the reader is familiar with general concepts of graph theory such as graphs, trees, and planar graphs. The reader is referred to standard references for appropriate background [BM76]. Here we review a few terms used in this paper; others will be defined in the context of their use.

Throughout this paper, all graphs are finite, simple, and undirected, unless indicated otherwise. A graph G is represented by $G = (V, E)$, where V (or $V(G)$) is the set of vertices and E (or $E(G)$) is the set of edges; we use n to denote $|V|$ when G is clear from context. An edge e in a graph G between u and v is denoted by $\{u, v\}$ or, equivalently, $\{v, u\}$. Here, vertices u and v are called the *endpoints* of e.

In our algorithms we will consider pieces of graphs as well as ways of joining pieces to form larger graphs. A graph $G' = (V', E')$ is a *subgraph of* G if $V' \subseteq V$ and $E' \subseteq E$, and is an *induced subgraph of* G, denoted by $G[V']$, if in addition E' contains all edges of E that have both endpoints in V'. The (disjoint) *union* of two disjoint graphs G_1 and G_2, $G_1 \cup G_2$, is a graph G formed by merging vertex and edge sets, so that $V(G) = V(G_1) \cup V(G_2)$ and $E(G) = E(G_1) \cup E(G_2)$.

We define the *r-neighborhood* of a set $S \subseteq V(G)$, denoted by $N_G^r(S)$, to be the set of vertices at distance at most r from at least one vertex of $S \subseteq V(G)$; if $S = \{v\}$ we simply use the notation $N_G^r(v)$. The *diameter* of G, denoted by $diam(G)$, is the maximum over all distances between pairs of vertices of G. An *n-clique (K_n)* is an n-vertex graph in which every pair of vertices is connected by an edge. The vertices of the graph $K_{n,m}$ can be partitioned into sets V_1 and V_2 such that $|V_1| = n$, $|V_2| = m$, and the edge set consists of all edges $\{u, v\}$ such that $u \in V_1$ and $v \in V_2$. A graph $G = (V, E)$ is *k-connected* if for any $S \subseteq V(G)$ such that $|S| \leq k$, $G[V - S]$ is connected.

A graph is *planar* if it can be drawn in the plane so that its edges intersect only at their endpoints; such a drawing is called an *embedding*. An embedding partitions the plane into connected regions called *faces*; the unbounded region is called the *outer face*. A graph with all vertices on the outer face is called *outerplanar*; a *k-outerplanar graph* has the property that k successive deletions of the vertices on the outer face results in the empty graph.

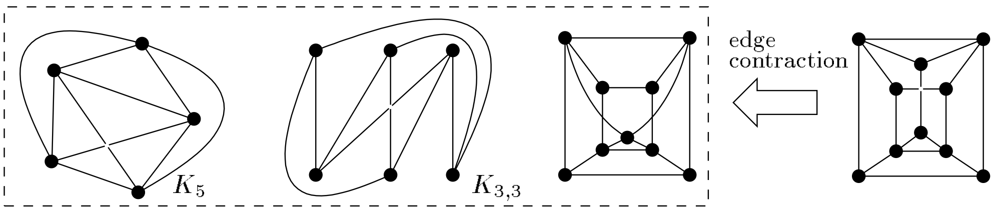

Fig. 1. Examples of single-crossing graphs

We consider classes of graphs that are associated with *single-crossing graphs*, as defined by Robertson and Seymour [RS93]. To define this notion, we first need the concept of a minor. A graph G is a *minor* of a graph H if H can be obtained from a subgraph of G by contracting zero or more edges. To *contract* an edge $e = \{u, v\}$ is to replace both u and v by a single vertex w and to form an edge between w and any neighbor of u or v other than u or v. A single-crossing graph can now be defined as a graph that is a minor of one that can be drawn in the plane with at most one pair of edges crossing. Note that a single-crossing graph may itself not be drawable in this fashion. Figure 1 shows to the left three examples of single-crossing graphs; the third one cannot be drawn in the plane with only one crossing, but is obtainable by edge contraction from the graph to the right, which can be so drawn.

We consider classes of graphs that do not contain a particular single-crossing graph as a minor. A graph class $\mathcal{C}$ is a *minor-closed* class if any minor of any graph in $\mathcal{C}$ is also a member of $\mathcal{C}$. A graph G is *H-minor-free* if H is not a minor of G; we will call such a graph *single-crossing-minor-free graph* when H is a single-crossing graph. The following lemma is a consequence of closure under minors; in contrast, the class of graphs that can be drawn in the plane with at most one pair of edges crossing is not closed under minors, as Figure 1 shows.

Lemma 1. *If single-crossing-minor-free graph G excludes a single-crossing graph H as a minor, any minor G' of G is also a single-crossing-minor-free graph which excludes H as a minor.*

The following example demonstrates that single-crossing-minor-free graphs can be considerably more complicated than single-crossing graphs. Form a graph on $n = 6k$ vertices by taking k copies of $K_{3,3}$ and adding $k-1$ edges to connect them into a path-like structure. This graph has $\Theta(n)$ crossings, but is K_5-minor-free. In fact, any graph can be shown to be a single-crossing-minor-free graph, where the excluded single-crossing graph is a sufficiently large grid. However, our algorithms are really only of interest when the excluded graph is small.

2.2 Treewidth and locally bounded treewidth

Treewidth plays an important role in this paper, as a property of inputs as well as the objective of an algorithm. The notion was first defined by Robertson and Seymour [RS84] and served as one of the cornerstones of their lengthy proof of Wagner's conjecture, now known as the Graph Minors Theorem [RS85]. Treewidth has several applications in algorithmic graph theory: a wide range of otherwise-intractable combinatorial problems are polynomially solvable, often linearly solvable, when restricted to graphs of bounded treewidth [ACP87,Bod93,Bod97,Bod98].

The *treewidth* of a graph is the minimum k such that the graph can be "decomposed" into a tree structure of *bags* of at most $k+1$ vertices such that each vertex of the graph appears in a connected subtree of bags, and endpoints of any edge appear together in at least one bag.

Definition 1. [RS86] *A* tree decomposition *of a graph $G = (V, E)$, denoted by $TD(G)$, is a pair (χ, T) in which $T = (I, F)$ is a tree and $\chi = \{\chi_i | i \in I\}$ is a family of subsets of $V(G)$ such that:*

1. $\bigcup_{i\in I} \chi_i = V$;
2. *for each edge $e = \{u, v\} \in E$ there exists an $i \in I$ such that both u and v belong to χ_i; and*
3. *for all $v \in V$, the set of nodes $\{i \in I | v \in \chi_i\}$ forms a connected subtree of T.*

We distinguish between *vertices* of the original graph G and *nodes* of T in $TD(G)$, and use the term *bag* to refer to a set χ_i. The maximum size of a bag in $TD(G)$ minus one is called the *width* of the tree decomposition. The *treewidth* of a graph G ($\mathrm{tw}(G)$) is the minimum width over all possible tree decompositions of G.

A *graph of bounded treewidth* is a graph of treewidth at most k, for k a constant independent of the size of the graph. A related notion is that of a k*-tree* [Ros74], a graph G such that either G is a k-clique or G has a vertex u of degree k such that u is adjacent to a k-clique, and the graph obtained by deleting u and all its incident edges is a k-tree. It has been shown that for any k, the class of graphs of treewidth at most k is equivalent to the class of *partial k-trees*, that is, subgraphs of k-trees [vL90]. The idea of treewidth can be generalized to that of *locally bounded treewidth* [Epp00], in which each local subgraph has treewidth bounded by a function of r.

Definition 2. *The* local treewidth *of a graph G is the function* $\mathrm{ltw}^G : \mathbb{N} \to \mathbb{N}$ *that associates with every $r \in \mathbb{N}$ the maximum treewidth of an r-neighborhood in G. We set* $\mathrm{ltw}^G(r) = \max_{v \in V(G)}\{\mathrm{tw}(G[N_G^r(v)])\}$*, and we say that a graph class $\mathcal{C}$ has* bounded local treewidth (or locally bounded treewidth) *when there is a function $f : \mathbb{N} \to \mathbb{N}$ such that for all $G \in \mathcal{C}$ and $r \in \mathbb{N}$, $\mathrm{ltw}^G(r) \leq f(r)$.*

Furthermore, Eppstein [Epp00] showed that a minor-closed graph class $\mathcal{C}$ has bounded local treewidth if and only if every graph in $\mathcal{C}$ is H-minor-free for some *apex graph* H, where in any apex graph there exists a vertex whose deletion produces a planar graph. The following lemma proves useful in our results:

Lemma 2. *For any graph G and subgraph G' of G, $\mathrm{ltw}^{G'}(k) \leq \mathrm{ltw}^G(k)$, for any $k \geq 0$.*

Proof. It is enough to observe that for any $v \in G'$ and $k \geq 0$, $N_{G'}^k(v) \subseteq N_G^k(v)$. Thus the removal of vertices of $N_G^k(v) \setminus N_{G'}^k(v)$ from bags of a tree decomposition of $N_G^k(v)$ results in a tree decomposition of $N_{G'}^k(v)$ with width at most that of G. □

3 Clique-sum decompositions

3.1 Clique sums

The structural properties proved in this paper rely on the operation of graph summation, the formation of a graph by identifying cliques in two smaller graphs, removing zero or more edges from the cliques, and then "gluing" the vertex sets of the cliques. More formally, we consider graphs G_1 and G_2 with disjoint vertex-sets. For each $i \in 1, 2$ and for $G_i[W_i]$ a clique of size $k \geq 0$, we define G_i' to be a graph formed from G_i by removing zero or more edges from $G_i[W_i]$. For any bijection $h : W_1 \to W_2$ we define a *k-sum* G of G_1 and G_2, denoted by $G = G_1 \oplus_k G_2$ (or, more simply, by $G = G_1 \oplus G_2$), to be the graph obtained by identifying w with $h(w)$ for all $w \in W_1$ to form the union of G_1' and G_2' and replacing the resultant double edges with single edges. The images of the vertices of W_1 and W_2 in $G_1 \oplus_k G_2$ form the *join set*. In the rest of this section, when we refer to a vertex v of G in G_1 or G_2, we mean the corresponding vertex of v in G_1 or G_2 (or both).

The result of the operation will depend on which (if any) edges are removed from the cliques as well as which bijection is selected, so the operation $\oplus$ can have a set of possible results, and hence is not well-defined. A series of k-sums (not necessarily unique) that generate a graph G is called *a decomposition of G into clique-sum operations*.

Figure 2 demonstrates an example of a 5-sum operation.

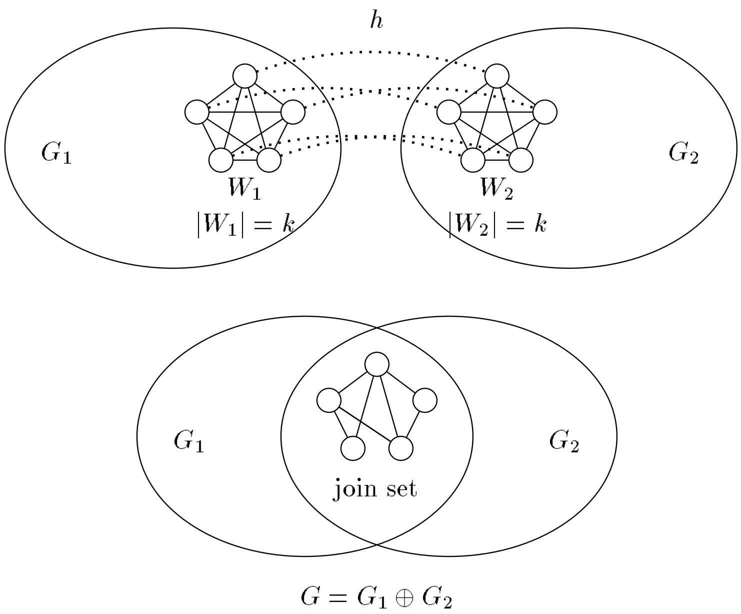

Fig. 2. Example of a 5-sum of two graphs.

3.2 Relating clique sums to treewidth and local treewidth

The following lemma shows how the treewidth changes when we apply a clique-sum operation, which will play an important role in our approximation algorithms in Section 5.

Lemma 3. [BvLTT97] *For any two graphs G and H,* $\mathrm{tw}(G \oplus H) \leq \max\{\mathrm{tw}(G), \mathrm{tw}(H)\}$.

Proof. We begin by observing that we can form a tree decomposition $TD(G)$ of G of width $\mathrm{tw}(G)$ and a tree decomposition $TD(H)$ of H of width $\mathrm{tw}(H)$. For W the set of vertices of G and H identified during the $\oplus$ operation, W is a clique in G and in H. As vertices in a clique must appear together in a bag in any decomposition of the graph [BM93], there exist a node α in $TD(G)$ such that $W \subseteq \chi_\alpha$ and a node β in $TD(H)$ such that $W \subseteq \chi_\beta$. Hence, we can form a tree decomposition of width $\max\{\mathrm{tw}(G), \mathrm{tw}(H)\}$ of $G \oplus H$ by adding an edge between α in $TD(G)$ and β in $TD(H)$. $\square$

To extend the result to graphs of bounded local treewidth in Lemma 6 (Section 4), in Lemma 4 we establish treewidth properties for neighborhoods of graphs.

Lemma 4. *For any graph G, any clique R of G, any $v \in R$, and any $k \geq 0$,* $\mathrm{tw}(G[N_G^k(R)]) \leq \mathrm{tw}(G[N_G^{k+1}(v)])$.

Proof. We note that all vertices in $R - v$ are at distance 1 from v. Therefore $N_{G_1}^k(R) \subseteq N_{G_1}^{k+1}(v)$, and the result follows from Lemma 2. $\square$

3.3 Decomposition algorithm

The main theorem of this section is a constructive version of the following non-algorithmic result of Robertson and Seymour, itself used in our algorithm.

Theorem 1. [RS93] *For any single-crossing graph H, there is an integer $c_H \geq 4$ (depending only on H) such that every H-minor-free graph can be obtained by* 0-, 1-, 2- *or* 3-*sums of planar graphs and graphs of treewidth at most c_H.*

In the remainder of the paper we will assume that c_H is the smallest integer for which Theorem 1 holds. Although previous algorithms have been developed for decomposing specific single-crossing-minor-free graphs into series of clique sums (Asano [Asa85] gave an $O(n)$-time construction for $K_{3,3}$-minor-free graphs and Kézdy and McGuinness [KM92] gave an $O(n^2)$-time construction for K_5-minor-free graphs), ours is the first general algorithm. For more examples of graph classes that can be characterized by clique-sum decompositions, see the work of Diestel [Die89,Die91]. Theorem 2 below describes a constructive algorithm to obtain a clique-sum decomposition of a single-crossing-minor-free graph G that satisfies the additional property that the smaller graphs are minors of G. The additional property is crucial for designing approximation algorithms in Section 5.

To form a clique-sum decomposition of graphs that are minors of the original graph, we consider graphs formed by first removing a set of vertices from the graph and then reinserting the removed vertices in each resulting connected component. More formally, we define a subset $S \subseteq V$ to be a *k-cut* if the induced subgraph $G[V-S]$ is disconnected and $|S| = k$, and to be a *strong k-cut* if in addition $G[V-S]$ either has more than two connected components or each component has more than one vertex. For S a strong cut that separates G into components $G_1, \ldots, G_h$, we form the *augmented components induced by S*, denoted $G_i \cup K(S)$ for $1 \leq i \leq h$, as the graphs obtained from graphs $G[V(G_i) \cup S]$ by adding an edge between each pair of nonadjacent vertices in S. Each augmented component will contain as a subgraph a clique on the vertices in S. The influence of strong cuts on augmented components is fundamental in the proof of our theorem. By introducing a less strict definition of strong cuts, we obtain a stronger version of an earlier lemma [KM92].

Lemma 5. *Let S be a strong 3-cut of a 3-connected graph $G = (V, E)$, and let $G_1, G_2, \ldots, G_h$ denote the h components of $G[V-S]$. Then each augmented component of G induced by S, $G_i \cup K(S)$, is a minor of G.*

Proof. We first consider the case in which $h \geq 3$. By symmetry, it will suffice to show that $G_1 \cup K(S)$ is a minor of G by forming the graph by a series of contractions in G. Starting with G, we first contract all edges of G_2 through G_h to obtain super-vertices y_2 through y_h (we use the term *super-vertices* to denote vertices that are the result of the contraction of all the edges in specific connected subgraphs of G). Because G is 3-connected, each super-vertex is adjacent to all vertices x_1, x_2, and x_3 in S. We now contract the edge $\{x_2, y_2\}$ to form a super-vertex x'_2 and contract edges between y_3 through y_h and x_3 to form a super-vertex x'_3. Again since each y_i was adjacent to each x_j, we can conclude that x_1, x_2', and x_3' form a clique and hence the resulting graph is the augmented component $G_1 \cup K(S)$, as needed.

If instead $h = 2$, then by the definition of a strong cut G_1 and G_2 each contain at least two vertices. As in the previous case, it will suffice to show that $G_1 \cup K(S)$ is a minor of G, as the argument for $G_2 \cup K(S)$ is symmetric. Again, we demonstrate a series of contractions that results in G_1 as well as a clique on the vertices x_1, x_2, and x_3 of S. We consider separately the cases in which G_2 is a tree and G_2 contains a cycle.

If G_2 is a tree, then because G is 3-connected, there is a vertex y_1 in G_2 that neighbors x_1, and similarly a vertex $y_3 \neq y_1$ in G_2 that neighbors x_3. We contract edges in G_2 until there remain two super-vertices y_1' and y_3', connected to x_1 and x_3 respectively. Since G_2 is connected but acyclic, there will be a single edge between y_1' and y_3'. Moreover, since G is 3-connected, there is an edge between x_2 and either y_1' or y_3', say y_1'. Finally, again because G is 3-connected, y_3' must be adjacent to a vertex of S other than x_3, that is, either x_2 or x_1. In either case we can form a clique on the vertices in S, by contracting edges $\{x_1, y_1'\}$ and $\{x_3, y_3'\}$ if y_3' is adjacent to x_2 or by contracting the edges $\{x_2, y_1'\}$ and $\{x_3, y_3'\}$ if y_3' is adjacent to x_1.

Finally suppose that G_2 has a cycle C. We claim that there are three vertex-disjoint paths connecting three vertices of C to three vertices of S in G_2. By contracting these paths and then contracting edges of C to form a triangle, we have a clique on the vertices of S as desired. To prove the claim, we augment the graph G by adding a vertex v_1 connected to every vertex in S, and by adding a vertex v_2 connected to every vertex in C. Because $|S| = 3$ and $|C| \geq 3$, the augmented graph is still 3-connected. Therefore there exist at least three vertex-disjoint paths from v_1 to v_2. Each internal vertex on each of these paths must be in G_2, and each path must contain an edge from v_1 to a vertex of S and an edge from a vertex of C to v_2. We obtain the desired paths by removing v_1 and v_2 from each path. □

Theorem 2. *For any graph G excluding a single-crossing graph H as a minor, we can construct in $O(n^4)$ time a series of clique-sum operations $G = G_1 \oplus G_2 \oplus \cdots \oplus G_m$ where each G_i, $1 \leq i \leq m$, is a minor of G and is either a planar graph or a graph of treewidth at most c_H. Here each $\oplus$ is a* 0-, 1-, 2- *or* 3-*sum.*

Proof. The algorithm proceeds by recursively determining connectivity of subgraphs of the original graph, where different decompositions are used depending on the type of cut. When considering graph G, we first determine whether it is 1-, 2-, or 3-connected. If G is disconnected, each of its connected components is considered separately, and all are joined by 0-sums to form G. If G has a 1-cut, 2-cut, or strong 3-cut S, we recursively apply the algorithm on the augmented components induced by S, applying, respectively, 1-, 2-, or 3-sums. Finally, if G is 3-connected but has no strong 3-cut, then we claim that it is either planar or has treewidth at most c_H.

We first prove the correctness of the outline above, and later fill in the algorithmic details and analyze the running time. To show that the recursive application of the algorithm will yield a correct solution, we need to show that each subgraph created is a minor of G (and hence is H-minor-free by Lemma 1). A connected component of a disconnected graph or an augmented component resulting from a 1-cut is a subgraph of G, and hence a minor. For an augmented component formed by a 2-cut, the 2-connectivity of G guarantees that any component must connect to both vertices in the cut, and hence edges can be contracted to yield the edge added in the augmentation. Lemma 5 handles the case in which there is a strong 3-cut.

Finally, we need to prove that if the graph G is 3-connected but has no strong 3-cut, then either the treewidth of G is at most c_H or G is planar. Suppose instead that neither of these properties hold. Since G is H-minor-free and 3-connected, by Theorem 1, G can be obtained by 3-sums of a sequence of graphs $\mathcal{C}$, where each graph in $\mathcal{C}$ is either planar or of treewidth at most $c_H \geq 4$. As G has no strong 3-cut, for any join set S in the clique-sum decomposition $G-S$ may contain at most one component with more than one vertex, and hence at most one graph in $\mathcal{C}$ can have more than four vertices. If every graph in $\mathcal{C}$ has at most four vertices and hence treewidth at most 3 (and hence less than c_H), then by Lemma 3, G would have treewidth less than c_H, contradicting our assumption. We can thus conclude that $\mathcal{C}$ contains subgraphs of K_4 and one planar graph of at least five vertices and of treewidth greater than c_H.

To complete the proof of correctness, we will show that our assumption about $\mathcal{C}$ yields a contradiction, that is, that G has a strong 3-cut. Since G is not planar but every graph in $\mathcal{C}$ is planar, during the clique-sum operations forming G, there exists a graph $J \in \mathcal{C}$ and a 3-sum $G'' = G' \oplus J$ with join set S such that G' is planar but G'' is not planar. It is not possible to form planar embeddings of both J and G' such that S forms the outer face, since if this were possible, the two embeddings could be joined to form a planar embedding of G'' (e.g. J would be embedded inside the triangle and G' outside). We can thus conclude that there are at least three components in $G''-S$, which implies that S is a strong 3-cut in G'' and hence in G, a contradiction.

To analyze the running time of the algorithm, we first recall that any graph G excluding an r-clique as a minor cannot have more than $(0.319+o(1))(r\sqrt{\log r})|V(G)|$ edges [Tho01]. This implies that for any single-crossing-minor-free graph G, $|E(G)| = O(|V(G)|)$.

To run the algorithm, we apply algorithms to obtain all connected components and 1-cuts in linear time [Tar72], all 2-cuts [HT73,MR92], and all $O(n^2)$ 3-cuts in $O(n^2)$ time [KR91]. Checking whether a particular 3-cut is strong can be accomplished in $O(n)$ time using depth-first search. All other operations, including checking if a graph is planar or has treewidth at most c_H, can be performed in linear time [Wil84,Bod96].

To set up recurrence relations, we make the assumption that for each cut we split a graph into two 0-, 1-, or 2-connected components or into at most three 3-connected components at a particular iteration. The running time of one iteration, excluding recursive calls, is $O(n)$. For $T(n)$ the running time on an input of size n, for a 0-sum involving a component of size n_1 we obtain the equation:

$$T(n) = T(n_1) + T(n-n_1) + O(n), \quad n_1 \geq 2.$$

The recursive calls for a 1-cut yield the following equation

$$T(n) = T(n_1) + T(n-n_1+1) + O(n), \quad n_1 \geq 2$$

where n_1 and $n - n_1 + 1$ are the sizes of the two augmented components. Similarly, for recursive calls for a 2-cut, we have

$$T(n) = T(n_1) + T(n - n_1 + 2) + O(n), \quad n_1 \geq 3.$$

For recursive calls for a strong 3-cut with exactly two components, we have

$$T(n) = T(n_1) + T(n - n_1 + 3) + O(n^3), \quad n_1 \geq 4.$$

Finally, if we have recursive calls for a strong 3-cut with at least three components, we have

$$T(n) = T(n_1) + T(n_2) + T(n - n_1 - n_2 + 6) + O(n^3), \quad 4 \leq n_1, n_2, n - n_1 - n_2 + 6 \leq n - 2$$

where n_1, n_2, and $n - n_1 - n_2 + 6$ are the sizes of the augmented components (the last being the third and all subsequent components taken together). The additive terms (+1, +2, +3, +6) are due to the duplication of the vertices of the cut in the augmented components. Solving this recurrence gives a worst-case running time of $O(n^4)$. □

Even for excluded graphs H where c_H is huge, the value of c_H does not contribute to the asymptotic complexity of the algorithm presented above. However, it does contribute heavily to the constant hidden in the order notation. In some contexts, it may make sense to replace Bodlaender's linear-time algorithm for determining treewidth exactly with an approximation. Amir [Ami01] gives an algorithm running in time $O(2^{4.38c_H} n^2 c_H)$ which either returns a tree-decomposition of width at most $4c_H$ or answers that the treewidth is more than c_H. This can be used to prove a version of Theorem 2 with c_H replaced by $4c_H$ but whose dependence on c_H is more reasonable. Similar substitutions will be possible in our approximation algorithms, discussed in Section 6.

4 Locally bounded treewidth of single-crossing-minor-free graphs

In this section we establish the locally bounded treewidth of single-crossing-minor-free graphs, which provides the structure on which the approximation schemes of Section 6 are built. First we demonstrate how clique sums and local treewidth are correlated. Next, we discuss layer decompositions and present a tree decomposition algorithm for a subgraph induced by a sequence of consecutive layers.

4.1 Bounded local treewidth

Lemma 6. *If G_1 and G_2 are graphs such that $\mathrm{ltw}^{G_1}(r) \leq f(r)$ and $\mathrm{ltw}^{G_2}(r) \leq f(r)$ for a function $f(r) \geq 0$ for all $r \in \mathbb{N}$, and $G = G_1 \oplus_k G_2$, then $\mathrm{ltw}^G(r) \leq f(r)$.*

Proof. To show $\mathrm{ltw}^G(r) \leq f(r)$, we prove that for any $v \in V(G)$ and for all $r \geq 0$, $\mathrm{tw}(G[N^r_G(v)]) \leq f(r)$. We use W to denote the join set of $G_1 \oplus_k G_2$ and without loss of generality, we assume v is from G_1. As the claim holds trivially for $r = 0$, in the remainder of the proof we assume $r > 0$. Moreover, since if $N^r_G(v)$ contains only vertices originally from G_1, the result follows from the fact that $\mathrm{ltw}^{G_1}(r) \leq f(r)$, we assume that $N^r_G(v)$ contains vertices from G_2 that are not in W. We consider two cases, depending on whether $v \in W$.

If $v \in W$, then $N^r_G(v) \subseteq N^r_{G_1}(v) \cup N^r_{G_2}(v)$. In addition, since $r \geq 1$ and vertices of W form a clique in G_i for $i = 1, 2$, $W \subseteq N^r_{G_i}(v)$. Using these two facts, we conclude that $G[N^r_G(v)]$ is a subgraph of $G_1[N^r_{G_1}(v)] \oplus G_2[N^r_{G_2}(v)]$ over the join set W. Thus, by Lemmas 2 and 3, we know

$$\mathrm{tw}(G[N^r_G(v)]) \leq \max\{\mathrm{tw}(G_1[N^r_{G_1}(v)]), \mathrm{tw}(G_2[N^r_{G_2}(v)])\} \leq f(r).$$

We now consider the case in which $v \notin W$. Each vertex in $N^r_G(v)$ is either in G_1, and hence is in $N^r_{G_1}(v)$, or it is in G_2, and hence is in $N^r_G(v) - W$. To further describe the latter set, we consider the distance between v and any vertex u in the set. Since W is the set of vertices shared by G_1 and G_2 in G, at least one vertex of W is on the shortest path from v to u in G and hence is at distance of at most $r - 1$ from v, hence $W \cap N^{r-1}_{G_1}(v) \neq \emptyset$. We let p be the minimum distance between v and any vertex in the set $W \cap N^{r-1}_{G_1}(v)$ and observe that since v in not in W, we can conclude that $1 \leq p \leq r - 1$. We further observe that since each vertex u in $N^r_G(v) - W$ is at distance at most r from v, it must be within distance $r - p$ of some vertex of W, or $u \in N^{r-p}_{G_2}(W)$. Thus $N^r_G(v) \subseteq N^r_{G_1}(v) \cup N^{r-p}_{G_2}(W)$.

To complete the proof, we use the characterization of $N^r_G(v)$ as a subset of $N^r_{G_1}(v) \cup N^{r-p}_{G_2}(W)$ to obtain an upper bound on its treewidth. First we show that $G_1[N^r_{G_1}(v)] \oplus G_2[N^{r-p}_{G_2}(W)]$ can be formed using the join set W, as W is a subset of each of the constituent graphs (each vertex of W is at distance at most r from v in G_1 since at least one vertex of W is at distance $p \leq r-1$ from v and vertices of W form a clique in G_1). Since $N^r_G(v) \subseteq N^r_{G_1}(v) \cup N^{r-p}_{G_2}(W)$, as shown in the previous paragraph, $G[N^r_G(v)]$ is a subgraph of $G_1[N^r_{G_1}(v)] \oplus G_2[N^{r-p}_{G_2}(W)]$, and hence we can apply Lemma 3 to obtain the following result:

$$\mathrm{tw}(G[N^r_G(v)]) \leq \max\{\mathrm{tw}(G_1[N^r_{G_1}(v)]), \mathrm{tw}(G_2[N^{r-p}_{G_2}(W)])\}. \tag{1}$$

By Lemma 2, since $p \geq 1$ clearly $G_2[N^{r-p}_{G_2}(W)]$ is a subgraph of $G_2[N^{r-1}_{G_2}(W)]$, and hence

$$\mathrm{tw}(G_2[N^{r-p}_{G_2}(W)]) \leq \mathrm{tw}(G_2[N^{r-1}_{G_2}(W)]). \tag{2}$$

Combining (1), (2), and the fact that $\mathrm{tw}(G_1[N^r_{G_1}(v)]) \leq f(r)$ (our assumption about G_1), we obtain

$$\mathrm{tw}(G[N^r_G(v)]) \leq \max\{f(r), \mathrm{tw}(G_2[N^{r-1}_{G_2}(W)])\}. \tag{3}$$

Since W is a clique in G_2, by Lemma 4,

$$\mathrm{tw}(G_2[N^{r-1}_{G_2}(W)]) \leq \mathrm{tw}(G_2[N^r_{G_2}(v)]) \leq f(r). \tag{4}$$

Finally, as a consequence of (3) and (4), we conclude that $\mathrm{tw}(G[N_G^r(v)]) \le f(r)$, as needed to complete the proof of the lemma. $\square$

Theorem 3 demonstrates our main result on the local treewidth of single-crossing-minor-free graphs.

Theorem 3. *For any single-crossing-minor-free graph G excluding a single-crossing graph H as a minor and for all $r \ge 0$, $\mathrm{ltw}^G(r) \le 3r + c_H$.*

Proof. By Theorem 1, we can assume $G = G_1 \oplus G_2 \oplus \cdots \oplus G_m$ where each G_i, $1 \le i \le m$, is either a planar graph or a graph of treewidth at most c_H; we prove the theorem by induction on m. In the base case, if G_1 is a planar graph then $\mathrm{ltw}^G(r) = \mathrm{ltw}^{G_1}(r) = 3r - 1 \le 3r + c_H$, $c_H \ge 0$, as the treewidth of a r-outerplanar graph is at most $3r - 1$ [ABF+02]. If instead G_1 has treewidth at most c_H, then $\mathrm{ltw}^G(r) = \mathrm{ltw}^{G_1}(r) = c_H \le 3r + c_H$, $r \ge 0$. To prove the general case, we assume the induction hypothesis is true for $m = h$, and we prove the hypothesis for $m = h + 1$ by setting $G' = G_1 \oplus G_2 \oplus \cdots \oplus G_h$ and $G'' = G_{h+1}$. By the induction hypothesis, $\mathrm{ltw}^{G'}(r) \le 3r + c_H$ and $\mathrm{ltw}^{G''}(r) \le 3r + c_H$; by applying Lemma 6, we conclude that $\mathrm{ltw}^G = \mathrm{ltw}^{G' \oplus G''}(r) \le 3r + c_H$, as needed. $\square$

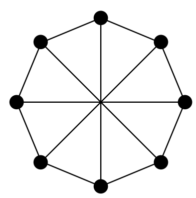

Fig. 3. The graph V_8.

Using the fact that K_5 and $K_{3,3}$ are single-crossing graphs (Figure 1), we observe that K_5-minor-free graphs and $K_{3,3}$-minor-free graphs are single-crossing-minor-free graphs. Although generalized by Theorem 2 for single-crossing-minor-free graphs, for more precise results we rely on Wagner's characterizations [Wag37]. He proved that a graph is $K_{3,3}$-minor-free if and only if it can be obtained from planar graphs and K_5 by 0-, 1-, and 2-sums and that a graph is K_5-minor-free if and only if it can be obtained from planar graphs and V_8 (the graph obtained from a cycle of length 8 by joining each pair of diagonally opposite vertices by an edge, shown in Figure 3) by 0-, 1-, 2-, and 3-sums. Since both K_5 and V_8 have treewidth four, the value of constant c_H in the proof of Theorem 3 is four, and we have:

Corollary 1. *If G is a K_5-minor-free or $K_{3,3}$-minor-free graph then $\mathrm{ltw}^G(r) \le 3r+4$.* $\square$

4.2 Local treewidth and layer decompositions

To take advantage of the bound on local treewidth, we define a layer decomposition, prove a bound on the treewidth of a subgraph induced on consecutive layers, and then provide an algorithm that forms a tree decomposition of such a subgraph. The concept of the kth outer face in planar graphs can be replaced by the concept of the kth layer (or level) in graphs of locally bounded treewidth. The kth layer (L_k) of a graph G consists of all vertices at distance k from an arbitrary fixed vertex $\hat{v}$ of $V(G)$. We denote *consecutive layers from i to j* by $L[i,j] = \bigcup_{i \le k \le j} L_k$, and call such a representation a *layer decomposition.*

Theorem 4. *For any graph G excluding a single-crossing graph H as a minor, the treewidth of $G[L[i,j]]$ is bounded above by $3(j-i+1)+c_H$.*

Proof. By contracting the connected subgraph $G[L[0,i-1]]$ to a vertex v' and applying Lemma 1, we obtain another H-minor-free graph G'. As all vertices at distance d, $i \le d \le j$, from $\hat{v}$ in G are at distance d', $1 \le d' \le j-i+1$, from v' in G' and all vertices at distance more than j from $\hat{v}$ in G are at distance more than $j-i+1$ from v' in G', we have $G[L[i,j]] = G'[L[1,j-i+1]]$. Thus $\mathrm{tw}(G[L[i,j]]) = \mathrm{tw}(G'[L[1,j-i+1]])$. Since all vertices of $L[1,j-i+1]$ in G' are in the $j-i+1$-neighborhood of v', $\mathrm{tw}(G'[L[1,j-i+1]]) \le \mathrm{tw}(G'[N_{G'}^{j-i+1}(v')])$. By the definition of local treewidth, $\mathrm{tw}(G'[N_{G'}^{j-i+1}(v')]) \le \mathrm{ltw}^{G'}(j-i+1)$. Finally by Theorem 3, we have $\mathrm{ltw}^{G'}(j-i+1) \le 3(j-i+1)+c_H$. Using these facts, $\mathrm{tw}(G[L[i,j]]) \le 3(j-i+1)+c_H$, as desired. □

Theorem 4 gives an upper bound on the treewidth of consecutive layers from i to j, but does not provide a constructive algorithm to obtain a tree decomposition of this width.

Although we can construct a tree decomposition of width $3(j-i)+c_H$ in linear time using Bodlaender's algorithm [Bod96], again the hidden constant factor will depend on this entire width. Below we show how to reduce the constant to depend only on c_H, permitting substitution of approximation algorithms as was done at the end of Section 3.

Before stating the main theorem on construction of a tree decomposition of consecutive layers, we present a simple lemma.

Lemma 7. *For $G = G_1 \oplus G_2 \oplus \cdots \oplus G_m$, if there exists a vertex $v \in V(G)$ such that each vertex of G is at distance at most r from v, then in each G_i, $1 \le i \le m$, there exists a vertex v_i such that each vertex of G_i is at distance at most r from v_i.*

Proof. We use induction on m, the number of G_i's. If $m=1$, the basis of induction is clearly true. We assume the induction hypothesis is true for $m \le h$, and we prove the hypothesis for $m = h+1$. We suppose $G = G' \oplus G''$, with join set W, where $G' = G_1 \oplus G_2 \oplus \cdots \oplus G_h$ and $G'' = G_{h+1}$. In order to apply the induction hypothesis to G' and G'', we will need to find vertices v' and v'' in G' and G'' such that each

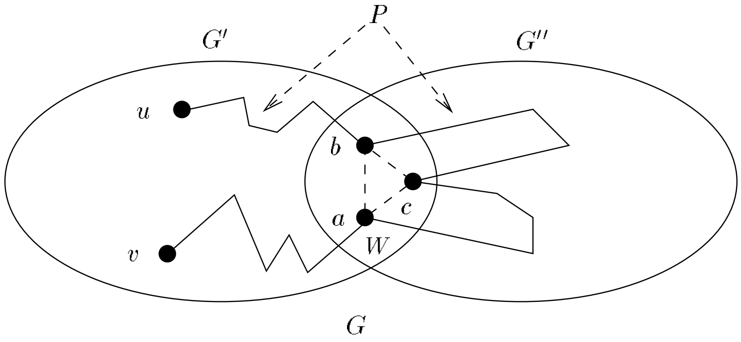

Fig. 4. The replacement of the part of path P between a and b by edge $\{a, b\}$.

vertex of G' is at distance at most r from v' and each vertex of G'' is at distance at most r from v''.

In order to apply the induction hypothesis to G' and G'', it will suffice to show that there is a path of length at most r from v to each vertex u in G' such that each vertex of the path is in G', that is $v = v'$ as defined in the previous paragraph (an analogous argument can be used to show the existence of a path in G'' to any vertex in G''). Suppose instead that every path of length at most r from v to u passed through at least one vertex w in $V(G'') - V(G')$, and consider one such path P. Clearly v and u are not both in W, since otherwise there would exist an edge $\{v, u\}$ in the clique with vertex set W. Without loss of generality we assume $u \in V(G') - V(G'')$, and observe that since the path from v to w and the path from w to u must pass through W, we can define a and b to be the first and last vertices in W on the path in order from v to u (see Figure 4), where possibly $v = a$ or $a = b$ (or both). We can then form a new path P' consisting of the subpath from v to a, the edge $\{a, b\}$ if $a \neq b$ (which must exist since the vertices of W form a clique in G'), and the subpath from b to u. The path P' has length at most the length of P and is entirely in G', contradicting our assumption and completing the proof. □

We are ready to present our algorithm for construction of a tree decomposition for a constant number of consecutive layers.

Theorem 5. *For any single-crossing-minor-free graph G, we construct a tree decomposition for $G[L[i, j]]$ of treewidth $3(j - i + 1) + c_H$ in $O((j - i + 1)^3 \cdot n + n^4)$ time; for a $K_{3,3}$-minor-free or K_5-minor-free graph G, the running time can be reduced to $O((j - i + 1)^3 \cdot n)$ or $O((j - i + 1)^3 \cdot n + n^2)$, respectively.*

Proof. As in the proof of Theorem 4, we contract the connected subgraph $G[L[0, i - 1]]$ to a vertex v' and obtain another single-crossing-minor-free graph G' such that $G[L[i, j]] = G'[L[1, j - i + 1]]$. By Lemma 1, the graph $G'' = G'[L[0, j - i + 1]]$ is a single-crossing-minor-free graph excluding the same H and by the definition of layers each vertex in G'' is at distance at most $j - i + 1$ from v'. By Theorem 2, we can determine a set of clique-sum operations of graph G'' in $O(n^4)$ time (improved to $O(n)$ for G $K_{3,3}$-minor-free using the result of Asano [Asa85] and $O(n^2)$ for G K_5-minor-free using the result of Kézdy and McGuinness [KM92]).

After determining a set of clique-sum operations of $G'' = G_1 \oplus G_2 \oplus \cdots \oplus G_m$, we construct a tree decomposition for each G_i, $1 \le i \le m$. If G_i is a graph of treewidth at most c_H, we can easily construct a tree decomposition of constant width in linear time [Bod96] (for the special cases, K_5 or V_8, a constant time construction is possible). We now consider the case in which G_i is a planar graph. By Lemma 7, in each G_i, there exists a vertex v_i such that each vertex in G_i is at distance at most $j - i + 1$ from v_i. It is known that if a planar graph has a rooted spanning tree T in which the longest path has length d, then a tree decomposition of the graph with width at most $3d$ can be found in time $O(dn)$ [Bak94,Epp99]. Since each vertex in G_i is at distance at most $j - i + 1$ from v_i, by breadth-first search we can construct a spanning tree rooted at v_i with the longest path of length at most $j - i + 1$. Hence we can construct a tree decomposition for G_i of treewidth $3(j - i + 1)$ in time $O((j - i + 1) \cdot |V(G_i)|)$.

Having tree decompositions of G_i's, $1 \le i \le m$, in the rest of the algorithm, we glue together the tree decompositions of G_i's using the construction given in the proof of Lemma 3. To this end, we introduce an array $Nodes$ indexed by all subsets of $V(G'')$ of size at most three. In this array, for each subset whose elements form a clique, we specify a node of the tree decomposition which contains this subset. We note that for each clique C in G_i, there exists a node z of $TD(G'')$ such that all vertices of C appear in the bag of z [BM93]. This array is initialized as part of a preprocessing stage of the algorithm. Now, for the $\oplus$ operation between $G_1 \oplus \cdots \oplus G_h$ and G_{h+1} over the join set W, using array $Nodes$, we find a node α in the tree decomposition of $G_1 \oplus \cdots \oplus G_h$ whose bag contains W. Since we have the tree decomposition of G_{h+1}, and W is a clique of size at most three that must appear in some bag in any tree decomposition, we can find the node α' of the tree decomposition whose bag contains W by brute force search over all subsets of bags of size at most three. Simultaneously, we update array $Nodes$ by subsets of $V(G'')$ which form a clique and appear in bags of the tree decomposition of G_{h+1}. Then we add an edge between α and α'. As the number of nodes in a tree decomposition of G_{h+1} is $O(|V(G_{h+1})|)$ and each bag has size at most $3(j - i + 1)$ (and thus there are at most $O((j - i + 1)^3)$ choices for a subset of size at most three), this operation takes $O((j - i + 1)^3 \cdot |V(G_{h+1})|)$ time for G_{h+1}.

The claimed running time follows from the time required to determine a set of clique-sum operations, the time required to construct tree decompositions, the time needed to glue tree decompositions together and the fact that $\sum_{i=1}^{m} |V(G_i)| = O(|V(G'')|)$. Here we note that the only difference between the running times for the general algorithm and those for $K_{3,3}$-minor-free or K_5-minor-free graphs is the time required to determine a set of clique-sum operations ($O(n)$ time for the former graphs and $O(n^2)$ time for the latter graphs). The rest of the algorithm requires linear time for all single-crossing-minor-free graphs.

Finally, we prove that the width of the constructed tree decomposition of G'' is $3(j - i + 1) + c_H$. We use induction on m, the number of G_i's, where $G'' = G_1 \oplus G_2 \oplus \cdots \oplus G_m$. For $m = 1$, G_1 is either a planar graph of treewidth at most $3(j - i + 1)$ or a graph of treewidth at most c_H. In both cases the basis of the induction is true. We assume the induction hypothesis is true for $m = h$, and we prove the hypothesis for $m = h+1$. For $\tilde{G} = G_1 \oplus G_2 \oplus \cdots \oplus G_h$, $G'' = \tilde{G} \oplus G_{h+1}$. By the induction hypothesis, $\tilde{G}$ and G_{h+1} each have treewidth at most $3(j-i+1)+c_H$. We can then apply Lemma 3 to conclude that the treewidth of G'' is also at most $3(j-i+1)+c_H$, as needed to complete the proof. $\square$

5 Approximating treewidth

A large amount of effort has been put into determining treewidth, which is NP-complete even if we restrict the input graph to graphs of bounded degree [BT97], cocomparability graphs [ACP87,HM94], bipartite graphs [Klo96], or the complements of bipartite graphs [ACP87]. However, treewidth can be computed exactly in polynomial time for chordal graphs, permutation graphs [BKK95], circular-arc graphs [SSR94], circle graphs [Klo96], distance-hereditary graphs [BDK00], and for graphs of a fixed treewidth [Bod96].

From the approximation viewpoint, Bodlaender et al. [BGHK95] gave an $O(\log n)$-approximation algorithm for treewidth on general graphs. A famous open problem is whether treewidth can be approximated within a constant factor. Constant-factor approximations are possible on AT-free graphs [BT01,BKMT01] and on planar graphs. The approximation for planar graphs is a consequence of the polynomial-time algorithm given by Seymour and Thomas [ST94] for computing the parameter *branchwidth*, whose value approximates treewidth within a factor of 1.5. Using the notions of branchwidth and clique-sum decomposition, we demonstrate a polynomial-time algorithm that approximates within a constant factor the treewidth of any single-crossing-minor-free graph.

Analogous to the relationship between treewidth and tree decompositions, the notion of branchwidth is related to a decomposition based on the edges. A *branch decomposition* of a graph G is a pair (T, τ), where T is a tree with vertices of degree 1 or 3 and τ is a bijection from the set of leaves of T to $E(G)$. The *order* of an edge e in T is the number of vertices $v \in V(G)$ such that there are leaves t_1, t_2 in T in different components of $T(V(T), E(T)-e)$ with $\tau(t_1)$ and $\tau(t_2)$ both containing v as an endpoint. The *width* of (T, τ) is the maximum order over all edges of T, and the *branchwidth* of G, $\mathrm{bw}(G)$, is the minimum width over all branch decompositions of G (if $|E(G)| \leq 1$, we define the branchwidth to be 0; if $|E(G)| = 0$, then G has no branch decomposition; if $|E(G)| = 1$, then G has a branch decomposition consisting of a tree with one vertex and the width of this branch decomposition is considered to be 0). Figure 5 provides examples of branch decompositions.

We make use of an approximation algorithm for computing treewidth of planar graphs as one of two "base cases" in our algorithm for single-crossing-minor-free graphs. While it remains an open question whether there exists a polynomial-time constant-factor approximation algorithm for computing the treewidth of general graphs, the branchwidth of a planar graph can be computed in polynomial time.

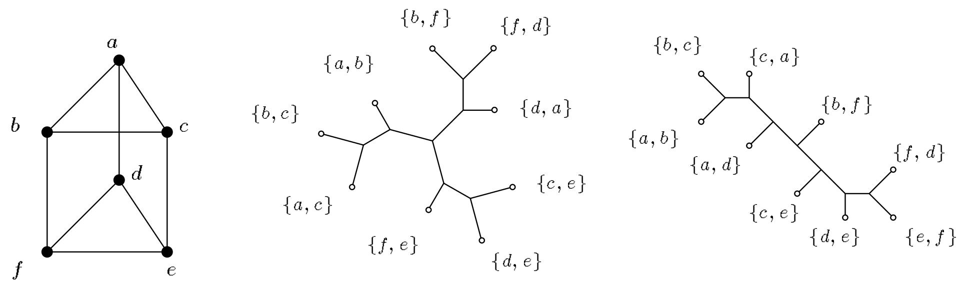

Fig. 5. A graph and two branch decompositions of it. The first has width 4 and the second has width 3.

Theorem 6. [ST94], (Sections 7 and 9) *One can construct an algorithm that, given a planar graph G,*

1. *computes in $O(n^2 \log n)$ time the branchwidth of G; and*
2. *computes in $O(n^4)$ time a branch decomposition of G with optimal width.*

Theorem 7. *One can construct an algorithm that, given a planar graph G,*

1. *computes in $O(n^2 \log n)$ time a value k with $tw(G) \leq k \leq \frac{3}{2}\mathrm{tw}(G)$; and*
2. *computes in $O(n^4)$ time a tree decomposition D of width k of G, where $\mathrm{tw}(G) \leq k \leq \frac{3}{2}\mathrm{tw}(G) + 1$.*

Proof. The proof is straightforward using $O(|E(G)|^2)$ algorithms of Robertson and Seymour [RS91] which convert a branch decomposition of width at most b to a tree decomposition of width at most $\frac{3}{2}b - 1$, and convert a tree decomposition of width at most k to a branch decomposition of width at most $k + 1$. □

The main theorem of this section relies on Theorem 7 as well as clique-sum decompositions.

Theorem 8. *For any single-crossing graph H, we can construct an algorithm that, given an H-minor-free graph as input, outputs in $O(n^4)$ time a tree decomposition of G of width k where $\mathrm{tw}(G) \leq k \leq \frac{3}{2}\mathrm{tw}(G) + 1$.*

Proof. The algorithm consists of the following four steps:

Step 1: Let G be a graph excluding a single-crossing graph H. By Theorem 2, we can obtain a clique-sum decomposition $G = G_1 \oplus G_2 \oplus \cdots \oplus G_m$ where each G_i, $1 \leq i \leq m$, is a minor of G and is either a planar graph or a graph of treewidth at most c_H. According to the same theorem, this step can be executed in $O(n^4)$ time. Let B be the set of the indices of the bounded treewidth components and P be the set of planar components: $B = \{i \mid 1 \leq i \leq m,\ \mathrm{tw}(G_i) \leq c_H\}$, $P = \{1, \ldots, m\} - B$.

Step 2: By Theorem 7, we can construct, for any $i \in P$, a tree decomposition $TD(G_i)$ of G_i with width k_i and such that

$$\mathrm{tw}(G_i) \leq k_i \leq \tfrac{3}{2}\mathrm{tw}(G_i) + 1 \quad \text{for all } i \in P. \tag{5}$$

The construction of each of these tree decompositions takes $O(|V(G_i)|^4)$ time. Because $m \le n$ and $\sum_{1\le i\le m} |V(G_i)| = O(n)$, (it is simple to prove this by induction looking at the sizes of the components in the recurrences used in the proof of Theorem 2) the total time for this step is $O(n^4)$.

Step 3: Using Bodlaender's algorithm [Bod96], for any $i \in B$, we can obtain a tree decomposition of G_i with minimum width k_i in linear time, where the hidden constant depends only on c_H. Combining (5) with the fact that $\mathrm{tw}(G_i) = k_i$ for each $i \in B$, we obtain

$$\mathrm{tw}(G_i) \le k_i \le \tfrac{3}{2}\mathrm{tw}(G_i) + 1 \quad \text{for all } i \in \{1, \dots, m\}. \tag{6}$$

Step 4: Now that we have tree decompositions $TD(G_i)$ of each G_i, we glue them together using the construction given in the proof of Lemma 3, as detailed in Theorem 5. In this way, we obtain a tree decomposition of G that has size $k = \max\{k_i \mid 1 \le i \le m\}$. Combining this equality with (6), we have

$$\max\{\mathrm{tw}(G_i) \mid i = 1, \dots, m\} \le k \le \tfrac{3}{2} \max\{\mathrm{tw}(G_i) \mid i = 1, \dots, m\} + 1. \tag{7}$$

To see that this step can be executed in $O(n^4)$ time, we observe that as described in Theorem 5, for each of the $O(|V(G)|)$ nodes in the tree decomposition, we execute a brute force search in the array $Nodes$, indexed by all $O(|V(G)|^3)$ subsets of $V(G)$ of size at most three.

Finally, we prove that the algorithm is a 1.5-approximation. By Lemma 3, we have $\mathrm{tw}(G) \le \max\{\mathrm{tw}(G_i) \mid i = 1, \dots, m\}$. By Theorem 2, each G_i is a minor of G and therefore $\mathrm{tw}(G_i) \le \mathrm{tw}(G)$ (since the class of graphs of treewidth at most k is minor-closed). Thus, $\mathrm{tw}(G) = \max\{\mathrm{tw}(G_i) \mid i = 1, \dots, m\}$ and from (7) we conclude that $\mathrm{tw}(G) \le k \le \frac{3}{2}\mathrm{tw}(G) + 1$ and the theorem follows. □

Using the same approach as Theorem 8, one can prove a potentially stronger theorem:

Theorem 9. *If we can compute the treewidth of any planar graph in polynomial time, then we can compute the treewidth of any single-crossing-minor-free graph in polynomial time.*

Proof. We use the polynomial-time algorithm for computing treewidth of planar graphs in Step 2 of the algorithm described in the proof of Theorem 8. □

6 Polynomial-time approximation schemes

6.1 General schemes for approximation on special classes of graphs

A polynomial-time approximation scheme for a problem is a family $\{\mathcal{A}_\epsilon\}$ of algorithms, where $\mathcal{A}_\epsilon$ is a $(1+\epsilon)$ approximation for the problem that runs in time polynomial in the length of its input (for fixed ϵ). Inherent in the design of many polynomial-time approximation schemes for NP-complete graph problems is the restriction of inputs to graph classes that guarantee additional structural properties. Early work in the area demonstrated the possibility of using planarity to obtain approximation schemes [LT80], later generalized to graphs without a fixed minor [AST90]. These approaches are impractical; a performance ratio of two for the independent set problem is achievable only for planar graphs of at least $2^{2^{400}}$ vertices [CNS82].

Practical approximation schemes for planar graphs were developed by Baker [Bak94], who formed a decomposition of G into overlapping k-outerplanar subgraphs. For any planar embedding, vertices can be put into layers by iteratively removing vertices on the outer face of the graph: vertices removed at the ith iteration are assigned to layer i. Since a k-outerplanar graph is decomposed into at most k layers, a k-outerplanar subgraph can be formed by removing vertices with layer number congruent to i mod k. Baker's technique immediately implies $(1 + 1/k)$-factor approximation algorithms for many problems that can be solved exactly on k-outerplanar graphs (e.g. maximum independent set, minimum dominating set, and minimum vertex cover), as it suffices to solve the problem exactly on each of the k subgraphs (one for each value of i) and return the best of the k results. Consider an optimal answer in the full graph. Since the sets of removed vertices partition the graph, one of these sets removes at most $1/k$ of the vertices in the answer, and the exact solution on the corresponding subgraph will be a $(1+1/k)$-approximation to the optimal answer for the full graph.

Chen [Che98] later generalized Baker's approach to form approximation algorithms of ratio $1 + 1/\log n$ for problems on $K_{3,3}$-minor-free graphs and K_5-minor-free graphs; due to the types of layers formed, these results were exclusively for maximization problems. Eppstein [Epp00] showed that Baker's technique can be extended by replacing bounded outerplanarity with bounded local treewidth. As with k-outerplanar graphs, a wide range of NP-complete problems can be solved in linear time on graphs of bounded local treewidth. The decomposition by deleting every kth face is replaced by deleting every kth level of a breadth-first tree of G, keeping in mind the fact that that the treewidth of the resulting graphs is a function of k.

6.2 Approximation schemes for single-crossing-minor-free graphs

In this section we use the bound on local treewidth established in Theorem 3 to obtain polynomial-time approximation schemes. Among NP-optimization problems, we mainly focus on those problems which are also *hereditary*, namely, problems which determine a property that if valid for an input graph is also valid for any induced subgraph of the input. For a property π, the maximum induced subgraph problem MISP(π) is finding a maximum induced subgraph with the property; in the weighted version (WMISP(π)), the input graph has weights on its vertices and the goal is to find a maximum weight induced subgraph with the property. For example, we might search for an induced subgraph of maximum size that is chordal, acyclic, without cycles of a specified length, without edges, of maximum degree $r \geq 1$, bipartite or a clique [Yan78]. For exact definitions of various NP-hard problems in this paper, the reader is referred to Garey and Johnson's seminal book [GJ79].

Yannakakis has shown that for many natural hereditary properties π, MISP(π) is NP-complete even when restricted to planar graphs [Yan78]. Using the results of Section 4, we obtain approximation algorithms for both maximization and minimization problems such as the maximum independent set problem, the minimum vertex cover problem and the minimum dominating set problem on single-crossing-minor-free graphs.

Theorem 10. *For G a non-negative vertex-weighted single-crossing-minor-free graph excluding H, $k \geq 1$ an integer, and $Time_\pi(w,n)$ the nondecreasing worst-case running time of WMISP(π) over an n-vertex partial w-tree whose tree decomposition is given, the maximization problem WMISP(π) for a hereditary property π over G admits a polynomial-time approximation scheme of ratio $1+1/k$ with worst-case running time in $O(k \cdot |V|^4 + k \cdot Time_\pi(3(k-1)+c_H, |V|))$. The running time improves to $O(k \cdot |V| + k \cdot Time_\pi(3(k-1)+4, |V|))$ for G $K_{3,3}$-minor-free and $O(k \cdot |V|^2 + k \cdot Time_\pi(3(k-1)+4, |V|))$ for G K_5-minor-free.*

Proof. Our algorithm proceeds by creating k subgraphs of G, solving the problem on each of the subgraphs, and returning the best solution for any of the subgraphs as the solution for all of G. We make use of the locally bounded treewidth of G in order to specify layers from which the subgraphs are derived and to prove that each subgraph has bounded treewidth.

Given an assignment of vertices to layers numbered $1, 2, \ldots$ created by breadth-first search (layer i is all vertices at depth i), we use $L_{i,j}$ to denote the consecutive layers numbered $(j-1)k+i$ through $jk+i-2$ for $1 \leq i \leq k$ and $j \geq 0$ where for convenience a layer is defined to be empty when its number is not between zero and the total number of layers. Furthermore, we let $\mathcal{L}_i = \bigcup_{j \geq 0} L_{i,j}$ and $G_i = G[\mathcal{L}_i]$. As neither $L_{i,j}$ nor $L_{i,j+1}$ contains the layer numbered $jk+i-1$ and all edges appear between consecutive layers, there are no edges between $L_{i,j}$ and $L_{i,j+1}$. Moreover, as no vertices in layer $i-1$ appear in $\mathcal{L}_h$ for $h = i \bmod k$, each vertex appears in exactly $k-1$ of the $\mathcal{L}_i$'s or G_i's, a fact we will use later in the proof.

We next use the bound on the treewidth of each G_i to obtain Opt_i, the maximum weighted solution of WMISP(π) on each G_i, $1 \leq i \leq k$. In particular, we construct a tree decomposition of width $3(k-1)+c_H$ for each G_i by adding edges between tree decompositions for each $G[L_{i,j}]$, which in turn can be formed in $O(n^4)$ time using Theorem 5 ($O(n)$ for $K_{3,3}$-minor-free graphs or $O(n^2)$ time for K_5-minor-free graphs). The fact that the graphs $G[L_{i,j}]$ are disjoint means that the process of adding edges to form a single tree is straight-forward. As in Lemma 3, by the definition of $Time_\pi(w,n)$ as a nondecreasing function, since $|V(G_i)| \leq |V(G)|$, Opt_i can be determined in $Time_\pi(3(k-1)+c_H, |V(G)|)$. The running time is $Time_\pi(3(k-1)+4, |V(G)|)$ for G $K_{3,3}$-minor-free or K_5-minor-free.

Finally, we take Opt_m, the solution with maximum weight among the Opt_i's, as our solution for graph G, and prove the ratio bound by showing that $\frac{weight(Opt)}{weight(Opt_m)} \leq \frac{k}{k-1}$, or $k \cdot weight(Opt_m) \geq (k-1) \cdot weight(Opt)$, where Opt is the maximum weighted solution on graph G. By observing that $Opt_m \geq Opt_i$ for each value of i, we show that $k \cdot weight(Opt_m) \geq \sum_{i=1}^{k} weight(Opt_i)$. Since π is hereditary, $weight(Opt_i) \geq weight(Opt \cap \mathcal{L}_i)$, and hence $\sum_{i=1}^{k} weight(Opt_i) \geq \sum_{i=1}^{k} weight(Opt \cap \mathcal{L}_i)$. Finally, we recall that each vertex appears in exactly $k-1$ of the $\mathcal{L}_i$'s, from which we can conclude $\sum_{i=1}^{k} weight(Opt \cap \mathcal{L}_i) = (k-1) \cdot weight(Opt)$, as needed to conclude the proof that $k \cdot weight(Opt_m) \geq (k-1) \cdot weight(Opt)$.

The claimed running time follows immediately from the time to construct the tree decomposition, the time to solve WMISP(π) for each G_i, and the number of G_i's. □

Corollary 2. *For G a non-negative vertex-weighted single-crossing-minor-free graph excluding H, the maximum independent set problem admits a polynomial-time approximation scheme of ratio $1 + 1/k$ with running time $O(k \cdot 2^{3k} \cdot n + k \cdot n^4)$. The running time improves to $O(k \cdot 2^{3k} \cdot n)$ for G $K_{3,3}$-minor-free and $O(k \cdot 2^{3k} \cdot n + k \cdot n^2)$ for G K_5-minor-free.*

Proof. Using dynamic programming on a tree decomposition, this problem can be solved in $O(2^w \cdot n)$ time, over each n-vertex partial w-tree whose tree decomposition is given [AN02]. Thus the result follows from Theorem 10 for $Time_\pi(w, n) = O(2^w \cdot n)$. □

Below we give examples that show how our result can be applied to NP-minimization problems, e.g., the minimum vertex cover problem and the minimum dominating set problem. The main difference between these two results and the previous one is that the previous construction avoided overlap between the various sets $L_{i,j}$ of consecutive layers, in the minimization setting we need to enforce overlap in order to achieve the approximation guarantee. The ideas of the proofs of Theorems 11 and 12 follow ideas of Grohe [Gro] for general graphs of locally bounded treewidth, which are in fact Baker's ideas for planar graphs. Grohe [Gro] remarks that he "never specified the exponents and coefficients of the polynomials bounding the running times of our algorithms; they seem to be enormous." The exponents are modest in our results; while we have not quantified our constants (coefficients) either, we suspect they will be more reasonable due to the more restricted setting, and can be made even more so by use of approximation algorithms for treewidth, as described earlier.

Theorem 11. *For G a single-crossing-minor-free graph and any integer $k \geq 1$, the minimum weighted vertex cover problem admits a polynomial-time approximation scheme of ratio $1 + 1/k$ with worst-case running time $O(k \cdot 2^{3k} \cdot n + k \cdot n^4)$. The running time improves to $O(k \cdot 2^{3k} \cdot n)$ for G $K_{3,3}$-minor-free and $O(k \cdot 2^{3k} \cdot n + k \cdot n^2)$ for G K_5-minor-free.*

Proof. As in the proof of Theorem 10, we first decompose graph G into induced subgraphs, each of bounded treewidth, on which we will solve the problem. In a slight variation from the definition used in that proof, for $1 \leq i \leq k$ and $j \geq 0$, we define $L_{i,j} = L[(j-1)k + i, jk + i]$ and $G_{i,j} = G[L_{i,j}]$. The following observations, which will be used later in the proof, are all consequences of the definition: vertices in the layer numbered $jk + i$ appear in both $G[L_{i,j}]$ and $G[L_{i,j+1}]$; for fixed i, each vertex appears in at most two $L_{i,j}$'s; for fixed i, each edge of G appears in at least one $G_{i,j}$; and each vertex appears in $k+1$ (successive) sets $L_{i,j}$.

As in Theorem 10, we wish to obtain tree decompositions of the subgraphs. By Theorem 5, we construct a tree decomposition of $G[L_{i,j}]$ of width $3(k+1)+c_H$ in $O(|V(G[L_{i,j}])|^4)$ time. A tree decomposition of width $3(k+1)+4$ is constructed in $O(|V(G[L_{i,j}])|)$ time for a $K_{3,3}$-minor-free graph and in $O(|V(G[L_{ij}])|^2)$ time for K_5-minor-free graph. Since each vertex of G appears in at most two $G[L_{i,j}]$'s for a fixed i (as observed above), constructing tree decompositions of all $G[L_{i,j}]$'s takes $O(|V(G)|^4)$ (or $O(|V(G)|)$ or $O(|V(G)|^2)$) time.

We make use of the layers to obtain a set of possible solutions for G. We first solve the minimum vertex cover problem for each $G_{i,j}$ to obtain a solution $Opt_{i,j}$, and then define Opt_i as $Opt_i = \cup_{j\geq 0} Opt_{i,j}$. To see that Opt_i is a vertex cover for all of G, we recall that for a fixed i each edge of G appears in at least one $G_{i,j}$, and thus will be covered by at least one of the partial solutions $Opt_{i,j}$.

We show that Opt_m, the solution with minimum weight among the Opt_i's, yields an approximation ratio of $1+1/k$, that is, $\frac{weight(Opt_m)}{weight(Opt)} \leq \frac{k+1}{k}$ (or $k \cdot weight(Opt_m) \leq (k+1)\cdot weight(Opt)$) for Opt the minimum weighted solution on graph G. By the definition of Opt_m as the solution of minimum weight and the definition of Opt_i as the union over $Opt_{i,j}$'s, we can show the following:

$$k \cdot weight(Opt_m) \leq \sum_{i=1}^{k} weight(Opt_i) \leq \sum_{i=1}^{k}\sum_{j\geq 0} weight(Opt_{i,j}).$$

We next observe that since $Opt \cap L_{i,j}$ is a vertex cover for $G_{i,j}$ and $Opt_{i,j}$ is a minimum vertex cover for $G_{i,j}$ we can conclude that $weight(Opt_{i,j}) \leq weight(Opt \cap L_{i,j})$, or

$$\sum_{i=1}^{k}\sum_{j\geq 0} weight(Opt_{i,j}) \leq \sum_{i=1}^{k}\sum_{j\geq 0} weight(Opt \cap L_{i,j}).$$

Finally, since by our earlier observation each vertex of Opt appears in $k+1$ $L_{i,j}$'s,

$$\sum_{i=1}^{k}\sum_{j\geq 0} weight(Opt \cap L_{i,j}) = (k+1)\cdot weight(Opt),$$

completing the proof of correctness.

To see that the claimed running time holds, it suffices to show that for a fixed i, the time to compute Opt_i is $O(2^{3k}\cdot|V(G)|)$, the total following from the time to construct tree decompositions and the number of Opt_i's. The minimum vertex cover problem can be solved in $O(2^w \cdot n)$ time over each n-vertex partial w-tree whose tree decomposition is given [AFN01]. Thus, computing Opt_{ij} takes $O(2^{3k}\cdot|V(G[L_{i,j}])|)$ time on graph G_{ij}, and hence for fixed i, as each vertex appears in at most two $L_{i,j}$'s, Opt_i can be determined in $O(2^{3k}\cdot|V(G)|)$ time, as claimed. □

To find an approximation algorithm for the dominating set (DS) problem, we first introduce a generalized version of the dominating set problem (Definition 3) and show

how we can solve this problem in linear time for graphs of constant treewidth (Lemma 8). Then we use the algorithm for solving this problem to obtain a polynomial-time approximation scheme for the dominating set problem (Theorem 12).

Definition 3. *The* generalized dominating set (GDS) *problem is defined as follows. Given a vertex-weighted graph G and a set $I \subseteq V(G)$, determine a subset W of $V(G)$ of minimum weight with the property that for every $u \in I - W$ there is a $w \in W$ such that $\{u, w\} \in E(G)$.*

The generalized dominating set problem reduces to the standard dominating set problem for $I = V(G)$. In both problems, we say that a vertex v is *dominated* by a vertex w if there is an edge between v and w and w is in W.

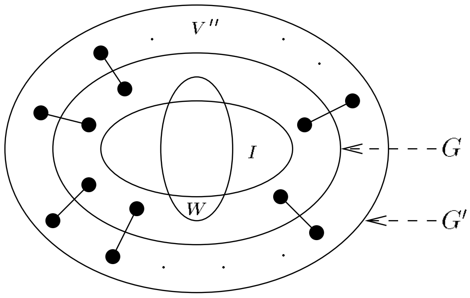

Fig. 6. Set V'' and graphs G and G' defined in the proof of Lemma 8.

Lemma 8. *The GDS problem for given graph G and set I can be solved in time $O(4^w \cdot |V(G)|)$ when a tree decomposition of width w for G is given.*

Proof. We solve the GDS problem by a reduction to the DS problem, and make use of a solution to the DS problem in time $O(4^w \cdot |V(G)|)$, given a tree decomposition of width w of a non-negative vertex-weighted graph G [AN02]. Given an input G to the GDS problem, we form G' as input to the DS problem by first setting $G' = G$ and then for each vertex $v \in V(G) - I$, adding another vertex v' with weight zero connected to v, forming the vertex set V'', as illustrated in Figure 6. It is then straightforward to verify that if W is a solution to the GDS problem, then W' formed by adding all vertices of V'' to W is a solution to the DS problem of the same weight.

To apply the solution to the dominating set problem, we must provide a tree decomposition of G'. We form $TD(G')$ from $TD(G)$ by adding, for each $v' \in V''$ connected to a vertex $v \in V(G)$, a node whose bag contains v' and v and an edge from the new node to a node of $TD(G)$ whose bag contains v. As treewidth of G' is the same as that of G, and $|V(G')| \leq 2|V(G)|$, the GDS problem can thus be solved in $O(4^w \cdot |V(G)|)$ time using the algorithm for the DS problem. $\square$

Theorem 12. *For G a single-crossing-minor-free graph and any integer $k \geq 1$, the minimum weighted dominating set problem admits a polynomial-time approximation* *scheme of ratio* $1+2/k$ *with worst-case running time* $O(k\cdot 4^{3k}\cdot n+k\cdot n^4)$. *The running time improves to* $O(k\cdot 4^{3k}\cdot n)$ *for* G $K_{3,3}$*-minor-free and* $O(k\cdot 4^{3k}\cdot n+k\cdot n^2)$ *for* G K_5*-minor-free.*

Proof. The methodology is similar to that of Theorem 11, but with subtle and important differences. Here the set of layers $L_{i,j}$ contains layers $(j-1)k+i-1$ through $jk+i$ for $1\le i\le k$ and $j\ge 0$. The properties we require are: for fixed i, $L_{i,j}$ and $L_{i,j+1}$ intersect only in two consecutive layers; for fixed i, each vertex appears in at most two $L_{i,j}$'s; and each vertex appears in at most $k+2$ (successive) sets L_{ij}. Using the argument developed in Theorem 11, as a consequence of the properties above and Theorem 5, we construct tree decompositions of all $G[L_{ij}]$'s in $O(n^4)$ time ($O(n)$ and $O(n^2)$ time for $K_{3,3}$-minor-free and K_5-minor-free graphs G).

To construct solutions Opt_i to the problem on G, we make use of generalized dominating set. To this end, the *interior* $I_{i,j}$ of $L_{i,j}$ is defined as layers $(j-1)k+i$ through $jk+i-1$. For $1\le i\le k$ and $j\ge 0$, by Lemma 8 we can obtain in $O(4^{3k}\cdot|V(G[L_{ij}])|)$ time the set $Opt_{ij}\subseteq L_{ij}$ of minimum weight that dominates $I_{ij}-Opt_{ij}$. We then define $Opt_i=\cup_{j\ge 0}Opt_{ij}$ and use the property above that for fixed i, each vertex of G appears in at most two L_{ij}'s to compute each Opt_i in $O(4^{3k}\cdot n)$ time. In addition, since for fixed i, each vertex appears one time in an interior set I_{ij}, we can conclude that Opt_i is a dominating set for G.

Finally, we define Opt_m to be the solution of minimum weight among the Opt_i's and show that $\frac{weight(Opt_m)}{weight(Opt)}\le\frac{k+2}{k}=1+2/k$, or $k\cdot weight(Opt_m)\le(k+2)\cdot weight(Opt)$, for Opt the minimum weight dominating set over G. By the definitions of Opt_m and Opt_i, we can show the following:

$$k\cdot weight(Opt_m)\le\sum_{i=1}^{k}weight(Opt_i)\le\sum_{i=1}^{k}\sum_{j\ge 0}weight(Opt_{ij}).$$

We now show that $Opt\cap L_{ij}$ dominates $I_{ij}-Opt_{ij}$: since Opt is a dominating set over G, there exists an edge $\{u,v\}$, $v\in Opt$, for each $u\in I_{ij}-(Opt\cap L_{ij})$, where $v\in L_{ij}$ as by construction u has no neighbors outside of L_{ij}. Since Opt_{ij} has minimum weight, $weight(Opt_{ij})\le weight(Opt\cap L_{ij})$, or $\sum_{i=1}^{k}\sum_{j\ge 0}weight(Opt_{ij})\le\sum_{i=1}^{k}\sum_{j\ge 0}weight(Opt\cap L_{ij})$. By noting the property that every vertex appears in exactly $k+2$ sets L_{ij}, we obtain the following:

$$k\cdot weight(Opt_m)\le\sum_{i=1}^{k}\sum_{j\ge 0}weight(Opt\cap L_{ij})=(k+2)\cdot weight(Opt).$$

The running time follows immediately from the time needed to construct the tree decompositions, the number of Opt_i's and the time to compute each of them. □

Theorem 13. *For single-crossing-minor-free graphs, there are polynomial-time approximation algorithms whose solutions converge toward optimal as* n *increases for maximum independent set, minimum vertex cover and minimum dominating set.*

Proof. The running time of algorithms introduced in Corollary 2 and Theorems 11 and 12 is $O(c^k n + n^4)$ where k is a parameter and c is a constant. Now, by taking $k = \lceil \log n \rceil$, we obtain polynomial-time approximation algorithms of ratio $1 + 1/(\log n)$ (or $1 + 2/(\log n)$ for dominating set). As both $1/(\log n)$ and $2/(\log n)$ decrease as n increases, the solutions converge toward optimal as n increases. □

6.3 Further applications

Our techniques are applicable to other problems solved by Eppstein using the adaptation of Baker's approach [Epp99]. For example, there exists an algorithm that determines whether a fixed pattern H is a subgraph of a single-crossing-minor-free graph G in $O(2^{O(|V(H)| \log |V(H)|)} \cdot |V(G)|)$ time. The algorithm makes use of locally bounded treewidth; since H is of constant size, if it is a subgraph of G, it is a subgraph of a subgraph of G consisting of a constant number of consecutive layers. The problem reduces to $O(n)$ subgraph isomorphism problems, each solvable in linear time for H fixed and G of bounded treewidth. It is possible to obtain linear-time algorithms on single-crossing-minor-free graphs for problems Eppstein solved by exploiting locally bounded treewidth: these include problems such as determining the diameter of a bounded-diameter graph and determining girth of a bounded-girth graph.

The approaches of Theorems 10, 11, and 12 can be used to adapt Eppstein's algorithms for other problems such as minimum edge dominating set, maximum triangle matching, maximum H-matching and maximum tile salvage to obtain polynomial-time approximation schemes for single-crossing-minor-free graphs [Epp00]. Moreover, for several hereditary maximization problems, the function $Time_\pi(w, n)$ introduced in Theorem 10 is $O(c^{p(k)} \cdot q(n))$, where c is a constant and p and q are polynomials of low degree [Bod88,TP93,TP97], hence yielding ratio $1 + 1/k$ polynomial-time approximation schemes for each such problem.

7 Fixed-parameter algorithms

By making use of known connections between locally bounded treewidth and *fixed-parameter tractability*, we derive algorithms for variants of dominating set on single-crossing-minor-free graphs. Downey and Fellows [DF99] introduced the notion of *fixed-parameter tractability* as a way to categorize the complexity of NP-hard problems based on one or more *parameters* of a problem. A problem is *fixed-parameter tractable* if the combinatorial explosion inherent in the problem can be attributed to the parameter(s), so that although the running time may be exponential in the parameter(s), it is polynomial in the problem size. One example is the problem of finding a vertex cover of size k, as k is the parameter and there is an algorithm running in time $O(kn + 1.2852^k)$ [CKJ01].

In this section, we extend to single-crossing-minor-free graphs a solution to k-dominating set on planar graphs, that is, the problem of finding a set S of at most k vertices such that each vertex in $V(G) - S$ has at least one neighbor in S. As a consequence of Lemma 1 and Theorem 3, any problem that is easily solved on minor-closed classes or graphs of locally bounded treewidth can be easily solved on single-crossing-minor-free graphs. In particular, Frick and Grohe demonstrated that for any property definable in first-order logic and for any minor-closed graph class of locally bounded treewidth, there exists a linear-time algorithm that determines whether a graph in the class has the property [FG01]. As k-dominating set is first-order expressible, the work of Frick and Grohe implies that k-dominating set can be decided in linear time on single-crossing-minor-free graphs.

Unfortunately, the constant hidden in the linear-time algorithm is prohibitively large, due in part to the constants involved in Bodlaender's linear-time tree decomposition algorithm [Bod96]. Alber et al. made use of Baker's approach to solve k-dominating set on planar graphs in time $O(4^{6\sqrt{34k}} \cdot n)$ [ABF+02]; it makes use of a smaller constant, and hence is practical for small values of k. Using the fact that the exponential term is sublinear in the parameter, this result has been the basis for exponential speedup of fixed-parameter algorithms for many NP-complete problems on planar graphs [CKL01,KLL01,AFN01]. We extend their result to single-crossing-minor-free graphs in Theorem 14 below.

Theorem 14. *The problem of k-dominating set for fixed k can be solved in $O(4^{9k} \cdot n + n^4)$ time for a single-crossing-minor-free graph G ($O(4^{9k} \cdot n)$ time for G $K_{3,3}$-minor-free and $O(4^{9k} \cdot n + n^2)$ time for G K_5-minor-free).*

Proof. We show that if G has a k-dominating set, then we can obtain a bound on its treewidth, which in turn can be used to apply an algorithm for finding a dominating set in such graphs. To see that having a k-dominating set implies bounded treewidth, we consider the layers of the vertices of G in the ordering generated by breadth-first search. Since a particular vertex in the dominating set can dominate vertices in any of three layers (its own, the previous, or the next layer), the k elements of the dominating set can dominate vertices on at most $3k$ layers. As each vertex of G is dominated, the total number of layers in G is at most $3k$.

Since G is a single-crossing-minor-free graph, we can use Theorem 5 to form a tree decomposition of $G[L[0, 3k]] = G$ of width at most $3(3k + 1) + c_H = 9k + 3 + c_H$ in $O(n^4)$ time. A tree decomposition of width at most $3(3k + 1) + 4 = 9k + 7$ can be formed in $O(n)$ time for G $K_{3,3}$-minor-free and $O(n^2)$ time for G K_5-minor-free. It has been shown that a minimum dominating set can be found in $O(4^w \cdot n)$ time given a tree decomposition of width w [AN02]; combined with the time to compute the tree decomposition, we obtain the claimed running times. $\square$

Similar techniques were used to solve related problems on planar graphs. Alber et al. [AFF+01] demonstrate an algorithm for dominating set on planar graphs running in time $O(8^k n)$; Ellis et al. [EFF02] gave an algorithm on graphs of bounded genus g running in time $O((24g^2 + 24g + 1)^k n^2)$. Recently, Demaine et al. [DHT02] extended planar graph results [ABF+02] to problems on single-crossing-minor-free graphs, proving the following general theorem (the reader is refered to their paper for problem definitions):

Theorem 15. *Given the clique-sum decomposition of a single-crossing-minor-free graph G, there are algorithms that in $O(2^{O(\sqrt{k})} \cdot n)$ time decide whether graph G has a subset of vertices of size k which is a dominating set, dominating set with property P, vertex cover, edge-dominating set, maximal matching, maximum independent set, clique-transversal set, kernel, feedback vertex set, or satisfies a series of vertex removal properties.*

Applying Theorem 2, we obtain:

Corollary 3. *There are algorithms that in $O(2^{O(\sqrt{k})} \cdot n + n^4)$ time decide whether any single-crossing-minor-free graph G has a subset of size k with one of the properties mentioned in Theorem 15.*

Though this result improves Theorem 14, the technique here is simpler and forms a basis for the newer results.

8 Conclusions and future work

In this paper, we introduced the class of single-crossing-minor-free graphs, which contains $K_{3,3}$-minor-free graphs and K_5-minor-free graphs, generalizations of planar graphs, and demonstrated structural properties which gave rise to new algorithms. We showed that single-crossing-minor-free graphs have linear local treewidth and demonstrated how to obtain a tree decomposition of a fixed number of layers. Algorithms obtained using these properties include both approximation algorithms (e.g. a 1.5-approximation algorithm for treewidth and many polynomial-time approximation schemes) and fixed-parameter algorithms (e.g. a k-dominating set algorithm).

Extensions to the structural results could include finding clique-sum characterizations of graphs such as graphs excluding a double-crossing graph (or a graph with a bounded number of crossings) as a minor. For each such class, polynomial-time constructions of the decompositions could be used for further algorithmic development.

Notice that the algorithm of Theorem 2 in Section 3.3 can serve as a general heuristic for the computation or approximation of treewidth in a graph when the resulting 3-connected components without strong 3-cuts are graphs whose treewidth can be computed or approximated efficiently. Additional approximation algorithms might include those which determine graph properties other than treewidth. As a consequence of Theorem 6, treewidth is a 1.5-approximation on branchwidth, which immediately implies the existence of a 2.25-approximation for branchwidth of single-crossing-minor-free graphs. It might be possible to use a clique-sum decomposition to obtain a better approximation or an exact algorithm for branchwidth. We believe that by using an approach similar to that described by Kezdy and McGuinness [KM92], it is possible to obtain a polylogarithmic parallel algorithm that constructs a series of clique-sum operations as in Theorem 2; details remain to be worked out.

We suspect that Baker's approach can be applied to obtain practical polynomial-time approximation schemes for other problems, such as variations on dominating sets [ABF+02] that have been solved on k-outerplanar graphs or graphs of bounded treewidth [BP92,DST96,ABF+02]. By considering other NP-complete problems that have good algorithms for planar graphs and graphs of bounded treewidth, we may be able to extend the range of problems to which using clique-sum decomposition techniques may be applied; some results in this direction have already been found [DHT02].

References

[ABF+02] Jochen Alber, Hans L. Bodlaender, Henning Fernau, Ton Kloks, and Rolf Niedermeier. Fixed parameter algorithms for dominating set and related problems on planar graphs. *Algorithmica*, 33:461–493, 2002.

[ACP87] Stefan Arnborg, Derek G. Corneil, and Andrzej Proskurowski. Complexity of finding embeddings in a k-tree. *SIAM J. Algebraic Discrete Methods*, 8(2):277–284, 1987.

[AFF+01] Jochen Alber, Hongbing Fan, Michael R. Fellows, Henning Fernau, Rolf Niedermeier, Frances A. Rosamond, and Ulrike Stege. Refined search tree technique for dominating set on planar graphs. In *Proceedings of 26th International Symposium on Mathematical Foundations of Computer Science*, volume 2136 of *LNCS*, pages 111–122. Springer, 2001.

[AFN01] Jochen Alber, Henning Fernau, and Rolf Niedermeier. Parameterized complexity: exponential speed-up for planar graph problems. In *Proceedings of the 28th Annual International Colloquium on Automata, Languages, and Programming*, volume 2076 of *LNCS*, pages 261–272, 2001.

[Ami01] Eyal Amir. Efficient approximation for triangulation of minimum treewidth. In *Proceedings of the 17th Annual Conference on Uncertainty in Artificial Intelligence*, pages 7–15. Morgan Kaufmann, 2001.

[AN02] Jochen Alber and Rolf Niedermeier. Improved tree decomposition-based algorithms for domination-like problems. In *Proceedings of LATIN 2002, Fifth Latin American Symposium on Theoretical Informatics*, volume 2286 of *LNCS*, pages 613–628, 2002.

[AP86] Stefan Arnborg and Andrzej Proskurowski. Characterization and recognition of partial 3-trees. *SIAM J. Algebraic Discrete Methods*, 7(2):305–314, 1986.

[Asa85] Takao Asano. An approach to the subgraph homeomorphism problem. *Theoret. Comput. Sci.*, 38(2-3):249–267, 1985.

[AST90] Noga Alon, Paul Seymour, and Robin Thomas. A separator theorem for for graphs with excluded minor and its applications. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing*, pages 293–299, 1990.

[Bak94] Brenda S. Baker. Approximation algorithms for NP-complete problems on planar graphs. *J. Assoc. Comput. Mach.*, 41(1):153–180, 1994.

[BDK00] Hajo J. Broersma, Elias Dahlhaus, and Ton Kloks. A linear time algorithm for minimum fill-in and treewidth for distance hereditary graphs. *Discrete Appl. Math.*, 99(1-3):367–400, 2000.

[BGHK95] Hans L. Bodlaender, John R. Gilbert, Hjálmtýr Hafsteinsson, and Ton Kloks. Approximating treewidth, pathwidth, frontsize, and shortest elimination tree. *J. Algorithms*, 18(2):238–255, 1995.

[BK96] Hans L. Bodlaender and Ton Kloks. Efficient and constructive algorithms for the pathwidth and treewidth of graphs. *J. Algorithms*, 21(2):358–402, 1996.

[BKK95] Hans L. Bodlaender, Ton Kloks, and Dieter Kratsch. Treewidth and pathwidth of permutation graphs. *SIAM J. Discrete Math.*, 8(4):606–616, 1995.

[BKMT01] Vincent Bouchitté, Dieter Kratsch, Haiko Müller, and Ioan Todinca. On treewidth approximations. In *Proceedings of the 1st Cologne-Twente Workshop on Graphs and Combinatorial Optimization*, volume 8 of *Electronic Notes in Discrete Mathematics*, 2001.

[BM76] John A. Bondy and U. S. R. Murty. *Graph Theory with Applications.* American Elsevier Publishing Co., Inc., New York, 1976.

[BM93] Hans L. Bodlaender and Rolf H. Möhring. The pathwidth and treewidth of cographs. *SIAM J. Discrete Math.*, 6(2):181–188, 1993.

[Bod88] Hans L. Bodlaender. Dynamic programming algorithms on graphs with bounded treewidth. In *Proceedings of the 15th International Colloquium on Automata, Languages, and Programming*, volume 317 of *LNCS*, pages 105–118, 1988.

[Bod93] Hans L. Bodlaender. A tourist guide through treewidth. *Acta Cybernetica*, 11:1–23, 1993.

[Bod96] Hans L. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM J. Comput.*, 25(6):1305–1317, 1996.

[Bod97] Hans L. Bodlaender. Treewidth: algorithmic techniques and results. In *Proceedings of the 22nd International Symposium on Mathematical Foundations of Computer Science*, pages 29–36, 1997.

[Bod98] Hans L. Bodlaender. A partial k-arboretum of graphs of bounded treewidth. *Theoretical Computer Science*, 209:1–45, 1998.

[BP92] Thang Nguyen Bui and Andrew Peck. Partitioning planar graphs. *SIAM J. Comput.*, 21(2):203–215, 1992.

[BT97] Hans L. Bodlaender and Dimitrios M. Thilikos. Treewidth for graphs with small chordality. *Discrete Appl. Math.*, 79(1-3):45–61, 1997.

[BT01] Vincent Bouchitté and Ioan Todinca. Treewidth and minimum fill-in: grouping the minimal separators. *SIAM J. Comput.*, 31(1):212–232, 2001.

[BvLTT97] Hans L. Bodlaender, Jan van Leeuwen, Richard Tan, and Dimitrios Thilikos. On interval routing schemes and treewidth. *Information and Computation*, 139(1):92–109, 1997.

[Che98] Zhi-Zhong Chen. Efficient approximation schemes for maximization problems on $K_{3,3}$-free or K_5-free graphs. *J. Algorithms*, 26(1):166–187, 1998.

[CKJ01] Jianer Chen, Iyad A. Kanj, and Weijia Jia. Vertex cover: further observations and further improvements. *J. Algorithms*, 41(2):280–301, 2001.

[CKL01] Maw-Shang Chang, Ton Kloks, and Chuan-Min Lee. Maximum clique transversals. In *Proceedings of the 27th International Workshop on Graph-Theoretic Concepts in Computer Science*, pages 300–310, 2001.

[CNS82] Norishige Chiba, Takao Nishizeki, and Nobuji Saito. An approximation algorithm for the maximum independent set problem on planar graphs. *SIAM J. Comput.*, 11(4):663–675, 1982.

[DF99] Rodney G. Downey and Michael R. Fellows. *Parameterized Complexity*. Springer-Verlag, New York, 1999.

[DHT02] Erik D. Demaine, Mohammadtaghi Hajiaghayi, and Dimitrios M. Thilikos. Exponential speedup of fixed parameter algorithms on $K_{3,3}$-minor-free or K_5-minor-free graphs. In *Proceedings of the 13th Annual Symposium on Algorithms and Computation*, pages 262–273, 2002.

[Die89] Reinhard Diestel. Simplicial decompositions of graphs: a survey of applications. *Discrete Math.*, 75(1–3):121–144, 1989.

[Die91] Reinhard Diestel. Decomposing infinite graphs. *Discrete Math.*, 95:69–89, 1991.

[DST96] Josep Díaz, Maria J. Serna, and Jacobo Torán. Parallel approximation schemes for problems on planar graphs. *Acta Inform.*, 33(4):387–408, 1996.

[EFF02] John Ellis, Hongbing Fan, and Michael R. Fellows. The dominating set problem is fixed parameter tractable for graphs of bounded genus. In *Proceedings of the 8th Scandinavian Workshop on Algorithm Theory*, volume 2368 of *LNCS*, pages 180–189. Springer, 2002.

[Epp99] David Eppstein. Subgraph isomorphism in planar graphs and related problems. *J. Graph Algorithms Appl.*, 3(3):1–27, 1999.

[Epp00] David Eppstein. Diameter and treewidth in minor-closed graph families. *Algorithmica*, 27(3-4):275–291, 2000.

[FG01] Markus Frick and Martin Grohe. Deciding first-order properties of locally tree-decomposable structures. *Journal of the ACM*, 48(6):1184–1206, 2001.

[GJ79] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*. W. H. Freeman and Co., San Francisco, Calif., 1979.

[Gro] Martin Grohe. Local tree-width, excluded minors, and approximation algorithms. to appear, Combinatorica.

[HM94] Michel Habib and Rolf H. Möhring. Treewidth of cocomparability graphs and a new order-theoretic parameter. *Order*, 11:47–60, 1994. Kluwer Academic Publishers.

[HT73] J. E. Hopcroft and R. E. Tarjan. Dividing a graph into triconnected components. *SIAM J. Comput.*, 2:135–158, 1973.

[KLL01] Ton Kloks, C. M. Lee, and Jim Liu. Feedback vertex sets and disjoint cycles in planar (di)graphs. Optimization Online, 2001.

[Klo96] Ton Kloks. Treewidth of circle graphs. *International Journal of Foundations of Computer Science*, 7(2):111–120, 1996.

[KM92] André Kézdy and Patrick McGuinness. Sequential and parallel algorithms to find a K_5 minor. In *Proceedings of the Third Annual ACM-SIAM Symposium on Discrete Algorithms (Orlando, FL, 1992)*, pages 345–356, 1992.

[KR91] Arkady Kanevsky and Vijaya Ramachandran. Improved algorithms for graph four-connectivity. *J. Comput. System Sci.*, 42(3):288–306, 1991.

[Lag96] Jens Lagergren. Efficient parallel algorithms for graphs of bounded tree-width. *J. Algorithms*, 20(1):20–44, 1996.

[LT80] Richard J. Lipton and Robert Endre Tarjan. Applications of a planar separator theorem. *SIAM J. Comput.*, 9(3):615–627, 1980.

[MR92] Gary L. Miller and Vijaya Ramachandran. A new graph triconnectivity algorithm and its parallelization. *Combinatorica*, 12(1):53–76, 1992.

[MT92] Jiří Matoušek and Robin Thomas. On the complexity of finding iso- and other morphisms for partial k-trees. *Discrete Math.*, 108(1-3):343–364, 1992. Topological, algebraical and combinatorial structures. Frolík's memorial volume.

[Ros74] Donald J. Rose. On simple characterizations of k-trees. *Discrete Math.*, 7:317–322, 1974.

[RS84] Neil Robertson and Paul D. Seymour. Graph minors. III. Planar tree-width. *Journal of Combinatorial Theory Series B*, 36:49–64, 1984.

[RS85] Neil Robertson and Paul D. Seymour. Graph minors — a survey. In I. Anderson, editor, *Surveys in Combinatorics*, pages 153–171. Cambridge Univ. Press, 1985.

[RS86] Neil Robertson and Paul D. Seymour. Graph minors. II. Algorithmic aspects of tree-width. *J. Algorithms*, 7(3):309–322, 1986.

[RS91] Neil Robertson and Paul D. Seymour. Graph minors. X. Obstructions to tree-decomposition. *Journal of Combinatorial Theory Series B*, 52:153–190, 1991.

[RS93] Neil Robertson and Paul Seymour. Excluding a graph with one crossing. In *Proceedings of the AMS-IMS-SIAM Joint Summer Research Conference on Graph Minors*, Graph Structure Theory, pages 669–675, 1993.

[San96] Daniel P. Sanders. On linear recognition of tree-width at most four. *SIAM J. Discrete Math.*, 9(1):101–117, 1996.

[SSR94] Ravi Sundaram, Karan S. Singh, and Pandu C. Rangan. Treewidth of circular-arc graphs. *SIAM J. Discrete Math.*, 7(4):647–655, 1994.

[ST94] Paul D. Seymour and Robin Thomas. Call routing and the ratcatcher. *Combinatorica*, 14(2):217–241, 1994.

[Tar72] Robert Tarjan. Depth-first search and linear graph algorithms. *SIAM J. Comput.*, 1(2):146–160, 1972.

[Tho01] Andrew Thomason. The extremal function for complete minors. *Journal of Combinatorial Theory, Series B*, 81(2):318–338, March 2001.

[TP93] Jan Arne Telle and Andrzej Proskurowski. Practical algorithms on partial k-trees with an application to domination-like problems. In *Proceedings of Third Workshop on Algorithms and Data Structures*, volume 709 of *LNCS*, pages 610–621, 1993.

[TP97] Jan Arne Telle and Andrzej Proskurowski. Algorithms for vertex partitioning problems on partial k-trees. *SIAM Journal on Discrete Mathematics*, 10(4):529–550, 1997.

[vL90] Jan van Leeuwen. Graph algorithms. In *Handbook of Theoretical Computer Science*, volume A, chapter 10, pages 525–631. Elsevier, 1990.

[Wag37] Klaus Wagner. Über eine Eigenschaft der ebenen Komplexe. *Math. Ann.*, 114:570–590, 1937.

[Wil84] S. G. Williamson. Depth-first search and Kuratowski subgraphs. *J. Assoc. Comput. Mach.*, 31(4):681–693, 1984.

[Yan78] Mihalis Yannakakis. Node- and edge-deletion NP-complete problems. In *Proceedings of the Tenth Annual ACM Symposium on Theory of Computing*, pages 253–264, 1978.