Polynomial Kernels for Dominating Set in Graphs of Bounded Degeneracy and Beyond*

Geevarghese Philip[1] Venkatesh Raman[1]
Somnath Sikdar[2,†]

[1]The Institute of Mathematical Sciences, Chennai, India.
{gphilip|vraman}@imsc.res.in

[2]RWTH Aachen University, Aachen, Germany.
sikdar@cs.rwth-aachen.de

Abstract

We show that for any fixed $j \geq i \geq 1$, the k-DOMINATING SET problem restricted to graphs that do not have $K_{i,j}$ (the complete bipartite graph on $(i+j)$ vertices, where the two parts have i and j vertices, respectively) as a subgraph is fixed parameter tractable (FPT) and has a polynomial kernel. We describe a polynomial-time algorithm that, given a $K_{i,j}$-free graph G and a nonnegative integer k, constructs a graph H (the "kernel") and an integer k' such that (1) G has a dominating set of size at most k if and only if H has a dominating set of size at most k', (2) H has $O((j+1)^{2(i+1)}k^{2i^2})$ vertices, and (3) $k' = O((j+1)^{i+1}k^{i^2})$.

Since d-degenerate graphs do not have $K_{d+1,d+1}$ as a subgraph, this immediately yields a polynomial kernel on $O((d+2)^{2(d+2)}k^{2(d+1)^2})$ vertices for the k-DOMINATING SET problem on d-degenerate graphs, solving an open problem posed by Alon and Gutner [3].

The most general class of graphs for which a polynomial kernel was previously known for k-DOMINATING SET is the class of K_h-topological-minor-free graphs [21]. Graphs of bounded degeneracy are the most general class of graphs for which an FPT algorithm was previously known for this problem. K_h-topological-minor-free graphs are $K_{h,h}$-free (but not vice versa), and so our results show that k-DOMINATING SET has both FPT algorithms and polynomial kernels in more general classes of graphs.

Using the same techniques, we also obtain an $O\left(jk^i\right)$ vertex-kernel for the k-INDEPENDENT DOMINATING SET problem on $K_{i,j}$-free graphs.

*A preliminary version of this paper appeared in the proceedings of ESA 2009 [25].
†Work done while the author was at the Institute of Mathematical Sciences, Chennai, India.

1 Introduction

A *dominating set* of a graph $G = (V, E)$ is a set $S \subseteq V$ of vertices of G such that every vertex in $V \setminus S$ is adjacent to some vertex in S. The DOMINATING SET problem is defined as follows:

DOMINATING SET
Input: A graph $G = (V, E)$ and a non-negative integer k.
Question: Does G have a dominating set with at most k vertices?

The DOMINATING SET problem is NP-hard, even in very restricted graph classes such as the class of planar graphs with maximum degree 3 [18]. Hence, unless P=NP, there is no polynomial-time algorithm that solves the problem even in such restricted graph classes.

Parameterized algorithms [11, 14, 24] constitute one approach towards solving NP-hard problems in "feasible" time. Each parameterized problem comes with an associated *parameter*, which is usually a non-negative integer, and the goal is to find algorithms that solve the problem in polynomial time *when the parameter is fixed*, where the degree of the polynomial is independent of the parameter. More precisely, if k is the parameter and n the size of the input, then the goal is to obtain an algorithm that solves the problem in time $f(k) \cdot n^c$ where f is some computable function and c is a constant independent of k. Such an algorithm is called a fixed-parameter-tractable (FPT) algorithm, and the class of all parameterized problems that have FPT algorithms is called FPT; a parameterized problem that has a fixed-parameter-tractable algorithm is said to be (in) FPT.

Together with this revised notion of tractability, parameterized complexity theory offers a corresponding notion of intractability as well, captured by the concept of *W-hardness*. In brief, the theory defines a hierarchy of complexity classes FPT $\subset$ W[1] $\subset$ W[2] $\cdots \subset$ XP, where each inclusion is believed to be strict — on the basis of evidence similar in spirit to the evidence for believing that P$\neq$NP — and XP is the class of all parameterized problems that can be solved in $O(n^{f(k)})$ time where n is the input size, k the parameter, and f is some computable function [11, 24].

One natural parameter for the DOMINATING SET problem is k, the size of the solution being sought. A natural parameterized version of the DOMINATING SET problem is thus the k-DOMINATING SET problem, defined as follows:

k-DOMINATING SET
Input: A graph $G = (V, E)$, and a non-negative integer k.
Parameter: k
Question: Does G have a dominating set with at most k vertices?

It turns out that the DOMINATING SET problem, with this parameterization, is still hard to solve. More precisely, k-DOMINATING SET is the canonical W[2]-hard problem [11], and the problem remains W[2]-hard even in many restricted classes of graphs — for example, it is W[2]-hard in classes of graphs with bounded average degree [19]. Thus there is no FPT algorithm that solves the problem, even when restricted to graphs of bounded average degree, unless FPT=W[2], which is considered unlikely.

The problem does have FPT algorithms in certain restricted families of graphs, such as in planar graphs [16], graphs of bounded genus [13], nowhere-dense classes of graphs[7], K_h-topological-minor-free graphs, and graphs of bounded degeneracy [2]. To the best of our knowledge, graphs of bounded degeneracy are the most general graph class previously known to have an FPT algorithm for this problem. In this paper we show that the problem has an FPT algorithm in a class of graphs that encompasses, and is strictly larger than, all these classes — namely, the class of $K_{i,j}$-free graphs.

Closely related to the notion of an FPT algorithm is the concept of a *kernel* for a parameterized problem. We say that two instances of a decision problem are *equivalent* if and only if they are either both yes-instances or both no-instances. A *kernelization algorithm* for a parameterized problem is a polynomial-time algorithm that converts an instance (x,k) of the problem to an equivalent instance (y,k') whose size $|y|$ and parameter k' are both bounded by functions of the original parameter k. The instance y output by the algorithm is said to be a *kernel* for the problem. It is not difficult to see that if a problem has a kernelization algorithm, then the problem is FPT. Somewhat more surprisingly, the converse is also true: A folklore theorem of parameterized complexity states that a parameterized problem has a kernelization algorithm if and only if it has an FPT algorithm [11].

For the k-DOMINATING SET problem, a kernelization algorithm is thus an algorithm that takes (G,k) as input, runs in polynomial time, and outputs an equivalent instance (H,k'), where $k' \leq g(k)$ and H is a graph with at most $h(k)$ vertices for some computable functions g and h. H is the kernel output by this algorithm. From the equivalence of FPT and kernelization mentioned above it follows that, unless FPT=W[2], there is no kernelization algorithm for k-DOMINATING SET on general graphs or on graphs with a bounded average degree. For the same reason, the problem admits kernelization algorithms when the input is restricted to planar graphs, graphs of bounded genus, K_h-topological-minor-free graphs, and graphs of bounded degeneracy. However, the *size* of the kernel implied by the proof of the folklore theorem is equal to the factor $f(k)$ in the running time of the corresponding FPT algorithm, and hence is exponential in k. The interesting problem is, therefore, to find if the kernel size can be made smaller — in particular, whether it can be made polynomial in k.

Proving polynomial bounds on the size of the kernel for different parameterized problems has been a significant practical aspect in the study of the

parameterized complexity of NP-hard problems, and many positive results are known; see the survey on kernelization results by Guo et al. [20].

For the k-DOMINATING SET problem, the first polynomial kernel result was obtained by Alber et al. [1] in 2004: they showed that in *planar graphs*, the problem has a *linear* kernel on at most $335k$ vertices. This bound for planar graphs was later improved to $67k$ by Chen et al. [5]. Fomin and Thilikos [15] showed in 2004 that the same reduction rules as used by Alber et al. give a linear kernel (linear in $k+g$) for k-DOMINATING SET restricted to graphs of genus g. The next advances in kernelizing this problem were made by Alon and Gutner in 2008 [3, 21]. They showed that the problem has a linear kernel in $K_{3,h}$-topological-minor-free graph classes (which include, for example, planar graphs), and a polynomial kernel in K_h-topological-minor-free graph classes. Here K_h denotes the complete graph on h vertices, and $K_{3,h}$ is the complete bipartite graph on $h+3$ vertices where one piece of the partition has 3 vertices and the other has h. The degree of the polynomial bound on the kernel size for K_h-topological-minor-free graphs depends on h, and these are the most general class of graphs for which the problem has been previously shown to have a polynomial kernel. In the meantime, the same authors had shown in 2007 that the problem is FPT in (the strictly larger class of) graphs of bounded degeneracy [2], but had left open the question whether the problem has a polynomial kernel in such graph classes. In this paper, we answer this question in the affirmative, and show that, in fact, even larger classes of graphs — the $K_{i,j}$-free graph classes — admit polynomial kernels for this problem. In Table 1 we summarize some FPT and kernelization results for the k-DOMINATING SET problem on various classes of graphs.

Our Results. $K_{i,j}$ denotes the complete bipartite graph on $i+j$ vertices where one piece of the partition has i vertices and the other part has j. A graph is said to be *$K_{i,j}$-free* if it does not contain $K_{i,j}$ as a (not necessarily induced) subgraph. We show that for any fixed $i, j \geq 1$, the k-DOMINATING SET problem has a polynomial kernel on $K_{i,j}$-free graphs. For input graph G and parameter k, the size of the kernel is bounded by k^c where c is a constant that depends only on i and j.

A graph G is said to be *d-degenerate* if every subgraph of G has a vertex of degree at most d. Since a d-degenerate graph does not have $K_{d+1,d+1}$ as a subgraph, it follows that the k-DOMINATING SET problem has a polynomial kernel on graphs of bounded degeneracy. This settles a question posed by Alon and Gutner [3].

A subset S of the vertex set of a graph is said to be *independent* if no two vertices in S have an edge between them in the graph. The k-INDEPENDENT DOMINATING SET problem asks whether the input graph G has an independent DOMINATING SET of size at most k, with the parameter being k. We show that the k-INDEPENDENT DOMINATING SET problem has a polynomial kernel in $K_{i,j}$-

Graph Class	*FPT Algorithm Running Time*	*Kernel Size*
Planar	$O(k^4 + 2^{15.13\sqrt{k}}k + n^3)$ [16]	$O(k)$ [1, 5]
Genus-g	$O((24g^2 + 24g + 1)^k n^2)$ [13]	$O(k + g)$ [15]
K_h-minor-free	$O(n^{3.5} + 2^{O(\sqrt{k})})$ [21]	$O(k^c)$ [21]
K_h-topological-minor-free	$(O(h))^{hk} \cdot n$ [2]	$O(k^c)$ [21]
d-degenerate	$k^{O(dk)}n$ [2]	$k^{O(dk)}$ [2], $O(k^{2(d+1)^2})^\dagger$
$K_{i,j}$-free	$O(n^{i+O(1)} + 2^{O(k^{2i^2})})^\dagger$	$O(k^{2i^2})^\dagger$

Table 1: Some FPT and kernelization results for k-DOMINATING SET. Results obtained in this paper are marked with a †.

free graphs.

Note that except for d-degenerate and $K_{i,j}$-free graphs, all the other graph classes in Table 1 are *minor-closed* (See e.g., [9], Chapter 12, for the definition of a minor-closed graph class.). This seems to be indicative of the state of the art — the only other previous FPT or kernelization result for the k-DOMINATING SET problem on a non-minor-closed class of graphs that we are aware of is the $O(k^3)$ kernel and the resulting FPT algorithm for graphs that exclude triangles and 4-cycles [26]. In fact, this result can be modified to obtain similar bounds on graphs with no 4-cycles (allowing triangles). Since a 4-cycle is a $K_{2,2}$, this result follows from the main result of this paper by setting $i = j = 2$.

Since, for a constant h, a K_h-topological-minor-free graph has bounded degeneracy [3, Proposition 3.1], the class of $K_{i,j}$-free graphs is more general than the class of K_h-topological-minor-free graphs. Thus we extend the class of graphs for which the k-DOMINATING SET problem is known to have (1) FPT algorithms and (2) polynomial kernels, to the class of $K_{i,j}$-free graphs.

Organization of the rest of the paper. In Section 2 we present our main kernelization algorithm that, for fixed $j \geq i \geq 2$, runs in $O(n^{i+O(1)})$ time* and constructs a kernel of size $O((j+1)^{2(i+1)}k^{2i^2})$ for k-DOMINATING SET on $K_{i,j}$-free graphs. As a corollary we obtain, in Section 3, a polynomial kernel for the problem restricted to d-degenerate graphs, where the kernelization algorithm runs in time $O(n^{O(d)})$ and outputs a kernel of size $O((d+2)^{2(d+2)}k^{2(d+1)^2})$. In

*Throughout this paragraph, n denotes the number of vertices in the input graph.

Section 3.1 we describe an improvement to the above algorithm that applies to d-degenerate input graphs, yields a kernel of the same size as above, and runs in time $O(2^d dn^2)$. In Section 4 we describe a modification of the algorithm in Section 2 that constructs a polynomial kernel for the k-INDEPENDENT DOMINATING SET problem on $K_{i,j}$-free graphs. This kernel has $O(jk^i)$ vertices, and so implies a kernel of size $O((d+1)k^{d+1})$ for this problem on d-degenerate graphs. In Section 5 we state our conclusions and list some open problems.

Notation. All the graphs in this paper are finite, undirected and simple. In general we follow the graph terminology of [9]. We let $V(G)$ and $E(G)$ denote, respectively, the vertex and edge sets of a graph G. The *open-neighborhood* of a vertex v in a graph G, denoted $N(v)$, is the set of all vertices that are adjacent to v in G. A k-DOMINATING SET of graph G is a vertex-subset S of size at most k such that for each $u \in V(G) \setminus S$ there exists $v \in S$ such that $\{u,v\} \in E(G)$. Given a graph G and $A, B \subseteq V(G)$, we say that A dominates B if every vertex in $B \setminus A$ is adjacent in G to some vertex in A.

Let H be a graph obtained from a copy of a graph G by applying some changes, and let S be a vertex subset of G whose copy survives in H. For ease of presentation, we sometimes abuse notation and use S to denote the copy of S in H as well.

2 A Polynomial Kernel for $K_{i,j}$-free Graphs

In this section we consider the k-DOMINATING SET problem on graphs that do not have $K_{i,j}$ as a subgraph, for fixed $j \geq i \geq 1$. If $k = 1$, then the problem can be solved in linear time by checking if there is any one vertex which is adjacent to all other vertices in the graph. For $i = 1, j \geq i$, a graph that does not have $K_{i,j}$ as a subgraph has degree at most $j-1$. Any set of k vertices in such a graph G can dominate at most $(j-1)k$ other vertices, and so G is a YES instance of k-DOMINATING SET only if G contains at most jk vertices. Thus the problem is (1) polynomial-time solvable when $k = 1$, and (2) has a linear vertex kernel when $i = 1, j \geq i$, and so in the rest of the paper we restrict our attention to the cases $k > 1, j \geq i \geq 2$.

We derive a polynomial kernel for a slightly more general, colored version of the k-DOMINATING SET problem. We define an *rwb-graph* (a red-white-blue graph) to be a graph whose vertices are (arbitrarily) colored with the three colors red, white, and blue. More precisely, an rwb-graph is a graph $G = (V, E)$ where V is partitioned into R_G, W_G, and B_G (colored red, white, and blue, respectively). An *rwb-dominating set* of an rwb-graph G is a vertex subset $S \subseteq V$ of G such that $R_G \subseteq S$ and S dominates B_G. The k-RWB-DOMINATING SET problem is defined as follows:

k-RWB-DOMINATING SET
Input: An rwb-graph $G = (V, E)$ and a non-negative integer k.
Parameter: k
Question: Does G have an rwb-dominating set with at most k vertices?

The following simple claim shows that it is sufficient to consider this colored version of the problem.

Claim 1. *Let G be a graph and H the rwb-graph obtained from G by coloring all the vertices blue. Then G has a dominating set of size at most k if and only if H has an rwb-dominating set of size at most k.*

Proof. Note that H is a copy of G with colored vertices. Let S be a dominating set of G of size at most k. Since the set R_H of red vertices of H is empty, $R_H \subseteq S$. Since H is isomorphic to G as a graph, S dominates all vertices in H. Hence S is an rwb-dominating set of H of size at most k.

Conversely, if S is an rwb-dominating set of H of size at most k, then since all vertices in H are blue, S dominates all vertices in H. S is thus a dominating set of G of size at most k. □

In our kernelization algorithm for k-DOMINATING SET, we first color all the vertices of the input graph G blue to obtain an rwb-graph H. Then we apply certain *reduction rules* to H. Roughly speaking, the reduction rules try to identify (1) vertices that must necessarily be in any rwb-dominating set of H of size at most k, and (2) vertices whose deletion from H does not affect the size of a minimal rwb-dominating set of H of size at most k.

The reduction rules also color various vertices red or white. Intuitively, the vertices colored red are those that will be picked up by the reduction rules in the rwb-dominating set D of size at most k that we are trying to construct. In particular, if there is a k-dominating set in the graph, the rules ensure that there will be one that contains all the red vertices. Vertices which are known to have been already dominated by D are colored white. Clearly all neighbors of red vertices are white, but our reduction rules color some vertices white even if they have no red neighbors (at that point). These are vertices that will be dominated by one of some constant number of vertices identified by the reduction rules[†]. The vertices that remain blue are those that are yet to be dominated.

We first describe an algorithm that takes as input an rwb-graph G on n vertices and a positive number k, and runs in $O(n^{i+O(1)})$ time. The algorithm either finds that G does not have any rwb-dominating set of size at most k, or it constructs an instance (H, k) on $O((j+1)^{i+1}k^{i^2})$ vertices such that G has an rwb-dominating set of size at most k if and only if H has an rwb-dominating set of size at most k. To complete the kernelization procedure, we show that this instance (H, k) of k-RWB-DOMINATING SET can be converted into an equivalent

[†]See reduction rule 2 for more details.

instance of k-DOMINATING SET with a polynomially bounded increase in both the number of vertices and the parameter value.

The algorithm applies a sequence of reduction rules in a specified order. The input and output of each reduction rule are rwb-graphs.

Definition 1. A graph G is said to be ***reduced*** with respect to a reduction rule if an application of the rule to G does not change G.

The correctness of the kernelization algorithm depends on the fact that each reduction rule satisfies the following correctness condition and preserves the invariants stated below:

Definition 2. *(Correctness)* A reduction rule R is said to be ***correct*** if the following condition holds: Let (G,k) be an instance of k-RWB-DOMINATING SET, and let (H,k') be the instance of k-RWB-DOMINATING SET obtained from (G,k) by one application of rule R. Then H has an rwb-dominating set D' of size at most k' if and only if G has an rwb-dominating set D of size at most k.[‡]

Invariants:

1. None of the reduction rules introduces a $K_{i,j}$ into a graph.
2. In the rwb-graphs constructed by the algorithm, no red vertex has a blue neighbor.
3. Let R_1 and R_2 be two reduction rules such that R_1 precedes R_2 in the order in which the rules are presented below. Suppose (G_1,k_1) is reduced with respect to R_1 and (G_2,k_2) is obtained by an application of rule R_2 to (G_1,k_1). Then (G_2,k_2) is reduced with respect to R_1.

2.1 The reduction rules and the kernelization algorithm

The kernelization algorithm assumes that the input graph is an rwb-graph. It applies the following rules exhaustively *in the given order*. Each rule is repeatedly applied till it causes no changes to the graph and then the next rule is applied.

We use some notational conventions in this section. For each rule below, (G,k) denotes the instance on which the rule is applied, and (H,k') the instance that is obtained when the rule is applied to (G,k). Further, D, D', k and k' are as in Definition 2: D is an rwb-dominating set of size k of G, and D' an rwb-dominating set of H of size k'.

Rule 1. Color all isolated blue vertices of G red. Set $k' := k$.

Lemma 1. *Rule 1 is correct and preserves the invariants.*

[‡]Note, however, that none of our reduction rules changes the value of k, and so $k' = k$ for every one of these rules.

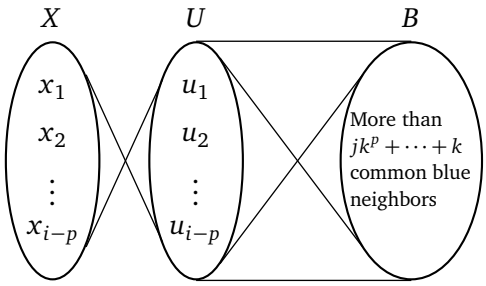

Figure 1: Rule 2

Proof. Let (G,k) be the instance on which the rule is applied, and (H,k) the resulting instance. Let I be the set of isolated blue vertices in G.

Let D be an rwb-dominating set of G of size at most k. From the definition of an rwb-dominating set, $R_G \subseteq D$. Since an isolated vertex can only be dominated by itself, $I \subseteq D$. Since the only thing that the rule does is to color isolated blue vertices of G red, $R_H = R_G \cup I$, and so $R_H \subseteq D$. Set $D' = D$ in H. Then D' dominates every vertex in H, $R_H \subseteq D'$, and $|D'| \leq k$. Thus D' is an rwb-dominating set of H of size at most k.

Conversely, let D' be an rwb-dominating set of H of size at most k. Set $D = D'$ in G. Since the only thing that the rule does is to color isolated blue vertices of G red, $R_G \subseteq R_H \subseteq D' = D$, and so D is an rwb-dominating set of G of size at most k. Thus Rule 1 is correct.

The rule trivially preserves the first two invariants, and vacuously preserves the third. □

Rule 2. For $p = 1, 2, \ldots, i-2$, in this order, apply Rule 2.p repeatedly till it no longer causes any changes in the graph.

Rule 2.p. Let $b = jk$ if $p = 1$, and $b = jk^p + k^{p-1} + k^{p-2} \cdots + k$ if $2 \leq p \leq i-2$. If a set $U = \{u_1, u_2, \ldots, u_{i-p}\}$ of $(i-p)$ vertices in G, none of which is red, has more than b **common** blue neighbors, then let B be this set of neighbors.

1. Add $(i-p)$ new (*gadget*) vertices $X = \{x_1, x_2, \ldots, x_{i-p}\}$ and all the edges $\{u,x\}; u \in U, x \in X$ to G, as in Figure 1.
2. Color all the vertices in B white.
3. Color all the vertices in X blue.
4. Set $k' := k$.

Claim 2. *Consider an instance of applying Rule* 2.$p, 1 \leq p \leq i-2$. *If U is a set of vertices of G that satisfies the condition in Rule* 2.p, *then at least one vertex in U must be in any subset of $V(G)$ of size at most k that dominates B.*

Proof. Let $p = 1$. Suppose there is a vertex set $S \subseteq V(G)$ of size at most k such that (1) S dominates B, and (2) S does not contain any vertex of U. Since $|B| > jk$, there is a vertex v in S that dominates at least $j+1$ vertices in B. Let $T = N(v) \cap B$. Then $|T| \geq j$, and the vertex sets $\{U \cup \{v\}, T\}$ form the two parts of a $K_{i,j}$ in G. This contradicts the $K_{i,j}$-free property of the input graph or the first invariant.

Now let $2 \leq p \leq (i-2)$. Let $S \subseteq V(G)$ be a set of size at most k that dominates B and does not contain any vertex of U. Since $|B| > b$, there is a vertex $v \in S$ that dominates at least $(b/k)+1$ vertices in B. Because of the second invariant, v is not red. Let $T = N(v) \cap B$. Then $|T| \geq (b/k)$, and T is in the common neighborhood of $(U \cup \{v\})$. Thus $(U \cup \{v\})$ is a set of $(i-(p-1))$ vertices in G, none of which is red, that has $b/k > jk^{p-1}+k^{p-2}+\cdots+k$ common blue neighbors. This contradicts the fact that G is reduced with respect to Rule 2.$(p-1)$. □

Proposition 1. *Rule* 2.p *is correct for* $1 \leq p \leq (i-2)$.

Proof. Let D be an rwb-dominating set of G of size at most k, and let U be as in the statement of Rule 2.p. Set $D' := D$. Since D is an rwb-dominating set of G, $R_G \subseteq D$. Since the rule does not add any new red vertices in H, $R_H \subseteq D'$. Since (1) D dominates all blue vertices of G, and (2) the rule removes no edges from G, $D' = D$ dominates all blue vertices in H that are copies of blue vertices in G. By Claim 2, $D \cap U \neq \emptyset$, and so $D' \cap U \neq \emptyset$. Since all the new blue vertices added to H — namely, those which constitute the set X — are adjacent to every vertex in U by construction, D' dominates all blue vertices in H. Thus D' is an rwb-dominating set of H of size at most k.

Conversely, let D' be an rwb-dominating set of H of size at most k. We consider three cases:

$D' \cap U = \emptyset$. In this case, since D' dominates X and X is an independent set, $X \subseteq D'$. Set $D := (D' \setminus X) \cup U$. Since X and U are disjoint sets of equal cardinality, $|D| \leq |D'| \leq k$. Since D' is an rwb-dominating set of H, $R_H \subseteq D'$. Since all vertices in X are blue and since the reduction rule does not delete any red vertex from G to obtain H, $R_G \subseteq D$. Since all the blue vertices dominated by X in H are contained in U and $U \subseteq D$, D dominates all blue vertices in G. Thus D is an rwb-dominating set of G of size at most k.

$D' \cap X = \emptyset$. In this case, since D' dominates X, it follows that $D' \cap U \neq \emptyset$. Set $D := D'$. Then $|D| = |D'| \leq k$. For the same reason as above, $R_G \subseteq D$. Since D' dominates all blue vertices in H, and since G can be obtained from H by deleting a set (namely, X) of blue vertices, $D = D'$ dominates all blue vertices in G. Thus D is an rwb-dominating set of G of size at most k.

$D' \cap U \neq \emptyset, D' \cap X \neq \emptyset$. In this case, pick an arbitrary vertex $v \in B$ and set $D := (D' \setminus X) \cup \{v\}$. Since $D' \cap X \neq \emptyset$, it follows that $|D| \leq |D'| \leq k$. For the same reason as above, $R_G \subseteq D$. Since D' dominates all blue vertices in H, $D' \setminus X$ dominates all blue vertices in $H \setminus X$, except possibly for some blue vertices in U whose only neighbors in D' belong to X. But the vertex v dominates *all* vertices in U, and so D is an rwb-dominating set of G of size at most k.

These three cases are exhaustive, and so it follows that for $1 \leq p \leq (i-2)$, Rule 2.p is correct. □

Proposition 2. *For $1 \leq p \leq (i-2)$, Rule 2.p preserves all the three invariants.*

Proof. Since we have shown that Rule 1 preserves all the invariants, we can assume inductively that all the rules that are applied before Rule 2.p preserve all the three invariants. We now consider the behaviour of Rule 2.p for each of the invariants:

Invariant 1. From the inductive assumption, and from the fact that the input graph is $K_{i,j}$-free, it follows that the graph G on which Rule 2.p is applied is $K_{i,j}$-free. Suppose the graph H resulting from the application of the rule contains a $K_{i,j}$, say K, that is introduced by the rule. Then K must necessarily contain at least one of the newly added vertices in X, or else $G = H \setminus X$ would also contain K. Since each vertex in X has degree exactly $(i-p) < i$ in H, no vertex in X can be part of a $K_{i,j}$ in H, and it follows that there is no $K_{i,j}$ in H.

Invariant 2. From the inductive assumption, the invariant holds for the graph G on which Rule 2.p is applied. The rule does not introduce new red vertices or color existing non-red vertices red. Further, it does not add new vertices as neighbors to any existing red vertex — observe that all vertices in U are non-red. Hence it follows that the rule preserves this invariant.

Invariant 3. Rule 2.1 preserves the invariant since it does not introduce isolated blue vertices into the graph. Assume inductively that for $2 \leq p \leq i-2$, Rules $2.1, \ldots, 2.(p-1)$ preserve the invariant. So the graph G on which Rule 2.p is applied is reduced with respect to Rules $1, 2.1, \ldots, 2.(p-1)$. Let H be the graph that results when Rule 2.p is applied to G. Then H is reduced with respect to Rule 1, since Rule 2.p does not introduce isolated blue vertices in H. Suppose H is *not* reduced with respect to Rule 2.q for some $1 \leq q \leq (p-1)$. Then H contains a set $U = \{u_1, u_2, \ldots, u_{i-q}\}$ of $(i-q)$ non-red vertices such that U has more than b common blue neighbors B, where $b = jk$ if $q = 1$ and $b = jk^q + k^{q-1} + k^{q-2} \cdots + k$ otherwise. Either U or B (or both) must necessarily contain at least one of the newly added vertices in X, or else $G = H \setminus X$ would also be not reduced with

respect to Rule 2.q. Note that each vertex in X has degree exactly $(i-p)$ in H. Any vertex in U has degree at least b, and any vertex in B has degree at least $(i-q)$. Since $p > q$, we have $(i-p) < (i-q)$. Since $i \leq j$ and $k \geq 1$, it follows that $(i-p) < b$. Thus no vertex in X can be part of either U or B in H, and it follows that H is reduced with respect to Rule 2.q, and so Rule 2.p preserves this invariant.

Thus the rule preserves all the three invariants. □

Putting together Propositions 1 and 2 we get

Lemma 2. *For $1 \leq p \leq (i-2)$, Rule 2.p is correct and preserves all the three invariants.*

Rule 3. If a blue or white vertex u has more than $h = jk^{i-1} + k^{i-2} + \cdots + k^2 + k$ blue neighbors, then let B be the set of blue neighbors of u.

1. Color u red.
2. Color all vertices in B white.
3. Set $k' := k$.

Claim 3. *Consider an instance of applying Rule 3. If u is a vertex of G that satisfies the condition in the rule, then u must be in any subset of $V(G)$ of size at most k that dominates B.*

Proof. Let $S \subseteq V(G)$ be a set of size at most k that dominates B. If S does not contain u, then there is a $v \in S$ that dominates at least $(h/k)+1$ of the vertices in B. The vertex v is not red (because of the second invariant), and u, v have $h/k > jk^{i-2} + k^{i-3} + \cdots + 1$ common blue neighbors, a contradiction to the fact that G is reduced with respect to Rule 2.$(i-2)$. □

Lemma 3. *Rule 3 is correct and preserves all the three invariants.*

Proof. Let D be an rwb-dominating set of G of size at most k, and let u be as in the statement of Rule 3. Set $D' := D$. Since D is an rwb-dominating set of G, $R_G \subseteq D$. From Claim 3, $u \in D$. The rule does not add any new vertex to G to get H. Since u is the only vertex that is red in H and not red in G, $R_H = (R_G \cup \{u\}) \subseteq D'$. Since (1) D dominates all blue vertices of G, (2) the rule does not add new blue vertices or make non-blue vertices blue, and (3) the rule removes no edges from the graph, $D' = D$ dominates all blue vertices in H. Thus D' is an rwb-dominating set of H of size at most k.

Conversely, let D' be an rwb-dominating set of H of size at most k. Then from Claim 3, $u \in D'$. Set $D := D'$. Since D' is an rwb-dominating set of H, $R_H \subseteq D'$.Since $R_G = R_H \setminus \{u\}$, $R_G \subseteq D$. Since (1) D' dominates all blue vertices in H, (2) the only blue vertices in G that are not blue in H are in $B \cup \{u\}$, and

(3) $u \in D$ dominates $B \cup \{u\}$, $D = D'$ dominates all blue vertices in G. Thus D is an rwb-dominating set of G of size at most k.

As for the invariants, Rule 3 does not change the structure of the graph. Further, it gives the color white to all blue neighbors of the only vertex — namely, u — whose color it changes to red. It follows that Rule 3 preserves all the three invariants. □

Rule 4. If a white vertex u is adjacent to at most one blue vertex in G, then

1. Delete u from G,
2. Set $k' := k$, and
3. Apply Rule 1.

Lemma 4. *Rule 4 is correct and preserves all the three invariants.*

Proof. Since we have already proved that Rule 1 is correct and preserves the three invariants, it suffices to show that the first two steps of Rule 4 have the stated properties[§]. We now proceed to do this; in the following, whenever we refer to Rule 4, we mean the first two steps of this rule.

Let D be an rwb-dominating set of G of size at most k, and let u be as in the statement of Rule 4. We consider two cases:

$u \notin D$. In this case, set $D' := D$. Since D is an rwb-dominating set of G, and since the rule only deletes a white vertex $u \notin D$ to obtain H, it follows that D' is an rwb-dominating set of H of size at most k.

$u \in D$. In this case, let $A = (N(u) \cap B_G)$ be the set of blue neighbors of u in G. Note that $|A| \leq 1$. Set $D' := (D \setminus \{u\}) \cup A$. Since $|A| \leq 1$, $|D'| \leq |D| \leq k$. Since D is an rwb-dominating set of G, $R_G \subseteq D$. Since the rule only deletes a white vertex u to obtain H, it follows that $R_H = R_G \subseteq D'$. Since (1) D dominates all blue vertices of G, (2) D' *contains* A, the set of all blue vertices of dominated by the vertex u that the rule deletes, and (3) the rule removes no edges from the graph other than those adjacent to the removed white vertex u, $D' = D$ dominates all blue vertices in H. Thus D' is an rwb-dominating set of H of size at most k.

Conversely, let D' be an rwb-dominating set of H of size at most k. Set $D := D'$. Since D' is an rwb-dominating set of H, and since G can be obtained from H by adding a white vertex u and some edges incident on u to H, $D = D'$ is an rwb-dominating set of G of size at most k.

As for the invariants, since Rule 4 only deletes a vertex, its application cannot introduce any of the subgraphs that make it possible to apply Rules 2 or 3 to H. It is possible that new isolated blue vertices may be introduced in

[§]Except for arguing that the rule preserves the invariants: see below.

H, but then the application of Rule 1 ensures that such vertices do not survive once Rule 4 is completely applied. It follows that Rule 4 preserves all the three invariants. □

Rule 5. If there is a white vertex u and a white or blue vertex v in G such that $N(u) \cap B_G \subseteq N(v) \cap B_G$ (that is, the blue neighborhood of u is contained in the blue neighborhood of v), then

1. Delete u from G,
2. Set $k' := k$, and
3. Apply Rule 1.

Lemma 5. *Rule 5 is correct and preserves all the three invariants.*

Proof. As in the proof of Lemma 4, it suffices to show that the first two steps of Rule 5 have the stated properties, except when arguing that the rule preserves the invariants. In the following, whenever we refer to Rule 5, we mean the first two steps of this rule. Let D be an rwb-dominating set of G of size at most k, and let u, v be as in the statement of Rule 5. We consider two cases:

$u \notin D$. In this case, set $D' := D$. Since D is an rwb-dominating set of G, and since the rule only deletes a white vertex $u \notin D$ to obtain H, it follows that D' is an rwb-dominating set of H of size at most k.

$u \in D$. In this case, set $D' := (D \setminus \{u\}) \cup \{v\}$. Then $|D'| = |D| \leq k$. Since D is an rwb-dominating set of G, $R_G \subseteq D$. Since the rule only deletes a white vertex u to obtain H, it follows that $R_H = R_G \subseteq D'$. Since (1) D dominates all blue vertices of G, (2) D' contains a vertex — namely, v — that dominates all blue vertices dominated by the vertex u that the rule deletes, and (3) the rule removes no edges from the graph other than those adjacent to the removed white vertex u, $D' = D$ dominates all blue vertices in H. Thus D' is an rwb-dominating set of H of size at most k.

Conversely, let D' be an rwb-dominating set of H of size at most k. Set $D := D'$. Since D' is an rwb-dominating set of H, and since G can be obtained from H by adding a white vertex u and some edges incident on u to H, $D = D'$ is an rwb-dominating set of G of size at most k.

Since Rule 5 only deletes a vertex, its application cannot introduce any of the subgraphs that make it possible to apply Rules 2, 3, or 4 to H. It is possible that new isolated blue vertices may be introduced in H, but then the application of Rule 1 ensures that such vertices do not survive once Rule 5 is completely applied. It follows that Rule 5 preserves all the three invariants. □

Rule 6. If the graph G contains more than k red vertices or more than $jk^i + k^{i-1} + k^{i-2} + \cdots + k^2$ blue vertices, then set (H, k') to be a trivial No-instance of the problem; for instance, make H the independent set on two vertices and set $k' = 1$. If neither of these conditions hold, set $H := G, k' := k$.

Lemma 6. *Rule 6 is correct and preserves all the three invariants.*

Proof. Note that if the instance (G,k) satisfies neither of the two conditions, then the rule returns the instance unchanged. Therefore, to show that the rule is correct, it is sufficient to show that an instance (G,k) that satisfies either of the two conditions is a No-instance.

If $|R_G| > k$, then since any rwb-dominating set of G must contain all of R_G, G has no rwb-dominating set of size at most k.

From the second invariant, no blue vertex of G has a red neighbor. Since G is reduced with respect to Rules 1 to 5, no white or blue vertex in G has more than $jk^{i-1} + k^{i-2} + \cdots + k$ blue neighbors, or else Rule 3 would have applied, contradicting the third invariant. So k white or blue vertices in G can dominate at most $jk^i + k^{i-1} + k^{i-2} + \cdots + k^2$ blue vertices. Hence if $|B_G| > jk^i + k^{i-1} + k^{i-2} + \cdots + k^2$, then no set of k vertices in G can dominate all of B_G, and so in this case G has no rwb-dominating set of size at most k.

The reduction rule either returns the instance unchanged or returns a simple No-instance. In both cases, it trivially satisfies all the three invariants. □

2.2 Algorithm correctness, running time, and kernel size

Recall that the input to the kernelization algorithm is a pair (G,k) where G is an rwb-graph and k is a non-negative integer. The algorithm applies reduction rules 1 to 6, in this order, to (G,k), exhaustively applying each rule before applying the next. From the correctness of rules 1 to 6 — see Lemma 1 to Lemma 6 — we get

Lemma 7. *The kernelization algorithm is correct: Let (G,k) be the input to the algorithm. If the algorithm says* No*, then G does not have an rwb-dominating set of size at most k. Otherwise, let H be the rwb-graph output by the algorithm. Then G has an rwb-dominating set of size at most k if and only if H has an rwb-dominating set of size at most k.*

We now show that the kernelization algorithm runs in polynomial time. To do so, we first show that the algorithm does not add too many gadget vertices to the input graph.

Claim 4. *Let (G,k) be the input to the kernelization algorithm, where G is a $K_{i,j}$-free rwb-graph on n vertices. The total number of gadget vertices that the algorithm adds to the graph, over* all *applications of rules* 2.1 *to* 2.$(i-2)$*, is less than n.*

Proof. We reuse the notation used to describe rules 2.1 to 2.$(i-2)$. Rule 2.1 colors all vertices in B white, and adds the new all-blue gadget vertex set X to the graph. The set B contains at least $(jk+1)$ blue vertices, and the set X has exactly $(i-1)$ blue vertices. Thus one application of Rule 2.1 reduces the count of blue vertices in the graph by at least $(jk-i+2)$. By a similar argument, we

can see that for $2 \leq p \leq i-2$, each application of Rule 2.p reduces the count of blue vertices by at least $(jk^p + k^{p-1} + \cdots + k - i + p + 1)$. Since, by assumption, $j \geq i \geq 2$ and $k \geq 2$, it follows that $(jk - i + 2) \leq (jk^p + k^{p-1} + \cdots + k - i + p + 1)$ for $2 \leq p \leq i-2$. Thus any one application of one of the rules 2.1 to 2.$(i-2)$ reduces the total number of blue vertices in the graph by at least $(jk - i + 2)$. Also, observe that the number of gadget vertices added to the graph in any one application of one of the rules 2.1 to 2.$(i-2)$ is at most $(i-1)$, where this maximum is attained for Rule 2.1.

Since the rwb-graph given as input to the kernelization algorithm has exactly n blue vertices, rules 2.1 to 2.$(i-2)$ can be applied at most $n/(jk - i + 2)$ times in total, over the full course of the algorithm. So the total number of gadget vertices added to the graph over all applications of rules 2.1 to 2.$(i-2)$ is $n(i-1)/(jk + 2 - i)$, and this number is less than n since we assume that k is at least 2. □

This bound on the number of gadget vertices helps in showing that the algorithm runs in polynomial time.

Lemma 8. *The kernelization algorithm can be implemented in such a way as to run in $O\left(\max(n^4, i^2 n^i)\right)$ time when the input instance is a $K_{i,j}$-free rwb-graph G on n vertices.*

Proof. We assume that the input graph is given as an adjacency list, with a provision for coloring each vertex red, white, or blue. Observe that from Claim 4 it follows that the total number of vertices in the graph at any point during the execution of the algorithm does not exceed $2n$.

Rule 1, Rule 3 Each application of one of these two rules colors at least one blue vertex red. No reduction rule changes the color of a red vertex. From the bound on the total number of vertices in the graph, it follows that Rule 1 and Rule 3 can each be applied at most $2n$ times. One application of each of these two rules can be done in $O(n)$ time by a constant number of scans of the vertex list in the input graph, and so both these rule can be applied exhaustively in $O(n^2)$ time.

Rule 2 Consider an application of Rule 2.p for some $1 \leq p \leq i-2$. A blue or white vertex can be part of a set U as mentioned in the rule only if it has at least $jk^p + k^{p-1} + k^{p-2} + \cdots + k$ blue neighbors. Since the maximum number of blue neighbors that a gadget vertex can have is $i-1$, it follows that no gadget vertex need ever be considered for inclusion in the set U in any application of Rule 2.p.

For applying Rule 2.p for a fixed $1 \leq p \leq i-2$, therefore, the algorithm iterates over all $(i-p)$-subsets of the set of all *original* (not gadget) vertices which are blue or white (at this point). This can be done in $O(\binom{n}{i-p})$

Algorithm 1 Rule 2.p: Finding the set of common blue neighbors of a vertex subset S.

```
1: A ← A binary array with indices ranging from 1 to |V(G)|, initialized to all 0s.
2: x ← A vertex in S.
3: for Each vertex y in the adjacency list of x do
4:     if y is blue then
5:         A[y] ← 1
6:     end if
7: end for
8: for Each vertex z ∈ S \ {x} do
9:     for Each vertex y in the adjacency list of z do
10:        if y is not blue then
11:            A[y] ← 0
12:        end if
13:    end for
14: end for
15: return {v ∈ V(G); A[v] = 1}
```

time, as was first shown by Ehrlich [12]. For each such subset S, the algorithm finds the set of common blue neighbors of S as in Algorithm 1. Since the total number of possible blue vertices — including gadget vertices — is at most $2n$ (see Claim 4), this can be done in time $O((i-p)n)$. A straightforward implementation of the remaining part of Rule 2.p runs in $O(n+(i-p)^2)$ time, and so Rule 2.p can be exhaustively applied in $\binom{n}{i-p} \cdot O((i-p)n+n+(i-p)^2) = O(in^{i-p+1})$ time. All the rules 2.p; $1 \leq p \leq i-2$ can therefore be exhaustively applied in $O(i^2n^i)$ time.

Rule 4 Each application of this rule deletes at least one white vertex. From the bound on the total number of vertices in the graph, it follows that the rule can be applied at most $2n$ times. One application of the rule can be done in $O(n^2)$ time, since it essentially consists of the deletion of one vertex from the graph. So the rule can be applied exhaustively in $O(n^3)$ time.

Rule 5 For one pair $\{u,v\}$ of vertices as in the statement of the rule, we can check in $O(n)$ time whether $N(u) \cap B_G \subseteq N(v) \cap B_G$, as shown in Algorithm 2. Since the number of possible choices for the pair $\{u,v\}$ is $O(n^2)$, one application of this rule can be done in $O(n^3)$ time. Since each application of the rule deletes one vertex, and the total number of vertices is bounded by $2n$, the rule can exhaustively be applied in $O\left(n^4\right)$ time.

Putting all these together, the kernelization algorithm can be implemented to run in $O\left(\max(n^4, i^2n^i)\right)$ time. □

Now we prove a polynomial bound on the size of the reduced instance.

Lemma 9. *Let (G,k) be the input to the kernelization algorithm. If the algorithm outputs the instance (H,k), then $|V(H)| = O((j+1)^{i+1}k^{i^2})$.*

Algorithm 2 An $O(n)$-time implementation of the check in Rule 5.

```
1: A ← A binary array with indices ranging from 1 to |V(G)|, initialized to all 0s.
2: for Each vertex x in the adjacency list of v do
3:     if x is blue then
4:         A[x] ← 1
5:     end if
6: end for
7: for Each vertex x in the adjacency list of u do
8:     if x is blue then
9:         if A[x] ≠ 1 then
10:            return FALSE
11:        end if
12:    end if
13: end for
14: return TRUE
```

Proof. From Rule 6, we get $|R_H| \leq k$ and $b = |B_H| \leq jk^i + k^{i-1} + \cdots + k \leq (j+1)k^i$. Now we bound $|W_H|$. Note that no two white vertices in H can have identical blue neighborhoods, or else Rule 5 would have applied. Also, each white vertex has at least two blue neighbors, or else Rule 4 would have applied. Hence the number of white vertices in H that have less than i blue neighbors is at most $\binom{b}{2} + \binom{b}{3} + \cdots + \binom{b}{i-1} \leq 2b^{i-1}$. No set of i blue vertices in H has more than $(j-1)$ common white neighbors, or else these form a $K_{i,j}$. Hence the number of white vertices that have i or more blue neighbors in H is at most $\binom{b}{i}(j-1) \leq (j-1)b^i$. So the total number of white vertices in H,

$$\begin{aligned} |W_H| &\leq 2b^{i-1} + (j-1)b^i \\ &= (2 + (j-1)b)b^{i-1} \\ &\leq (j+1)b^i \\ &\leq (j+1)((j+1)k^i)^i \\ &= (j+1)^{i+1}k^{i^2} \end{aligned}$$

The bound in the lemma follows. □

From lemmas 8 and 9 we get

Corollary 1. *For any fixed $j \geq i \geq 1$, the k-RWB-DOMINATING SET problem on $K_{i,j}$-free graphs has a polynomial kernel with $O((j+1)^{i+1}k^{i^2})$ vertices.*

2.3 Removing the colors

By Claim 1, the k-RWB-DOMINATING SET problem is a more general version of the k-DOMINATING SET problem. By Corollary 1, k-RWB-DOMINATING SET has a polynomial kernel in $K_{i,j}$-free graphs, and therefore, intuitively, so should k-DOMINATING SET. This intuition is in fact justified, because the notion of k-RWB-DOMINATING SET being "more general" than k-DOMINATING SET captures

the fact that there is a "nice" polynomial-time many-to-one reduction from k-DOMINATING SET to k-RWB-DOMINATING SET. To be more precise:

- By Claim 1, k-DOMINATING SET polynomial-time reduces to k-RWB-DOMINATING SET, and the reduction *preserves the parameter* – k goes to k.
- k-RWB-DOMINATING SET is in NP — a solution by itself is a certificate which is verifiable in polynomial time.
- k-DOMINATING SET is NP-hard in $K_{i,j}$-free graphs, since it is NP-hard in planar graphs[18] and planar graphs are $K_{3,3}$-free.

Therefore, to obtain a polynomial kernel for k-DOMINATING SET in $K_{i,j}$-free graphs, one can do the following:

- Use Claim 1 to reduce k-DOMINATING SET to k-RWB-DOMINATING SET in polynomial time, preserving the parameter.
- Use Corollary 1 to obtain a polynomial kernel for the k-RWB-DOMINATING SET instance obtained in the previous step.
- Apply the polynomial-time many-to-one reduction from k-RWB-DOMINATING SET to k-DOMINATING SET in $K_{i,j}$-free graphs, to the k-RWB-DOMINATING SET kernel obtained in the above step.

Since the k-RWB-DOMINATING SET kernel is of polynomial size in the original parameter k, and the last reduction runs in polynomial time, the resulting k-DOMINATING SET instance has size, and hence parameter, polynomial in k. This argument shows that k-DOMINATING SET has a polynomial kernel when restricted to $K_{i,j}$-free graphs, but does not give an explicit bound on the kernel size. We now describe a specific polynomial-time many-to-one reduction from k-RWB-DOMINATING SET to k-DOMINATING SET in $K_{i,j}$-free graphs and derive a concrete upper bound on the size of the kernel.

Let (G,k) be an instance of the k-DOMINATING SET problem on $K_{i,j}$-free graphs. To obtain a polynomial kernel for this instance, we first color all the vertices of G blue to get an equivalent instance of the k-RWB-DOMINATING SET problem. Then we apply Corollary 1 on this k-RWB-DOMINATING SET instance to obtain a reduced instance (G',k) of the problem, where $|V(G')| = O((j+1)^{i+1}k^{i^2})$. We then apply the steps in Algorithm 3 to transform the reduced colored instance (G',k) to an instance $(H, k-|R_{G'}|+|W_{G'}|)$ of (uncolored) k-DOMINATING SET.

We delete all the red vertices since they have no blue neighbors. We use the extra vertices to capture the fact that the white vertices need not be dominated. More precisely, suppose G' has an rwb-dominating set of size at most k. Then, since no red vertex dominates a blue vertex in G', there is a set S of at most $(k-|R_{G'}|)$ white/blue vertices that dominate all the blue vertices in G'. Let T be the set of new vertices in H that are adjacent to the white vertices of G'

Algorithm 3 Reduction from k-RWB-DOMINATING SET to k-DOMINATING SET.

1: Delete all red vertices in G'.
2: **for** Each white vertex x in G' **do**
3: Add a new vertex v_x.
4: Add an edge from x to v_x.
5: Add $k - |R_{G'}| + |W_{G'}| + 1$ new vertices of degree exactly 1, each adjacent to v_x.
6: **end for**

— that is, T is the set of all vertices added in line 3 of Algorithm 3. Then T dominates in H all the vertices that are white in G', and also all the new vertices added by the construction. Thus $S \cup T$ is a dominating set of H, of size at most $k - |R_{G'}| + |W_{G'}|$.

Conversely, suppose H has a dominating set X of size at most $k - |R_{G'}| + |W_{G'}|$. As above, let T be the set of new vertices in H that are adjacent to the white vertices of G'. Let $x \in T$, and let F_x be the set of all "new" neighbors of x — that is, F_x is the set of all neighbors of x vertices added in line 5 of Algorithm 3. Each vertex in F_x is dominated only by itself and by x, and so if $x \notin X$, then $F_x \subseteq X$. But this latter containment cannot happen because $|F_x| = (k - |R_{G'}| + |W_{G'}| + 1) > |X|$, and so $T \subseteq X$. Since $|T| = |W_{G'}|$ and T dominates $W_{G'}$ in H, this means that there exists a set Y of at most $k - |R_{G'}|$ vertices in H that (1) are white or blue in G', and (2) dominate all vertices that are blue in G'. $Y \cup R_{G'}$ is thus an rwb-dominating set of G' of size at most k.

Thus the above reduction from k-RWB-DOMINATING SET to k-DOMINATING SET is sound, and so from Corollary 1 we have

Theorem 1. *For any fixed $j \geq i \geq 1$, the k-DOMINATING SET problem on $K_{i,j}$-free graphs has a polynomial kernel with $O((j+1)^{2(i+1)}k^{2i^2})$ vertices.*

3 A Polynomial Kernel for d-degenerate Graphs

A d-degenerate graph does not contain $K_{d+1,d+1}$ as a subgraph, and so the kernelization algorithm of the previous section can be applied to a d-degenerate graph, setting $i = j = d+1$. The algorithm runs in time $O((d+1)^2 n^{d+O(1)})$ and constructs a kernel with $O((d+2)^{2(d+2)} \cdot k^{2(d+1)^2})$ vertices. Since a d-degenerate graph on v vertices has at most dv edges, we have:

Corollary 2. *The k-DOMINATING SET problem on d-degenerate graphs has a kernel on $O((d+2)^{2(d+2)} \cdot k^{2(d+1)^2})$ vertices and edges.*

Corollary 2 settles an open problem posed by Alon and Gutner [3].

3.1 Improving the running time

We describe a modification to our algorithm that reduces the running time to $O(2^d \cdot dn^2)$ when the input is restricted to d-degenerate graphs; the bound on

Algorithm 4 A faster implementation of Rule 2.p in d-degenerate graphs.

```
1: for l ← 1 to n do
2:     if v_l is blue and its degree in G[v_{l+1}, ..., v_n] is at least d − p + 1 then
3:         Find the neighborhood N of v_l in G[v_{l+1}, ..., v_n]
4:         for each (d − p + 1)-subset S of N do
5:             if S has more than (d + 1)k^p + k^{p−1} + ··· + k common blue neighbors in G then
6:                 Apply the three steps of Rule 2.p, taking S as U
7:             end if
8:         end for
9:     end if
10: end for
```

the kernel size remains the same. The modified algorithm makes use of the following well-known property of d-degenerate graphs:

Fact 1. *[17, Theorem 2.10] Let G be a d-degenerate graph on n vertices. Then one can compute, in $O(dn)$ time, an ordering $v_1, v_2, \ldots, v_n$ of the vertices of G such that for $1 \leq i \leq n$, v_i has at most d neighbors in the subgraph of G induced on $\{v_{i+1}, \ldots, v_n\}$.*

The modification to the algorithm pertains to the way rules 2.1 to 2.$(d-1)$ are implemented: the rest of the algorithm remains the same.

In implementing Rule 2.$p, 1 \leq p \leq (d-1)$, instead of checking each $(d-p+1)$-subset of vertices in the graph to see if it satisfies the condition in the rule, we make use of Fact 1 to quickly find such a set of vertices, if it exists. Let G be the graph instance on n vertices on which Rule 2.p is to be applied. First we delete, temporarily, all the red vertices in G. We then find an ordering $v_1, v_2, \ldots, v_n$ of the kind described in Fact 1, of all the remaining vertices in G. Let U and B be as defined in the rule. Since each vertex in U has degree greater than d, the first vertex v_l in $U \cup B$ that appears in the ordering has to be from B. The vertex v_l will then have a neighborhood of size $d-p+1$ that in turn has B as its common neighborhood. We use this fact to look for such a pair (U, B) and exhaustively apply Rule 2.p to G; see Algorithm 4. We then add back the red vertices that we deleted prior to this step, along with all their edges to the rest of the graph.

As $|N| \leq d$, the inner **for** loop is executed at most $\binom{d}{p-1}$ times for each iteration of the outer loop. Each of the individual steps in the algorithm can be done in $O(dn)$ time, and so Rule 2.p can be applied in $O(dn\sum_{l=1}^{n}\binom{d}{p-1})$ time. All the rules 2.p can therefore be applied in $O(dn\sum_{l=1}^{n}\sum_{p=1}^{d-1}\binom{d}{p-1}) = O(2^d \cdot dn^2)$ time. Thus we have:

Theorem 2. *For any fixed $d \geq 1$, the k-Dominating Set problem on d-degenerate graphs has a kernel on $O((d+2)^{2(d+2)} \cdot k^{(2(d+1))^2})$ vertices and edges, and this kernel can be found in $O(2^d \cdot dn^2)$ time for an input graph on n vertices.*

4 Independent Dominating Set in $K_{i,j}$-free graphs

The k-Independent Dominating Set problem asks, for a graph G and a positive integer k given as inputs, whether G has a dominating set S of size at most k such that S is an independent set in G(that is, no two vertices in S are adjacent in G). This problem is known to be NP-hard for general graphs [18], and the problem parameterized by k is $W[2]$-complete [11]. Using a modified version of the set of reduction rules in Section 2 we show that k-Independent Dominating Set has a polynomial kernel in $K_{i,j}$-free graphs for $j \geq i \geq 1$. For $i = 1, j \geq 1$ we can easily obtain trivial kernels as before, and for $i = 2, j \geq 2$ a simplified version of the following algorithm gives a kernel of size $O(j^3k^4)$.

4.1 The reduction rules

Rule 1 is the same as for the k-Dominating Set kernel for $K_{i,j}$-free graphs (Section 2.1). Rules 2.1 to 2.$(i-2)$ and Rule 3 are modified to make use of the fact that we are looking for a dominating set that is independent. A vertex u that is made white will never be part of the independent dominating set D that is sought to be constructed by the algorithm, since u is adjacent to some vertex $v \in D$. So a vertex can be deleted as soon as it is made white. Also, rules $1, 2.1 \ldots 2.(i-2)$ and 3 are the only rules. Rules 4 and 5 from that section do not apply, because of the same reason as above. The modified rules ensure that no vertex is colored white, and so they work on *rb-graphs*: graphs whose vertex set is partitioned into red and blue vertices. Using these modified rules, the bounds of $|R_H|$ and $|B_H|$ in the proof of Lemma 9, and the fact that there are no white vertices, we have

Theorem 3. *For any fixed $j \geq i \geq 1$, the k-Independent Dominating Set problem on $K_{i,j}$-free graphs has a polynomial kernel with $O(jk^i)$ vertices.*

For d-degenerate graphs, we have $i = j = d + 1$, and therefore we have:

Corollary 3. *For any fixed $d \geq 1$, the k-Independent Dominating Set problem on d-degenerate graphs has a polynomial kernel with $O((d+1)k^{d+1})$ vertices and edges.*

5 Conclusion

We derive a polynomial kernel for the k-Dominating Set problem on graphs that do not have $K_{i,j}$ as a subgraph, for any fixed $j \geq i \geq 1$. We use this result to show that the k-Dominating Set problem has a polynomial kernel of size $O((d+2)^{2(d+2)} \cdot k^{2(d+1)^2})$ on graphs of degeneracy at most d, thereby settling an open problem posed by Alon and Gutner [3]. A modified version of our kernelization algorithm for k-Dominating Set yields a (smaller) kernel for the k-Independent Dominating Set problem on $K_{i,j}$-free and d-degenerate graphs,

as well. All our kernelization algorithms are based on simple reduction rules that look at the common neighborhoods of sets of vertices.

By extending the kernel lower-bound techniques of Bodlaender et al. [4], Dom et al. [10] have shown that the k-DOMINATING SET problem on d-degenerate graphs does not have a kernel of size polynomial in *both* d and k unless the Polynomial Hierarchy collapses to the third level. This shows that it is unlikely that the kernel size that we have obtained for this class of graphs can be significantly improved.

Many interesting classes of graphs are of bounded degeneracy. These include all nontrivial minor-closed families of graphs such as planar graphs, graphs of bounded genus, graphs of bounded treewidth, and graphs excluding a fixed minor, and some non-minor-closed families such as graphs of bounded degree. Graphs of degeneracy d are $K_{d+1,d+1}$-free. Since any $K_{i,j}; j \geq i \geq 2$ contains a 4-cycle, every graph of girth 5 is $K_{i,j}$-free. Sachs [27, Theorem 1] showed that there exist graphs of girth 5 and arbitrarily large degeneracy. Therefore, $K_{i,j}$-free graphs are strictly more general than graphs of bounded degeneracy. To the best of our knowledge, $K_{i,j}$-free graphs form the largest class of graphs for which FPT algorithms and polynomial kernels are known for the dominating set problem variants discussed in this paper.

One interesting direction of future work is to try to demonstrate (no) kernels of size $f(d)\cdot k^c$ for the k-DOMINATING SET problem on d-degenerate graphs, where c is independent of d. Note that the result of Dom et al. mentioned above does *not* suggest that such kernels are unlikely. A *graph property* Π is a set of graphs. The *vertex deletion problem for* Π asks, given a graph G and a non-negative integer k as inputs, whether there exist at most k vertices in G whose deletion from G results in a graph that belongs to Π. A graph property Π is said to be (1) *nontrivial* if neither Π nor its complement is finite, and (2) *hereditary* if $(G \in \Pi, H$ is a subgraph of $G) \implies H \in \Pi$. Dell and van Melkebeek [8] have recently developed a lower-bound technique which allows them to show, *inter alia*, that the vertex deletion problem for any nontrivial hereditary graph class has no kernel of size $O(k^{2-\epsilon})$ for any $\epsilon > 0$. It will be interesting to see if this new machinery can be used to show that the k-DOMINATING SET problem does not have kernels of size $f(d)\cdot k^c$ on d-degenerate graphs.

Another challenge is to improve the running times of the kernelization algorithms: to remove the exponential dependence on d of the running time for d-degenerate graphs, and to get a running time of the form $O(n^c)$ for $K_{i,j}$-free graphs where c is independent of i and j. It would also be interesting to see if other NP-hard variants of k-DOMINATING SET — of which there are many[23, 22] — have FPT algorithms and polynomial kernels $K_{i,j}$-free graphs and graphs of bounded degeneracy. Very recently, Cygan et al.[6] showed that k-CONNECTED DOMINATING SET, where one asks for a dominating set of size at most k that induces a connected subgraph of the input graph, has no polynomial kernels in graphs of degeneracy $d \geq 2$ unless the Polynomial Hierarchy collapses to the third level. Note that our kernelization procedure breaks down

(as it should) when we insist that the solution be connected, since rules 4 and 5 can no longer be applied: white vertices which are useless in dominating blue vertices may still be useful in providing connectivity for the dominating set that is being constructed.

Acknowledgments. We thank Aravind Natarajan for pointing out the connection between $K_{i,j}$-free and d-degenerate graphs, and Shai Gutner for his comments on an earlier draft of this paper.

References

[1] ALBER, J., FELLOWS, M. R., AND NIEDERMEIER, R. Polynomial-time data reduction for Dominating Set. *Journal of the ACM 51*, 3 (2004), 363–384.

[2] ALON, N., AND GUTNER, S. Linear Time Algorithms for Finding a Dominating Set of Fixed Size in Degenerated Graphs. In *Computing and Combinatorics, 13th Annual International Conference, COCOON 2007, Banff, Canada, July 16-19, 2007, Proceedings* (2007), G. Lin, Ed., vol. 4598 of *Lecture Notes in Computer Science*, Springer, pp. 394–405.

[3] ALON, N., AND GUTNER, S. Kernels for the Dominating Set Problem on Graphs with an Excluded Minor. Technical Report TR08-066, The Electronic Colloquium on Computational Complexity (ECCC), 2008. A revised version titled "Polynomial kernels and faster algorithms for the dominating set problem on graphs with an excluded minor", by S. Gutner, has been accepted at IWPEC 2009.

[4] BODLAENDER, H. L., DOWNEY, R. G., FELLOWS, M., AND HERMELIN, D. On Problems without Polynomial Kernels (Extended Abstract). In *Proceedings of ICALP 2008* (2008), LNCS, Springer, pp. 563–574.

[5] CHEN, J., FERNAU, H., KANJ, I. A., AND XIA, G. Parametric Duality and Kernelization: Lower Bounds and Upper Bounds on Kernel Size. *SIAM Journal of Computing 37*, 4 (2007), 1077–1106.

[6] CYGAN, M., PILIPCZUK, M., PILIPCZUK, M., AND WOJTASZCZYK, J. Kernelization hardness of connectivity problems in d-degenerate graphs. Accepted at WG 2010. Available at http://wg2010.thilikos.info/booklet.

[7] DAWAR, A., AND KREUTZER, S. Domination problems in nowhere-dense classes. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2009)* (Dagstuhl, Germany, 2009), R. Kannan and K. N. Kumar, Eds., vol. 4 of *Leibniz International Proceedings in Informatics (LIPIcs)*, Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, pp. 157–168.

[8] DELL, H., AND VAN MELKEBEEK, D. Satisfiability allows no nontrivial sparsification unless the polynomial-time hierarchy collapses. In *Proceedings of the 42nd ACM Symposium on Theory of Computing (STOC 2010)* (2010). To appear.

[9] DIESTEL, R. *Graph Theory*, third ed. Springer-Verlag, Heidelberg, 2005.

[10] DOM, M., LOKSHTANOV, D., AND SAURABH, S. Incompressibility through Colors and IDs. In *Proceedings of ICALP 2009* (2009), vol. 5555 of *LNCS*, Springer, pp. 378–389.

[11] DOWNEY, R. G., AND FELLOWS, M. R. *Parameterized Complexity*. Springer, 1999.

[12] EHRLICH, G. Loopless algorithms for generating permutations, combinations, and other combinatorial configurations. *Journal of the ACM 20*, 3 (1973), 500–513.

[13] ELLIS, J. A., FAN, H., AND FELLOWS, M. R. The Dominating Set problem is fixed parameter tractable for graphs of bounded genus. *Journal of Algorithms 52*, 2 (2004), 152–168.

[14] FLUM, J., AND GROHE, M. *Parameterized Complexity Theory*. Springer-Verlag, 2006.

[15] FOMIN, F. V., AND THILIKOS, D. M. Fast Parameterized Algorithms for Graphs on Surfaces: Linear Kernel and Exponential Speed-Up. In *Automata, Languages and Programming: 31st International Colloquium, ICALP 2004, Turku, Finland, July 12-16, 2004. Proceedings* (2004), vol. 3142 of *Lecture Notes in Computer Science*, Springer, pp. 581–592.

[16] FOMIN, F. V., AND THILIKOS, D. M. Dominating Sets in Planar Graphs: Branch-Width and Exponential Speed-Up. *SIAM Journal of Computing 36*, 2 (2006), 281–309.

[17] FRANCESCHINI, G., LUCCIO, F., AND PAGLI, L. Dense trees: a new look at degenerate graphs. *Journal of Discrete Algorithms 4* (2006), 455–474.

[18] GAREY, M. R., AND JOHNSON, D. S. *Computers and Intractability: A Guide to the Theory ofNP–Completeness*. Freeman, San Francisco, 1979.

[19] GOLOVACH, P. A., AND VILLANGER, Y. Parameterized Complexity for Domination Problems on Degenerate Graphs. In *Proceedings of the 34th International Workshop on Graph-Theoretic Concepts in Computer Science, WG 2008* (2008), vol. 5344 of *Lecture Notes in Computer Science*, Springer.

[20] GUO, J., AND NIEDERMEIER, R. Invitation to data reduction and problem kernelization. *SIGACT News 38*, 1 (2007), 31–45.

[21] GUTNER, S. Polynomial kernels and faster algorithms for the dominating set problem on graphs with an excluded minor. In *Parameterized and Exact Computation: 4th International Workshop, IWPEC 2009, Copenhagen, Denmark, September 10-11, 2009, Revised Selected Papers* (Berlin, Heidelberg, 2009), Springer-Verlag, pp. 246–257.

[22] HAYNES, T. W., HEDETNIEMI, S. T., AND SLATER, P. J. *Domination in graphs: advanced topics*, vol. 209 of *Pure and Applied Mathematics : A Series of Monographs and Textbooks*. Marcel Dekker, Inc, 1998.

[23] HAYNES, T. W., HEDETNIEMI, S. T., AND SLATER, P. J. *Fundamentals of Domination in Graphs*, vol. 208 of *Pure and Applied Mathematics : A Series of Monographs and Textbooks*. CRC Press, 1998.

[24] NIEDERMEIER, R. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press, 2006.

[25] PHILIP, G., RAMAN, V., AND SIKDAR, S. Solving Dominating Set in Larger Classes of Graphs: FPT Algorithms and Polynomial Kernels. In *Algorithms - ESA 2009, 17th Annual European Symposium, Copenhagen, Denmark, September 7-9, 2009. Proceedings* (2009), A. Fiat and P. Sanders, Eds., vol. 5757 of *Lecture Notes in Computer Science*, pp. 694–705.

[26] RAMAN, V., AND SAURABH, S. Short Cycles Make W-hard Problems Hard: FPT Algorithms for W-hard Problems in Graphs with no Short Cycles. *Algorithmica 52*, 2 (2008), 203–225.

[27] SACHS, H. Regular graphs with given girth and restricted circuits. *Journal of the London Mathematical Society s1-38*, 1 (1963), 423–429.