Alcom-FT Technical Report Series
ALCOMFT-TR-03-70

Fast approximation schemes for $K_{3,3}$-minor-free or K_5-minor-free graphs*

Mohammadtaghi Hajiaghayi[1], Naomi Nishimura[1],
Prabhakar Ragde[1], and Dimitrios M. Thilikos[2]

[1] Department of Computer Science, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada.
[2] 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. As the class of graphs of bounded treewidth is of limited size, we need to solve NP-hard problems for wider classes of graphs than this class. Eppstein introduced a new concept which can be considered as a generalization of bounded treewidth. A graph G has *locally bounded treewidth* if for each vertex v of G, the treewidth of the subgraph of G induced on all vertices of distance at most r from v is only a function of r, called *local treewidth*. So far the only graphs determined to have small local treewidth are planar graphs. In this paper, we prove that any graph excluding one of K_5 or $K_{3,3}$ as a minor has local treewidth bounded by $3k+4$. As a result, we can design *practical* polynomial-time approximation schemes for both minimization and maximization problems on these classes of non-planar graphs.

1 Introduction

Many results design PTASs restricted to certain special graphs, especially graphs of bounded or locally bounded treewidth. Lipton and Tarjan [LT80] were the first who proved various NP-optimization problems have PTASs over planar graphs. Unfortunately, Chiba et al. have shown to reach a performance ratio half of the optimal in Lipton and Tarjan's work, the graph must have at least $2^{2^{400}}$ vertices and so their approach is known to be impractical [CNS82]. Using a different approach, Baker [Bak94] gave practical PTASs for the problems considered by Lipton and Tarjan. Alon *et al.* [AST90] generalized Lipton and Tarjan's ideas to graphs without a fixed minor. Like Lipton and Tarjan's PTASs, their PTASs were impractical too.

By partitioning a graph into three forests, Chen and He [CH95,Che95] obtained efficient approximation algorithms of ratio 3 for many NP-hard hereditary maximization problems on planar, $K_{3,3}$-minor-free graphs and K_5-minor-free graphs. After that, Chen [Che98], using Baker's approach, found approximation algorithms of ratio $1 + 1/\log n$ for NP-hard hereditary maximization problems on $K_{3,3}$-minor-free graphs and K_5-minor-free graphs. His approach was a non-trivial generalization of Baker's approach for these types of graphs. Baker's technique decomposes a planar embedding by successively deleting outer faces; vertices are given level numbers corresponding to the iterations in which they are deleted. Removing only those vertices with level number congruent to i mod k results in a k-outerplanar graph; there are k choices of i, and every vertex is in exactly $k-1$ of the resulting k-outerplanar graphs. Many NP-complete problems (such as maximum independent set, minimum dominating set, and minimum vertex cover) can be solved exactly on k-outerplanar graphs by dynamic programming. Suppose s_i, $1 \leq i \leq k$, is the optimal solution for the ith k-outerplanar graph. Baker [Bak94] shows that by taking the best among $s_1, \cdots, s_k$ as a (nearly

* This research was done during the visit of the 4th author to the Department of Computer Science of the University of Waterloo on January 2001. The research of the first three authors was supported by the Natural Sciences and Engineering Research Council of Canada and Communication and Information Technology Ontario and the research of the 4th 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). Emails: {`mhajiagh`,`plragde`,`nishi`}`@uwaterloo.ca`, and `sedthilk@lsi.upc.es`

optimal) solution for the original graph, we have a solution within a factor of $(1 + 1/k)$ of the optimal. Chen's approach differs from Baker's mainly in construction of layers. Because of his special construction of layers, his approach only applies to inherent maximization problems such as the maximum independent set problem.

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 treewidth. The decomposition by deleting every kth face is replaced by deleting every kth level of a breadth-first tree of G, provided that the treewidth of the resulting graphs is a function of k. In Section 5, we consider this approach in more detail, when we solve many NP-optimization problems on $K_{3,3}$-minor-free graphs and K_5-minor-free graphs in linear and quadratic time.

Another important related problem is determining, when given a graph G and an integer k, whether the treewidth of G is at most k. This problem is NP-complete [ACP87] even for graphs of maximum degree at most nine, bipartite graphs and cocomparability graphs [BT97]. This problem has been solved for several classes of graphs such as chordal graphs, permutation graphs [BKK95], circular arc graphs [SSR94], circle graphs [Klo93] and distance hereditary graphs [BDK00]. Bodlaender et al. [BGHK95] gave an approximation algorithm with performance ratio $O(\log n)$ for this problem on general graphs. Solving the problem for the case in which the parameter k is fixed is also interesting. The first polynomial-time algorithm for this problem was presented by Arnborg, Corneil and Proskurowski [ACP87]. The running time of this algorithm is $O(n^{k+2})$. Using fundamental results on graph minors, Robertson and Seymour gave a non-constructive proof of the existence of a decision algorithm with running time $O(n^2)$. An algorithm with a faster running time was developed by Lagergren [Lag96] and Bodlaender and Kloks [BK96]. Finally, Bodlaender [Bod96] found a constructive linear-time algorithm for the problem. All these algorithms have a hidden constant factor that is at least exponential in k, and hence they are impractical in general. In the cases $k = 2, 3, 4$, practical linear-time algorithms exist [AP86,MT92,San96]. Alber et al. [ABFN00] presented a constructive efficient algorithm for finding a tree decomposition of an r-outerplanar graph G in time $O(r|V(G)|)$ (the treewidth of an r-outerplanar graph is bounded by $3r - 1$ [ABFN00]). In this paper, we generalize this result demonstrating linear local treewidth of planar graphs to $K_{3,3}$-minor-free graphs and K_5-minor-free graphs. In fact, we present a more general theorem: we prove that if a graph H is a single-crossing graph (can be drawn on the plane with at most one crossing) then the local treewidth of any H-minor-free graph is bounded above by $3k + c_H$ where c_H is a constant depending only on H. In addition, we show how we can construct a tree decompostion of width $3r + 4$ for a $K_{3,3}$-minor-free or K_5-minor-free graph of diameter at most r. Using this algorithm, we will design an efficient linear-time algorithm for constructing tree decompositions of $K_{3,3}$-minor-free or K_5-minor-free bounded diameter graphs (see Section 3).

This paper is organized as follows. First, we introduce the terminology used throughout the paper, and formally define tree decompositions, treewidth and local treewidth in Section 2. Then, we present our main results on local treewidth and construction of tree decompositions of $K_{3,3}$-minor-free or K_5-minor-free graphs (Section 3). Next, we show how our results can be applied to find algorithms and practical PTASs for these graphs (Sections 4 and 5). Finally we conclude with a list of open problems and potential extensions for future work in Section 6.

2 Preliminaries

We assume the reader is familiar with general concepts of graph theory such as directed graphs, trees and planar graphs. The reader is referred to standard references for appropriate background [BM76].

Our graph terminology is as follows. 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 denote an edge e in a graph G between u and v by $\{u, v\}$. Here, vertices u and v are called the *end-vertices* of e. We define n to be the number of vertices of a graph when it is clear from context. 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. The *union* of two disjoint graphs G_1 and G_2, $G_1 \cup G_2$, is a graph G such that $V(G) = V(G_1) \cup V(G_2)$ and $E(G) = E(G_1) \cup E(G_2)$. An *n-clique* *($K_n$)* is a graph G with n vertices in which every pair of vertices is connected by an edge. A graph G is represented by $K_{n,m}$ if its vertices can be partitioned into sets V_1 and V_2 such that $|V_1| = n$, $|V_2| = m$ and edge $\{u, v\} \in E(G)$ if and only if $u \in V_1$ and $v \in V_2$ or vice versa.

For generalizations of algorithms on undirected graphs to directed graphs, we consider underlying graphs of directed graphs. An *underlying graph* of a directed graph $H = (V, E)$ is an undirected graph $G = (V, E)$ in which $V(G) = V(H)$ and $\{u, v\} \in E(G)$ if and only if $(u, v) \in E(H)$ or $(v, u) \in E(H)$.

A graph $G' = (V', E')$ is a *subgraph of* G if $V' \subseteq V$ and $E' \subseteq E$. A graph $G' = (V', E')$ is an *induced subgraph of* G, denoted by $G[V']$, if $V' \subseteq V$ and E' contains all edges of E which have both end vertices in V'.

One way of describing classes of graphs is by using *minors*, introduced below.

Definition 1. Contracting *an edge $e = \{u, v\}$ is the operation of replacing both u and v by a single vertex w whose neighbors are all vertices that were neighbors of u or v, except u and v themselves. A graph G is a* minor *of a graph H if H can be obtained from a subgraph of G by contracting edges. 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 minor-closed graph class $\mathcal{C}$ is H*-minor-free *if $H \notin \mathcal{C}$. The* minor containment *problem determines whether a graph is a minor of another graph.*

For example, a planar graph is a graph excluding both $K_{3,3}$ and K_5 as minors.

Baker [Bak94] introduced a property of planar graphs useful for designing approximation algorithms, namely a decomposition into *outerplanar graphs* (see Definition 2).

Definition 2. *[Bak94] Suppose graph G is embedded on the plane without any crossing. A vertex in the embedding is called a* level 1 *vertex if it is on the outer face. If an embedding obtained by removing all vertices in levels 1 to i is denoted by G^i, then the vertices on the outer face of G^i are the* level $i+1$ *vertices. A crossing-free embedding of a graph G is* r-outerplanar *if it has no vertices of level greater than r. A graph G is called r-outerplanar if it admits an r-outerplanar embedding. The smallest number such that G is r-outerplanar is called the* outerplanarity *number. The terms* outerplanar *and* 1-outerplanar *are equivalent.*

In Section 5, we design *approximation algorithms* for several *NP-optimization problems*. Definition 3 presents exact descriptions of these terms.

Definition 3. *[GJ79] An* NP-optimization problem *is a tuple $(\mathcal{I}, S, f, opt)$ such that:*

1. *$\mathcal{I}$ is the set of input instances. We assume that $\mathcal{I}$ can be recognized in polynomial time;*

2. *$S(x)$ is a set of feasible solutions associated to each input instance $x \in \mathcal{I}$. We assume that each element in $S(x)$ has size polynomially bounded in the size of x;*
3. *f is an objective function which maps to real numbers each pair (x, y) with $x \in \mathcal{I}$ and $y \in S(x)$. We assume that this function is computable in polynomial time; and*
4. *opt is a goal which belongs to set $\{min, max\}$.*

Given an $x \in \mathcal{I}$, we want to find a $y \in S(x)$ such that $f(x, y) = opt\{f(x, z)|z \in S(x)\}$. Let $x \in \mathcal{I}$ and $\epsilon > 0$. A solution $y \in S(x)$ for x is ϵ-close *if*

$$(1 - \epsilon)opt(x) \leq f(x, y) \leq (1 + \epsilon)opt(x).$$

A *polynomial time approximation scheme (PTAS)* for $(\mathcal{I}, S, f, opt)$ is a *uniform family* $(A_\epsilon)_{\epsilon \geq 0}$ of approximation algorithms, where A_ϵ is a polynomial time algorithm that, given an $x \in \mathcal{I}$, computes an ϵ-close solution for x in polynomial time. In the above definition, uniformity means that there is an algorithm that, given ϵ, computes A_ϵ.

Among NP-optimization problems, we mainly focus on those problems which are also *hereditary* (see Definition 4). In fact, Yannakakis [Yan78] has shown that many natural hereditary problems are NP-complete even when the graphs under consideration are planar graphs.

Definition 4. *[Yan78] Property π on graphs is called* hereditary *if, whenever π holds for G, π holds for all induced subgraphs of G. For hereditary property π, the* maximum induced subgraph problem *associated with π (MISP(π)) is the problem of finding a maximum subset U of vertices of a graph G such that $G[U]$ has property π. In the* weighted *case (WMISP(π)), each vertex in the given graph $G = (V, E)$ has a nonnegative weight and the problem is to find a maximum weight subset U of V such that $G[U]$ has property π, where the weight of U is the total weight of the vertices in it. We call a problem P associated with the hereditary property π a* hereditary maximization problem.

Examples of hereditary maximization problems are those in which we search for an induced subgraph of maximum size that is *chordal*, *acyclic*, *without cycles of specified length*, *without edges*, *bounded degree with maximum degree $r \geq 1$*, *bipartite* or *forms a clique* [Yan78].

For exact definitions of various NP-hard problems in this paper, the reader is referred to Garey and Johnson's book on computers and intractability [GJ79].

The notion of treewidth was introduced by Robertson and Seymour [RS86] and plays an important role in their fundamental work on graph minors. To define this notion, first we consider the representation of a graph as a tree, which is the basis of our algorithms.

Definition 5. *[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.*

To distinguish between vertices of the original graph G and vertices of T in $TD(G)$, we call vertices of T *nodes* and their corresponding χ_i's *bags*. 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 ($tw(G)$) is the minimum width over all possible tree decompositions of G.

A graph G is called a *k-tree* [Ros74] if 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 again a k-tree. A graph G is a *partial k-tree* if it is a subgraph of a k-tree.

Lemma 1. *(van Leeuwen [Lee90]) G is a partial k-tree if and only if G has treewidth at most k.*

Many NP-complete problems have linear-time or polynomial-time algorithms when they are restricted to graphs of bounded treewidth. There are a few techniques for obtaining such algorithms. The main technique is called *computing tables of characterizations of partial solutions.* This technique is a dynamic programming approach, first introduced by Arnborg and Proskurowski [AP89]. This technique also appeared in a paper written by Bern et al. [BLW87]. Bodlaender [Bod97] described a better presentation of this technique. Other approaches applicable for solving problems on graphs of bounded treewidth are *graph reduction* [ACPS93,BdF96] and *describing the problems in logic* [ALS88,Cou90].

The class of graphs of bounded treewidth is of limited size; we would like to solve NP-complete problems for wider classes of graphs. Baker [Bak94] developed several approximation algorithms to solve NP-complete problems for planar graphs. To extend these algorithms to other graph families, Eppstein [Epp00] introduced the notion of bounded local treewidth, defined formally below, which is a generalization of the notion of treewidth. Intuitively, a graph has bounded local treewidth (or locally bounded treewidth) if the treewidth of an r-neighborhood of each vertex $v \in V(G)$ is a function of r, $r \in \mathbb{N}$, and not $|V(G)|$.

Definition 6. *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 $r \in \mathbb{N}$, $\mathrm{ltw}^G(r) \leq f(r)$.*

A graph is called an *apex graph* if deleting one vertex produces a planar graph. We call a graph class $\mathcal{E}$ a *minor-closed* class if the minor of any graph in $\mathcal{E}$ is also a member of $\mathcal{E}$. A minor-closed graph class $\mathcal{E}$ is *H-minor free* if $H \notin \mathcal{E}$. Eppstein [Epp00] showed that a minor-closed graph class $\mathcal{E}$ has bounded local treewidth if and only if $\mathcal{E}$ is H-minor free for some apex graph H.

3 Local treewidth of clique-sum graphs

In this section, first we show H-minor-free graphs, where H is a single-crossing graph, have linear local treewidth. Then we introduce the concept of layers for these graphs and present a practical algorithm for construction of a tree decomposition of any subgraph induced on a constant number of consecutive layers.

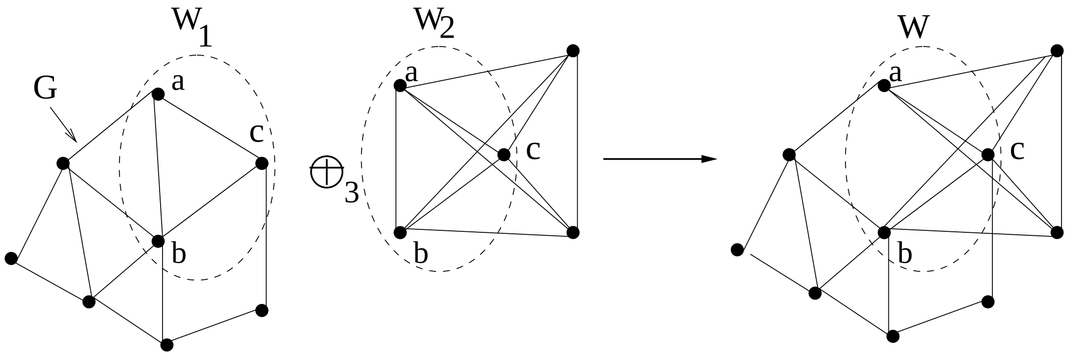

Fig. 1. Graph summation operation: identifying sets W_1 and W_2 and deleting edge $\{a, b\}$

The graph summation operation plays an important role in our results. Suppose G_1 and G_2 are graphs with disjoint vertex-sets and $k \geq 0$ is an integer. For $i = 1, 2$, let $W_i \subseteq V(G_i)$ form a clique of size k and let

G_i' $(i = 1, 2)$ be obtained from G_i by deleting some (possibly no) edges from $G_i[W_i]$ with both endpoints in W_i. Consider a 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 simply by $G = G_1 \oplus G_2$, to be the graph obtained from the union of G_1' and G_2' by identifying w with $h(w)$ for all $w \in W_1$. 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 reader is referred to Figure 1 to see an example in which a 3-sum of a graph G and K_5 is depicted.

We use the following three simple lemmas to obtain our main result.

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

Proof. It is enough to observe that for any $v \in G'$ and $k \ge 0$, $N^k_{G'}(v) \subseteq N^k_G(v)$. Thus the removal of vertices of $N^k_G(v) \setminus N^k_{G'}(v)$ from bags of a tree decomposition of $N^k_G(v)$ results in a tree decomposition of $N^k_{G'}(v)$ with the same local treewidth or less. □

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

Proof. Let W be the set of vertices of G and H identified during the $\oplus$ operation. Since W is a clique in G, in every tree decomposition of G, there exists a node α such that W is a subset of χ_α [BM93]. Similarly, it is true for W and a node α' of each tree decomposition of H. Hence, we can construct a tree decomposition of G and a tree decomposition of H and add an edge between α and α'. □

Lemma 4. *For any graph G, any clique R of G, any $v \in R$, and any $k \ge 0$,*
$\mathrm{tw}(G[N^k_G(R)]) \le \mathrm{tw}(G[N^{k+1}_G(v)])$.

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

Lemma 5 shows how the local treewidth changes when we apply a graph summation operation.

Lemma 5. *If G_1 and G_2 are graphs where $\mathrm{ltw}^{G_i}(r) \le f(r)$, $f(r) \ge 0$ for all $r \in \mathbb{N}$, and $G = G_1 \oplus_k G_2$, then $\mathrm{ltw}^{G}(r) \le f(r)$.*

Proof. To show $\mathrm{ltw}^G(r) \le f(r)$, we prove for any $v \in V(G)$ and for all $r \ge 0$, $\mathrm{tw}(G[N^r_G(v)]) \le f(r)$. Since $f(r) \ge 0$, the claim is clear for $r = 0$. Thus we assume $r > 0$ in the rest of the proof. Let W be the join set of $G_1 \oplus_k G_2$. Without loss of generality, we can assume v is from G_1. If $N^r_G(v)$ contains only vertices originally from G_1, the result follows from our initial assumption about G_1, i.e. $\mathrm{ltw}^{G_1}(r) \le f(r)$.

We now assume $N^r_G(v)$ contains vertices from G_2. If $v \in W$, then $N^r_G(v) \subseteq N^r_{G_1}(v) \cup N^r_{G_2}(v)$. In addition, since $r \ge 1$ 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, $G[N^r_G(v)]$ 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

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

We now consider the case in which $v \notin W$ and there exists a vertex $u \in N^r_G(v) - W$ which is from G_2. Since W is the only common set of vertices of 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. Therefore, $W \cap N^{r-1}_{G_1}(v) \ne \emptyset$. Let $w \in W \cap N^{r-1}_{G_1}(v)$ be the vertex with minimum distance p from v where $1 \le p \le r - 1$. We observe that each vertex u with the aforementioned property is at distance at most $r - p$ from at least one vertex of W. Thus $N^r_G(v) \subseteq N^r_{G_1}(v) \cup N^{r-p}_{G_2}(W)$. As one vertex of W is at distance $p \le r - 1$ from v and vertices of

W form a clique in G_1, each vertex of W is at distance at most r from v in G_1, *i.e.* $W \subseteq N^r_{G_1}(v)$. Also, $W \subseteq N^{r-p}_{G_2}(W)$. Thus we can obtain $G_1[N^r_{G_1}(v)] \oplus G_2[N^{r-p}_{G_2}(W)]$ over the join set W. As mentioned above $N^r_G(v) \subseteq N^r_{G_1}(v) \cup N^{r-p}_{G_2}(W)$, and thus $G[N^r_G(v)]$ is a subgraph of $G_1[N^r_{G_1}(v)] \oplus G_2[N^{r-p}_{G_2}(W)]$. Hence, by Lemma 3,

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

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

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

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

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

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

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

Finally using 3 and 4, we conclude that $tw(G[N^r_G(v)]) \leq f(r)$. □

It is known that any H-minor-free graph G, for single-crossing graph H (see the introduction), can be obtained from planar graphs and graphs of treewidth at most c_H by means of a series of k-sums, $0 \leq k \leq 3$, where c_H is a constant dependent only on the single-crossing graph H [RS93]. Because of this property, we call H-minor-free graphs *clique-sum graphs* when H is a single crossing graph. A series of k-sums (not necessarily unique) which generate a clique-sum graph G are called *a set of clique-sum operations* of G. Lemma 6 follows from this definition of clique-sum graphs:

Lemma 6. *For any clique-sum graph G which excludes a single crossing graph H as a minor, any minor G' of G is also a clique-sum graph which excludes the same graph H as a minor.*

Proof. The proof follows from the fact that if G' is a minor of G and G is H-minor-free, then G' is H-minor-free too. □

Theorem 1 demonstrates our main result on the local treewidth of clique-sum graphs.

Theorem 1. *For any clique-sum graph G excluding a single-crossing graph H as a minor and for all $r \geq 0$, $\mathrm{ltw}^G(r) \leq 3r + c_H$.*

Proof. By the definition of clique-sum graphs, we can assume $G = G_1 \oplus G_2 \oplus \cdots \oplus G_m$ where each G_i, $1 \leq i \leq m$, is either a planar graph or a graph of treewidth at most c_H. We use induction on m, the number of G_i's. For $m = 1$, we wish to show that G_1 is either a planar graph whose local treewidth is $3r - 1$ or a graph of treewidth at most c_H. In the former case $\mathrm{ltw}^G(r) = \mathrm{ltw}^{G_1}(r) = 3r - 1 \leq 3r + c_H$, $c_H \geq 0$, and in the latter case $\mathrm{ltw}^G(r) = \mathrm{ltw}^{G_1}(r) = c_H \leq 3r + c_H$, $r \geq 0$. Thus the basis of induction is true for both cases. We assume the induction hypothesis is true for $m = h$, and we prove the hypothesis for $m = h + 1$. Let $G' = G_1 \oplus G_2 \oplus \cdots \oplus G_h$ and $G'' = G_{h+1}$. Thus $G = G' \oplus G''$. By the induction hypothesis, $ltw^{G'}(r) \leq 3r + c_H$ and $ltw^{G''}(r) \leq 3r + c_H$. The proof, for $m = h + 1$, follows from this fact and Lemma 5. □

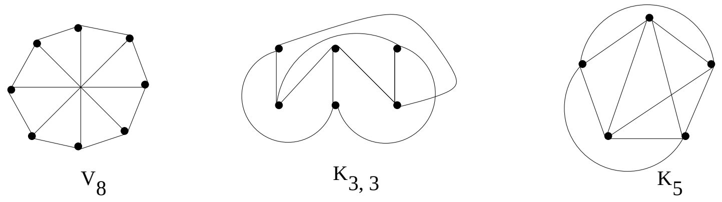

Fig. 2. Graph V_8 and single-crossing embeddings of $K_{3,3}$ and K_5

Using the fact that K_5 and $K_{3,3}$ are single-crossing graphs (Figure 2), we observe that K_5-minor-free graphs and $K_{3,3}$-minor-free graphs are clique-sum graphs. Wagner [Wag37] gave a better characterization for these graphs. He proved that a graph has no minor isomorphic to $K_{3,3}$ if and only if it can be obtained from planar graphs and K_5 by 0-,1-, and 2-sums. He also showed that a graph has no minor isomorphic to K_5 if and only if it can be obtained from planar graphs and V_8, shown in Figure 2, 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 1 is four, and we have:

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

As mentioned before, 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 v of $V(G)$. We denote *consecutive layers from i to j* by $L[i, j] = \bigcup_{i \leq k \leq j} L_k$.

Theorem 2. *For any clique-sum graph G, 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 6, we obtain another clique-sum graph G'. As all vertices at distance d, $i \leq d \leq j$, from v in G are at distance d', $1 \leq d' \leq j - i + 1$, from v' in G' and all vertices at distance more than j from 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 $tw(G[L[i, j]]) = tw(G'[L[1, j - i + 1]])$. Since all vertices of $L[1, j - i + 1]$ in G' are in the $j - i + 1$-neighborhood of v', $tw(G'[L[1, j - i + 1]]) \leq tw(G'[N_{G'}^{j-i+1}(v')])$. By the definition of local treewidth, $tw(G'[N_{G'}^{j-i+1}(v')]) \leq ltw^{G'}(j - i + 1)$. Finally by Theorem 1, we have $ltw^{G'}(j - i + 1) \leq 3(j - i + 1) + c_H$. Using these facts, $tw(G[L[i, j]]) \leq 3(j - i + 1) + c_H$, as desired. □

Theorem 2 gives an upper bound on the treewidth of consecutive layers from i to j, but it does not provide a constructive algorithm to obtain a tree decomposition of this width. Using Bodlaender's algorithm [Bod96], we can construct a tree decomposition of this width in linear time, but the hidden constant factor is huge, and the algorithm is impractical. Below we give a practical algorithm which constructs a tree decomposition of width $3(j - i + 1) + c_H$ for consecutive layers from i to j in $K_{3,3}$-minor-free or K_5-minor-free graphs. First we determine a set of clique-sum operations of $K_{3,3}$-minor-free or K_5-minor-free graphs.

Theorem 3. *[KM92] A set of clique-sum operations of a K_5-minor-free graph $G = G_1 \oplus G_2 \oplus \cdots \oplus G_m$ can be found in $O(n^2)$ time such that $\sum_{i=1}^{m} |V(G_i)| = O(|V(G)|)$.* □

Asano [Asa85] presented an $O(n)$ time algorithm for finding a set of clique-sum operations of a graph with no subgraph homomorphic to $K_{3,3}$. As for a cubic graph H (degree of each vertex is at most three), H is a minor of G if and only if G contains a subgraph homeomorphic to H, we have:

Theorem 4. *A set of clique-sum operations of a $K_{3,3}$-minor-free graph $G = G_1 \oplus G_2 \oplus \cdots \oplus G_m$ can be found in $O(n)$ time such that $\sum_{i=1}^{m} |V(G_i)| = O(|V(G)|)$.* $\square$

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

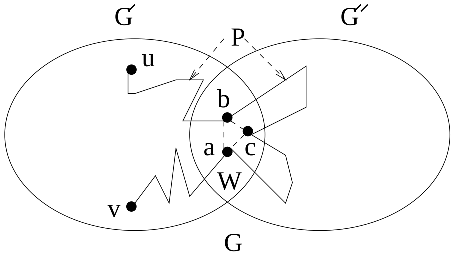

Fig. 3. The replacement of the part of path P between a and b by edge $\{a, b\}$

Lemma 7. *Let $G = G_1 \oplus G_2 \oplus \cdots \oplus G_m$ be a clique-sum graph. 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, $1 \leq i \leq m$, 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 $m \leq h$, and we prove the hypothesis for $m = h+1$. We suppose $G = G' \oplus G''$ where $G' = G_1 \oplus G_2 \oplus \cdots \oplus G_h$ and $G'' = G_{h+1}$. We let W be the join set of $G' \oplus G''$.

First we consider the case in which vertex v defined in the statement of Lemma is in W. We show that v has a path of length at most r in G' (G'') to each vertex u in G' (G''). Without loss of generality, we take a vertex $u \in V(G')$. If a part of a shortest path P in G from v to u goes through vertices in $V(G'') - V(G')$, this part goes through vertices of W. Let a be the first vertex in W and b be the last vertex in W on path P from v to u (see Figure 3). We note that a and b can be the same vertex. Since vertices of W form a clique in G', edge $\{a, b\}$ is present in G'. We can replace the part of path P which goes through vertices in $V(G'')$ (and has length at least one) by edge $\{a, b\}$ and obtain a path from v to u in G' with length less than the length of P. Thus, there is a path of length at most r in G' from v to each vertex u in G'. Using the induction hypothesis for G' and G'', we obtain the result for G.

We now consider the case in which $v \in V(G) - W$. Without loss of generality, we assume $v \in V(G') - W$. A shortest path in G from v to a vertex u of G'' goes through a vertex in W whose distance is at least one from v. Hence each vertex of G'' is at distance at most $r - 1$ from a vertex in W. Since vertices of W form a clique in G'', each vertex of G'' is at distance at most r from each vertex w of W. We now show that v has a path of length at most r in G' to each vertex u in G'. If a part of a shortest path P in G from v to u goes through vertices in $V(G'') - V(G')$, this part (which has length at least one) can be replaced in G' by an edge between vertices of W without increasing the length of the path (see the proof of the previous case). Applying the induction hypothesis for G' and G'' obtains the desired result for G. $\square$

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

Theorem 5. *For $K_{3,3}$-minor-free (K_5-minor-free) graph G, we can construct a tree decomposition for $G[L[i, j]]$ of treewidth $3(j - i + 1) + c_H$ in $O((j - i + 1)^3 \cdot n)$ ($O((j - i + 1)^3 \cdot n + n^2)$) time.*

Proof. As in the proof of Theorem 2, we contract the connected subgraph $G[L[0, i-1]]$ to a vertex v' and obtain another clique-sum graph G' such that $G[L[i,j]] = G'[L[1, j-i+1]]$. By Lemma 6, graph $G'' = G'[L[0, j-i+1]]$ is a $K_{3,3}$-minor-free (K_5-minor-free) graph and by the definition of layers each vertex in G'' is at distance at most $j-i+1$ from v'. By Theorem 3 (4), we can determine a set of clique-sum operations of graph G'' in $O(n)$ ($O(n^2)$) time.

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, $1 \leq i \leq m$. If G_i is a K_5 (V_8), we can easily construct a tree decomposition of width four in constant time. 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 G has a rooted spanning tree T in which the longest path has length d, then a tree decomposition of G 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, $1 \leq i \leq m$, 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}, we can find the node α' of the tree decomposition whose bag contains W by brute force over all subsets of size at most three of bags. 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 in $O(|V(G_{h+1})|)$ and each bag has size at most $3(j-i+1)$ (and thus there are at most $27(j-i+1)^3$ 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 for gluing 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 time of the algorithm for $K_{3,3}$-minor-free graphs and that for K_5-minor-free 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 both graphs.

Finally, we prove that the width of the constructed tree decomposition of G is $3(j-i+1)+4$. 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 with treewidth $3(j-i+1)$ or a graph of treewidth at most 4. In both cases the basis of induction is true. We assume the induction hypothesis is true for $m = h$, and we prove the hypothesis for $m = h+1$. Let $G' = G_1 \oplus G_2 \oplus \cdots \oplus G_h$ and $G'' = G_{h+1}$. Thus $G = G' \oplus G''$. By the induction hypothesis, treewidth of both G' and G'' is at most $3(j-i+1)+4$. The proof, for $m = h+1$, follows from this fact and Lemma 3. □

In the rest of this paper, we show how the results of this section can be applied to find algorithms for clique-sum graphs, especially $K_{3,3}$-minor-free graphs and K_5-minor-free graphs.

4 Fixed parameter algorithms

As mentioned in the introduction, describing problems in logic is an applicable approach for solving graph-theoretic problems on graphs of bounded treewidth. A general framework for describing several graph-theoretic properties in logic is monadic second order logic, defined formally below.

Monadic second order logic (MSOL) is a language for expressing properties, especially graph-theoretic ones, in logic. It has variables for vertices, edges, sets of edges and sets of vertices. Its logical connectives are **and**, **or** and **not**. Quantifiers $\forall$ and $\exists$ can be applied to the variables. In addition, it has four binary relations: set membership ($s \in S$), adjacency test for vertices ($adj(u, v)$), incidency test for vertices and edges ($inc(v, e)$) and equality for variables ($a = b$).

Arnborg et al. [ALS88] extend MSOL to extended monadic second order logic (EMSOL) to have counting or summing evaluations over sets. This feature allows us to define minimization or maximization problems in monadic second order logic. They show that problems definable in EMSOL have linear-time or polynomial-time algorithms for graphs of bounded treewidth. The reader is referred to this paper for a list of some famous NP-optimization problems definable in EMSOL. Courcelle [Cou90] related MSOL to the notion of treewidth.

Theorem 6. *[Cou90] Let w be a fixed constant and ϕ be a property of graphs that is definable in monadic second order logic. Then ϕ can be decided in linear time on graphs of treewidth at most w.* □

Because of the large hidden constant in the complexity of linear-time algorithm of Theorem 6, this theorem does not provide practical algorithms. It still provides a simple way to determine if a property is linear-time decidable on partial k-trees.

Unfortunately, the analogue of Courcelle's theorem does not hold for NP-complete problems which have a monadic second order definition on graphs of locally bounded treewidth. Instead, there is a similar theorem for a somewhat limited class of NP-complete problems which can be defined using *first-order logic*, a restricted form of monadic second order logic, in which we do not have variables for sets and operations for set membership.

Theorem 7. *[FG99] Let C be a class of graphs of locally bounded treewidth and let ϕ be a property definable in first-order logic. Then for every $k \geq 1$, there is an algorithm which in time $O(n^{1+(1/k)})$ decides whether a given graph $G \in C$ has property ϕ, where n is the number of vertices of the graph G.* □

Hamiltonicity and 3-colorability are examples of properties which have monadic second order logic descriptions but not first-order logic descriptions [DF99]. Examples of first-order definable problems are the k-dominating set problem and the k-independent set problem for fixed k [DF99]. In the former problem, one searches for a set of k vertices of a graph such that each of the rest of the vertices has at least one neighbor in the set, and in the latter problem, one searches for a set of k vertices of a graph such that there exists no edge of the graph both of whose end-vertices are in the set.

Frick and Grohe also improved the running time of the algorithm mentioned in Theorem 7 for minor-closed families of graphs of locally bounded treewidth.

Theorem 8. *[FG99] Let C be a minor-closed class of graphs that have locally bounded treewidth and ϕ be a property definable in first-order logic. Then there is a linear-time algorithm deciding whether a given graph $G \in C$ has property ϕ.* □

The hidden constant in the complexity of the linear-time algorithm of Theorem 8, similar to that of Courcelle's theorem, is very large. Since by Lemma 6, clique-sum graphs are closed under taking of minors and by Theorem 1 they have locally bounded treewidth, we conclude:

Corollary 2. *Any first-order logic property ϕ can be decided in linear time over clique-sum graphs.* □

Frick and Grohe's linear-time algorithm has an immense hidden constant resulting from several factors including the cost of computing tree decompositions. Bodlaender's linear-time algorithm for constructing a tree decomposition, used in Frick and Grohe's algorithm, is only of theoretical interest due to very large constants involved in the algorithm. In contrast, our practical algorithm for construction of tree decompositions helps to improve the constants for $K_{3,3}$-minor-free graphs and K_5-minor-free graphs.

Using Theorem 8 and the fact that for fixed k, k-dominating set and k-independent set are first-order expressible properties on graphs, we have linear-time algorithms deciding whether a given graph G has these properties. Frick and Grohe [FG99] also generalized Theorem 8 to structures other than graphs. For example, consider the (k, d)-circuit satisfiability problem, for $d \geq 1$, in which one decides whether a given boolean circuit of depth at most d has a satisfying assignment such that at most k input gates are set to **true**. They proved this problem can be solved in linear time for circuits whose underlying graphs are in a minor-closed family of graphs of locally bounded treewidth. Another example is evaluating a (boolean) database query against a relational database expressed in the relational calculus. As relational calculus is contained in first-order logic, they showed that Boolean relational calculus queries can be evaluated in linear time for a database whose underlying graph is in a minor-closed family of graphs of locally bounded treewidth. Therefore, algorithms for k-dominating set, k-independent set, (k, d)-circuit satisfiability and evaluating a (boolean) database query against a relational database expressed in the relational calculus have better running times when the graphs or underlying graphs under consideration are $K_{3,3}$-minor-free graphs and K_5-minor-free graphs.

The hidden constant in the linear-time algorithm of Theorem 8 is still large. Alber et al. [ABFN00] designed a fixed parameter algorithm for finding a k-dominating set (dominating set of size k) in planar graphs. Here, we extend their result to $K_{3,3}$-minor-free or K_5-minor-free graphs. The constant involved in this algorithm (Theorem 11) is very much smaller than that in the linear-time algorithm, mentioned above, and it is practical for small values of k. First, we present two preliminary theorems.

Theorem 9. *[ABFN00] If a tree decomposition of width w of a graph is known, then a minimum dominating set can be determined in time $O(3^w \cdot n)$, where n is the number of vertices.* □

The proof of Theorem 9 mainly follows the general dynamic programming approach introduced in the introduction. Suppose we formed layers of vertices of a graph G (see Section 3). The next theorem relates the number of vertices of a dominating set to the number of layers.

Theorem 10. *If a graph $G = (V, E)$ has a k-dominating set, then the number of layers in layering of vertices of G from any vertex $v \in V(G)$ is at most $3k$.*

Proof. The idea of the proof follows from an idea of Alber et al. [ABFN00]. We note that each vertex in the dominating set can dominate vertices from the previous, the next, or its own layer only. Hence, each vertex in the dominating set can contribute to at most three layers and hence the number of layers is at most $3k$. □

Theorem 11. *For $K_{3,3}$-minor-free (K_5-minor-free) graphs, the problem of k-dominating set for fixed k can be solved in $O(3^{9k}n)$ ($O(3^{9k}n + n^2)$) time. Thus this problem is FPT on these graphs.*

Proof. If a graph G has a k-dominating set, the number of layers is at most $3k$ by Theorem 10. We can construct a tree decomposition of $G[L[0, 3k]] = G$ of width $3(3k+1)+4 = 9k+7$ in $O(n)$ ($O(n^2)$) time by Theorem 5. Finally, using this tree decomposition, we can solve the problem in $O(3^{9k} \cdot n)$ time by Theorem 9. Thus the overall running time is $O(3^{9k}n)$ ($O(3^{9k}n + n^2)$). □

Alber et al. [ABFN00] proved that if a tree decomposition of width w of a graph is known, then a solution to each of variants of dominating set such as independent dominating set, total dominating set, perfect dominating set, perfect independent dominating set and total perfect dominating set can be determined in at most $O(4^w \cdot n)$ time. In addition, since a solution to each of these problems still is a dominating set for the graph, a theorem similar to Theorem 10 holds for each of them, *i.e.* if a graph has a solution of size k to each of these problems, then the number of layers of the graph is at most $3k$. Using these two facts we can solve these problems on $K_{3,3}$-minor-free (K_5-minor-free) graphs in $O(4^{9k}n)$ ($O(4^{9k}n + n^2)$) time (the proof is the same as the proof of Theorem 11).

5 Approximation algorithms

In this section, using Baker's approach on planar graphs, we will derive several PTASs for graph-theoretic optimization problems on $K_{3,3}$-minor-free graphs and K_5-minor-free graphs. Most problems considered in this section are hereditary maximization problems. Yannakakis has shown that for many natural hereditary properties π (see Section 2 for the definition), MIPS(π) is NP-complete even when the graphs under consideration are planar graphs [Yan78]. This result provides motivation to find approximation algorithms for such problems. Here we show how our results in Section 3 can be applied to 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 $K_{3,3}$-minor-free graphs and K_5-minor-free graphs. In the rest of this section, parenthesized parts pertain to K_5-minor-free graphs when it is clear from context. Also by superscripts on equalities and inequalities, we mean the facts from which the equalities and inequalities are obtained.

Theorem 12. *Let G be a non-negative vertex-weighted $K_{3,3}$-minor-free (K_5-minor-free) graph and let $k \geq 1$ be an integer. The maximization problem WMISP(π) for a hereditary property π over G has a PTAS of ratio $1 + 1/k$ of the optimal with worst-case running time in $O(k|V| + kTime_\pi(3(k-1)+4, |V|))$ ($O(k|V|^2 + kTime_\pi(3(k-1)+4, |V|))$), where $Time_\pi(w, n)$ is the worst-case running time of WMISP(π) over an n-vertex partial w-tree whose tree decomposition is given. $Time_\pi(w, n)$ is nondecreasing as n increases.*

Proof. First we decompose graph G into several induced subgraphs, each of which having bounded treewidth, and mention some properties of these induced subgraphs. For $1 \leq i \leq k$ and $j \geq 0$, we define $L_{ij} = L[(j-1)k + i, jk + i - 2]$. Here we assume a layer is empty when its level number is not between zero and the total number of layers, e.g. consider $j = 0$. We note that there is no edge between L_{ij} and $L_{i(j+1)}$. Let $\mathcal{L}_i = \bigcup_{j \geq 0} L_{ij}$ and $G_i = G[\mathcal{L}_i]$. Here every vertex appears in exactly $k-1$ of the $\mathcal{L}_i$'s or G_i's (vertices in layer L_h only do not appear in $\mathcal{L}_i$ where i is congruent to $h+1$ mod k). We label this fact by **[Fact a]**.

Then, we construct a tree decomposition of width $3(k-1)+4$ for each G_i as follows. By Theorem 5, we can construct a tree decomposition of width $3(k-1)+4$ for $G[L_{ij}]$ in linear (quadratic) time. Since $G_i = \bigcup_{j \geq 0} G[L_{ij}]$, a tree decomposition of width $3(k-1)+4$ for G_i can be constructed by gluing tree decompositions of $G[L_{ij}]$'s together (adding edges to become one tree) in $O(|V|)$ ($O(|V|^2)$) time (note that $G[L_{ij}]$'s are disjoint).

Next, we solve the WMISP(π) on each G_i, $1 \leq i \leq k$. Since $|V(G_i)| \leq |V(G)|$, Opt_i, the maximum weighted solution of WMISP(π) over G_i, can be constructed in $Time_\pi(3(k-1)+4, |V(G)|)$.

Finally, we take Opt_m the solution with maximum weight among $Opt_1, Opt_2, \cdots, Opt_k$ as our solution for graph G, and show that it has a ratio $1 + 1/k$ of the optimal. Suppose Opt is the maximum weighted solution on graph G. We prove $\frac{weight(Opt)}{weight(Opt_m)} \leq \frac{k}{k-1}$. Because of the hereditary property of WMISP(π), we have:

$$weight(Opt \cap \mathcal{L}_i) \leq weight(Opt_i) \tag{5}$$

Using 5, we have:

$$k \cdot weight(Opt_m) \geq \sum_{i=1}^{k} weight(Opt_i) \geq^{(5)} \sum_{i=1}^{k} weight(Opt \cap \mathcal{L}_i) =^{[Fact\ a]} (k-1) \cdot weight(Opt).$$

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

Corollary 3. *For $K_{3,3}$-minor-free (K_5-minor-free) graphs, there exist a PTAS of ratio $1+1/k$ of the optimal with running time $O(k \cdot 4^{3k} \cdot n)$ ($O(k \cdot 4^{3k} \cdot n + k \cdot n^2)$), for maximum independent set.*

Proof. Using dynamic programming on a tree decomposition, this problem can be solved in $O(4^w \cdot n)$ time, over each n-vertex partial w-tree whose tree decomposition is given [AP89]. Thus $Time_\pi(w,n) = O(4^w \cdot n)$ and the result follows from Theorem 12. □

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 ideas of the proofs of Theorems 13 and 14 follow ideas of Grohe [Gro] for general graphs of locally bounded treewidth, which are in fact Baker's ideas for planar graphs. We note that the constants involved in Grohe's work are immense in contrast to those in our work.

Theorem 13. *For any integer $k \geq 1$, the minimum weighted vertex cover problem on $K_{3,3}$-minor-free (K_5-minor-free) graphs has a PTAS of ratio $1+1/k$ of the optimal with worst-case running time $O(k \cdot 8^{3k} n)$ ($O(k \cdot 8^{3k} n + k \cdot n^2)$).*

Proof. As in the proof of Theorem 12, we first decompose graph G into several induced subgraphs each has bounded treewidth. For $1 \leq i \leq k$ and $j \geq 0$, we define $L_{ij} = L[(j-1)k+i, jk+i]$ and $G_{ij} = G[L_{ij}]$. Here L_{ij} is slightly different from that in the proof of Theorem 12. The following facts are easy to observe:

[Fact b] Vertices of the layer L_{jk+i} appear in both $G[L_{ij}]$ and $G[L_{i(j+1)}]$ and for fixed i, each vertex appears in at most two L_{ij}'s.

[Fact c] For fixed i, each edge of G appears in at least one G_{ij}.

[Fact d] Every vertex appears in $k+1$ (successive) sets L_{ij}.

Now, by Theorem 5, we construct a tree decomposition of width $3(k+1)+4$ for $G[L_{ij}]$ in $O(|V(G[L_{ij}])|)$ ($O(|V(G[L_{ij}])|^2)$) time. For fixed i, since each vertex of G appears in at most two $G[L_{ij}]$'s (see [Fact b]), constructing tree decompositions of all $G[L_{ij}]$'s takes $O(|V(G)|)$ ($O(|V(G)|^2)$) time. Since $1 \leq i \leq k$ and k is a constant, the running time for constructing tree decompositions of all $G[L_{ij}]$'s is linear (quadratic).

Now, for fixed i, we wish to construct solution Opt_i for graph G over L_{ij}'s. To this end, we solve the minimum vertex cover problem for each G_{ij} to obtain a solution Opt_{ij}. Then we let

$$Opt_i = \cup_{j \geq 0} Opt_{ij} \tag{6}$$

First we note that for fixed i, by [Fact c] each edge of G appears in at least one G_{ij}, and thus has at least one end-vertex in Opt_i by 6. Hence Opt_i is a solution for the whole graph G.

Now we compute the running time to obtain each Opt_i. The minimum vertex cover problem can be solved in $O(8^w \cdot n)$ time over each n-vertex partial w-tree whose tree decomposition is given [ALS88]. Thus computing Opt_{ij} takes $O(8^{3k}|V(G[L_{ij}])|)$ time on graph G_{ij}. Thus for fixed i, by [Fact b], computing Opt_i takes $O(8^{3k}|V(G)|)$ time.

Finally, we take Opt_m the solution with minimum weight among $Opt_1, Opt_2, \cdots, Opt_k$ as our solution on graph G and show that it has a ratio $1 + 1/k$ of the optimal. Suppose Opt is the minimum weighted solution on graph G. We prove that $\frac{weight(Opt_m)}{weight(Opt)} \leq \frac{k+1}{k}$. Since $Opt \cap L_{ij}$ is a vertex cover for G_{ij} and Opt_{ij} is a minimum vertex cover for G_{ij}

$$weight(Opt \cap L_{ij}) \geq weight(Opt_{ij}) \tag{7}$$

Using 6 and 7, we have:

$$k \cdot weight(Opt_m) \leq \sum_{i=1}^{k} weight(Opt_i) \leq^{(6)} \sum_{i=1}^{k} \sum_{j \geq 0} weight(Opt_{ij})$$

$$\leq^{(7)} \sum_{i=1}^{k} \sum_{j \geq 0} weight(Opt \cap L_{ij}) =^{[Fact\ d]} (k+1) \cdot weight(Opt).$$

The last equality follows from the fact that each vertex of Opt appears in $k+1$ L_{ij}'s (see [Fact d]).

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

To find an approximation algorithm for the dominating set problem, we first introduce a generalized version of the dominating set problem (Definition 7) and show how we can solve this problem in linear time (Lemma 8). Then we use the algorithm for solving this problem to obtain a PTAS for the dominating set problem (Theorem 14).

Definition 7. *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 $(u, w) \in E(G)$.*

It is worth mentioning that if we set $I = V(G)$ in the GDS problem, then this problem is the same as the dominating set problem.

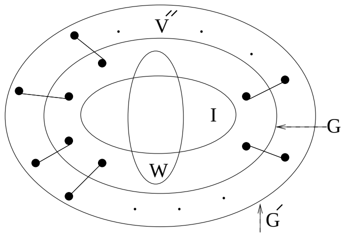

Fig. 4. Set V'' and graphs G and G' defined in the proof of Lemma 8.

Lemma 8. *The GDS problem for given graph G and set I can be solved in time $O(3^w \cdot |V(G)|)$ when a tree decomposition of width w for G is given.*

Proof. Alber et al. proved that if a tree decomposition of width w of a non-negative vertex-weighted graph G is known, then the dominating set (DS) problem can be solved in time $O(3^w \cdot |V(G)|)$ [ABFN00]. We reduce the GDS problem to the DS problem. To solve the GDS problem on graph G, we construct graph G' on which we solve the DS problem. First we let $G' = G$ and then for each vertex $v \in V(G) - I$, we add another vertex v' with weight zero connected to v. We call this set of vertices V'' (see Figure 4). Suppose W is a solution to the GDS problem. We can construct a solution W' to the DS problem in graph G' by adding all vertices of V'' to W. Here each vertex of I is dominated by a vertex in W and each vertex of $V(G') - I$ is dominated by a vertex in V''. Thus W' is a dominating set for G' with the same weight of W. On the other hand, by deleting all vertices in V'' from a solution W' to the DS problem on graph G', we obtain a solution W to the GDS problem on graph G with the same weight. In fact, vertices of W' in V'' can only dominate vertices in $V(G) - I$ and thus each vertex of I is dominated by a vertex of W' which is in G, i.e. it is dominated by a vertex of W. The treewidth of G' is the same as that of G, since for $w \in V''$ connected to a vertex $v \in V(G)$ we can simply add a node whose bag contains w and v to a node of $TD(G)$ whose bag contains v. As $|V(G')| \leq 2|V(G)|$ and treewidth of G' is w, the GDS problem can be solved in $O(3^w \cdot |V(G)|)$ time using the algorithm for the DS problem. □

Theorem 14. *For any integer $k \geq 1$, the minimum weighted dominating set problem on $K_{3,3}$-minor-free (K_5-minor-free) graphs has a PTAS of ratio $1+2/k$ of the optimal with worst-case running time $O(k \cdot 3^{3k} n)$ ($O(k \cdot 3^{3k} n + k \cdot n^2)$).*

Proof. We first decompose the vertex set of G into some sets such that the subgraph induced on each set has bounded treewidth. For $1 \leq i \leq k$ and $j \geq 0$, we define $L_{ij} = L^G[(j-1)k + i - 1, jk + i]$. The following facts are easy to observe:

[Fact g] For fixed i, L_{ij} and $L_{i(j+1)}$ intersect only in two consecutive layers and each vertex appears in at most two L_{ij}'s.

[Fact h] Each vertex appears in exactly $k+2$ (successive) sets L_{ij}.

Next, by Theorem 5, we construct a tree decomposition of width $3(k+2)+4$ for $G[L_{ij}]$ in $O(|V(G[L_{ij}])|)$ ($O(|V(G[L_{ij}])|^2)$) time. For fixed i, since each vertex of G appears in at most two $G[L_{ij}]$'s (see [Fact g]), constructing tree decompositions of all $G[L_{ij}]$'s takes linear (quadratic) time.

Now, for fixed i, we wish to construct solution Opt_i over all vertices in L_{ij}'s, as we did in the proofs of Theorems 12 and 13. To this end, we use the solutions to instances of the *GDS* problem as follows. The *interior* of each L_{ij} is defined as the set $I_{ij} = L^G[(j-1)k + i, jk + i - 1]$. For $1 \leq i \leq k$ and $j \geq 0$, let $Opt_{ij} \subseteq L_{ij}$ be a vertex set of minimum weight with the property that

[PD] for every $u \in I_{ij} - Opt_{ij}$ there is a $v \in Opt_{ij}$ such that $(u, v) \in E(G)$.

We let

$$Opt_i = \cup_{j \geq 0} Opt_{ij} \tag{8}$$

By Lemma 8, we can obtain Opt_{ij} for graph $G[L_{ij}]$ and set I_{ij} in $O(3^{3k}|V(G[L_{ij}])|)$ time. Using the fact that for fixed i, each vertex of G appears in at most two L_{ij}'s (see [Fact g]), computing each Opt_i takes $O(3^{3k} n)$ time. In addition, by Property [PD] of Opt_{ij}'s, Opt_i is a dominating set for G (for fixed i, each vertex appears one time in an interior set I_{ij} and thus dominated by at least one vertex).

Finally, we take Opt_m the solution of minimum weight among $Opt_1, Opt_2, \cdots, Opt_k$ as our solution on graph G, and show that it has at most a ratio $1 + 2/k$ of the optimal. Suppose Opt is the minimum weight dominating set over the whole graph G. We show that $\frac{weight(Opt_m)}{weight(Opt)} \leq \frac{k+2}{k} = 1+2/k$. We first show $Opt \cap L_{ij}$ has Property [PD] for L_{ij}. In fact, for each vertex $u \in I_{ij} \subset L_{ij}$, either $u \in Opt$ or $(u,v) \in E(G)$ where $v \in Opt$. In the latter case, v belongs to L_{ij}. Thus, in dominating set Opt, each vertex in I_{ij} is dominated by a vertex in L_{ij}. Now, since Opt_{ij} is a set of minimum weight with Property [PD], we have:

$$weight(Opt \cap L_{ij}) \geq weight(Opt_{ij}) \tag{9}$$

Using equations 8 and 9 and the fact that every vertex appears in exactly $k+2$ sets L_{ij} ([Fact h]), we have:

$$k \cdot weight(Opt_m) \leq \sum_{i=1}^{k} weight(Opt_i) \leq^{(8)} \sum_{i=1}^{k} \sum_{j \geq 0} weight(Opt_{ij})$$

$$\leq^{(9)} \sum_{i=1}^{k} \sum_{j \geq 0} weight(Opt \cap L_{ij}) =^{[Fact\ h]} (k+2) \cdot weight(Opt).$$

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 15. *For $K_{3,3}$-minor-free (K_5-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 3 and Theorems 13 and 14 is in $O(c^k n)$ ($O(c^k n + n^2)$) where k is a parameter and c is a constant. Now, by taking $k = \lceil c' \log n \rceil$, where c' is a constant, we obtain efficient polynomial-time approximation algorithms of ratio $1+1/(\log n)$ of the optimal (or $1+2/(\log n)$ for dominating set). Here $1/(\log n)$ $(2/(\log n))$ decreases as n increases. Thus the solutions converge toward optimal as n increases. □

We can generalize this approach for testing graph isomorphism of a fixed pattern H:

Theorem 16. *Subgraph isomorphism and induced subgraph isomorphism for a fixed pattern H in clique-sum graphs can be tested in $O(2^{O(|V(H)| \log |V(H)|)} n)$ time. Also the problems introduced in Eppstein's paper such as finding diameter if we know the graph has bounded diameter, h-clustering for constant h, finding girth if we know the graph has bounded girth can be tested in $O(n)$ for clique-sum graphs.*

Proof. The ideas mainly follow from Baker's approach which are introduced in Eppstein's paper [Epp99]. If a graph contains a subgraph with constant size, this subgraph must be included in a constant number of consecutive layers introduced in Theorems 12 and 14. By $O(n)$ times checking subgraph isomorphism, each consists of solving this problem for a fixed pattern H opposed a graph with bounded treewidth, we can solve the problem in linear time. The other variants also can be solved using this technique. The details are similar to Eppstain's paper [Epp99] and has been omitted from this paper. □

It is worth mentioning that for several hereditary maximization problems, the function $Time_\pi(w,n)$ introduced in Theorem 12 is in $O(c^{p(k)} \cdot q(n))$, where c is a constant and p and q are polynomials of low degree [Bod88,TP93]. Thus, by Theorem 12, there are PTASs of ratio $1+1/k$ of the optimal for them. In addition, approaches very similar to those used in Theorems 12, 13 and 14 can be applied to other problems such as minimum edge dominating set, maximum triangle matching, maximum H-matching and maximum tile salvage. The full presentation of these PTASs is beyond the scope of this paper and hence omitted. The reader is referred to papers due to Baker [Bak94] and Eppstein [Epp00] to obtain further details.

6 Conclusions and future work

In this paper, we introduced the class of clique-sum graphs, which contains $K_{3,3}$-minor-free graphs and K_5-minor-free graphs, and showed the graphs in the class have linear local treewidth. In addition, we presented a practical algorithm for constructing the tree decomposition of every subgraph induced on a constant number of consecutive layers in $K_{3,3}$-minor-free or K_5-minor-free graphs. Finally, we mentioned applications of our result to algorithms and PTASs for NP-hard problems on these graphs. Here, we present several open problems that can be considered as possible extensions of this paper.

Parallelizing exact algorithms and PTASs for problems given in Section 3 is a topic of interest. As the general dynamic programming approach can be parallelized easily (see Bodlaender's paper [Bod97]), in this extension, one needs to parallelize computing the tree decompositions of each constant number of consecutive layers introduced for clique-sum graphs in Section 3.

We suspect that Baker's approach can be applied to obtain practical PTASs for other problems. Some examples are as follows: graph s-partitioning, in which one searches for a partition of the vertex set of a graph into sets of size s and $n-s$ such that it minimizes the cutsize, maximum matching and variants of dominating sets introduced by Alber et al. [ABFN00], such as independent dominating set, total dominating set, perfect dominating set, perfect independent dominating set and total perfect dominating set. All these problems were solved for k-outerplanar graphs [BP92,DST96,ABFN00]. Using these results, one only needs to remove the layers in Baker's approach appropriately and obtain an approximation from the solution for each resulting k-outerplanar graph, as in our PTAS for minimum dominating set.

Alber et al. [ABFN00] showed that a planar graph with a dominating set of size k has treewidth $O(6\sqrt{34}\sqrt{k})$. Using this result, they concluded that finding a dominating set of size k in a planar graph can be solved in time $O(c^{\sqrt{k}}n)$, where $c = 4^{6\sqrt{34}}$ (the proof of the second result follows immediately from Theorem 9). We believe that in the proof of the first result one can replace the concept of outerplanarity by the concept of layers introduced for clique-sum graphs and obtain the similar result for these graphs. The reader is referred to the original paper [ABFN00] for further detail.

References

[ABFN00] Jochen Alber, Hans L. Bodlaender, Henning Fernau, and Rolf Niedermeier. Fixed parameter algorithms for planar dominating set and related problems. In *Algorithm theory—Scandinavian Workshop on Algorithm Theory 2000 (Bergen, 2000)*, pages 97–110. Springer, Berlin, 2000.

[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.

[ACPS93] Stefan Arnborg, Bruno Courcelle, Andrzej Proskurowski, and Detlef Seese. An algebraic theory of graph reduction. *J. Assoc. Comput. Mach.*, 40(5):1134–1164, 1993.

[ALS88] Stefan Arnborg, Jens Lagergren, and Detlef Seese. Problems easy for tree-decomposable graphs (extended abstract). In *Automata, languages and programming (Tampere, 1988)*, pages 38–51. Springer, Berlin, 1988.

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

[AP89] Stefan Arnborg and Andrzej Proskurowski. Linear time algorithms for NP-hard problems restricted to partial k-trees. *Discrete Appl. Math.*, 23(1):11–24, 1989.

[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 (Baltimore, MD, 1990)*, 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.

[BdF96] Hans L. Bodlaender and Babette de Fluiter. Reduction algorithms for constructing solutions in graphs with small treewidth. In *Computing and combinatorics (Hong Kong, 1996)*, pages 199–208. Springer, Berlin, 1996.

[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.

[BLW87] Marshall W. Bern, Eugene L. Lawler, and Alice L. Wong. Linear-time computation of optimal subgraphs of decomposable graphs. *J. Algorithms*, 8(2):216–235, 1987.

[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 Fifteenth International Colloquium on Automata, Languages and Programming (Tampere, 1988)*, pages 105–119. Springer, Berlin, 1988.

[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 *Mathematical foundations of computer science 1997 (Bratislava, 1997)*, pages 19–36. Springer, Berlin, 1997.

[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.

[CH95] Zhi-Zhong Chen and Xin He. NC algorithms for partitioning planar graphs into induced forests and approximating NP-hard problems. In *Graph-theoretic concepts in computer science (Aachen, 1995)*, pages 275–289. Springer, Berlin, 1995.

[Che95] Zhi-Zhong Chen. NC algorithms for partitioning sparse graphs into induced forests with an application. In *Sixth Annual International Symposium on Algorithms and Computation (Cairns, 1995)*, pages 428–437. Springer, Berlin, 1995.

[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.

[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.

[Cou90] Bruno Courcelle. Graph rewriting: an algebraic and logic approach. In *Handbook of theoretical computer science, Vol. B*, pages 193–242. Elsevier, Amsterdam, 1990.

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

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

[Epp99] David Eppstein. Subgraph isomorphism in planar graphs and related problems. *J. Graph Algorithms Appl.*, 3:no. 3, 27 pp. (electronic), 1999.

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

[FG99] Markus Frick and Martin Grohe. Deciding first-order properties of locally tree-decomposable graphs. In *Automata, languages and programming (Prague, 1999)*, pages 331–340. Springer, Berlin, 1999.

[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 in Combinatorica in 2001.

[Klo93] Ton Kloks. Treewidth of circle graphs. In *Algorithms and computation (Hong Kong, 1993)*, pages 108–117. Springer, Berlin, 1993.

[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.

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

[Lee90] Jan van Leeuwen. Graph algorithms. In *Handbook of theoretical computer science, Vol. A*, pages 525–631. Elsevier, Amsterdam, 1990.

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

[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.

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

[RS93] Neil Robertson and Paul Seymour. Excluding a graph with one crossing. In *Graph structure theory (Seattle, WA, 1991)*, pages 669–675. Amer. Math. Soc., Providence, RI, 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.

[TP93] Jan A. 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 (Montreal, 1993)*, pages 610–621. Springer, Berlin, 1993.

[Wag37] Kehrer Wagner. Über eine Eigenschaft der eben Komplexe. *Deutsche Math.*, 2:280–285, 1937.

[Yan78] Mihalis Yannakakis. Node- and edge-deletion NP-complete problems. In *Conference Record of the Tenth Annual ACM Symposium on Theory of Computing (San Diego, CA, 1978)*, pages 253–264. ACM press, New York, 1978.