Diminishable Parameterized Problems and Strict Polynomial Kernelization

Henning Fernau[1], Till Fluschnik[2(✉)], Danny Hermelin[3], Andreas Krebs[4], Hendrik Molter[2], and Rolf Niedermeier[2]

[1] Fachbereich 4 – Abteilung Informatik, Universität Trier, Trier, Germany
fernau@uni-trier.de
[2] Institut für Softwaretechnik und Theoretische Informatik, TU Berlin, Berlin, Germany
{till.fluschnik,h.molter,rolf.niedermeier}@tu-berlin.de
[3] Ben Gurion University of the Negev, Beersheba, Israel
hermelin@bgu.ac.il
[4] Wilhelm-Schickard-Institut für Informatik, Universität Tübingen, Tübingen, Germany
krebs@informatik.uni-tuebingen.de

Abstract. Kernelization—a mathematical key concept for provably effective polynomial-time preprocessing of NP-hard problems—plays a central role in parameterized complexity and has triggered an extensive line of research. In this paper we consider a restricted yet natural variant of kernelization, namely *strict kernelization*, where one is not allowed to increase the parameter of the reduced instance (the kernel) by more than an additive constant. Building on earlier work of Chen, Flum, and Müller [CiE 2009, Theory Comput. Syst. 2011], we underline the applicability of their framework by showing that a variety of fixed-parameter tractable problems, including graph problems and Turing machine computation problems, does not admit strict polynomial kernels under the weaker assumption of P $\neq$ NP. Finally, we study a relaxation of the notion of strict kernels.

Work initiated by the research retreat of the Theoretical Computer Science group of the Universität of Tübingen in Sulz (Neckar), September 2016.

T. Fluschnik—Supported by the DFG, project DAMM (NI 369/13) and project TORE (NI 369/18).

D. Hermelin—Supported by the People Programme (Marie Curie Actions) of the European Union's Seventh Framework Programme (FP7/2007-2013) under REA grant agreement number 631163.11, and by the ISRAEL SCIENCE FOUNDATION (grant No. 551145/14). Also supported by a DFG Mercator fellowship, project DAMM (NI 369/13) while staying at TU Berlin (August 2016).

A. Krebs—Supported by the DFG Emmy Noether program (KR 4042/2).

H. Molter—Partially supported by the DFG, project DAPA (NI 369/12).

F. Manea et al. (Eds.): CiE 2018, LNCS 10936, pp. 161–171, 2018.
https://doi.org/10.1007/978-3-319-94418-0_17

1 Introduction

Kernelization is one of the most fundamental concepts in the field of parameterized complexity analysis. Given an instance $(x,k) \in \{0,1\}^* \times \mathbb{N}$ of some parameterized problem L (we assume the parameter to be encoded in unary), a *kernelization* for L produces in polynomial time an instance (x',k') satisfying: $(x',k') \in L \iff (x,k) \in L$ and $|x'| + k' \leq f(k)$ for some fixed computable function $f(k)$. In this way, kernelization can be thought of as a preprocessing procedure that reduces an instance to its "computationally hard core" (*i.e.*, the *kernel*). The function $f(k)$ is accordingly called the *size* of the kernel, and it is typically the measure that one wishes to minimize. Kernelization is a central concept in parameterized complexity not only as an important algorithmic tool, but also because it provides an alternative definition of *fixed-parameter tractability* (FPT) [6]. An algorithm with running time $f(k) \cdot |x|^{O(1)}$ for a parameterized problem L implies that L has a kernel of size $f(k)$, but in the converse direction one cannot always take the same function f. The goal of minimizing the size of the problem kernel leads to the question of what is the kernel with the smallest size one can obtain in polynomial time for a given problem. In particular, do all fixed-parameter tractable problems have small kernels, say, of linear or polynomial size?

The latter question was answered negatively [5,14] by proving that various problems in FPT do not admit a polynomial-size kernel (*polynomial kernel* for short) under the assumption that NP $\not\subseteq$ coNP/poly. The framework has been extended in several directions [18]. Regardless, all of these frameworks rely on the assumption that NP $\not\subseteq$ coNP/poly, an assumption that, while widely believed in the community, is a much stronger assumption than P $\neq$ NP.

Throughout the years, researchers have considered different variants of kernelization such as *fidelity-preserving preprocessing* [11], *partial kernelization* [3], *lossy kernelization* [20], and *Turing kernelization* [4,21]. In this paper, we consider a variant which has been considered previously quite a bit, which is called *proper kernelization* [1] or *strict kernelization* [8]:

Definition 1 (Strict Kernel). *A* strict kernelization *for a parameterized problem L is a polynomial-time algorithm that on input $(x,k) \in \{0,1\}^* \times \mathbb{N}$ outputs $(x',k') \in \{0,1\}^* \times \mathbb{N}$, the* strict kernel*, satisfying: (i) $(x,k) \in L \iff (x',k') \in L$, (ii) $|x'| \leq f(k)$, for some function f, and (iii) $k' \leq k + c$, for some constant c. We say that L admits a strict* polynomial *kernelization if $f(k) \in k^{O(1)}$.*

Thus, a strict kernelization is a kernelization that does not increase the output parameter k' by more than an additive constant. While the term "strict" in the definition above makes sense mathematically, it is actually quite harsh from a practical perspective. Indeed, most of the early work on kernelization involved applying so-called *data reduction rules* that rarely ever increase the parameter value (see *e.g.* the surveys [15,18]). Furthermore, strict kernelization is clearly preferable to kernelizations that increase the parameter value in a dramatic way:

Often a fixed-parameter algorithm on the resulting problem kernel is applied, whose running time highly depends on the value of the parameter, and so a kernelization that substantially increases the parameter value might in fact be useless. Finally, the equivalence with FPT is preserved: A parameterized problem is solvable in $f(k) \cdot |x|^{O(1)}$ time if and only if it has a strict kernel of size $g(k)$ (where f and g are some computable functions).

Chen *et al.* [8] showed that Rooted Path, the problem of finding a simple path of length k in a graph that starts from a prespecified root vertex, parameterized by k has no strict polynomial kernel unless P = NP. They also showed a similar result for CNF-Sat parameterized by the number of variables. Both of these results seemingly are the only known polynomial kernel lower bounds that rely on the assumption of P $\neq$ NP (see Chen *et al.* [7] for a few linear lower bounds that also rely on P $\neq$ NP). The goal of our work is to show that Chen *et al.*'s framework applies for more problems and is easy to extend.

Our Results. We build on the work of Chen *et al.* [8], and further develop and widen the framework they presented for excluding strict polynomial kernels. Using this extended framework, we show that several natural parameterized problems in FPT have no strict polynomial kernels under the assumption that P $\neq$ NP. The main result of our work is given in Theorem 2 below. Note that we use the brackets in the problem names to denote the parameter under consideration.[1]

Theorem 2. *Unless* P = NP*, each of the following fixed-parameter tractable problems does not admit a strict polynomial kernel:*

- Multicolored Path(k) *and* Multicolored Path($k \log n$)*;*
- Clique(Δ)*,* Clique(tw)*,* Clique(bw)*, and* Clique(cw)*;*
- Biclique(Δ)*,* Biclique(tw)*,* Biclique(bw)*, and* Biclique(cw)*;*
- Colorful Graph Motif(k) *and* Terminal Steiner Tree($k + |T|$)*;*
- Multi-Component Annotated Defensive Alliance(k) *and* Multi-Component Annotated Vertex Cover(k)*;*
- Short NTM Computation($k+|\Sigma|$), Short NTM Computation($k+|Q|$), *and* Short Binary NTM Computation(k).

(Herein, k denotes the solution size, n, Δ, tw*,* bw*, and* cw *denote the number of vertices, the maximum vertex degree, the treewidth, bandwidth, and cutwidth of the graph, respectively, T denotes the set of terminals, $|\Sigma|$ denotes the alphabet size, and $|Q|$ denotes the number of states.)*

Finally, we also explore how "tight" the concept of strict polynomial kernels is. We modify the framework for "less" strict kernels, and, employing the Exponential Time Hypothesis (ETH), conclude that we often cannot hope for significantly relaxing the concept of strict kernelization to achieve a comparable list of such analogous kernel lower bounds under P $\neq$ NP. Notably, our

[1] For a complete list of problem definitions we refer to a long version [12] of the paper.

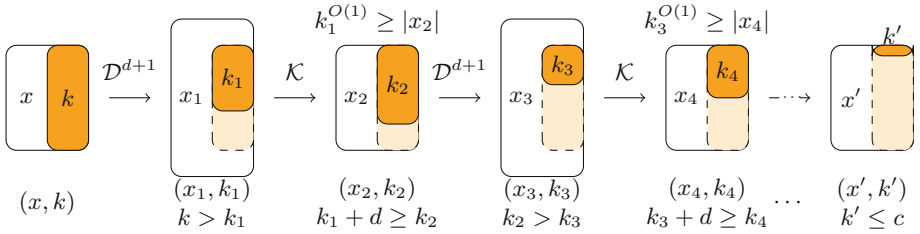

Fig. 1. Illustration to the proof of Theorem 4 for an input instance (x, k). Herein, $\mathcal{K}$ denotes the strict kernelization with additive constant d and $\mathcal{D}$ denotes the parameter diminisher. We represent each instance by boxes: the size of a box symbolizes the size of the instance or the value of the parameter (each dashed box refers to k).

modified framework herein was recently applied [13] for proving the first direct kernelization lower bounds for polynomial-time solvable problems.

We remark that for the problems we discuss in this paper, one can exclude polynomial kernels under the assumption that NP $\not\subseteq$ coNP/poly using the existing frameworks [5,18]. In contrast, our results base on a weaker assumption, but exclude a more restricted version of polynomial kernels. Hence, our results are incomparable with the existing no-polynomial-kernel results.

Notation. We use basic notation from parameterized complexity [10] and graph theory [9]. Let $G = (V, E)$ be an undirected graph. For $W \subseteq V$, let $G - W := (V \setminus W, \{e \in E \mid e \cap W = \emptyset\})$. If $W = \{v\}$, then we write $G - v$. For $v \in V$, we denote by $N_G(v) := \{w \in V \mid \{v, w\} \in E\}$ the neighborhood of v in G. We denote by $[\ell]$, $\ell \in \mathbb{N}$, the set $\{1, \ldots, \ell\}$ and by log the logarithm with base two.

2 Framework

In this section we present the general framework used throughout the paper. Firstly, we define the central notion of a *parameter diminisher* referring to *parameter decreasing polynomial reduction* introduced by Chen *et al.* [8].

Definition 3 (Parameter Diminisher). *A* parameter diminisher *for a parameterized problem L is a polynomial-time algorithm that maps instances $(x, k) \in \{0,1\}^* \times \mathbb{N}$ to instances $(x', k') \in \{0,1\}^* \times \mathbb{N}$ such that (i) $(x, k) \in L$ if and only if $(x', k') \in L$ and (ii) $k' < k$.*

We call a parameterized problem L *diminishable* if there is a parameter diminisher for L. The following theorem was proved initially by Chen *et al.* [8], albeit for slightly weaker forms of diminisher and strict polynomial kernels.

Theorem 4 ([8]). *Let L be a parameterized problem such that its unparameterized version is* NP*-hard and $\{(x, k) \in L \mid k \leq c\} \in$ P*, for some constant c. If L is diminishable and admits a strict polynomial kernel, then* P = NP*.*

The idea behind Theorem 4 is to repeat the following two procedures until the parameter value drops below c (see Fig. 1 for an illustration). First, apply the parameter diminisher a constant number of times such that when, second, the strict polynomial kernelization is applied, the parameter value is decreased. The strict polynomial kernelization keeps the instances small, hence the whole process runs in polynomial time.

Reductions transfer diminishability from one parameterized problem to another if they do not increase the parameter value and run in polynomial time. Formally, given two parameterized problems L with parameter k and L' with parameter k', a *parameter-non-increasing reduction* from L to L' is an algorithm that maps each instance (x, k) of L to an equivalent instance (x', k') of L' in $\text{poly}(|x| + k)$ time such that $k' \leq k$. Note that to transfer diminishability, we need parameter-non-increasing reductions between two parameterized problems in both directions—a crucial difference to other reduction-based hardness results.

Lemma 5 ($\star$[2]). *Let L_1 and L_2 be two parameterized problems such that there are parameter-non-increasing reductions from L_1 to L_2 and from L_2 to L_1. Then we have that L_1 is diminishable if and only if L_2 is diminishable.*

Parameter-Decreasing Branching and Strict Composition. To construct parameter diminishers, it is useful to follow a "branch and compose" technique: Herein, first *branch* into several subinstances while decreasing the parameter value in each, and then *compose* the subinstances into one instance without increasing the parameter value by more than an additive constant. We first give the definitions and then show that both combined form a parameter diminisher.

A *parameter-decreasing branching rule* for a parameterized problem L is a polynomial-time algorithm that on input $(x, k) \in \{0,1\}^* \times \mathbb{N}$ outputs a sequence of instances $(y_1, k'), \ldots, (y_t, k') \in \{0,1\}^* \times \mathbb{N}$ such that $(x, k) \in L \iff (y_i, k') \in L$ for at least one $i \in [t]$ and $k' < k$. Composition is the core concept behind the standard kernelization lower bound framework introduced by Bodlaender *et al.* [5]. Here we use a more restrictive notion of this concept: A *strict composition* for a parameterized problem L is an algorithm that receives as input t instances $(x_1, k), \ldots, (x_t, k) \in \{0,1\}^* \times \mathbb{N}$, and outputs in polynomial time a single instance $(y, k') \in \{0,1\}^* \times \mathbb{N}$ such that (i) $(y, k') \in L \iff (x_i, k) \in L$ for some $i \in [t]$ and (ii) $k' \leq k + c$ for some constant c. If we now combine (multiple applications of) a parameter-decreasing branching rule with a strict composition, then we get a parameter diminisher. We remark that this also holds if we require in both definitions that the equivalence holds for all $i \in [t]$.

Lemma 6 ($\star$). *Let L be a parameterized problem. If L admits a parameter-decreasing branching rule and a strict composition, then it is diminishable.*

[2] Full proofs of results marked with ($\star$) are deferred to a long version [12] of the paper.

On the Exclusion of Non-Uniform Kernelization Algorithms. The presented framework can be easily adapted to exclude different forms of strict kernelizations. As our example, we show that the framework can be used to exclude strict polynomial kernels computed in *non-uniform* polynomial time (the corresponding complexity class is called P/poly) under the assumption that NP $\not\subseteq$ P/poly. A *non-uniform strict kernelization* for a parameterized problem L is a *non-uniform* polynomial-time algorithm that on input $(x,k) \in \{0,1\}^* \times \mathbb{N}$ outputs $(x',k') \in \{0,1\}^* \times \mathbb{N}$, the *strict kernel*, satisfying: (i) $(x,k) \in L \iff (x',k') \in L$, (ii) $|x'| \leq f(k)$, for some function f, and (iii) $k' \leq k + c$, for some constant c. We say that L admits a non-uniform strict *polynomial* kernelization if $f(k) \in k^{O(1)}$.

Proposition 7 ($\star$). *Let L be a parameterized problem such that its unparameterized version is* NP*-hard and we have that $\{(x,k) \in L \mid k \leq c\} \in$* P/poly*, for some constant c. If L is diminishable and admits a non-uniform strict polynomial kernel, then* NP $\subseteq$ P/poly*.*

We remark that if NP $\subseteq$ P/poly, then the Polynomial Hierarchy collapses to its second level [17] (note that NP $\subseteq$ coNP/poly implies a collapse in the Polynomial Hierarchy to its third level).

3 Problems Without Strict Polynomial Kernels

In this section, we exemplify the proof of Theorem 2 on two selected graph problems. The complete proof of Theorem 2 can be found in a long version [12].

First, we present a diminisher for the Multicolored Path(k) problem: Given an undirected graph $G = (V,E)$ with a vertex coloring function $\mathrm{col} : V \to [k]$, determine whether there exists a simple path of length k containing one vertex of each color. This problem is NP-complete as it generalizes Hamiltonian Path. Furthermore, Multicolored Path(k) is fixed-parameter tractable as it can be solved in $2^{O(k)}n^2$ time [2]. The idea used in the parameter diminisher for Multicolored Path(k) can also be applied for the Colorful Graph Motif problem, a problem with applications in bioinformatics.

Proposition 8. Multicolored Path(k) *is diminishable.*

For graph problems, a vertex-coloring seems to help to construct diminishers. As an example, the diminishability of the (uncolored) Path(k) problem, asking whether a given graph contains a simple path of length k, remains open.

Proof. We give a parameter-decreasing branching rule and a strict composition for Multicolored Path(k). The result then follows from Lemma 6. Let $(G = (V,E), \mathrm{col})$ be an instance of Multicolored Path(k). Our parameter-decreasing branching rule for $(G = (V,E), \mathrm{col})$ creates a graph $G_{(v_1,v_2,v_3)}$ for each ordered triplet (v_1,v_2,v_3) of vertices of V such that v_1,v_2,v_3 is a multicolored path in G. The graph $G_{(v_1,v_2,v_3)}$ is constructed from G as follows: Delete from G all vertices $w \in V \setminus \{v_2,v_3\}$ with $\mathrm{col}(w) \in \{\mathrm{col}(v_1), \mathrm{col}(v_2), \mathrm{col}(v_3)\}$.

Following this, only vertices of $k-1$ colors remain, and v_2 and v_3 are the only vertices colored $\mathrm{col}(v_2)$ and $\mathrm{col}(v_3)$, respectively. Then delete all edges incident with v_2, apart from $\{v_2, v_3\}$, and relabel all colors so that the image of col for $G_{(v_1,v_2,v_3)}$ is $[k-1]$.

Clearly our parameter-decreasing branching rule can be performed in polynomial time. Furthermore, the parameter decreases in each output instance. We show that the first requirement holds as well: Indeed, suppose that G has a multicolored path $v_1, v_2, \ldots, v_k$ of length k. Then $v_2, \ldots, v_k$ is a multicolored path of length $k-1$ in $G_{(v_1,v_2,v_3)}$ by construction. Conversely, suppose that there is a multicolored path $u_2, \ldots, u_k$ of length $k-1$ in some $G_{(v_1,v_2,v_3)}$. Then since v_2 is the only vertex of color $\mathrm{col}(v_2)$ in $G_{(v_1,v_2,v_3)}$, and since v_2 is only adjacent to v_3, it must be without loss of generality that $u_2 = v_2$ and $u_3 = v_3$. Hence, since v_1 is adjacent to v_2 in G, and no vertices of $u_2, \ldots, u_k$ have color $\mathrm{col}(v_1)$ in G, the sequence of $v_1, u_2, \ldots, u_k$ forms a multicolored path of length k in G.

Our strict composition for Multicolored Path(k) is as follows. Given a sequence of inputs $(G_1, \mathrm{col}_1), \ldots, (G_t, \mathrm{col}_t)$, the strict composition constructs the disjoint union G and the coloring function col of all graphs G_i and coloring functions col_i, $1 \leq i \leq t$. Clearly, (G, col) contains a multicolored path of length k if and only if there is a multicolored path of length k in some (G_i, col_i). The result thus follows directly from Lemma 6. □

Proposition 9 (⋆). *Unless* P = NP, Multicolored Path$(k \log n)$ *has no strict polynomial kernel.*

We next consider the NP-complete Terminal Steiner Tree (TST) [19] problem: given an undirected graph $G = (V = N \uplus T, E)$ (T is called the terminal set) and a positive integer k, decide whether there is a subgraph $H \subseteq G$ with at most $k + |T|$ vertices such that H is a tree and T is a subset of the set of leaves of H. TST forms a variant of the well-known Steiner Tree problem. When parameterized by $k+|T|$, TST is fixed-parameter tractable (see long version [12]).

Proposition 10 (⋆). Terminal Steiner Tree$(k + |T|)$ *is diminishable.*

Proof (Diminisher Construction). We present a parameter-decreasing branching rule and a strict composition for Terminal Steiner Tree$(k + |T|)$. Together with Lemma 6, the claim then follows. Let $(G = (N \uplus T, E), k)$ be an instance of TST$(k+|T|)$ (we can assume that G has a connected component containing T). We make several assumptions first. We can assume that $|T| \geq 3$ (otherwise a shortest path is the optimal solution) and additionally that for all terminals $t \in T$ it holds that $N_G(t) \not\subseteq T$ (as otherwise the instance is a no-instance). Moreover, we can assume that there is no vertex $v \in N$ such that $T \subseteq N_G(v)$, as otherwise we immediately output whether $k \geq 1$.

For the parameter decreasing branching rule, select a terminal $t^* \in T$, and let $v_1, \ldots, v_d$ denote the neighbors of t^* in $G - (T \setminus \{t^*\})$. We create d instances $(G_1, k-1), \ldots, (G_d, k-1)$ as follows. Define G_i, $i \in [d]$, by $G_i := G - v_i$. Turn the vertices in $N_G(v_i)$ in G_i into a clique, that is, for each distinct vertices $v, w \in$

$N_G(v_i)$ add the edge $\{v, w\}$ if not yet present. This finishes the construction of G_i. It is not hard to see that the construction can be done in polynomial time.

Next, we describe the strict composition for TST$(k+|T|)$. Given the instances $(G_1, k), \ldots, (G_d, k)$, we create an instance (G', k) as follows. Let G' be initially the disjoint union of $G_1, \ldots, G_d$. For each $t \in T$, identify its copies in $G_1, \ldots, G_d$, say $t_1, \ldots, t_d$, with one vertex t' corresponding to t. This finishes the construction of G'. Note that for every $i, j \in [d]$, $i \neq j$, any path between a vertex in G_i and a vertex in G_j contains a terminal vertex. Hence, any terminal Steiner tree in G' contains non-terminal vertices only in G_i for exactly one $i \in [d]$. It is not difficult to see that (G', k) is a yes-instance if and only if one of the instances $(G_1, k), \ldots, (G_d, k)$ is a yes-instance. □

4 Problems Without Semi-strict Polynomial Kernels

As strict kernels only allow an increase of the parameter value by an additive constant (Definition 1), one may ask whether one can exclude less restrictive versions of strict kernels for parameterized problems using the concept of parameter diminishers. Targeting this question, in this section we study scenarios with a multiplicative (instead of additive) parameter increase by a constant. That is, property (iii) in Definition 1 is replaced by $k' \leq c \cdot k$, for some constant c. We refer to this as *semi-strict kernels*.

Note that Theorem 4 does not imply that the problems mentioned in Theorem 2 do not admit semi-strict polynomial kernelizations unless P = NP. Intuitively, the parameter diminisher is constantly often applied to decrease the parameter, while dealing only with a constant additive blow-up of the parameter caused by the strict kernelization. When dealing with a constant multiplicative blow-up of the parameter caused by the semi-strict kernelization, the parameter diminisher is required to be applied a non-constant number of times. Hence, to deal with semi-strict kernelization, we introduce a stronger version of our parameter diminisher: Formally, we replace property (ii) in Definition 3 by $k' \leq k/c$, for some constant $c > 1$. We refer to this as *strong parameter diminishers*.

Next, we show an analogue of Theorem 4 for semi-strict polynomial kernelizations and strong parameter diminishers.

Theorem 11 (⋆). *Let L be a parameterized problem such that its unparameterized version is* NP*-hard and $\{(x, k) \in L \mid k \leq c\} \in$ P*, for some constant $c \geq 1$. If L is strongly diminishable and admits a semi-strict polynomial kernel, then* P = NP*.*

By Theorem 11, if we can prove a strong diminisher for a parameterized problem, then it does not admit a semi-strict polynomial kernel, unless P = NP. We give a strong diminisher for the Set Cover problem: Given a set U called the universe, a family $\mathcal{F} \subseteq 2^U$ of subsets of U, and an integer k, the question is whether there are k sets in the family $\mathcal{F}$ that cover the whole universe. We show that Set Cover parameterized by $k \log n$, where $n = |U|$, is strongly diminishable.

Theorem 12 (⋆). *Unless* P = NP, Set Cover($k \log n$) *and* Hitting Set($k \log m$) *do not admit a semi-strict polynomial kernel.*

Proof (Strong Diminisher Construction). Let $(U, \mathcal{F} = \{F_1, \ldots, F_m\}, k)$ be an instance of Set Cover($k \log n$) and assume that $k \geq 2$ and $n \geq 5$. If k is odd, then we add a unique element to U, a unique set containing only this element to $\mathcal{F}$, and we set $k = k + 1$. Hence, we assume that k is even. The following procedure is a strong parameter diminisher for the problem parameterized by $k \log n$. Let $U' = U$ and for all F_i, F_j create $F'_{\{i,j\}} = F_i \cup F_j$. Let $\mathcal{F}' = \{F'_{\{i,j\}} \mid i \neq j\}$ and set $k' = k/2$. This yields the instance $(U', \mathcal{F}', k')$ of Set Cover($k \log n$) in polynomial time. The proof of correctness is deferred to a long version [12]. □

Seeking for parameter diminishers to exclude strict polynomial kernelizations raises the question whether there are parameterized problems that are not (strongly) diminishable. In the following, we prove that under the *Exponential Time Hypothesis*, or ETH for short [16], there are natural problems that do not admit strong parameter diminishers. Here we restrict ourselves to problems where we have a parameter diminisher. The Exponential Time Hypothesis states that there is no algorithm for 3-CNF-Sat running in $2^{o(n)}\,\mathrm{poly}(n+m)$ time, where n and m denote the number of variables and clauses, respectively.

Theorem 13 (⋆). *Assuming* ETH*, none of the following is strongly diminishable:* CNF-Sat(n)*,* Rooted Path(k)*,* Clique(Δ)*,* Clique(tw)*, and* Clique(bw)*.*

The following lemma is the key tool for excluding strong parameter diminishers under ETH. Roughly, it can be understood as saying that a strong parameter diminisher can improve the running time of existing algorithms.

Lemma 14 (⋆). *Let L be a parameterized problem. If there is an algorithm $\mathcal{A}$ that solves any instance $(x, k) \in L$ in $2^{O(k)} \cdot |x|^{O(1)}$ time and L is strongly diminishable, then there is an algorithm $\mathcal{B}$ that solves L in $2^{O(k/f(x,k))} \cdot |x|^{f(x,k)^{O(1)}}$ time, where $f : L \to \mathbb{N}$ is a function mapping instances of L to the natural numbers with the following property: For every constant c there is a natural number n such that for all instances $(x, k) \in L$ we have that $|x| \geq n$ implies that $f(x, k) \geq c$.*

Intuitively, we apply Lemma 14 to exclude the existence of strong parameter diminishers under ETH as follows. Consider a problem where we know a running time lower bound based on the ETH and we also know an algorithm that matches this lower bound. Then, due to Lemma 14, for many problems a strong parameter diminisher and a suitable choice for the function f would imply the existence of an algorithm whose running time breaks the lower bound.

5 Conclusion

We showed that for several natural problems a strict polynomial-size problem kernel is as likely as P = NP. Since basically all observed (natural and practically

relevant) polynomial kernels are strict, this reveals that the existence of valuable kernels may be tighter connected to the P vs. NP problem than previously expected (in almost all previous work a connection is drawn to a collapse of the polynomial hierarchy to its third level, and the conceptual framework used there seems more technical than the one used here). Our work is based on results of Chen *et al.* [8] and shows that their basic ideas can be extended to a larger class of problems than dealt with in their work.

The diminisher framework leaves several challenges for future work. Are there natural problems where the presented framework is able to refute strict polynomial kernels while the composition framework [5] is not? It is not clear whether a framework based on a weaker assumption is even able to produce results that a framework based on a stronger assumption is not able to produce. This possibly also ties in with the question whether there are "natural" parameterized problems that admit a polynomial kernel but no strict polynomial kernel.[3] We close with two concrete open problems:

- We proved that Multicolored Path(k) is diminishable (and thus refutes a strict polynomial kernel unless P = NP). Can this result be extended to the uncolored version of the problem? This is also open for the directed case.
- Clique(Δ), Clique(tw), Clique(bw) do not have strong diminishers under the ETH (Sect. 4). Is this also true for Clique(cw)?

References

1. Abu-Khzam, F.N., Fernau, H.: Kernels: annotated, proper and induced. In: Bodlaender, H.L., Langston, M.A. (eds.) IWPEC 2006. LNCS, vol. 4169, pp. 264–275. Springer, Heidelberg (2006). https://doi.org/10.1007/11847250_24
2. Alon, N., Yuster, R., Zwick, U.: Color-coding. J. ACM **42**(4), 844–856 (1995)
3. Betzler, N., Guo, J., Komusiewicz, C., Niedermeier, R.: Average parameterization and partial kernelization for computing medians. J. Comput. Syst. Sci. **77**(4), 774–789 (2011)
4. Binkele-Raible, D., Fernau, H., Fomin, F.V., Lokshtanov, D., Saurabh, S., Villanger, Y.: Kernel(s) for problems with no kernel: on out-trees with many leaves. ACM Trans. Algorithms **8**(4), 38 (2012)
5. Bodlaender, H.L., Downey, R.G., Fellows, M.R., Hermelin, D.: On problems without polynomial kernels. J. Comput. Syst. Sci. **75**(8), 423–434 (2009)
6. Cai, L., Chen, J., Downey, R.G., Fellows, M.R.: Advice classes of parameterized tractability. Ann. Pure Appl. Logic **84**(1), 119–138 (1997)
7. Chen, J., Fernau, H., Kanj, I.A., Xia, G.: Parametric duality and kernelization: lower bounds and upper bounds on kernel size. SIAM J. Comput. **37**(4), 1077–1106 (2007)
8. Chen, Y., Flum, J., Müller, M.: Lower bounds for kernelizations and other preprocessing procedures. Theory Comput. Syst. **48**(4), 803–839 (2011)
9. Diestel, R.: Graph Theory, 4th edn. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-662-53622-3

[3] Note that Chen *et al.* [8, Proposition 3.3] presented an artificial parameterized problem admitting a polynomial kernel but no strict polynomial kernel.

10. Downey, R.G., Fellows, M.R.: Fundamentals of Parameterized Complexity. Springer, Heidelberg (2013). https://doi.org/10.1007/978-1-4471-5559-1
11. Fellows, M.R., Kulik, A., Rosamond, F.A., Shachnai, H.: Parameterized approximation via fidelity preserving transformations. J. Comput. Syst. Sci. **93**, 30–40 (2018)
12. Fernau, H., Fluschnik, T., Hermelin, D., Krebs, A., Molter, H., Niedermeier, R.: Diminishable parameterized problems and strict polynomial kernelization. CoRR abs/1611.03739 (2018). http://arxiv.org/abs/1611.03739
13. Fluschnik, T., Mertzios, G.B., Nichterlein, A.: Kernelization lower bounds for finding constant size subgraphs. CoRR abs/1710.07601 (2017). http://arxiv.org/abs/1710.07601
14. Fortnow, L., Santhanam, R.: Infeasibility of instance compression and succinct PCPs for NP. J. Comput. Syst. Sci. **77**(1), 91–106 (2011)
15. Guo, J., Niedermeier, R.: Invitation to data reduction and problem kernelization. ACM SIGACT News **38**(1), 31–45 (2007)
16. Impagliazzo, R., Paturi, R.: On the complexity of k-SAT. J. Comput. Syst. Sci. **62**(2), 367–375 (2001)
17. Karp, R.M., Lipton, R.: Turing machines that take advice. L'Enseignement Mathématique **28**(2), 191–209 (1982)
18. Kratsch, S.: Recent developments in kernelization: a survey. In: Bulletin of the EATCS, no. 113 (2014)
19. Lin, G., Xue, G.: On the terminal Steiner tree problem. Inf. Process. Lett. **84**(2), 103–107 (2002)
20. Lokshtanov, D., Panolan, F., Ramanujan, M.S., Saurabh, S.: Lossy kernelization. In: Proceedings of 49th STOC, pp. 224–237. ACM (2017)
21. Schäfer, A., Komusiewicz, C., Moser, H., Niedermeier, R.: Parameterized computational complexity of finding small-diameter subgraphs. Optim. Lett. **6**(5), 883–891 (2012)