Subexponential Algorithms for Partial Cover Problems

Fedor V. Fomin[*] Daniel Lokshtanov[*] Venkatesh Raman[†] Saket Saurabh[*]

July 4, 2009

Abstract

Partial Cover problems are optimization versions of fundamental and well studied problems like Vertex Cover and Dominating Set. Here one is interested in covering (or dominating) the maximum number of edges (or vertices) using a given number (k) of vertices, rather than covering all edges (or vertices). In general graphs, these problems are hard for parameterized complexity classes when parameterized by k. In planar and non trivial minor closed family of graphs, Amini et. al. [*FSTTCS 08*] gave a $2^{O(k)}n^c$ algorithm improving on the naive $k^{O(k)}n^c$ algorithm and ask whether there is a subexponential algorithm. In this paper, we answer the question affirmatively by giving a subexponential ($2^{O(\sqrt{k})}n^d$) algorithms for these problems using a simple degree sequence argument. Though these parameters are neither minor nor contraction closed, we find a way to apply bidimensionality theory after identifying and removing redundant vertices. This technique can be of independent interest for other problems on planar graphs.

1 Introduction and Motivation

A generic instance to a covering problem consists of a family of sets over an universe and the objective is to cover the universe with as few sets from the family as possible. Covering problems are basic problems not only in combinatorial optimization and algorithms but occur naturally in variety of applications. One of the prominent covering problem is the classical Set Cover problem. Other classical problems in the framework of covering include well known problems like Vertex Cover, Dominating Set, Facility Location, k-Median, k-Center problems, on which hundreds of papers have been written.

As the name suggests in partial cover problems we are not interested in covering the whole universe but partially, unlike covering problems where the goal is to cover the whole universe. This makes the partial cover problems natural generalizations to the well known covering problems. More precisely, in the partial covering problem, for a given integer $t \geq 0$, we want to cover at least t elements rather than covering all the elements. For an example, in Partial Vertex Cover (PVC), the goal is to cover at least t edges with the minimum number of vertices not all the edges while in Partial Dominating Set (PDS) the goal is to dominate at least t vertices of the input graph with the minimum number of vertices. Partial covering problems have been studied intensively not only because they generalize classical covering problems, but also because of many real life applications. They have been at forefront of research recently, see, for example [2, 4, 3, 5, 16, 18].

Partial cover problems have been investigated extensively and are well understood in the context of polynomial time approximation [2, 4, 3, 5, 16, 18]. However, these problems hold a lot of promise and remain hitherto unexplored in the light of parameterized complexity; with

---

[*] Department of Informatics, University of Bergen, Norway. {`fedor.fomin|daniello|saket.saurabh`}`@ii.uib.no`.

[†] The Institute of Mathematical Sciences, Chennai, India. `vraman@imsc.res.in`

exceptions that are few and far between [1, 4, 24, 25, 23, 27]. In this paper we study partial cover problems defined on graphs namely PARTIAL VERTEX COVER and PARTIAL r-DOMINATING SET from the view point of parameterized algorithms. PARTIAL VERTEX COVER is defined as follows.

> PARTIAL VERTEX COVER (PVC): Given a graph $G = (V, E)$ and positive integers k and t, check whether there exists a set of vertices $C \subseteq V$ such that $|C| \leq k$ and there are at least t edges incident on C.

The PARTIAL r-DOMINATING SET is a generalization of PARTIAL DOMINATING SET and is defined as follows.

> PARTIAL r-DOMINATING SET (P-r-DS): Given a graph $G = (V, E)$ and positive integers k and t, check whether there exists a set of vertices $D \subseteq V$ such that $|D| \leq k$ and there are at least t vertices such that their distance to at least one of the vertices in D is at most r.

In parameterized algorithms, for decision problems with input size n, and a parameter k, the goal is to design an algorithm with runtime $f(k) \cdot n^{O(1)}$, where f is a function of k alone. Problems having such an algorithm are said to be fixed parameter tractable (FPT). There is also a theory of hardness using which one can identify parameterized problems that are not amenable to such algorithms. This hardness hierarchy is represented by $W[i]$ for $i \geq 1$. For an introduction and more recent developments see the books [13, 14, 29]. In this paper, we always parameterize a problem by the size of the cover, that is, the positive integer k.

Most of the research on partial cover problems in parameterized complexity has considered the number of objects to be covered (t) as a parameter rather than the the size of the cover (k). Bläser [4] initiated the study of partial cover problems parameterized by t and obtained a randomized algorithm with running time $5.45^t n^{\mathcal{O}(1)}$ for PDS. Kneis et al. [25] improved this algorithm and obtained a randomized algorithm with running time $(4+\epsilon)^t n^{\mathcal{O}(1)}$ for every fixed $\epsilon > 0$. Recently, Koutis and Williams [27] obtained an even faster randomized algorithm for PDS, which runs in time $2^t n^{\mathcal{O}(1)}$. Kneis et al. [24] studied the PVC problem when parameterized by the number edged to be covered (t) and obtained a randomized algorithm running in time $2.0911^t n^{\mathcal{O}(1)}$. The algorithm for PVC was recently improved by Kneis et al. [23]. They obtain a randomized algorithm with running time $1.2993^t n^{\mathcal{O}(1)}$ and a deterministic algorithm with running time $1.396^t n^{\mathcal{O}(1)}$ for PVC. When parameterized by the size of cover k, PVC is known to be $W[1]$-complete [17]. The P-r-DS problem being a generalization of DOMINATING SET is also known to be $W[2]$-hard on general graphs when parameterized by the cover size. Amini et al. [1] considered these problems with the size of the cover k being the parameter and initiated a study of these problem on sparse graphs namely planar graphs, apex minor free graphs and H-minor free graphs. They obtained algorithms with running time $2^{\mathcal{O}(k)} n^{\mathcal{O}(1)}$ for PVC and P-r-DS and left an open question of whether these problems have an algorithm with running time $2^{o(k)} n^{\mathcal{O}(1)}$, like their non partial counterpart on planar graphs or more generally on H-minor free graphs. In this paper we answer this question in affirmative an obtain algorithms with running time $2^{\mathcal{O}(\sqrt{k})} n^{\mathcal{O}(1)}$ on apex minor free graphs for PVC and P-r-DS.

Most of the known sub-exponential time algorithms on planar graphs, graphs of bounded genus, apex minor free graphs and H-minor free graphs are based on the meta-algorithmic theory of Bidimensionality, developed by Demaine et al. [7]. The bidimensionality theory is based on algorithmic and combinatorial extensions to parts of Graph Minors Theory of Robertson and Seymour [30] and provides a simple criteria for checking whether a parameterized problem is solvable in subexponential time on sparse graphs. The theory applies to the graph problems that are *bidimensional* in the sense that the value of the solution for the problem in question on $k \times k$ grid or "grid like graph" is at least $\Omega(k^2)$ and the value of solution decreases while contracting or sometime deleting the edges. Problems that are bidimensional include k-Feedback Vertex Set, k-Edge Dominating Set, k-Leaf Spanning Tree, k-Path, k-rDominating Set, k-Vertex Cover and many others. We refer to surveys by Demaine and Hajiaghayi [10] and Dorn et al. [12] for further details on bidimensionality and subexponential parameterized algorithms. But neither PVC nor P-r-DS are bidimensional problems and hence this theory is *not amenable* to our problems.

However our subexponential time algorithms for PVC and P-r-DS are based on a technique used to solve the classical Disjoint Path problem in the Graph Minors Theory of Robertson and Seymour [31], called *irrelevant vertex* argument. The technique can be described as follows, in polynomial time we either find a vertex which is irrelevant for the solution and hence can be deleted or when we can not find an irrelevant vertex then show that the reduced instance has bounded treewidth. This technique has recently been used to solve several problems around finding disjoint paths [19, 20, 21, 22, 26]. To obtain subexponential time algorithms for PVC and P-r-DS we introduce a notion of "lexicographically smallest" solution and use its properties to obtain an irrelevant vertex in the graph. When we can not find any irrelevant vertex then we are able to show that that the treewidth of the reduced graph is at most $\mathcal{O}(\sqrt{k})$. Once we have a sublinear bound on the treewidth of the input graph, we can solve the problem in $2^{\mathcal{O}(\sqrt{k})}n^{\mathcal{O}(1)}$ time using dynamic programming over graphs of bounded treewidth.

2 Preliminaries

Let $G = (V, E)$ be an undirected graph where V is the set of vertices and E is the set of edges. We denote the number of vertices by n and number of edges by m. For a subset $V' \subseteq V$, by $G[V']$ we mean the subgraph of G induced by V'. By $N(u)$ we denote (open) neighborhood of u that is set of all vertices adjacent to u and by $N[u] = N(u) \cup \{u\}$. Similarly, for a subset $D \subseteq V$, we define $N[D] = \cup_{v\in D} N[v]$. The *distance* $d_G(u, v)$ between two vertices u and v of G is the length of the shortest path in G from u to v. For $r \geq 0$, the *r-neighborhood* of a vertex $v \in V$ is defined as $N_r[v] = \{u \mid d_G(v, u) \leq r\}$. We also let $B_r(v) = N_r[v]$ and call it a ball of radius r around v. Similarly $B_r(A) = \cup_{v\in A} N_G^r[v]$ for $A \subseteq V$. For a given vertex $v \in V$ by $\partial(v)$ we denote the set of edges which are incident with v. For a subset $X \subseteq V$, $\partial(S) = \cup_{v\in S}\partial(v)$.

Given an edge $e = (u, v)$ of a graph G, the graph G/e is obtained by contracting the edge (u, v) that is we get G/e by identifying the vertices u and v and removing all the loops and duplicate edges. A *minor* of a graph G is a graph H that can be obtained from a subgraph of G by contracting edges. A graph class $\mathcal{C}$ is *minor closed* if any minor of any graph in $\mathcal{C}$ is also an element of $\mathcal{C}$. A minor closed graph class $\mathcal{C}$ is *H-minor-free* or simply *H-free* if $H \notin \mathcal{C}$. A graph H is called an apex graph if the removal of one vertex makes it a planar graph.

A *tree decomposition* of a graph $G = (V, E)$ is a pair (X, T) where T is a tree on vertex set $V(T)$ whose vertices we call *nodes* and $X = (\{X_i \mid i \in V(T)\})$ is a collection of subsets of V such that

1. $\bigcup_{i\in V(T)} X_i = V$,
2. for each edge $(v, w) \in E$, there is an $i \in V(T)$ such that $\{v, w\} \in X_i$, and
3. for each $v \in V$ the set of nodes $\{i \mid v \in X_i\}$ forms a subtree of T.

The *width* of a tree decomposition $(\{X_i \mid i \in V(T)\}, T)$ equals $\max_{i\in V(T)}\{|X_i| - 1\}$. The *treewidth* of a graph G is the minimum width over all tree decompositions of G. We use notation $\mathbf{tw}(G)$ to denote the treewidth of a graph G.

Given a graph $G = (V, E)$ a set of vertices D of V is called a *r-dominating set* for G if $N_r(D) = V$. For $r = 1$ the set D is called a *dominating set*. In the r-Dominating Set problem, we are given a graph $G = (V, E)$ and the objective is to find the smallest sized D such that $N_r(D) = V$.

3 Subexponential algorithm for Partial Vertex Cover

In this section we consider the PVC problem. In fact we will solve a slightly general problem, that is, given an undirected graph, a non negative integer k, we find the *maximum* number of edges that can be covered by a subset of at most k vertices. The decision version of the problem is precisely PVC. If the maximum number of edges covered by any vertex set of size at most k is at least t then we return "yes" else we return "no".

The key idea of the algorithm is to identify an *irrelevant* set of vertices, I, which can be deleted without destroying at least one set $C \subseteq V$ such that $|C| \leq k$ and $|\partial(C)| \geq t$. Then we will show that the $\mathbf{tw}(G[V \setminus I]) \leq \mathcal{O}(\sqrt{k})$ and hence the dynamic programming over graphs of bounded treewidth comes into picture. To identify a set of irrelevant vertices we introduce the notion of *lexicographically smallest solution*.

Definition 1. *Given a graph $G = (V, E)$ and an ordering $\sigma = v_1 \dots v_n$ of the vertex set, we say that a subset X is less that a subset Y if X is smaller than the Y with respect to the lexicographic ordering defined by σ and is denoted by $X \leq_\sigma Y$. We call a set $C \subseteq V$ a* lexicographically smallest solution *for PVC if for any other solution C' for the PVC we have that $C \leq_\sigma C'$.*

We are interested in a particular ordering of a vertex set. Let $\sigma = v_1 v_2 \dots v_n$ be an ordering of the vertices such that the vertices are in non increasing order of their degrees, with ties being broken arbitrarily. That is,

$$d(v_1) \geq d(v_2) \cdots \geq d(v_{n-1}) \geq d(v_n).$$

From now onwards we will assume that the vertex set of the input graph is ordered by *this* fixed ordering σ and denote the graph by $G = (V_\sigma, E)$ to *emphasize* the fact that the vertex set is order with respect to σ. By V_σ^i we denote the vertex set $v_1 \dots v_i$. Our goal will be to find a lexicographically smallest solution for PVC. The algorithm is based on the following properties of a lexicographically smallest solution for PVC.

Lemma 1. *Let $G = (V_\sigma, E)$ be an yes instance to* PVC*, $C = \{u_{i_1}, \dots, u_{i_k}\}$ be the lexicographically smallest solution for* PVC *and $u_{i_k} = v_j$ for some j. Then C is a dominating set of size at most k for $G[V_\sigma^j]$.*

Proof. Let us assume to the contrary that C is not a dominating set for $G[V_\sigma^j]$. Then there exist a vertex v_i, $1 \leq i < j$ such that $N[v_i] \cap C = \emptyset$. Set $C' := C \setminus \{v_j\} \cup \{v_i\}$. We claim that the edges covered by C' is at least as mcuh as the edges covered by C. That is, $|\partial(C')| \geq |\partial(C)|$. Since $d(v_i) \geq d(v_j)$, we have that

$$|\partial(C')| \geq |\partial(C)| - d(v_j) + d(v_i) \geq |\partial(C)|.$$

Hence, $|C'| = |C|$, C' is lexicographically smaller than C and $|\partial(C')| \geq |\partial(C)|$ a contradiction to the choice of C. ☐

We also need the following results for our algorithm.

Lemma 2. *Let G be a n-vertex graph excluding an apex graph H as a minor. If G has a r-dominating set of size at most k, then G has treewidth at most $c_H r\sqrt{k} = \mathcal{O}(r\sqrt{k})$, where c_H is a constant only depending on the size of H.*

Lemma 2 follows from the fact that r-dominating set is a "contraction bidimensional" parameter and then applying the result that if a contraction bidimensional parameter has value at most k on a graph G which excludes an apex graph H then $\mathbf{tw}(G) \leq \mathcal{O}(r\sqrt{k})$ [6, 8, 15] . We will use the following known algorithm to solve PVC on graphs of bounded treewidth.

Lemma 3 ([28])**.** *Let G be an undireceted graph such that the treewidth of G is at most w. Then in time $2^w n^{\mathcal{O}(1)}$ we can find a subset C of at most k vertices that cover the maximum number of edges of G.*

For our proof we also need the following result by Demaine and Hajiaghayi to obtain a polynomial time approximation scheme (PTAS) for r-Dominating Set.

Lemma 4 ([9])**.** *There is a PTAS for* r-Dominating Set *on apex-minor-free graphs.*

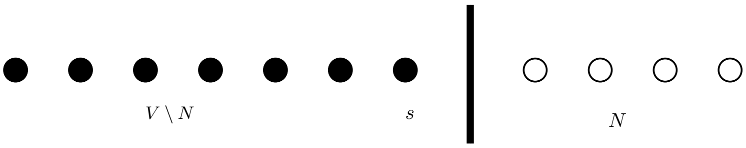

Figure 1: The Algorithmic Schema

The basic schema of the algorithm is as follows. We start with the vertex set V_σ and scan the vertices following the *reverse* of the ordering $\sigma = v_1 v_2 \ldots v_n$. That is, we scan the vertices in the order $v_n v_{n-1} \ldots v_2 v_1$. The algorithm can be viewed intuitively as follows. We think that we have a stick, initially positioned to the left of v_n and then we *slide* it towards left if the vertex to its left satisfies certain properties. At any intermediate stage we have a vertex set N which is a consecutive block of the ordering σ at the end of it. The vertex set s is the first vertex to the left of N. We refer to Figure 1 for a better viewpoint of the scenario we are trying to describe. The stick essentially represents the fact that lexicographically smallest solution C we are looking for lies completely in $V \setminus N$, that is, $C \subseteq V \setminus N$. To slide the stick we do as follows. Let $s = v_j$ for some j. Now we check whether $G[V_\sigma^j]$ has a dominating set of "roughly size k". If the dominating set of $G[V_\sigma^j]$ is not roughly equal to k then we slide. Else we find an appropriate induce subgraph $G' = (V', E')$ of G such that $\mathbf{tw}(G') \leq \mathcal{O}(\sqrt{k})$ and G has a set C of size at most k such that $|\partial(C)| \geq t$ if and only if there exists a set $C' \subseteq V'$ such that $|C'| \leq k$ and $|\partial(C')| \geq t$. A formal description of our algorithm for partial vertex cover is given in Figure 2. The Algo-PC is called with the parameter $(G = (V_\sigma, E), k, \epsilon, \emptyset)$. Now we state our main theorem for this section.

Theorem 1. *Let $G = (V, E)$ a graph that excludes an apex graph H as a minor and k and t be a positive integers. Then in $2^{\mathcal{O}(\sqrt{k})} n^{\mathcal{O}(1)}$ time we can solve determine whether there exists a subset $C \subseteq V$ of size at most k such that $|\partial(C)| \geq t$.*

Proof. We argue the correctness of the algorithm. In the first part of the algorithm we try to identify the subset N of vertices such that it does not intersect with the lexicographically least solution, C, we are looking for. We iteratively run through the vertices in the reverse order and try to maintain the invariant that N is a subset of the vertices that does not intersect with the lexicographically least solution. Initially N is empty, so the invariant trivially holds. The set N only grows if in any step we have that $G[V \setminus N]$ has a dominating set of size more than $(1+\epsilon)k$. Let v_p be the largest indexed vertex in $V \setminus N$, that is, v_p is to the left of the set N in the ordering σ. Now by Lemma 1, we know that if $v_p \in C$ then $G[V \setminus N]$ has a dominating set of size at most k and hence the PTAS from Lemma 4 would find an approximate dominating

```
ALGO-PC($G = (V_\sigma, E), k, \epsilon, N$)
(Here $G$ is a graph, $k$ is a non negative integer, $\epsilon > 0$ is an arbitrary fixed constant, $N$ is a set of vertices, and the goal is to find a subset of $V \setminus N$ of size at most $k$ that covers the maximum number of edges of $G = (V, E)$.)

  1. Let $p := n$ and $yesdom \leftarrow false$.
  2. While $yesdom = false$
       • Using Lemma 4 find, if exists, a dominating set $D$ of size at most $(1+\epsilon)k$ for $G[V \setminus N]$. If such a dominating set exists, then $yesdom \leftarrow true$. Else set $N := N \cup \{v_p\}$ and $p := p - 1$.
     endwhile
  3. Let $I = \{u \mid u \in N, N(u) \subseteq N\}$ and set $V' = V \setminus I$. Find a tree-decomposition $(U, T)$ of $G[V']$ using the constant factor approximation algorithm of Demaine et al. [11] for computing the treewidth of $H$-minor free graph. Now applying the Lemma 3 find a subset $C'$ of size at most $k$ of $G[V']$ which covers the maximum number of edges.
```

Figure 2: Description of the partial cover Algorithm

set of size at most $(1+\epsilon)k$. This implies that $v_p \notin C$ and hence we can safely place v_p in N. This proves the correctness of the first part.

Note that edges in $G[N]$ will not be covered by C, and hence vertices in N that have neighbors only in N are deleted at the end. The set I is the irrelevant set of vertices we were looking for. Let $V' = V \setminus I$. Thus we have shown that G has a set C of size at most k such that $|\partial(C)| \geq t$ if and only if there exists a set $C' \subseteq V'$ such that $|C'| \leq k$ and $|\partial(C')| \geq t$. Now applying the Lemma 3 we find a subset C' of size at most k of $G[V']$ which covers the maximum number of edges. So if $|\partial(C')| \geq t$ then we return "yes" else we return "no". The correctness of this step follows from Lemma 3.

Now we analyze the time complexity of the algorithm. We know that $G[V \setminus N]$ has a dominating set of size at most $(1+\epsilon)k$. Let D be a dominating set of $G[V \setminus N]$ of size at most $(1+\epsilon)k$. This implies that D is a 2-*dominating set* of $G[V']$ as every vertex $v \in (N \cap V')$ has a neighbor in $V \setminus N$. Hence by Lemma 2, $\mathbf{tw}(G') \leq \mathcal{O}(\sqrt{(1+\epsilon)k}) = \mathcal{O}(\sqrt{k})$. Now using the constant factor approximation algorithm of Demaine et al. [11] for computing the treewidth of H-minor free graph, we find a tree-decomposition of $G[V']$ of width $\mathcal{O}(\sqrt{k})$ in time $n^{\mathcal{O}(1)}$. Finally, the dynamic programming algorithm mentioned in Lemma 3 runs in time $2^w n^{\mathcal{O}(1)}$ on graphs of treewidth w and hence our algorithm has running time $2^{\mathcal{O}(\sqrt{k})} n^{\mathcal{O}(1)}$. ☐

4 Partial dominating set problems

Here the problem is to cover (dominate) as many vertices as possible using at most k vertices, in their balls of radius r.

We first modify the Lemma **??** to prove the following.

Lemma 5. *Given an undirected graph $G(V, E)$, let $v_1, v_2, \ldots v_n$ be an ordering of the vertices in non increasing order of their sizes of $N_r(v)$, i.e. $|N_r(v_i)| \geq |N_r(v_{i+1})|$ with ties being broken arbitrarily. Let S be the lexicographically smallest subset of at most k vertices that r-dominates* *the maximum number of vertices of G. (I.e. whenever we have a choice of two sets of size at most k that r-dominate the same number of vertices, we pick the set which has a smaller label when we order the vertices in increasing order of their labels.). If the vertex v_n is in S then $S - v_n$ is a $2r$-dominating set for $V - v_n$ of size at most $k - 1$.*

Proof. Let $N_r(S) = \bigcup_{s \in S} N_r(s)$ be the number of vertices r-dominated by S, and suppose $S - v_n$ is not a $2r$-dominating set of $V - v_n$. Let x be a vertex of $V - v_n$ not $2r$-dominated by $S - v_n$. Then $N_r(x) \cup N_r(s) = \emptyset$ for every $s \in S - v_n$ as otherwise if for some vertex $s \in S - v_n$, the intersection is non empty, then x will be $2r$ dominated by s. Let $S' = S - v_n \cup \{x\}$, then $|S'| = |S|$, S' is lexicographically smaller than S and $N_r(S') \geq N_r(S) + N_r(x) - N_r(v_n) \geq N_r(S)$ a contradiction to the choice of S. □

We also need a lemma similar to Lemma 3 which we state below.

Lemma 6 ([7]). *Let G be an undireceted graph such that the treewidth of G is at most w. Then in time $(2r+1)^{1.5w} n^{\mathcal{O}(1)}$ we can find a subset C of at most k vertices that that r-dominate the maximum number of vertices of G.*

With all these ingredients, the subexponential algorithm for the r-dominating set is very similar to our algorithm for partial vertex cover.

Theorem 2. *Given a graph $G(V, E)$ from a non-trivial minor closed family of graphs , and an integer k, we can determine in $2^{O(\sqrt{k})} + n^{O(1)}$ time, the maximum number of vertices of G that can be r-dominated by a subset of at most k vertices.*

5 Conclusion

References

[1] O. Amini, F. V. Fomin, and S. Saurabh. Implicit branching and parameterized partial cover problems (extended abstract). In *FSTTCS*, 2008.

[2] S. Arora and G. Karakostas. A 2+epsilon approximation algorithm for the -mst problem. In *SODA*, pages 754–759, 2000.

[3] R. Bar-Yehuda. Using homogenous weights for approximating the partial cover problem. In *SODA*, pages 71–75, 1999.

[4] M. Bläser. Computing small partial coverings. *Inf. Process. Lett.*, 85(6):327–331, 2003.

[5] M. Charikar, S. Khuller, D. M. Mount, and G. Narasimhan. Algorithms for facility location problems with outliers. In *SODA*, pages 642–651, 2001.

[6] E. D. Demaine, F. V. Fomin, M. T. Hajiaghayi, and D. M. Thilikos. Bidimensional parameters and local treewidth. *SIAM J. Discrete Math.*, 18(3):501–511, 2004.

[7] E. D. Demaine, F. V. Fomin, M. T. Hajiaghayi, and D. M. Thilikos. Subexponential parameterized algorithms on bounded-genus graphs and -minor-free graphs. *J. ACM*, 52(6):866–893, 2005.

[8] E. D. Demaine and M. Hajiaghayi. Linearity of grid minors in treewidth with applications through bidimensionality. *Combinatorica*, 28(1):19–36, 2008.

[9] E. D. Demaine and M. T. Hajiaghayi. Bidimensionality: new connections between fpt algorithms and ptass. In *SODA*, pages 590–601, 2005.

[10] E. D. Demaine and M. T. Hajiaghayi. The bidimensionality theory and its algorithmic applications. *Computer Journal*, 51(3):292–302, 2008.

[11] E. D. Demaine, M. T. Hajiaghayi, and K. Kawarabayashi. Algorithmic graph minor theory: Decomposition, approximation, and coloring. In *FOCS*, pages 637–646, 2005.

[12] F. Dorn, F. V. Fomin, and D. M. Thilikos. Subexponential parameterized algorithms. *Computer Science Review*, 2(1):29–39, 2008.

[13] R. G. Downey and M. R. Fellows. *Parameterized complexity.* Springer-Verlag, New York, 1999.

[14] J. Flum and M. Grohe. *Parameterized Complexity Theory.* Texts in Theoretical Computer Science. An EATCS Series. Springer-Verlag, Berlin, 2006.

[15] F. V. Fomin, P. A. Golovach, and D. M. Thilikos. Contraction bidimensionality: the accurate picture. In *ESA 09*, LNCS, Berlin Heidelberg, 2009. Springer-Verlag.

[16] R. Gandhi, S. Khuller, and A. Srinivasan. Approximation algorithms for partial covering problems. *J. Algorithms*, 53(1):55–84, 2004.

[17] J. Guo, R. Niedermeier, and S. Wernicke. Parameterized complexity of vertex cover variants. *Theory Comput. Syst.*, 41(3):501–520, 2007.

[18] E. Halperin and A. Srinivasan. Improved approximation algorithms for the partial vertex cover problem. In K. Jansen, S. Leonardi, and V. V. Vazirani, editors, *APPROX*, volume 2462 of *Lecture Notes in Computer Science*, pages 161–174. Springer, 2002.

[19] K. Kawarabayashi. An improved algorithm for finding cycles through elements. In *IPCO*, volume 5035 of *Lecture Notes in Computer Science*, pages 374–384, 2008.

[20] K. Kawarabayashi and Y. Kobayashi. The induced disjoint paths problem. In *IPCO*, volume 5035 of *Lecture Notes in Computer Science*, pages 47–61, 2008.

[21] K. Kawarabayashi and B. A. Reed. A nearly linear time algorithm for the half integral disjoint paths packing. In *SODA*, pages 446–454, 2008.

[22] K. Kawarabayashi and B. A. Reed. A nearly linear time algorithm for the half integral parity disjoint paths packing problem. In *SODA*, pages 1183–1192, 2009.

[23] J. Kneis, A. Langer, and P. Rossmanith. Improved upper bounds for partial vertex cover. In *WG*, volume 5344 of *Lecture Notes in Computer Science*, pages 240–251, 2008.

[24] J. Kneis, D. Mölle, S. Richter, and P. Rossmanith. Intuitive algorithms and t-vertex cover. In *ISAAC*, volume 4288 of *Lecture Notes in Computer Science*, pages 598–607, 2006.

[25] J. Kneis, D. Mölle, and P. Rossmanith. Partial vs. complete domination: t-dominating set. In *SOFSEM (1)*, volume 4362 of *Lecture Notes in Computer Science*, pages 367–376, 2007.

[26] Y. Kobayashi and K. Kawarabayashi. Algorithms for finding an induced cycle in planar graphs and bounded genus graphs. In *SODA*, pages 1146–1155, 2009.

[27] I. Koutis and R. William. Limits and applications of group algebras for parameterized problems. In *ICALP 09*, LNCS, Berlin Heidelberg, 2009. Springer-Verlag.

[28] H. Moser. Exact Algorithms for Generalizations of Vertex Cover. Master's thesis, Institut für Informatik, Friedrich-Schiller-Universität Jena, Germany, 2005.

[29] R. Niedermeier. *Invitation to fixed-parameter algorithms*, volume 31 of *Oxford Lecture Series in Mathematics and its Applications.* Oxford University Press, Oxford, 2006.

[30] N. Robertson, P. Seymour, and R. Thomas. Quickly excluding a planar graph. *Journal of Combinatorial Theory Series B*, 62:323–348, 1994.

[31] N. Robertson and P. D. Seymour. Graph minors .xiii. the disjoint paths problem. *J. Comb. Theory, Ser. B*, 63(1):65–110, 1995.