Theory Comput Syst (2011) 48: 803–839
DOI 10.1007/s00224-010-9270-y

# Lower Bounds for Kernelizations and Other Preprocessing Procedures

**Yijia Chen · Jörg Flum · Moritz Müller**

Published online: 26 May 2010


**Abstract** We first present a method to rule out the existence of parameter non-increasing polynomial kernelizations of parameterized problems under the hypothesis $\mathrm{P} \neq \mathrm{NP}$. This method is applicable, for example, to the problem SAT parameterized by the number of variables of the input formula. Then we obtain further improvements of corresponding results in (Bodlaender et al. in Lecture Notes in Computer Science, vol. 5125, pp. 563–574, Springer, Berlin, 2008; Fortnow and Santhanam in Proceedings of the 40th ACM Symposium on the Theory of Computing (STOC'08), ACM, New York, pp. 133–142, 2008) by refining the central lemma of their proof method, a lemma due to Fortnow and Santhanam. In particular, assuming that the polynomial hierarchy does not collapse to its third level, we show that every parameterized problem with a "linear OR" and with NP-hard underlying classical problem does not have polynomial self-reductions that assign to every instance $x$ with parameter $k$ an instance $y$ with $|y| = k^{O(1)} \cdot |x|^{1-\varepsilon}$ (here $\varepsilon$ is any given real number greater than zero). We give various applications of these results. On the structural side we prove several results clarifying the relationship between the different notions of pre-

M. Müller wishes to thank the John Templeton Foundation for its support under Grant #13152, *The Myriad Aspects of Infinity*.

Y. Chen (✉)
Department of Computer Science and Engineering, Shanghai Jiaotong University, Dongchuan Road, No. 800, 200240 Shanghai, China
e-mail: yijia.chen@cs.sjtu.edu.cn

J. Flum
Abteilung für mathematisches Logik, Albert-Ludwigs-Universität Freiburg, Eckerstr. 1, 79104 Freiburg, Germany
e-mail: joerg.flum@math.uni-freiburg.de

M. Müller
Centre de Recerca Matemàtica Barcelona, Science Building, Campus de Bellaterra, 08193 Bellaterra, Spain
e-mail: mmoeller@crm.cat

processing procedures, namely the various notions of kernelizations, self-reductions and compressions.

**Keywords** Kernelization · Parameterized complexity · Lower bound

## 1 Introduction

Often, if a computationally hard problem must be solved in practice, one tries, in a preprocessing step, to reduce the size of the input data. This approach has been widely studied and applied in parameterized complexity and it is known as *kernelization* there. We recall the basic concepts.

Parameterized complexity is a refinement of classical complexity theory, in which one measures the complexity of an algorithm not only in terms of the total input length, but also takes into account other aspects of the input codified in a natural number, its *parameter*. Hence, a parameterized problem $(Q, \kappa)$ consists of a classical problem $Q$ together with a function $\kappa$ which assigns to every instance $x$ of $Q$ its parameter $\kappa(x) \in \mathbb{N}$.

Central to parameterized complexity theory is the notion of fixed-parameter tractability. It relaxes the classical notion of tractability by allowing algorithms whose running time can be exponential but only in terms of the parameter. This is based on the idea to choose the parameter in such a way that it can be assumed to be small for the instances one is interested in. To be precise, a problem $(Q, \kappa)$ is said to be *fixed-parameter tractable* if $x \in Q$ can be decided by an *fpt-algorithm*, that is, an algorithm whose running time is $f(k) \cdot p(n)$, where $n$ denotes the length $|x|$ of $x$, $k := \kappa(x)$, and where $f$ is an arbitrary computable function and $p$ a polynomial.

A *kernelization* $\mathbb{K}$ of a parameterized problem is a polynomial time algorithm that computes for every instance $x$ of the problem an equivalent instance $\mathbb{K}(x)$ of a size bounded in terms of the parameter $\kappa(x)$. This suggests a method for designing fpt-algorithms: To decide a given instance $x$, we compute the *kernel* $\mathbb{K}(x)$ and then decide if $\mathbb{K}(x)$ is a yes-instance by brute-force. The converse holds, too: Every fixed-parameter tractable problem has a kernelization. The proof of this fact is easy; however it gives only a "trivial" kernel with no algorithmic impact.

Besides efficient computability, an important quality of a good kernelization is *small kernel size*. The notion of *polynomial kernelization* is an abstract model for small kernel size. A kernelization $\mathbb{K}$ is polynomial if there is a polynomial $p$ such that for all instances $x$ the size of $\mathbb{K}(x)$ is bounded by $p(\kappa(x))$.

Polynomial kernelizations are known for many parameterized problems (compare the survey [16]). However, till recently, besides artificial problems, only few natural problems were known to have *no* polynomial kernelizations (one being the model-checking for monadic second-order logic on trees parameterized by the length of the second-order formula). This has changed, since a general method to exclude polynomial kernelizations has been developed (cf. [1, 7]). It is based on a lemma due to Fortnow and Santhanam [7]: Recall that an OR for a classical problem $Q$ is a polynomially time computable function that assigns to every finitely many instances $x_1, \ldots, x_t$ of $Q$ an instance $y$ such that ($y \in Q$ if and only if $x_i \in Q$ for some $i \in \{1, \ldots, t\}$).

In [7] it is shown that no NP-complete problem can have an OR with the additional property that the length $|y|$ of $y$ is polynomially bounded in $\max_{1\leq i\leq t}|x_i|$ unless the polynomial hierarchy collapses to its third level.

However there are natural parameterized problems $(Q,\kappa)$ with NP-complete problem $Q$ having an OR such that the *parameter* $\kappa(y)$ of $y$ is polynomially bounded in $\max_{1\leq i\leq t}|x_i|$. If such a problem would have a polynomial kernelization, then composing it with such an OR would yield an OR with the additional property excluded by the lemma of Fortnow and Santhanam. Various applications of this result were given in [1, 7], in particular, in [7] it was shown that the problem SAT parameterized by the number of propositional variables of the input formula has no polynomial kernelizations (unless the polynomial hierarchy collapses to its third level). This settled a question repeatedly posed in [6, p. 231], [12], and implicitly already in [9].

As already mentioned, concrete kernelizations yield algorithms for solving parameterized problems efficiently for small parameter values. Conceptually similar are *compression* algorithms, even though the intention is slightly different. There the question is whether one can efficiently compress every "long" instance $x$ of a problem $Q$ with "a short witness" to a shorter equivalent instance $x'$ of a problem $Q'$ (here equivalent means that $x\in Q$ if and only if $x'\in Q'$). "Such compression enables to succinctly store instances until a future setting will allow solving them, either via a technological or algorithmic breakthrough or simply until enough time has elapsed" (see [12]). Using this terminology Harnik and Naor [12] addressed questions similar to that of the existence of an OR with the additional property mentioned above. By suitably generalizing the notion of a kernelization of a parameterized problem to the notion of a kernelization from some parameterized problem to another one, Fortnow and Santhanam [7] introduce a framework which allows to deal with kernelizations and compressions at the same time (in [7] a different terminology is used). Nevertheless we stick to the traditional notion of kernelization as we mainly address problems of parameterized complexity.

We explain the contents of our paper. To the best of our knowledge all reasonable kernelizations $\mathbb{K}$ for concrete parameterized problems $(Q,\kappa)$ are *parameter non-increasing*, that is, the parameter of the kernel of an instance $x$ is less than or equal to the parameter of $x$, more succinctly, $\kappa(\mathbb{K}(x))\leq\kappa(x)$. Moreover it is known that every parameterized problem that has a kernelization already has a parameter non-increasing kernelization. In Sect. 4 we present a result (Theorem 4.2) with a quite simple proof showing that every parameterized problem with "parameter decreasing" self-reductions has no parameter non-increasing *polynomial* kernelizations. This result only requires that $\mathrm{P}\neq\mathrm{NP}$ (instead of the assumption that the polynomial hierarchy does not collapse to its third level). As an application we get that the problem SAT has no parameter non-increasing polynomial kernelization if $\mathrm{P}\neq\mathrm{NP}$ and no parameter non-increasing subexponential kernelization if the exponential time hypothesis (ETH) holds.

However, polynomial kernelizations, which are not parameter non-increasing are not only interesting from a theoretical point of view but also for practical purposes: such a polynomial kernelization for SAT would be sufficient for some significant application in cryptography [12]. It is perfectly conceivable that a parameterized problem has a useful preprocessing procedure that decreases the size of the input considerably at the cost of a slight increase of the (small) parameter. Such a slight increase

may even be necessary: In Sect. 3.1 we prove that there exist parameterized problems that have polynomial kernelizations but all of them 'slightly' increase the parameter.

In Sect. 5 we recall results of Bodlaender et al. [1] and of Fortnow and Santhanam [7] relevant in our context and we give some new applications, in particular to variants of the PATH problem. Then we refine the central lemma due to Fortnow and Santhanam to obtain better lower bounds. Applied to the SAT problem we show in Sect. 6:

> If the polynomial hierarchy does not collapse to its third level, then for every $\varepsilon > 0$ there is no polynomial time algorithm that for every instance $\alpha$ of SAT with $k$ variables computes an equivalent instance $\alpha'$ with
>
> $$|\alpha'| \leq k^{O(1)} \cdot |\alpha|^{1-\varepsilon}. \tag{1}$$

This result is a particular instance of a general theorem that yields lower bounds of the type in (1) for every parameterized problem "having a linear OR" (compare Theorem 6.5 for the precise statement). Note that nothing is said about the number of variables of the formula $\alpha'$. Thus, even though (in the main text) we state our results for parameterized problems, it addresses arbitrary problems, where the inputs have a natural (not necessarily small) parameter.

In Sect. 7 for problems satisfying an apparently weaker condition, namely only "having an OR for instances with constant parameter" we still get quite good lower bounds; in case of SAT it would be:

$$|\alpha'| \leq k^{O(1)} \cdot |\alpha|^{o(1)}. \tag{2}$$

In the last section we compare the different notions of OR considered in this paper and we also compare the notions of polynomial kernelizations and those of polynomial reductions leading to the lower bounds in (1) and (2).

Finally we should mention that after recalling some definitions and fixing our notation in Sect. 2, we consider and analyze some basic questions concerning kernelizations in Sect. 3. In particular, we shall see that "most" parameterized problems (more precisely, all problems in EXPT) have polynomial kernelizations if and only if they are self-compressible.

## 2 Preliminaries

The set of natural numbers (that is, nonnegative integers) is denoted by $\mathbb{N}$. For a natural number $n$ let $[n] := \{1, \ldots, n\}$. By $\log n$ we mean $\lceil \log n \rceil$ if an integer is expected. For $n = 0$ the term $\log n$ is undefined. We trust the reader's common sense to interpret such terms reasonably.

We identify problems (or languages) with subsets $Q$ of $\{0,1\}^*$. Clearly, as done mostly, we present concrete problems in a verbal, hence uncodified form or as a set of strings over an arbitrary finite alphabet. We use both P and PTIME to denote the class of problems $Q$ such that $x \in Q$ is solvable in polynomial time.

A *reduction from a problem $Q$ to a problem $Q'$* is a mapping $R : \{0,1\}^* \to \{0,1\}^*$ such that for all $x \in \{0,1\}^*$ we have $(x \in Q \Leftrightarrow R(x) \in Q')$. We write $R : Q \leq^p Q'$ if $R$ is a reduction from $Q$ to $Q'$ computable in polynomial time, and $Q \leq^p Q'$ if there is a polynomial time reduction from $Q$ to $Q'$.

### 2.1 Parameterized Complexity

A *parameterized problem* is a pair $(Q, \kappa)$ consisting of a classical problem $Q \subseteq \{0,1\}^*$ and a *parameterization* $\kappa : \{0,1\}^* \to \mathbb{N}$, which is required to be polynomial time computable even if the result is encoded in unary.

We introduce some parameterized problems, which will be used later, thereby exemplifying our way to represent parameterized problems. We denote by $p$-SAT the parameterized problem

| $p$-SAT | |
|---|---|
| *Instance:* | A propositional formula $\alpha$ in conjunctive normal form. |
| *Parameter:* | Number of variables of $\alpha$. |
| *Question:* | Is $\alpha$ satisfiable? |

By $p$-PATH and $p$-CLIQUE we denote the problems:

| $p$-PATH | |
|---|---|
| *Instance:* | A graph $G$ and $k \in \mathbb{N}$. |
| *Parameter:* | $k$. |
| *Question:* | Does $G$ have a path of length $k$? |

| $p$-CLIQUE | |
|---|---|
| *Instance:* | A graph $G$ and $k \in \mathbb{N}$. |
| *Parameter:* | $k$. |
| *Question:* | Does $G$ have a clique of cardinality $k$? |

Similarly we define $p$-DOMINATING-SET. If C is a class of graphs, then $p$-PATH(C) denotes the problem

| $p$-PATH(C) | |
|---|---|
| *Instance:* | A graph $G$ in C and $k \in \mathbb{N}$. |
| *Parameter:* | $k$. |
| *Question:* | Does $G$ have a path of length $k$? |

We use similar notations for other problems.

We recall the definitions of the classes FPT, EXPT, EPT and SUBEPT. A parameterized problem $(Q, \kappa)$ is *fixed-parameter tractable* (or, in FPT) if $x \in Q$ is solvable in time $f(\kappa(x)) \cdot |x|^{O(1)}$ for some computable $f : \mathbb{N} \to \mathbb{N}$. If $f$ can be chosen such that $f(k) = 2^{k^{O(1)}}$, then $(Q, \kappa)$ is in EXPT. If $f$ can be chosen such that $f(k) = 2^{O(k)}$, then $(Q, \kappa)$ is in EPT. If $f$ can be chosen such that $f(k) = 2^{o^{\text{eff}}(k)}$, then $(Q, \kappa)$ is in SUBEPT.

Here $o^{\text{eff}}$ denotes the effective version of little oh: For computable functions $f, g : \mathbb{N} \to \mathbb{N}$ we say that $f$ *is effectively little oh of* $g$ and write $f = o^{\text{eff}}(g)$ if there is a *computable*, nondecreasing, and unbounded function $\iota : \mathbb{N} \to \mathbb{N}$ such that for sufficiently large $k \in \mathbb{N}$

$$f(k) \leq \frac{g(k)}{\iota(k)}.$$

As usual we often write $f(k) = o^{\text{eff}}(g(k))$ instead of $f = o^{\text{eff}}(g)$.

At some places in this paper, it will be convenient to consider *preparameterized problems*; these are pairs $(Q, \kappa)$, where again $Q$ is a classical problem and $\kappa$ is a *preparameterization*, that is, an arbitrary function from $\{0,1\}^*$ to the set $\mathbb{R}_{\geq 0}$ of non-negative real numbers.

## 3 Fundamentals of Kernelization

In this section we start by recalling the notion of kernelization and by introducing some refinements. Then we compare the different notions of kernelizations (in Sect. 3.1), study the complexity of problems with such kernelizations (in Sect. 3.2) and finally, we analyze the relationship between polynomial kernelizations and compressions (in Sect. 3.3).

**Definition 3.1** Let $(Q, \kappa)$ be a parameterized problem and $f : \mathbb{N} \to \mathbb{N}$ be a function. An *$f$-kernelization* for $(Q, \kappa)$ is a polynomial time algorithm $\mathbb{K}$ that on input $x \in \{0,1\}^*$ outputs $\mathbb{K}(x) \in \{0,1\}^*$ such that

$$(x \in Q \Leftrightarrow \mathbb{K}(x) \in Q) \quad \text{and} \quad |\mathbb{K}(x)| \leq f(\kappa(x)).$$

In particular, $\mathbb{K}$ is a polynomial time reduction from $Q$ to itself. If in addition for all $x \in \{0,1\}^*$

$$\kappa(\mathbb{K}(x)) \leq \kappa(x),$$

then $\mathbb{K}$ is a *parameter non-increasing* $f$-kernelization. A (*parameter non-increasing*) *kernelization* is a (parameter non-increasing) $f$-kernelization for some computable function $f : \mathbb{N} \to \mathbb{N}$.

We say that $(Q, \kappa)$ has a *linear, polynomial*, *subexponential*, *simply exponential*, and *exponential kernelization* if there is an $f$-kernelization for $(Q, \kappa)$ with $f(k) = O(k)$, $f(k) = k^{O(1)}$, $f(k) = 2^{o^{\text{eff}}(k)}$, $f(k) = 2^{O(k)}$, and $f(k) = 2^{k^{O(1)}}$, respectively.

The following result is well-known:

**Proposition 3.2** *Let $(Q, \kappa)$ be a parameterized problem with decidable $Q$. The following statements are equivalent.*

(1) *$(Q, \kappa)$ is fixed-parameter tractable.*
(2) *$(Q, \kappa)$ has a kernelization.*
(3) *$(Q, \kappa)$ has a parameter non-increasing kernelization.*

*Furthermore, if $f$ is computable and $x \in Q$ is solvable in time $f(\kappa(x)) \cdot |x|^{O(1)}$, then $(Q, \kappa)$ has a parameter non-increasing $f$-kernelization.*

### 3.1 Comparing the Different Notions of Kernelizations

We are mainly interested in polynomial kernelizations. First we show that the notions of polynomial kernelization and of parameter non-increasing polynomial kernelization are distinct:

**Proposition 3.3** *There is a parameterized problem $(Q, \kappa)$ that has a polynomial kernelization but no parameter non-increasing polynomial kernelization.*

*Proof* Let $Q$ be a classical problem that is not solvable in time $2^{O(|x|)}$. We define a parameterized problem $(P, \kappa)$ with $P \subseteq \{0,1\}^* \times \{1\}^*$ and with $\kappa((x, 1^k)) = k$. By $1^k$ we denote the string consisting of $k$ many 1s. For each $k \in \mathbb{N}$ we define the *$k$-projection* $P[k] := \{x \mid (x, 1^k) \in P\}$ *of* $P$ by:

- If $k = 2\ell + 1$, then
$$P[k] := Q_{=\ell} \left( := \left\{x \in Q \;\middle|\; |x| = \ell\right\}\right).$$
Hence, all elements in $P[k]$ have length $\ell$.
- If $k = 2\ell$, then
$$P[k] := \left\{x1^{2^\ell} \;\middle|\; x \in Q_{=\ell}\right\},$$
where $x1^{2^\ell}$ is the concatenation of $x$ with the string $1^{2^\ell}$. Hence, all elements in $P[k]$ have length $\ell + 2^\ell$.

Intuitively, an element in the $2\ell$-projection is an element in the $(2\ell + 1)$-projection padded with $2^\ell$ many 1s. It is not hard to see that $P$ has a linear kernelization (which "on the even projections" increases the parameter).

We claim that $P$ has no parameter non-increasing polynomial kernelization. Assume $\mathbb{K}$ is such a kernelization and $c, d \in \mathbb{N}$ such that

$$\left|\mathbb{K}((z, 1^m))\right| \leq d \cdot m^c.$$

We use $\mathbb{K}$ to solve $x \in Q$ in time $2^{O(|x|)}$: Let $x$ be an instance of $Q$ and let $\ell := |x|$. We may assume that

$$d \cdot (2\ell)^c < 2^\ell$$

(note that there are only finitely many $x$ not satisfying this inequality). We compute (in time $2^{O(\ell)}$)

$$(u, k) := \mathbb{K}\big((x1^{2^\ell}, 2\ell)\big).$$

We know that $k \leq 2\ell$ and $|u| \leq d \cdot (2\ell)^c < 2^\ell$. If $u$ does not have the length of the strings in $P[k]$, then $(u, k) \notin P$ and therefore $x \notin Q$. In particular, this is the case if $k = 2\ell$ (as $|u| < 2^\ell$). If $u$ has the length of the strings in $P[k]$ and hence $k < 2\ell$, then it is easy to read off from $u$ an instance $y$ with $|y| < |x|$ and $(y \in Q \iff x \in Q)$. We then apply the same procedure to $y$. □

The survey [11] contains examples of natural problems whose currently best known kernelizations are polynomial, simply exponential and exponential. We show that all these different degrees of kernelizability are indeed different:

**Proposition 3.4** *The classes of parameterized problems with a linear, a polynomial, a subexponential, a simply exponential, and an exponential kernelization are pairwise different.*

The claim immediately follows from the following lemma.

**Lemma 3.5** *Let $g : \mathbb{N} \to \mathbb{N}$ be nondecreasing and unbounded and let $f : \mathbb{N} \to \mathbb{N}$ be such that $f(k) \leq g(k-1)$ for all sufficiently large $k$. Then there is a $Q \subseteq \{0,1\}^*$ and a preparameterization $\kappa$ such that $(Q, \kappa)$ has a $g$-kernelization but no $f$-kernelization.*

*If in addition $g$ is increasing and time-constructible, then we can choose $\kappa$ to be a parameterization.*

*Proof* Let $g$ and $f$ be as in the statement. We choose $k_0$ such that $f(k) \leq g(k-1)$ for all $k \geq k_0$. We consider the "inverse function" $\iota_g$ of $g$ given by

$$\iota_g(m) := \min\{s \in \mathbb{N} \mid g(s) \geq m\}.$$

Then for all $n \in \mathbb{N}$

$$n \leq g(\iota_g(n)) \quad \text{and} \quad \text{if } \iota_g(n) \geq 1, \text{ then } g(\iota_g(n)-1) < n. \tag{3}$$

Let $Q$ be a problem not in PTIME and define the preparameterization $\kappa$ by $\kappa(x) := \iota_g(|x|)$. By the first inequality in (3) the identity is a $g$-kernelization of $(Q, \kappa)$, even a parameter non-increasing one.

Assume that there is an $f$-kernelization $\mathbb{K}$ of $(Q, \kappa)$. As $\iota_g$ is unbounded, we have $\iota_g(|x|) \geq k_0$ for sufficiently long $x \in \{0,1\}^*$. Then

$$|\mathbb{K}(x)| \leq f(\kappa(x)) = f(\iota_g(|x|)) \leq g(\iota_g(|x|)-1) < |x|.$$

Thus applying $\mathbb{K}$ at most $|x|$ times we get an equivalent instance of length at most $\max_{0 \leq i < k_0} f(i)$. Therefore, $Q \in$ PTIME, a contradiction.

If $g$ is increasing and time-constructible, then $\iota_g$ is polynomial time computable and hence $\kappa$ is a parameterization. □

### 3.2 Complexity of Problems with Kernelizations

We know that a parameterized problem is fixed-parameter tractable if and only if it has a kernelization (see Proposition 3.2). The next result shows that a parameterized problem $(Q, \kappa)$ in FPT \ EXPT with $Q \in$ NP cannot have polynomial kernelizations. We show a little bit more. Recall that EXP is the class of classical problems $Q$ such that $x \in Q$ is solvable in deterministic time $2^{|x|^{O(1)}}$.

**Proposition 3.6** *Let $f, t : \mathbb{N} \to \mathbb{N}$ be computable, non-decreasing and let $(Q, \kappa)$ be a parameterized problem. If $Q$ is decidable in time $t$ and has an $f$-kernelization, then $(Q, \kappa)$ can be solved in time $t(f(\kappa(x))) \cdot |x|^{O(1)}$.*

*In particular, if $(Q, \kappa)$ has a polynomial kernelization and $Q \in$ EXP, then $(Q, \kappa) \in$ EXPT.*

*Proof* Let $\mathbb{K}$ be a $f$-kernelization of $(Q, \kappa)$ and let $\mathbb{A}$ be an algorithm solving $x \in Q$ in time $t(|x|)$. The algorithm that on $x \in \{0,1\}^*$ first computes $\mathbb{K}(x)$ and then applies $\mathbb{A}$ to $\mathbb{K}(x)$ solves $x \in Q$ in time $|x|^{O(1)} + t(|\mathbb{K}(x)|) = |x|^{O(1)} + t(f(\kappa(x)))$. □

The model-checking of monadic second-order logic on the class of trees is in EXP. The corresponding parameterized problem with the length of the formula as parameter is in FPT but, by a result of [8], not in EXPT unless P = NP. Hence, by the preceding proposition, it has no polynomial kernelization (unless P = NP).

In later sections, under some complexity-theoretic assumptions, we will present various examples of natural problems that are in EPT and have no polynomial kernelizations. Here we give a simple, artificial example that is provably without polynomial kernelizations.

*Example 3.7* Let $Q$ be a classical problem not in PTIME but solvable in time $O(|x|^{\log|x|})$. Let $\kappa$ be the parameterization mapping $x$ to $(\log|x|)^2$. Then $(Q,\kappa) \in$ EPT, because $2^{\kappa(x)} = |x|^{\log|x|}$.

For the sake of contradiction assume that $(Q,\kappa)$ has a polynomial kernelization $\mathbb{K}$. Then to decide if $x \in Q$ it suffices to decide if $\mathbb{K}(x) \in Q$. Since $|\mathbb{K}(x)| = (\log|x|)^{O(1)}$ this can be done in time

$$|\mathbb{K}(x)|^{\log|\mathbb{K}(x)|} \leq (\log|x|)^{O(\log\log|x|)} \leq 2^{(\log\log|x|)^{O(1)}} \leq |x|^{O(1)}.$$

Thus $Q \in$ PTIME, a contradiction.

However, if we would allow kernelizations to have slightly superpolynomial running time, then *every* EPT problem would have subexponential kernelizations:

**Proposition 3.8** *Let $(Q,\kappa) \in$ EPT and $\iota : \mathbb{N} \to \mathbb{N}$ be a nondecreasing unbounded and computable function.*[1] *Then there is an algorithm $\mathbb{K}$ that for every instance $x$ of $Q$ outputs an instance $\mathbb{K}(x)$ in time*

$$|x|^{O(\iota(\kappa(x)))}$$

*such that*

$$(x \in Q \iff \mathbb{K}(x) \in Q) \quad \text{and} \quad |\mathbb{K}(x)| = 2^{o(\kappa(x))}.$$

To obtain this proposition we just refine the "standard" proof of the implication (1) $\Rightarrow$ (2) of Proposition 3.2 and show that every problem in EPT has arbitrarily small exponential kernelizations, that is, for every $\varepsilon \in \mathbb{R}$ with $\varepsilon > 0$ there is a polynomial kernelization with kernels of size $\leq O(1) + (1+\varepsilon)^{\kappa(x)}$, even more:

**Lemma 3.9** *Let $(Q,\kappa)$ be a parameterized problem in* EPT. *There is an algorithm $\mathbb{I}$ that takes as inputs an instance $x$ of $Q$ and $\ell \in \mathbb{N}$ and outputs an instance $\mathbb{I}(x,\ell)$ of $Q$ in time $|x|^{O(\ell)}$ such that*

$$(x \in Q \iff \mathbb{I}(x,\ell) \in Q) \quad \text{and} \quad |\mathbb{I}(x,\ell)| = 2^{O(\kappa(x))/\ell}.$$

[1] To get a "slightly superpolynomial running time" we choose as $\iota$ an "extremely slowly" growing function.

*Proof* We choose $c \in \mathbb{N}$ and an algorithm $\mathbb{A}$ solving $x \in Q$ in time $2^{c\cdot\kappa(x)} \cdot |x|^{O(1)}$. Furthermore we fix $x_+ \in Q$ and $x_- \notin Q$. (If $Q$ is trivial, that is, $Q = \emptyset$ or $Q = \{0,1\}^*$, we let $\mathbb{I}(x,\ell)$ always be the empty string.) Then the following is the desired algorithm.

$\mathbb{I}(x,\ell)$ // $x$ an instance of $Q$ and $\ell \in \mathbb{N}$.

(1) **if** $|x| \le 2^{\kappa(x)/\ell}$ **then** output $x$.
(2) **else** simulate $\mathbb{A}$ on $x$
// the running time is bounded by $2^{c\cdot\kappa(x)} \cdot |x|^{O(1)} \le |x|^{c\cdot\ell+O(1)}$.
(3) **if** $\mathbb{A}$ accepts $x$ **then** output $x_+$ **else** output $x_-$.

□

*Proof of Proposition 3.8* We choose a polynomial time computable $\nu : \mathbb{N} \to \mathbb{N}$ with $\nu \le \iota$ and set $\mathbb{K}(x) := \mathbb{I}(x, \nu(\kappa(x)))$, where $\mathbb{I}$ is the algorithm of the preceding lemma. □

### 3.3 Polynomial Kernelization and Compression

Most natural problems $Q \in \mathrm{NP}$ have a *canonical* representation of the form

$$x \in Q \iff \text{there is } y \in \{0,1\}^{g(x)} \text{ such that } (x,y) \in Q_0 \tag{4}$$

for some polynomial time computable function $g : \{0,1\}^* \to \mathbb{N}$ (with the output represented in unary) and some $Q_0 \in \mathrm{PTIME}$. In [4] the problem $(Q,g)$ has been called the *canonical parameterization* of $Q$ (more precisely, one should speak of the canonical parameterization induced by the representation (4) of $Q$). Clearly $(Q,g)$ is fixed-parameter tractable, it is even in EPT. If $(Q,\kappa)$ was a parameterized problem, then $(Q,g)$ is called the *canonical reparameterization* of $(Q,\kappa)$.

The canonical reparameterization of $p$-SAT is $p$-SAT itself; the canonical reparameterizations of the problems $p$-PATH, $p$-CLIQUE and $p$-DOMINATING-SET are the problems *uni*-PATH, *uni*-CLIQUE and *uni*-DOMINATING-SET,[2] respectively, where for all three cases, we have $g((G,k)) = k \cdot \log|V|$; hence in particular,

*uni*-PATH
*Instance:* A graph $G = (V,E)$ and $k \in \mathbb{N}$.
*Parameter:* $k \cdot \log|V|$.
*Question:* Does $G$ have a path of length $k$?

Many fixed-parameter tractable problems, namely all in EXPT and hence, in particular, $p$-PATH, have polynomial kernelizations if and only if their canonical reparameterizations have. This is shown by the following proposition.

**Proposition 3.10** *Let $(Q,\kappa) \in \mathrm{EXPT}$ and let $(Q,g)$ be the canonical reparameterization of $(Q,\kappa)$. Assume that $g$ has the form*

$$g(x) = \kappa(x) \cdot \log h(x) \quad \text{with } h(x) = |x|^{O(1)}$$

[2] We use the prefix *uni* to indicate that the parameter depends on the cardinality of the *uni*verse.

*and $h(x) \geq 2$ for sufficiently large $x$. Then*

*$(Q, \kappa)$ has a polynomial kernelization if and only if $(Q, g)$ has a polynomial kernelization.*

*Proof* Clearly, every polynomial kernelization of $(Q, \kappa)$ is a polynomial kernelization of $(Q, g)$. Conversely, let $\mathbb{K}$ be a polynomial kernelization of $(Q, g)$. Choose $c, c' \in \mathbb{N}$ and an algorithm $\mathbb{A}$ solving $x \in Q$ in time $2^{\kappa(x)^c}|x|^{c'}$. We define a polynomial kernelization $\mathbb{K}'$ for $(Q, \kappa)$.

Fix $x_+ \in Q$ and $x_- \notin Q$. Let $x \in \{0,1\}^*$. If $\kappa(x) < (\log|x|)^{1/c}$, the algorithm $\mathbb{A}$ on input $x$ needs at most $|x|^{c'+1}$ steps. In this case we let $\mathbb{K}'(x)$ be $x_+$ or $x_-$ according to the answer of $\mathbb{A}$. Otherwise $\kappa(x)^c \geq \log|x|$. Then $|\mathbb{K}(x)| = (\kappa(x) \cdot \log h(x))^{O(1)} = (\kappa(x) \cdot \log|x|)^{O(1)} = \kappa(x)^{O(1)}$, so we can set $\mathbb{K}'(x) := \mathbb{K}(x)$. □

The reader familiar with [12] will realize that this result shows that any natural parameterized problem $(Q, \kappa)$ in EXPT with a canonical reparameterization of the specified form has a polynomial kernelization if and only if the problem $Q$ is self-compressible.

## 4 Excluding Parameter Non-increasing Kernelizations

In this section we exemplify how self-reducibility can be used to rule out *parameter non-increasing* polynomial kernelizations. This method is very simple and works under the assumption that P ≠ NP. We use it to give three natural examples of problems that do not have parameter non-increasing polynomial kernelizations, the first two being in EPT.

We will revisit these examples in Sect. 5. There we will see that these problems do not even have polynomial kernelizations using the stronger assumption that the polynomial hierarchy does not collapse to its third level.

The main result of this section is based on the following lemma.

**Lemma 4.1** *Let $(Q, \kappa)$ be a parameterized problem and assume that the 0th slice $Q(0) := \{x \in Q \mid \kappa(x) = 0\}$ is in* PTIME. *If there is a polynomial (subexponential) kernelization $\mathbb{K}$ such that for all $x$ with $\kappa(x) > 0$*

$$\kappa(\mathbb{K}(x)) < \kappa(x), \tag{5}$$

*then $Q \in$ PTIME ($(Q, \kappa) \in$ SUBEPT).*

*Proof* Let $\mathbb{K}$ be a kernelization satisfying (5). The following algorithm $\mathbb{A}$ decides $Q$ (using a polynomial time decision procedure $\mathbb{B}$ for $Q(0)$). Given an instance $x$ of $Q$, the algorithm $\mathbb{A}$ computes $\mathbb{K}(x), \mathbb{K}(\mathbb{K}(x)), \ldots$; by (5) after at most $\kappa(x)$ steps we obtain an instance $y$ with $\kappa(y) = 0$; hence ($x \in Q \iff y \in Q(0)$); now $\mathbb{A}$ simulates $\mathbb{B}$ on $y$.

If $\mathbb{K}$ was a polynomial kernelization, say, $|\mathbb{K}(x)| \leq \kappa(x)^c$, then, again by (5), all of $|\mathbb{K}(\mathbb{K}(x))|, |\mathbb{K}(\mathbb{K}(\mathbb{K}(x)))|, \ldots$ are bounded by $\kappa(x)^c$. Recall that parameterizations

are computable in polynomial time even if the result is encoded in unary. Hence $\kappa(x) = |x|^{O(1)}$. It follows that $\mathbb{A}$ runs in polynomial time.

If $\mathbb{K}$ was a subexponential kernelization, choose $c, d \in \mathbb{N}$ and a computable, non-decreasing and unbounded $\iota : \mathbb{N} \to \mathbb{N}$ such that $\mathbb{K}(x)$ is computable in time $|x|^c$ and $|\mathbb{K}(x)| \leq 2^{d \cdot \kappa(x)/\iota(\kappa(x))}$. Then, by (5), the computation of $y$ by the algorithm $\mathbb{A}$ needs time at most

$$|x|^c + 2^{c \cdot d \cdot \kappa(x)/\iota(\kappa(x))} + 2^{c \cdot d \cdot (\kappa(x)-1)/\iota(\kappa(x)-1)} + \cdots + 2^{c \cdot d \cdot 1/\iota(1)}. \tag{6}$$

Write $k := \kappa(x)$. Then

$$\begin{aligned} \sum_{\ell=1}^{k} 2^{c \cdot d \cdot \ell/\iota(\ell)} &= \sum_{\ell=1}^{\lfloor \sqrt{k} \rfloor} 2^{c \cdot d \cdot \ell/\iota(\ell)} + \sum_{\ell=\lfloor \sqrt{k} \rfloor + 1}^{k} 2^{c \cdot d \cdot \ell/\iota(\ell)} \\ &\leq \lfloor \sqrt{k} \rfloor \cdot 2^{c \cdot d \cdot \lfloor \sqrt{k} \rfloor} + k \cdot 2^{c \cdot d \cdot k/\iota(\lfloor \sqrt{k} \rfloor)}. \end{aligned}$$

Since the function $k \mapsto \iota(\lfloor \sqrt{k} \rfloor)$ is unbounded, the sum in (6) is bounded by $|x|^c + 2^{o^{\mathrm{eff}}(\kappa(x))}$; hence $(Q, \kappa) \in \mathrm{SUBEPT}$. □

**Theorem 4.2** *Let $(Q, \kappa)$ be a parameterized problem with $Q(0) \in \mathrm{PTIME}$. Assume that there is a polynomial reduction $R$ from $Q$ to itself which is* parameter decreasing, *that is, for all $x$ with $\kappa(x) > 0$*

$$\kappa(R(x)) < \kappa(x).$$

- *If $(Q, \kappa)$ has a parameter non-increasing polynomial kernelization, then $Q \in \mathrm{PTIME}$.*
- *If $(Q, \kappa)$ has a parameter non-increasing subexponential kernelization, then $(Q, \kappa) \in \mathrm{SUBEPT}$.*

*Proof* Let $R$ be as in the statement and let $\mathbb{K}$ be a parameter non-increasing polynomial (subexponential) kernelization of $(Q, \kappa)$. Then the composition $\mathbb{K} \circ R$, that is, the mapping $x \mapsto \mathbb{K}(R(x))$, is a polynomial (subexponential) kernelization of $(Q, \kappa)$ satisfying (5); hence, by the previous lemma, we get $Q \in \mathrm{PTIME}$ ($Q \in \mathrm{SUBEPT}$). □

We close this section with some applications.

*Example 4.3* The parameterized problem $p$-SAT has a parameter-decreasing polynomial reduction to itself.

*Proof* We define a parameter-decreasing polynomial reduction $R$ from $p$-SAT to itself as follows: Let $\alpha$ be a CNF formula. If $\alpha$ has no variables, we set $R(\alpha) := \alpha$. Otherwise let $X$ be the first variable in $\alpha$. We let $R(\alpha)$ be a formula in CNF equivalent to

$$\left(\alpha \frac{\mathrm{TRUE}}{X} \vee \alpha \frac{\mathrm{FALSE}}{X}\right),$$

where, for example, $\alpha\frac{\text{TRUE}}{X}$ is the formula obtained from $\alpha$ by replacing $X$ by TRUE everywhere. Clearly $R(\alpha)$ can be computed from $\alpha$ in polynomial time. □

*Example 4.4* The parameterized problem

| $p$-POINTED-PATH | |
|---|---|
| *Instance:* | A graph $G = (V, E)$, a vertex $v \in V$, and $k \in \mathbb{N}$. |
| *Parameter:* | $k$. |
| *Question:* | Does $G$ have a path of length $k$ starting at $v$? |

has a parameter-decreasing polynomial reduction to itself.

*Proof* The following is a parameter-decreasing polynomial self-reduction $R$ for $p$-POINTED-PATH: Let $(G, v, k)$ be an instance of $p$-POINTED-PATH and assume $k \geq 3$. For any path $P : v, v_1(P), v_2(P)$ of length 2 starting from $v$ let $G_P$ be the graph obtained from $G$ by deleting the two vertices $v, v_1(P)$ (and all the edges incident with one of these vertices). Let $H$ be the graph obtained from the disjoint union of all the graphs $G_P$ (where $P$ ranges over all paths of length 2 starting in $v$) by adding a new vertex $w$ and all edges $\{w, v_2(P)\}$. Then $H$ has a path of length $(k-1)$ starting at $w$ if and only if $G$ has a path of length $k$ starting at $v$. Hence we can set $R((G, v, k)) := (H, w, k-1)$. □

*Example 4.5* The parameterized problem

| $p$-BIPARTITE-CLIQUE | |
|---|---|
| *Instance:* | A graph $G = (V, E)$ and $k \in \mathbb{N}$. |
| *Parameter:* | $k$. |
| *Question:* | Does $G$ have a subgraph isomorphic to the $K_{k,k}$? Or equivalently, do there exist $A, B \subseteq V$ such that $\lvert A\rvert = \lvert B\rvert = k$ and for every $u \in A$, $v \in B$ we have $\{u, v\} \in E$? |

has a parameter-decreasing polynomial reduction to itself.

*Proof* Let $G = (V, E)$ and $k \in \mathbb{N}$ and let $e = \{u, v\}$ be an edge of $G$. We create the following bipartite graph $G_e$: on the left it contains a copy $(u', \ell)$ for each neighbor $u'$ of $v$ in $G$ with $u' \neq u$, and on the right it contains a copy $(v', r)$ for each neighbor $v'$ of $u$ in $G$ with $v' \neq v$; we create an edge in $G_e$ between $(u', \ell)$ and $(v', r)$ if and only if there is an edge between $u'$ and $v'$ in $G$. Let $G'$ be the disjoint union of the graphs $G_e$ for $e \in E$. Then

$G'$ contains a subgraph isomorphic to $K_{k-1,k-1}$
$\iff$ there is $e \in E$ such that $G_e$ contains a subgraph isomorphic to $K_{k-1,k-1}$
$\iff$ there is $e \in E$ such that $G$ contains a subgraph isomorphic to $K_{k,k}$ including $e$
$\iff$ $G$ contains a subgraph isomorphic to $K_{k,k}$. □

It is open if $p$-BIPARTITE-CLIQUE is fixed-parameter tractable (the question is posed e.g. in [6, p. 355]), thus at present it is not known if it has a kernelization at

all. The construction in the above example shows that it unlikely has a parameter non-increasing polynomial kernelization:

**Corollary 4.6**

(1) *If* P ≠ NP, *then* $p$-SAT, $p$-POINTED-PATH, *and* $p$-BIPARTITE-CLIQUE *have no parameter non-increasing polynomial kernelizations*.
(2) *If* ETH[3] *holds*, *then* $p$-SAT *and* $p$-POINTED-PATH *have no parameter non-increasing subexponential kernelizations*.

*Proof* Part (1) is immediate by Theorem 4.2, as all three underlying problems are NP-hard.[4] Moreover, we know by this corollary that if one of the three problems has a parameter non-increasing subexponential kernelization, then it is in SUBEPT. However then ETH would fail in the case of $p$-SAT by [13], in the case of $p$-POINTED-PATH by [3]. □

## 5 Excluding Polynomial Kernelizations

As mentioned in the Introduction and Sect. 3.1, there are polynomial kernelizations which are not parameter non-increasing. We cannot apply the technique of the previous section to rule out such kernelizations. Furthermore, many parameterized problems apparently do not have parameter-decreasing polynomial self-reductions, so that again we cannot apply the main result of the previous section. We use the method of [1, 7] to deal with these situations.

The following type of reductions that preserve polynomial kernels was introduced in [7] (based on a notion of [12]) under the name "$W$-reductions."

**Definition 5.1** Let $(Q, \kappa)$ and $(Q', \kappa')$ be parameterized problems. A *polynomial reduction* from $(Q, \kappa)$ to $(Q', \kappa')$ is a polynomial reduction $R$ from $Q$ to $Q'$ such that

$$\kappa'(R(x)) = \kappa(x)^{O(1)}.$$

We then write $R : (Q, \kappa) \leq^p (Q', \kappa')$. Furthermore $(Q, \kappa) \leq^p (Q', \kappa')$ means that there is a polynomial reduction from $(Q, \kappa)$ to $(Q', \kappa')$.

*Example 5.2* *uni*-PATH $\leq^p$ $p$-SAT.

*Proof* Let $(G, k)$ with $G = (V, E)$ be an instance of *uni*-PATH. We may assume that $V = \{0, 1, \ldots, n-1\}$ and (by adding isolated points if necessary) that $n$ is a power of 2. We will assign to $(G, k)$ a formula $\alpha$ in CNF containing variables $X_{s,i}$ with $s \in [\log n]$ and $i \in [k]$ with the intended meaning "the $s$th bit of the $i$th vertex of a path of length $k$ is 1." For $i, j \in [k]$, $i \neq j$, one has to express by a clause that the

[3] Recall that ETH, the exponential time hypothesis, is the statement that SAT is not decidable in time $2^{o(n)}$.

[4] In the case of $p$-BIPARTITE-CLIQUE, see [14].

selected vertices as $i$th and $j$th point of the path are distinct and for $i \in [k-1]$ that the $i$th and the $(i+1)$th selected vertices are related by an edge. For example the second one may be expressed by letting, for every $i \in [k-1]$ and every $u, v \in V$ with $\{u, v\} \notin E$,

$$\bigvee_{s \in [\log n]} \neg X_{s,i}^{\mathrm{bit}(s,u)} \vee \bigvee_{s \in [\log n]} \neg X_{s,i+1}^{\mathrm{bit}(s,v)},$$

be a clause of $\alpha$, where $\mathrm{bit}(s, u)$ denotes the $s$th bit in the binary representation of $u$ of length $\log n$ and where $X^1 := X$ and $X^0 := \neg X$ for every variable $X$.

Then $G$ has a path of length $k$ if and only if $\alpha$ is satisfiable. As $\alpha$ has $k \cdot \log |V|$ variables, the mapping $(G, k) \mapsto \alpha$ is a polynomial reduction. □

*Example 5.3* ([12]) $p$-SAT $\leq^p$ *uni*-DOMINATING-SET.

Polynomial reductions preserve polynomial kernelizations in the following sense:

**Lemma 5.4** *Let $(Q, \kappa)$ and $(Q', \kappa')$ be parameterized problems with*

$$(Q, \kappa) \leq^p (Q', \kappa') \quad \text{and} \quad Q' \leq^p Q.$$

*If $(Q', \kappa')$ has a polynomial kernelization, then $(Q, \kappa)$ has a polynomial kernelization.*

Note that $Q' \leq^p Q$ is always satisfied for NP-complete problems $Q$ and $Q'$.

*Proof of Lemma 5.4* Let $R : (Q, \kappa) \leq^p (Q', \kappa')$ and $S : Q' \leq^p Q$. Assume that $\mathbb{K}$ is a polynomial kernelization for $(Q', \kappa')$. Then $S \circ \mathbb{K} \circ R$ is a polynomial kernelization for $(Q, \kappa)$, as for all $x \in \{0, 1\}^*$

$$|S(\mathbb{K}(R(x)))| = |\mathbb{K}(R(x))|^{O(1)} = \kappa'(R(x))^{O(1)} = \kappa(x)^{O(1)}.$$ □

In order to exclude polynomial kernelizations using the previous lemma one needs a primal problem without a polynomial kernelization. A central ingredient needed to obtain such problems was provided by Fortnow and Santhanam [7]. It is contained in Theorem 5.6.

**Definition 5.5** ([1]) Let $Q, Q' \subseteq \{0, 1\}^*$ be classical problems. A *distillation from $Q$ in $Q'$* is a polynomial time algorithm $\mathbb{D}$ that receives as inputs finite sequences $\bar{x} = (x_1, \ldots, x_t)$ with $x_i \in \{0, 1\}^*$ for $i \in [t]$ and outputs a string $\mathbb{D}(\bar{x}) \in \{0, 1\}^*$ such that

(1) $|\mathbb{D}(\bar{x})| = (\max_{i \in [t]} |x_i|)^{O(1)}$;
(2) $\mathbb{D}(\bar{x}) \in Q'$ if and only if for some $i \in [t] : x_i \in Q$.

If $Q' = Q$ we speak of a *self-distillation*. We say that $Q$ *has a distillation* if there is a distillation from $Q$ in $Q'$ for some $Q'$.

"Self-distillations" without property (1) has been called $\mathrm{OR}_\omega$ functions in [2]. Their importance for classical complexity has been studied in various papers (see [2] and its references). Every NP-complete problem $Q$ has an $\mathrm{OR}_\omega$ function: Take a polynomial time reduction of the problem $\{(x_1, \ldots, x_t) \mid t \in \mathbb{N} \text{ and } x_i \in Q \text{ for some } i \in [t]\}$ to $Q$. However:

**Theorem 5.6** ([7]) *If* $\mathrm{PH} \neq \Sigma_3^{\mathrm{P}}$ (*that is, if the polynomial hierarchy* PH *does not collapse to its third level*), *then no* NP*-hard problem has distillations.*

Clearly each problem in PTIME has a self-distillation. However, we prove that NP-problems with a self-distillation are not necessarily in PTIME (under some plausible complexity assumption):

**Proposition 5.7** *If* $\mathrm{NE} \neq \mathrm{E}$, *then there is a problem in* $\mathrm{NP} \setminus \mathrm{P}$ *that has a self-distillation.*

By E and NE we denote the class of problems $Q$ such that $x \in Q$ is solvable by a deterministic algorithm and a nondeterministic algorithm, respectively, in time $2^{O(|x|)}$.

*Proof of Proposition 5.7* Let $Q_0 \subseteq \{0,1\}^*$ be a language in $\mathrm{NE} \setminus \mathrm{E}$. We assume that each yes instance of $Q_0$ starts with a 1, and can thus be viewed as a natural number in binary. For $n \in \mathbb{N}$ let $bin(n)$ denote its binary representation. We set

$$Q := \{1^n \mid bin(n) \in Q_0\}.$$

It is easy to see that $Q \in \mathrm{NP} \setminus \mathrm{P}$. Now let $Q'$ be the "OR-closure" of $Q$, that is

$$Q' := \{(x_1, \ldots, x_m) \mid m \geq 1 \text{ and } x_i \in Q \text{ for some } i \in [m]\}.$$

Again it is easy to see that $Q' \in \mathrm{NP} \setminus \mathrm{P}$. We claim that $Q'$ has a self-distillation.

Let $(x_{11}, \ldots, x_{1m_1}), \ldots, (x_{t1}, \ldots, x_{tm_t})$ be a sequence of instances of $Q'$. We can assume that all $x_{ij}$ are sequences of 1s (otherwise we simply ignore those which are not). Let $n$ be the maximal length of the $x_{ij}$. Then

$$\{x_{11}, \ldots, x_{1m_1}, \ldots, x_{t1}, \ldots, x_{tm_t}\} = \{y_1, \ldots, y_q\}$$

for some $q \leq n$. Thus $(y_1, \ldots, y_q)$ has length $O(n^2)$. Clearly $(y_1, \ldots, y_q)$ is in $Q'$ if and only if $(x_{i1}, \ldots, x_{im_i}) \in Q'$ for some $i \in [t]$. □

To see how Theorem 5.6 (and the polynomial reductions) can be used to exclude polynomial kernelizations we include applications from [1] and [7].

**Corollary 5.8** ([1]) *If* $\mathrm{PH} \neq \Sigma_3^{\mathrm{P}}$, *then* $p$-PATH *has no polynomial kernelizations.*

*Proof* We assume that $p$-PATH has a polynomial kernelization $\mathbb{K}$ and show that then the (classical) problem PATH has a self-distillation. In fact, let $(G_1, k_1), \ldots, (G_t, k_t)$

be instances of PATH. Let $k := 1 + 2 \cdot \max_{i \in [t]} k_i$. Let $i \in [t]$. By adding to $G_i$ a path of length $k - k_i - 1$ with one endpoint connected to all vertices of $G_i$ we obtain a graph $G_i'$ such that the instance $(G_i', k)$ of PATH is equivalent to $(G_i, k_i)$. Let $G$ be the disjoint union of all the graphs $G_i'$. Clearly, $G$ has a path of length $k$ if and only if there exists an $i \in [t]$ such that $G_i'$ has a path of length $k$ and hence, if and only if there exists an $i \in [t]$ such that $G_i$ has a path of length $k_i$. As $|\mathbb{K}((G, k))|$ is polynomially bounded in $k$ and hence in $\max_{i \in [t]} \|(G_i, k_i)\|$, the mapping $(G_1, k_1), \ldots, (G_t, k_t) \mapsto \mathbb{K}((G, k))$ is a self-distillation of PATH. Here, by $\|(G, k)\|$ we denote the size of $(G, k)$, that is the length of a (reasonable) encoding of the instance $(G, k)$. □

**Corollary 5.9** ([7]) *If* PH $\neq \Sigma_3^{\mathrm{P}}$, *then* $p$-SAT *and uni*-DOMINATING-SET *have no polynomial kernelizations*.

*Proof* Assume PH $\neq \Sigma_3^{\mathrm{P}}$. By the previous corollary we know that $p$-PATH has no polynomial kernelization. Hence, as $p$-PATH $\in$ EPT, its canonical reparameterization *uni*-PATH has no polynomial kernelization by Proposition 3.10. The claims follow from Examples 5.2 and 5.3 by Lemma 5.4. □

The proof of Corollary 5.8 consists of two parts. Let $(G_1, k_1), \ldots, (G_t, k_t)$ and $(G, k)$ be as there. In the first part we show that $\mathbb{O}$ with $\mathbb{O}((G_1, k_1), \ldots, (G_t, k_t)) := (G, k)$ is an OR for $p$-PATH in the sense of the following definition.

**Definition 5.10** Let $(Q, \kappa)$ be a parameterized problem. An *OR for* $(Q, \kappa)$ is a polynomial time algorithm $\mathbb{O}$ that for every finite tuple $\bar{x} = (x_1, \ldots, x_t)$ of instances of $Q$ outputs an instance $\mathbb{O}(\bar{x})$ of $Q$ such that

(1) $\kappa(\mathbb{O}(\bar{x})) = (\max_{i \in [t]} |x_i|)^{O(1)}$;
(2) $\mathbb{O}(\bar{x}) \in Q$ if and only if for some $i \in [t]$: $x_i \in Q$.

The second part of the proof of Corollary 5.8 shows the following lemma (there the argument is presented for $(Q, \kappa) := p$-PATH).

**Lemma 5.11** *Assume that* $(Q, \kappa)$ *has an* OR $\mathbb{O}$ *and a polynomial kernelization* $\mathbb{K}$. *Then* $\mathbb{D}$ *with*

$$\mathbb{D}(x_1, \ldots, x_t) := \mathbb{K}(\mathbb{O}(x_1, \ldots, x_t))$$

*is a self-distillation of* $Q$.

Hence by Theorem 5.6:

**Corollary 5.12** *Assume that* $(Q, \kappa)$ *has an* OR $\mathbb{O}$ *and that* $Q$ *is* NP*-hard. If* PH $\neq \Sigma_3^{\mathrm{P}}$, *then* $(Q, \kappa)$ *has no polynomial kernelizations*.

### 5.1 Some Further Applications

Perhaps the reader might object that the proof of Corollary 5.8 is algorithmically not convincing, as the OR function used in the first part essentially yields the disjoint union of given graphs, while probably any reasonable algorithm for determining

whether a graph has a path of a given length will first compute its connected components and then check these components for such a path. Hence the question arises whether the path problem for the class of *connected* graphs has polynomial kernelizations. Assuming PH $\neq \Sigma_3^{\mathrm{P}}$, we deny this, we even show that the path problem for the class PLAN-CONN of *planar* connected graphs has no polynomial kernelizations:

**Proposition 5.13** *If* PH $\neq \Sigma_3^{\mathrm{P}}$, *then* $p$-PATH(PLAN-CONN) *has no polynomial kernelizations*.

To show this claim we show in a first step:

**Lemma 5.14** *If* PH $\neq \Sigma_3^{\mathrm{P}}$, *then* $p$-POINTED-PATH(PLAN-CONN) *has no polynomial kernelizations*.

*Proof* We show that $p$-POINTED-PATH(PLAN-CONN) has an OR (then our claim follows from Corollary 5.12). Let $(G_1, v_1, k_1) \ldots, (G_t, v_t, k_t)$ be instances of the problem. First let us assume that for every $i \in [t]$, we take a drawing of $G_i$ such that $v_i$ lies on the boundary of its outer face.[5] Let $k := \max_{i\in[t]} k_i$. By adding to every $G_i$ a path of length $k - k_i$ starting in $v_i$ and ending in a vertex $v_i'$ we obtain an equivalent instance $(G_i', v_i', k)$. Let $G$ be the planar and connected graph obtained from the disjoint union of the $G_i'$s by adding a new vertex $v$ and edges from $v$ to all $v_i'$. It is easy to verify that

$G$ has a path of length $k+1$ starting at $v$

$\iff$ there exists an $i \in [t]$ such that $G_i$ has a path of length $k$ starting at $v_i$.

Hence we can set $\mathbb{O}((G_1, v_1, k_1), \ldots, (G_t, v_t, k_t)) := (G, v, k+1)$. □

*Remark 5.15* For the NP-complete problem $p$-POINTED-PATH introduced in Sect. 4, obviously we have

$$p\text{-POINTED-PATH(PLAN-CONN)} \leq^p p\text{-POINTED-PATH}.$$

Therefore, by Lemma 5.14 and Lemma 5.4, if PH $\neq \Sigma_3^{\mathrm{P}}$, then $p$-POINTED-PATH has no polynomial kernelizations.

*Proof of Proposition 5.13* We show that there is a polynomial reduction from $p$-POINTED-PATH(PLAN-CONN) to $p$-PATH(PLAN-CONN). Then our claim follows from the previous lemma by Lemma 5.4.

Let $(G, v, k)$ be an instance of $p$-POINTED-PATH(PLAN-CONN). Using the connectedness of $G$ one easily verifies:

$$\begin{aligned} &\text{if } G \text{ contains a path of length } 2k-1, \\ &\quad \text{then } G \text{ contains a path of length } k \text{ starting at } v. \end{aligned} \tag{7}$$

[5]Note that we actually do not need to compute the drawing of $G_i$. It is only needed to show that the graph $G$ we construct is planar.

We add to $G$ in $v$ a path $P$ of length $k-1$ of new vertices, thereby obtaining the planar and connected graph $G'$. We show that

$$(G, v, k) \in p\text{-POINTED-PATH(PLAN-CONN)}$$
$$\iff (G', 2k-1) \in p\text{-PATH(PLAN-CONN)}.$$

Then $(G, v, k) \mapsto (G', 2k-1)$ is the desired reduction.

Assume first that $G$ has a path of length $k$ starting at $v$. Clearly, then $G'$ has a path of length $2k-1$. Conversely, let $P'$ be a path of length $2k-1$ in $G'$. If $v$ is a vertex of $P'$, then the vertices of $P'$ contained in $G$ constitute a path of $G$ of length at least $k$ starting at $v$. If $v$ is not a vertex of $P'$, then $P'$ is a path in $G$ and by (7) the graph $G$ contains a path of length $k$ starting at $v$. □

We have already seen in Corollary 4.6 that $p$-BIPARTITE-CLIQUE has no parameter non-increasing polynomial kernelizations assuming NP $\neq$ P. Now we show that $p$-BIPARTITE-CLIQUE has an OR; thus, by its NP-hardness [14], it is unlikely that it has polynomial kernelizations.

**Proposition 5.16** *If* PH $\neq \Sigma_3^{\mathrm{P}}$, *then* $p$-BIPARTITE-CLIQUE *has no polynomial kernelizations.*

As a technical tool, we first show that there is a "parameter increasing" self-reduction for $p$-BIPARTITE-CLIQUE.

**Lemma 5.17** *There is an algorithm* $\mathbb{A}$ *such that for every graph* $G$ *and* $k \leq k' \in \mathbb{N}$, *the algorithm* $\mathbb{A}$ *computes in time polynomial in* $\|G\| + k'$ *a graph* $G'$ *such that*

$$G \text{ has a subgraph isomorphic to } K_{k,k}$$
$$\iff G' \text{ has a subgraph isomorphic to } K_{k',k'}.$$

*Proof* Let $G$, $k$, and $k'$ be as stated above. First we construct a bipartite graph $G_b = (V_b, E_b)$ with

$$V_b := V \times \{0, 1\},$$
$$E_b := \big\{\{(u, 0), (v, 1)\} \bigm| \{u, v\} \in E\big\}.$$

It is easy to verify that

$$(G, k) \in p\text{-BIPARTITE-CLIQUE} \iff (G_b, k) \in p\text{-BIPARTITE-CLIQUE}.$$

Now the desired graph $G' = (V', E')$ is defined by

$$V' := V_b \dot\cup \big\{(p, i) \bigm| k < p \leq k' \text{ and } i \in \{0, 1\}\big\},$$
$$E' := E_b \cup \big\{\{(u, i), (p, 1-i)\} \bigm| u \in V, i \in \{0, 1\}, \text{ and } k < p \leq k'\big\}.$$ □

*Proof of Proposition 5.16* We show that $p$-BIPARTITE-CLIQUE has an OR. Let $(G_1, k_1), \ldots, (G_t, k_t)$ be instances of $p$-BIPARTITE-CLIQUE. By Lemma 5.17, we can assume that $k_1 = \cdots = k_t =: k$. Moreover, let $G$ be the disjoint union of all $G_i$. Clearly

$$G \text{ has a subgraph isomorphic to } K_{k,k}$$
$$\iff \text{ there exists an } i \in [t] \text{ such that } G_i \text{ has a subgraph isomorphic to } K_{k_i,k_i}.$$

□

## 6 Strong Lower Bounds

In this section and the next one, by a careful analysis of the proof of Theorem 5.6 as given in [7], we obtain improvements, which yield better lower bounds for kernelizations. In particular for the path problem we will show:

**Theorem 6.1** *Let $\varepsilon > 0$. If $\mathrm{PH} \neq \Sigma_3^{\mathrm{P}}$, then there is* no *polynomial reduction from* PATH *to itself computing for each instance $(G, k)$ of* PATH *an instance $(G', k')$ with*

$$\|G'\| = k^{O(1)} \cdot \|G\|^{1-\varepsilon}.$$

We define:

**Definition 6.2** Let $\varepsilon > 0$. A parameterized problem $(Q, \kappa)$ *has an $\varepsilon$ self-reduction* if there is a polynomial reduction from $Q$ to itself that assigns to every instance $x$ of $Q$ an instance $y$ with

$$|y| = \kappa(x)^{O(1)} \cdot |x|^{1-\varepsilon}.$$

Note that it is not required that the parameter of $y$ is bounded in terms of the parameter of $x$.

Clearly, if $(Q, \kappa)$ has a polynomial kernelization, then $(Q, \kappa)$ has an $\varepsilon$ self-reduction for every $\varepsilon > 0$. The converse does not hold, as shown in Sect. 8.2. Now we can rephrase Theorem 6.1 by saying that, if $\mathrm{PH} \neq \Sigma_3^{\mathrm{P}}$, then for every $\varepsilon > 0$ the problem $p$-PATH has no $\varepsilon$ self-reductions. This result will be a special instance of a more general result stating similar lower bounds for problems with a linear OR.

**Definition 6.3** Let $(Q, \kappa)$ be a parameterized problem. A *linear OR for* $(Q, \kappa)$ is a polynomial time algorithm $\mathbb{O}$ that for every finite tuple $\bar{x} = (x_1, \ldots, x_t)$ of instances of $Q$ outputs an instance $\mathbb{O}(\bar{x})$ of $Q$ such that

(1) $|\mathbb{O}(\bar{x})| = t \cdot (\max_{i \in [t]} |x_i|)^{O(1)}$;
(2) $\kappa(\mathbb{O}(\bar{x})) = (\max_{i \in [t]} |x_i|)^{O(1)}$;
(3) $\mathbb{O}(\bar{x}) \in Q$ if and only if for some $i \in [t]$: $x_i \in Q$.

Hence a linear OR is an OR with the additional property (1).

*Examples 6.4* The parameterized problems $p$-PATH and $p$-POINTED-PATH(PLAN-CONN) have a linear OR. In fact, the ORs defined in the proofs of Corollary 5.8 and of Lemma 5.14 are linear ones.

**Theorem 6.5** *Let $\varepsilon > 0$. Let $(Q, \kappa)$ be a parameterized problem with a linear* OR *and with* NP*-hard $Q$. If* PH $\neq \Sigma_3^{\mathrm{P}}$*, then the problem $(Q, \kappa)$ has no $\varepsilon$ self-reductions.*

In particular, if PH $\neq \Sigma_3^{\mathrm{P}}$, then the problems mentioned in Examples 6.4 do not have $\varepsilon$ self-reductions. This proves Theorem 6.1.

### 6.1 Some Further Applications

Before proving Theorem 6.5 in Sect. 6.2 we first give more applications.

*Example 6.6* The parameterized problem $p$-SAT has a linear OR.

*Proof* We define a linear OR $\mathbb{O}$. Let $\alpha_1, \ldots, \alpha_t$ be CNF formulas, say, $\alpha_i$ a formula with $n_i$ variables. We set

$$n := \max_{i \in [t]} n_i \quad \text{and} \quad m := \max_{i \in [t]} |\alpha_i|.$$

We may assume that all $\alpha_i$ have variables in $\{X_1, \ldots, X_n\}$ and that $\log t$ is a natural number (if $t$ is not a power of two we duplicate one of the formulas for an appropriate number of times).

If $t \geq 2^n$, the algorithm $\mathbb{O}$ proves whether one of the $\alpha_i$s is satisfiable (by systematically checking all assignments) and outputs a CNF formula $\mathbb{O}(\alpha_1, \ldots, \alpha_t)$ satisfying condition (3) of the preceding definition.

Assume $t < 2^n$. We introduce $\log t$ new variables $Y_1, \ldots, Y_{\log t}$. For $i \in [t]$ we set

$$\beta_i := \bigwedge_{s \in [\log t]} Y_s^{\mathrm{bit}(s,i)}$$

(recall that $\mathrm{bit}(s, i)$ denotes the $s$th bit in the binary representation of $i$ and that $X^1 = X$ and $X^0 = \neg X$ for every variable $X$).

We bring each $(\beta_i \to \alpha_i)$ into conjunctive normal form: Assume $\alpha_i = \bigwedge_\ell \bigvee_{\ell'} \lambda_{\ell\ell'}$ with literals $\lambda_{\ell\ell'}$, then $(\beta_i \to \alpha_i)$ is equivalent to

$$\gamma_i := \bigwedge_\ell \left( \bigvee_{s \in [\log t]} Y_s^{1-\mathrm{bit}(s,i)} \vee \bigvee_{\ell'} \lambda_{\ell\ell'} \right).$$

We let $\gamma$ be the CNF formula $\gamma := \bigwedge_{i \in [t]} \gamma_i$. We set $\mathbb{O}(\alpha_1, \ldots, \alpha_t) := \gamma$.

Clearly $\mathbb{O}$ is computable in polynomial time. Furthermore, by construction the formula $\mathbb{O}(\alpha_1, \ldots, \alpha_t)$ is equivalent to $\bigwedge_{i \in [t]} (\beta_i \to \alpha_i)$. Because any assignment to $Y_1, \ldots, Y_{\log t}$ satisfies exactly one of the $\beta_i$s, the formula $\mathbb{O}(\alpha_1, \ldots, \alpha_t)$ is satisfiable if and only if there is an $i \in [t]$ such that $\alpha_i$ is satisfiable; hence condition (3) of Definition 6.3 is satisfied. Furthermore, $\mathbb{O}$ also satisfies the conditions (1) and (2).

For (2) note that $\gamma$ has $n + \log t$ variables. By our assumption on $t$, we have $n + \log t \leq 2n \leq 2m$. For (1) note that each $\gamma_i$ has length $O(m \cdot (m + \log t))$ and hence, $\mathbb{O}(\alpha_1, \ldots, \alpha_t)$ has length $O(m^3)$. □

*Example 6.7* The parameterized problem

| | |
|---|---|
| $p$-CYCLE | |
| *Instance:* | A graph $G$ and $k \in \mathbb{N}$. |
| *Parameter:* | $k$. |
| *Question:* | Does $G$ have a cycle of length $k$? |

has a linear OR. This example is due to Martin Grohe [10].

If $(G_1, k_1), \ldots, (G_t, k_t)$ are instances of $p$-CYCLE with the same parameter, $k_1 = \cdots = k_t =: k$, then for the disjoint union $G$ of the $G_i$s we have $(G, k) \in p$-CYCLE if and only if $(G_i, k_i) \in p$-CYCLE for some $i \in [t]$. With the following observations we will reduce the general case to the case of instances with the same parameter.

So let $(G_1, k_1), \ldots, (G_t, k_t)$ be instances of $p$-CYCLE. We set $p := \max_{i \in [t]} |V_i|$, where $V_i$ is the vertex set of $G_i$, and $k := \max_{i \in [t]} k_i$. For $i \in [t]$ and $v \in V_i$ we let $G_i(v)$ be the graph obtained from $G_i$ by replacing the vertex $v$ by a path $P_i(v)$ of length $p + k - k_i$ of new vertices and by replacing edges of $G_i$ of the form $\{v, w\}$ by two edges, namely by edges incident with $w$ and one of the endpoints of the path $P_i(v)$. Clearly,

$$G_i(v) \text{ has a cycle of length } p + k \iff G_i \text{ has a cycle through } v \text{ of length } k_i.$$

Hence, we can set $\mathbb{O}((G_1, k_1), \ldots, (G_t, k_t)) := (G, p + k)$, where $G$ denotes the disjoint union of the graphs $G_i(v)$ for all $i \in [t]$ and $v \in V_i$.

*Example 6.8* The parameterized problem *uni*-CLIQUE has a linear OR.

*Proof* Let $(G_1, k_1), \ldots, (G_t, k_t)$ be instances of *uni*-CLIQUE. Of course, we can assume that $k_i \leq |V_i|$, where $V_i$ is the set of vertices of $G_i$. Let $k := \max_{i \in [t]} k_i$. By adding a clique of $k - k_i$ new vertices to $G_i$ and connecting all new vertices to all old vertices in $V_i$ we can pass to an instance $(G_i', k)$ equivalent to $(G_i, k_i)$. Let $m := \max_{i \in [t]} |V_i'|$ $(\leq 2 \cdot \max_{i \in [t]} |V_i|)$.

If $t \geq 2^m$, by exhaustive search the algorithm $\mathbb{O}$ checks whether one of the $G_i'$s has a clique of size $k$; if this is the case $\mathbb{O}$ outputs $(G_i, k_i)$ for such a $G_i'$ and otherwise it outputs, say, $(G_1, k_1)$.

Assume that $t < 2^m$. We set $\mathbb{O}((G_1, k_1), \ldots, (G_t, k_t)) := (G, k)$, where $G$ denotes the disjoint union of the graphs $G_i'$. Clearly, $\mathbb{O}$ is computable in polynomial time and condition (3) is satisfied. For condition (1) note that we have for the set $V$ of vertices of $G$ the inequality $|V| \leq t \cdot m$. The parameter of $\mathbb{O}((G_1, k_1), \ldots, (G_t, k_t))$ is $k \cdot \log |V| \leq k \cdot \log(t \cdot m) \leq k \cdot (m + \log m) = O(m^2)$. □

*Example 6.9* The parameterized problem *uni*-DOMINATING-SET has a linear OR.

*Proof* Let $(G_1, k_1), \ldots, (G_t, k_t)$ be instances of *uni*-DOMINATING-SET. Let $k := \max_{i \in [t]} k_i$. By adding $k - k_i$ isolated vertices, we can pass to equivalent instances

$(G'_1, k), \ldots, (G'_t, k)$. Let $G'_i = (V'_i, E'_i)$. We may assume that $t > k$ and that the vertex sets $V'_i$ are pairwise disjoint.

If $t \geq 2^m$, where $m := \max_{i\in[t]} |V'_i|$, the algorithm $\mathbb{O}$ checks by exhaustive search whether one of the $G'_i$s has a dominating set of size $k$; if so $\mathbb{O}$ outputs $(G_i, k_i)$ for such a $G'_i$ and otherwise it outputs $(G_1, k_1)$.

Assume that $t < 2^m$. For $i \in [t]$ and $j \in [0, k] := \{0, 1, \ldots, k\}$ let $V'_i(j)$ be a copy of $V'_i$, say,

$$V'_i(j) := \{(v, j) \mid v \in V'_i\}.$$

Let $G = (V, E)$ be the graph with vertex set

$$V := \bigcup_{s\in[\log t]} \{s(-), s(0), s(1)\} \cup \bigcup_{i\in[t],\, j\in[0,k]} V'_i(j).$$

The edge set $E$ contains

- edges that make $\{s(-), s(0), s(1)\}$ a clique for $s \in [\log t]$;
- for $s \in [\log t]$ and $i \in [t]$ edges from $s(1)$ to all vertices in $V'_i(0)$ if $\mathrm{bit}(s, i) = 0$ and edges from $s(0)$ to all vertices in $V'_i(0)$ if $\mathrm{bit}(s, i) = 1$;
- for $i, i' \in [t]$, $v \in V'_i$, $w \in V'_{i'}$, and $j, j' \in [0, k]$ the edge $\{(v, j), (w, j')\}$ if
  - $i \neq i'$ and $j = j' > 0$ or
  - $i = i'$ and $\{v, w\} \in E_i$ or
  - $i = i'$, $j \neq j'$ and $v = w$.

We claim that

$$\begin{aligned} &(G, k + \log t) \in \textit{uni}\text{-DOMINATING-SET} \\ &\quad\Longleftrightarrow \text{ there is an } i \in [t]\colon\ (G'_i, k) \in \textit{uni}\text{-DOMINATING-SET}. \end{aligned} \tag{8}$$

For the backward direction assume for $i \in [t]$ that $\{v_1, \ldots, v_k\}$ is a dominating set in $G'_i$. Then

$$\{(v_1, 1), \ldots, (v_k, k)\} \cup \{s(\mathrm{bit}(s, i)) \mid s \in [\log t]\}$$

is a dominating set of $G$.

For the forward direction let $X$ be a dominating set of $G$ of size $k + \log t$. For $s \in [\log t]$ in order to dominate the point $s(-)$ we see that at least one point of the clique $\{s(-), s(0), s(1)\}$ has to be contained in $X$.

Clearly, as $k < t$, there is an $i_0 \in [t]$ such that

$$X \cap \bigcup_{j\in[0,k]} V'_{i_0}(j) = \emptyset.$$

For $j \in [k]$ (in particular $j \neq 0$), in order to dominate the elements of $V'_{i_0}(j)$, the set $X$ must contain an element of the form $(v_j, j)$ with $v_j \in V'_{i_j}$ for some $i_j \neq i_0$. Moreover, as $X$ only contains $k + \log t$ elements, the vertex $v_j$ (and hence $i_j$) are uniquely determined by $j$. Then it is not hard to see that the set $\{v_j \mid j \in [k] \text{ and } i_j = i_1\}$ is a dominating set in $G'_{i_1}$. This finishes the proof of the equivalence (8).

We set $\mathbb{O}((G_1,k_1),\ldots,(G_t,k_t)) := (G,k)$. That $\mathbb{O}$ also satisfies condition (2) of a linear OR is shown as in the case of *uni*-CLIQUE. □

*Example 6.10* The problem *alpha*-LCS has a linear OR. Here *alpha*-LCS denotes the canonical parameterization of the longest common subsequence problem:

| *alpha*-LCS | |
|---|---|
| *Instance:* | An alphabet $\Sigma$, strings $X_1,\ldots,X_\ell \in \Sigma^*$, and $m \in \mathbb{N}$. |
| *Parameter:* | $m \cdot \log\lvert\Sigma\rvert$. |
| *Question:* | Is there a common subsequence of $X_1,\ldots,X_\ell$ of length $m$? |

*Proof* Let $(\Sigma_1, X_{11},\ldots,X_{1\ell_1},m_1)\ldots(\Sigma_t,X_{t1},\ldots,X_{t\ell_t},m_t)$ be instances of *alpha*-LCS. We can assume that $\ell_1 = \cdots = \ell_t = \ell$ (by repeating a sequence if necessary) and that $m_1 = \cdots = m_t = m$ (by adding $c_i^{m-m_i}$ to each $X_{ij}$ for some new letter $c_i$). Moreover we can assume that the alphabets $\Sigma_i$ are disjoint. Now we consider the $\ell$ strings over $\Sigma_1 \cup \cdots \cup \Sigma_t$

$$X_{11}X_{21}\cdots X_{t1}, \quad X_{12}X_{22}\cdots X_{t2}, \quad \ldots \quad X_{1\ell}X_{2\ell}\cdots X_{t\ell}$$

and the string $X_{t1}X_{(t-1)1}\cdots X_{11}$.

One easily verifies that these $(\ell+1)$ strings have a common subsequence of length $m$ if and only if for some $i \in [t]$ the strings $X_{i1},\ldots,X_{i\ell_i}$ have one (for the forward direction note that a common subsequence of $X_{11}X_{21}\cdots X_{t1}$ and $X_{t1}X_{(t-1)1}\cdots X_{11}$ is a sequence over $\Sigma_i$ for some $i \in [t]$). Now, if $t \geq \max_{i\in[t]}|\Sigma_i|^m$ we determine the value of $\mathbb{O}$ by exhaustive search and otherwise, we use the set of strings just constructed. □

Even though we could add further examples of parameterized problems with a linear OR, there are also many problems where we do not know whether they have a linear OR. We just mention one example, the problem *uni*-RED/BLUE-NONBLOCKER, the canonical reparameterization of the problem *p*-RED/BLUE-NONBLOCKER.

## 6.2 Proof of Theorem 6.5

It will be convenient to reformulate Theorem 6.5. For this purpose we need some further notions.

**Definition 6.11** A function $f : \mathbb{N} \to \mathbb{R}_{\geq 0}$ is *pseudo-linear* if there is some $c \in \mathbb{N}$ and some $\varepsilon \in \mathbb{R}$ with $\varepsilon > 0$ such that for all $t \in \mathbb{N}$

$$f(t) \leq c \cdot t^{1-\varepsilon}.$$

The property that we need of pseudo-linear functions is contained in the following lemma. It is easy to prove.

**Lemma 6.12** *Let $\varepsilon > 0$ and $f : \mathbb{N} \to \mathbb{R}_{\geq 0}$ be a pseudo-linear function. Then for every $c \in \mathbb{N}$ there exists a $d \in \mathbb{N}$ such that for sufficiently large $n$ we have*

$$f(n^d) \cdot n^c + 1 \leq n^d.$$

*Remark 6.13* It is worthwhile to note that a weak converse of the previous lemma holds: Let $f$ satisfy the conclusion of Lemma 6.12. Then there is some $\varepsilon > 0$ such that $f(t) < t^{1-\varepsilon}$ for infinitely many $t$.

To see this write $f(t) = t^{g(t)}$ for some $g$. Then for $c = 1$ there are $d, n_0 \in \mathbb{N}$ such that $n^{d \cdot g(n^d)} < n^{d-1}$ for all $n \geq n_0$. Thus $g(t) < 1 - 1/d$, i.e. $f(t) \leq t^{1-1/d}$, for $t = n_0^d, (n_0+1)^d, (n_0+2)^d, \ldots$.

For a parameterized problem $(Q, \kappa)$, a constant $c \in \mathbb{N}$, and a function $f : \mathbb{N} \to \mathbb{R}_{\geq 0}$ consider the preparameterized problem

| $(Q, \kappa^c \times f)$ | |
|---|---|
| *Instance:* | $x \in \{0,1\}^*$. |
| *Parameter:* | $\kappa(x)^c \cdot f(\lvert x \rvert)$. |
| *Question:* | $x \in Q$? |

Theorem 6.5 follows from:

**Lemma 6.14** *Let $c \in \mathbb{N}$ and $f : \mathbb{N} \to \mathbb{R}_{\geq 0}$ be pseudo-linear. Let $(Q, \kappa)$ be a parameterized problem with a linear* OR *and with* NP*-hard $Q$. If* PH $\neq \Sigma_3^{\mathrm{P}}$*, then $(Q, \kappa^c \times f)$ has no linear kernelizations.*

We prove this lemma by generalizing Theorem 5.6.

**Definition 6.15** Let $Q, Q' \subseteq \{0,1\}^*$ be classical problems and let $f : \mathbb{N} \to \mathbb{R}_{\geq 0}$ be a function. An *$f$-distillation from $Q$ in $Q'$* is a polynomial time algorithm $\mathbb{D}$ that receives as inputs finite sequences $\bar{x} = (x_1, \ldots, x_t)$ with $x_i \in \{0,1\}^*$ for $i \in [t]$ and outputs a string $\mathbb{D}(\bar{x}) \in \{0,1\}^*$ such that

(1) $|\mathbb{D}(\bar{x})| = f(t) \cdot (\max_{i \in [t]} |x_i|)^{O(1)}$;
(2) $\mathbb{D}(\bar{x}) \in Q'$ if and only if for some $i \in [t]: \ x_i \in Q$.

We say that *$Q$ has an $f$-distillation* if there is an $f$-distillation from $Q$ in $Q'$ for some problem $Q'$.

**Lemma 6.16** *Let $f : \mathbb{N} \to \mathbb{R}_{\geq 0}$ be pseudo-linear. If* PH $\neq \Sigma_3^{\mathrm{P}}$*, then no* NP*-hard problem has $f$-distillations.*

*Proof* Let $f : \mathbb{N} \to \mathbb{R}_{\geq 0}$ be pseudo-linear and $Q \subseteq \{0,1\}^*$ be NP-hard. Assume that $\mathbb{D}$ is an $f$-distillation from $Q$ in some problem $Q'$. We choose a constant $c \in \mathbb{N}$ such that

$$|\mathbb{D}(\bar{x})| \leq f(t) \cdot \Big(\max_{i \in [t]} |x_i|\Big)^c \tag{9}$$

for all $t \in \mathbb{N}$ and all sequences $\bar{x}$ of $t$ instances of $Q$.

Let $\overline{Q} := \{0,1\}^* \setminus Q$ be the complement of $Q$ and similarly $\overline{Q'}$ the complement of $Q'$. Clearly $\overline{Q}$ is coNP-hard. We show that $\overline{Q} \in \mathrm{NP/poly}$ and hence, $\mathrm{coNP} \subseteq \mathrm{NP/poly}$. This yields our claim, as then $\mathrm{PH} = \Sigma_3^{\mathrm{P}}$ by a result of Yap [17, Theorem 2]. Note that for all $\bar{x} = (x_1, \ldots, x_t)$ we have

$$\mathbb{D}(\bar{x}) \in \overline{Q'} \iff \text{for all } i \in [t] : \ x_i \in \overline{Q}. \tag{10}$$

To prove $\overline{Q} \in \mathrm{NP/poly}$ it suffices to show that for sufficiently large $n \in \mathbb{N}$ there is a $t = n^{O(1)}$ and a set $S$ of strings with $\|S\| := \sum_{x \in S} |x| = n^{O(1)}$ such that for all $x \in \{0,1\}^n$

$$x \in \overline{Q} \iff \exists x_1, \ldots, x_t \in \{0,1\}^n : (x \in \{x_1, \ldots, x_t\} \text{ and } \mathbb{D}(x_1, \ldots, x_t) \in S). \tag{11}$$

In other words, $S$ can be viewed as a polynomial size advice string for instances of length $n$. As we will see, the elements of $S$ are strings in $\overline{Q'}$, more precisely, we will choose $\mathbb{D}$-values "with many preimages."

For every $m \in \mathbb{N}$, we have $|\{0,1\}^{\leq m}| \leq 2^{m+1}$, in particular,

$$|\{0,1\}^{\leq f(m)\cdot n^c}| \leq 2^{f(m)\cdot n^c + 1}. \tag{12}$$

As $f$ is pseudo-linear, by Lemma 6.12 there is a constant $d \in \mathbb{N}$ such that for all sufficiently large $n \in \mathbb{N}$

$$\frac{f(n^d)\cdot n^c + 1}{n^d} \leq 1. \tag{13}$$

For $n \geq 1$ we set

$$t := n^d.$$

Then (12) and (13) imply for $Y := \overline{Q'} \cap \{0,1\}^{\leq f(t)\cdot n^c}$ that

$$|Y|^{1/t} \leq 2. \tag{14}$$

Recall that $\overline{Q}_{=n} := \overline{Q} \cap \{0,1\}^n$. By (9) we can define a function $g : (\overline{Q}_{=n})^t \to Y$ by

$$g(\bar{x}) := \mathbb{D}(\bar{x}).$$

We construct the advice string $S$ inductively. First we let $X_0 := \overline{Q}_{=n}$. Choose $y_0 \in Y$ such that

$$g^{-1}(y_0) := \left\{\bar{x} \in X_0^t \mid g(\bar{x}) = y_0\right\}$$

contains at least $|X_0|^t/|Y|$ many tuples. Let $\mathit{string}(g^{-1}(y_0))$ be the set components of tuples in $g^{-1}(y_0)$, that is,

$$\mathit{string}(g^{-1}(y_0)) := \big\{x \in X_0 \mid \text{there exists some } (x_1, \ldots, x_t) \in g^{-1}(y_0) \text{ such that } x \in \{x_1, \ldots, x_t\}\big\}.$$

It follows that $g^{-1}(y_0) \subseteq (string(g^{-1}(y_0)))^t$ and hence

$$|string(g^{-1}(y_0))| \geq |g^{-1}(y_0)|^{1/t} \geq \left(\frac{|X_0|^t}{|Y|}\right)^{1/t} \geq \frac{|X_0|}{2},$$

the last inequality holding by (14). If $X_0 \neq string(g^{-1}(y_0))$, then let $X_1 := X_0 \setminus string(g^{-1}(y_0))$. Now, we view $g$ as a function of $X_1$ to $Y$ and, by the same argument as above, we choose $y_1 \in Y$ such that $|string(g^{-1}(y_1))| \geq |X_1|/2$. We iterate this process until we reach the first $\ell \in \mathbb{N}$ with $X_\ell = string(g^{-1}(y_\ell))$. We let

$$S := \{y_0, \ldots, y_\ell\}.$$

Then $S \subseteq Y \subseteq \overline{Q'}$ and $|S| = \ell \leq \log |X_0| \leq n$ and thus $\|S\| \leq n \cdot f(t) \cdot n^c \leq n^{d+1}$ (by (13)). Hence $\|S\|$ is polynomially bounded in $n$.

We show the equivalence (11). Let $x \in \{0,1\}^n$. If $x \in \overline{Q}$, by our construction of $S$, there is a tuple $\bar{x}$ containing $x$ as a component such that $g(\bar{x}) = \mathbb{D}(\bar{x}) \in S$.

Conversely, assume $x \notin \overline{Q}$. Then for every $\bar{x} := (x_1, \ldots, x_t)$ with $x_1, \ldots, x_t \in \{0,1\}^n$ and $x \in \{x_1, \ldots, x_t\}$, we have, by (10), that $\mathbb{D}(\bar{x}) \notin \overline{Q'}$ and hence $\mathbb{D}(\bar{x}) \notin S \subseteq \overline{Q'}$. □

*Proof of Lemma 6.14* Let $c \in \mathbb{N}$ and $f$ be pseudo-linear, say $f(t) = O(t^{1-\varepsilon})$. Assume that $(Q, \kappa)$ is a parameterized problem with a linear OR $\mathbb{O}$ and NP-hard $Q$. Assume $\Sigma_3^{\mathrm{P}} \neq \mathrm{PH}$. For the sake of contradiction assume that $(Q, \kappa^c \times f)$ has a linear kernelization $\mathbb{K}$. By Lemma 6.16 it suffices to show that $Q$ has an $f$-distillation $\mathbb{D}$.

We define $\mathbb{D}$ on finite sequences $\bar{x} = (x_1, \ldots, x_t)$ by

$$\mathbb{D}(\bar{x}) := \mathbb{K}(\mathbb{O}(\bar{x})).$$

It is clear that

$$\mathbb{D}(\bar{x}) \in Q \quad \Longleftrightarrow \quad \text{for some } i \in [t]\colon\ x_i \in Q.$$

Write $n := \max_{i \in [t]} |x_i|$. Then, because $\mathbb{K}$ is a linear kernelization for $(Q, \kappa^c \times f)$,

$$|\mathbb{D}(\bar{x})| = O(\kappa(\mathbb{O}(\bar{x}))^c \cdot f(|\mathbb{O}(\bar{x})|)) = n^{O(1)} \cdot |\mathbb{O}(\bar{x})|^{1-\varepsilon},$$

where the second equality follows from Definition 6.3(2). Now, by Definition 6.3(1) we know $|\mathbb{O}(\bar{x})| = t \cdot n^{O(1)}$. Hence $|\mathbb{D}(\bar{x})| = t^{1-\varepsilon} \cdot n^{O(1)}$ and therefore $\mathbb{D}$ is a $f$-distillation from $Q$ in itself. □

## 7 Lower Bounds for Problems with an OR for Instances with Constant Parameter

Recall that a *hole* in a graph is an induced cycle of length at least four. While the problems whether a graph contains a hole and whether it contains an even hole are solvable in polynomial time, it is not known whether there is such an algorithm deciding if a graph has an odd hole. Recently problems concerning holes have received

much attention as they are related to the Strong Perfect Graph Theorem [5] ("A graph is perfect if it contains neither an odd hole nor the complement of an odd hole"). We consider the parameterized problem (see [3])

| $p$-ODD-HOLE$_{\leq}$ | |
|---|---|
| *Instance:* | A graph $G$ and $k \in \mathbb{N}$. |
| *Parameter:* | $k$. |
| *Question:* | Does $G$ have a hole of *odd* length at most $k$? |

Let $(G_1, k_1), \ldots, (G_t, k_t)$ be instances of $p$-ODD-HOLE$_{\leq}$. If $k_1 = \cdots = k_t =: k$, then for the disjoint union $G$ of the $G_i$s we have $(G, k) \in p$-ODD-HOLE$_{\leq}$ if and only if $(G_i, k_i) \in p$-ODD-HOLE$_{\leq}$ for some $i \in [t]$. However, it is not clear how to define such an instance $(G, k)$ if $k_1, \ldots, k_t$ are distinct, more precisely, we do not know whether $p$-ODD-HOLE$_{\leq}$ has an OR. The following concept is tailored for such situations.

**Definition 7.1** Let $(Q, \kappa)$ be a parameterized problem and let $\lambda$ be a further parameterization. An *OR for $\lambda$-constant instances of* $(Q, \kappa)$ is a polynomial time algorithm $\mathbb{O}$ that for every finite tuple $\bar{x} = (x_1, \ldots, x_t)$ of instances of $Q$ with $\lambda(x_1) = \cdots = \lambda(x_t)$ outputs an instance $\mathbb{O}(\bar{x})$ of $Q$ such that

(1) $\kappa(\mathbb{O}(\bar{x})) = (\max_{i \in [t]} |x_i|)^{O(1)}$;
(2) $\mathbb{O}(\bar{x}) \in Q$ if and only if for some $i \in [t]$: $x_i \in Q$.

*Examples 7.2* The instances of the following problems are pairs $(G, k)$, where $G$ is a graph and $k \in \mathbb{N}$. We let $\lambda$ always be the function with $\lambda(G, k) := k$. In all examples we get the claimed OR for $\lambda$-constant instances by setting $\mathbb{O}((G_1, k), \ldots, (G_t, k)) := (G, k)$, where the graph $G$ is the disjoint union of the $G_i$s. In all cases we do not know whether the corresponding problem has an OR.

(a) The problem $p$-ODD-HOLE$_{\leq}$ has an OR for $\lambda$-constant instances.
(b) The problems *uni*-CHORDLESS-PATH and *uni*-CHORDLESS-CYCLE have an OR for $\lambda$-constant instances. Here, for example,

| *uni*-CHORDLESS-CYCLE | |
|---|---|
| *Instance:* | A graph $G = (V, E)$ and $k \in \mathbb{N}$. |
| *Parameter:* | $k \cdot \log \lvert V \rvert$. |
| *Question:* | Does $G$ have a chordless cycle of length $k$? |

Note that in the last example $\lambda(G, k) = k$ is not the parameter of $(G, k)$ as instance of *uni*-CHORDLESS-CYCLE.

For problems with an OR for constant instances we get a slightly weaker result than that in Theorem 6.5 for problems with a linear OR. To state the result we first define:

**Definition 7.3** Let $(Q, \kappa)$ be a parameterized problem. A *subexponential self-reduction of* $(Q, \kappa)$ is a polynomial reduction from $Q$ to itself that assigns to every

instance $x$ of $Q$ an instance $y$ with

$$|y| = \kappa(x)^{O(1)} \cdot |x|^{o(1)}.$$

Clearly if $(Q, \kappa)$ has a subexponential self-reduction, then it has an $\varepsilon$ self-reduction for every $\varepsilon > 0$.

**Theorem 7.4** *Let $(Q, \kappa)$ be a parameterized problem with* NP*-hard $Q$. Furthermore assume that $(Q, \kappa)$ has an* OR *for $\lambda$-constant instances, where $\lambda$ is a further parameterization. If* PH $\neq \Sigma_3^{\mathrm{P}}$*, then $(Q, \kappa)$ has* no *subexponential self-reductions.*

This improves the corresponding result of [1] in the following respects:

- it assumes $Q$ to be only NP-hard instead of NP-complete;
- it assumes a weaker notion of OR (the OR used in [1] is ours for $\lambda = \kappa$);
- it excludes subexponential self-reductions instead of polynomial kernelizations.

In particular, we can apply Theorem 7.4 to the problems in Examples 7.2 (b). It is not known whether $p$-HOLE$_{\leq}$ is in FPT. If not, then it would not have polynomial kernelizations. At the moment we cannot apply Theorem 7.4 to rule out polynomial kernelizations, as to the best of knowledge it is not known whether the underlying problem is NP-hard. To get a further application of the theorem we need the following lemma whose proof is simple and similar to that of Lemma 5.4.

**Lemma 7.5** *Let $(Q, \kappa)$ and $(Q', \kappa')$ be parameterized problems.with*

$$(Q, \kappa) \leq^p (Q', \kappa') \quad \text{and} \quad Q' \leq^p Q.$$

*If $(Q', \kappa')$ has a subexponential self-reduction, then $(Q, \kappa)$ has a subexponential self-reduction.*

*Example 7.6* If PH $\neq \Sigma_3^{\mathrm{P}}$, then $p$-PATH(PLAN-CONN) has no subexponential self-reductions.

*Proof* We know that the problem $p$-POINTED-PATH(PLAN-CONN) has an OR and hence no subexponential self-reduction. In the proof of Proposition 5.13 we showed that there is a polynomial reduction from the problem $p$-POINTED-PATH(PLAN-CONN) to $p$-PATH(PLAN-CONN). Hence, the claim follows from the previous lemma. □

### 7.1 Proof of Theorem 7.4

Recall the reparameterization $(Q, \kappa^c \times f)$ of $(Q, \kappa)$ for $c \in \mathbb{N}$ and $f : \mathbb{N} \to \mathbb{R}_{\geq 0}$. Clearly $(Q, \kappa^c \times f)$ has a polynomial kernelization if and only if $(Q, \kappa \times f)$, the problem for $c = 1$, has one.

For the purposes of the proof of Theorem 7.4 we call a function $f : \mathbb{N} \to \mathbb{R}_{\geq 0}$ *good* if $f(t) = t^{o(1)}$ (that is, if we can write $f(t) = t^{1/h(t)}$ for some function $h : \mathbb{N} \to \mathbb{R}_{\geq 0}$ with $\lim_{t \to \infty} h(t) = \infty$).

The statement of this theorem can be equivalently formulated as:

**Lemma 7.7** *Let $(Q,\kappa)$ be a parameterized problem with* NP*-hard $Q$. Furthermore assume that $(Q,\kappa)$ has an* OR *for $\lambda$-constant instances, where $\lambda$ is a further parameterization. If* $\mathrm{PH} \neq \Sigma_3^{\mathrm{P}}$*, then, for every good $f:\mathbb{N}\to\mathbb{R}_{\geq 0}$ the problem $(Q,\kappa\times f)$ has no polynomial kernelizations.*

*Proof* Assume $\mathrm{PH} \neq \Sigma_3^{\mathrm{P}}$. Furthermore, we choose for $(Q,\kappa)$ an OR $\mathbb{O}$ for $\lambda$-constant instances.

Let $f:\mathbb{N}\to\mathbb{R}_{\geq 0}$ be good. One easily sees that there is a good increasing function $f':\mathbb{N}\to\mathbb{R}_{\geq 0}$ of the form

$$f'(t) = 2^{\log t/\iota(\log t)} \tag{15}$$

with a nondecreasing and unbounded function $\iota:\mathbb{N}\to\mathbb{R}_{\geq 0}$ such that $f(t)\leq f'(t)$ for all (sufficiently large) $t$.

For the sake of contradiction assume also that $(Q,\kappa\times f)$ has a polynomial kernelization. Of course, then $(Q,\kappa\times f')$ has a polynomial kernelization $\mathbb{K}$. Now let $Q'$ be the "OR-closure" of $Q$, that is

$$Q' := \big\{(x_1,\ldots,x_m) \;\big|\; m\geq 1 \text{ and } x_i\in Q \text{ for some } i\in[m]\big\}.$$

Let $x_1,\ldots,x_t$ be instances of $Q$. We let $n := \max_{i\in[t]}|x_i|$ and $\ell := \max_{i\in[t]}\lambda(x_i)$. Then $\ell = n^{O(1)}$. For $j\leq\ell$ let

$$y_j := \mathbb{K}(\mathbb{O}(\bar{x}_j)),$$

where $\bar{x}_j$ stands for the subsequence of $x_1,\ldots,x_t$ consisting of the instances with $\lambda$-value $j$.

We show that for some good function $f_1$ and all $j\leq\ell$

$$|y_j| = f_1(t)\cdot n^{O(1)}. \tag{16}$$

In fact, as $\mathbb{K}$ is a polynomial kernelization of $(Q,\kappa\times f')$, we know

$$|y_j| = |\mathbb{K}(\mathbb{O}(\bar{x}_j))| = (\kappa(\mathbb{O}(\bar{x}_j))\cdot f'(|\mathbb{O}(\bar{x}_j)|))^{O(1)} = n^{O(1)}\cdot f'(|\mathbb{O}(\bar{x})|)^{O(1)},$$

where the last equality holds by Definition 7.1 (1). We show that $f'(|\mathbb{O}(\bar{x})|) = f'(t)^d\cdot n^d$ for some $d\in\mathbb{N}$. Then we get (16) for $f_1(t) := f'(t)^d$. As $f'$ is good, so is $f_1$.

As $\mathbb{O}$ is polynomial time computable, we know $|\mathbb{O}(\bar{x}_j)|\leq t^c\cdot n^c$ for some constant $c\in\mathbb{N}$. Since $f'$ is increasing, it is enough to show

$$f'(t^c\cdot n^c)\leq (f'(t)\cdot n)^{2c}.$$

By (15)

$$f'(t^c\cdot n^c) = 2^{\frac{c\cdot\log t + c\cdot\log n}{\iota(c\cdot\log t + c\cdot\log n)}}.$$

We distinguish two cases.

– If $t \geq n$, then, as $\iota$ is nondecreasing, we get

$$f'(t^c \cdot n^c) \leq 2^{\frac{2c \cdot \log t}{\iota(\log t)}} = f'(t)^{2c}.$$

– If $t < n$, then

$$f'(t^c \cdot n^c) \leq 2^{2c \cdot \log n} = n^{2c}.$$

This finishes the proof of (16).

Now we claim that

$$\mathbb{D}(x_1, \ldots, x_t) := (y_1, \ldots, y_\ell)$$

defines an $f_1$-distillation from $Q$ to $Q'$ (cf. Definition 6.15). As $f_1$ is good and hence, pseudo-linear, this contradicts Lemma 6.16. Obviously the condition (2) in Definition 6.15 is satisfied. To see (1), we observe that

$$\begin{aligned} |(y_1, \ldots, y_\ell)| &\leq \ell \cdot f_1(t) \cdot n^{O(1)} \quad \text{(by (16))}, \\ &= f_1(t) \cdot n^{O(1)} \quad \text{(by } \ell = n^{O(1)}\text{)}. \end{aligned}$$

Altogether $\mathbb{D}$ is an $f_1$-distillation from $Q$ and $Q'$. □

## 8 Concluding Remarks

### 8.1 Comparing the Different Notions of OR

From Theorem 5.6, Corollary 5.12, and Theorem 6.5 we know:

**Proposition 8.1** *Assume that* PH $\neq \Sigma_3^{\mathrm{P}}$. *Then*:

(1) *No* NP*-complete problem has a self-distillation.*
(2) *No parameterized problem* $(Q, \kappa)$ *with polynomial kernelization and with* NP*-complete* $Q$ *has an* OR.
(3) *No parameterized problem* $(Q, \kappa)$ *with polynomial kernelization and with* NP*-complete* $Q$ *has a linear* OR.

We do not know whether one of the three conclusions holds under weaker assumptions, say, under P $\neq$ NP. In this context it might be interesting to be aware of:

**Proposition 8.2** *The conclusions* (1), (2), *and* (3) *of Proposition* 8.1 *are mutually equivalent.*

*Proof* The implication (2) $\Rightarrow$ (3) is trivial. For (3) $\Rightarrow$ (1) assume, by contradiction, that $Q$ is NP-complete and has a self-distillation $\mathbb{D}$. Define $\kappa(x) := |x|$. Then $x \mapsto x$ is a polynomial kernelization of $(Q, \kappa)$ and $\mathbb{D}$ is a linear OR of $(Q, \kappa)$, the desired contradiction to (3).

For the implication (1) $\Rightarrow$ (2) assume that $(Q,\kappa)$ with NP-complete $Q$ has a polynomial kernelization $\mathbb{K}$ and an OR $\mathbb{O}$. Then $\mathbb{K}\circ\mathbb{O}$ is a self-distillation, as

$$\mathbb{K}(\mathbb{O}(\bar{x})) = \kappa(\mathbb{O}(\bar{x}))^{O(1)} = \left(\max_i |x_i|\right)^{O(1)}. \qquad \square$$

The next result shows in particular that every parameterized problem $(Q,\kappa)$ with polynomial kernelization and NP-complete $Q$ already has no OR if it has no linear OR. For example, the parameterized vertex cover problem $p$-VC has no linear OR if and only if it has no OR.

**Proposition 8.3** *Assume that $(Q,\kappa)$ and $(Q',\kappa')$ are parameterized problems with* NP*-complete $Q$ and $Q'$ and that $(Q',\kappa')$ has a polynomial kernelization. If $(Q,\kappa)$ has no linear OR, then $(Q',\kappa')$ has no* OR.

*Proof* Let $R : Q \leq^p Q'$ and $S : Q' \leq^p Q$ be polynomial reductions and $\mathbb{K}$ a polynomial kernelization of $(Q',\kappa')$ and assume that $\mathbb{O}$ is an OR of $Q'$, then

$$x_1,\ldots,x_t \mapsto S(\mathbb{K}(\mathbb{O}(R(x_1),\ldots,R(x_t))))$$

is a linear OR of $(Q,\kappa)$. $\square$

## 8.2 Comparing the Different Notions of Self-reduction

Clearly, every parameterized problem with a polynomial kernelization has a subexponential self-reduction, and every parameterized problem with a subexponential self-reduction has an $\varepsilon$ self-reduction for every $\varepsilon > 0$. The following two propositions establish that these inclusions are proper.

**Proposition 8.4** *There exists a fixed-parameter tractable parameterized problem that has $\varepsilon$ self-reductions for all $\varepsilon > 0$ but does not have subexponential self-reductions.*

**Proposition 8.5** *There exists a fixed-parameter tractable problem that has subexponential self-reductions but does not have polynomial kernelizations.*

*Proof of Proposition 8.4* Let $Q \subseteq \mathbb{N}$ be a classical problem such that every $x \in Q$ is a power of 2 with an odd exponent and is written in unary. We define the parameterized problem $p$-$Q$ by

| $p$-$Q$ | |
|---|---|
| *Instance:* | $m,k \in \mathbb{N}$ in unary with $\log k \geq \frac{\log m}{\log\log m}$. |
| *Parameter:* | $k$. |
| *Question:* | Is $(\log m)\cdot(\log k) \in Q$? |

It suffices to show

(1) If $Q$ is decidable, then $p$-$Q$ is fixed-parameter tractable.
(2) For every $\varepsilon > 0$ the problem $p$-$Q$ has an $\varepsilon$ self-reduction.

(3) If $Q \notin \mathrm{E}$, then $p\text{-}Q$ has no subexponential reductions.

(1) As for yes-instances $(m,k)$ of $p\text{-}Q$, we have $\log k \geq \log m / \log\log m$, the problem $p\text{-}Q$ has a kernelization and hence is fixed-parameter tractable by Proposition 3.2.

(2) Let $t \in \mathbb{N}$. We show that there is an $1/d$ self-reduction of $p\text{-}Q$ for $d := 2^t$.

Let $(m,k)$ be an instance of $p\text{-}Q$. We can assume that $m = 2^{2^u}$ and $k = 2^{2^v}$ (otherwise, $(m,k)$ is a no-instance of $p\text{-}Q$).

We set

$$m' := 2^{2^{u-t}} \ (= (2^{2^u})^{1/d}) \quad \text{and} \quad k' := 2^{2^{v+t}} \ (= (2^{2^v})^d).$$

Clearly, $(m,k) \in p\text{-}Q$ if and only if $(m',k') \in p\text{-}Q$. Moreover, $|m'| = |m|^{1/d}$ and $|k'| = |k|^d$ and hence, $|(m',k')| = O(k^d \cdot m^{1/d})$. Altogether, $(m,k) \mapsto (m',k')$ is an $1/d$ self-reduction of $p\text{-}Q$.

(3) We assume that $p\text{-}Q$ has a subexponential self-reduction $(m,k) \mapsto (m',k')$. Then

$$|(m',k')| = k^c \cdot (m+k)^{o(1)} = k^c \cdot m^{o(1)}$$

for some $c \in \mathbb{N}$. We can assume that $c$ is a power of 2. We show that $Q \in \mathrm{E}$.

Let $x$ be an instance of $Q$ with $x \geq d \geq 2^{4c^2}$, where $d \in \mathbb{N}$ will be fixed later. We assume that $x$ is an odd power of 2 (otherwise, $x \notin Q$). We set

$$u := \sqrt{2c^2 \cdot x} \quad \text{and} \quad v := \frac{u}{2c^2}.$$

Then, $u$ and $v$ are powers of 2 (note that $v = \sqrt{x/2c^2}$) and $u \cdot v = x$. Moreover, $v \geq u/\log u$ by our assumption $x \geq 2^{4c^2}$. Hence, $(2^u, 2^v) \in p\text{-}Q$ if and only if $x \in Q$. We apply the subexponential self-reduction to $(2^u, 2^v)$ obtaining an equivalent instance $(m',k')$ of $p\text{-}Q$ with

$$m', k' \leq 2^{v \cdot c} \cdot (2^u)^{o(1)} = 2^{v \cdot c + u \cdot o(1)}.$$

If $d$ has been chosen big enough, we have

$$\begin{aligned} x' := (\log m') \cdot (\log k') &\leq (v \cdot c)^2 + v \cdot u \cdot o(1) + u^2 \cdot o(1) \\ &\leq (u/2c)^2 + u^2 \cdot o(1) < u^2/2c^2 = uv = x. \end{aligned}$$

Thus, $x' < x$. If $k' < m'/\log m'$, then $(m',k') \notin p\text{-}Q$ and hence, $x \notin Q$. Otherwise, $(x' \in Q \iff x \in Q)$. We continue this way and obtain equivalent instances $x'', x''', \ldots$ of $Q$ till we get an instance $\leq d$, which is decided directly. Altogether, we have a single exponential decision procedure for $Q$. □

*Proof of Proposition 8.5* Let $Q \subseteq \mathbb{N}$ be a classical problem such that every $x \in Q$ is represented in unary and has the form

$$x = 2^{2^t} \tag{17}$$

for some $t \in \mathbb{N}$. We define the parameterized problem $p\text{-EXP}(Q)$ by

| $p$-EXP($Q$) | |
|---|---|
| *Instance:* | $m, k \in \mathbb{N}$ in unary with $k \geq \log\log m$. |
| *Parameter:* | $k$. |
| *Question:* | Is $m^k \in Q$? |

It is sufficient to show

(1) If $Q$ is decidable, then $p$-EXP($Q$) is fixed-parameter tractable.
(2) The problem $p$-EXP($Q$) has a subexponential self-reduction.
(3) If $Q \notin$ PTIME, then $p$-EXP($Q$) has no polynomial kernelizations.

(1) As for yes-instances $(m, k)$ of $p$-EXP($Q$), we have $k \geq \log\log m$, the problem $p$-EXP($Q$) has a kernelization and hence is fixed-parameter tractable by Proposition 3.2.

(2) Let $(m, k)$ be an instance of $p$-EXP($Q$). By (17), we can assume that $m = 2^{2^t}$ for some $t \in \mathbb{N}$ (otherwise, $(m, k)$ is a no-instance of $p$-EXP($Q$)). Then

$$(m, k) \in p\text{-EXP}(Q) \iff 2^{k \cdot 2^t} \in Q \iff (2, k \cdot 2^t) \in p\text{-EXP}(Q).$$

Therefore the mapping $(m, k) \mapsto (2, k \cdot \log m)$ is the desired reduction.

(3) We assume that $\mathbb{K}$ is a polynomial kernelization of $p$-EXP($Q$) and show that $Q \in$ PTIME.

Let $x = 2^{2^t}$ be an instance of $Q$. We let $t'$ be the minimum power of 2 with $t' \geq t$. Thus, $2t \geq t' \geq t$. Clearly

$$x \in Q \iff (2^{2^t/t'}, t') \in p\text{-EXP}(Q).$$

Furthermore we set $(m, k) := \mathbb{K}(2^{2^t/t'}, t')$. We know that

$$|(m, k)| = t'^{O(1)} = t^{O(1)}$$

and that $x \in Q$ if and only if $m^k \in Q$. As

$$m^k = t^{O(t^{O(1)})} = 2^{t^{O(1)}}$$

we see that this is strictly smaller than $x$ if $x$ is sufficiently large. □

### 8.3 Comparing $\varepsilon$ Self-reductions and Kernelizations

We showed (Theorem 6.5) that a refinement of the method used in [1, 7] to exclude polynomial kernelizations, actually works to exclude $\varepsilon$ self-reductions. Although this gives some interest to the concept of $\varepsilon$ self-reduction, the question remains how natural this concept is. In this last section we want to present results clarifying how close the concepts of polynomial kernelization and of $\varepsilon$ self-reductions are.

Note that $\varepsilon$ self-reductions are allowed to increase the parameter arbitrarily. By a straightforward argument we shall see in Proposition 8.6 that a parameterized problem has an $\varepsilon$ self-reduction which does not increase the parameter if and only if it has

a parameter non-increasing polynomial kernelization. We then look what happens if we allow some 'moderate' increase in the parameter. Different renderings of what 'moderate' means, allow to iterate $\varepsilon$ self-reductions to yield polynomial or subexponential kernelizations.

Given $\ell \in \mathbb{N}$ and a function $f$ whose range is included in its domain, let $f^\ell$ denote the function given by $f^\ell(a) := \underbrace{f \circ f \circ \cdots \circ f}_{\ell \text{ times}}(a)$.

**Proposition 8.6** *Let $0 < \varepsilon < 1$ and let $(Q, \kappa)$ be a parameterized problem. Then $(Q, \kappa)$ has a parameter non-increasing polynomial kernelization if and only if it has parameter non-increasing $\varepsilon$ self-reduction $R$.*

*Sketch of proof* The forward direction is trivial. Conversely, let $R$ be an $\varepsilon$ self-reduction of $(Q, \kappa)$ such that $\kappa \circ R \leq \kappa$. Choose $c \in \mathbb{N}$ such that $|R(x)| \leq \kappa(x)^c \cdot |x|^{(1-\varepsilon)}$ for all $x \in \Sigma^*$. A straightforward induction shows

$$|R^\ell(x)| \leq \kappa(x)^{c \cdot \sum_{0 \leq i \leq \ell-1} (1-\varepsilon)^i} \cdot |x|^{(1-\varepsilon)^\ell}, \tag{18}$$

for all $\ell \geq 1$ and $x \in \Sigma^*$. Furthermore, a simple computation shows that for $m := (\log \log |x|)/\varepsilon$ we get

$$|x|^{(1-\varepsilon)^m} \leq 2. \tag{19}$$

Using $\sum_{i=0}^{\infty} (1-\varepsilon)^i = 1/\varepsilon$, the inequalities (18) and (19) imply

$$|R^m(x)| \leq \kappa(x)^{c/\varepsilon} \cdot 2 \leq \kappa(x)^{O(1)}.$$

By (18) we get $|R^\ell(x)| \leq \kappa(x)^{c/\varepsilon} \cdot |x| \leq |x|^{O(1)}$ for all $\ell \geq 1$ (recall $\kappa(x) \leq |x|^{O(1)}$), and hence $R^m$ can be computed in polynomial time. Thus $R^m$ is a parameter non-increasing polynomial kernelization of $(Q, \kappa)$. □

**Definition 8.7** A function $f : \mathbb{N} \to N$ is *moderate* (*strongly moderate*) if and only if it is nondecreasing and $f^\ell(k) \leq k^{O(\ell)}$ (respectively $f^\ell(k) \leq k^{O(1)}$) for all $k, \ell \in \mathbb{N}$ with $k/\ell$ sufficiently large.

E.g. linear functions are moderate. On the other hand, a "slightly polynomial" function $k \mapsto \lfloor k^{1+\varepsilon} \rfloor$ for a constant $\varepsilon > 0$ is not moderate. Clearly, the identity is strongly moderate, but $k \mapsto \lfloor k \cdot (1+\varepsilon) \rfloor$ for $\varepsilon > 0$ is not. We give further examples.

*Examples 8.8* (a) The function given by $f(k) := \lfloor k \cdot \log k \rfloor$ is moderate.

*Proof* It is enough to show $f^\ell(k) \leq k^\ell \cdot (\log k)^\ell$ for all $k, \ell \in \mathbb{N}$ with $k/\ell \geq 2$. Inductively

$$\begin{aligned} f^{\ell+1}(k) &\leq f^\ell(k) \cdot \log f^\ell(k) \leq k^\ell (\log k)^\ell \cdot \log(k^\ell (\log k)^\ell) \\ &= k^\ell (\log k)^\ell \cdot (\ell \log k + \ell \log \log k) \end{aligned} \tag{20}$$

Now, $\log k + \log\log k \leq 2\log k$, so $k/2 \cdot (\log k + \log\log k) \leq k \log k$. But if $k/\ell \geq 2$, i.e. $\ell \leq k/2$, we get $(\ell \log k + \ell \log\log k) \leq k \log k$. Hence by (20) we get $f^{\ell+1}(k) \leq k^{\ell+1}(\log k)^{\ell+1}$ as we want. □

(b) The function given by $f(k) := \lfloor k \cdot \sqrt[k]{k} \rfloor$ is strongly moderate.

*Proof* An easy induction shows $f^{\ell}(k) \leq k^{(1+1/k)^{\ell}}$ for all $\ell, k \in \mathbb{N}$. If $k/\ell \geq 1$, i.e. $k \geq \ell$, this is at most $k^{(1+1/k)^k} = k^{O(1)}$. □

**Proposition 8.9** *Let* $0 < \varepsilon < 1$ *and let* $(Q, \kappa)$ *be a parameterized problem in* EXPT *with an* $\varepsilon$ *self-reduction* $R$. *Then*

(1) *if* $\kappa \circ R \leq f \circ \kappa$ *for some strongly moderate* $f$, *then* $(Q, \kappa)$ *has a polynomial kernelization*;
(2) *if* $\kappa \circ R \leq f \circ \kappa$ *for some moderate* $f$, *then* $(Q, \kappa)$ *has a subexponential kernelization*; *more specifically, it has a* $k^{O(\log k)}$*-kernelization*.

We omit the proof as it consists mainly in tedious computations along the line of argument for Proposition 8.6. For details see the third author's PhD thesis [15].

**Acknowledgements** The third author wants to thank Mike Fellows, Danny Hermelin, Mihai Prunescu and Frances Rosamond for helpful discussions.

## References

1. Bodlaender, H.L., Downey, R.G., Fellows, M.R., Hermelin, D.: On problems without polynomial kernels. In: Proceedings of the 35th International Colloquium on Automata, Languages and Programming (ICALP'08, Track A). Lecture Notes in Computer Science, vol. 5125, pp. 563–574. Springer, Berlin (2008)
2. Chang, R., Kadin, Y.: On computing boolean connectives of characteristic functions. Math. Syst. Theory **28**, 173–198 (1995)
3. Chen, Y., Flum, J.: On parameterized path and chordless path problems. In: Proceedings of the 22nd IEEE Conference on Computational Complexity (CCC'07), pp. 250–263 (2007)
4. Chen, Y., Flum, J.: Subexponential time and fixed-parameter tractability: exploiting the miniaturization mapping. J. Log. Comput. **19**(1), 89–122 (2009)
5. Chudnovsky, M., Robertson, N., Seymour, P., Thomas, R.: The strong perfect graph theorem. Ann. Math. **164**, 51–229 (2006)
6. Flum, J., Grohe, M.: Parameterized Complexity Theory. Springer, Berlin (2006)
7. Fortnow, L., Santhanam: Infeasibility of instance compression and succinct PCPs for NP. In: Proceedings of the 40th ACM Symposium on the Theory of Computing (STOC'08), pp. 133–142. ACM, New York (2008). Full version available at: http://lance.fortnow.com/papers/. Accessed 23 May 2010
8. Frick, M., Grohe, M.: The complexity of first-order and monadic second-order logic revisited. Ann. Pure Appl. Log. **130**, 3–31 (2004)
9. Grandjean, E., Kleine-Büning, H.: SAT-problems and reductions with respect to the number of variables. J. Log. Comput. **7**(4), 457–471 (1997)
10. Grohe, M.: Private communication (2008)
11. Guo, J., Niedermeier, R.: Invitation to data reduction and problem kernelization. ACM SIGACT News **38**(1) (2007)
12. Harnik, D., Naor, M.: On the compressibility of NP instances and cryptographic applications. In: Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06), pp. 719–728 (2006). Full version appears as TR06-022 in ECCC Reports 2006, available at http://eccc.hpi-web.de/year/2006/. Accessed 23 May 2010

13. Impagliazzo, R., Paturi, R., Zane, F.: Which problems have strongly exponential complexity? J. Comput. Syst. Sci. **63**, 512–530 (2001)
14. Johnson, D.: Announcements, updates, and greatest hits. J. Algorithms **8**(3), 438–448 (1987)
15. Müller, M.: Parameterized randomization. PhD thesis, Albert-Ludwigs-Universität Freiburg i.Br. URN: urn:nbn:de:bsz:25-opus-64017. Available at http://www.freidok.uni-freiburg.de/volltexte/6401/ (2009). Accessed 23 May 2010
16. Niedermeier, R.: Invitation to Fixed-Parameter Algorithms. Oxford University Press, London (2006)
17. Yap, C.K.: Some consequences of non-uniform conditions on uniform classes. Theor. Comput. Sci. **26**, 287–300 (1983)