EDGE DOMINATING SET:
Efficient Enumeration-Based Exact Algorithms

Henning Fernau[1,2]

[1] Univ.Trier, FB 4—Abteilung Informatik, 54286 Trier, Germany
[2] Univ.Tübingen, WSI für Informatik, Sand 13, 72076 Tübingen, Germany
`fernau@informatik.uni-trier.de`

Abstract. We analyze EDGE DOMINATING SET from a parameterized perspective. More specifically, we prove that this problem is in $\mathcal{FPT}$ for general (weighted) graphs. The corresponding algorithms rely on enumeration techniques. In particular, we show how the use of compact representations may speed up the decision algorithm.

1 Introduction

Graphs and line graphs. It is a common observation that problems that are hard for general graphs become easier when considered on *line graphs*, i.e., graphs whose adjacency relation can be thought of as originating from the edge-to-edge neighborhood of another graph. More specifically, if $G = (V, E)$ is some graph, then its line graph $L(G)$ has E has the set of "vertices," and there is an "edge" (in $L(G)$) between $e_1, e_2 \in E$ if e_1 and e_2 share a common endpoint in G. For example, while VERTEX COVER is $\mathcal{NP}$-complete on general graphs, it can be solved in polynomial time on line graphs, since this corresponds to the EDGE COVER problem. The same comment applies to the more general problem WEIGHTED VERTEX COVER. However, DOMINATING SET remains $\mathcal{NP}$-complete even when restricted to line graphs, see [23], even when restricted to planar cubic graphs [15]. Does this mean that there is actually "no difference" between general graphs and line graphs with respect to DOMINATING SET? This is in fact the case from a classical perspective, but the picture changes when one considers two prominent approaches of how to deal with computationally hard problems:

- while DOMINATING SET is hard to approximate on general graphs [7] (it cannot be approximated better than $\ln n$ unless $\mathcal{NP} \subseteq \mathcal{DTIME}(n^{\ln \ln n})$), EDGE DOMINATING SET is constant-factor approximable (also in the weighted case), see [3,13,18]; moreover, it is $\mathcal{MAXSNP}$-hard and hence there is no polynomial-time approximation scheme to be expected [3,23];
- while DOMINATING SET is W[2]-hard on general graphs, EDGE DOMINATING SET is in $\mathcal{FPT}$; in this paper, we are going to show that EDGE DOMINATING SET, when parameterized by the number of elements k in the dominating set, can be solved in time $\mathcal{O}^*(2.62^k)$.
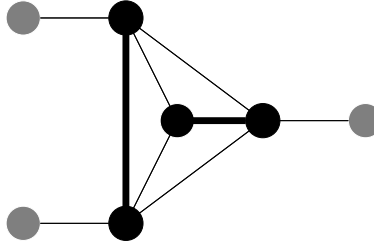
Fig. 1. The two thick-line edges form an edge dominating set

Problem statements. Let us formulate the problem dealt with in this paper without reference to line graphs. An *edge dominating set* of a graph is a subset D of edges such that each edge is either in D or incident to an edge in D. An instance of EDGE DOMINATING SET (EDS) is given by a graph $G = (V, E)$, and the parameter, a positive integer k. The task is: Is there an *edge dominating set* $D \subseteq E$ with $|D| \leq k$? Consider Fig. 1 for an illustration; the two thick edges form an edge dominating set. We will also consider a weighted variant: An instance of WEIGHTED EDGE DOMINATING SET (WEDS) is given by a graph $G = (V, E)$ with edge weights $\omega : E \to \mathbb{R}_{\geq 1}$, and a positive integer k (the parameter). We ask: Is there an edge dominating set $D \subseteq E$ with $\omega(D) \leq k$?

We will analyze these problems in the framework of *parameterized complexity* [6]. A *parameterized problem* P is a subset of $\Sigma^* \times \mathbb{N}$, where Σ is a fixed alphabet and $\mathbb{N}$ is the set of all non-negative integers. Therefore, each instance of the parameterized problem P is a pair (I, k), where the second component k is called the *parameter*. The language $L(P)$ is the set of all YES-instances of P. We say that the parameterized problem P is *fixed-parameter tractable* [6] if there is an algorithm that decides whether an input (I, k) is a member of $L(P)$ in time $f(k)|I|^c$, where c is a fixed constant and $f(k)$ is a function independent of the overall input length $|I|$. The class of all fixed-parameter tractable problems is denoted by $\mathcal{FPT}$. We will make use of the $\mathcal{O}^*$-notation that has now become standard in exact algorithmics: in contrast to the better known $\mathcal{O}$-notation, it not only suppresses constants but also polynomial parts of the run time denoted this way. We are dealing with (sometimes weighted) undirected (hyper-)graphs throughout this paper and use according standard notations.

Results. In previous work, it was claimed that EDGE DOMINATING SET on bipartite graph (this problem is also known as MATRIX DOMINATION SET) belongs to $\mathcal{FPT}$. More precisely, this was posed as an exercise on the kernelization chapter in [6]. Rather recently, Weston [22] exhibited how to obtain a kernel of exponential size. Moreover, he shows a (not quite convincing) enumeration-based (this is our interpretation of the paper) search tree approach. E. Prieto in her PhD thesis [21] obtained a kernel of quadratic size for MINIMUM MAXIMAL MATCHING (i.e., INDEPENDENT DOMINATING SET in line graphs) on general graphs, a problem that is basically the same as EDGE DOMINATING SET, see [3,17,23].

The contributions of this paper are the following ones: (1) We generalize the enumeration-based algorithm for MATRIX DOMINATION SET to EDGE DOMINATING SET. (2) We further show how these techniques may also apply to the weighted case; notice that the close relation between MINIMUM MAXIMAL MATCHING and EDGE DOMINATING SET is lost in the weighted case. (3) Incidentally, we also present a search tree algorithm for WEIGHTED HITTING SET, parameterized by the number of edges. (4) We show how enumeration-based algorithms that are usually deemed to be quite inefficient can be sped up by exploiting that not all minimal vertex covers (in our case) need to be "completely" enumerated. Rather, certain non-determinism can be left to the following computation phase in the leaves of the search tree. (5) We show that auxiliary vertex cover structures can be quite useful for other algorithmic problems. The speed-up technique mentioned in (4) can then be formalized as *compact representations* of vertex covers. This can be seen as extensions of ideas presented by Damaschke [4]. (6) We also use the mentioned vertex cover structures to obtain quadratic kernels for the considered problems.

2   Relating to VERTEX COVER and to WEIGHTED HITTING SET

First, we observe that to every EDGE DOMINATING SET instance $G = (V, E)$ with a solution $D \subseteq E$ of size at most k, there corresponds a vertex cover of size at most $2k$: take all vertices incident to the at most k edges: $C_D := \bigcup_{e \in D} e$.[1] This observation is also true for WEIGHTED EDGE DOMINATING SET: by our problem definitions, a solution of weight at most k can contain at most k edges. Conversely, any vertex cover C can be extended to some edge set D with $C \subseteq C_D$ with $|C_D| \leq 2|C|$; since C is a vertex cover, D will be an edge dominating set.

This gives the following idea: (1) we cycle through all minimal vertex covers C of size up to $2k$ of the given graph $G = (V, E)$; (2) we use this additional structure to solve the problem of finding an edge dominating set D of size at most k (or weight at most k) that contains all vertices of C, i.e., $C \subseteq C_D$.

As to step (1), it is known how to list all minimal vertex covers up to size $2k$ in time $\mathcal{O}^*(4^k)$, see [4]. To see that the second step works, consider the following auxiliary hypergraph $G' = (V', E')$: G' contains the edges E of G as its vertices, i.e., $V' = E$, and for every x contained in the cover C, we introduce a hyperedge h_x that contains all edges of G that are incident with x. Hence, $|E'| \leq |C| \leq 2k$.

Fomin, Kratsch and Woeginger [12] recently came up with an efficient parameterized algorithm for the following problem: An instance of MINIMUM HITTING SET, PARAMETERIZED BY # EDGES (HSE) is given by a hypergraph $G = (V, E)$, and the parameter, $|E|$. The task is: Find a minimum hitting set $C \subseteq V$!

We will generalize that algorithm in the following so that it can also cope with WEIGHTED HITTING SET. The algorithm uses a technique known as *dynamic programming on subsets*. To this end, given a hypergraph $G = (V, E)$ with $V = \{v_1, \ldots, v_n\}$, the algorithm maintains a 2-dimensional array F that contains, for

---

[1] This observation was also the basis of a first, simple factor-4 approximation algorithm for MINIMUM EDGE DOMINATING SET presented in [3].

Algorithm 1. A dynamic programming algorithm for MINIMUM HITTING SET, PARAMETERIZED BY # EDGES, called HSE

Input(s): a hypergraph $G = (V, E)$ with vertex weights ω, $V = \{v_1, \ldots, v_n\}$
Output(s): a hitting set $C \subset V$ of minimum weight

  for all $E' \subseteq E$ **do**
    $F[E, 0] = \infty$
  end for
  $F[\emptyset, 0] = 0$
  for $j = 1, \ldots, n$ **do**
    for all $E' \subseteq E$ **do**
      Let $E'' := \{e \in E' \mid v_j \in e\}$.
      $F[E', j] := \min\{F[E', j-1], F[E' \setminus E'', j-1] + w(v_j)\}$
      {Two cases arise: either v_j is not belonging to a minimum hitting set for E',
      then, $F[E', j] = F[E', j-1]$; or v_j belongs to a minimum hitting set C for E',
      but then, $C \setminus \{v_j\}$ is a minimum hitting set for $E' \setminus E''$ relative to V_{j-1}, so that
      $F[E', j] = F[E' \setminus E'', j-1] + w(v_j)$.}
    end for
  end for

$E' \subseteq E$ and for $j = 0, \ldots, n$, in $F[E', j]$ the minimum weight of a subset C of $V_j := \{v_1, \ldots, v_j\}$ that covers E' (C is also called a *hitting set for E'* (*relative to* V_j)); if no such cover exists, set $F[E', j] = \infty$. More details on how to construct the entries for F are contained in Alg. 1. There, also the basic reasoning for the inductive step in the correctness proof of that algorithm is given.

Theorem 1. MINIMUM HITTING SET, PARAMETERIZED BY # EDGES *can be solved in time* $\mathcal{O}^*(2^{|E|})$ *on a hypergraph* $G = (V, E)$ *with vertex weights* ω.

Corollary 1. WEIGHTED EDGE DOMINATING SET *can be solved in time* $\mathcal{O}^*(16^k)$.

3   Replacing the Hitting Set Phase

In this section, we are going to explain that the second HITTING SET phase can be replaced by a polynomial-time computation, which already considerably improves on the run time stated in Cor. 1.

    If we have a *minimal* vertex cover C of size at most $2k$, we first compute a maximum matching M in the induced graph $G[C]$. Clearly, $|M| \leq k$. There might be a set $C' \subseteq C$ of vertices not matched by M. Since M is maximal, no two vertices of C' are neighbors. Hence, for all $x \in C'$, we can take any edge incident with x into the edge dominating set to be constructed.

Theorem 2. *Alg. 2 runs in time* $\mathcal{O}^*(4^k)$ *and solves* EDGE DOMINATING SET.

Proof.   The run time is dominated by the enumeration of at most 4^k minimal vertex covers of size $2k$, see [4]. The correctness is based on the fact that the following problem can be solved in polynomial time by matching techniques: An

Algorithm 2. EDS-enum: An enumeration-based search tree algorithm for EDS

Input(s): a graph $G = (V, E)$, a positive integer k
Output(s): if possible: a subset $D \subset E$, $|D| \leq k$, that dominates all edges or
  NO if no such set exists.

  Create a list L of minimal vertex covers C of G with $|C| \leq 2k$, see [4].
  {This hides the search tree part.}
  for all $C \in L$ **do**
    Set $D :=$GECmatch(G, C)
    if $|D| \leq k$ **then**
      return D
    end if
  end for
  return NO

---

instance of GENERALIZED EDGE COVER (GEC) is given by a graph $G = (V, E)$ together with a set of red vertices $R \subseteq V$. The task is to construct a minimum set $C \subseteq E$ of edges that cover all vertices from R. Namely, a maximum matching of $G[R]$ plus adding edges to cover the hitherto unmatched vertices will do. This describes the procedure GECmatch that gets as arguments the graph and the set of red vertices. Moreover, if R is a vertex cover, then a GEC solution will be an edge dominating set. ∎

Remark 1. Observe the crucial difference between the weighted and the unweighted case here. For example, notice that the thick edges in Fig. 1 (that form a minimum edge dominating set) can be obtained by computing a maximum matching from a vertex cover formed by the black vertices (or from a minimal vertex cover formed by the "outermost" three black vertices). However, if the edges that connect the outermost black vertices with the black center vertex have edge weight one and all other edges weight three, then those three edges with weight one will form a minimum edge dominating set, in there is only one minimum edge dominating set in this weighted graph. However, starting with a minimum matching in the graph induced by the outermost three black vertices will never find the mentioned minimum edge dominating set, since some edge that connects the outermost black vertices will be contained in such a solution.

It is known (see [19]) that MINIMUM WEIGHTED EDGE COVER can be optimally solved in polynomial (cubic) time. Based on this result, Plesník gave a cubic time algorithm for a generalization of the problem we are interested in, see [20]; below, we give a shorter construction for a cubic time algorithm for GENERALIZED MINIMUM WEIGHTED EDGE COVER, directly based on [19].

So, let $G = (V, E)$ be a graph with edge weight function ω and with a set R of distinguished vertices. The task is to find a minimum weight edge set that covers R. As explained above, we cannot restrict our attention to $G[R] = (R, E_R)$. In addition, we consider the following edges. For each $x \in R$, let e_x be the edge incident with x that has lowest weight amongst all edges incident with x and not contained in E_R. If all edges incident with x are contained in E_R, then e_x is

undefined. So, we add all such edges e_x to $G[R]$ to obtain a graph $G' = (V', E')$ with $R \subseteq V'$ and $E_R \subseteq E'$. Now, we identify all vertices in $V' \setminus R$ to get one new vertex r, so that we obtain a graph $G'' = (V'', E'')$ with $V'' = R \cup \{r\}$. The edge weight function is accordingly adapted and also called ω. Let f be the function that maps edges from E' onto edges from E'' (by the described identification).

Proposition 1. (G, ω, R) *has a minimum generalized weighted edge cover of weight* ω_{opt} *if and only if either* $(G[R], \omega)$ *or* (G'', ω) *has a minimum weighted edge cover of weight* ω_{opt}.

Proof.   Let C be a minimum generalized weighted edge cover covering R. Without loss of generality, edges from C that are not contained in $G[R]$ could be the edges of the form e_x chosen as described above. If C contains no edges of the form e_x, then C is also a minimum weighted edge cover for $G[R]$. If C contains edges of the form e_x, then $f(C)$ is a minimum weighted edge cover for G''.

Conversely, a set C that is a feasible edge cover for $G[R]$ is also a feasible generalized edge cover for (G, R). Similarly, an edge set C (containing no edges outside of E') such that $f(C)$ is a feasible edge cover of G'' is also a feasible generalized edge cover for (G, R). Taking the cover of lowest weight ensures to produce a minimum generalized weighted edge cover for (G, ω, R).   ∎

Hence, Theorem 2 is also valid in the weighted case.

4   A Closer Look at the Search Tree

Can we further improve on the running time of Theorem 2 ? To this end, reconsider the main idea behind the enumeration of all minimal vertex covers C of size up to $2k$: starting from such a cover, we are going to construct a corresponding edge dominating set D of minimum cardinality among all edge dominating sets D' with $C \subseteq C_{D'}$.

It is usually a good strategy to look at small-degree vertices to find nice branching scenarios (or kernelizations). Assume that we do not branch at vertices of degree one. If ℓ is a bound on the vertex cover size, this gives the recurrence

$$T(\ell) \leq T(\ell - 1) + T(\ell - 2)$$

for the running time, i.e., $T(\ell) \leq 1.6181^\ell$, i.e., with $\ell = 2k$, 2.6181^k is an upper-bound for the run time in this case.[2]

So, after branching we are left with a set of vertices C that covers all vertices of the originally given graph $G = (V, E)$ but an edge set E' that is the set of edges of $G[V \setminus C]$ that has maximum degree one. Now, we can produce a corresponding edge dominating set for G from this structure by finding an edge set D with the following properties:

---

[2] The related so-called enumerate-and-expand speed-up technique was independently developed by Mölle, Richter and Rossmanith (see Proc. CSR and COCOON, appearing in 2006), worked out with the example of CONNECTED VERTEX COVER.

- every $v \in C$ is covered by some $e \in D$, i.e., $C \subseteq C_D$;
- every $e' \in E'$ is dominated by some $e \in D$, i.e., $\forall e' \in E' \exists e \in D : e' \cap e \neq \emptyset$.

We will call a D satisfying both properties (C, E')-satisfying. Coming back to our HITTING SET model from the beginning, one can see that these properties can be modeled by aiming at constructing a minimum hitting set for a hypergraph whose vertex set is E and which has the following edges:

- for every $v \in C$, introduce the hyperedge $\{\{u, v\} \mid u \in N(v)\}$;
- for every $e' \in E'$, introduce the hyperedge $\{e \mid e \cap e' \neq \emptyset\}$.

Notice that there cannot be more than $2k$ hyperedges in the constructed HITTING SET instance (unless we face a NO-instance), since in the previously described complete vertex enumeration scenario, each of the edges from E' would have been "resolved" by a further branching step.

We can go one step further; namely, notice that we can ignore the possibility to include any $e' \in E'$ into the corresponding edge dominating set we construct, as long as there is any edge $e \in E$ with $|e \cap e'| = 1$. To exclude that special case, we employ in the very beginning of the algorithm (before even starting the vertex cover enumeration phase) the following reduction rule as long as possible:

Reduction rule 1. *(isolated edges) Let $(G = (V, E), k)$ be an instance of* EDGE DOMINATING SET. *If $e \in E$ such that $\forall \hat{e} \in E : \hat{e} \cap e \neq \emptyset \Rightarrow \hat{e} = e$, then delete e from the graph instance and decrease the parameter k by one.*

After having performed the vertex cover enumeration phase (without branching at vertices of degree one) on such a reduced instance G (without isolated edges), we have arrived (at each leave of the search tree) at a partial cover set C plus the above-mentioned edge set E'. Now, we form a new graph G' from G by contracting all edges from E'. Moreover, let M be the set of $|E'|$ vertices obtained by merging endpoints of edges from E'. Let $C' = C \cup M$. We claim that D is a minimum edge dominating set for G that is (C, E')-satisfying if and only if there is a minimum general edge cover D' for G' (with red vertex set C') with $|D'| = |D|$. Namely, if D' is a general edge cover D' for G' covering all vertices from C', then after "unmerging" we recover the graph $G = (V, E)$ in which we can view the edges from D' as elements from E. Then, D' is a (C, E')-satisfying edge set that is an edge dominating set according to our previous reasoning. Conversely, if D is a (C, E')-satisfying edge dominating set, then first we can transform D into a (C, E')-satisfying edge dominating set D' with $D' \cap E' = \emptyset$; namely, since G contains no isolated edges, we can replace any $e \in D \cap E'$ by some (arbitrarily chosen) incident edge e'. The edge set D' constructed this way can be interpreted as an edge set of G', and now D' (with $|D'| = |D|$ if we assume minimality of D) is a general edge cover for G'.

Theorem 3. EDGE DOMINATING SET *can be solved in time* $\mathcal{O}^*((2.6181)^k)$.

For MINIMUM MAXIMAL MATCHING we can conclude (based on [3,17,23]):

Corollary 2. *The problem* MMM *can be solved in time* $\mathcal{O}^*((2.6181)^k)$.

Algorithm 3. Reducing MATRIX DOMINATION SET to EDGE DOMINATING SET.

Input(s): a matrix instance (M, k) of MATRIX DOMINATION SET.
Output(s): a graph instance (G, k) of EDGE DOMINATING SET such that (M, k) is a
  YES-instance iff (G, k) is a YES-instance.

  Let C be the set of columns of M.
  Let R be the set of rows of M.
  Form the vertex set $V = C \cup R$ of $G = (V, E)$.
  for all $i \in R$, $j \in C$ **do**
    Put $\{i, j\} \in E$ iff entry (i, j) of M is one.
  end for

The sketched unweighted case can be transferred to the weighted case. More
precisely, the yet uncovered edges (left over from the vertex cover enumeration
phase) can be modelled by first introducing a fresh vertex u and connecting u
to all vertices $[v, w]$ obtained by merging v and w as described above. The new
edge $\{u, [v, w]\}$ will get the same weight as the former edge $\{v, w\}$ had, and all
other edge weights will stay the same.

Corollary 3. *The problem* WEDS *can be solved in time* $\mathcal{O}^*((2.6181)^k)$.

Let us finally consider a related problem, also mentioned in [23]: An instance
of MATRIX DOMINATION SET (MDS) is given by an $n \times n$ matrix with entries
from $\{0, 1\}$, and a positive integer k (the parameter). We ask: Is there a set D of
one-entries in the matrix, where $|D| \leq k$, such that every other one-entry has at
least one row or one column in common with some one-entry from D? Observe
that this problem can be also seen as a chess piece domination problem: interpret
the matrix as a chessboard showing places where it is allowed to place a rook or
where not (by having a one- or a zero-entry in the corresponding position).

Lemma 1 (Yannakakis/Gavril). MATRIX DOMINATION SET *can be reduced
(via* $\mathcal{FPT}$ *reduction) to* EDGE DOMINATING SET.

The corresponding reduction is formulated in Alg. 3, hence making explicit the
remark in [8, p. 249] that MDS can be solved via EDS. In [6, Exercise 3.2.9],
solving MATRIX DOMINATION SET by means of a kernelization and search tree
based algorithm is proposed as an exercise. [3]

  Hence, MDS is in one-to-one correspondence to EDS, restricted to bipartite
graphs. Since our solution of EDGE DOMINATING SET is based on VERTEX COVER,
and the latter (in its decision version!) is known to be easier on bipartite graph,
the following corollary might see some improvements; however, we did not man-
age to get improvements in a straightforward manner, since we are rather relying
on the enumeration than on the decision version of VC.

Corollary 4. MATRIX DOMINATION SET *can be solved in time* $\mathcal{O}^*((2.6181)^k)$.

---

[3] In [22], a $\mathcal{O}^*(c^k)$ algorithm is proposed for MATRIX DOMINATION SET with $c < 2$; how-
  ever, this is based on a wrong interpretation of our results on CONSTRAINT BIPARTITE
  VERTEX COVER as detailed in [11].

5   Compact Representations

We have shown that auxiliary vertex cover structures can be quite useful to solve EDGE DOMINATING SET and related problems. The speed-up described in the previous section can be also interpreted as enumerating compact representations of vertex covers that can be formalized similar to regular expressions.

1. $\emptyset$ is an expression denoting a compact representation that denotes no sets at all, i.e., $C(\emptyset) = \{\emptyset\}$.
2. If a is a vertex, then a is an (atomic) compact representation of the cover collection $C(a)$ only containing the cover $\{a\}$, i.e., $C(a) = \{\{a\}\}$.
3. If $e = \{a, b\}$ is an edge, then $\hat{e}$ is an (atomic) compact representation of the cover collection $C(\hat{e})$ only containing the covers $\{a\}$ and $\{b\}$,
4. If A and B are compact representations that represent cover collections $C(A)$ and $C(B)$, resp., then $A + B$ represents the cover collections

$$C(A + B) = \{X \cup Y \mid X \in C(A), Y \in C(B)\}.$$

5. If A and B are compact representations that represent cover collections $C(A)$ and $C(B)$, then $A \cup B$ represents the cover collection $C(A \cup B) = C(A) \cup C(B)$.
6. Nothing else are compact representations.

Example 1. For example, the minimal vertex covers of the graph

$$(\{1, \ldots, k\} \times \{1, 2\}, \{\{(i, 1), (i, 2)\} \mid 1 \le i \le k\})$$

can be written as

$$\{(1, \widehat{1), (1}, 2)\} + \{(2, \widehat{1), (2}, 2)\} + \cdots + \{(k, \widehat{1), (k}, 2)\}.$$

For instance, if $k = 3$,

$$\begin{aligned}
&\{(1, \widehat{1), (1}, 2)\} + \{(2, \widehat{1), (2}, 2)\} + \{(3, \widehat{1), (3}, 2)\} \\
&= \{\{(1, 1)\}, \{(1, 2)\}\} + \{\{(2, 1)\}, \{(2, 2)\}\} + \{\{(3, 1)\}, \{(3, 2)\}\} \\
&= \{\{(1, i), (2, j), (3, \ell)\} \mid 1 \le i, j, \ell \le 2\}
\end{aligned}$$

Theorem 4. *Representations of all minimal vertex covers of size up to k (and possibly some more non-minimal cover representations) can be listed in time $\mathcal{O}^*(1.6181^k)$.*

Proof.   (Sketch) The usual enumeration algorithm for listing all minimal vertex covers up to size k can be modified by avoiding branches at vertices of degree one. After this branching, the remaining graph has maximum degree one, and the cover of an edge $e = \{x, y\}$ can be described by $\hat{e}$.   ∎

The difference to the results of Damaschke [4] is that he insists on enumerating only minimal vertex covers up to size k; hence, his running times are worse. Since

vertex cover structure already found many applications, we believe that such representations (or possibly similar ones) may yield interesting improvements in the development of exact graph algorithms. For example, in [10], the problem of finding a *total vertex cover* of size up to k was discussed, where a vertex cover C is called total if every $x \in C$ satisfies $N(x) \cap C \neq \emptyset$. By a vertex cover enumeration phase yielding covers C in the leaves, followed by a hitting set phase (with hyperedge set $\{N(x) \mid x \in C\}$), this problem can be solved in time $\mathcal{O}^*(4^k)$. With compact representations, this can be readily improved to $\mathcal{O}^*(3.2361^k)$: for $\hat{e}$ in the compact representation of some covers with $e = \{x, y\}$, we can introduce a hyperedge $N(x) \cup N(y)$. Choosing $z \in N(x) \cup N(y)$ also determines whether x or y is put into the cover. However, different techniques allowed to further lower the constants for that problem to $\mathcal{O}^*(2.3655^k)$ in [10].

6   Kernels

We have solved the problems dealt with in this paper by a search-tree technique based on enumerating minimal vertex covers. This approach can be also used to show quadratic kernels for all these problems. Namely, as explained in [9], Buss' kernelization rules are also valid for the enumeration task. Hence, we can assume that the reduced graph has no more than $2(2k)^2 = 8k^2$ vertices in the enumeration phase, where k is the parameter of the say EDS instance. We can turn this into a kernel for EDS by the following observations: (a) We keep each vertex in the vertex cover enumeration kernel in the EDS kernel. (b) For each vertex v that was put into each vertex cover by Buss' rule, we have to put v plus an arbitrary neighbor u of v that is not yet in the EDS kernel, provided that v does not have already neighbors in the EDS kernel. This possibly complicated-looking rule allows to cover v by some edge also in the reduced instance. Since the number of vertices added by this special treatment of vertices put into the vertex cover by Buss' rule is smaller than the quantity in the general case, we get an upper bound of $8k^2$ vertices for the number of vertices in the EDS kernel.

Lemma 2. *Given an instance* (G, k) *of* EDGE DOMINATING SET*, it is possible to produce a kernel* (G', k') *of* EDS *with* $|V(G')| \leq 8k^2$ *and* $k' \leq k$*. Similar results are true for* WEDS*,* MMM*, and* MDS*.*

Notice that a kernel of size $4k(k + 2)$ was obtained by Prieto for MMM by adapting crown reduction techniques, see [21]. We can improve on the kernel size for EDS at the cost of introducing annotations (marking vertices that should go into the vertex cover):[4] (a) isolated edges are to be put into the edge dominating set anyways; (b) vertices of degree two with two neighbors of degree one comprise a component that can be solved by arbitrarily taking one of either edges of the component into the dominating set; (c) vertices of degree one that are not covered by rules (a) and (b) need not be put into the vertex covers considered in the

---

[4] The notion of *annotated kernel* is further discussed in joined work of Abu-Khzwam and Fernau, also contained in these proceedings.

enumeration phase but rather their unique neighbors; this is reflected by deleting the degree-one vertex and by marking its unique neighbor. By (c), also marked vertices have minimum degree of two. Hence, the kernel (possibly containing some marked vertices doomed to go into the vertex cover) contains at most $4k^2$ vertices: Buss' rule yields that there are at most $(2k)^2$ many edges in the graph instance, and knowing that both marked and unmarked vertices have minimum degree of two means: there are also at most $(2k)^2$ many vertices in the graph.

7   Conclusions and Open Problems

We have shown how ideas stemming from the area of parameterized enumeration can be useful to obtain efficient parameterized algorithms for decision problems. More precisely, we derived an $\mathcal{O}^*(2.62^k)$ algorithms for EDGE DOMINATING SET and many variants. It would be interesting to see if these constants could be further improved. In particular, it might be possible to avoid branching at degree-two vertices in the enumeration phase, as argued before Theorem 3 for degree-one vertices. Notice that in terms of approximation factors, there seems to be no difference between VERTEX COVER, TOTAL VERTEX COVER, EDGE DOMINATING SET, and FEEDBACK VERTEX SET; however, in terms of search tree based parameterized algorithms, VERTEX COVER appears to be the easiest of the three, while FEEDBACK VERTEX SET seems to be the hardest one, see [1,5,14,18].

It would be interesting to see if and how the ideas presented in this paper can be applied to solve WEIGHTED MINIMUM MAXIMAL MATCHING. Notice that the main problem is to allow Alg. 1 to cope with the additional independence condition. In the literature, several other variants of EDGE DOMINATING SET have been considered that might deserve further studies from the viewpoint of parameterized complexity; recent papers are [2,16].

References

1. V. Bafna, P. Berman, and T. Fujito. A 2-approximation algorithm for the undirected feedback vertex set problem. *SIAM Journal of Discrete Mathematics*, 12:289–297, 1999.
2. A. Berger and O. Parekh. Linear time algorithms for generalized edge dominating set problems. Technical Report TR-2005-002-A, Emory University Math/CS department, 2005.
3. R. Carr, T. Fujito, G. Konjevod, and O. Parekh. A 2 1/10 approximation algorithm for a generalization of the weighted edge-dominating set problem. *Journal of Combinatorial Optimization*, 5:317–326, 2001.
4. P. Damaschke. Parameterized enumeration, transversals, and imperfect phylogeny reconstruction. In R. Downey, M. Fellows, and F. Dehne, editors, *International Workshop on Parameterized and Exact Computation IWPEC 2004*, volume 3162 of *LNCS*, pages 1–12. Springer, 2004.

5. F. Dehne, M. R. Fellows, M. Langston, F. Rosamond, and K. Stevens. An $o(2^{O(k)}n^3)$ fpt algorithm for the undirected feedback vertex set problem. In *Proc. 11th International Computing and Combinatorics Conference COCOON*, volume 3595 of *LNCS*, pages 859–869. Springer, 2005.

6. R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer, 1999.

7. U. Feige. A threshold of $\ln n$ for approximating set cover. *Journal of the ACM*, 45:634–652, 1998.

8. M. R. Fellows, C. McCartin, F. A. Rosamond, and U. Stege. Coordinatized kernels and catalytic reductions: an improved FPT algorithm for Max Leaf Spanning Tree and other problems. In S. Kapoor and S. Prasad, editors, *FST TCS 2000*, volume 1974 of *LNCS*, pages 240–251. Springer, 2000.

9. H. Fernau. On parameterized enumeration. In O. H. Ibarra and L. Zhang, editors, *Computing and Combinatorics, Proceedings COCOON 2002*, volume 2383 of *LNCS*, pages 564–573. Springer, 2002.

10. H. Fernau and D. F. Manlove. Vertex and edge covers with clustering properties: Complexity and algorithms. Technical Report TR-2006-210, Department of Computing Science DCS Glasgow, UK, April 2006. Available through: `http://www.dcs.gla.ac.uk/publications/paperdetails.cfm?id=8137`.

11. H. Fernau and R. Niedermeier. An efficient exact algorithm for constraint bipartite vertex cover. *Journal of Algorithms*, 38(2):374–410, 2001.

12. F. Fomin, D. Kratsch, and G. Woeginger. Exact (exponential) algorithms for the dominating set problem. In J. Hromkovic et al., editors, *30th International Workshop on Graph-Theoretic Concepts in Computer Science WG 2004*, volume 3353 of *LNCS*, pages 245–256. Springer, 2004.

13. T. Fujito and H. Nagamochi. A 2-approximation algorithm for the minimum weight edge dominating set problem. *Discrete Applied Mathematics*, 118:199–207, 2002.

14. J. Guo, J. Gramm, F. Hüffner, R. Niedermeier, and S. Wernicke. Improved fixed-parameter algorithms for two feedback set problems. In *Proceedings of the 9th Workshop on Algorithms and Data Structures WADS*, volume 3608 of *LNCS*, pages 158–168. Springer, 2005.

15. J. D. Horton and K. Kilakos. Minimum edge dominating sets. *SIAM Journal of Discrete Mathematics*, 6:375–387, 1993.

16. C. L. Lu, M.-T. Ko, and C. Y. Tang. Perfect edge domination and efficient edge domination in graphs. *Discrete Applied Mathematics*, 119(3):227–250, 2002.

17. D. F. Manlove. *Minimaximal and maximinimal optimisation problems: a partial order-based approach*. PhD thesis, University of Glasgow, Computing Science, 1998.

18. O. Parekh. Edge domination and hypomatchable sets. In *Symposium on Discrete Algorithms SODA 2002*, pages 287–291. ACM Press, 2002.

19. J. Plesník. Equivalence between the minimum covering problem and the maximum matching problem. *Discrete Mathematics*, 49:315–317, 1984.

20. J. Plesník. Constrained weighted matchings and edge coverings in graphs. *Discrete Applied Mathematics*, 92:229–241, 1999.

21. E. Prieto. *Systematic Kernelization in FPT Algorithm Design*. PhD thesis, The University of Newcastle, Australia, 2005.

22. M. Weston. A fixed-parameter tractable algorithm for matrix domination. *Information Processing Letters*, 90:267–272, 2004.

23. M. Yannakakis and F. Gavril. Edge dominating sets in graphs. *SIAM Journal of Applied Mathematics*, 38(3):364–372, June 1980.