Theoretical Computer Science 565 (2015) 1–15

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

On the parameterized complexity of vertex cover and edge cover with connectivity constraints [☆]

Henning Fernau [a], Fedor V. Fomin [b], Geevarghese Philip [d,*], Saket Saurabh [b,c]

[a] *Universität Trier FB 4, Abteilung Informatik, 54286 Trier, Germany*
[b] *Department of Informatics, University of Bergen, 5020 Bergen, Norway*
[c] *The Institute of Mathematical Sciences, Chennai 600 113, India*
[d] *Max-Planck-Institut für Informatik, 66123 Saarbrücken, Germany*

ARTICLE INFO

Article history:
Received 21 October 2013
Received in revised form 30 September 2014
Accepted 21 October 2014
Available online 28 October 2014
Communicated by G. Ausiello

Keywords:
Parameterized complexity
Connectivity problems
Kernel lower bounds
Vertex cover
Edge cover

ABSTRACT

We investigate the effect of certain natural connectivity constraints on the parameterized complexity of two fundamental graph covering problems, namely VERTEX COVER and EDGE COVER. Specifically, we impose the additional requirement that each connected component of a solution have at least t vertices (resp. edges from the solution) for a fixed positive integer t, and call the problem t-TOTAL VERTEX COVER (resp. t-TOTAL EDGE COVER). In both cases the parameter k is the size of the solution. We show that

- both problems remain fixed-parameter tractable with these restrictions, with running times of the form $\mathcal{O}^\star(c^k)$ for some constant $c > 0$ in each case, where the $\mathcal{O}^\star$ notation hides polynomial factors;
- for each fixed $t \geq 2$, t-TOTAL VERTEX COVER has no polynomial kernel unless CoNP $\subseteq$ NP/poly;
- for each fixed $t \geq 2$, t-TOTAL EDGE COVER has a linear vertex kernel of size $\frac{t+1}{t}k$.

These results significantly improve earlier work on these problems. We illustrate the utility of the technique used to solve t-TOTAL VERTEX COVER, by applying it to derive an $\mathcal{O}^\star(c^k)$-time FPT algorithm for the t-TOTAL EDGE DOMINATING SET problem.
Our no-poly-kernel result for t-TOTAL VERTEX COVER, and the known NP-hardness result for t-TOTAL EDGE COVER, are in stark contrast to the fact that VERTEX COVER has a $2k$ vertex kernel, and that EDGE COVER is solvable in polynomial time. This illustrates how even the slightest connectivity requirement results in a drastic change in the tractability of problems—the curse of connectivity!

[☆] A preliminary version of this paper appeared in the proceedings of COCOON 2010 [1]. In addition to the detailed proofs which were omitted in that version, the current article also includes an FPT algorithm for the t-TOTAL EDGE DOMINATING SET problem which was not discussed in the shorter version.
[*] Corresponding author.
E-mail addresses: fernau@uni-trier.de (H. Fernau), fomin@ii.uib.no (F.V. Fomin), gphilip@mpi-inf.mpg.de (G. Philip), saket@imsc.res.in, saket@ii.uib.no (S. Saurabh).

http://dx.doi.org/10.1016/j.tcs.2014.10.035
0304-3975/© 2014 Elsevier B.V. All rights reserved.

1. Introduction

Given a graph G and a positive integer k as input, the Vertex Cover problem asks whether G has a set of at most k vertices—a *vertex cover* of G—such that every edge of G has at least one of these vertices as an end-point. The Edge Cover problem is quite similar: given (G, k) as input, the question here is whether G has a set of at most k *edges*—an *edge cover* of G—such that every vertex in G is an end-point of at least one of these edges.

Vertex Cover is a classical NP-hard problem [2] whose parameterized version with k as the parameter is arguably the most investigated problem in parameterized algorithmics. The fastest known FPT algorithm for Vertex Cover runs in $\mathcal{O}^\star(1.2738^k)$ time [3], and the problem has a kernel with at most $2k$ vertices [4]. The parameterized complexity of Vertex Cover—with various parameterizations—continues to be actively investigated. Recent results of this nature include faster FPT algorithms for certain “above-guarantee” versions of Vertex Cover [5–7]. In contrast, the Edge Cover problem is solvable in polynomial time [8].

We investigate the parameterized complexity of variants of Vertex Cover and Edge Cover where additional connectivity constraints are imposed on the solution set S. More specifically, for each fixed integer t; $1 \le t \le k$ we define variants of the two problems, named t-Total Vertex Cover and t-Total Edge Cover, respectively, as follows.

t-Total Vertex Cover
Input: A graph $G = (V, E)$, and a non-negative integer k.
Parameter: k
Question: Does G have a vertex cover S of size at most k such that each connected component of the subgraph of G induced by S contains at least t vertices?

t-Total Edge Cover
Input: A graph $G = (V, E)$, and a non-negative integer k.
Parameter: k
Question: Does G have an edge cover T of size at most k such that each connected component of the subgraph of G induced by T contains at least t edges?

Observe that for $t = 1$, these problems are identical to Vertex Cover and Edge Cover, respectively. Observe also that if $k < t$ then the answer is trivially No for both the problems. Hence the interesting values of t are those in the range $1 \le t \le k$. A vertex cover satisfying the conditions specified in the first problem is called a *t-total vertex cover* of the graph G; an edge cover satisfying the conditions of the second problem is called a *t-total edge cover* of G.

The underlying classical problems were first investigated by Fernau and Manlove [9], who showed that t-Total Vertex Cover is NP-hard for each fixed $t \in \mathbb{N}$; $1 \le t \le k$, and that t-Total Edge Cover is NP-hard for each fixed $t \in \mathbb{N}$; $2 \le t \le k$. They also initiated the study of the parameterized variants of these problems.

Małafiejski and Żylinski studied 2-Total Edge Cover as a model of weak cooperation of guards in an art gallery problem [10]. Both Fernau and Manlove [9] and Małafiejski and Żylinski [10] derived a Gallai type identity which says that under certain conditions, the sum of (i) the cardinality of the largest possible packing of a graph with vertex-disjoint copies of a path of length two and (ii) the size of the smallest 2-total edge cover, equals the number of vertices of the graph. Fernau and Manlove also derived a generalization of this result to every fixed integer $t \ge 2$ [9]. Combining this with the result of Kirkpatrick and Hell [11], who proved that finding a packing of vertex-disjoint copies of trees on t edges in a graph is NP-hard, they showed that t-Total Edge Cover is NP-complete for each fixed $t \ge 2$.

Besides the art gallery problem mentioned above, further motivation for studying these problems can be drawn from certain models of fault-tolerant computing [12]. These problems are interesting from the point of view of computational biology as well, due to the close relation that these problems have to variants of the so-called Test Cover Problem [13].

Fernau and Manlove [9] derived a number of results for these problems. Apart from the NP-hardness results mentioned above, they showed that for each fixed $t \in \mathbb{N}$; $2 \le t \le k$ the t-Total Vertex Cover problem has a polynomial-time 2-approximation, but cannot be approximated within a factor of $10\sqrt{5} - 21 - \epsilon$ for any $\epsilon > 0$ in polynomial time unless $P = NP$. They further showed that for each fixed $t \in \mathbb{N}$; $2 \le t \le k$ the t-Total Edge Cover problem has a polynomial-time 2-approximation, and that there exists a constant $\delta > 1$ such that 2-Total Edge Cover cannot be approximated within a factor of δ in polynomial time unless $P = NP$. As for the parameterized versions of these problems, they showed that (i) 2-Total Vertex Cover is FPT and can be solved in $\mathcal{O}^\star(2.3655^k)$ time, and that (ii) for each fixed $t \in \mathbb{N}$; $2 \le t \le k$, t-Total Edge Cover is FPT and can be solved in $\mathcal{O}^\star((2k)^{2k})$ time. They also claimed to prove that the t-Total Vertex Cover problem has a kernel of size $\mathcal{O}(k(k+t))$ for each fixed $t \in \mathbb{N}$; $2 \leq t \leq k$. However, as we show in this paper, such is not the case unless CoNP $\subseteq$ NP/poly, which is considered unlikely.[1]

1.1. *Our results*

We advance the study of the parameterized complexity of t-Total Vertex Cover and t-Total Edge Cover initiated by Fernau and Manlove [9]. We significantly improve their results and obtain several new results; in particular, we complete the picture on how even the slightest connectivity requirement dramatically changes the complexity of these problems. As noted above, Edge Cover has been known to be solvable in polynomial time for over half a century, and it was recently shown [9] that the least possible connectivity requirement on the solution set T, namely that each connected component of the graph induced by T have at least 2 edges from T, makes the problem NP-hard.

We show a similar result for t-Total Vertex Cover, not in the context of classical complexity, but within the ambit of parameterized complexity. It is a well-known result in parameterized complexity that Vertex Cover has a kernel on at most $2k$ vertices [4]. We show that adding a connectivity constraint results in a dramatic change in kernelizability: we show that for any fixed $2 \leq t \leq k$, the t-Total Vertex Cover problem has no polynomial-size kernel unless CoNP $\subseteq$ NP/poly, which is deemed unlikely in complexity theory. We complement this no-polynomial-kernel result with results on the fixed-parameter tractability of t-Total Vertex Cover and t-Total Edge Cover. Specifically, we show the following for each fixed integer $2 \leq t \leq k$:

- t-Total Vertex Cover can be solved in $\mathcal{O}(16.1^{k+\mathcal{O}(\log^2 k)} \times n^{\mathcal{O}(1)})$ time. To obtain this result we combine the classical result of Otter [14] on the number of unlabelled trees with a modification of the colour-coding technique of Alon et al. [15];
- t-Total Edge Cover has a kernel on at most $\frac{t+1}{t}k$ vertices;
- t-Total Edge Cover can be solved in $\mathcal{O}(2^{\frac{t+1}{t}k+\mathcal{O}(\sqrt{k})} \times n^{\mathcal{O}(1)})$ time. To obtain this result, we combine kernelization techniques with a classical result of Hardy and Ramanujan [16] and the Fast Fourier Transform [17].

The techniques used to derive the FPT algorithm for t-Total Vertex Cover can be used to solve the "t-total" versions of other problems which share a certain structural similarity with t-Total Vertex Cover. To illustrate this point, we use these techniques to derive an FPT algorithm for t-Total Edge Dominating Set.

1.2. *Organization of the rest of the paper*

In Section 2 we set down some basic definitions related to graph theory and parameterized complexity, and describe the results which we use to obtain kernel lower bounds. We take up the t-Total Vertex Cover problem in Section 3. In Section 3.1 we show that the problem does not admit polynomial kernels unless CoNP $\subseteq$ NP/poly. In Section 3.2 we present our FPT algorithm for the problem. In Section 4 we turn to the t-Total Edge Cover problem. We present our improved kernel for this problem in Section 4.1. In Section 4.2 we describe our FPT algorithm for the problem. We describe an FPT algorithm for t-Total Edge Dominating Set in Section 5. We conclude in Section 6.

2. Preliminaries

In this section we state some basic definitions related to graph theory and parameterized complexity, and give an overview of the notation used in this paper. We use $G = (V, E)$ to refer to a graph with vertex set V (with $n := |V|$). A graph $G' = (V', E')$ is a *subgraph* of G if $V' \subseteq V$ and $E' \subseteq E$. The subgraph G' is called an *induced subgraph* of G if $E' = \{uv \in E \mid u, v \in V'\}$. In this case, G' is also called the subgraph *induced by* V' and is denoted $G[V']$. For a set $S \subseteq E$ of edges of G, we use $V(S)$ to denote the set of all vertices which are end-points of at least one edge in S, and $G(S)$ to denote the subgraph $(V(S), S)$ of G; this is called the subgraph *induced by* S. Let X, Y be two vertex subsets of graph G. We say that the set X *dominates* the set Y if for every vertex $y \in Y$, there is a vertex $x \in X$ such that $\{x, y\}$ is an edge in graph G. A vertex subset D of graph $G = (V, E)$ is a *dominating set* of G if D dominates V.

To describe running times of algorithms we sometimes use the $\mathcal{O}^\star$ notation. Given $f : \mathbb{N} \to \mathbb{N}$, we define $\mathcal{O}^\star(f(n))$ to be $O(f(n) \cdot p(n))$, where $p(\cdot)$ is some polynomial function. That is, the $\mathcal{O}^\star$ notation suppresses polynomial factors in the expression for the running time.

[1] The flaw in the proof of Fernau and Manlove [9, Proof of Theorem 7] is in the sentence "But $|E_2| \geq |V_2|$, since each vertex in V_2 has degree at least 2". Specifically: while each vertex in the set V_2 does have degree at least 2 in the graph G', the set E_2 is the set of edges in the induced subgraph $G'[V_2]$, and it is not necessarily the case that each vertex in the set V_2 has degree at least 2 *in this subgraph*. Hence their upper bound on $|V_2|$ in terms of $|E_2|$ is fallacious. To see a concrete instance of this bug, consider an input instance $(G = (V, E), k)$ of t-Total Vertex Cover where (i) $n = |V| \gg k > 2$, (ii) $V = A \uplus B$, (iii) $|A| = k$ and $G[A]$ is connected, (iv) B is an independent set in G, and (v) every vertex in B has all of A as its neighbourhood. Then (G, k) is a Yes instance and A is a t-total vertex cover of G of size at most k. The three reduction rules in the proof of Fernau and Manlove mark all of A and none of B and set $k' = 0$, and do nothing else. Further, $V_2 = B$ and $E_2 = \emptyset$ and so the inequality $|E_2| \geq |V_2|$ does not hold, although each vertex in V_2 does have degree $k > 2$ in $G' = G$.

A parameterized problem Π is a subset of $\Sigma^* \times \mathbb{N}$, where Σ is a finite alphabet. An instance of a parameterized problem is a tuple (x, k), where k is called the parameter. A central notion in parameterized complexity is *fixed-parameter tractability (FPT)* which means, for a given instance (x, k), decidability in time $f(k) \cdot p(|x|)$, where f is an arbitrary function of k and p is a polynomial in the input size. The notion of *kernelization* is formally defined as follows.

Definition 1 *(Kernelization).* (See [18,19].) A kernelization algorithm for a parameterized problem $\Pi \subseteq \Sigma^* \times \mathbb{N}$ is an algorithm that, given $(x, k) \in \Sigma^* \times \mathbb{N}$, outputs, in time polynomial in $|x| + k$, a pair $(x', k') \in \Sigma^* \times \mathbb{N}$ such that (a) $(x, k) \in \Pi$ if and only if $(x', k') \in \Pi$ and (b) $|x'|, k' \leq g(k)$, where g is some computable function. The output instance x' is called the kernel, and the function g is referred to as the size of the kernel. If $g(k) = k^{O(1)}$ (resp. $g(k) = O(k)$) then we say that Π admits a polynomial (resp. linear) kernel.

2.1. Kernel lower bound machinery

We now describe the notions and results from the recently developed theory of kernel lower bounds [20–22] which are used to prove lower bounds on the size of kernels. We begin by associating a classical decision problem with a parameterized problem in a natural way, as follows:

Definition 2 *(Derived classical problem).* (See [21].) Let $\Pi \subseteq \Sigma^* \times \mathbb{N}$ be a parameterized problem, and let $1 \notin \Sigma$ be a new symbol. We define the *derived classical problem* associated with Π to be $\{x1^k | (x, k) \in \Pi\}$.

That is, to obtain the "unparameterized", classical version of a parameterized problem instance, we merely write the parameter out in unary.

Definition 3. (See [21].) Let P and Q be parameterized problems. We say that P is *polynomial parameter reducible* to Q, written $P \leq_{ppt} Q$, if there exists a polynomial time computable function $f : \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$ and a polynomial $p : \mathbb{N} \to \mathbb{N}$ such that for all $x \in \Sigma^*$ and $k \in \mathbb{N}$,

- $(x, k) \in P \iff f(x, k) \in Q$, and,
- $f(x, k) = (x', k') \Longrightarrow k' \leq p(k)$

We call f a polynomial parameter transformation (or a PPT) from P to Q.

The following theorem captures the reason why this notion of a reduction is useful in showing kernel lower bounds:

Theorem 1. *(See [21, Theorem 3].) Let P and Q be parameterized problems whose derived classical problems are P^c, Q^c, respectively. Let P^c be* NP*-complete, and $Q^c \in$* NP*. Suppose there exists a PPT from P to Q. Then, if Q has a polynomial kernel, then P also has a polynomial kernel.*

As a consequence, to show that the problem Q (conditionally) has no polynomial kernels, it is sufficient to show that the problem P—again, conditionally—has no polynomial kernels, and then exhibit a PPT from P to Q. Observe that this is quite similar to the way in which polynomial-time reductions are used in classical complexity to propagate NP-hardness results.

3. Computing total vertex covers

We now consider the t-Total Vertex Cover problem. The problem is NP-complete for all each fixed integral value of t; $1 \leq t \leq k$. For $t = 1$, t-Total Vertex Cover is the Vertex Cover problem, and for instances which have $t = k$ the problem becomes identical to the Connected Vertex Cover problem[2]; these are two classical NP-complete problems [23, Problem GT1]. For any fixed $t \in \mathbb{N}$; $2 \leq t \leq k$, the t-Total Vertex Cover problem has been shown to be NP-hard by a reduction from Vertex Cover [9, Theorem 3]; we give an alternate proof of NP-hardness in Proposition 1 below.

[2] It is not very rigorous to talk about setting $t = k$ in t-Total Vertex Cover, and to refer to the resulting problem as Connected Vertex Cover. This is because in t-Total Vertex Cover, t is a *fixed* number and is not part of the input (so we cannot *set* it equal to k), while in Connected Vertex Cover the number k *is* part of the input. However, every instance of t-Total Vertex Cover for which k happens to be equal to t can be thought of as an instance of Connected Vertex Cover. Hence for illustrative purposes alone, we slightly abuse the notation and terminology; we talk about setting $t = k$ in a t-Total Vertex Cover instance, and refer to the resulting problem as Connected Vertex Cover.

3.1. Kernelization complexity

Recall that when $t = 1$, t-Total Vertex Cover is just Vertex Cover, and for instances which satisfy $t = k$ it becomes identical to Connected Vertex Cover. The former problem has a vertex kernel of size at most $2k$ [4], and the latter problem does not have a polynomial kernel [22]. It turns out that this change in polynomial kernelizability occurs at the smallest possible value of t.

Theorem 2. *For each fixed $t \geq 2$, t-*Total Vertex Cover *has no kernel of size bounded by k^c, for any fixed constant c, unless CoNP $\subseteq$ NP/poly and the Polynomial Hierarchy collapses to the third level.*

As we show later in this section, the derived classical problem—the “unparameterized” version—of t-Total Vertex Cover is NP-complete for each fixed $t \in \mathbb{N}$; $2 \leq t \leq k$ (Proposition 1). To obtain our lower bound on the kernel size, we introduce an intermediate problem named Red Blue Dominating Set. As we show below, the unparameterized version of this problem is NP-complete, and it is known that the problem does not admit polynomial kernels. For each fixed t; $2 \leq t \leq k$, we give a polynomial parameter transformation from Red Blue Dominating Set to t-Total Vertex Cover, which implies, by Theorem 1, that the latter problem has no polynomial kernel. We start off by defining the intermediate problem:

Red Blue Dominating Set
Input: An undirected bipartite graph $G = (R \uplus B, E)$, and a positive integer k.
Parameter: $k + |B|$
Question: Does there exist a set $D \subseteq R$ of at most k vertices of G such that D dominates B?

The derived classical problem corresponding to Red Blue Dominating Set, defined below, is NP-complete.

Padded Red Blue Dominating Set
Input: An undirected bipartite graph $G = (R \uplus B, E)$, a positive integer k, and the number $k + |B|$ written in unary.
Question: Does there exist a set $D \subseteq R$ of at most k vertices of G such that D dominates B?

Fact 1. *(See [24].) The* Padded Red Blue Dominating Set *problem is NP-complete.*

It has been shown that the parameterized version of this problem does not admit polynomial kernels:

Fact 2. *(See [22, Theorem 2].)* Red Blue Dominating Set *does not admit a polynomial kernel unless CoNP $\subseteq$ NP/poly.*

We are now ready to show that t-Total Vertex Cover has no polynomial kernel for each fixed $t \in \mathbb{N}$; $2 \leq t \leq k$.

Proof of Theorem 2. By Fact 1, the derived classical problem corresponding to Red Blue Dominating Set is NP-complete. Also, the derived classical problem corresponding to t-Total Vertex Cover is evidently in NP. Now suppose t-Total Vertex Cover has a polynomial kernel for any fixed value of t; $2 \leq t \leq k$, and that there is a polynomial parameter transformation from Red Blue Dominating Set to t-Total Vertex Cover. Then by Theorem 1, Red Blue Dominating Set has a polynomial kernel, and hence by Fact 2 CoNP $\subseteq$ NP/poly; it follows that t-Total Vertex Cover does not have a polynomial kernel unless CoNP $\subseteq$ NP/poly. Hence to prove the theorem, it suffices to show that there is a polynomial parameter transformation from Red Blue Dominating Set to t-Total Vertex Cover for each fixed t; $2 \leq t \leq k$. We now proceed to give such a transformation.

Given an instance $(G = (R \uplus B, E), k)$ of Red Blue Dominating Set and a fixed t; $2 \leq t \leq k$, we construct an instance of t-Total Vertex Cover as follows: If B contains isolated vertices then (G, k) is a NO instance, and in this case we construct a trivial NO instance of t-Total Vertex Cover. Otherwise, we add a distinct path of length (number of edges) $t - 1$ starting from each vertex $v \in B$. Thus, if $t = 2$, then we attach a new, distinct pendant vertex w_i to each $v_i \in B$; if $t = 3$, then we add a path (v_i, u_i^1, w_i) to each $v_i \in B$. In general, for a fixed $t \in \mathbb{N}$; $2 \leq t \leq k$, we add a path $(v_i, u_i^1, u_i^2, \ldots, u_i^{t-2}, w_i)$ to each $v_i \in B$, where the vertices u_i^j and w_i are all new and distinct: see Fig. 1 for an illustration. We call the resulting graph H, and $(H, k + (t-1)|B|)$ is the constructed instance of t-Total Vertex Cover. To complete the proof, we show:

Proposition 1. *Let $(G = (R \uplus B, E), k)$ be an instance of* Red Blue Dominating Set*, and t a fixed positive integer. Let H be the graph constructed from G as described above. Then (G, k) is a YES instance of* Red Blue Dominating Set *if and only if $(H, k + (t-1)|B|)$ is a YES instance of t-*Total Vertex Cover.

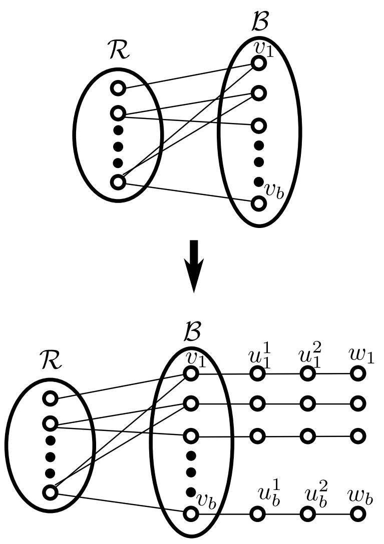

Fig. 1. Polynomial parameter transformation from Red Blue Dominating Set to t-Total Vertex Cover, for the case $t = 4$. The construction appends a path with $t - 1 = 3$ edges to each vertex in the set B; the new parameter is $k + 3|B|$.

Proof. Let $(G = (R \uplus B, E), k)$ be a YES instance of Red Blue Dominating Set. Then there is an inclusion-minimal set $D \subseteq R$, $|D| \le k$, that dominates B. Let S be the set of all new vertices added by the construction to H, *except* for the pendant vertex w_i in each new path. Thus, e.g., $S = \emptyset$ when $t = 2$, and in general $|S| = (t - 2)|B|$. Define $C = D \cup B \cup S$. Now,

1. $|C| = |D| + |B| + |S| \le k + |B| + (t - 2)|B| \le k + (t - 1)|B|$.
2. C is a vertex cover of H: $B \subseteq C$ covers all original edges, and all new edges adjacent to vertices in B; S covers the rest of the new edges, if any.
3. Each connected component of $H[C]$ contains at least t vertices:
   (a) Since D dominates B in G, any vertex $v_i \in B$ has at least one neighbour $w \in D \subseteq C$ in $H[C]$. Then v_i, w, and the $t - 2$ new vertices $\{u_i^1, u_i^2, \ldots, u_i^{t-2}\} \subseteq S$ are all part of the same component in $H[C]$, as witnessed by the path $w, v_i, u_i^1, u_i^2, \ldots, u_i^{t-2}$. Thus each vertex $v_i \in B$ is part of a connected component of size at least t in $H[C]$.
   (b) D is an inclusion-minimal dominating set of B and $D \cup B \subseteq C$, and so each $w \in D$ has at least one neighbour $v_i \in B$ in the graph $H[C]$. It follows that each vertex $w \in D$ is part of a connected component of size at least t in $H[C]$, namely the component to which v_i belongs.
   (c) Each vertex $u_i^j \in S$ is in the same component in $H[C]$ as the vertex $v_i \in B$, and so u_i^j is part of a connected component of size at least t in $H[C]$, namely the component to which v_i belongs.

Thus C is a t-total vertex cover of H, of size at most $k + (t - 1)|B|$. This proves the forward direction.

To prove the reverse direction, suppose $(H, k + (t - 1)|B|)$ is a YES instance of t-Total Vertex Cover, and let C be a t-total vertex cover of H of size at most $k + (t - 1)|B|$. Consider any path $P = (u_i^0 = v_i, u_i^1, u_i^2, \ldots, u_i^{t-2}, w_i)$ in H consisting of a vertex $v_i \in B$ and new vertices added by the construction. Since C is a vertex cover such that each connected component of $G[C]$ has at least t vertices, we have that $|P \cap C| \ge t - 1$. Now suppose there exists a vertex $x \in P \setminus C$. If $x = u_i^j$ for some $0 \le j \le t - 2$, then the vertices $u_i^{j+1}, u_i^{j+2}, \ldots, u_i^{t-2}, w_i$ form a connected component of $H[C]$ of size strictly less than t, a contradiction. So one of the following holds:

1. All the t vertices of P are in C, or,
2. $\{v_i, u_i^1, u_i^2, \ldots, u_i^{t-2}\} \in C$, and $w_i \notin C$.

Let p_t be the number of paths of the first kind in H, and let p_{t-1} be the number of paths of the second kind in H. There is exactly one such path corresponding to each vertex of B, and so $p_t + p_{t-1} = |B|$. The total number of vertices contributed to C by these paths is $tp_t + (t - 1)p_{t-1}$, and so the number of vertices in $C \cap R$ is at most $k + (t - 1)|B| - (tp_t + (t - 1)p_{t-1}) = k - p_t$. Now let P be a path of the second kind. By definition, $|P \cap C| = (t - 1)$, and since each connected component of $G[C]$ has at least t vertices, there is at least one more vertex x in C that is adjacent in H to one of the vertices, say y, of $P \cap C$. The only possibility is that $x \in R$ and $y \in B$, and so at most $k - p_t$ vertices in $C \cap R$ dominate $p_{t-1} = |B| - p_t$ vertices in B. Since each vertex of B has a neighbour in R, at most k vertices in R dominate all of B. □

This completes the proof of the theorem. □

The reduction employed in the above argument also implies:

Corollary 1. *For each fixed $t \geq 2$ the t-Total Vertex Cover problem is* NP*-hard on bipartite graphs, and the t-Total Vertex Cover problem does not admit a polynomial kernel on bipartite graphs unless CoNP $\subseteq$ NP/poly.*

3.2. Fixed parameter tractability

We now turn to the fixed-parameter tractability of t-Total Vertex Cover. Two special cases of the problem, for the two extreme values namely $t = 1$ (Vertex Cover) and $t = k$ (Connected Vertex Cover), have been studied extensively from the perspective of parameterized algorithms. The Vertex Cover problem is perhaps the most well-studied problem in parameterized algorithmics. After a long series of improvements, the current fastest FPT algorithm for this problem runs in time $\mathcal{O}^\star(1.2738^k)$ [3]. Similarly Connected Vertex Cover also has a history of improvements, and the current fastest FPT algorithm for this problem runs in $\mathcal{O}^\star(2^k)$ time [25]. We show in this section that t-Total Vertex Cover is FPT parameterized by the solution size k for each fixed $t \in \mathbb{N}$; $2 \leq t \leq k$, by deriving an $\mathcal{O}^\star(16.1^{k+\mathcal{O}(\log^2 k)})$ time algorithm for these problems.

Let $G = (V, E)$ be the input graph, and let $|V| = n$. Observe that the set V is a vertex cover of G. Therefore, if $|V| \leq k$, then we can solve the problem in polynomial time by checking whether each component of G has at least t vertices. Also, deleting isolated vertices does not affect the solution. Hence we assume without loss of generality that $|V| > k$, and that G has no isolated vertices. We start with a structural proposition which is useful later.

Proposition 2. *Let $G = (V, E)$, $|V| > k$ be a graph without isolated vertices, and let $t \in \mathbb{N}$; $2 \leq t \leq k$ be a fixed integer. Then G has a t-total vertex cover of size at most k if and only if G has a t-total vertex cover of size exactly k.*

Proof. If G has a t-total vertex cover, say S, of size exactly k, then S itself is a t-total vertex cover of G of size at most k. For the other direction, let S be a t-total vertex cover of G size $l < k$. Consider any set of $k - l$ vertices $T \subseteq (V \setminus S)$. Since G has no isolated vertex, each $v \in T$ has at least one edge incident on it; since S is a vertex cover of G, the other end of this edge, say w, is in S. Now notice that every connected component of $G[S \cup T]$ has at least t vertices as each connected component of $G[S]$ has at least t vertices and every vertex of T gets attached to one of the components of $G[S]$. □

The number of unlabelled trees on k vertices is known to be singly exponential in k, and all these trees can be enumerated with polynomial delay:

Fact 3. *(See [26,14].) The number of unlabelled trees on k vertices is at most 2.96^k. Moreover, all non-isomorphic unlabelled trees on k vertices can be enumerated in time $\mathcal{O}(2.96^k k^c)$ for some constant c independent of k.*

From this we get:

Lemma 1. *All unlabelled* forests *on k vertices can be enumerated in $\mathcal{O}^\star(2.96^k)$ time.*

Proof. Let F be a forest on k vertices. Add a new vertex v and one edge from v to an arbitrary vertex of each tree in F, to obtain a tree T on $k+1$ vertices. Clearly, the forest F can be obtained by deleting one vertex (namely v) from T. It follows that a graph is a forest on k vertices if and only if it can be obtained by deleting a vertex from some tree on $k+1$ vertices.

To enumerate all forests on k vertices, we first enumerate all trees on $k+1$ vertices. From Fact 3, this can be done in $\mathcal{O}(2.96^{k+1}(k+1)^c)$ time where c is a constant independent of k. For each tree T on $k+1$ vertices obtained in this manner, we delete each of its $k+1$ vertices, one at a time, to obtain a set of forests. By the above observation, this procedure yields every forest on k vertices (some of them perhaps many times). The procedure takes $\mathcal{O}((k+1)2.96^{k+1}(k+1)^c) = \mathcal{O}^\star(2.96^k)$ time. □

We are now ready to prove the main result of this section.

Theorem 3. *For every fixed integer $t \geq 1$ the t-Total Vertex Cover problem is fixed-parameter tractable, and can be solved in time $\mathcal{O}^\star(16.1^k)$.*

Proof. Observe that any t-total vertex cover, say S, of G is also a vertex cover of G and hence contains an inclusion-minimal vertex cover $S' \subseteq S$ of G. The idea of our proof is to enumerate all the inclusion-minimal vertex covers of G of size at most k and then try to expand each one to a t-total vertex cover of G. We will use Fact 3 and the colour-coding technique of Alon et al. [15] to do the expansion phase of our algorithm. Our algorithm is based on the following proposition.

Proposition 3. *A graph $G = (V, E)$ has a t-total vertex cover of size k if and only if there exists an inclusion-minimal vertex cover C of G of size at most k, and a subset $T \subseteq V \setminus C$ of size $k - |C|$, such that there exists a forest F on k vertices which is isomorphic to a spanning subgraph of $G[C \cup T]$, and in which each connected component has at least t vertices.*

Proof. If G has a t-total vertex cover S of size k, then by definition S is a vertex cover of G. Let C be any inclusion-minimal vertex cover contained in S, and let $T = S \setminus C$. Then $|T| = k - |C|$, and each connected component of $G[C \cup T] = G[S]$ has, from the definition of a t-total vertex cover, at least t vertices. Let F be a forest formed by picking one spanning tree from each connected component of $G[S]$. Then F satisfies the conditions of the proposition.

Conversely, suppose there exist (i) an inclusion-minimal vertex cover C of G, (ii) a set $T \subseteq V \setminus C$ of size $k - |C|$, and (iii) a forest F on k vertices that is isomorphic to a spanning subgraph $G' = (S = C \cup T, E')$ of $G[C \cup T]$, and in which each connected component has at least t vertices. Since $C \subseteq S$, S is a vertex cover of G. Also $|S| = |C \cup T| = k$. Now since G' is a subgraph of $G[S]$, and each connected component of G' contains at least t vertices, each connected component of $G[S]$ has at least t vertices. It follows that S is a t-total vertex cover of G of size k. □

If G has a t-total vertex cover of size at most k, then from Proposition 2 we know that G has a t-total vertex cover of size exactly k. Let S be a fixed t-total vertex cover of G of size exactly k, if there exists one. From Proposition 3 we get that S contains an inclusion-minimal vertex cover C of G, of size at most k. A "colouring" of the vertex set V of G is a function from V to some specified set of "colours". A "good" colouring of V is a colouring in which the vertices in S are all distinctly coloured.

Our algorithm tries to find S by mimicking Proposition 3. First we enumerate all inclusion-minimal vertex covers of G of size at most k. This can be done in time $\mathcal{O}^\star(2^k)$ by a simple 2-way branching on edges—for every edge at least one of its end-points should be in any vertex cover. For each such vertex cover C, we do the following:

1. Colour each $v \in C$ with a distinct colour from $\{1, 2, \ldots, |C|\}$.
2. Let $\ell = k - |C|$. Colour the vertices of the independent set $V \setminus C$ uniformly at random with ℓ new colours $\tilde{1}, \tilde{2}, \ldots, \tilde{\ell}$.

Observe that $|S \setminus C| = \ell$. The number of ways of colouring the vertices in $S \setminus C$ with ℓ distinct colours is $\ell!$, and the total number of ways of colouring these vertices with these ℓ colours is ℓ^ℓ. The random colouring described above will therefore yield a good colouring of V with probability $\ell!/\ell^\ell \geq e^{-\ell}$.

We now check if the random colouring is a good colouring. For this, we iterate through all unlabelled forests on k vertices, and check if at least one of these forests is isomorphic to a spanning forest $\mathcal{F}$ of $G[S]$, where each connected component of $\mathcal{F}$ has at least t vertices. By Lemma 1, we can iterate through all such forests in $\mathcal{O}^\star(2.96^k)$ time. To check if a given forest F on k vertices is isomorphic with such a spanning forest $\mathcal{F}$ of $G[S]$, we do the following:

1. We check if there is at least one tree in F that has less than t vertices. If yes, then we reject F.
2. Next we check if there is a colourful subgraph (one in which each vertex has a distinct colour) isomorphic to F in the coloured graph obtained above. Since F is of treewidth at most 1, this can be done in $\mathcal{O}(2^k \cdot k \cdot n^2)$ time [27, Corollary 6]. If such a subgraph is present, then F satisfies the requirements of Proposition 3, and so we return Yes; the underlying uncoloured graph of this colourful subgraph is a t-total vertex cover of G of size at most k. Otherwise we reject the forest F.

If the above check rejects all unlabelled forests on k vertices, then we return No: this colouring is not a good colouring.

Observe that if the input graph G has a t-total vertex cover of size k, then the above algorithm will discover this t-total vertex cover with probability at least $e^{-\ell}$, and so will return Yes with probability at least $e^{-\ell}$. If the input graph G is a No instance, then the algorithm will always return No. The expected number of times the algorithm has to be repeated before it finds a t-total vertex cover of size k of G, if it exists, is thus e^ℓ. The expected running time of this procedure is thus

$$\mathcal{O}^\star\left(\sum_{\ell=0}^{k} 2^{k-\ell} \times e^\ell \times 2.96^k \times 2^k\right) = \mathcal{O}^\star\left((2 \times 2.96 \times 2)^k \times \sum_{\ell=0}^{k}\left(\left(\frac{e}{2}\right)^\ell\right)\right)$$
$$= \mathcal{O}^\star\big((5.92e)^k\big) = \mathcal{O}^\star\big(16.1^k\big).$$

To obtain a deterministic algorithm we have to replace the randomized step of the algorithm—where we colour the vertices of $G[V \setminus C]$ uniformly at random by ℓ colours—with a deterministic procedure. We do this using the so-called *(n, ℓ, ℓ)-perfect hash families.*

An (n, ℓ, ℓ)-perfect hash family $\mathcal{H}$ is a set of functions from $\{1, \ldots, n\}$ to $\{1, \ldots, \ell\}$ such that for every subset $S \subseteq \{1, \ldots, n\}$ of size ℓ there exists a function $f \in \mathcal{H}$ such that f is injective on S. That is, such that for all $i, j \in S$, $f(i) \neq f(j)$. There exists a construction of an (n, ℓ, ℓ)-perfect hash family of size $\mathcal{O}(e^\ell \cdot \ell^{\mathcal{O}(\log \ell)} \cdot \log n)$ and one can produce this family in time linear in the output size [28].

Let $\tilde{n} = |V \setminus C|$. To derandomize our algorithm, we construct an $(\tilde{n}, \ell, \ell)$-perfect hash family $\mathcal{H}$ from $\{1, \ldots, \tilde{n}\}$ to $\{\tilde{1}, \ldots, \tilde{\ell}\}$. This has to be done only *once* during the algorithm. Now we replace the second step in the colouring process with the following:

2′. Instead of colouring the vertices of $V \setminus C$ uniformly at random with ℓ colours and checking if this yields a good colouring, we "colour" the vertices of $V \setminus C$ with *each* function in $\mathcal{H}$, in turn. For each such colouring, we check if it is a good colouring as before.

If G has a t-total vertex cover S of size k, then from the definition of an $(\tilde{n}, \ell, \ell)$-perfect hash family, it follows that at least one of the functions in $\mathcal{H}$ will result in a good colouring of V. Thus, if G has a t-total vertex cover S of size k, then this algorithm will always discover S and return Yes. If the input graph G is a No instance, then the algorithm will always return No. Instead of the multiplicative factor of e^{ℓ}—which came from repeating the randomized algorithm these many times—in the running time of the randomized procedure, the derandomized version incurs the following costs:

1. An additive cost of $\mathcal{O}(e^{\ell} \cdot \ell^{\mathcal{O}(\log \ell)} \cdot \log \tilde{n})$ to compute the hash family, and
2. A multiplicative factor of $\mathcal{O}(e^{\ell} \cdot \ell^{\mathcal{O}(\log \ell)} \cdot \log \tilde{n})$ which arises from repeating the check for a good colouring once for each function in the hash family.

The running time of the derandomized algorithm is thus

$$\begin{aligned}
&\mathcal{O}^{\star}\left(\sum_{\ell=0}^{k} 2^{k-\ell} \times e^{\ell} \times \ell^{\mathcal{O}(\log \ell)} \times 2.96^{k} \times 2^{k}\right) \\
&= \mathcal{O}^{\star}\left((2 \times 2.96 \times 2)^{k} \times \sum_{\ell=0}^{k}\left(\left(\frac{e}{2}\right)^{\ell} \times \ell^{\mathcal{O}(\log \ell)}\right)\right) \\
&= \mathcal{O}^{\star}\left((11.84)^{k} \times k^{\mathcal{O}(\log k)} \times \left(\frac{e}{2}\right)^{k}\right) \\
&= \mathcal{O}^{\star}\left(16.1^{k}\right).
\end{aligned}$$

This concludes the proof of the theorem. □

4. Computing total edge covers

We now consider the t-Total Edge Cover problem. For $t = 1$ this problem becomes the Edge Cover problem, which has long been known to be solvable in polynomial time [8]. The problem is NP-complete for each fixed $2 \le t \le k$ [9, Theorem 3]. In this section we investigate the parameterized complexity of the t-Total Edge Cover problem for $2 \le t \le k$. We first study the kernelization complexity of the problem, and improve the size of the kernel for each fixed $t \ge 2$. Then we take up the fixed parameter tractability of this problem, and obtain an FPT algorithm with a significantly improved running time.

In our analysis we make use of a different formulation of the problem, other than the one presented at the beginning of this paper. The following fact, culled from the proof of a theorem in the previous work due to Fernau and Manlove [9, Theorem 16], helps us show that the two formulations are equivalent.

Fact 4. *(See [9].) In any connected graph G with n vertices, and for any fixed $t < n$, there exists a t-total edge cover of G of the smallest size—call it S—such that the graph $G(S)$ induced by the edge set S is acyclic.*

4.1. Kernelization complexity

Fernau and Manlove [9] observed the following simple vertex kernel of size at most $2k$ for t-Total Edge Cover: any edge in a graph covers exactly 2 vertices, and a YES instance of the problem has an edge cover of size (number of edges) at most k, and so such an instance cannot have more than $2k$ vertices. In other words, if the input instance has more than $2k$ vertices, then the answer is NO. Otherwise, the input instance itself forms a kernel on at most $2k$ vertices. We can improve this bound on the kernel size for larger values of t by observing the following:

Lemma 2. *Given a graph $G = (V, E)$, a non-negative integer k, and a fixed integer t; $2 \le t \le k$, solving the t-Total Edge Cover instance (G, k) is equivalent to solving the following problem: does there exist a partition of the vertex set V into q parts $V_1, \ldots, V_q$, for some q, such that (i) $G[V_i]$ is connected, (ii) $|V_i| \ge t + 1$ for each $1 \le i \le q$, and (iii) $\sum_{i=1}^{q}(|V_i| - 1) \le k$?*

Proof. Let $(G = (V, E), k)$ be a Yes instance of t-Total Edge Cover and let S be a t-total edge cover of G of the smallest size which satisfies the conditions of Fact 4. Let $V_1, \ldots, V_q$ be the vertex sets of the connected components of $G(S) = (V, S)$. It directly follows from Fact 4 and the properties of S given in the definition of t-Total Edge Cover that $V_1, \ldots, V_q$ satisfy all the conditions in the statement of the lemma. For the reverse direction, observe first that the edges of a spanning tree of any connected graph form an edge cover of the graph. Now, if $V_1, \ldots, V_q$ satisfy all the conditions in the statement of the lemma, then let S_i be the edges of a spanning tree of $G[V_i]$, for $1 \le i \le q$, and let $S = \bigcup_{i=1}^{q} S_i$. Observe now that S has the properties stated in the definition of t-Total Edge Cover. □

We use this reformulation to get smaller bounds on the kernel size for t-Total Edge Cover.

Theorem 4. *t-Total Edge Cover admits a vertex kernel of size $\frac{t+1}{t}k$.*

Proof. Let $(G=(V,E),k)$ be a Yes instance of t-Total Edge Cover. By Lemma 2, there exists a partition of V into q parts of the kind stated in Lemma 2. Now $|V_i| \geq t+1 \Longrightarrow |V_i|-1 \geq t \Longrightarrow \sum_{i=1}^{q}(|V_i|-1) \geq qt$. By Lemma 2, $\sum_{i=1}^{q}(|V_i|-1) \leq k$, and so $qt \leq k$, and $q \leq \frac{k}{t}$. Also, $\sum_{i=1}^{q}(|V_i|-1) \leq k \Longrightarrow \sum_{i=1}^{q}|V_i| \leq k+q \leq k+\frac{k}{t} = \frac{t+1}{t}k$, and so G has at most $\frac{t+1}{t}k$ vertices. □

Thus, if the input graph G has more than $\frac{t+1}{t}k$ vertices, then the answer is No. Therefore we can assume without loss of generality that the input graph has at most $\frac{t+1}{t}k$ vertices, and so *any* exact algorithm for the problem is in fact an FPT algorithm. In particular we have:

Corollary 2. *If t-Total Edge Cover has an exact exponential time algorithm that runs in $\mathcal{O}^\star(c^{f(|V|)})$ time on an input instance $(G=(V,E),k)$ for some constant c and function $f()$, then the problem has an FPT algorithm that runs in $\mathcal{O}^\star(c^{f(\frac{t+1}{t}k)})$ time.*

4.2. Fixed parameter tractability

We now present an exact exponential-time algorithm for t-Total Edge Cover which runs in $\mathcal{O}^\star(2^{n+\mathcal{O}(\sqrt{n})})$ time where n is the number of vertices in the input graph. By Corollary 2 this yields an FPT algorithm for the problem which runs in $\mathcal{O}^\star(c^k)$ time for some fixed constant $c<3$ (when $t \geq 2$). This is a significant improvement over the previous best upper bound of $\mathcal{O}^\star((2k)^{2k})$ [9].

Let $(G=(V,E),k)$ be an input instance of t-Total Edge Cover, and let $|V|=n$. We start by enumerating all unordered partitions of n. An *unordered partition* of a positive integer n is a way of writing n as a sum of positive integers, where the order of the summands is ignored. By the Hardy–Ramanujan asymptotic formula, n has at most $2^{\mathcal{O}(\sqrt{n})}$ unordered partitions [16]. The partitions of n can be generated with constant average delay [29], and so we can enumerate all unordered partitions of n in $2^{\mathcal{O}(\sqrt{n})}$ time.

We consider those partitions of n of the form $n=n_1+n_2+\cdots+n_q$ which satisfy the following conditions of Lemma 2:

1. $\sum_{i=1}^{q}(|n_i|-1) \leq k$
2. $|n_i| \geq t+1$

For each such partition of n, we check if there exists a partition of the vertex set V into q parts $V_1,\dots,V_q$ such that

1. $|V_i|=n_i$ for $1 \leq i \leq q$, and
2. each induced subgraph $G[V_i]$ is connected.

To do these latter checks, we construct the q lists

$$L_i = \{V' \subseteq V \mid |V'| = n_i \text{ and } G[V'] \text{ is connected}\}$$

for $1 \leq i \leq q$. For $1 \leq i \leq q$ we compute the polynomials

$$P_i = \sum_{V' \in L_i} z^{\chi(V')}$$

where z is a formal variable and $\chi(V')$ is the (binary number represented by the) characteristic vector of $V' \subseteq V$. That is, let $V=\{v_1,v_2,\dots,v_n\}$. Then $\chi(V')$ is a bit vector with $|V|$ bits where, for $1 \leq j \leq |V|$, the jth bit of $\chi(V')$ is 1 if and only if $v_j \in V'$. We treat $\chi(V')$ as a binary number in all computations. The lists L_i and the polynomials P_i can be computed in $\mathcal{O}^\star(2^n)$ time, by enumerating all subsets of V. We now compute the product

$$Q = P_1 \times P_2 \times \cdots \times P_q$$

in the given order, with a small modification: at each step, given the partial product Q_i of the first i terms, we first compute $Q_i \times P_{i+1}$. Then we delete all those terms αz^β in $Q_i \times P_{i+1}$ where (the binary representation of) β does not contain exactly $\sum_{j=1}^{i+1} n_j$ 1s, and set Q_{i+1} to be the resulting polynomial. This pruning operation ensures that the partial product Q_i, for $1 \leq i \leq q$, represents exactly those sets of size $\sum_{j=1}^{i} n_j$ that can be obtained by taking the union of one set each from $L_1, L_2, \dots, L_i$. Observe that the product $Q_q = Q$ is non-zero if and only if there exists a partition of V into q parts satisfying the conditions stated above.

The degree of each polynomial involved in the multiplications is at most $2^{|V|}-1=2^n-1$, and so, using the Fast Fourier Transform, we can multiply two of these polynomials in $\mathcal{O}(2^n \log 2^n) = \mathcal{O}(n2^n)$ time [30, Chapter 30]. We have to perform at most $q \le n$ such multiplications to compute Q, and so given the P_is we can compute Q in $\mathcal{O}(n^2 2^n) = \mathcal{O}^\star(2^n)$ time. The running time of this algorithm is thus $2^{\mathcal{O}(\sqrt{n})} \times (\mathcal{O}^\star(2^n) + \mathcal{O}^\star(2^n)) = \mathcal{O}^\star(2^{n+\mathcal{O}(\sqrt{n})})$, and so we have:

Theorem 5. *t-Total Edge Cover can be solved in $\mathcal{O}^\star(2^{n+\mathcal{O}(\sqrt{n})})$ time, where n is the number of vertices in the input graph.*

From this theorem and Corollary 2 we get:

Theorem 6. *t-Total Edge Cover can be solved in $\mathcal{O}^\star(2^{\frac{t+1}{t}k+\mathcal{O}(\sqrt{k})})$ time.*

The above algorithm uses exponential space, for constructing the lists L_i. We can use an approach similar to the one used in Section 3.2 to get an FPT algorithm which runs in polynomial space. Specifically, we enumerate all unordered partitions $(n_1, \dots, n_q)$ of n where $n - k \le q \le \frac{n}{t+1}$, such that $n_i \ge t+1$ and $\sum_{i=1}^{q}(n_i - 1) \le k$. As mentioned above, this can be done in $\mathcal{O}^\star(2^{\sqrt{n}})$ time. For each such partition, we enumerate all trees with number of vertices n_i; $1 \le i \le q$. As mentioned above, this can be done in $\mathcal{O}^\star(2.96^n)$ time. Then, for each enumerated q-tuple of trees $(T_1, \dots, T_q)$, we test whether the forest $T_1 \uplus T_2 \uplus \dots \uplus T_q$ is a subgraph of G. Since the forest has treewidth one and has the same number of vertices as G, this test for subgraph isomorphism can be done in $\mathcal{O}^\star(2^n)$ time and polynomial space [31, Theorem 5]. Combining this with Corollary 2 we get:

Theorem 7. *t-Total Edge Cover can be solved in $\mathcal{O}^\star(2^{c\frac{t+1}{t}k+\mathcal{O}(\sqrt{\frac{t+1}{t}k})})$ time and using polynomial space, where $c = (1 + \log_2 2.96)$.*

5. *t*-Total Edge Dominating Set

The techniques developed in the previous sections can be used to solve other problems with “total connectivity” constraints. As an illustration, we now show how to solve the t-total version of the NP-hard Edge Dominating Set problem.

An *edge dominating set* of a graph $G = (V, E)$ is a set $F \subseteq E$ of edges of G such that every edge in $E \setminus F$ has at least one vertex in common with some edge in F; the edges in F *dominate* all the other edges in G. The Edge Dominating Set problem takes a graph G and a positive integer k as input, and asks whether the graph G has an edge dominating set of size at most k. This problem is NP-hard, even for planar bipartite graphs with maximum degree 3 [32]. When parameterized by k, the problem is FPT and has a quadratic kernel [33]. We investigate the t-total version of this problem, which is defined for each fixed $t \in \mathbb{N}$; $1 \le t \le k$ as follows:

t-Total Edge Dominating Set
Input: A graph $G = (V, E)$, and a non-negative integer k.
Parameter: k
Question: Does G have an edge dominating set F of size at most k such that each component of the subgraph of G induced by F contains at least t edges?

We call an edge dominating set which satisfies the conditions specified in the problem a *t-total edge dominating set* of G. If the input graph has a sufficient number of edges, then it suffices to look for a t-total edge dominating set of size exactly k.

Proposition 4. *Let $G = (V, E)$ be a graph such that $|E| \ge k$, and let $t \in \mathbb{N}$, $1 \le t \le k$. Then G has a t-total edge dominating set of size at most k if and only if G has a t-total edge dominating set of size exactly k.*

Proof. If G has a t-total edge dominating set, say F, of size exactly k, then F itself is a t-total edge dominating set of G of size at most k. For the other direction, let F be a t-total edge dominating set of G size $l < k$. Consider any set of $k - l$ edges $T \subseteq (E \setminus F)$. Since F is an edge dominating set, every edge in T shares at least one vertex with some edge in F. It follows that $F \cup T$ is a t-total edge dominating set of G, of size exactly k. □

The vertices involved in an edge dominating set form a vertex cover of the graph. Also, if a vertex cover meets a simple connectivity requirement, then there exists a t-total edge dominating set which “contains” the vertex cover.

Proposition 5. *Let $G = (V, E)$ be a graph.*

1. *If F is an edge dominating set of G, then $V(F)$ is a vertex cover of G.*
2. *Let C be a vertex cover of G, and let t be a fixed integer; $1 \le t \le k$. Let $\tilde{F} = (\tilde{V}, \tilde{E})$ be a subgraph of G such that $\tilde{F}$ is a forest, $C \subseteq \tilde{V}$, and each tree in $\tilde{F}$ has at least t edges. Then $\tilde{E}$ is a t-total edge dominating set of G.*

Proof.

1. Let $e = \{u, v\}$ be an arbitrary edge in G. Since F is an edge dominating set of G, there exists an edge $f = \{x, y\} \in F$ such that $e \cap f \neq \emptyset$; so suppose $u = x$. Since $x \in V(F)$ we have that $V(F)$ covers the edge e, and since e was chosen arbitrarily we get that $V(F)$ covers all the edges in G.
2. Let $e = \{u, v\}$ be an arbitrary edge in G. Since C is a vertex cover of G, $e \cap C \neq \emptyset$; so suppose $u \in C$. Let $f \in \tilde{F}$ be an edge such that u is incident with f. The lower bound on the component sizes of $\tilde{F}$ guarantees the existence of such an edge, and so we have that $\tilde{F}$ dominates the edge e. Since e was chosen arbitrarily we get that $\tilde{F}$ dominates all the edges in G. The component size lower bound implies that $\tilde{F}$ is a t-total edge dominating set. □

Edge dominating sets split cleanly across connected components, in the following sense:

Observation 1. *An edge subset F of a graph G is a minimum-size t-total edge dominating set of G if and only if F is the disjoint union of minimum-size t-total edge dominating sets of the connected components of G.*

This observation helps us get rid of small components in the input graph.

Lemma 3. *Given an instance (G, k) of t-*Total Edge Dominating Set *one can, in polynomial time, either decide that (G, k) is a* No *instance, or compute an equivalent instance (G', k') such that each component of the graph G' has more than $(t + 1)$ vertices.*

Proof. Let $(G = (V, E), k)$ be an instance of t-Total Edge Dominating Set. Isolated vertices can safely be deleted from G, since they do not affect the solution. If any component of G has less than t *edges* then (G, k) is a No instance, and this can be detected in polynomial time. So we assume, without loss of generality, that each component of the graph G has at least t edges.

Let $C = (V_C, E_C)$ be a component of G with at most $t + 1$ vertices. Let F be a t-total edge dominating set of G, and let $F_C = F \cap E_C$. Further, let F' be a connected subgraph of C, with t edges, obtained by adding sufficiently many edges to an arbitrary spanning tree of C. Since the component C has at least t edges such an F' always exists, and it can be found in polynomial time.

The set F' (i) induces a connected subgraph of C with exactly t edges, and (ii) dominates all the edges in C, and so F' is a minimum-size t-total edge dominating set of the component C. It follows from Observation 1 that (G, k) is a Yes instance of t-Total Edge Dominating Set if and only if $G' = G[V \setminus V_C], k' = k - t$ is a Yes instance. Hence we have the following reduction rule: Delete the component C and reduce k by t.

We repeatedly apply this reduction rule, till (i) there are no more components with at most $t + 1$ vertices, or (ii) the budget k becomes negative, whichever happens first. If the budget becomes negative, then the original input instance is a No instance, and we would have found this in polynomial time. Otherwise we would have computed—in polynomial time—an equivalent instance which has the stated property. □

An advantage of working with a graph which does not have "small" components is that it is sufficient to look for a solution which induces a *forest*.

Lemma 4. *Let G be a graph in which each component has more than $t + 1$ vertices. Then there exists a t-total edge dominating set F of G such that (i) F is a minimum-size t-total edge dominating set, and (ii) F induces a forest in G.*

Proof. Let $\mathcal{F}$ be the set all minimum-size t-total edge dominating sets of the graph G, and let $F \in \mathcal{F}$ be such that F induces a subgraph of G with the smallest number of cycles. We show that the subgraph $G(F)$ is a forest. The argument is essentially the same as the one which Fernau and Manlove used to prove a similar claim [9, proof of Theorem 16].

Suppose $G(F)$ contains a cycle C. Let $X_F \subseteq F$ induce a connected component of $G(F)$, such that the cycle C belongs to the component $G(X_F)$. Let e be an arbitrary edge in the cycle C, and let $X'_F = X_F \setminus \{e\}$. Consider the edge set $F' = (F \setminus X_F) \cup X_{F'} = F \setminus \{e\}$. Since $F \in \mathcal{F}$ and $|F'| = |F| - 1$, it cannot be the case that F' is a t-total edge dominating set of G. But X'_F induces a connected subgraph, and dominates all the edges which are dominated by X_F. So the only condition which can possibly be violated by F' is the lower bound on the number of edges in each component; thus $|X'_F| < t$. But $|X_F| \geq t$ and $|X_{F'}| = |X_F| - 1$, and so we get that $|X_F| = t$ and $|X'_F| = t - 1$.

Let X be the component of G which contains $G(X_F)$. Then X has more than $t + 1$ vertices by assumption, and $G(X'_F)$ is a connected subgraph of X with exactly $t - 1$ edges. Therefore there is a vertex in X which is not in $G(X'_F)$, and so there is an edge e' in X which is incident on exactly one vertex in $G(X'_F)$. The set $(F \setminus \{e\}) \cup \{e'\}$ then belongs to $\mathcal{F}$ and induces a subgraph of G which has a smaller number of cycles than $G(F)$, a contradiction. It follows that $G(F)$ contains no cycles. □

Lemma 5. *Let $G = (V, E)$ be a graph and let F be a t-total edge dominating set of G, of size at most k. Then $V(F)$ is a vertex cover of G, of size at most $(t + 1)k/t$.*

Proof. From Proposition 5 we get that $V(F)$ is a vertex cover of G. For the size bound, observe that F is a t-total edge *cover* of the subgraph $G[V(F)]$, and $|F| \leq k$. Thus $(G[V(F)], k)$ is a Yes instance of t-Total Edge Cover, and so from the proof of Theorem 4 we get that $|V(F)| \leq (t+1)k/t$. □

We are now ready to show that the t-Total Edge Dominating Set problem can be solved in FPT time.

Theorem 8. *For every fixed* $t \geq 1$ *the* t-Total Edge Dominating Set *problem is fixed-parameter tractable, and can be solved in time* $\mathcal{O}^\star(16.1^{(t+1)k/t})$.

Proof. Let $(G = (V, E), k)$ be an instance of t-Total Edge Dominating Set. Suppose $|E| \leq k$. Observe that E itself is an edge dominating set of G, of size at most k. To solve the problem it is therefore enough to check if each connected component of G has at least t edges. This can be done in polynomial time, and so we assume without loss of generality that $|E| > k$. Then from Proposition 4 we know that it is sufficient to check if the graph G has a t-total edge dominating set of size *exactly* k.

Further, we know—from Lemma 3, Lemma 4, and Proposition 5—that if $(G = (V, E), k)$ is a Yes instance of t-Total Edge Dominating Set, then there is a t-total edge dominating set F of G, of size at most k such that (i) $G(F)$ is a forest in which each tree has at least t edges, and (ii) $V(F)$ is a vertex cover of G of size at most $(t+1)k/t$. Our algorithm looks for such an F, using exactly the same strategy as the one which we used in the proof of Theorem 3. We now briefly describe the algorithm.

We first guess an inclusion-minimal vertex cover C of G, of size at most $(t+1)k/t$, such that $C \subseteq V(F)$. Then we guess the structure of a forest $\bar{F}$ which is isomorphic to $G[F]$. We then use colour-coding, exactly as in the proof of Theorem 3, to find a subgraph $\tilde{F}$ of G which (i) contains all the vertices of C and (ii) is isomorphic to $\bar{F}$. If (G, k) is a Yes instance, then by the above discussion there must exist a C and an $\bar{F}$ such that this procedure finds such a subgraph $\tilde{F}$. Conversely, if the procedure finds such a forest $\tilde{F}$, then from the second part of Proposition 5 we get that the edge set of $\tilde{F}$ is a t-total edge dominating set F of G of size at most k. This—together with the details in the proof of Theorem 3—shows that the algorithm is correct. Essentially the same analysis as in the proof of Theorem 3 yields the stated running time. □

6. Conclusion

In this paper we studied the effect of imposing some natural connectivity constraints on the subgraph induced by the solution set for two classical problems, namely Vertex Cover and Edge Cover. These problems exhibit contrasting behaviour with respect to classical complexity: Vertex Cover is NP-hard while Edge Cover is solvable in polynomial time. For both these problems, the additional constraint imposed is that each connected component of the subgraph induced by the solution set be at least as large as some specified number t; these problems were introduced by Fernau and Manlove [9], who named the problems t-Total Vertex Cover and t-Total Edge Cover, respectively. They showed that these problems are NP-hard for each fixed $t \in \mathbb{N}$; $2 \leq t \leq k$, and initiated the study of the parameterized complexity of these problems when the parameter is the solution size k. We take this study further, and improve on their results.

In both cases we showed that adding a connectivity constraint (each component of the solution must have at least a certain number of vertices/edges from the solution) causes a drastic change in the computational complexity of the problem. In the case of t-Total Edge Cover, the shift is from polynomial-time computability to NP-hardness, as had been observed earlier [9]. We showed that a similar shift occurs in the case of the parameterized version t-Total Vertex Cover of the NP-hard problem t-Total Vertex Cover. As is well known, for $t = 1$ the problem has a linear vertex kernel [4]. In contrast, we showed that for any fixed $2 \leq t \leq k$ the t-Total Vertex Cover problem has no polynomial-size kernel unless CoNP $\subseteq$ NP/poly, which is considered unlikely. We also showed that both these problems have FPT algorithms that run in time $\mathcal{O}^\star(c^k)$ for different constants c. These results improve known bounds for these problems [9].

It has been shown [34] that the standard parameterization of the Dominating Set problem—which is W[2]-hard in general, and hence is unlikely to have FPT algorithms—is FPT in graphs of girth at least 5, and has a cubic vertex-kernel on graphs of girth at least 7. Similarly, it can be shown—and this is not difficult—that any “reasonable” variant of the Vertex Cover problem has kernels of size $\mathcal{O}(k^2)$ on graphs which have no small cycles. In particular, t-Total Vertex Cover can be shown to have kernels of size $\mathcal{O}(k^2)$ in graphs of girth at least 5, for fixed $1 \leq t \leq k$. In other words, one can attribute the lower bound on kernel size for vertex cover problems with the slightest connectivity property, to the existence of small cycles in the input graph.

In his note titled “Measure & Conquer for Parameterized Branching Algorithms” in the “Parameterized Complexity News” of September 2009 [35], Serge Gaspers mentions that “... it would be very interesting to see a parameterized M&C analysis for a problem that does not have a linear kernel.” Our lower bound on the kernel size for t-Total Vertex Cover furnishes such an example; Fernau and Manlove’s algorithm for t-Total Vertex Cover [9] is essentially a “measure and conquer” (M&C) algorithm. Binkele-Raible and Fernau [36] give another example, when they derive a measure and conquer algorithm for Connected Vertex Cover which—conditionally—has no polynomial kernels.

As we illustrated in Section 5, the approach we used to solve t-Total Vertex Cover can be used to solve other t-total problems which share a certain characteristic with t-Total Vertex Cover, namely: any solution to the problem can be expressed as a minimal solution to the unrestricted problem, embedded in a graph of bounded treewidth. As another example, the same technique can be used to derive an FPT algorithm for the standard parameterization of t-Total Dominating Set when the input graph has girth at least 5. This is because in this case all inclusion-minimal dominating sets can be enumerated in FPT time [34]. One interesting direction of future research would be to examine the effect of such connectivity constraints on parameterized graph problems which are not known to have this property. Another open problem is to try to improve the base c of the exponent of the running times that we obtained for t-Total Vertex Cover and t-Total Edge Cover. Recall—Theorem 3—that our algorithm for the t-Total Vertex Cover problem runs in $\mathcal{O}^\star(16.1^k)$ time. For the three special values $t = 1$, $t = 2$ and $t = k$, the t-Total Vertex Cover problem is known to be solvable in $\mathcal{O}^\star(c^k)$ time for have much smaller values of c, namely $c = 1.2738$, $c = 2.3655$ and $c = 2$, respectively [3,9,25]. It will be interesting to see if the value of c for the general case can be brought closer to these smaller values.

Acknowledgements

We thank the anonymous reviewer for comments which helped us improve the presentation.

References

[1] H. Fernau, F.V. Fomin, G. Philip, S. Saurabh, The curse of connectivity: t-Total Vertex (Edge) Cover, in: M.T. Thai, S. Sahni (Eds.), Computing and Combinatorics, 16th Annual International Conference, Proceedings, COCOON 2010, Nha Trang, Vietnam, July 19–21, 2010, in: Lecture Notes in Computer Science, vol. 6196, Springer, 2010, pp. 34–43.
[2] R.M. Karp, Reducibility among combinatorial problems, in: Complexity of Computer Communications, 1972, pp. 85–103.
[3] J. Chen, I.A. Kanj, G. Xia, Improved upper bounds for vertex cover, Theoret. Comput. Sci. 411 (2010) 3736–3756.
[4] J. Chen, I.A. Kanj, W. Jia, Vertex Cover: further observations and further improvements, J. Algorithms 41 (2001) 280–301.
[5] M. Cygan, M. Pilipczuk, M. Pilipczuk, J.O. Wojtaszczyk, On multiway cut parameterized above lower bounds, in: D. Marx, P. Rossmanith (Eds.), Parameterized and Exact Computation—6th International Symposium, IPEC 2011, Saarbrücken, Germany, September 6–8, 2011, in: Lecture Notes in Computer Science, vol. 7112, Springer, 2012, pp. 1–12 (revised selected papers).
[6] V. Raman, M.S. Ramanujan, S. Saurabh, Paths, flowers and vertex cover, in: C. Demetrescu, M.M. Halldórsson (Eds.), Algorithms—ESA 2011—19th Annual European Symposium, Proceedings, Saarbrücken, Germany, September 5–9, 2011, in: Lecture Notes in Computer Science, vol. 6942, Springer, 2011, pp. 382–393.
[7] N.S. Narayanaswamy, V. Raman, M.S. Ramanujan, S. Saurabh, LP can be a cure for parameterized problems, in: 29th International Symposium on Theoretical Aspects of Computer Science, STACS 2012, February 29th–March 3rd, 2012, Paris, France, in: LIPIcs, vol. 14, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2012, pp. 338–349.
[8] R.Z. Norman, M.O. Rabin, An algorithm for a minimum cover of a graph, Proc. Amer. Math. Soc. 10 (1959) 315–319.
[9] H. Fernau, D.F. Manlove, Vertex and edge covers with clustering properties: complexity and algorithms, J. Discrete Algorithms 7 (2009) 149–167.
[10] M. Małafiejski, P. Żyliński, Weakly cooperative guards in grids, in: O. Gervasi, M.L. Gavrilova, V. Kumar, A. Laganà, H.P. Lee, Y. Mun, D. Taniar, C.J.K. Tan (Eds.), Computational Science and Its Applications, Proceedings, Part I, ICCSA 2005, International Conference, Singapore, May 9–12, 2005, in: Lecture Notes in Computer Science, vol. 3480, Springer, 2005, pp. 647–656.
[11] D.G. Kirkpatrick, P. Hell, On the completeness of a generalized matching problem, in: Proceedings of the 10th Annual ACM Symposium on Theory of Computing, May 1–3, 1978, San Diego, California, USA, ACM, 1978, pp. 240–245.
[12] A. Kosowski, M. Małafiejski, P. Żyliński, Parallel processing subsystems with redundancy in a distributed environment, in: R. Wyrzykowski, J. Dongarra, N. Meyer, J. Wasniewski (Eds.), Parallel Processing and Applied Mathematics, 6th International Conference, PPAM 2005, Poznań, Poland, September 11–14, 2005, in: Lecture Notes in Computer Science, vol. 3911, Springer, 2006, pp. 1002–1009 (revised selected papers).
[13] K.M.J. De Bontridder, B.V. Halldórsson, M.M. Halldórsson, C.A.J. Hurkens, J.K. Lenstra, R. Ravi, L. Stougie, Approximation algorithms for the Test Cover problem, Math. Program., Ser. B 98 (2003) 477–491.
[14] R. Otter, The number of trees, Ann. Math. 49 (1948) 583–599.
[15] N. Alon, R. Yuster, U. Zwick, Color-coding, J. ACM 42 (1995) 844–856.
[16] G.H. Hardy, S. Ramanujan, Asymptotic formulae in combinatory analysis, Proc. Lond. Math. Soc. 17 (1918).
[17] D.E. Knuth, The Art of Computer Programming, vol. 3, Seminumerical Algorithms, third edition, Addison-Wesley, 1998.
[18] R. Niedermeier, Invitation to Fixed-Parameter Algorithms, Oxford Lecture Series in Mathematics and Its Applications, vol. 31, Oxford University Press, Oxford, 2006.
[19] J. Flum, M. Grohe, Parameterized Complexity Theory, Springer-Verlag, 2006.
[20] H.L. Bodlaender, R.G. Downey, M.R. Fellows, D. Hermelin, On problems without polynomial kernels, J. Comput. System Sci. 75 (2009) 423–434.
[21] H.L. Bodlaender, S. Thomassé, A. Yeo, Kernel bounds for disjoint cycles and disjoint paths, in: Algorithms—ESA 2009, 17th Annual European Symposium, Proceedings, Copenhagen, Denmark, September 7–9, 2009, in: Lecture Notes in Computer Science, vol. 5757, 2009, pp. 635–646.
[22] M. Dom, D. Lokshtanov, S. Saurabh, Incompressibility through colors and IDs, in: S. Albers, A. Marchetti-Spaccamela, Y. Matias, S.E. Nikoletseas, W. Thomas (Eds.), Automata, Languages and Programming, 36th International Colloquium, Proceedings, Part I, ICALP 2009, Rhodes, Greece, July 5–12, 2009, in: Lecture Notes in Computer Science, vol. 5555, Springer, 2009, pp. 378–389.
[23] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, Freeman, San Francisco, 1979.
[24] R.G. Downey, M.R. Fellows, A. Vardy, G. Whittle, The parametrized complexity of some fundamental problems in coding theory, SIAM J. Comput. 29 (1999) 545–570.
[25] M. Cygan, Deterministic parameterized connected vertex cover, Accepted at the 13th Scandinavian Symposium and Workshops on Algorithm Theory, SWAT 2012. Available online at http://arxiv.org/abs/1202.6642.
[26] T. Beyer, S.M. Hedetniemi, Constant time generation of rooted trees, SIAM J. Comput. 9 (1980) 706–712.
[27] O. Amini, F.V. Fomin, S. Saurabh, Counting subgraphs via homomorphisms, SIAM J. Discrete Math. 26 (2012) 695–717.
[28] M. Naor, L.J. Schulman, A. Srinivasan, Splitters and near-optimal derandomization, in: 36th Annual Symposium on Foundations of Computer Science, Milwaukee, Wisconsin, 23–25 October 1995, IEEE Computer Society Press, Los Alamitos, 1995, pp. 182–193.
[29] A. Zoghbi, I. Stojmenović, Fast algorithms for generating integer partitions, Int. J. Comput. Math. 70 (1998) 319–332.
[30] T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, Introduction to Algorithms, second edition, The MIT Press, 2001.
[31] O. Amini, F.V. Fomin, S. Saurabh, Counting subgraphs via homomorphisms, in: Automata, Languages and Programming, 36th International Colloquium, Proceedings, Part I, ICALP 2009, Rhodes, Greece, July 5–12, 2009, in: Lecture Notes in Computer Science, vol. 5555, Springer, 2009, pp. 71–82.
[32] M. Yannakakis, F. Gavril, Edge dominating sets in graphs, SIAM J. Appl. Math. 38 (1980) 364–372.

[33] M. Xiao, T. Kloks, S.-H. Poon, New parameterized algorithms for the edge dominating set problem, in: F. Murlak, P. Sankowski (Eds.), Mathematical Foundations of Computer Science 2011—36th International Symposium, Proceedings, MFCS 2011, Warsaw, Poland, August 22–26, 2011, in: Lecture Notes in Computer Science, vol. 6907, Springer, 2011, pp. 604–615.
[34] N. Misra, G. Philip, V. Raman, S. Saurabh, The effect of girth on the kernelization complexity of Connected Dominating Set, in: K. Lodaya, M. Mahajan (Eds.), IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2010, December 15–18, 2010, Chennai, India, in: LIPIcs, vol. 8, Schloss Dagstuhl—Leibniz-Zentrum für Informatik, 2010, pp. 96–107.
[35] Parameterized complexity news, http://mrfellows.net/Newsletters/09_Sept_News.pdf, September 2009.
[36] D. Binkele-Raible, H. Fernau, Parameterized measure & conquer for problems with no small kernels, Algorithmica 64 (2012) 189–212.