Lossy Kernelization for (Implicit) Hitting Set Problems

Fedor V. Fomin
University of Bergen, Norway

Tien-Nam Le
École Normale Supérieure de Lyon, France

Daniel Lokshtanov
University of California Santa Barbara, CA, USA

Saket Saurabh
The Institute of Mathematical Sciences, HBNI, Chennai, India
University of Bergen, Norway

Stéphan Thomassé
École Normale Supérieure de Lyon, France

Meirav Zehavi
Ben-Gurion University of the Negev, Beer-Sheva, Israel

Abstract

We re-visit the complexity of polynomial time pre-processing (kernelization) for the d-Hitting Set problem. This is one of the most classic problems in Parameterized Complexity by itself, and, furthermore, it encompasses several other of the most well-studied problems in this field, such as Vertex Cover, Feedback Vertex Set in Tournaments (FVST) and Cluster Vertex Deletion (CVD). In fact, d-Hitting Set encompasses any deletion problem to a hereditary property that can be characterized by a finite set of forbidden induced subgraphs. With respect to bit size, the kernelization complexity of d-Hitting Set is essentially settled: there exists a kernel with $\mathcal{O}(k^d)$ bits ($\mathcal{O}(k^d)$ sets and $\mathcal{O}(k^{d-1})$ elements) and this it tight by the result of Dell and van Melkebeek [STOC 2010, JACM 2014]. Still, the question of whether there exists a kernel for d-Hitting Set with *fewer elements* has remained one of the most major open problems in Kernelization.

In this paper, we first show that if we allow the kernelization to be *lossy* with a qualitatively better loss than the best possible approximation ratio of polynomial time approximation algorithms, then one can obtain kernels where the number of elements is linear for every fixed d. Further, based on this, we present our main result: we show that there exist approximate Turing kernelizations for d-Hitting Set that even beat the established bit-size lower bounds for exact kernelizations – in fact, we use a *constant* number of oracle calls, each with *"near linear"* ($\mathcal{O}(k^{1+\epsilon})$) bit size, that is, almost the best one could hope for. Lastly, for two special cases of implicit 3-Hitting set, namely, FVST and CVD, we obtain the "best of both worlds" type of results – $(1+\epsilon)$-approximate kernelizations with a linear number of vertices. In terms of size, this substantially improves the exact kernels of Fomin et al. [SODA 2018, TALG 2019], with simpler arguments.

2012 ACM Subject Classification Theory of computation → Parameterized complexity and exact algorithms

Keywords and phrases Hitting Set, Lossy Kernelization

Digital Object Identifier 10.4230/LIPIcs.ESA.2023.49

Related Version *Full Version*: `https://arxiv.org/abs/2308.05974`

Funding *Fedor V. Fomin*: Research Council of Norway via the project BWCA (grant no. 314528).
Daniel Lokshtanov: Supported by NSF award CCF-2008838.
Saket Saurabh: European Research Council (ERC) grant agreement no. 819416, and Swarnajayanti Fellowship no. DST/SJF/MSA01/2017-18.
Stéphan Thomassé: ANR projects TWIN-WIDTH (CE48-0014-01) and DIGRAPHS (CE48-0013-01).
Meirav Zehavi: European Research Council (ERC) grant titled PARAPATH.

1 Introduction

In d-Hitting Set, the input consists of a universe U, a family $\mathcal{F}$ of sets over U, where each set in $\mathcal{F}$ is of size at most d, and an integer k. The task is to determine whether there exists a set $S \subseteq U$, called a *hitting set*, of size at most k that has a nonempty intersection with every set of $\mathcal{F}$. The d-Hitting Set problem is a classical optimization problem whose computational complexity has been studied for decades from the perspectives of different algorithmic paradigms. Notably, d-Hitting Set is a generic problem, and hence, in particular, various computational problems can be re-cast in terms of it. Of course, Vertex Cover, the most well-studied problem in Parameterized Complexity, is the special case of d-Hitting Set with $d = 2$. More generally, d-Hitting Set encompasses a variety of (di)graph modification problems, where the task is to delete at most k vertices (or edges) from a graph such that the resulting graph does not contain an induced subgraph (or a subgraph) from a family of forbidden graphs $\mathcal{F}$. Examples of some such well-studied problems include Cluster Vertex Deletion, d-Path Vertex Cover, d-Component Order Connectivity, d-Bounded-Degree Vertex Deletion, Split Vertex Deletion and Feedback Vertex Set in Tournaments.

Kernelization, a subfield of Parameterized Complexity, provides a mathematical framework to capture the performance of polynomial time preprocessing. It makes it possible to quantify the degree to which polynomial time algorithms succeed at reducing input instances of NP-hard problems. More formally, every instance of a parameterized problem Π is associated with an integer k, which is called the *parameter*, and Π is said to admit a *kernel* if there is a polynomial-time algorithm, called a *kernelization algorithm*, that reduces the input instance of Π down to an equivalent instance of Π whose size is bounded by a function $f(k)$ of k. (Here, two instances are equivalent if both of them are either Yes-instances or No-instances.) Such an algorithm is called an $f(k)$-*kernel* for Π. If $f(k)$ is a polynomial function of k, then we say that the kernel is a *polynomial kernel*. Over the last decade, Kernelization has become a central and active field of study, which stands at the forefront of Parameterized Complexity, especially with the development of complexity-theoretic lower bound tools for kernelization. These tools can be used to show that a polynomial kernel [3, 12, 18, 23], or a kernel of a specific size [9, 10, 21] for concrete problems would imply an unlikely complexity-theoretic collapse. We refer to the recent book on kernelization [17] for a detailed treatment of the area of kernelization. In this paper, we provide a number of positive results on the kernelization complexity of d-Hitting Set, as well as on several special cases of 3-Hitting Set.

The most well-known example of a polynomial kernel, which, to the best of our knowledge, is taught in the first class/chapter on kernelization of any course/book that considers this subject, is the classic kernel for Vertex Cover (2-Hitting Set) that is based on Buss rule. More generally, one of the most well-known examples of a polynomial kernel is a kernel with $\mathcal{O}(k^d)$ sets and elements for d-Hitting Set (when d is a fixed constant) using the Erdös-Rado Sunflower lemma.[1] Complementing this positive result, originally in 2010, a celebrated result by Dell and van Melkebeek [10] showed that unless co-NP $\subseteq$ NP/ poly, for any $d \geq 2$ and any $\epsilon > 0$, d-Hitting Set does not admit a kernel with $\mathcal{O}(k^{d-\epsilon})$ sets. Hence, the kernel with $\mathcal{O}(k^d)$ sets is essentially tight with respect to size. However, when it comes to the bound on the number of elements in a kernel, the situation is unclear. Abu-Khzam [1]

[1] The origins of this result are unclear. The first kernel with $\mathcal{O}(k^d)$ sets appeared in 2004 [13], but the authors do not make use of the Sunflower Lemma. To the best of our knowledge, the first exposition of the kernel based on the Sunflower Lemma appears in the book of Flum and Grohe [15].

showed that d-Hitting Set admits a kernel with at most $(2d-1)k^{d-1}+k$ elements. However, we do not know whether this bound is tight or even close to that. As it was written in [17, page 470]:

> *Could it be that* d-Hitting Set *admits a kernel with a polynomial in k number of elements, where the degree of the polynomial does not depend on d? This does not look like a plausible conjecture, but we do not know how to refute it either.*

The origins of this question can be traced back to the open problems from WorKer 2010 [4, page 4]. Moreover, in the list of open problems from WorKer 2013 and FPT School 2014 [7, page 4], the authors asked whether d-Hitting Set admits a kernel with $f(d) \cdot k$ elements for some function f of d only. After being explicitly stated at these venues, this question and its variants have been re-stated in a considerable number of papers (see, e.g., [11, 17, 29, 2]), and is being repeatedly asked in annual meetings centered around parameterized complexity. Arguably, this question has become the most major and longstanding open problem in kernelization for a specific problem. In spite of many attempts, even for $d = 3$, the question whether d-Hitting Set admits a kernel with $\mathcal{O}(k^{2-\varepsilon})$ elements, for some $\epsilon > 0$, has still remained open.

From an approximation perspective, the optimization version of d-Hitting Set admits a trivial d-approximation. Up to the Unique Game Conjecture, this bound is tight – for any $\varepsilon > 0$, d-Hitting Set does not admit a polynomial time $(d-\varepsilon)$-approximation [22]. So, at this front, the problem is essentially resolved.

With respect to kernelization, firstly, the barrier in terms of number of sets, and secondly, the lack of progress in terms of the number of elements, coupled with the likely impossibility of $(d-\varepsilon)$-approximation of d-Hitting Set, bring lossy kernelization as a natural tool for further exploring of the complexity of this fundamental problem. We postpone the formal definition of lossy kernelization to Section 2. Informally, a polynomial size α-approximate kernel consists of two polynomial-time procedures. The first is a pre-processing algorithm that takes as input an instance (I, k) to a parameterized problem, and outputs another instance (I', k') to the same problem, such that $|I'| + k' \leq k^{\mathcal{O}(1)}$. The second transforms, for every $c \geq 1$, a c-approximate solution S' to the pre-processed instance (I', k') into a $(c \cdot \alpha)$-approximate solution S to the original instance (I, k). Then, the main question(s) that we address in this paper is:

> Is it possible to obtain a lossy kernel for d-Hitting Set with a qualitatively better loss than d and with $\mathcal{O}(k^{d-1-\varepsilon})$ bit-size, or at least with $\mathcal{O}(k^{d-1-\varepsilon})$ elements?

In this paper, we present a surprising answer: *not only the number of elements can be bounded by $\mathcal{O}(k)$ (rather than just $\mathcal{O}(k^{d-1-\varepsilon})$), but even the bit-size can "almost" be bounded by $\mathcal{O}(k)$!* From the perspective of the size of the kernel, this is essentially the best that one could have hoped for. Still, we only slightly (though non-negligibly) improve on the approximation ratio d. For example, for $d = 2$ (Vertex Cover), we attain an approximation ratio of 1.721. So, while we make a critical step that is also the first – in particular, we show that, conceptually, the combination of kernelization and approximation breaks their independent barriers – we also open up the door for further research of this kind, on this problem as well as other problems.

More precisely, we present the following results and concept. We remark that for all of our results, we use an interesting fact about the natural Linear Programming (LP) relaxation of d-Hitting Set: the support of any optimal LP solution to the LP-relaxation of d-Hitting Set is of size at most $d \cdot \mathsf{frac}$ where frac is the optimum (minimum value) of the

LP [20]. Furthermore, to reduce bit-size rather than only element number, we introduce an "adaptive sampling strategy" that is, to the best of our knowledge, also novel in parameterized complexity. We believe that these ideas will find further applications in kernelization in the future. More information on our methods can be found in the next section.

- **Starting Point: Linear-Element Lossy Kernel for d-Hitting Set.** First, we show that d-Hitting Set admits a $(d - \frac{d-1}{d})$-approximate $d \cdot \mathsf{opt}$-element kernel, where $\mathsf{opt} \leq k$ is the (unknown) optimum (that is, size of smallest solution).[2] For example, when $d = 3$, the approximation ratio is $d - \frac{d-1}{d} = 2\frac{1}{3}$, which is a notable improvement over 3. When $d = 2$, this result encompasses the classic (exact) $2 \cdot \mathsf{opt}$-vertex kernel for Vertex Cover [6, 27]. We also remark that our linear-element lossy kernel for d-Hitting Set is a critical component (used as a black box) in all of our other results.
- **Conceptual Contribution: Lossy Kernelization Protocols.** We extend the notions of lossy kernelization and kernelization protocols[3] to *lossy kernelization protocols*. Roughly speaking, an α-approximate kernelization protocol can perform a bounded in k number of calls (called *rounds*) to an oracle that solves the problem on instances of size (called *call size*) bounded in k, and besides that it runs in polynomial time. Ideally, the number of calls is bounded by a fixed constant, in which case the protocol is called *pure*. Then, if the oracle outputs c-approximate solutions to the instances it is given, the protocol should output a $(c \cdot \alpha)$-approximate solution to the input instance. In particular, a lossy kernel is the special case of a lossy protocol with one oracle call. The *volume* of a lossy kernelization protocol is the sum of the sizes of the calls it performs.
- **Main Contribution: Near-Linear Volume and Pure Lossy Kernelization Protocol for d-Hitting Set.** We remark that the work of Dell and van Melkebeek [10] further asserts that also the existence of an exact (i.e., 1 approximate in our terms) kernelization protocol for d-Hitting Set of volume $\mathcal{O}(k^{d-\epsilon})$ is impossible unless $\mathsf{co\text{-}NP} \subseteq \mathsf{NP}/\,\mathrm{poly}$.
  First, we show that Vertex Cover admits a (randomized) 1.721-approximate kernelization protocol of 2 rounds and call size $\mathcal{O}(k^{1.5})$. This special case is of major interest in itself: Vertex Cover is the most well-studied problem in Parameterized Complexity, and, until now, no result that breaks both bit-size and approximation ratio barriers simultaneously has been known.
  Then, we build upon the ideas exemplified for the case of Vertex Cover to significantly generalize the result: while Vertex Cover corresponds to $d = 2$, we are able to capture *all* choices of d. Thereby, we prove our main result: for any $\epsilon > 0$, d-Hitting Set admits a (randomized) pure $(d - \delta)$-approximate kernelization protocol of call size $\mathcal{O}(k^{1+\epsilon})$. Here, the number of rounds and δ are fixed constants that depend only on d and ϵ. While the improvement over the barrier of d in terms of approximation is minor (though still notable when $d = 2$), it is a *proof of concept* – that is, it asserts that d is not an impassable barrier.[4] Moreover, it does so with almost the best possible (being almost linear) output size.
- **Outlook: Relation to Ruzsa-Szemerédi Graphs.** Lastly, we present a connection between the possible existence of a $(1 + \epsilon)$-approximate kernelization protocol for Vertex Cover of call size $\mathcal{O}(k^{1.5})$ and volume $\mathcal{O}(k^{1.5+o(1)})$ and a known open problem about Ruzsa-Szemerédi graphs. We discuss this result in more detail in Section 3.

[2] In fact, when the parameter is k, we show that the bound is better.
[3] We remark that kernelization protocols are a highly restricted special case of Turing kernels, that yet generalizes kernels.
[4] Possibly, building upon our work, further improvements on the approximation factor (though perhaps at the cost of an increase in the output size) may follow.

Kernels for Implicit 3-Hitting Set Problems. Lastly, we provide better lossy kernels for two well-studied graph problems, namely, Cluster Vertex Deletion and Feedback Vertex Set in Tournaments, which are known to be implicit 3-Hitting Set problems [8]. Notably, both our algorithms are based on some of the ideas and concepts that are part of our previous results, and, furthermore, we believe that the approach underlying the parts common to both these algorithms may be useful when dealing also with other hitting and packing problems of constant-sized objects. In the Cluster Vertex Deletion problem, we are given a graph G and an integer k. The task is to decide whether there exists a set S of at most k vertices of G such that $G - S$ is a cluster graph. Here, a cluster graph is a graph where every connected component is a clique. It is known that this problem can be formulated as a 3-Hitting Set problem where the family $\mathcal{F}$ contains the vertex sets of all *induced* P_3*'s* of G. (An induced P_3 is a path on three vertices where the first and last vertices are non-adjacent in G.) In the Feedback Vertex Set in Tournaments problem, we are given a tournament G and an integer k. The task is to decide whether there is a set S of k vertices such that each directed cycle of G contains a member of S (i.e., $G - S$ is acyclic). It is known that Feedback Vertex Set in Tournaments can be formulated as a 3-Hitting Set problem as well, where the family $\mathcal{F}$ contains the vertex sets of all directed cycles on three vertices (triangles) of G.

In [16], it was shown that Feedback Vertex Set in Tournaments and Cluster Vertex Deletion admit kernels with $\mathcal{O}(k^{\frac{3}{2}})$ vertices and $\mathcal{O}(k^{\frac{5}{3}})$ vertices, respectively. This answered an open question from WorKer 2010 [4, page 4], regarding the existence of kernels with $\mathcal{O}(k^{2-\epsilon})$ vertices for these problems. The question of the existence of linear-vertex kernels for these problems is open. In the realm of approximation algorithms, for Feedback Vertex Set in Tournaments, Cai , Deng and Zang [5] gave a factor 2.5 approximation algorithm, which was later improved to 7/3 by Mnich, Williams and Végh [26]. Recently, Lokshtanov, Misra, Mukherjee, Panolan, Philip and Saurabh [24] gave a 2-approximation algorithm for Feedback Vertex Set in Tournaments. For Cluster Vertex Deletion, You, Wang and Cao [29] gave a factor 2.5 approximation algorithm, which later was improved to 7/3 by Fiorini, Joret and Schaudt [14]. It is open whether Cluster Vertex Deletion admits a 2-approximation algorithm. We remark that both problems admit approximation-preserving reductions from Vertex Cover, and hence they too do not admit $(2 - \epsilon)$-approximation algorithms up to the Unique Games Conjecture.

We provide the following results for Feedback Vertex Set in Tournaments and Cluster Vertex Deletion.

- **Cluster Vertex Deletion.** For any $0 < \epsilon < 1$, the Cluster Vertex Deletion problem admits a $(1 + \epsilon)$-approximate $\mathcal{O}(\frac{1}{\epsilon} \cdot \mathsf{opt})$-vertex kernel.
- **Feedback Vertex Set in Tournaments.** For any $0 < \epsilon < 1$, the Feedback Vertex Set in Tournaments problem admits a $(1 + \epsilon)$-approximate $\mathcal{O}(\frac{1}{\epsilon} \cdot \mathsf{opt})$-vertex kernel.

Reading Guide. First, in Section 2, we present basic terminology regarding lossy kernelization. Due to lack of space, we omit the formal proofs and technical details. Instead, in Section 3, we present an overview of our proofs Afterwards, in Section 4, we conclude the extended abstract with some open problems.

2 Lossy Kernelization: Algorithms and Protocols

Lossy Kernelization Algorithms. We follow the framework of lossy kernelization presented in [25]. Here, we deal only with minimization problems where the value of a solution is its size, and where the computation of an arbitrary solution (where no optimization is enforced) is trivial. Thus, for the sake of clarity of presentation, we only formulate the definitions for this context, and remark that the definitions can be extended to the more general setting in the straightforward way (for more information, see [25]). To present the definitions, consider a parameterized problem Π. Given an instance I of Π with parameter $k = \kappa(I)$, denote: if k is a structural parameter, then $\pi_I(\mathsf{opt}) = \mathsf{opt}$, and otherwise (if k is a bound on the solution size given as part of the input) $\pi_I(\mathsf{opt}) = \min\{\mathsf{opt}, k+1\}$. Moreover, for any solution S to I, denote: if k is a structural parameter, then $\pi_I(S) = |S|$, and otherwise $\pi_I(S) = \min\{|S|, k+1\}$. We remark that when π is irrelevant (e.g., when the parameter is structural), we will drop it. A discussion of the motivation behind this definition of π_I can be found in [25]; here, we only briefly note that it signifies that we "care" only for solutions of size at most k – all other solutions are considered equally bad, treated as having size $k+1$.

▶ **Definition 1.** *Let Π be a parameterized minimization problem. Let $\alpha \geq 1$. An* α-approximate kernelization algorithm *for Π consists of two polynomial-time procedures:* **reduce** *and* **lift**. *Given an instance I of Π with parameter $k = \kappa(I)$,* **reduce** *outputs another instance I' of Π with parameter $k' = \kappa(I')$ such that $|I'| \leq f(k, \alpha)$ and $k' \leq k$. Given I, I' and a solution S' to I',* **lift** *outputs a solution S to I such that $\frac{\pi_I(S)}{\pi_I(\mathsf{opt}(I))} \leq \alpha \frac{\pi_{I'}(S')}{\pi_{I'}(\mathsf{opt}(I'))}$. If $\frac{\pi_I(S)}{\pi_I(\mathsf{opt}(I))} \leq \max\{\alpha, \frac{\pi_{I'}(S')}{\pi_{I'}(\mathsf{opt}(I'))}\}$ holds, then the algorithm is termed* strict.

In case Π admits an α-approximate kernelization algorithm where the output has size $f(k, \alpha)$, or where the output has $g(k, \alpha)$ "elements" (e.g., vertices), we say that Π admits an α-approximate kernel of size $f(k, \alpha)$, or an α-approximate $g(k, \alpha)$-element kernel, respectively. When it is clear from context, we simply write $f(k)$ and $g(k)$. When it is guaranteed that $|I'| \leq f(k', \alpha)$ rather than only $|I'| \leq f(k, \alpha)$, then we say that the lossy kernel is *output-parameter sensitive*.

We only deal with problems that have constant-factor polynomial-time approximation algorithms, and where we may directly work with (the unknown) opt as the parameter (then, π can be dropped). However, working with k (and hence π) has the effect of artificially altering kernel sizes, but not so if one remembers that k and opt are different parameterizations. The following lemma clarifies a relation between these two parameterizations.

▶ **Lemma 2.** *Let Π be a minimization problem that, when parameterized by the optimum, admits an α-approximate kernelization algorithm $\mathfrak{A}$ of size $f(\mathsf{opt})$ (resp., an α-approximate $g(\mathsf{opt})$-element kernel). Then, when parameterized by k, a bound on the solution size that is part of the input, it admits an α-approximate kernelization algorithm $\mathfrak{B}$ of size $f(\frac{k+1}{\alpha})$ (resp., an α-approximate $g(\frac{k+1}{\alpha})$-element kernel).*

Proof. We design $\mathfrak{B}$ as follows. Given an instance (I, k) of Π, **reduce** of $\mathfrak{B}$ calls **reduce** of $\mathfrak{A}$ on I. If the output instance size is at most $f(\frac{k+1}{\alpha})$ (resp., the output has at most $g(\frac{k+1}{\alpha})$ elements), then it outputs this instance with parameter $k' = k$. Otherwise, it outputs a trivial constant-sized instance. Given $(I, k), (I', k')$ and a solution S' to (I', k'), if I' is the output of the **reduce** procedure of $\mathfrak{A}$ on I, then **lift** of $\mathfrak{B}$ calls **lift** of $\mathfrak{A}$ on I, I', S' and outputs the result. Otherwise, it outputs a trivial solution to I.

The **reduce** and **lift** procedures of $\mathfrak{B}$ clearly have polynomial time complexities, and the definition of $\mathfrak{B}$ implies the required size (or element) bound on the output of **reduce**. It remains to prove that the approximation ratio is α. To this end, consider an input $(I, k), (I', k'), S'$ to **lift** of $\mathfrak{B}$. Let S be its output. We differentiate between two cases.

- First, suppose that $\mathsf{opt}(I) \geq \frac{k+1}{\alpha}$. Then, $\frac{\pi_I(S)}{\pi_I(\mathsf{opt}(I))} \leq \frac{k+1}{\frac{k+1}{\alpha}} = \alpha \leq \alpha \frac{\pi_{I'}(S')}{\pi_{I'}(\mathsf{opt}(I'))}$ (where the last inequality follows because $|S'| \geq \mathsf{opt}(I')$ and hence $\pi_{I'}(S') \geq \pi_{I'}(\mathsf{opt}(I'))$).

- Second, suppose that $\mathsf{opt}(I) < \frac{k+1}{\alpha}$. Then, it necessarily holds that I' is the output of the **reduce** procedure of $\mathfrak{A}$ on I. Moreover, note that $\mathsf{opt}(I') \leq \mathsf{opt}(I)$ and $k' = k$. So, if $|S'| \geq k' + 2$, then $\frac{\pi_I(S)}{\pi_I(\mathsf{opt}(I))} \leq \frac{k+1}{\pi_I(\mathsf{opt}(I))} = \frac{k'+1}{\mathsf{opt}(I)} \leq \frac{k'+1}{\mathsf{opt}(I')} = \frac{\pi_{I'}(S')}{\pi_{I'}(\mathsf{opt}(I'))}$. Else, we suppose that $|S'| \leq k' + 1$ and hence $\pi_{I'}(S') = |S'|$. Then,

$$\frac{\pi_I(S)}{\pi_I(\mathsf{opt}(I))} \leq \frac{|S|}{\pi_I(\mathsf{opt}(I))} = \frac{|S|}{\mathsf{opt}(I)} \leq \alpha \frac{|S'|}{\mathsf{opt}(I')} = \alpha \frac{\pi_{I'}(S')}{\pi_{I'}(\mathsf{opt}(I'))}.$$

Here, the second inequality follows because the approximation ratio of $\mathfrak{A}$ is α.

This completes the proof. ◀

Approximate kernelization algorithm often use strict reduction rules, defined as follows.

▶ **Definition 3.** *Let Π be a parameterized minimization problem. Let $\alpha \geq 1$. An α-strict reduction rule for Π consists of two polynomial-time procedures:* **reduce** *and* **lift**. *Given an instance I of Π with parameter $k = \kappa(I)$,* **reduce** *outputs another instance I' of Π with parameter $k' = \kappa(I') \leq k$. Given I, I' and a solution S' to I',* **lift** *outputs a solution S to I such that* $\frac{\pi_I(S)}{\pi_I(\mathsf{opt}(I))} \leq \max\{\alpha, \frac{\pi_{I'}(S')}{\pi_{I'}(\mathsf{opt}(I'))}\}$.

▶ **Proposition 4** ([25]). *Let Π be a parameterized problem. For any $\alpha \geq 1$, an approximate kernelization algorithm for Π that consists only of α-strict reduction rules has approximation ratio α. Furthermore, it is strict.*

Lossy Kernelization Protocols. We extend the notion of lossy kernelization algorithms to lossy kernelization protocols as follows.

▶ **Definition 5** (Lossy Kernelization Protocol). *Let Π be a parameterized minimization problem with parameter k. Let $\alpha \geq 1$. An α-*approximate kernelization protocol *of* call size *$f(k, \alpha)$ and $g(k, \alpha)$* rounds *for Π is defined as follows. First, the protocol assumes to have access to an oracle $\mathfrak{O}$ that, given an instance I' of Π of size at most $f(k, \alpha)$, returns a solution S' to I' such that $\pi_{I'}(S') \leq \beta \pi_{I'}(\mathsf{opt}(I'))$ for minimization and $\pi_{I'}(S') \geq \frac{1}{\beta}\pi_{I'}(\mathsf{opt}(I'))$ for maximization, for some fixed $\beta > 0$. Second, for the same fixed $\beta > 0$, given an instance I of Π, the protocol may perform $g(k, \alpha)$ calls to $\mathfrak{O}$ and other operations in polynomial time, and then output a solution S to I such that $\frac{\pi_I(S)}{\pi_I(\mathsf{opt}(I))} \leq \alpha\beta$.*

The volume *(or* size*) of the protocol is $f(k, \alpha)g(k, \alpha)$. In case $g(k, \alpha) = g(\alpha)$ (i.e., g depends only on α), the protocol is called* pure.

Notice that an α-approximate kernelization algorithm is the special case of an α-approximate kernelization protocol when the number of rounds is 1.

Practically, we think that (lossy) kernelization protocols can often be as useful as standard (lossy) kernels, and, in some cases, more useful. Like standard (lossy) kernels, they reduce the total size of what we need to solve, only that now what we need to solve is split into several instances, to be solved one after another. On the one hand, this relaxation seems to, in most cases, not be restrictive (as what we really care about is the total size of what we need to solve). On the other hand, it might be helpful if by using this relaxation one can achieve better bounds than what is known (or, even, what is possible) on the sizes of the reduced instances, or to simplify the algorithm. For example, for the case of d-Hitting Set, we do not know how to beat $\mathcal{O}(k^d)$ using a lossy kernel rather than a protocol.

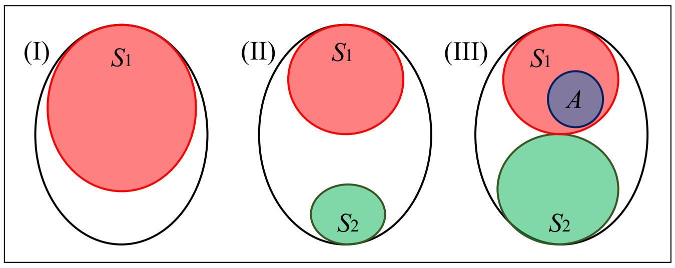

Figure 1 The three cases encountered by our 2-call lossy kernelization protocol for Vertex Cover: (I) $|S_1|$ is large, and we return $V(G)$; (II) $|S_1|$ is small and $|S_2|$ is small, and we return $S_1 \cup S_2$; (III) $|S_1|$ is small and $|S_2|$ is large, and we return $(V(G) \setminus S_1) \cup A$.

3 Overview of Our Proof Ideas

In this section, we present a high-level overview of our proof ideas.

3.1 Linear-Element Lossy Kernel for d-Hitting Set

We make use of a known result about the natural LP relaxation of d-Hitting Set: the support of any optimal LP solution to the LP-relaxation of d-Hitting Set is of size at most $d \cdot \mathsf{frac}$ where frac is the optimum (minimum value) of the LP [20]. For the sake of completeness, we provide a proof. We then provide a lossy reduction rule that computes an optimal LP solution, and deletes all vertices assigned values at least $\frac{1}{d-1}$. Having applied this rule exhaustively, we arrive at an instance having an optimal LP solution that assigns only values strictly smaller than $\frac{1}{d-1}$. Then, it can be shown that all hitting sets are contained within the support of this LP solution. In turn, in light of the aforementioned known result, this yields an approximate $d \cdot \mathsf{frac}$-element and $(d\mathsf{frac})^d$-set kernel that is output-parameter sensitive.

The analysis that the approximation factor is $d - \frac{d-1}{d}$ is slightly more involved, and is based on case distinction. In case the number of vertices deleted is "small enough", the cost of adding them is "small enough" as well. In the more difficult case where the number of vertices deleted is "large", by making use of the already established bound on the output size as well as the drop in the fractional optimum, we are able to show that, in fact, we return a solution of approximation factor $d - \frac{d-1}{d}$ irrespective of the approximation ratio of the solution we are given. More generally, the definition of "small enough" and "large" gives rise to a trade-off that is critical for our kernelization protocol for d-Hitting Set, which in particular yields that we can either obtain a *negligible additive error* or directly a solution of the desired (which is some fixed constant better than d but worse than $d - \frac{d-1}{d}$) approximation ratio. Specifically, this means that it is "safe" to compose our element kernel as part of other kernelization algorithms or protocols.

3.2 2-Round $\mathcal{O}(\mathsf{frac}^{1.5})$-Call Size Lossy Kernelization Protocol for Vertex Cover

Towards the presentation of our near-linear call size lossy kernelization protocol for d-Hitting Set, we abstract some of the ideas using a simpler 2-round $\mathcal{O}(\mathsf{frac}^{1.5})$-call size 1.721-approximate kernelization protocol for Vertex Cover (where $\mathsf{frac} \leq \mathsf{opt} \leq k$ is the optimum of the natural LP relaxation of Vertex Cover). First, we apply an (exact) kernelization algorithm to have a graph G on at most $2\mathsf{frac}$ vertices. The purpose of having only $2\mathsf{frac}$ vertices is twofold. First, it means that to obtain a "good enough" approximate solution, it suffices that we do not pick a "large enough" (linear fraction) of vertices of G to our solution. Second, it is required for a probability bound derived using union bound over vertex subsets to hold. Then, roughly speaking, the utility of the first oracle call is mainly, indirectly, to uncover a "large" (linear in $n = |V(G)|$) induced subgraph of G that is "sparse", and hence can be sent to the second oracle call to be solved optimally.

More precisely, after applying the initial kernelization, we begin by sampling roughly $\mathsf{frac}^{1.5}$ edges from G. Then, we call the oracle on the sampled graph to obtain a solution S_1 to it (but not to G). In case that solution S_1 is "large" compared to the size of the vertex set of G (that is, sufficiently larger than $n/2 \leq \mathsf{frac}$), we can just return the entire vertex set of G (see Fig. 1). Else, we know that the subgraph of the sampled graph that is induced by $V(G) \setminus S_1$ is edgeless. In addition, we can show (due to the initial kernelization) that with high probability, every set of edges of size (roughly) at least $\mathsf{frac}^{1.5}$ that is the edge set of some induced subgraph of G has been hit by our edge sample. Together, this implies that the subgraph of G induced by $V(G) \setminus S_1$ has at most $\mathsf{frac}^{1.5}$ edges, and hence can be solved optimally by a second oracle call. Then, because we know that this subgraph is large compared to G (else S_1 is large), if the oracle returned a "small" solution S_2 to it, we may just take this solution together with S_1 (which will form a vertex cover), and yet not choose sufficiently many vertices so that this will be good enough in terms of the approximation ratio achieved. Else, also because we know that this subgraph is large compared to G, if the second oracle returned a "large" solution S_2, then we know that every optimal solution must take many vertices from this subgraph, and hence, to compensate for this, the optimum of $G[S_1]$ must be "very small". So, we compute a 2-approximate solution A to $G[S_1]$, which we know should not be "too large", and output the union of A and $V(G) \setminus S_1$ (which yields a vertex cover).

3.3 Near-Linear Volume and Pure Lossy Kernelization Protocol for d-Hitting Set

For any fixed $\epsilon > 0$, we present a pure $d(1 - h(d, \epsilon))$-approximate (randomized) kernelization protocol for d-Hitting Set with call size $\mathcal{O}((\mathsf{frac})^{1+\epsilon})$ where $h(d, \epsilon)$ is a fixed positive constant that depends only on d, ϵ. On a high-level, the idea of our more general lossy kernelization protocol is to compute a nested family of solutions based on the approach described above for Vertex Cover (see Fig. 2). Intuitively, as we now can sample only few sets (that is, $\mathsf{frac}^{1+\epsilon}$), when we compute a solution that hits them using an oracle call, the number of sets it misses can still be huge, and hence we will need to iteratively use the oracle (a constant number of times) until we reach a subuniverse such that we can optimally solve the subinstance induced by it by a single oracle call. Below, we give a more elaborate overview.

First, we apply our linear-element lossy kernel to have an instance $I_0 = (U_0, \mathcal{T}_0)$ where the universe U_0 consists of at most $d\mathsf{frac}$ elements. Here, the error of this application is not multiplied by the error attained next, but will only yield (as mentioned earlier) a negligible additive error (or directly a solution of the desired approximation ratio). The purpose of having only $d\mathsf{frac}$ elements is twofold, similarly as it is in the protocol described earlier for Vertex Cover. Afterwards, we begin by sampling a family $\mathcal{F}_1$ of roughly $\mathsf{frac}^{1+\epsilon}$ sets from $\mathcal{T}_0$. Then, we call the oracle on the sampled family $\mathcal{F}_1$ to obtain a solution S_1 to it. In case that solution S_1 is "large" (sufficiently larger than $|U_0|/d \leq \mathsf{frac}$), we can just return U_0. Else, we know that the family of sets corresponding to the subinstance I_1 induced by $U_1 = U_0 \setminus S_1$ – that is, the family of all sets in $\mathcal{T}_0$ contained in U_1, which we denote by $\mathcal{T}_1$ –

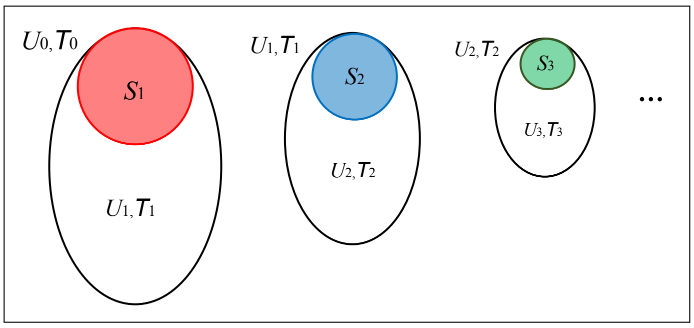

Figure 2 The nested solutions computed by oracle calls in our lossy kernelization protocol for d-Hitting Set. Each S_i is a solution to a subinstance $(U_{i-1}, \mathcal{F}_{i-1})$ sampled from $(U_{i-1}, \mathcal{T}_{i-1})$.

was missed by our set sample. In addition, we can show (due to the initial kernelization) that with high probability, every family of sets of size (roughly) at least $\mathsf{frac}^{d-\epsilon}$ that corresponds to a subinstance induced by a subset of U_0 has been hit by our set sample. Together, this implies that $\mathcal{T}_1$ has at most $\mathsf{frac}^{d-\epsilon}$ (rather than frac^{d}) sets. Hence, in some sense, we have made progress towards the discovery of a sparse subinstance that we can optimally solve.

Due to important differences, let us describe also the second iteration – among at most $\frac{1}{\epsilon}(d-1)$ iterations performed in total – before skipping to the (last) one where we have a subinstance that we can optimally solve by an oracle call. The last iteration may not even be reached, if we find a "good enough" solution earlier. We remark that it is critical to stop and return a solution as soon as we find a "large enough" one by an oracle call[5] as for our arguments to work, we need to always deal with subinstances whose universe is large (a linear fraction of $|U_0|$), and these are attained by removing oracle solutions we got along the way. We begin the second iteration by sampling a family $\mathcal{F}_2$ of roughly $\mathsf{frac}^{1+\epsilon}$ sets from $\mathcal{T}_1$. Then, we call the oracle on the sampled family $\mathcal{F}_2$ to obtain a solution S_2 to it. On the one hand, in case that solution S_2 is "large" (sufficiently larger than $|U_1|/d$), we cannot just return U_0 as in the first iteration, as now it may not be true that the optimum of I_0 is large compared to $|U_0|$. Still, it is true that the optimum of I_1 is large compared to $|U_1|$. So, every optimal solution (to I_0) must take many elements from $U_1 \setminus S_2$, and hence, to compensate for this, the optimum of the subinstance induced by S_1 must be "very small". So, we compute a d-approximate solution to this subinstance, which we know should not be "too large" , and output the union of it and U_1 (which yields a hitting set). On the other hand, in case S_2 is "small", we proceed as follows. We observe that the family of sets corresponding to the subinstance I_2 induced by $U_2 = U_1 \setminus S_2$, whose family of sets we denote by $\mathcal{T}_2$, was missed by our set sample. In addition, we can show (due to the initial kernelization) that with high probability, every family of sets of size (roughly) at least $\mathsf{frac}^{d-2\epsilon}$ that corresponds to a subinstance induced by a subset of U_1 has been hit by our set sample. Together, this implies that $\mathcal{T}_2$ has at most $\mathsf{frac}^{d-2\epsilon}$ (rather than just $\mathsf{frac}^{d-\epsilon}$ as in the first iteration) sets. Hence, in some sense, we have made further progress towards the discovery of a sparse subinstnace that we can optimally solve.

Finally, we arrive at a subinstance I' induced by a subuniverse $U' \subseteq U_0$ that is of size linear in U_0 (else we should have returned a solution earlier) and where the family of sets, $\mathcal{F}'$, is of size at most $\mathsf{frac}^{1+\epsilon}$. Then, we call the oracle on I' to obtain a solution S' to it. On

[5] The solution we return is not the one given by the oracle call, but its union with another solution, as will be clarified immediately, or just U_0 in case of the first iteration describe above.

the one hand, in case that solution S' is "large" (sufficiently larger than $|U'|/d$), we compute a d-approximate solution to the subinstance induced by $U_0 \setminus U'$ (which is the union of all solutions returned by oracle calls except the last one), and output the union of it and U'. Otherwise, we output $(U_0 \setminus U') \cup S'$, which is "good enough" because U' is sufficiently large while S' is sufficiently small compared to it, it does not contain a "large enough" number of elements from U_0.

3.4 Outlook: Relation to Ruzsa-Szemerédi Graphs

A graph G is an (r,t)-*Ruzsa-Szemerédi graph* if its edge set can be partitioned into t edge-disjoint induced matchings, each of size r. These graphs were introduced in 1978 [28], and have been extensively studied since then. When r is a function of n, let $\gamma(r)$ denote the maximum t (which is a function of n) such that there exists an (r,t)-Ruzsa-Szemerédi graph. In [19], the authors considered the case where $r = cn$. They showed that when $c = \frac{1}{4}$, $\gamma(r) \in \Theta(\log n)$, and when $\frac{1}{5} \leq c \leq \frac{1}{4}$, $t \in \mathcal{O}(\frac{n}{\log n})$. It is an open problem whether whenever c is a fixed constant, $t \in \mathcal{O}(n^{1-\epsilon})$. For any fixed constant $0 < c < \frac{1}{4}$, we present a $(1+4c)$-approximate (randomized) kernelization protocol for Vertex Cover with $t+1$ rounds and call size $\mathcal{O}(t(\mathsf{frac})^{1.5})$. Clearly, this result makes sense only when $t \in o(\sqrt{n})$, preferably $t \in \mathcal{O}(n^{\frac{1}{2}-\lambda})$ for λ as close to $1/2$ as possible, because the volume is $\mathcal{O}(\mathsf{opt}^{2-\lambda})$. If t is "sufficiently small" (depending on the desired number of rounds) whenever c is a fixed constant (specifically, substitute $c = \frac{\epsilon}{4}$), this yields a $(1+\epsilon)$-approximate kernelization protocol.

We observe that, for a graph G, $r = r(n), t = t(n) \in \mathbb{N}$ and $U_1, U_2, \ldots, U_t \subseteq V(G)$ such that for all $i \in \{1,2,\ldots,t\}$, $G[U_i]$ has a matching M_i of size at least r, and for all distinct $i,j \in \{1,2,\ldots,t\}$, $E(G[U_i]) \cap E(G[U_j]) = \emptyset$, we have that G is a supergraph of an (r,t)-Ruzsa-Szemerédi graph. Having this observation in mind, we devise our protocol as follows. After applying an exact $2\mathsf{frac}$-vertex kernel, we initialize $E' = \emptyset$, and we perform $t+1$ iterations of the following procedure. We sample a set of roughly $\mathsf{frac}^{1.5}$ edges from G, and call the oracle on the subgraph of G whose edge set is the set of samples edges union E' to obtain a solution S to it (but not to G), and compute a maximal matching M in $G - S$. If $|M|$ is smaller than $cn \leq 2c\mathsf{frac}$, then we return the union of the set of vertices incident to edges in M (which is a solution to $G-S$) and S. Else, similarly to the first protocol we described for Vertex Cover, we can show that with high probability, $G-S$ has (roughly) at most $k^{1.5}$ edges, and we add this set of edges to E'. The crux of the proof is in the argument that, at the latest, at the $(t+1)$-st iteration the computed matching will be of size smaller than $cn \leq 2c\mathsf{frac}$, as otherwise we can use the matchings we found, together with the vertex sets (of the form $G-S$) we found them in, to construct an $(r,t+1)$-Ruzsa-Szemerédi graph based on the aforementioned observation, which contradicts the choice of t.

3.5 $(1+\epsilon)$-Approximate $\mathcal{O}(\frac{1}{\epsilon} \cdot \mathsf{opt})$-Vertex Kernel for Implicit 3-Hitting Set Problems

Both of our lossy kernels share a common scheme, which might be useful to derive $(1+\epsilon)$-approximate linear-vertex kernels for other implicit hitting and packing problems as well. Essentially, they both consist of two rules (although in the presentation, they are merged for simplicity). To present them, we remind that a module (in a graph) is a set of vertices having the same neighborhood relations with all vertices outside the set. Now, our first rule reveals some modules in the graph, and our second rule shrinks their size. The first rule in both of our lossy kernels is essentially the same.

Now, we elaborate on the first rule. We start by computing an optimal solution α to the LP-relaxation of the corresponding 3-Hitting Set problem. Notice that $\mathsf{support}(\alpha)$ is a solution, and its size is at most $3\mathsf{frac}$ (in fact, we show that it is at most $3\mathsf{frac} - 2|\alpha^{-1}(1)|$). Then, the first rule is as follows. At the beginning, no vertex is marked. Afterwards, one-by-one, for each vertex v assigned 1 by α (i.e., which belongs to $\alpha^{-1}(1)$), we construct a graph whose vertex set is the set of yet unmarked vertices in $V(G) \setminus \mathsf{support}(\alpha)$ and where there is an edge between every two vertices that create an obstruction together with v (that is, an induced P_3 in Cluster Vertex Deletion and a triangle in Feedback Vertex Set in Tournaments). We compute a maximal matching in this graph, and decrease its size to $\frac{1}{\epsilon}$ if it is larger (in which case, it is no longer maximal). The vertices incident to the edges in the matching are then considered marked. We prove that among the vertices in $\alpha^{-1}(1)$ whose matching size was decreased, whose set is denoted by D, any solution can only exclude an ϵ fraction of its size among the vertices in D, and hence it is "safe" (in a lossy sense) to delete D. Let M be the set of all marked vertices. Then, we show that $(\mathsf{support}(\alpha) \cup M) \setminus \{v\}$, for any $v \in \mathsf{support}(\alpha)$ (including those not in $\alpha^{-1}(1)$), is also a solution.

For Cluster Vertex Deletion, we prove that the outcome of the first rule means that the vertex set of every clique in $G - (\mathsf{support}(\alpha) \cup M)$ is a module in $G - D$, and that for every vertex in $\mathsf{support}(\alpha)$, the set of its neighbors in $V(G - (\mathsf{support}(\alpha) \cup M))$ is the vertex set of exactly one of these cliques. So, for Cluster Vertex Deletion, this gives rise to the following second reduction rule (which is, in fact, exact) to decrease the size of module. For every clique among the aforementioned cliques whose size is larger than that of its neighborhood, we arbitrarily remove some of its vertices so that its size will be equal to the size of its neighborhood. This rule is safe since if at least one of the vertices in such a clique is deleted by a solution, then because it is a module, either that deletion is irrelevant or the entire clique is deleted, and in the second case we might just as well delete its neighborhood instead. Because the neighborhoods of the cliques are pairwise-disjoint (since for every vertex in $\mathsf{support}(\alpha)$, the set of its neighbors in $V(G - (\mathsf{support}(\alpha) \cup M))$ is the vertex set of exactly one of the cliques), this means that now their total size is at most $(\mathsf{support}(\alpha) \setminus D) \cup M$, and hence we arrive at the desired kernel.

For Feedback Vertex Set in Tournaments, we consider the unique (because G is a tournament) topological ordering of the vertices in $G - \mathsf{support}(\alpha)$, so that all arcs are "forward" arcs. We prove that the outcome of the first rule means that each vertex $v \in \mathsf{support}(\alpha)$ has a unique position within this ordering when restricted to $G - (\mathsf{support}(\alpha) \cup M)$, so that still all arcs (that is, including those incident to v) are forward arcs in $G - (\mathsf{support}(\alpha) \cup M) \cup \{v\}$. (Further, the vertex set of each subtournament induced by the vertices "between" any two marked vertices in $G - \mathsf{support}(\alpha)$ is a module in $G - D$.) We are thus able to characterize all triangles in $G - D$ as follows: each either consists of three vertices in $(\mathsf{support}(\alpha) \setminus D) \cup M$, or it consists of a vertex $v \in \mathsf{support}(\alpha) \setminus D$, a vertex $u \in (\mathsf{support}(\alpha) \setminus D) \cup M$ and a vertex $w \in V(G) \setminus (\mathsf{support}(\alpha) \cup M)$ with a backward arc between v and u and where w is "in-between" the positions of v and u. This gives rise to a reduction rule for module shrinkage whose presentation and analysis are more technical than that of Cluster Vertex Deletion (in particular, unlike the second rule of Cluster Vertex Deletion, the second rule of Feedback Vertex Set in Tournaments is lossy) and of the first rule; hence, due to lack of space, we omit them.

4 Conclusion

In this paper, we presented positive results on the kernelization complexity of d-Hitting Set, as well as for its special cases Cluster Vertex Deletion and Feedback Vertex Set in Tournaments. First, we proved that if we allow the kernelization to be *lossy* with a qualitatively better loss than the best possible approximation ratio of polynomial time approximation algorithms, then one can obtain kernels where the number of elements is linear for every fixed d. Further, we extended the notion of lossy kernelization algorithms to *lossy kernelization protocols* and, then, presented our main result: For any $\epsilon > 0$, d-Hitting Set admits a (randomized) pure $(d-\delta)$-approximate kernelization protocol of call size $\mathcal{O}(k^{1+\epsilon})$. Here, the number of rounds and δ are fixed constants (that depend only on d and ϵ). Finally, we complemented the aforementioned results as follows: for the special cases of 3-Hitting Set, namely, Cluster Vertex Deletion and Feedback Vertex Set in Tournaments, we showed that for any $0 < \epsilon < 1$, they admits a $(1+\epsilon)$-approximate $\mathcal{O}(\frac{1}{\epsilon} \cdot \mathsf{opt})$-vertex kernel.

We conclude the paper with a few interesting open problems.

1. Does d-Hitting Set admit a kernel with $f(d) \cdot k^{d-1-\epsilon}$ elements for some fixed $\epsilon > 0$, or, even, with just $f(d) \cdot k$ elements?
2. Does d-Hitting Set admit a $(1+\epsilon)$-approximate $\mathcal{O}(f(\epsilon) \cdot k)$-element kernel (or protocol)?
3. Does d-Hitting Set admit a $(1+\epsilon)$-approximate $\mathcal{O}(f(\epsilon) \cdot k)$-bits kernel (or protocol)?
4. Do Feedback Vertex Set in Tournaments and Cluster Vertex Deletion admit linear vertex kernels?
5. Are lossy kernelization protocols "more powerful" than lossy kernelization algorithms?

References

1 Faisal N. Abu-Khzam. A kernelization algorithm for d-Hitting Set. *J. Comput. Syst. Sci.*, 76(7):524–531, 2010. doi:10.1016/j.jcss.2009.09.002.

2 Stéphane Bessy, Fedor V. Fomin, Serge Gaspers, Christophe Paul, Anthony Perez, Saket Saurabh, and Stéphan Thomassé. Kernels for feedback arc set in tournaments. *J. Comput. Syst. Sci.*, 77(6):1071–1078, 2011.

3 Hans L. Bodlaender, Rodney G. Downey, Michael R. Fellows, and Danny Hermelin. On problems without polynomial kernels. *Journal of Computer and System Sciences*, 75(8):423–434, 2009.

4 Hans L. Bodlaender, Fedor V. Fomin, and Saket Saurabh. Open problems, worker 2010. *Available at* *http://fpt.wikidot.com/open-problems*, 2010.

5 Mao-cheng Cai, Xiaotie Deng, and Wenan Zang. An approximation algorithm for feedback vertex sets in tournaments. *SIAM J. Comput.*, 30(6):1993–2007, 2000.

6 Jianer Chen, Iyad A Kanj, and Weijia Jia. Vertex cover: further observations and further improvements. *Journal of Algorithms*, 41(2):280–301, 2001.

7 Marek Cygan, Fedor V. Fomin, Bart MP Jansen, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. Open problems for fpt school 2014. *URL: http://fptschool. mimuw. edu. pl/opl. pdf*, 2014.

8 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.

9 Holger Dell and Dániel Marx. Kernelization of packing problems. In *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 68–81, 2012.

10 Holger Dell and Dieter van Melkebeek. Satisfiability allows no nontrivial sparsification unless the polynomial-time hierarchy collapses. *Journal of the ACM*, 61(4):23:1–23:27, 2014.

11 Michael Dom, Jiong Guo, Falk Hüffner, Rolf Niedermeier, and Anke Truß. Fixed-parameter tractability results for feedback set problems in tournaments. *J. Discrete Algorithms*, 8(1):76–86, 2010.

12 Andrew Drucker. New limits to classical and quantum instance compression. *SIAM Journal on Computing*, 44(5):1443–1479, 2015.

13 Michael R. Fellows, Christian Knauer, Naomi Nishimura, Prabhakar Ragde, Frances A. Rosamond, Ulrike Stege, Dimitrios M. Thilikos, and Sue Whitesides. Faster fixed-parameter tractable algorithms for matching and packing problems. *Algorithmica*, 52(2):167–176, 2008.

14 Samuel Fiorini, Gwenaël Joret, and Oliver Schaudt. Improved approximation algorithms for hitting 3-vertex paths. In *Integer Programming and Combinatorial Optimization - 18th International Conference, IPCO 2016, Liège, Belgium, June 1-3, 2016, Proceedings*, volume 9682 of *Lecture Notes in Computer Science*, pages 238–249, 2016.

15 Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer-Verlag, Berlin, 2006.

16 Fedor V. Fomin, Tien-Nam Le, Daniel Lokshtanov, Saket Saurabh, Stéphan Thomassé, and Meirav Zehavi. Subquadratic kernels for implicit 3-Hitting Set and 3-Set Packing problems. *ACM Trans. Algorithms*, 15(1):13:1–13:44, 2019. `doi:10.1145/3293466`.

17 Fedor V. Fomin, Daniel Lokshtanov, Saket Saurabh, and Meirav Zehavi. *Kernelization. Theory of parameterized preprocessing*. Cambridge University Press, Cambridge, 2019.

18 Lance Fortnow and Rahul Santhanam. Infeasibility of instance compression and succinct PCPs for NP. *Journal of Computer and System Sciences*, 77(1):91–106, 2011.

19 Jacob Fox, Hao Huang, and Benny Sudakov. On graphs decomposable into induced matchings of linear sizes. *Bulletin of the London Mathematical Society*, 49(1):45–57, 2017.

20 Zoltán Füredi. Matchings and covers in hypergraphs. *Graphs and Combinatorics*, 4(1):115–206, 1988.

21 Danny Hermelin and Xi Wu. Weak compositions and their applications to polynomial lower bounds for kernelization. In *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 104–113, 2012.

22 Subhash Khot and Oded Regev. Vertex cover might be hard to approximate to within 2-epsilon. *J. Comput. Syst. Sci.*, 74(3):335–349, 2008. `doi:10.1016/j.jcss.2007.06.019`.

23 Stefan Kratsch and Magnus Wahlström. Representative sets and irrelevant vertices: New tools for kernelization. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 450–459, 2012. `doi:10.1109/FOCS.2012.46`.

24 Daniel Lokshtanov, Pranabendu Misra, Joydeep Mukherjee, Fahad Panolan, Geevarghese Philip, and Saket Saurabh. 2-approximating feedback vertex set in tournaments. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 1010–1018, 2020.

25 Daniel Lokshtanov, Fahad Panolan, M. S. Ramanujan, and Saket Saurabh. Lossy kernelization. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 224–237, 2017.

26 Matthias Mnich, Virginia Vassilevska Williams, and László A. Végh. A 7/3-approximation for feedback vertex sets in tournaments. In *24th Annual European Symposium on Algorithms, ESA 2016, August 22-24, 2016, Aarhus, Denmark*, volume 57 of *LIPIcs*, pages 67:1–67:14, 2016.

27 George L Nemhauser and Leslie Earl Trotter. Properties of vertex packing and independence system polyhedra. *Mathematical programming*, 6(1):48–61, 1974.

28 Imre Z Ruzsa and Endre Szemerédi. Triple systems with no six points carrying three triangles. *Combinatorics (Keszthely, 1976), Coll. Math. Soc. J. Bolyai*, 18:939–945, 1978.

29 Jie You, Jianxin Wang, and Yixin Cao. Approximate association via dissociation. *Discrete Applied Mathematics*, 219:202–209, 2017.