PARAMETERIZED RANDOM COMPLEXITY

JUAN ANDRÉS MONTOYA AND MORITZ MÜLLER

1. Introduction

1.1. **Parameterized complexity.** While classically running times or other resources of algorithms are measured by a function in the length of the input only, in parameterized complexity a secondary measurement is introduced: each problem instance comes along with an associated natural number – its parameter. The parameter of an instance is intended to encode some knowledge we have about 'typical' instances or instances we are interested in and that we want to exploit algorithmically. Intuitively, the associated parameter is small compared to the instance length. To allow full exploitation of this knowledge the notion of tractability is adjusted accordingly: an algorithm that decides instances of length n with parameter k in time $f(k) \cdot n^{O(1)}$ is called fixed-parameter tractable (fpt). Here, f is an arbitrary computable function.

The theory of parameterized intractability is centered around the W-hierarchy

$$\mathrm{W}[1] \subseteq \mathrm{W}[2] \subseteq \cdots \subseteq \mathrm{W}[\mathrm{SAT}] \subseteq \mathrm{W}[\mathrm{P}].$$

The classes W[1] and W[P] are two natural parameterized analogues of NP in that they can be characterized via RAMs using certain restricted existential nondeterminism [22]. A 'W[P]-machine' is a RAM that on instances of size n with parameter k makes 'few' ($f(k)$ many) existential guesses of 'large' (size $f(k) \cdot n^{O(1)}$) natural numbers. A 'W[1]-machine' additionally has to make these guesses in the end of the computation (during the last $f(k)$ many steps).

1.2. **Randomness in the parameterized setting.** First, there are methods to design fpt algorithms by first designing a certain type of randomized algorithm which is then fully derandomized. The most famous example is "Color-Coding" [3] which is based on a certain use of random colourings that can be derandomized using perfect hash families. See [37, Section 13.3] for an exposition of the method, a sample of examples is [34, 49, 47, 16]. Other such methods are "Random Separation" [10] and "Randomized Disposal of Unknowns" [17], both employing derandomization via universal sets [57, 58].

Second, some randomized fpt algorithms have been designed. Early examples as [33] can be found in the (short) list in [27, Appendix A.3]. R.G.Downey et al. [29] constructed randomized reductions providing parameterized analogues of the Valiant-Vazirani Lemma (cf. Section 8). V.Arvind and V.Raman [5] designed randomized approximate counting algorithms (cf. Section 7).

Third, some works use hypotheses on randomized parameterized intractability in the one or the other sense. An example is M.Alekhnovich and A.A.Razborov's [2] proof that short resolution refutations are hard to find. This result has been derandomized by K.Eickmeyer et al. [32]. As another example, Y.Chen and J.Flum [19] disprove #W[1]-hardness of certain counting problems.

1.3. **Parameterized random complexity.** Except [19] all work cited above is concerned with randomized algorithms running in fpt time. A genuinely parameterized view on random complexity would instead mean to measure random complexity using parameterizations. How?

We look at the classical world: there a randomized polynomial time algorithm is explained to be a binary 'NP-machine' where nondeterministic steps are interpreted as coin tosses. Using the machine characterizations of parameterized NP-analogues, this procedure can be mimicked in the parameterized world as follows.

Interpret a nondeterministic step as rolling a die instead of tossing a coin. Taking W[P] as analogue of NP, a W[P]-*randomized* algorithm is then an algorithm rolling, on instances of length n with parameter k, only very 'few' ($f(k)$ many) but possibly 'large' ($f(k) \cdot n^{O(1)}$-sided) dice. In terms of Turing machines this amounts to a random complexity of $f(k) \cdot \log n$ many random bits, and in some sense this can be vaguely called an 'arbitrarily low but nontrivial' amount of randomness (cf. Section 5). Taking W[1] as analogue of NP amounts to additionally restricting the *use* of randomness: dice rolls are allowed only at the end of the computation.

This way, parameterized complexity theory can provide a genuine view on randomization by reusing the parameterized restrictions on nondeterminism as restrictions on randomness. This possibility has already been shown by R.G.Downey et al. in [29]. They set up a framework for parameterized randomization based on a parameterized analogue of the classical BP-operator and a characterization of parameterized classes by bi-indexed circuit families. Within that framework W[P]-randomization has been systematically studied in [51, 52].

Meanwhile, the machine characterization theorems are available and, as sketched above, give rise to somewhat handier definitions. Here we develop the corresponding theory. Amongst other things this leads to new proofs of some of the results in [29, 51, 52] (more precisely, the analogous statements in the current framework). These proofs significantly simplify and strengthen the results concerned.

1.4. **This work.** Section 3 defines W[1]- and W[P]-randomization, the corresponding parameterized analogues of BPP, namely BPFPT[1] and BPFPT, and contains some basic observations concerning the robustness of these definitions.

Strong probability amplification for W[P]-randomization has been established in [51] within the framework given by the parameterized BP-operator. In our setting we obtain a new and simpler proof of this fact in Section 4 (Theorem 4.1). We therefore suggest that BPFPT is a fairly robust class. However, is it useful?

Algorithmically, we give some W[P]-randomized algorithms: we use a parameterized version of polynomial identity testing as a toy example throughout the text; Sections 7 and 8 construct some more involved W[P]-randomized algorithms.

Theoretically, one may ask what W[P]-randomization can tell us about the classical derandomization question. In Section 5 we shall see that "black-box" derandomization of BPFPT is equivalent to, vaguely said, 'arbitrarily weak but nontrivial' classical derandomization.

We then ask for parameterized analogues of classical results, questions that often call for new arguments. We give such analogues for the Sipser-Gács Theorem [62] in Section 6 (Theorem 6.3), a theorem of Stockmeyer [63] in Section 7 (Theorem 7.3) and the Valiant-Vazirani Lemma [66] in Section 8 (Theorem 8.6).

An analogue of the Sipser-Gács Theorem has already been established in [52] for W[P]-randomization (Theorem 6.3 (1)). Section 6 presents a new and simple argument in our framework that extends to W[1]-randomization (Theorem 6.3 (2)). The paper [52] also contains an analogue of Stockmeyer's theorem for W[P]. Section 7 gives a more general statement (Theorem 7.5) that implies such analogues for all classes of the W-hierarchy and more (Theorem 7.3). Our parameterized analogue of the Valiant-Vazirani Lemma (Theorem 8.6) strengthens the one from [29] and generalizes it to various classes outside the W-hierarchy. We obtain it as a corollary to a result on model-checking problems (Theorem 8.3) that may be of independent interest.

For more information the reader may consult the introductions preluding each section. These state the main results of the section and detail motivation, background and related work.

1.5. **References.**

This paper reports parts of the authors PhD Theses [50, 56]. Some results appeared in the extended abstracts [53, 54, 55] and, as already mentioned, in the articles [51, 52]. To be precise, the latter contain statements (formulated in the framework there) analoguous to Theorems 4.1 and 6.3 (1) and 7.3 (3).

1.6. **Acknowledgements.** We thank again Jörg Flum, the advisor of our PhD Theses. The second author thanks the John Templeton Foundation for its support under Grant #13152, *The Myriad Aspects of Infinity*, and the FWF (Austrian Research Fund) for its support through Grant P 23989 - N13

2. Preliminaries

Our mode of speech and notation closely follow [37].

2.1. **Parameterized problems.**

Fix a finite alphabet Σ containing $0, 1$. A *parameterized (decision) problem* is a pair (Q, κ) of a classical problem $Q \subseteq \Sigma^*$ and a polynomial time computable *parameterization* $\kappa : \Sigma^* \to \mathbb{N}$ mapping any string x to its *parameter* $\kappa(x)$. We often identify $Q \subseteq \Sigma^*$ with its characteristic function

$$Q : \Sigma^* \to \{0, 1\}$$

mapping every $x \in Q$ to 1 and every $x \in \Sigma^* \setminus Q$ to 0. Then parameterized decision problems are special cases of *parameterized counting problems*: pairs (F, κ) of a function $F : \Sigma^* \to \mathbb{N}$ and a parameterization κ.

A function from strings to strings is *fpt computable* with respect to κ if it is computable by an *fpt algorithm*, i.e. one running in *fpt time*

$$f(\kappa(x)) \cdot |x|^{O(1)}$$

for some computable $f : \mathbb{N} \to \mathbb{N}$. A parameterized counting problem (F, κ) is *fixed-parameter tractable* if F is fpt computable with respect to κ (values of F coded in binary); the class of such problems is FPT. The class XP contains those (F, κ) such that for some computable $f : \mathbb{N} \to \mathbb{N}$, F is computable in time

$$|x|^{f(\kappa(x))}.$$

2.2. **Parameterized reductions.** A parameterized counting problem (F,κ) is *fpt reducible* to another (F',κ') if there is an fpt computable (with respect to κ) function $r:\Sigma^*\to\Sigma^*$ such that $F'\circ r=F$ and $\kappa'\circ r\leq g\circ\kappa$ for some computable $g:\mathbb{N}\to\mathbb{N}$. Note that this definition includes the case of parameterized decision problems. Intuitively, fpt reductions are required to keep the parameter small.

An *fpt Turing reduction* from a parameterized decision problem (Q,κ) to another (Q',κ') is an fpt algorithm for Q that on input $x\in\Sigma^*$ makes *balanced* oracle queries to Q', that is, for some computable $g:\mathbb{N}\to\mathbb{N}$ every query $x'\in Q'$? made on input x must satisfy $\kappa'(x')\leq g(\kappa(x))$.

Originally (see e.g. [27]) the classes W$[t]$ of the W-hierarchy have been defined as those parameterized problems that are for some constant $d\in\mathbb{N}$ fpt reducible to

p-WSat$(\Omega_{t,d})$	
Input:	a weft t, depth d Boolean circuit C, and $k\in\mathbb{N}$.
Parameter:	k.
Problem:	does C have a satisfying assignment of weight k?

For the purpose of this paper it is most convenient to define the classes by their complete model-checking problems. We recall the necessary notation.

2.3. **First-order logic.** A (relational) vocabulary τ is a *finite* non-empty set of relation symbols each with an associated arity. A τ-structure $\mathcal{A}$ consists of a nonempty *finite universe* A and an interpretation $R^{\mathcal{A}}\subseteq A^r$ of each r-ary relation symbol $R\in\tau$. A τ-structure $\mathcal{B}$ is an (induced) *substructure* of a τ-structure $\mathcal{A}$ if $B\subseteq A$ and $R^{\mathcal{B}}=R^{\mathcal{A}}\cap B^r$ for each r-ary $R\in\tau$; in this case $\mathcal{A}$ is an *extension* of $\mathcal{B}$. For $\tau\subseteq\tau'$, a τ'*-expansion* of a τ-structure $\mathcal{A}$ is a τ'-structure obtained from $\mathcal{A}$ by adding interpretations of the symbols in $\tau'\setminus\tau$.

First-order τ-formulas are build from τ-atoms using Boolean connectives and existential and universal quantification. τ-atoms are of the form $x_1=x_2$ or $Rx_1\cdots x_r$ for a relation symbol $R\in\tau$ of arity r where $x_1,x_2,\dots$ are individual variables. We write $\varphi(\bar{x})$ to indicate that the free variables in φ are among $\bar{x}$. The set of tuples $\bar{a}$ from A of the same length as $\bar{x}$ that satisfy $\varphi(\bar{x})$ in $\mathcal{A}$ is denoted

$$\varphi(\mathcal{A}).$$

A τ-formula *with parameters in* $\mathcal{A}$ is one containing besides symbols from τ also *parameters* $a\in A$ in place of variables. It can be interpreted only in structures containing its parameters. $\varphi\frac{\bar{a}}{\bar{x}}$ is the formula with parameters obtained from φ by substituting in φ the parameters $\bar{a}$ for the free occurences of the variables $\bar{x}$.

2.4. **Least fixed-point logic.** LFP extends first-order logic by declaring $[\mathrm{lfp}_{\bar{x},X}\varphi]\bar{z}$ a τ-formula of LFP whenever $|\bar{x}|=|\bar{z}|$ and $\varphi=\varphi(\bar{x}\bar{y})$ is a $\tau\,\dot\cup\,\{X\}$-formula of LFP, where X is a $|\bar{x}|$-ary relation symbol outside τ that occurs positively in φ. It has free variables $\bar{y}\bar{z}$. It is satisfied by $\bar{b}\bar{c}$ in a τ-structure $\mathcal{A}$ if $\bar{c}$ is in the least fixed-point reached when, starting with $\emptyset$, iterating the operation

$$B\mapsto\varphi\frac{\bar{b}}{\bar{y}}\big((\mathcal{A},B)\big)$$

on subsets of A. Here, $(\mathcal{A},B)$ is the $\tau\,\dot\cup\,\{X\}$-expansion of $\mathcal{A}$ interpreting X by B.

2.5. **Model-checking problems.** For a set of LFP formulas Φ we consider the *parameterized model-checking problem* p-MC(Φ) and its counting version p-#MC(Φ):

p-MC(Φ)	
Input:	a structure $\mathcal{A}$ and a formula $\varphi \in \Phi$.
Parameter:	the length of φ.
Problem:	Is $\varphi(\mathcal{A}) \neq \emptyset$?

p-#MC(Φ)	
Input:	a structure $\mathcal{A}$ and a formula $\varphi \in \Phi$.
Parameter:	the length of φ.
Problem:	compute $|\varphi(\mathcal{A})|$.

Using instead the number of (free or bound) variables in φ as parameter results in the decision problem *var*-MC(Φ) and its counting version *var*-#MC(Φ).

We introduce notation for important classes Φ. Let $\ell, t, u \in \mathbb{N}$. As usual

$$\Pi_\ell$$

denotes the class of prenex first-order formulas with ℓ alternating blocks of quantifiers starting with $\forall$. Continuing the alternating quantifier blocks by t such blocks each of length at most u we get the class

$$\Pi_{\ell,t,u}.$$

E.g. a $\Pi_{2,3,1}$ formula looks like $\forall \bar{x}_1 \exists \bar{x}_2 \forall y_1 \exists y_2 \forall y_3\ \varphi$ for quantifier free φ, single variables y_1, y_2, y_3 and tuples of variables $\bar{x}_1, \bar{x}_2$. Note $\Pi_\ell = \Pi_{\ell,0,0}$ and Π_0 is the class of quantifier-free formulas. Further,

$$\mathrm{LFP}^u$$

denotes the class of LFP formulas of the form $[\mathrm{lfp}_{\bar{x},X}\varphi]\bar{z}$ such that φ is first-order with at most u *bounded* quantifiers and $|\bar{x}| \leq u$.

2.6. **Parameterized classes.** Parameterized complexity knows two sources of complexity, namely quantificational and propositional alternation ([27, p.337], [37, p.195]). These lead to a two-dimensional array of apparently intractable classes, the A-matrix [29, 36] containing the W-hierarchy as its first row.

Definition 2.1. Let $\ell, t \geq 1$.

(a) A$[\ell, t]$ is the class of those parameterized problems that are fpt reducible to p-MC$(\Pi_{\ell-1,t-1,1})$;
(b) W[SAT] is the class of those parameterized problems that are fpt reducible to *var*-MC(Π_0);
(c) W[P] is the class of those parameterized problems that are fpt reducible to p-MC(LFP^2);
(d) W$[t]$:= A$[1, t]$ and A$[\ell]$:= A$[\ell, 1]$.

The counting classes #A$[\ell, t]$, #W[SAT] and #W[P] are analogously defined via the couting problems p-#MC$(\Pi_{\ell-1,t-1,1})$, *var*-#MC(Π_0) and p-#MC(LFP^2). Further, #W$[t]$:= #A$[1, t]$ and #A$[\ell]$:= #A$[\ell, 1]$.

Remark 2.2. Note that in this setup W[1] = A[1] holds by definition. For $u \geq 1$, the class A$[\ell, t]$ also contains the problems fpt reducible to p-MC$(\Pi_{\ell-1,t-1,u})$, and

W[P] contains those fpt reducible to p-MC(LFP^u) for all $u \geq 1$. [1] Concerning our definition of p-MC(Π_ℓ), observe that deciding non-emptiness of extensions of Π_ℓ-*formulas* is tantamount to deciding truth of $\Sigma_{\ell+1}$-*sentences.* E.g. to decide, whether a given existential first-order sentence holds in a given structure, is complete for W[1] (the parameter is the length of the sentence).

2.7. **Machine characterizations.** The machine model (cf. [37]) is based on a standard notion [25] of a random access machine (RAM) with the uniform cost measure. A RAM works with an infinite sequence of registers each containing a natural number. The first register is called the *accumulator.* A *program* is a finite sequence of basic instructions allowing to copy around register contents, add and subtract (cut off at 0) them and to test if the accumulator is 0 or not.

A program for a *nondeterministic* RAM can additionally guess a number by nondeterministically replacing its accumulator content by a smaller number (and make, say, no change if the accumulator content is 0). This accumulator content is called the *guess bound* of the nondeterministic step. Such a nondeterministic step can be either *existential* or *universal* and acceptance is explained as for alternating Turing machines. For $\ell \in \mathbb{N}$, such a program $\mathbb{P}$ is ℓ*-alternating* if on any run on any input it makes at most ℓ alternating guesses starting with an existential one.

Let κ be a parameterization. A program $\mathbb{P}$ (for a nondeterministic RAM) is κ*-restricted* if there are a computable $f : \mathbb{N} \to \mathbb{N}$ and a constant $c \in \mathbb{N}$ such that for every $x \in \Sigma^*$ and every run of $\mathbb{P}$ on x, $f(\kappa(x))$ bounds the number of nondeterministic steps and $f(\kappa(x)) \cdot |x|^c$ bounds the number of steps, the indices of registers used and the numbers contained in any register at any time; if additionally all nondeterministic steps occur among the last $f(\kappa(x))$ many steps, $\mathbb{P}$ is *tail-nondeterministic* (with respect to κ).

Definition 2.3. A program $\mathbb{P}$ or a nondeterministic Turing machine is *exact* if for every input $x \in \Sigma^*$ any two runs of $\mathbb{P}$ on x contain the same number of nondeterministic steps.

Definition 2.4. A program $\mathbb{P}$ has *uniform guess bounds* if for every input $x \in \Sigma^*$ there is an $n_x \in \mathbb{N}$ such that n_x is the guess bound of every nondeterministic step in every run of $\mathbb{P}$ on x.

The following result is from [21]. The additional and optional claims in parentheses are very easy to verify. Note that 1-alternating programs are those making only existential nondeterministic steps.

Theorem 2.5. *Let (Q, κ) be a parameterized problem and $\ell \in \mathbb{N}$.*

(1) $(Q, \kappa) \in$ W[P] *if and only if Q can be decided by a κ-restricted, 1-alternating program (with uniform guess bounds).*
(2) $(Q, \kappa) \in$ A[ℓ] *if and only if Q can be decided by a tail-nondeterministic κ-restricted, ℓ-alternating program(with uniform guess bounds).*

The classes of the W-hierarchy and, more generally, the A-matrix can be characterized by machines using a second sort of nondeterminism [22] (for "propositional" alternation). For proofs of these and the other results mentioned in this section we refer to the monograph [37] and to [21] for W[P]. A relatively short presentation of these proofs can be found in [56, Chapter 1].

[1]The authors do not know whether p-#MC(LFP^1) is #W[P]-complete.

3. Parameterized randomization: basic observations and techniques

3.1. **Introduction.**

This section defines the two modes of parameterized randomization, namely W[P]- and W[1]-randomization, according to the idea outlined in the Introduction and defines the corresponding two analogues of BPP, namely BPFPT and BPFPT[1]. We give a sequence of propositions and lemmas, meant to provide some first, preliminary understanding of these concepts. These statements are mainly concerned with the following three types of robustness of the definitions.

First, to what extent do the classes depend on the bound on the error probability? We make clear what can be achieved by standard, classical amplification methods.

Second, to what extent does the computational power depend on the size of dice used? We essentially answer this question (Lemma 3.16 and Propositions 3.17 and 3.18). The main result is the so-called Dice Lemma, our main technical tool used throughout the paper. It tells us how to alter a given program to one using larger dice with a desired number of sides.

Third, we characterize W[P]-randomized computations by Turing machines. This characterization is exploited later in Section 5.

3.2. **Parameterized randomized tractability.**

Recall Definitions 2.3 and 2.4.

Definition 3.1. Let κ be a parameterization.

(a) An exact, κ-restricted, 1-alternating program with uniform guess bounds is W[P]*-randomized (with respect to κ).*
(b) An exact, tail-nondeterministic, κ-restricted, 1-alternating program with uniform guess bounds is W[1]*-randomized (with respect to κ).*

We usually omit the phrase "with respect to κ" as usually κ is clear from the context. Some mode of speech and notation: let κ be a parameterization and $\mathbb{P}$ be a W[P]- or W[1]-randomized program (with respect to κ). Say, $\mathbb{P}$ on input $x \in \Sigma^*$ performs exactly $d(x)$ many nondeterministic steps each with guess bound n_x. Then we say that $\mathbb{P}$ *on x uses $d(x)$ many n_x-sided dice.* For $n, m \in \mathbb{N}, n < m$, write

$$[n, m] := \{n, n+1, \dots, m\}.$$

We refer to a possible outcome of the dice rolls of $\mathbb{P}$ on x as a *random seed for $\mathbb{P}$ on x*, i.e. a random seed is an element of $[0, n_x - 1]^{d(x)}$. By $\mathbb{P}(x)$ we denote the output of $\mathbb{P}$ on x: this is the function mapping a random seed for $\mathbb{P}$ on x to the output of $\mathbb{P}$ on x of the run determined by the random seed. $\mathbb{P}(x)$ is a random variable with respect to the probability space given by the uniform probability measure on the set of random seeds of $\mathbb{P}$ on x.

As usual, we sloppily denote various probability measures always by Pr and let the context determine what is meant. E.g. when talking about $\mathbb{P}(x)$, by Pr we refer to the uniform measure on $[0, n_x - 1]^{d(x)}$.

Remark 3.2. We can usually assume that the number of dice a given W[P]- or W[1]-randomized program uses depends only on the input parameter, that is, on input x the program uses exactly $g(\kappa(x))$ many dice for some computable $g : \mathbb{N} \to \mathbb{N}$: if a W[P]- or W[1]-randomized program $\mathbb{P}$ on an input x uses $d(x)$ many dice, then $d \le g \circ \kappa$ for some computable $g : \mathbb{N} \to \mathbb{N}$. Consider the program $\mathbb{P}'$ that on x simulates $\mathbb{P}$ and, when entering the first nondeterministic step, rolls exactly $g(\kappa(x))$ many dice and then continues simulating $\mathbb{P}$ using as random seed the first $d(x)$ many outcomes of its dice rolls. Note that the number $g(\kappa(x))$ is fpt-computable from x and

$$\mathbb{P}'(x) \sim \mathbb{P}(x),$$

that is, the random variables $\mathbb{P}'(x)$ and $\mathbb{P}(x)$ have the same distribution.

Recall that we do not distinguish notationally between a problem $Q \subseteq \Sigma^*$ and its characteristic function.

Definition 3.3. Let $b : \Sigma^* \to [0, 1)$, (Q, κ) be a parameterized problem and $\mathbb{P}$ be W[P]- or W[1]-randomized program with respect to κ. $\mathbb{P}$ *decides* (Q, κ) *with two-sided (one-sided) error* b if

$$\Pr\left[\mathbb{P}(x) \neq Q(x)\right] < \; b(x)$$

for all $x \in \Sigma^*$ (and additionally $\Pr\left[\mathbb{P}(x) = 1\right] = 0$ for all $x \notin Q$).

Definition 3.4.

(a) BPFPT and RFPT are the classes of parameterized problems (Q, κ) decidable by a W[P]-randomized program with two-sided, respectively, one-sided error $1/|x|$.

(b) BPFPT[1] and RFPT[1] are the classes of parameterized problems (Q, κ) decidable by a W[1]-randomized program with two-sided, respectively, one-sided error $1/|x|$.

Proposition 3.5. RFPT $\subseteq$ W[P], RFPT[1] $\subseteq$ W[1] *and* BPFPT $\subseteq$ XP.

The first two statements follow directly from the definitions and Theorem 2.5. The third follows by brute force derandomization: simulate the algorithm exhaustively on all its random seeds. We omit the details.

We use polynomial identity testing for arithmetical terms as a running example throughout the text. We parameterize it by the number of variables:

var-PIT[terms]	
Input:	an arithmetical term C.
Parameter:	the number of variables in C.
Problem:	is p_C nonzero?

Here, p_C denotes the polynomial over $\mathbb{Z}$ computed by C. Recall, an arithmetical circuit is a circuit whose inner nodes compute, over the integers, binary $+$ (addition) or binary $\times$ (multiplication) or unary $-$ (unary additive inverse) and whose input nodes are variables or constants $0, 1$. It is a *term* if inner nodes have fan-out one.

Remark 3.6. From an arithmetical circuit C one can get in polynomial time another C' by replacing a variable X in C by a sufficently large power of X such that $p_C \neq 0 \iff p_{C'} \neq 0$. This well-known reduction shows that parameterizing by the number of variables does not make sense for circuits. For terms this reduction can move from a circuit C with k variables to one with k/ℓ variables in time $|C|^{O(\ell)}$. In the terminology of parameterized complexity theory this means that *var*-PIT[terms] is *scalable* [24] or *length-condensable* [20].

Example 3.7 (PIT)**.** *var*-PIT[terms] *is decidable by a* W[P]*-randomized program with one-sided error* $1 - \Omega(1/\log n)$.

Of course, here n refers to the size of the input. The proof relies on [7, 8]

Lemma 3.8. *There is a polynomial time computable function mapping every arithmetical term to an equivalent one (with the same variables) of logarithmic depth.*

Here, being equivalent means to compute the same polynomial.

Sketch of proof of Example 3.7: Let C be an arithmetical term of size n with k variables. By the lemma above we can assume that C has depth $O(\log n)$. It is routine to verify that then for some suitably large constant c we have $\deg(p_C) \leq n^c$ and $|p_C(\bar{a})| < \|\bar{a}\|^{n^c}$ where $\|\bar{a}\| := \max\{2, \max_i |a_i|\}$ for $\bar{a} \in \mathbb{Z}^k$.

Let q be the least prime bigger than $2n \log n$. By Bertrand's Postulate $q < 4n \log n$ and thus q can be computed in polynomial time.

Our program rolls 'few', $k+1$ many, 'large' q^c-sided dice, say with outcome $\bar{a}b$ where $\bar{a} \in [0, q^c - 1]^k$ and $b \in [0, q^c - 1]$. It computes $p_C(\bar{a}) \bmod b$ and accepts if the outcome is not 0; otherwise it rejects.

If $p_C = 0$ the program rejects for sure. If otherwise $p_C \neq 0$, then the algorithm accepts with probability at least $\Omega(1/\log n)$. This follows according to the standard analysis of the Schwarz-Zippel Algorithm. □

We shall see in Section 4 how to improve the error from $1 - \Omega(1/\log n)$ to $1/n$.

3.3. **Random kernels.**

Just as deterministic fixed-parameter tractability is characterized by the existence of kernelizations, randomized fixed-parameter tractability is characterized by the existence of randomized kernelizations. For brevity, we state and prove this simple result only for the case of W[1]-randomized computations with two-sided error $1/|x|$.

Definition 3.9. Let (Q, κ) be a parameterized problem and $b : \Sigma^* \to [0, 1)$. A W[P]*-randomized (*W[1]*-randomized) kernelization with two-sided error* b is a W[P]-randomized (W[1]-randomized) program $\mathbb{P}$ running in polynomial time such that for some computable $f : \mathbb{N} \to \mathbb{N}$ we have for all $x \in \Sigma^*$

$$\Pr\left[Q(x) \neq Q(\mathbb{P}(x)) \text{ or } |\mathbb{P}(x)| > f(\kappa(x))\right] < b(x).$$

Proposition 3.10. *A parameterized problem (Q, κ) is in* BPFPT[1] *if and only if Q is decidable and (Q, κ) has a* W[1]*-randomized kernelization with two-sided error* $1/|x|$.

Proof. Assume $(Q, \kappa) \in$ BPFPT[1], say witnessed by the program $\mathbb{P}_Q$ with running time bounded by $f(\kappa(x)) \cdot |x|^c$ for computable $f : \mathbb{N} \to \mathbb{N}$ and $c \in \mathbb{N}$. Then Q is decidable by Proposition 3.5. We assume that both $Q \neq \emptyset$ and $\Sigma^* \setminus Q \neq \emptyset$ and choose $x_1 \in Q$ and $x_0 \in \Sigma^* \setminus Q$. The kernelization $\mathbb{P}$ simulates $\mathbb{P}_Q$ on x for at most $|x|^{c+1}$ steps. If this computation halts, $\mathbb{P}$ outputs x_0 or x_1 according to the answer obtained. Otherwise $f(\kappa(x)) > |x|$ and $\mathbb{P}$ outputs x.

Conversely, choose $\mathbb{P}$ and $f : \mathbb{N} \to \mathbb{N}$ as stated and let $\mathbb{A}$ decide Q in time $t_{\mathbb{A}}(|x|)$. We can assume that $t_{\mathbb{A}}$ is nondecreasing and computable. A randomized program $\mathbb{P}_Q$ for Q simply does the following: on $x \in \Sigma^*$ it simulates first $\mathbb{P}$ on x, and then $\mathbb{A}$ on the output obtained for at most $t_{\mathbb{A}}(f(\kappa(x)))$ steps. If $\mathbb{A}$ does not halt within that time, $\mathbb{P}_Q$ rejects. Otherwise it answers as $\mathbb{A}$.

With probability at least $1 - 1/|x|$ both $Q(\mathbb{P}(x)) = Q(x)$ and $|\mathbb{P}(x)| \leq f(\kappa(x))$. In this case $\mathbb{A}$ halts within $t_{\mathbb{A}}(f(\kappa(x)))$ steps (recall $t_{\mathbb{A}}$ is nondecreasing) and then $\mathbb{P}_Q$ answers correctly. Hence $\mathbb{P}_Q$ has error $1/|x|$. Clearly $\mathbb{P}_Q$ uses the same dice as $\mathbb{P}$. After the simulation of $\mathbb{P}$ it simulates at most $t_{\mathbb{A}}(f(\kappa(x)))$ many steps of $\mathbb{A}$. Thus $\mathbb{P}_Q$ uses its dice only in the end of the computation, i.e. $\mathbb{P}_Q$ is W[1]-randomized. □

Remark 3.11. A slight modification of the above argument shows that we can equivalently require the kernelization to additionally satisfy

$$\Pr\left[\kappa(\mathbb{P}(x)) \leq \kappa(x)\ ,\ |\mathbb{P}(x)| \leq f(\kappa(x))\right] = 1.$$

This parallels the deterministic case, where the existence of kernelizations and *parameter non-increasing* kernelizations are equivalent (this fails, however, for *polynomial* kernelizations [23, Proposition 3.3]).

3.4. **Trivial amplification.**

A trivial method of probability amplification is naive repetition:

Lemma 3.12. *Let $b : \Sigma^* \to [0,1)$ and (Q,κ) be a parameterized problem. Assume $\ell : \Sigma^* \to \mathbb{N}$ is fpt computable with respect to κ and $\ell \leq h \circ \kappa$ for some computable $h : \mathbb{N} \to \mathbb{N}$. Then*

(1) *if (Q,κ) can be decided by a* W[P]*-randomized (*W[1]*-randomized) program with two-sided error b, then also with two-sided error $\binom{2\ell}{\ell+1} \cdot b^{\ell+1}$.*
(2) *if (Q,κ) can be decided by a* W[P]*-randomized (*W[1]*-randomized) program with one-sided error b, then also with one-sided error b^{ℓ}.*

Proof. We only show (1). Let $\mathbb{P}$ be a W[P]- or W[1]-randomized program deciding (Q,κ) with two-sided error b. A program $\mathbb{P}'$ with the claimed error runs $\mathbb{P}$ independently for $2\ell(x)$ times and takes a majority vote.

If $\mathbb{P}$ is W[P]-randomized, then so is $\mathbb{P}'$. If $\mathbb{P}$ is W[1]-randomized, then we implement $\mathbb{P}'$ as a W[1]-randomized program as follows: on input x first simulate $\mathbb{P}$ until it reaches the configuration $\mathcal{C}$ when it is about to roll the first of its, say, $g(\kappa(x))$ many dice; compute $k^* := \ell(x) \cdot g(\kappa(x))$ and roll k^* many dice (note that k^* can be computed from x in fpt time); simulate $\mathbb{P}$ started at $\mathcal{C}$ using the first $g(\kappa(x))$ outcomes of your dice rolls, then simulate $\mathbb{P}$ started at $\mathcal{C}$ using the second $g(\kappa(x))$ outcomes of your dice rolls and so on. Thus, $\mathbb{P}'$ simulates $\ell(x)$ computations that are effectively time-bounded in terms of the parameter ($\mathbb{P}$ is W[1]-randomized). Since $\ell(x)$ too is effectively bounded in terms of the parameter, $\mathbb{P}'$ is W[1]-randomized.□

Proposition 3.13. *Let $h : \mathbb{N} \to \mathbb{N}$ be computable. Every (Q,κ) in* BPFPT[1] *(in* BPFPT*) can be decided by a* W[1]*-randomized (*W[P]*-randomized) program with two-sided error $|x|^{-h(\kappa(x))}$.*

Proof. Let (Q,κ) be decidable by a W[1]-randomized or W[P]-randomized program with two-sided error $1/|x|$. Applying the previous lemma with $\ell = h \circ \kappa$ gives a program with error $1/|x|^{h(\kappa(x))}$ on instances x with $|x| \geq \binom{2h(\kappa(x))}{h(\kappa(x))+1}$. On other instances we run a 'brute force' decision procedure for Q (note Q is decidable by Proposition 3.5). The time needed is bounded effectively in $\kappa(x)$, so the running time is fpt. □

Lemma 3.12 is useless to amplify e.g. constant success probabilities. Amplification from e.g. $3/4$ to $1 - 2^{2^{-k}}$ can be done by standard methods:

Lemma 3.14. *Let (Q,κ) be a parameterized problem and $g : \mathbb{N} \to \mathbb{N}$ be unbounded and nondecreasing. There is a* W[P]*-randomized (*W[1]*-randomized) program deciding (Q,κ) with two-sided error $1/4$ if and only if there is such a program with two-sided error $1/g \circ \kappa$.*

Sketch of Proof. The forward direction follows by standard probability amplification based on Chernoff bounds. W[1]-randomization can be preserved by the same trick as in the previous proof. Conversely, assume $\mathbb{P}$ is a program for (Q,κ) with two-sided error $1/g\circ\kappa$. This is error $1/4$ on instances with a parameter k such that $g(k)\geq 4$. There are only finitely many parameters k such that $g(k)<4$. On instances with such a parameter a deterministic XP-algorithm $\mathbb{A}$ (from Proposition 3.5) runs in polynomial time, say $|x|^{100}$. Hence we can first run $\mathbb{A}$ for at most $|x|^{100}$ steps and, if this does not lead to an answer, then run $\mathbb{P}$. □

3.5. **The size of dice.** Assume we run a given program with some larger dice by interpreting in some reasonable way each outcome of a roll with a larger die as an outcome of a roll with the original smaller die. The new program can be seen as the old one using a defective random source, i.e. loaded dice. We need to estimate the loss of the success probability. First, intuitively, the larger the new dice (compared with the old one) the better. Secondly, the more dice, the worse. This is made precise by the "Dice Lemma" below.

Recall that two random variables X,Y with range E have *distance in variation*

$$d_V(X,Y) := \sup_{A\subseteq E} \big|\Pr[X\in A]-\Pr[Y\in A]\big|.$$

We shall use the following elementary lemma (see e.g. [9, Chapter 4, Lemma 1.1]):

Lemma 3.15. *Let X,Y be random variables with an at most countable range E. Then*

$$d_V(X,Y)=\frac{1}{2}\sum_{e\in E}\big|\Pr[X=e]-\Pr[Y=e]\big|.$$

Lemma 3.16 (Dice Lemma)**.** *Let κ be a parameterization and $g:\mathbb{N}\to\mathbb{N}$ computable. Let $\mathbb{P}$ be a* W[P]*-randomized (*W[1]*-randomized) program that on $x\in\Sigma^*$ uses $g(\kappa(x))$ many n_x-sided dice. Let $(q_x)_{x\in\Sigma^*}:\Sigma^*\to\mathbb{N}$ be fpt computable (output coded in unary) such that for all $x\in\Sigma^*$*

$$2g(\kappa(x))\cdot n_x<q_x. \tag{1}$$

Then there is a W[P]*-randomized (*W[1]*-randomized) program $\mathbb{P}'$ which on $x\in\Sigma^*$ uses $g(\kappa(x))$ many q_x-sided dice such that for all $x\in\Sigma^*$*

$$d_V(\mathbb{P}'(x),\mathbb{P}(x))\leq\begin{cases}0 & \text{, if } n_x \text{ divides } q_x\\ g(\kappa(x))\cdot n_x/q_x & \text{, else.}\end{cases}$$

Proof. On input x, the program $\mathbb{P}'$ first computes q_x and then starts simulating $\mathbb{P}$ on x as follows. When $\mathbb{P}$ is about to roll its first n_x-sided die $\mathbb{P}'$ interrupts the simulation and computes a table of the values $r \bmod n_x$ for all $r\in[n_x,q_x-1]$ enabling it to compute $r\mapsto r\bmod n_x$ in constant time. Then $\mathbb{P}'$ continues the simulation of $\mathbb{P}$ in the following manner: if $\mathbb{P}$ rolls one of its n_x-sided dice, $\mathbb{P}'$ rolls one of its q_x-sided dice, say with outcome r, computes $r\bmod n_x$ and continues simulating $\mathbb{P}$ using $r\bmod n_x$ as outcome of the the dice roll. It is clear that $\mathbb{P}'$ is W[P]-randomized (W[1]-randomized) with respect to κ.

Fix an instance $x\in\Sigma^*$ and write

$$q:=q_x, n:=n_x \text{ and } k:=g(\kappa(x)).$$

We show that $\mathbb{P}'(x)$ is distributed as claimed. $\mathbb{P}'$ outputs on some run determined by the outcomes of dice rolls $\bar{a}$ as $\mathbb{P}$ outputs on the (componentwise) residual $\bar{a}\bmod n$. Clearly, having the same such residual is an equivalence relation on $[0,q-1]^k$. Then the preimage of an event A (subset of the range of $\mathbb{P}(x)$) under $\mathbb{P}'(x)$ is the disjoint union of such equivalence classes $\subseteq [0, q-1]^k$ represented by the elements in the preimage of A under $\mathbb{P}(x)$. Let R be a random variable with values in $[0, n-1]^k$ taking value $\bar{a} \in [0, n-1]^k$ with the (uniform) probability of the equivalence class of $\bar{a}$ in $[0, q-1]^k$. Then

$$\mathbb{P}'(x) \sim \mathbb{P}(x) \circ R.$$

In case n divides q, all equivalence classes $\subseteq [0, q-1]^k$ have the same size and hence R is uniformly distributed in $[0, n-1]^k$. Then $\mathbb{P}(x) \circ R \sim \mathbb{P}(x)$ and $d_V(\mathbb{P}'(x), \mathbb{P}(x)) = 0$ follows.

If n does not divide q, then to run $\mathbb{P}'$ on x amounts to run $\mathbb{P}$ on x using the 'defective' (non uniform) random source R. Intuitively, if the equivalence classes have 'almost' the same size, then R is 'almost' uniform. More precisely, for all $\bar{a} \in [0, n-1]^k$

$$\Pr[R = \bar{a}] \geq \left(\frac{\lceil q/n \rceil - 1}{q}\right)^k \geq (1/n - 1/q)^k,$$

$$\Pr[R = \bar{a}] \leq \left(\frac{\lceil q/n \rceil}{q}\right)^k \leq (1/n + 1/q)^k.$$

Let U be a random variable uniformly distributed in $[1, n-1]^k$. Observe that in general $d_V(X, Y) \geq d_V(f \circ X, f \circ Y)$ for all random variables X, Y and all functions f. Thus, in order to bound $d_V(\mathbb{P}'(x), \mathbb{P}(x)) = d_V(\mathbb{P}(x) \circ R, \mathbb{P}(x) \circ U)$, it suffices to bound $d_V(R, U)$.

Let m be a real > 1. Call m *good* if $n/q \leq 1/(2km)$. We claim that for good m

$$\begin{aligned} (1/n + 1/q)^k - n^{-k} &\leq n^{-k}/m, & (2)\\ n^{-k} - (1/n - 1/q)^k &\leq n^{-k}/m. & (3) \end{aligned}$$

The proofs are similar, we only show (2). Observe

$$(1/n + 1/q)^k - n^{-k} = \Big(1/n \cdot (1 + n/q)\Big)^k - n^{-k} = n^{-k} \cdot ((1 + n/q)^k - 1)$$

is at most n^{-k}/m in case $(1 + n/q)^k \leq 1 + 1/m$. This is the case if and only if $\ln(1+n/q) \leq \ln(1+1/m)/k$. This is true if $n/q \leq \ln(1+1/m)/k$ since $\ln(1+r) \leq r$ for reals $r \geq 0$. This holds if $n/q \leq 1/(km) - 1/(2km^2)$ since $\ln(1+r) \geq r - r^2/2$ for reals $r \in (0, 1)$. The last inequality holds if $m > 1$ is good.

If m is good, (2) and (3) imply

$$|\Pr[R = \bar{a}] - \Pr[U = \bar{a}]| \leq n^{-k}/m$$

for all $\bar{a} \in [0, n-1]^k$. Then $d_V(R, U) \leq 1/(2m)$ by Lemma 3.15. Especially,

$$m := q/(2kn)$$

is good. Note that $m > 1$ by (1). Hence $d_V(R, U) \leq 1/(2m) = kn/q$ as claimed. □

The Dice Lemma implies that 'you can always do with $|x|$-sided dice':

Proposition 3.17 (Small Dice)**.** *Let $d \geq 1, b : \Sigma^* \to [0, 1)$, $g : \mathbb{N} \to \mathbb{N}$ be computable and (Q, κ) be a parameterized problem. Assume (Q, κ) can be decided by a* W[P]*-randomized (*W[1]*-randomized) program $\mathbb{P}$ with two-sided error b such that $\mathbb{P}$ on $x \in \Sigma^*$ uses $g(\kappa(x))$ many dice. Then (Q, κ) can be decided by a* W[P]*-randomized (*W[1]*-randomized) program $\mathbb{P}$ with two-sided error $b + |x|^{-d}$ such that $\mathbb{P}'$ on $x \in \Sigma^*$ uses $O(g(\kappa(x)))$ many $|x|$-sided dice.*

Proof. Say, $\mathbb{P}$ on x uses $g(\kappa(x))$ many n_x-sided dice. Then $n_x \leq h(\kappa(x)) \cdot |x|^c$ for some computable h and some $c \in \mathbb{N}$. It suffices to find a program $\mathbb{P}'$ which works as desired on *large* inputs x with $|x| > \max\{h(\kappa(x)) \cdot g(\kappa(x)), 2\}$. Non-large instances can be decided by 'brute force' in time effectively bounded in the parameter.

First apply the Dice Lemma to get a program $\mathbb{P}''$ which on large inputs x uses $g(\kappa(x))$ many $|x|^{c+d+1}$-sided dice. Note that for large x we have $|x|^{c+d+1} > 2g(\kappa(x)) \cdot n_x$ and thus

$$d_V(\mathbb{P}(x), \mathbb{P}''(x)) \leq g(\kappa(x)) \cdot h(\kappa(x)) \cdot |x|^c / |x|^{c+d+1} < |x|^{-d}.$$

The desired program $\mathbb{P}'$ on a large input x first computes a table containing a bijection $B : [0, |x|-1]^{c+d+1} \to [0, |x|^{c+d+1}-1]$ allowing it to compute B in constant time. It then simulates $\mathbb{P}''$ on x as follows. Whenever $\mathbb{P}''$ rolls one of its $|x|^{c+d+1}$-sided dice, $\mathbb{P}'$ rolls $(c+d+1)$ many $|x|$-sided dice, say with outcome $(a_1, \ldots, a_{c+d+1})$. Using the table it continues the simulation with $B(a_1, \ldots, a_{c+d+1})$.

It is clear that $\mathbb{P}'(x) \sim \mathbb{P}''(x)$ and that $\mathbb{P}'$ is a W[P]-randomized (W[1]-randomized) program using $(c + d + 1) \cdot g(\kappa(x))$ many $|x|$-sided dice. □

In the following sense, improving this lower bound on the size of dice amounts to derandomization.

Proposition 3.18. *Let (Q, κ) be a parameterized problem. If (Q, κ) can be decided by a* W[P]*-randomized program with two-sided error $< 1/2$ that on $x \in \Sigma^*$ uses $|x|^{o^{\text{eff}}(1)}$-sided dice, then (Q, κ) is fixed-parameter tractable.*

Here, by $|x|^{o^{\text{eff}}(1)}$ many sides we mean $\leq |x|^{1/\iota(|x|)}$ many sides for some computable, nondecreasing and unbounded $\iota : \mathbb{N} \to \mathbb{N}$ and sufficiently long x.

Proof. Choose a program $\mathbb{P}$ according to the assumption. Choose a computable $g : \mathbb{N} \to \mathbb{N}$ and a computable, nondecreasing and unbounded $\iota : \mathbb{N} \to \mathbb{N}$ such that $\mathbb{P}$ on input $x \in \Sigma^*$ uses $g(\kappa(x))$ many dice with at most $|x|^{1/\iota(|x|)}$ sides (provided x is sufficiently long). Let $\mathbb{P}'$ on x simulate $\mathbb{P}$ on x exhaustively on all its random seeds. We show that $\mathbb{P}'$ runs in fpt time. For this it suffices to show that the number of random seeds of $\mathbb{P}$ on x obeys an fpt bound.

Let $h : \mathbb{N} \to \mathbb{N}$ be some computable, nondecreasing function such that $h(\iota(n)) \geq n$ for all $n \in \mathbb{N}$. We write $n := |x|$ and $k := \kappa(x)$ and distinguish two cases:

- if $g(k) < \iota(n)$, then there are at most $(n^{1/\iota(n)})^{g(k)} < n$ many random seeds.
- if $g(k) \geq \iota(n)$, then $h(g(k)) \geq h(\iota(n)) \geq n$, and hence there are at most $h(g(k))^{g(k)}$ many random seeds. □

3.6. **Turing characterizations.** As a second application of the Dice Lemma we characterize W[P]-randomized computations by Turing machines:

Proposition 3.19. *Let (Q, κ) be a parameterized problem. The following statements are equivalent.*

(1) $(Q, \kappa) \in$ BPFPT.
(2) *There is a computable $h : \mathbb{N} \to \mathbb{N}$ and an exact randomized fpt time bounded (with respect to κ) Turing machine $\mathbb{A}$ such that for all $x \in \Sigma^*$ and every run of $\mathbb{A}$ on x the machine $\mathbb{A}$ tosses at most $h(\kappa(x)) \cdot \log |x|$ many coins and decides Q with two-sided error at most $1/|x|$.*

This follows from trivial amplification (Proposition 3.13) and Lemma 3.22 below. We derive a characterization of the subclass of BPFPT consisting of parameterized versions of problems in BPP:

Proposition 3.20. *Let (Q, κ) be a parameterized problem. The following statements are equivalent.*

(1) $(Q, \kappa) \in \mathrm{BPFPT}$ *and* $Q \in \mathrm{BPP}$.
(2) *There is a computable $h : \mathbb{N} \to \mathbb{N}$ and an exact, randomized, polynomially time bounded Turing machine $\mathbb{A}$ such that for all $x \in \Sigma^*$ and every run of $\mathbb{A}$ on x the machine $\mathbb{A}$ tosses at most $h(\kappa(x)) \cdot \log |x|$ many coins and decides Q with two-sided error at most $1/|x|$.*

Proof. That (2) implies (1) follows by Proposition 3.19. Conversely, let $\mathbb{P}$ witness that $(Q, \kappa) \in \mathrm{BPFPT}$ and let $\mathbb{A}_Q$ witness that $Q \in \mathrm{BPP}$. We can assume that $\mathbb{A}_Q$ is exact and has error at most $1/|x|$. Choose for $\mathbb{P}$ a Turing machine $\mathbb{A}$ and a function h according to Proposition 3.19 (2). Say, $\mathbb{A}$ on x needs time at most $f(\kappa(x)) \cdot |x|^{O(1)}$ for some time-constructible[2] f.

Let $\mathbb{A}'$ be the following Turing machine: on x it checks if $f(\kappa(x)) \leq |x|$ (this can be done in polynomial time because f is time-constructible); in case, it simulates $\mathbb{A}$, and otherwise $\mathbb{A}_Q$. Then $\mathbb{A}'$ is an exact randomized Turing machine deciding Q in polynomial time with two-sided error $1/|x|$. It uses at most $h(\kappa(x)) \cdot \log |x|$ many coins if $|x| \geq f(\kappa(x))$ and at most $f(\kappa(x))^{O(1)}$ many coins otherwise. □

The last proposition parallels [11, Theorem 3.7]:

Theorem 3.21 (Cai, Chen, Downey, Fellows 1995)**.** *A parameterized problem (Q, κ) with $Q \in \mathrm{NP}$ is in $\mathrm{W[P]}$ if and only if $x \in Q$ can be decided in nondeterministic polynomial time with at most $h(\kappa(x)) \cdot \log |x|$ nondeterministic bits for some computable $h : \mathbb{N} \to \mathbb{N}$.*

Lemma 3.22. *Let κ be a parameterization.*

(1) *Let $d \in \mathbb{N}$. For any (with respect to κ) $\mathrm{W[P]}$-randomized program $\mathbb{P}$ there exists an exact randomized fpt time bounded Turing machine $\mathbb{A}$ and a computable $h : \mathbb{N} \to \mathbb{N}$ such that for all $x \in \Sigma^*$ we have $d_V(\mathbb{P}(x), \mathbb{A}(x)) \leq |x|^{-d}$ and $\mathbb{A}$ on x tosses at most $h(\kappa(x)) \cdot \log |x|$ many coins.*
(2) *For any exact randomized fpt time bounded Turing machine $\mathbb{A}$ such that for some computable $h : \mathbb{N} \to \mathbb{N}$ the machine $\mathbb{A}$ on any $x \in \Sigma^*$ tosses at most $h(\kappa(x)) \cdot \log |x|$ many coins there exists a $\mathrm{W[P]}$-randomized program $\mathbb{P}$ such that $\mathbb{P}(x) \sim \mathbb{A}(x)$ for all $x \in \Sigma^*$.*

Proof. By straightforward simulations between Turing machines and RAMs. We only show (1) to make clear where the Dice Lemma is used:

Say, $\mathbb{P}$ on x uses $g(\kappa(x))$ many n_x-sided dice for some computable $g : \mathbb{N} \to \mathbb{N}$. Apply the Dice Lemma to get a program $\mathbb{P}'$ which on x uses q_x-sided dice for q_x the least power of two above $g(\kappa(x)) \cdot n_x \cdot |x|^d$. Then

$$d_V(\mathbb{P}(x), \mathbb{P}'(x)) \leq g(\kappa(x)) \cdot n_x / q_x \leq |x|^{-d}.$$

The Turing machine $\mathbb{A}$ on x simulates $\mathbb{P}'$ on x tossing $\log q_x$ many coins whenever $\mathbb{P}'$ rolls a die. Then $\mathbb{A}$ is exact and $\mathbb{A}(x) \sim \mathbb{P}'(x)$. The machine $\mathbb{A}$ tosses $g(\kappa(x)) \cdot$

[2]Recall a function $f : \mathbb{N} \to \mathbb{N}$ is time-constructible if and only if there is a (deterministic) Turing machine that on every x halts after exactly $f(|x|)$ many steps.

$\log q_x$ many coins. But $\log q_x \leq h(\kappa(x)) \cdot \log |x|$ for some computable h, because n_x is fpt bounded. □

4. Deterministic probability amplification

4.1. **Introduction.** In the last section we observed that standard methods can amplify success probabilities from $1 - 1/n$ to $1 - 1/n^k$ (on instances of length n with parameter k) and from $3/4$ to $1 - 1/2^k$. Can we amplify from $3/4$ to $1 - 1/n$ (two-sided error) or from $1/2 - 1/n$ to $3/4$? In the classical setting one repeats the random experiment for a sufficiently large number ℓ of times and votes by majority. But in the parameterized setting this does not work because this number ℓ grows with n and thus exceeds the resources of random complexity a W[P]-randomized algorithm has at its disposal.

In the setting given by the parameterized BP operator [29], the first author [51] linked the possibility of parameterized probability amplification to certain closure conditions of parameterized classes. For example, he showed that a parameterized analogue of the Arthur-Merlin class (which is not studied here) enjoys probability amplification assuming it is closed under parameterized truth-table reductions. Unconditionally, he amplified success probabilities for W[P]-randomization from $3/4$ to $1 - 1/n$ by a construction based on the Ajtai-Komlós-Szemerédi generator [1]. Here we give a simpler, direct proof by a method called "deterministic amplification" in [44].

Theorem 4.1. *Let (Q, κ) be a parameterized problem, $g : \mathbb{N} \to \mathbb{N}$ an arbitrary computable function and $c \geq 1$. The following statements are equivalent.*

1. $(Q, \kappa) \in$ BPFPT.
2. (Q, κ) *can be decided by a* W[P]*-randomized program with two-sided error* $\frac{1}{2} - |x|^{-c}$.
3. (Q, κ) *can be decided by a* W[P]*-randomized program with two-sided error* $|x|^{-g(\kappa(x))}$.

Remark 4.2. A similar statement holds true for one-sided error where in (2) the constant $1/2$ is replaced by 1.

4.2. **A simple expander.** We recall some basics from expander theory (see [41, Appendix E], [48] or the comprehensive [44]).

A multigraph is an undirected graph possibly with parallel edges and possibly with self loops. It is d-regular if each vertex is incident to exactly d edges. Let $\mathcal{G}$ be a d-regular multigraph with n vertices. The normalized adjacency matrix $A_{\mathcal{G}}$ of $\mathcal{G}$ is the $n \times n$-matrix whose ij^{th} entry is $1/d$ times the number of edges from the ith to the jth vertex. Since $A_{\mathcal{G}}$ is symmetric there is an orthogonal basis of eigenvectors corresponding to real eigenvalues $\lambda_1, \ldots, \lambda_n$. The matrix $A_{\mathcal{G}}$ is stochastic, i.e. the sum of each row equals 1. Hence 'the uniform distribution' $(n^{-1}, \ldots, n^{-1})$ is an eigenvector of $A_{\mathcal{G}}$ to the eigenvalue, say, $\lambda_1 = 1$. λ_1 is the largest eigenvalue and the corresponding eigenspace is one dimensional in case $\mathcal{G}$ is connected. For all $i \in [n] \setminus \{1\}$ we have $|\lambda_i| \leq 1$ and $|\lambda_i| < 1$ in case $\mathcal{G}$ is aperiodic. The second largest eigenvalue modulus of $\mathcal{G}$ is

$$\lambda(\mathcal{G}) := \max_{i \in [n] \setminus \{1\}} |\lambda_i| \ .$$

An (n, d, λ)-graph is a connected, aperiodic, d-regular multigraph $\mathcal{G}$ on n vertices such that $\lambda(\mathcal{G}) < \lambda$. The following is known as the Expander Mixing Lemma:

Lemma 4.3. *Let $\mathcal{G}$ be a (n, d, λ)-graph and B, B' sets of vertices of $\mathcal{G}$. Write $e(B, B')$ for the number of edges in $\mathcal{G}$ going from B to B'. Then*

$$\big|e(B,B') - |B| \cdot |B'| \cdot d/n\big| \leq d \cdot \lambda \cdot \sqrt{|B| \cdot |B'|}.$$

We shall need a 'strongly explicit' such graph allowing to feasibly determine the list of all neighbors of a given point without computing the graph explicitly. We use the following example:

Example 4.4. Let $q, k \geq 1$, q prime and let $\mathbb{F}_q$ denote the field with q elements. We do not distinguish $\mathbb{F}_q$ from $\{0, \ldots, q-1\}$ notationally. $\mathbb{F}_q^k$ is the k-dimensional vector space over $\mathbb{F}_q$. Vectors are represented with relation to the standard basis as k-tuples over $\{0, \ldots, q-1\}$. The multigraph

$$\mathrm{LD}_{q,k}$$

has vertices $\mathbb{F}_q^{k+1}$ where each vertex $\bar{a} \in \mathbb{F}_q^{k+1}$ has for each $b, c \in \mathbb{F}_q$ an edge to

$$\bar{a} + (b, b \cdot c, \ldots, b \cdot c^k).$$

It is shown in [61, Proposition 5.3] that $\mathrm{LD}_{q,k}$ is a $(q^{k+1}, q^2, k/q)$-graph.

4.3. **Proof of Theorem 4.1.** The amplification procedure is very simple: define an expander graph on the sample space of your given program; the new program samples a point (random seed) in the graph and simulates the old program on all its neighbors. The Expander Mixing Lemma 4.3 tells us how much the success probability is amplified. The method is called "deterministic" because no additional die is rolled. Note that the sample space and thus also the expander has unfeasible size. That's why we need a 'strongly explicit' expander. The following Proposition 4.5 makes this precise. Together with Proposition 3.13 it implies Theorem 4.1.

Proposition 4.5. *Let $c, c' \geq 1$ and (Q, κ) be a parameterized problem. If there is a* W[P]*-randomized program $\mathbb{P}$ solving (Q, κ) with two-sided error $1/2 - |x|^{-c}$, then there is a* W[P]*-randomized program $\mathbb{P}'$ solving (Q, κ) with two-sided error $|x|^{-c'}$, which on any $x \in \Sigma^*$ uses the same number of dice as $\mathbb{P}$.*

Proof. Let $c, c' \geq 1$. Let $c, c' \geq 1$ and assume $\mathbb{P}$ is a W[P]-randomized program deciding (Q, κ) with two-sided error $\frac{1}{2} - \frac{1}{|x|^c}$; let $g : \mathbb{N} \to \mathbb{N}$ be computable that $\mathbb{P}$ on x uses $g(\kappa(x))$ many n_x-sided dice. We first move to a program that uses prime sided dice and whose error is only a little bit worse than that of $\mathbb{P}$: apply the Dice Lemma with q_x the smallest prime such that

$$q_x \geq \max\big\{g(\kappa(x)) \cdot n_x \cdot 2|x|^c \ , \ 2|x|^{2c+c'} \ , \ (g(\kappa(x)) - 1)^2\big\}. \tag{4}$$

Since by Bertrand's Postulate there is a prime between n and $2n$ for every $n > 1$, we can compute q_x from x by brute force in fpt time.

Fix some input $x \in \Sigma^*$ and write

$$n := |x|,\ k := \kappa(x),\ q := q_x \text{ and } p := 1/2 - 1/(2n^c).$$

By the Dice Lemma, the new program errs on x with probability at most $1/2 - n^{-c} + g(k) \cdot n_x/q \leq p$. For simplicity, we denote the new program again by $\mathbb{P}$.

The set of 'bad' random seeds is $B := \{\mathbb{P}(x) \neq Q(x)\}$. Then

$$|B|/q^{g(k)} \leq p. \tag{5}$$

We view B as a subset of $\mathbb{F}_q^{g(k)}$, i.e. a set of vertices of the expander $\mathrm{LD}_{q,g(k)-1}$ from Example 4.4.

On input x, the program $\mathbb{P}'$ first guesses $\bar{r} \in \mathbb{F}_q^{g(k)}$ and then simulates q^2 many runs of $\mathbb{P}$ on x, namely for every $(a,b) \in \mathbb{F}_q^2$ the run determined by the random seed

$$\bar{r}' := \bar{r} + (a, a \cdot b, \dots, a \cdot b^{g(k)-1}).$$

$\mathbb{P}'$ answers according to the majority of the answers obtained. Note that $\bar{r}'$ can be computed given $\bar{r}, a, b$ with at most $g(k)^2$ operations in $\mathbb{F}_q$. Hence $\mathbb{P}'$ is W[P]-randomized and has the same random complexity as $\mathbb{P}$, namely $g(k)$ many q-sided dice. Let $B' := \{\mathbb{P}'(x) \neq Q(x)\}$ be the set of 'bad' random seeds for $\mathbb{P}'$. We aim to show

$$|B'|/q^{g(k)} < n^{-c'}. \tag{6}$$

Each vertex in B' has more than $q^2/2$ edges to vertices in B, i.e.

$$e(B', B) > |B'| \cdot q^2/2. \tag{7}$$

By Lemma 4.3 for $\mathrm{LD}_{q,g(k)-1}$ with $\lambda := \lambda\left(\mathrm{LD}_{q,g(k)-1}\right)$ and (5) we get

$$\begin{aligned} e(B', B) &\leq q^2 \cdot |B| \cdot |B'|/q^{g(k)} + q^2 \cdot \lambda \cdot \sqrt{|B| \cdot |B'|} \\ &\leq q^2 \cdot |B'| \cdot p + q^2 \cdot \lambda \cdot \sqrt{q^{g(k)} \cdot |B'| \cdot p}\ . \end{aligned} \tag{8}$$

Combining (7) and (8) yields

$$(1/2 - p) \cdot q^2 \cdot |B'| < q^2 \cdot \lambda \cdot \sqrt{q^{g(k)} \cdot |B'| \cdot p}$$

and hence (recall $p < 1/2$)

$$|B'|/q^{g(k)} < \lambda^2 \cdot p/(1/2 - p)^2. \tag{9}$$

By (4), $p/(1/2-p)^2 < 2n^{2c}$ and $\lambda^2 \leq (g(k)-1)^2/q^2 \leq 1/q \leq 1/(2n^{2c+c'})$. Thus (9) implies (6). □

Note that the argument given breaks down for W[1]-randomized computations because after rolling dice we run $\mathbb{P}$ on q^2 many, that is, fpt many seeds.

Example 4.6 (PIT, continued). *var*-PIT[terms] $\in$ RFPT.

Proof. By Theorem 4.1 (and Remark 4.2) and Example 3.7. □

5. Parameterized derandomization

5.1. **Introduction.** How does parameterized derandomization relate to classical derandomization? In this section we show that, in a certain sense, parameterized derandomization implies classical derandomization and vice-versa.

We start with an observation due to M. Grohe. For a concise formulation denote by paraNP-BPFPT the class of parameterized problems that can be decided by a randomized fpt time bounded Turing machine with two-sided error $1/n$ (where n is the size of the input). This notation will not be used later on.

Proposition 5.1. *The following statements are equivalent.*

(1) paraNP-BPFPT = FPT.
(2) paraNP-BPFPT = BPFPT.
(3) BPP = P.

Sketch of proof. Trivially (1) implies (2). That (3) implies (1) follows by standard arguments, and we omit the proof. The implication of interest is from (2) to (3):

Assume (2) and let $Q \in$ BPP. Then $(Q, 0)$, i.e. Q with the parameterization which is constantly zero, is in paraNP-BPFPT. By assumption $(Q, 0) \in$ BPFPT. Choose a Turing machine $\mathbb{A}$ according to Proposition 3.20 (2). Then $\mathbb{A}$ on an input $x \in \Sigma^*$ runs in polynomial time and tosses $O(\log |x|)$ many coins. Thus simulating $\mathbb{A}$ on all $|x|^{O(1)}$ many random seeds requires only polynomial time. Thus $Q \in$ P.□

Hence derandomizing randomized fpt algorithm to deterministic fpt algorithms or to W[P]-randomized algorithms, are both tasks that are equivalent to full classical derandomization. What does derandomization of W[P]-randomized algorithms mean in classical terms? The main result of this section reads as follows.

Theorem 5.2. *The following statements are equivalent.*

(1) BPFPT *has a (weakly or strongly) black-box derandomization.*
(2) *There is a polynomial time computable, nondecreasing, unbounded function* $c : \mathbb{N} \to \mathbb{N}$ *such that* BPP$[c]$ = P.

Here, BPP$[c]$ is BPP restricted to at most $c(n) \cdot \log n$ many random bits. Observe that for bounded c we trivially have BPP$[c]$ = P. Thus nontrivial classical derandomization would mean to show BPP$[c]$ = P for some unbounded c. Being “black-box” is some extra condition on BPFPT = FPT which is mathematically strong, but, as we are going to argue, philosophically weak. Informally, Theorem 5.2 states that (“black-box”) derandomization of W[P]-randomized computations means the same as arbitrarily weak but nontrivial classical derandomization. Note that c can grow extremely slowly.

We only treat two-sided error. This is, however, not essential. All results of this section have analogues for one-sided error with ‘the same’ proof.

The reader should compare Theorem 5.2 with [11, Theorem 3.8]:

Theorem 5.3 (Cai, Chen, Downey, Fellows 1995)**.** W[P] = FPT *if and only if there is a polynomial time computable, nondecreasing, unbounded function* $c : \mathbb{N} \to \mathbb{N}$ *such that* NP$[c]$ = P.

Here, NP$[c]$ is the class of classical problems decidable in polynomial time with at most $c(n) \cdot \log n$ nondeterministic bits. An analogue of this statement for the class EW[P] appears in [38, Theorem 25].

5.2. **Black-box derandomization.** In this section we shall use the following notation: a parameterized problem (Q, κ) can be decided in time

$$O^*(f)$$

if it can be decided in time $f(\kappa(x)) \cdot |x|^{O(1)}$. Thus, (Q, κ) is in FPT if and only if it can be decided in time $O^*(f)$ for some computable function f. So derandomization of BPFPT means that each parameterized problem with a W[P]-randomized algorithm can be decided in time $O^*(f)$ for some computable function f. In general this function f may depend on the problem, in other words, on the specific program we are derandomizing. However, it is plausible that if we succeed in proving this, we do so by a general method, a method working well for all algorithms with similar running time and random complexity. In other words, it is plausible that, if we succeed in derandomizing a class of algorithms, then we do so by treating these algorithms as black boxes.

For example in the classical setting derandomization is done by constructing (under certain hardness assumptions) pseudorandom strings, i.e. strings which algorithms obeying a given running time are not able to distinguish from truly random seeds. Given an algorithm to derandomize we feed it with these pseudorandom strings and learn about its acceptance probability. Thereby we decide the problem deterministically. The running time of the deterministic algorithm we get in this way depends only on the running time of the randomized algorithm and the time we need to construct the pseudorandom strings, that is, the new running time depends only on the old running time and the old random complexity.

We therefore think of a "black-box" derandomization as being such that the running time of the new deterministic algorithms it produces depends only on the old time and random complexity. In our setting the random complexity is given by the number and size of dice. But the size of dice can be assumed to be $|x|$ by the Small Dice Lemma 3.17. Formalizing these intuitions leads to the concept of being *strongly black-box*, as defined below (Definition 5.5).

M.Grohe suggested considering the following relaxation. While in the classical setting the length of the random seed can be assumed to be polynomially related to the running time, in the parameterized setting we are asked to produce short pseudorandom sequences of length, say, $g(k) \cdot \log n$ which fool algorithms running in time, say, $g(k) \cdot n^d$. It may be conceivable that the running time of a suitable pseudorandom generator increases with this running time, say, it is d-fold exponential in k. So instead of a single f we may only expect a successful derandomization to produce a family $(f_d)_d$ of functions such that randomized algorithms with $g(k)$ dice and running time $g(k) \cdot n^d$ have determinizations running in time $O^*(f_d)$. This relaxes the property of being strongly black-box to that of being *weakly black-box* (see Definition 5.5 below).

Definition 5.4. Let $t, c : \Sigma^* \to \mathbb{N}$. A classical or parameterized problem *has a* (t, c)*-machine* if and only if it can be decided with two-sided error $1/|x|$ by an exact randomized Turing machine which on x runs for at most $t(x)$ many steps and tosses at most $c(x) \cdot \log |x|$ many coins. For $t, c : \mathbb{N} \to \mathbb{N}$, by a (t, c)*-machine* we mean a $(t \circ |\cdot|, c \circ |\cdot|)$-machine.

For $c : \mathbb{N} \to \mathbb{N}$ we let BPP$[c]$ denote the class of all classical problems that have a (t, c)-machine for some polynomial t.

The following modes of speech rely on Propositions 3.19 and 3.20. As usual we call a family $(f_d)_d = (f_d)_{d\in\mathbb{N}}$ of functions $f_d : \mathbb{N} \to \mathbb{N}$ *computable* if and only if $(d, k) \mapsto f_d(k)$ is computable.

Definition 5.5. BPFPT *has a weakly black-box derandomization* if and only if for all computable $g : \mathbb{N} \to \mathbb{N}$ there is a computable family of functions $(f_d)_d$ such that for all $d \in \mathbb{N}$ and all parameterized problems (Q, κ):

$$\text{if } (Q, \kappa) \text{ has a } \big(g(\kappa(x))\cdot|x|^d, g(\kappa(x))\big)\text{-machine, then } (Q, \kappa) \in O^*(f_d).$$

If we weaken the last condition to:

$$\text{if } (Q, \kappa) \text{ has a } \big(|x|^d, g(\kappa(x))\big)\text{-machine, then } (Q, \kappa) \in O^*(f_d).$$

then we say that *the BPP part of* BPFPT *has a weakly black-box derandomization.*

If the family $(f_d)_d$ is constant, i.e. there is a f such that $f_d = f$ for all d, then we say that BPFPT *has a strongly black-box derandomization.*

Theorem 5.6. *The following statements are equivalent.*

1. BPFPT *has a strongly black-box derandomization.*
2. BPFPT *has a weakly black-box derandomization.*
3. *The BPP part of* BPFPT *has a weakly black-box derandomization.*
4. *There is a polynomial time computable, increasing, time-constructible function $g : \mathbb{N} \to \mathbb{N}$ and there is a computable family of functions $(f_d)_d$ such that for all $d \in \mathbb{N}$ and all parameterized problems (Q, κ):*
   if (Q, κ) has a $\big(|x|^d, g(\kappa(x))\big)$-machine, then $(Q, \kappa) \in O^*(f_d)$.
5. *There is a polynomial time computable, nondecreasing, unbounded function $c : \mathbb{N} \to \mathbb{N}$ such that* $\mathrm{BPP}[c] = \mathrm{P}$.

Remark 5.7. Any computable, nondecreasing and unbounded function is lower bounded by a polynomial time computable such function. Hence in (5) one can equivalently write "computable" instead of "polynomial time computable".

5.3. **Proof of Theorem 5.6.** We need the following simple lemma.

Definition 5.8. For nondecreasing, unbounded $f : \mathbb{N} \to \mathbb{N}$ the *inverse of f* is the function $\iota_f : \mathbb{N} \to \mathbb{N}$ given by

$$\iota_f(n) := \max\{i \in \mathbb{N} \mid f(i) \le n\},$$

where we agree that $\max \emptyset := 0$. Further let $\iota_f^+ := \iota_f + 1$.

Lemma 5.9. *If $f : \mathbb{N} \to \mathbb{N}$ is nondecreasing and unbounded $f : \mathbb{N} \to \mathbb{N}$, then ι_f is nondecreasing and unbounded and for all $n \in \mathbb{N}$*

$$f \circ \iota_f(n) \le \max\{n, f(0)\} \le f \circ \iota_f^+(n).$$

If f is increasing, then $\iota_f \circ f = \mathrm{id}_{\mathbb{N}}$. Furthermore, if $f : \mathbb{N} \to \mathbb{N}$ is increasing and time-constructible, then ι_f is computable in polynomial time.

Proof of Theorem 5.6: The implications from (1) to (2), from (2) to (3) and from (3) to (4) are trivial, so it suffices to show that (4) implies (5) and that (5) implies (1).

We first show that (4) implies (5). So assume (4) and choose a polynomial time computable, increasing, time-constructible $g : \mathbb{N} \to \mathbb{N}$ and a computable family of functions $(f_d)_d$ accordingly. We can assume that for all $d \in \mathbb{N}$ the function $f_d : \mathbb{N} \to \mathbb{N}$ is increasing and time-constructible.

Because the family $(f_d)_d$ is computable there is a time-constructible increasing $\tilde{f} : \mathbb{N} \to \mathbb{N}$ such that for all $n \in \mathbb{N}$

$$\widetilde{f}(n) \ge \max\big\{f_0(n), \dots, f_n(n)\big\}.$$

By Lemma 5.9 the inverse $\iota_{\widetilde{f}}$ of $\widetilde{f}$ is polynomial time computable, nondecreasing and unbounded. For all $d \in \mathbb{N}$ and all $n \ge d$ we have $\widetilde{f}(n) \ge f_d(n)$, so

$$\forall d \in \mathbb{N}\ \forall n \ge \widetilde{f}(d):\ \iota_{\widetilde{f}}(n) \le \iota_{f_d}(n). \tag{10}$$

We denote subtraction of 1 by s, that is, $s(n) := \max\{n - 1, 0\}$ and define

$$c := g \circ s \circ \iota_{\widetilde{f}}.$$

Then c is polynomial time computable, nondecreasing and unbounded, because it is a composition of such functions.

Let $Q \in \mathrm{BPP}[c]$. Then there is a constant $d > 0$ such that Q has a $\big(|x|^d, c(|x|)\big)$-machine $\mathbb{A}$. We aim to show $Q \in \mathrm{P}$.

For $\kappa_c(x) := c(|x|)$ define

$$\kappa := \iota_g^+ \circ \kappa_c.$$

Then κ is polynomial time computable by Lemma 5.9, so (Q, κ) is a parameterized problem. Because $c(|x|) \leq g \circ \iota_g^+ \circ \kappa_c(x) \leq g(\kappa(x))$, $\mathbb{A}$ is a $\big(|x|^d, g(\kappa(x))\big)$-machine. By assumption (4) we get $(Q, \kappa) \in O^*(f_d)$. Thus to show $Q \in \mathrm{P}$ it suffices to show $f_d(\kappa(x)) \leq |x|$ for sufficiently long x.

Observe that for all $n > 0$

$$\iota_g^+ \circ g \circ s(n) = \iota_g(g(s(n)) + 1 = s(n) + 1 = n, \tag{11}$$

so $\iota_g^+ \circ g \circ s$ is the identity on positive numbers. Note further that for $n \geq \widetilde{f}(d)$

$$\iota_{\widetilde{f}}(n) \geq \iota_{\widetilde{f}}(\widetilde{f}(d)) = d > 0. \tag{12}$$

Hence for instances x with $|x| \geq \widetilde{f}(d)$

$$f_d(\kappa(x)) = f_d \circ \iota_g^+ \circ g \circ s \circ \iota_{\widetilde{f}}(|x|) = f_d \circ \iota_{\widetilde{f}}(|x|) \leq f_d \circ \iota_{f_d}(|x|) \leq |x|.$$

The first equality follows from the definition of κ, the second equality from (11) and (12), the first inequality from (10) and f_d being increasing, and the final inequality from $|x| \geq \widetilde{f}(d) \geq f_d(d) > f_d(0)$ and Lemma 5.9.

We now show that (5) implies (1). Assume (5) and choose a (polynomial time) computable, nondecreasing, unbounded $c : \mathbb{N} \to \mathbb{N}$, such that $\mathrm{BPP}[c] = \mathrm{P}$. We have to show for all computable $g : \mathbb{N} \to \mathbb{N}$ how to decide deterministically problems with a $(g(\kappa(x)) \cdot |x|^{O(1)}, g(\kappa(x)))$-machine. Clearly, it is sufficient to do so for all computable nondecreasing g. So fix such a g.

We leave it to the reader to verify

Claim 1: There is a computable $r : \mathbb{N} \to \mathbb{N}$ such that any parameterized problem (Q, κ) with a $(g(\kappa(x)) \cdot |x|^{O(1)}, g(\kappa(x)))$-machine can be decided in (deterministic) time $r(|x|) \cdot |x|^{O(1)}$.

Clearly, ι_c^+ is computable. Choose a time-constructible $f : \mathbb{N} \to \mathbb{N}$ such that

$$f(k) \geq \max\left\{\iota_c^+ \circ g(k)\ ,\ g(k)\right\} \tag{13}$$

for all $k \in \mathbb{N}$. Further, choose r according to Claim 1. Clearly, we can assume that r is nondecreasing. Let (Q, κ) have a $(g(\kappa(x)) \cdot |x|^{O(1)}, g(\kappa(x)))$-machine $\mathbb{A}$. We aim to show

$$(Q, \kappa) \in O^*(r \circ f). \tag{14}$$

Claim 2: $Q_{\geq f} := \{x \in Q \mid |x| \geq f(\kappa(x))\} \in \mathrm{BPP}[c]$.

Proof of Claim 2: Define the machine $\mathbb{A}'$ as follows. On x it checks if $f(\kappa(x)) > |x|$. This can be done in polynomial time since f is time-constructible. If this is the case, it rejects. If $f(\kappa(x)) \leq |x|$ it simulates $\mathbb{A}$. But then $\mathbb{A}$ needs time at most

$$g(\kappa(x)) \cdot |x|^{O(1)} \leq f(\kappa(x)) \cdot |x|^{O(1)} \leq |x| \cdot |x|^{O(1)},$$

where the first inequality is due to (13). Thus $\mathbb{A}'$ runs in polynomial time. Clearly, $\mathbb{A}'$ decides $Q_{\geq f}$ with two-sided error at most $1/|x|$. We estimate its number of coins: if $f(\kappa(x)) > |x|$, then $\mathbb{A}'$ uses no coins at all. If $f(\kappa(x)) \leq |x|$, then $\mathbb{A}'$ uses at most $g(\kappa(x)) \cdot \log |x|$ many coins. But then (recall that c is nondecreasing)

$$g(\kappa(x)) \leq c \circ \iota_c^+ \circ g \circ \kappa(x) \leq c \circ f \circ \kappa(x) \leq c(|x|).$$

The first inequality follows from $c \circ \iota_c^+ \geq \mathrm{id}_{\mathbb{N}}$ by Lemma 5.9, the second from $f \geq \iota_c^+ \circ g$ by (13), and the third from $f(\kappa(x)) \leq |x|$. $\dashv$

By Claim 2 and our assumption we get $Q_{\geq f} \in \mathrm{P}$. We get the following algorithm solving Q: on x it first checks in polynomial time if $f(\kappa(x)) > |x|$. If this is the case it simulates a decision procedure for Q running in time (recall that r is nondecreasing)

$$r(|x|) \cdot |x|^{O(1)} \leq r(f(\kappa(x))) \cdot |x|^{O(1)}.$$

Otherwise it runs a polynomial time procedure deciding $Q_{\geq f}$. This shows (14). $\Box$

Example 5.10 (PIT, continued)**.** *If* $\mathrm{RP}[c] = \mathrm{P}$ *for some polynomial time computable, nondecreasing, unbounded* $c : \mathbb{N} \to \mathbb{N}$*, then var*-PIT[terms] *is fixed-parameter tractable and decidable in time*

$$2^{o(k \log n)} \cdot n^{O(1)},$$

on instances of length n with parameter k.

Sketch of Proof: The first statement follows by (the version for one-sided error of) Theorem 5.6 and Example 4.6.

The statement on the time-bound follows from miniaturization theory [26, 24, 20]: *var*-PIT[terms] is length-condensable by Remark 3.6. Then *var*-PIT[terms] is fpt equivalent to the miniaturization of its reparameterization by $k \log n$ [20, Theorem 10]. Hence this miniaturization is fixed-parameter tractable too. Now the claim follows from the miniaturization theorem (see e.g. [37, Theorem 16.30]). $\Box$

6. Upper bounds

6.1. **Introduction.** The Sipser-Gács Theorem [62, Theorem 6] states that BPP is contained in $\Sigma_2^{\mathrm{P}} \cap \Pi_2^{\mathrm{P}}$. We ask for parameterized analogues of this result. Therefore, we have to choose parameterized analogues of the polynomial hierarchy.

Recall that W[1]-randomization is motivated by the choice of W[1] as parameterized analogue of NP. Since W[1] is characterized by restricted, tail-nondeterministic, *existential* machines (Theorem 2.5), the A-hierarchy is an analogue of the polynomial hierarchy. W[P]-randomization is motivated by the choice of W[P] as analogue of NP. The class W[P] is characterized by restricted existential machines (Theorem 2.5), i.e. by machines for W[1] without the restriction of tail-nondeterminism. Hence we get as analogue of the polynomial hierarchy the classes given by machines for the A-hierarchy without the restriction to tail-nondeterminism. These classes have been introduced by Y.Chen [18] via certain weighted alternating satisfiability problems. We define them by their machine characterizations [18, Theorem 4.23]:

Definition 6.1. Let $\ell \geq 1$. The class AWP[ℓ] contains those parameterized problems (Q, κ) that can be decided by κ-restricted, ℓ-alternating programs.

By Theorem 2.5 we immediately get [18, Corollary 4.24]:

Proposition 6.2. A[ℓ] $\subseteq$ AWP[ℓ] *for all* $\ell \geq 1$ *and* AWP[1] = W[P].

Our analogues of the Sipser-Gács Theorem read as follows.

Theorem 6.3.

(1) BPFPT $\subseteq$ AWP[2] $\cap$ coAWP[2].
(2) BPFPT[1] $\subseteq$ A[2] $\cap$ coA[2].

The first statement can be inferred from a quite general parameterized analogue of the Sipser-Gács Theorem from [52], that holds for all parameterized classes that are closed under certain forms of parameterized reductions. Here we give a relatively simple direct argument. An advantage is that this argument also works for W[1]-randomization and thereby proves the new second statement.

The Sipser-Gács Theorem implies that BPP = P if P = NP. We get an analogue:

Corollary 6.4. *If* W[P] = FPT*, then* BPFPT = FPT*.*

Proof. It is not hard to see that the AWP-hierarchy satisfies a collapse theorem [18, Corollary 4.25], i.e. AWP[ℓ] = FPT for all $\ell \geq 1$ in case W[P] = FPT. By Theorem 6.3 this implies the corollary. □

6.2. **Proof of Theorem 6.3.** We first prove (1). Let $(Q, \kappa) \in$ BPFPT and let $\mathbb{P}$ be a program witnessing this. By the Small Dice Lemma 3.17 we can assume that $\mathbb{P}$ on $x \in \Sigma^*$ uses $g(\kappa(x))$ many $|x|$-sided dice for some computable g.

Let $x \in \Sigma^*$ and write $k := \kappa(x)$ and $n := |x|$. Consider a random seed of $\mathbb{P}$ as an element of the group $\mathbb{Z}_n^{g(k)}$: by this we mean the group on tuples $[0, n-1]^{g(k)}$ with componentwise addition modulo n. For the *good seeds*

$$G_x := \{\mathbb{P}(x) = Q(x)\}$$

we know $\Pr[G_x] > 1 - 1/n$. In a first step we show by the probabilistic method that 'a few' translates of this set cover the whole sample space.

For $\ell \in \mathbb{N}$ to be specified later, let $X_1, \ldots, X_\ell$ be mutually independent random variables uniformly distributed in $\mathbb{Z}_n^{g(k)}$.

Fix some $\bar{r} \in \mathbb{Z}_n^{g(k)}$. Note that the function $\bar{x} \mapsto \bar{r} - \bar{x} \bmod n$ is a permutation of $\mathbb{Z}_n^{g(k)}$ (being a group). Thus also $(\bar{r} - X_1) \bmod n, \ldots, (\bar{r} - X_\ell) \bmod n$ are mutually independent and uniformly distributed in $\mathbb{Z}_n^{g(k)}$. Hence

$$\Pr\Big[\bar{r} \notin \bigcup_{i \in [\ell]} (G_x + X_i) \bmod n\Big] = \prod_{i \in [\ell]} \Pr\big[(\bar{r} - X_i) \bmod n \notin G_x\big] < n^{-\ell}.$$

Here, $(G_x + \bar{r}) \bmod n$ denotes $\big\{(\bar{a} + \bar{r}) \bmod n \mid \bar{a} \in G_x\big\}$.

By the union bound

$$\begin{aligned} \Pr\Big[\mathbb{Z}_n^{g(k)} = \bigcup_{i \in [\ell]} (G_x + X_i) \bmod n\Big] &\geq 1 - \sum_{\bar{r} \in \mathbb{Z}_n^{g(k)}} \Pr\Big[\bar{r} \notin \bigcup_{i \in [\ell]} (G_x + X_i) \bmod n\Big] \\ &> 1 - n^{g(k) - \ell}. \end{aligned}$$

This probability is positive for

$$\ell := g(k) + 1.$$

Hence there are $\bar{r}_1, \ldots, \bar{r}_\ell \in \mathbb{Z}_n^{g(k)}$ such that

$$\mathbb{Z}_n^{g(k)} = \bigcup_{i \in [\ell]} (G_x + \bar{r}_i) \bmod n \ .$$

Next we show that the sample space can be covered by 'few' translates of seeds causing $\mathbb{P}$ to accept if and only if $x \in Q$.

If $x \in Q$ then $G_x = \big\{\mathbb{P}(x) \text{ accepts}\big\}$ and there are $\bar{r}_1, \ldots, \bar{r}_\ell \in \mathbb{Z}_n^{g(k)}$ such that for all $\bar{r}$ there is an $i \in [\ell]$ such that $\mathbb{P}(x)((\bar{r} - \bar{r}_i) \bmod n)$ accepts.

Conversely, assume $x \notin Q$ and let $\bar{r}_1, \dots, \bar{r}_\ell \in \mathbb{Z}_n^{g(k)}$ be arbitrary. Then the set $\{\mathbb{P}(x) \text{ accepts}\} = \mathbb{Z}_n^{g(k)} \setminus G_x$ has cardinality at most $1/n \cdot n^{g(k)} = n^{g(k)-1}$. Thus

$$\bigcup_{i\in[\ell]} \big(\{\mathbb{P}(x) \text{ accepts}\} + \bar{r}_i\big) \bmod n \le \ell \cdot n^{g(k)-1} < n^{g(k)},$$

where we assume without loss of generality that $n > \ell = g(k)+1$. Hence there exists an $\bar{r} \in \mathbb{Z}_n^{g(k)} \setminus \bigcup_{i\in[\ell]} \big(\{\mathbb{P}(x) \text{ accepts}\} + \bar{r}_i\big) \bmod n$, that is, an $\bar{r} \in \mathbb{Z}_n^{g(k)}$ such that $\mathbb{P}(x)((\bar{r} - \bar{r}_i) \bmod n)$ accepts for no $i \in [\ell]$.

In summary:

$$\begin{aligned} x \in Q \iff\ & \exists \bar{r}_1, \dots, \bar{r}_{g(k)+1} \in \mathbb{Z}_n^{g(k)} \ \ \forall \bar{r} \in \mathbb{Z}_n^{g(k)} \ \ \exists i \in [g(k)+1] : \\ & \mathbb{P}(x)((\bar{r} - \bar{r}_i) \bmod n) \text{ accepts.} \end{aligned}$$

It is not hard to see that this condition can be checked by a restricted 2-alternating program $\mathbb{P}'$: on x guess existentially $\bar{r}_1, \dots, \bar{r}_{g(k)+1} \in \mathbb{Z}_n^{g(k)}$, then guess universally $\bar{r} \in \mathbb{Z}_n^{g(k)}$ and simulate $\mathbb{P}$ with random seed $\bar{r} - \bar{r}_i$ for all $i \in [g(k)+1]$; accept if at least one of these simulations is accepting. This shows $(Q, \kappa) \in$ AWP[2].

Thus BPFPT $\subseteq$ AWP[2] and thus also BPFPT $\subseteq$ coAWP[2] since BPFPT is closed under complementation (i.e. $(\Sigma^* \setminus Q, \kappa) \in$ BPFPT if $(Q, \kappa) \in$ BPFPT).

To prove (2) simply note that the program $\mathbb{P}'$ above can be implemented tail-nondeterministically in case $\mathbb{P}$ is tail-nondeterministic. □

7. Probably almost correct counting

7.1. **Introduction.** It is common to consider besides decision problems also search, listing or counting problems. For, in a certain sense, self-reducible problems the decision, search and listing versions all have the same complexity[3]. In contrast, counting versions may be harder: e.g. counting satisfying assignments of DNFs is #P-hard (it is 'the same as' doing so for CNFs), while deciding satisfiability of DNFs is trivial. A famous and more surprising example (see [65, 45]) is Valiant's theorem stating that #PMatch, the problem to count perfect matchings in a given bipartite graph, is #P-complete under polynomial time Turing reductions. Deciding if there is such a matching is tractable and, by self-reducibility, so are the associated search and listing problems. Twelve years later S.Toda reveiled [64, 59] the surprising power of counting. He showed how to solve any problem in PH in polynomial time with a single query to a #P-oracle.

This apparent intractability of (exact) counting suggests the quest for feasible approximations, e.g. in the sense of a fully polynomial time randomized approximation scheme (*fpras*). Randomized approximation turns out to be related to almost uniform sampling: here again self-reducibility implies equitractability [40, 45], a fact opening the door to the rich theory of Markov chains. This finally enabled Jerrum, Sinclair and Vigoda to construct an fpras for #PMatch (cf. [45]).

Thus some concrete hard counting problems have fast randomized approximations. In general, the complexity of randomized almost correct counting is much lower than that of exact counting. While by Toda's theorem the latter is at least as hard as PH, the former is at most as hard as NP [63]:

[3]Concepts of tractability for listing problems have been introduced in [46].

Theorem 7.1 (Stockmeyer 1985)**.** *Any counting problem in* #P *has an fpras using an* NP *oracle.*

How about parameterized analogues of these results? Again it is easy to see that there are fixed-parameter tractable decision problems with hard associated counting versions, even quite natural ones: e.g. the parameterized weighted satisfiability problem for *positive* 2DNFs is fixed-parameter tractable. Its counting version is 'the same' as that for *negative* 2CNFs, a problem that is #W[1]-complete under parsimonious fpt reductions (see e.g. [37, Theorem 14.18]).

A more surprising example is p-Cycle: decide if a given graph contains a cycle of length k (the parameter is k). This problem is fixed-parameter tractable, but J.Flum and M.Grohe proved [35] that its counting version is #W[1]-complete under fpt Turing reductions. So as in the classical setting we are faced with natural tractable parameterized decision problems having an intractable counting version. The quest is again to find fast randomized approximations.

Notation: For reals r and $\epsilon > 0$ we write $(1 \pm \epsilon) \cdot r$ for the open real interval $(r - \epsilon \cdot r, r + \epsilon \cdot r)$.

Definition 7.2. An *fptras (fixed-parameter tractable randomized approximation scheme)* for a parameterized counting problem (F, κ) is a randomized fpt algorithm $\mathbb{P}$ expecting inputs (x, ℓ, ℓ') for $x \in \Sigma^*$ and positive $\ell, \ell' \in \mathbb{N}$ (in unary) such that

$$\Pr\left[\mathbb{P}(x, \ell, \ell') \in (1 \pm 1/\ell) \cdot F(x)\right] > 1 - 1/\ell'.$$

Here, fpt is understood with respect to the parameterization mapping (x, ℓ, ℓ') to $\kappa(x)$. If $\mathbb{P}$ is W[P]-randomized (with respect to the parameterization above), then we call it a W[P]*-fptras.*

In [5] V.Arvind and V.Raman construct an fptras for p-#Cycle. In fact, they construct an fptras for counting embeddings of structures coming from any polynomial time decidable class of structures of bounded tree-width.

This section proves the following parameterized analogues of Theorem 7.1:

Theorem 7.3.

(1) *Let* $\ell, t \geq 1$. *Each parameterized counting problem in* #A$[\ell, t]$ *has a* W[P]*-fptras using a balanced oracle for* A$[\ell, t]$.
(2) *Each parameterized counting problem in* #W[SAT] *has a* W[P]*-fptras using a balanced oracle for* W[SAT].
(3) *Each parameterized counting problem in* #W[P] *has a* W[P]*-fptras using a balanced oracle for* W[P].

Recall, that an oracle is *balanced* if the algorithm asks only queries whose parameter is effectively bounded in the input parameter (cf. Section 2).

The third statement above has already been shown in [53, 52]. In [52] the first author discusses certain gap problems associated to #W[P] approximate counting problems and shows they are contained in AWP[2] (cf. Definition 6.1). All three statements of Theorem 7.3 follow from the following 'logical' version of Stockmeyer's Theorem 7.1.

Definition 7.4. A class Φ of LFP-formulas is *robust* if it is polynomial time decidable, closed under relativizations, closed under conjunctions with quantifier free first-order formulas (of arbitrary vocabulary) and closed under substitution of free variables by parameters.

Here by "closure under relativizations" we mean: if $\varphi \in \Phi$ and P is a unary relation symbol then also $\varphi^P \in \Phi$ where φ^P is the relativization of φ to P, i.e. the formula obtained from φ by replacing all quantifiers $\exists x \dots$ and $\forall x \dots$ in φ by $\exists x(Px \wedge \dots)$ and $\forall x(Px \to \dots)$ respectively.

The main result of this section is the following. Recall the notation for model-checking problems from Section 2.5.

Theorem 7.5. *Let Φ be a robust class of* LFP*-formulas. Then the parameterized counting problem var-*$\#\mathrm{MC}(\Phi)$ *(p-*$\#\mathrm{MC}(\Phi)$*) has a* W[P]*-fptras using a balanced oracle for the parameterized decision problem var-*$\mathrm{MC}(\Phi)$ *(p-*$\mathrm{MC}(\Phi)$*).*

7.2. **Affine hashing.** The technical trick we use is to view the set of solutions as a subset of a vector space of parameter bounded dimension k over some large finite field $\mathbb{F}_p$ (recall the notation from page 16). We use a suitable hash family on this encoding. The following is particularly simple:

Definition 7.6. Let $p \in \mathbb{N}$ be prime and let $i, k \in \mathbb{N}$ be positive. Let $H^p_{k,i}$ denote the set of all affine transformations h from $\mathbb{F}_p^k$ to $\mathbb{F}_p^i$, i.e. mappings of the form $\bar{x} \mapsto M\bar{x} + \bar{b}$ for $\bar{x} \in \mathbb{F}_p^k$, where M is an $i \times k$-matrix with entries in $\mathbb{F}_p$, and $\bar{b} \in \mathbb{F}_p^i$. Set $h^{-1}(\bar{a}) := \{\bar{x} \in \mathbb{F}_p^k \mid h(\bar{x}) = \bar{a}\}$ for $\bar{a} \in \mathbb{F}_p^i$.

The following two lemmas are well-known (see e.g. [40, Lecture 4]).

Lemma 7.7. *$H^p_{k,i}$ is a* 2-universal family of hash functions from $\mathbb{F}_p^k$ to $\mathbb{F}_p^i$*, that is, for all distinct $\bar{x}, \bar{y} \in \mathbb{F}_p^k$ and all $\bar{a}_0, \bar{b}_0 \in \mathbb{F}_p^i$:*

$$\Pr_{h \in H^p_{k,i}} \big[h(\bar{x}) = \bar{a}_0 \ , \ h(\bar{y}) = \bar{b}_0\big] = 1/p^{2i}.$$

Here, $\Pr_{h \in H^p_{k,i}}$ refers to the uniform measure on $H^p_{k,i}$.

Lemma 7.8 (Hashing Lemma)**.** *Let p be prime and $k \geq 1$. For $S \subseteq \mathbb{F}_p^k$ and $i \in [k]$ define*

$$Y_i^S : H^p_{k,i} \to \mathbb{N} : h \mapsto |S \cap h^{-1}(0^i)|,$$

where 0^i is the zero vector in $\mathbb{F}_p^i$. Then for all $\epsilon > 0$ we have for $\rho_i := |S|/p^i$

$$\Pr_{h \in H^p_{k,i}} \big[\,\big|Y_i^S - \rho_i\big| \geq \epsilon\rho_i\big] < \frac{1}{\epsilon^2 \rho_i}.$$

7.3. **Proofs.** Theorem 7.5 implies Theorem 7.3 via the following trivial lemma:

Lemma 7.9. *Let (F, κ) and (F', κ') be parameterized counting problems such that (F, κ) is fpt reducible to (F', κ'). If (F', κ') has a* W[P]*-fptras, then so does (F, κ).*

Proof of Theorem 7.3 from Theorem 7.5: For each class mentioned in Theorem 7.3 there is a set Φ of LFP-formulas such that the model-checking problem $(\mathrm{MC}(\Phi), \kappa)$ is complete for the class where κ is either the parameterization by the length of the input formula or by its number of variables. Furthermore, the counting version $(\#\mathrm{MC}(\Phi), \kappa)$ of the problem is complete for the counting version of the class.

By Theorem 7.5 there is a W[P]-fptras for $(\#\mathrm{MC}(\Phi), \kappa)$ using a balanced oracle for $(\mathrm{MC}(\Phi), \kappa)$. Then there is an W[P]-fptras for all problems in the counting version of the class by Lemma 7.9 (clearly we have this lemma also in the presence of oracles).

This argument needs a correction: strictly speaking, the classes Φ considered are not robust. However, they are robust 'up to polynomial time transformations': e.g. Π_{17} is not closed under relativizations, but any relativization of a formula from Π_{17} can easily be transformed in polynomial time to an equivalent formula in Π_{17}. Of course, this is good enough. $\square$

To prove Theorem 7.5 the following simple characterization of W[P]-fptrases turns out useful.

Definition 7.10. A parameterized counting problem (F, κ) is *fpt paddable* if there is a function $r : \Sigma^* \times \mathbb{N} \to \Sigma^*$, fpt computable (in unary) with respect to the parameterization mapping (x, ℓ) to $\kappa(x)$ such that for some computable $g : \mathbb{N} \to \mathbb{N}$

(1) $F(r(x, \ell)) = F(x)$;
(2) $\kappa(r(x, \ell)) \leq g(\kappa(x))$;
(3) $|r(x, \ell)| \geq |x| + \ell$.

Lemma 7.11. *Let let $c, c' \geq 1$ and (F, κ) be an fpt paddable parameterized counting problem. Then (F, κ) has a* W[P]*-fptras $\mathbb{P}$ if and only if there is a* W[P]*-randomized program $\mathbb{P}'$ such that for all $x \in \Sigma^*$*

$$\Pr\left[\mathbb{P}'(x) \in (1 \pm |x|^{-c}) \cdot F(x)\right] > 1 - |x|^{-c'}.$$

Proof. To see sufficiency define $\mathbb{P}$ on (x, ℓ, ℓ') to simulate the given $\mathbb{P}'$ on x in case $|x| \geq \max\{\ell, \ell'\}$; otherwise run $\mathbb{P}'$ on $r\big(x, \max\{\ell, \ell'\} - |x|\big)$ for r witnessing fpt paddability of (F, κ).

To see necessity define $\mathbb{P}'$ on x to simulate the given $\mathbb{P}$ on $(x, |x|^c, |x|^{c'})$. $\square$

Proof of Theorem 7.5. The approximation scheme specified below has a similar high-level description as the one constructed in the proof for Theorem 7.1 as given in [40]; however, the parameterized setting calls for new subroutines to feasibly implement the scheme.

Let Φ be as stated. We present the proof for *var*-MC(Φ); the 'same' argument works for *p*-MC(Φ).

We restrict attention to instances $(\mathcal{A}, \varphi)$ of *var*-MC(Φ) which have size bounded by $|A|^2$: if an instance is not of this form we enlarge the universe A of $\mathcal{A}$ by sufficiently many new elements, colour the old universe black, relativize the formula to a new predicate symbol for blackness and add to φ conjuncts "x is black" for all variables x free in φ. The new formula is again from Φ. Note that this also shows that *var*-#MC(Φ) is fpt paddable.

Let $\bar{x} = x_1 \cdots x_k$ be the free variables of φ. We assume $A = \{0, \ldots, n-1\}$ for some $n \in \mathbb{N}$ with $n \geq k$. In particular, for a prime $p > n$ we have $\varphi(\mathcal{A}) \subseteq \mathbb{F}_p^k$. We write $S := \varphi(\mathcal{A})$ and use the notation from the Hashing Lemma 7.8. Additionally we set

$$\min \emptyset := 0 \text{ and } Y_0^S \text{ constantly } 0.$$

We describe a program $\mathbb{P}$ approximating $|S|$. Thereby we use a constant $\ell \in \mathbb{N}$ that is to be fixed later:

```
$\mathbb{P}(\mathcal{A}, \varphi)$   //   $\mathcal{A}$ a structure, $\varphi \in \Phi$.
    1. $p \leftarrow$ the smallest prime $> n$.
    2. **if** $|S| < p^\ell$ **then** return $|S|$.
    3. **else for all** $i \in [k]$: guess $h_i \in H^p_{k,i}$.
    4. $j \leftarrow \min\{i \in [k] \mid Y^S_i(h_i) < p^\ell\}$.
    5. return $p^j \cdot Y^S_j(h_j)$.
```

The proof now is given in two parts, a combinatorial part verifying that $\mathbb{P}$ does what it is supposed to do, and an algorithmic part showing how $\mathbb{P}$ can be implemented as a W[P]-randomized program.

Combinatorics. Let p be the prime computed in line *1.* $\mathbb{P}$ gives with probability one the correct answer if $|S| < p^\ell$, so let's assume that $|S| \geq p^\ell$. Choose m such that

$$p^{m-1} < |S| \leq p^m.$$

Since $|S| \geq p^\ell$ and $S \subseteq \mathbb{F}^k_p$ we have $\ell \leq m \leq k$. As we will choose $\ell \geq 2$, we have

$$1 \leq m - \ell + 1 < k \tag{15}$$

We refer to a random seed of $\mathbb{P}$ on x with $(h_1, \ldots, h_k)$, a k-tuple of hash functions guessed by $\mathbb{P}$ in line *3.* Let $j_{\mathbb{P}}(h_1, \ldots, h_k)$ be the value j computed by $\mathbb{P}$ in line *4* in the run determined by $(h_1, \ldots, h_k)$. Note $j_{\mathbb{P}}$ is a random variable with range $\mathbb{N}$ defined on $\prod_{i\in[k]} H^p_{k,i}$ endowed with the uniform probability measure Pr. For $i \in [k]$ let π_i be the projection to the ith component $(h_1, \ldots, h_k) \mapsto h_i$. Because the random variable $Y^S_i \circ \pi_i$ is distributed as Y^S_i we have the Hashing Lemma for $Y^S_i \circ \pi_i$ on the present probability space on $\prod_{i\in[k]} H^p_{k,i}$. For notational simplicity we denote $Y^S_i \circ \pi_i$ again by Y^S_i.

Claim: $\Pr\left[j_{\mathbb{P}} \notin [m - \ell + 1]\right] < n^{-\ell}$.

Proof of the Claim: Recall the notation $\rho_i := |S|/p^i$ and note

$$p^{\ell-1} = p^m/p^{m-\ell+1} \geq \rho_{m-\ell+1} > p^{m-1}/p^{m-\ell+1} = p^{\ell-2}.$$

Hence

$$\begin{aligned}
\{j_{\mathbb{P}} \notin [m - \ell + 1]\} &\subseteq \{Y^S_{m-\ell+1} \geq p \cdot p^{\ell-1}\} \\
&\subseteq \{Y^S_{m-\ell+1} \geq (p-1) \cdot \rho_{m-\ell+1} + \rho_{m-\ell+1}\} \\
&\subseteq \{|Y^S_{m-\ell+1} - \rho_{m-\ell+1}| \geq (p-1) \cdot \rho_{m-\ell+1}\}.
\end{aligned}$$

By (15) we can apply the Hashing Lemma on $Y^S_{m-\ell+1}$ and conclude that

$$\Pr\left[j_{\mathbb{P}} \notin [m - \ell + 1]\right] < \frac{1}{(p-1)^2 \cdot \rho_{m-\ell+1}} < \frac{1}{(p-1)^2 \cdot p^{\ell-2}} < n^{-\ell}.$$

$\dashv$

Our program outputs $p^{j_{\mathbb{P}}} \cdot Y^S_{j_{\mathbb{P}}}$ as approximation of $|S|$, so we are interested in the probability of the events

$$A_\epsilon := \left\{\left|p^{j_{\mathbb{P}}} \cdot Y^S_{j_{\mathbb{P}}} - |S|\right| < \epsilon|S|\right\},$$

for $\epsilon > 0$. Let $\overline{A}_\epsilon$ denote the complement of A_ϵ. Then

$$\overline{A}_\epsilon \cap \{j_{\mathbb{P}} = i\} = \left\{\left|p^i \cdot Y^S_i - |S|\right| \geq \epsilon|S|\right\} = \{|Y^S_i - \rho_i| \geq \epsilon\rho_i\}. \tag{16}$$

For $i \in [m-\ell+1]$ we have $\rho_i > p^{m-1}/p^{m-\ell+1} = p^{\ell-2}$. By the Hashing Lemma we thus get for all $\epsilon > 0$ and all $i \in [m-\ell+1]$

$$\Pr\left[\left|Y_i^S - \rho_i\right| \geq \epsilon\rho_i\right] \leq 1/(\epsilon^2\rho_i) < 1/(\epsilon^2 p^{\ell-2}) < 1/(\epsilon^2 n^{\ell-2}). \tag{17}$$

Thus

$$\begin{aligned}
\Pr\left[\overline{A}_\epsilon\right] &= \Pr\left[\overline{A}_\epsilon \cap \left\{j_{\mathbb{P}} \notin [m-\ell+1]\right\}\right] + \Pr\left[\overline{A}_\epsilon \cap \left\{j_{\mathbb{P}} \in [m-\ell+1]\right\}\right] \\
&< n^{-\ell} + \sum_{i=1}^{m-\ell+1} \Pr\left[\overline{A}_\epsilon \cap \left\{j_{\mathbb{P}} = i\right\}\right] \qquad \text{by the Claim} \\
&< n^{-\ell} + \frac{m-\ell+1}{\epsilon^2 n^{\ell-2}} \qquad \text{by (17) and (16).}
\end{aligned}$$

Now, let $c, c' \geq 1$ be arbitrary. For $\epsilon := n^{-c}$ the above is by (15)

$$\leq n^{-\ell} + (k-1)n^{2c}n^{-\ell+2} < kn^{2c+2-\ell}$$

and hence $< n^{2c+3-\ell}$ as we assumed $k \leq n$. Choosing $\ell := 2c+3+c'$ we get

$$\Pr\left[A_{n^{-c}}\right] = \Pr\left[\mathbb{P}(\mathcal{A},\varphi) \in (1 \pm n^{-c}) \cdot |S|\right] > 1 - n^{-c'}.$$

This suffices by our assumption that the size of $(\mathcal{A},\varphi)$ is $\leq n^2$ and by Lemma 7.11 (we already noted that our problem is fpt paddable).

Algorithmics. We now explain how to implement $\mathbb{P}$ as a W[P]-randomized program. Concerning the random complexity, note that to guess some $h_i \in H^p_{k,i}$ in line *3*, $\mathbb{P}$ guesses the entries of a $i \times k$-matrix and a length i vector over $\mathbb{F}_p$. Hence line *3* can be executed by rolling $\sum_{i=1}^k (ik+i) = O(k^3)$ many p-sided dice.

Concerning the time complexity we have to explain how $\mathbb{P}$ does the jobs we demand it to do in fpt time.

The prime p in line *1* can be computed in polynomial time (since $p < 2n+1$ by Bertrand's Postulate). Since ℓ is a constant, $p^\ell \leq n^{O(1)}$.

In a first step we explain how $\mathbb{P}$ works using a balanced oracle for the following auxiliary parameterized counting problem:

var-BOUNDED-#MC(Φ)

Input:	a structure $\mathcal{B}$, $\psi \in \Phi$ and $r \in \mathbb{N}$ in unary.
Parameter:	the number of variables of ψ.
Problem:	compute $\min\{|\psi(\mathcal{B})|, r\}$.

In a second step we will show how the oracle to *var*-BOUNDED-#MC(Φ) can be replaced by an oracle to *var*-MC(Φ).

Line *2* is directly solved by one balanced oracle call to *var*-BOUNDED-#MC(Φ). The remaining time needed is dominated by the computations of j in line *4* and $Y_i^S(h_i)$ in line *5*. Our program $\mathbb{P}$ has to solve the following problem:

Input:	a structure $\mathcal{A}$, $\varphi \in \Phi, h \in H^p_{k,i}$ with $i \in [k]$ (where p is the smallest prime $> |A|$, φ has k free variables, and i is determined by h).
Parameter:	the number of variables of φ.
Problem:	decide if $Y_i^S(h) < p^\ell$ and if so compute $Y_i^S(h)$ (where $S = \varphi(\mathcal{A})$).

Let $(\mathcal{A}, \varphi, h)$ be an instance of this problem and let $\bar{x} = x_1 \cdots x_k$ comprise the free variables of φ. Furthermore, let h be given by the $i \times k$-matrix M and the i-vector $\bar{b}$ over $\mathbb{F}_p$.

To solve our problem with a balanced oracle for *var*-BOUNDED-#MC(Φ) it suffices to express $Y_i^S(h)$ as $|\varphi^*(\mathcal{A}^*)|$ for a formula $\varphi^* \in \Phi$ and a structure $\mathcal{A}^*$. Moreover, for the oracle to be balanced we need that the number of variables of φ^* is effectively bounded in that of φ.

We move from $\mathcal{A}$ to an arithmetical structure $\mathcal{A}^*$: the universe of $\mathcal{A}^*$ is that of $\mathbb{F}_p$; $\mathcal{A}^*$ interprets all symbols as $\mathcal{A}$ does plus two new ternary relation symbols:

$$\begin{aligned} R_+^{\mathcal{A}^*} &:= \big\{(a,b,c) \in \mathbb{F}_p^3 \mid a+b=c \text{ in } \mathbb{F}_p\big\}, \\ R_\times^{\mathcal{A}^*} &:= \big\{(a,b,c) \in \mathbb{F}_p^3 \mid a \cdot b=c \text{ in } \mathbb{F}_p\big\}. \end{aligned}$$

Further, $\mathcal{A}^*$ interprets a new unary predicate $\dot{A}$ by A. Define the formula

$$\varphi^* := \varphi^{\dot{A}}(\bar{x}) \wedge \bigwedge_{\nu \in [k]} \dot{A}x_\nu \wedge \text{“}h(\bar{x}) = 0^i\text{”}.$$

Here $\varphi^{\dot{A}}$ denotes the relativization of φ to $\dot{A}$ (cf. page 26). We now explain what formula “$h(\bar{x}) = 0^i$” stands for. First, for $\bar{m} \in (A^*)^k$ and $b \in A^*$ we introduce a formula “$\langle \bar{m}, \bar{x} \rangle + b = 0$” (where $\langle \cdot, \cdot \rangle$ stands for the inner product of $\mathbb{F}_p^k$). Write m_μ for the μth component of $\bar{m}$. We use auxiliary variables u_μ intended to stand for $m_\mu \cdot x_\mu$ and v_μ to stand for the intermediate results when computing the sum $\langle \bar{m}, \bar{x} \rangle$. Precisely, “$\langle \bar{m}, \bar{x} \rangle + b = 0$” is the formula

$$\bigwedge_{\mu \in [k]} R_\times m_\mu x_\mu u_\mu \wedge v_1 = u_1 \wedge \bigwedge_{\mu \in [k-1]} R_+ v_\mu u_{\mu+1} v_{\mu+1} \wedge R_+ v_k b 0.$$

The formula is quantifier free, uses $\bar{m}, b$ and 0 as parameters and has variables $\bar{x}$ and auxiliary variables $\bar{u} = u_1 \cdots u_k$ and $\bar{v} = v_1 \cdots v_k$.

Now define “$h(\bar{x}) = 0^i$” to be the conjunction of the formulas “$\langle \bar{m}_\nu, \bar{x} \rangle + b_\nu = 0$” for $\nu \in [i]$, where $\bar{m}_\nu$ denotes the νth row of M and b_ν the νth component of $\bar{b}$. For each $\nu \in [i]$ we use a different set of auxiliary variables.

Then

$$|\varphi^*(\mathcal{A}^*)| = \big|\varphi(\mathcal{A}) \cap h^{-1}(0^i)\big| = Y_i^S(h).$$

Moreover, φ^* has $i \cdot 2k$ more variables than φ and, as Φ is robust, $\varphi^* \in \Phi$.

We are left to show that *var*-BOUNDED-#MC(Φ) is fixed-parameter tractable using a balanced oracle for the decision problem *var*-MC(Φ).

But *var*-MC(Φ) is ‘self-reducible’. It follows by general means that there is an algorithm $\mathbb{A}$ (and we shall define such an algorithm below) using a balanced oracle for *var*-MC(Φ) that solves the associated listing problem with *fpt delay* [43]: $\mathbb{A}$ on an instance $(\mathcal{B}, \psi)$ of *var*-MC(Φ) outputs without repetitions all tuples in $\psi(\mathcal{B})$ such that the delay of $\mathbb{A}$ is fpt bounded; here *delay* means the maximum number of steps until the first output, between any two outputs and from the last output to the halting configuration (see [46]). Granted such an algorithm $\mathbb{A}$, we proceed straightforwardly: let $(\mathcal{B}, \psi, r)$ be an instance of *var*-BOUNDED-#MC(Φ); on $(\mathcal{B}, \psi, r)$ simulate $\mathbb{A}$ on $(\mathcal{B}, \psi)$; increase a counter (initialized by 0) for each output of $\mathbb{A}$; stop the simulation in case the counter becomes r; return the counter. The time needed here is roughly r times the delay of $\mathbb{A}$ and thus fpt bounded.

We describe the algorithm $\mathbb{A}$: on $(\mathcal{B}, \psi)$ it runs $\mathbb{A}'$ on $(\mathcal{B}, \psi, \lambda)$ for the empty tuple λ. Algorithm $\mathbb{A}'$ takes inputs $(\mathcal{B}, \psi, \bar{b})$ for structures $\mathcal{B}$, $\psi \in \Phi$, and $\bar{b}$ a tuple over B. For $b \in B$ let $\psi^b := \psi\frac{b}{x}$ for x the first free variable in ψ (and undefined if ψ is a sentence). $\mathbb{A}'$ on $(\mathcal{B}, \psi, \bar{b})$ is the following recursive procedure:

```
$\mathbb{A}'(\mathcal{B}, \psi, \bar{b})$   //   $\mathcal{B}$ a structure, $\psi \in \Phi$ and $\bar{b}$ a tuple over $B$.
    1.  **if** $\psi(\mathcal{B}) = \emptyset$ **then** stop.
    2.  **else if** $\psi$ has no free variables **then** return $\bar{b}$
    3.          **else for all** $b \in B$ **do**
    4.          return $\mathbb{A}'(\mathcal{B}, \psi^b, \bar{b}b)$
```

(Note that for a sentence ψ we have $\psi(\mathcal{B}) \in \{\emptyset, \{\lambda\}\}$.) Algorithm $\mathbb{A}'$ uses an oracle for *var*-MC(Φ) to check the *if*-condition in line *1*. It is routine to verify that this defines a listing algorithm as desired. □

8. Uniqueness promises

8.1. **A logical form of the Valiant-Vazirani Lemma.** In classical complexity theory the famous Valiant-Vazirani Lemma states that "the problems of distinguishing between instances of SAT having zero or one solution, or finding solutions to instances of SAT having unique solutions, are as hard as SAT itself" ([66, Abstract]). Here hardness refers to randomized reductions with one-sided error. This result can be shaped as a probabilistic statement about Boolean logic:

Theorem 8.1 (Valiant, Vazirani 1985)**.** *There is a polynomial time algorithm computing for any Boolean formula $\alpha(\bar{X})$ a Boolean formula $\beta(\bar{X}\bar{Y})$ such that, if α is satisfiable, then*

$$\Pr_{\bar{b}\in\{0,1\}^{|\bar{Y}|}} \left[\bar{b} \models \exists^{=1}\bar{X} \left(\alpha(\bar{X}) \wedge \beta(\bar{X}\bar{Y})\right)\right] > 1/10 \cdot |\bar{X}|^{-1}.$$

It is easy to see that we can also allow the formula α to come from Boolean logic extended by various quantifiers. In this section we intend to show such results for logics beyond Boolean logic. We show that given a structure $\mathcal{A}$ and a formula φ of least fixed-point logic LFP we can do the following in polynomial time: first, we *enlarge* $\mathcal{A}$ by increasing its universe and declare some additional (mainly arithmetical) relations on it; second, we compute a formula ψ of roughly the same logical complexity as φ such that if you randomly assign values to some distinguished variables of ψ, then in the new structure with 'good' probability ψ singles out exactly one of the solutions of φ in $\mathcal{A}$ provided there are any; furthermore we have one-sided error in the sense that ψ has no solution in case φ has none. To make this precise we employ the following mode of speech:

Definition 8.2. A τ^**-enlargement* of a τ-structure $\mathcal{A}$ is a $(\tau \cup \tau^*)$-structure $\mathcal{A}^*$ which is an expansion of an extension of $\mathcal{A}$ such that there is some unary relation symbol $\dot{A} \in \tau^* \setminus \tau$ with $\dot{A}^{\mathcal{A}^*} = A$.

Then the main result of this section reads:

Theorem 8.3. *Let τ be a vocabulary. There is a relational vocabulary τ^* and a polynomial time algorithm which, given a τ-structure $\mathcal{A}$ and $\varphi = \varphi(\bar{x}) \in$ LFP$[\tau]$,* *computes a τ^*-enlargement $\mathcal{A}^*$ of $\mathcal{A}$ and a* quantifier free *formula $\rho = \rho(\bar{x}\bar{y}\bar{z}) \in$* FO$[\tau^*]$ *with parameters in A^* such that, if $\varphi(\mathcal{A}) \neq \emptyset$, then*

$$\Pr_{\bar{b}\in(A^*)^{|\bar{y}|}}\left[\mathcal{A}^* \models \exists^{=1}\bar{x}\bar{z}\left(\varphi^{\dot{A}}(\bar{x}) \wedge \rho(\bar{x}\bar{y}\bar{z})\right)\frac{\bar{b}}{\bar{y}}\right] > |A^*|^{-2}.$$

Furthermore, the length of ρ is polynomially bounded in $|\bar{x}|$.

Here $\varphi^{\dot{A}}$ is the relativization of φ to $\dot{A}$ (see page 26).

This allows us to move in polynomial time to a 'probably unique' formula without essentially increasing the logical complexity of the input formula. The 'probably unique' formula can be seen as an ordinary LFP-formula where certain designated individual variables are interpreted randomly in A^*. This is the view taken in K.Eickmeyer and M.Grohe's *randomized logics* [31].

The logic LFP does not play an essential role here and could be replaced by certain abstract logics in the sense of [30]. We state the result for LFP for simplicity and because it suffices for the applications we have in mind.

8.2. **Corollaries on uniqueness promises.** Theorem 8.1 has proven useful in many respects. For example it is a main step in Toda's [64] original proof that PH $\subseteq$ BP $\oplus$ P $\subseteq$ P$^{\#\mathrm{P}}$. Because of these applications some work has been done on improving the involved parameters, especially the number of random bits [60, 15].

Another application of Theorem 8.1 concerns the problem

USat	
Input:	a CNF α.
Problem:	does α have exactly one satisfying assignment?

This problem has received considerable attention [66, 6, 14, 13, 42, 12]. It is known to be coNP-hard and contained in the class DP[4]. Moreover it has, in some sense, the same exponential time complexity as Sat [42, 12].

Theorem 8.1 gives a randomized reduction of NP problems to Sat *with uniqueness promise*, i.e. such that output formulas probably either have exactly one solution or no solution at all. This implies that USat is NP-hard under randomized polynomial time reductions with one-sided error. The reductions have an inverse polynomial success probability, and, in fact, USat is complete for DP under such reductions [66, Corollary 5] (see [14] for a discussion).

Corollary 8.4. *If* NP $\not\subseteq$ RP*, then* USat $\notin$ P.

R.G.Downey, M.R.Fellows and K.W.Regan [29] ask for analogues of these results in the parameterized setting. They proved the following. Recall the definition of the problem p-WSat$(\Omega_{t,d})$ from page 4.

Theorem 8.5 (Downey, Fellows, Regan 1998)**.** *Let $t \geq 1$. There is a $d \in \mathbb{N}$ such that for all $(Q, \kappa) \in$ W$[t]$ there is a randomized reduction with one-sided fpt error from (Q, κ) to p-*WSat$(\Omega_{t,d})$ *with uniqueness promise.*

[4]DP is sometimes written D$^{\mathrm{P}}$ and comprises the differences of NP problems; [27, Exercise 16.04] introduces a parameterized 'stratification' of this class.

Informally, such a reduction probably produces instances having either exactly one or no solution. "Probably" refers to an inverse fpt success probability, i.e. $1/(f(k) \cdot n^c)$ for some computable $f : \mathbb{N} \to \mathbb{N}$ and $c \in \mathbb{N}$ (cf. Definitions 8.10 here and 6.1 in [29]). The reductions constructed in [29] use $k \cdot n \cdot \log n$ random bits. They actually, achieve a better error bound of order $1/(f(k) \cdot \log n)$.

With an eye on the goal of finding some parameterized analogue of Toda's theorem Downey et al. [29] asked how to derandomize their reduction to – in the present terminology – W[P]-randomized reductions. The following result answers this question affirmatively for all classes of the A-matrix and more:

Theorem 8.6. *Let (Q, κ) be a parameterized problem. Then*

(1) *if $(Q, \kappa) \in \mathrm{A}[\ell, t]$, then there is a* W[1]*-randomized reduction with one-sided fpt error to p-$\mathrm{MC}(\Sigma_{\ell,t-1,1})$ with uniqueness promise.*
(2) *if $(Q, \kappa) \in \mathrm{W[SAT]}$, then there is a* W[1]*-randomized reduction with one-sided fpt error to var-$\mathrm{MC}(\Sigma_1)$ with uniqueness promise.*
(3) *if $(Q, \kappa) \in \mathrm{W[P]}$, then there is a* W[1]*-randomized reduction with one-sided fpt error to var-$\mathrm{MC}(\Sigma_1\mathrm{LFP}^{[1]})$ with uniqueness promise.*

The main message of this result is, roughly, that hard problems remain hard (in a randomized sense) even with a uniqueness promise. E.g. an fpt algorithm solving

p-CLIQUE	
Input:	a graph G and $k \in \mathbb{N}$.
Parameter:	k.
Problem:	does G contain a k-clique?

with uniqueness promise would be an algorithm which behaves like an fpt algorithm for p-CLIQUE *provided* it is fed with an input (G, k) such that G either has exactly one or no k-clique; on other instances the algorithm may do whatever it wants (cf. Definition 8.11).

A second message of Theorem 8.6 is, roughly, that 'uniqueness-versions' of hard problems are hard too. Specifically, [29] and [28] asked for the complexity of

p-UCLIQUE	
Input:	a graph G and $k \in \mathbb{N}$.
Parameter:	k.
Problem:	does G contain exactly one k-clique?

We get the following parameterized analogue of Corollary 8.4:

Corollary 8.7. *If* $\mathrm{W}[1] \not\subseteq \mathrm{RFPT}$*, then p-CLIQUE with uniqueness promise is not fixed-parameter tractable, and in particular, p-UCLIQUE $\notin$ FPT.*

This is not a special property of the clique problem and we shall prove something more general (Theorems 8.17 and 8.18).

Remark 8.8. As for an an upper bound on the complexity of p-UCLIQUE first observe that W[1] contains the problem p-TWO-CLIQUES that asks, given a graph G and a parameter k, whether G contains at least two k-cliques. Hence this problem is fpt reducible to p-CLIQUE. But p-UCLIQUE is the intersection of p-CLIQUE and the complement of p-TWO-CLIQUES. So two non-adaptive, balanced oracle queries to p-CLIQUE suffice to decide p-UCLIQUE.

Remark 8.9. In [19] Y.Chen and J.Flum apply Theorem 8.3 to show that certain counting problems are not #W[1]-hard unless W[1] $\subseteq$ RFPT.

8.3. **Proof of Theorem 8.3.** Let $\varphi(\bar{x}) \in \mathrm{LFP}[\tau]$ be a formula and $\mathcal{A}$ be a τ-structure. Let $k := |\bar{x}|$. If $|A| = 1$, we take $\rho := y = y$ and as $\mathcal{A}^*$ any τ^*-enlargement of $\mathcal{A}$ with $A^* = A$. The same works in case $k = 0$, since then $\varphi(\mathcal{A})$ either contains nothing or exactly the empty tuple. Hence we can assume $k \geq 1, |A| \geq 2$ and $A = \{0, \ldots, |A|-1\}$. We can further assume that $k \geq 3$ and that $\bar{0}$ (the all 0 k-tuple) does not satisfy φ in $\mathcal{A}$. To see this, note that $\tilde{\varphi}(\bar{x}x_1x_2) := \varphi(\bar{x}) \wedge x_1 = 1 \wedge x_2 = 1$ (where 1 is a parameter) satisfies this assumption and $|\tilde{\varphi}(\mathcal{A})| = |\varphi(\mathcal{A})|$; if $\mathcal{A}^*$ and $\tilde{\rho}(\bar{x}x_1x_2\bar{y}\bar{z})$ satisfy our claim for $\tilde{\varphi}$, then $\mathcal{A}^*$ and $\rho(\bar{x}\bar{y}\bar{z}') := x_1 = 1 \wedge x_2 = 1 \wedge \tilde{\rho}$ with $\bar{z}' := \bar{z}x_1x_2$ satisfy our claim for $\varphi(\bar{x})$.

Construction of $\mathcal{A}^*$. Let p be the smallest prime bigger than $|A|$ and $k + 2$. By Bertrand's Postulate we can compute p in polynomial time. The structure $\mathcal{A}^*$ has universe

$$A^* := \{0, \ldots, k \cdot p - 1\}.$$

We set $\tau^* := \{R_+, R_\times, R_{\mathrm{mod}}, \dot{A}, \leq\}$ for ternary $R_+, R_\times, R_{\mathrm{mod}}$, binary $\leq$ and unary $\dot{A}$. We let $\mathcal{A}^*$ be a τ^*-enlargement of $\mathcal{A}$ with

$$\begin{array}{rcl} \dot{A}^{\mathcal{A}^*} & := & A, \\ R_+^{\mathcal{A}^*} & := & \{(a,b,c) \in (A^*)^3 \mid a + b = c \bmod p\}, \\ R_\times^{\mathcal{A}^*} & := & \{(a,b,c) \in (A^*)^3 \mid a \cdot b = c \bmod p\}, \\ R_{\mathrm{mod}}^{\mathcal{A}^*} & := & \{(a,b,c) \in (A^*)^3 \mid a < c \text{ and } a = b \bmod c\}, \\ \leq^{\mathcal{A}^*} & := & \text{the natural order on } A^*. \end{array}$$

In $\mathcal{A}^*$ the set $\{0, \ldots, p-1\}$ carries the structure of $\mathbb{F}_p$, the field with p elements. The set of solutions $\varphi(\mathcal{A}) \subseteq A^k \subseteq \mathbb{F}_p^k$ lives in the k-dimensional vectorspace over $\mathbb{F}_p$.

The hyperplanes $H_{\bar{a}}$. For a vector $\bar{a} \in \mathbb{F}_p^k \setminus \{\bar{0}\}$ let $\langle \bar{a} \rangle^\perp$ denote the hyperplane of vectors orthogonal to $\bar{a}$. If we translate $\langle \bar{a} \rangle^\perp$ by $\bar{a}$ we get $H_{\bar{a}}$, that is

$$H_{\bar{a}} := \langle \bar{a} \rangle^\perp + \bar{a} = \{\bar{b} + \bar{a} \mid \bar{b} \in \langle \bar{a} \rangle^\perp\}.$$

This is a hyperplane in $\mathbb{F}_p^k$ and hence

$$|H_{\bar{a}}| = p^{k-1}. \tag{18}$$

Let $\bar{a}, \bar{b} \in \mathbb{F}_p^k \setminus \{\bar{0}\}$ be distinct. If $\bar{a}$ and $\bar{b}$ are linearly independent, the hyperplanes $H_{\bar{a}}$ and $H_{\bar{b}}$ are not parallel and so the affine subspace $H_{\bar{a}} \cap H_{\bar{b}}$ has dimension $k-2$. If $\bar{a}$ and $\bar{b}$ are linearly dependent, then $H_{\bar{a}} \cap H_{\bar{b}} = \emptyset$ (it is herefore that we use the translation $H_{\bar{a}}$ instead of $\langle \bar{a} \rangle^\perp$). In particular

$$|H_{\bar{a}} \cap H_{\bar{b}}| \leq p^{k-2}. \tag{19}$$

To prove the classical Valiant-Vazirani Lemma one typically splits the space containing the solutions (there a cartesian power of $\{0,1\}$) randomly in half for a random number of times, and analyzes the probability that exactly one solution remains. We argue similarly. Our solutions are contained in $\mathbb{F}_p^k$. An idea now is to cut down this space to a $1/p$-fraction using random hyperplanes and to do so subsequently for a random number of times. We do not select random hyperplanes but a random number of random vectors from $\mathbb{F}_p^k$ and look at the event that all these are contained in a hyperplane $H_{\bar{a}}$ associated with a solution $\bar{a} \in \varphi(\mathcal{A})$. We then bound from below the probability that the above event holds for exactly one solution. It is here where we need that $\bar{0} \notin \varphi(\mathcal{A})$. Details follow.

Construction of ρ. Intuitively, the following formula wants $\bar{x}$ from A such that the first $u+2$ of $\bar{y}_1, \dots, \bar{y}_{k+2} \in \mathbb{F}_p^k$ are in $H_{\bar{x}}$:

$$\begin{aligned}\rho' \quad := \quad & \bigwedge_{j\in[k]} \dot{A}x_j \wedge \bigwedge_{i\in[k+2]} \text{“}\bar{y}'_i = \bar{y}_i \bmod p\text{”} \wedge R_{\mathrm{mod}} u' u (k+1) \\ & \wedge \bigwedge_{i\in[k+2]} \big(\max\{i-2,0\} \le u' \to \text{“}\bar{y}'_i \in H_{\bar{x}}\text{”}\big).\end{aligned}$$

This formula uses $0, \dots, k+1$ and p as parameters from A^*. Further "$\bar{y}'_i = \bar{y}_i \bmod p$" abbreviates the formula $\bigwedge_{j\in[k]} R_{\mathrm{mod}} y'_{ij} y_{ij} p$, where we write e.g. $\bar{y}_1 = y_{11} \cdots y_{1k}$. We now explain for what formula "$\bar{y} \in H_{\bar{x}}$" stands for.

Observe that, loosely written, $\bar{y} \in H_{\bar{x}}$ if and only if $\sum_j (y_j - x_j) \cdot x_j = 0$. We introduce auxiliary variables for all intermediate results obtained when computing this sum, namely we intend $u_j = y_j - x_j$, the v_j's to denote the products summed and the w_j's to denote the partial sums obtained when adding up the v_j's. Precisely, we let "$\bar{y} \in H_{\bar{x}}$" stand for the formula:

$$\bigwedge_{j\in[k]} R_+ u_j x_j y_j \wedge \bigwedge_{j\in[k]} R_\times u_j x_j v_j \wedge \bigwedge_{j\in[k-1]} R_+ w_j v_j w_{j+1} \wedge w_1 = v_1 \wedge w_k = 0.$$

So "$\bar{y} \in H_{\bar{x}}$" is a formula in the variables $\bar{x}\bar{y}\bar{u}\bar{v}\bar{w}$ and with parameter 0. In ρ' we use for each $i \in [k+2]$ distinct auxiliary variables $\bar{u}_i\bar{v}_i\bar{w}_i$ in "$\bar{y}'_i \in H_{\bar{x}}$", that is, "$\bar{y}'_i \in H_{\bar{x}}$" has free variables $\bar{x}\bar{y}'_i\bar{u}_i\bar{v}_i\bar{w}_i$.

We aim at a formula ρ such that with good probability for a random assignment to the $\bar{y}_i$ and u we have exactly one solution of $(\varphi^{\dot{A}} \wedge \rho)$. So far this cannot work since we can assign whatever we want to the auxiliary variables $\bar{u}_i\bar{v}_i\bar{w}_i$ for i larger than the value of u' plus 2. We set

$$\rho := \rho' \wedge \bigwedge_{i\in[k+2]} (\neg \max\{i-2,0\} \le u' \to \text{“}\bar{u}_i\bar{v}_i\bar{w}_i = \bar{0}\text{”}).$$

In the notation of our claim we let $\bar{y}$ comprise all the $\bar{y}_i$'s and u and we let $\bar{z}$ comprise all primed variables as well as all the auxiliary variables $\bar{u}_i\bar{v}_i\bar{w}_i$.

This completes the construction of $\mathcal{A}^*$ and ρ. It is clear that $\mathcal{A}^*$ and ρ can be computed in polynomial time given $\mathcal{A}$ and φ.

Probability analysis. Assume $\varphi(\mathcal{A}) \neq \emptyset$. Let $b \in A^*$ and $\bar{b}_1, \dots, \bar{b}_{k+2} \in (A^*)^k$ be arbitrary and abbreviate

$$\psi := (\varphi^{\dot{A}} \wedge \rho) \frac{b}{u} \frac{\bar{b}_1 \cdots \bar{b}_{k+2}}{\bar{y}_1 \cdots \bar{y}_{k+2}}.$$

The assignment of $\bar{a}$ to $\bar{x}$ can be extended to an assignment satisfying ψ in $\mathcal{A}^*$ if and only if $\bar{a} \in \varphi(\mathcal{A})$ and $(\bar{b}_i \bmod p) \in H_{\bar{a}}$ for all $i \in [(b \bmod (k+1)) + 2]$. Moreover, in this case there is exactly one such extension. In particular

$$|\psi(\mathcal{A}^*)| = \left|\left\{\bar{a} \in \varphi(\mathcal{A}) \mid \bar{b}_i \bmod p \in H_{\bar{a}} \text{ for all } i \in [(b \bmod (k+1)) + 2]\right\}\right|. \qquad (20)$$

We define a function f by

$$f\left(b\bar{b}_1\cdots\bar{b}_{k+2}\right) := \left|(\varphi^{\dot{A}}\wedge\rho)\frac{b}{u}\frac{\bar{b}_1\cdots\bar{b}_{k+2}}{\bar{y}_1\cdots\bar{y}_{k+2}}(\mathcal{A}^*)\right|.$$

Think of the set of the $b\bar{b}_1\cdots\bar{b}_{k+2}$'s as carrying a probability space with the uniform probability measure. Declare two points $b\bar{b}_1\cdots\bar{b}_{k+2}$ and $b'\bar{b}'_1\cdots\bar{b}'_{k+2}$ of this space to be *equivalent* if $b\equiv b' \bmod (k+1)$ and componentwise $\bar{b}_i\equiv\bar{b}'_i \bmod p$ for all $i\in[k+2]$. Then the event $\{f=1\}$ is a union of such equivalence classes. All equivalence classes have the same size since both p and k divide $|A^*|$ (therefore we chose A^* as we did). Thus, the probability that f is 1 on a random argument $b\bar{b}_1\cdots\bar{b}_{k+2}$ is the same as the probability that f is 1 on an argument chosen uniformly at random from those $b\bar{b}_1\cdots\bar{b}_{k+2}$ with $b<k+1$ and $\bar{b}_i\in\mathbb{F}_p^k$ for all $i\in[k+2]$.

Let $B,B_1,\ldots,B_{k+2}$ be independent random variables such that B is uniformly distributed in $\{0,\ldots,k\}$ and each B_i is uniformly distributed in $\mathbb{F}_p^k$. Then the theorem claims $\Pr\left[f(BB_1\cdots B_{k+2})=1\right]\geq 1/|A^*|^2$.

To prove this, call m *good* if

$$p^m\leq|\varphi(\mathcal{A})|\leq p^{m+1}.$$

Since $1\leq|\varphi(\mathcal{A})|\leq|A|^k\leq p^k$ we have $0\leq m\leq k$ for any good m. Hence $\Pr[B \text{ is good}]$ is at least $1/(k+1)$. If we could find some t such that

$$\Pr[f(mB_1\cdots B_{k+2})=1]>t$$

for all good m, then we would know the following: for at least a $1/(k+1)$ fraction of b's we find at least a t fraction of $\bar{b}_1\cdots\bar{b}_{k+2}$'s such that f is 1; hence $\Pr[f(BB_1\cdots B_{k+2})=1]>t/(k+1)$.

Claim 1: If m is good, then $\Pr\left[f(mB_1\cdots B_{k+2})=1\right]>1/(2p^2)$.

This implies the theorem: $1/((k+1)2p^2)\geq 1/(k^2p^2)=1/|A^*|^2$ (as $k\geq 3$).

Proof of Claim 1: Let m be good. By (20)

$$f(mB_1\cdots B_{k+2})=|\{\bar{a}\in\varphi(\mathcal{A})\mid B_1,\ldots,B_{m+2}\in H_{\bar{a}}\}|.$$

Hence $f(m\bar{b}_1\cdots\bar{b}_{k+2})=1$ if and only if there is a solution $\bar{a}\in\varphi(\mathcal{A})$ such that $H_{\bar{a}}$ contains all $\bar{b}_1,\ldots,\bar{b}_{m+2}$ but there is no other solution with this property. Define for $\bar{a}\in\varphi(\mathcal{A})$ the event

$$A_m(\bar{a}):=\{B_1,\ldots,B_{m+2}\in H_{\bar{a}}\}\cap\bigcap_{\bar{a}'\in\varphi(\mathcal{A})\setminus\{\bar{a}\}}\ \bigcup_{i\in[m+2]}\{B_i\notin H_{\bar{a}'}\}. \tag{21}$$

Then $A_m(\bar{a})\cap A_m(\bar{a}')=\emptyset$ for distinct $\bar{a},\bar{a}'\in\varphi(\mathcal{A})$ and

$$\{f(mB_1\cdots B_{k+2})=1\}=\dot{\bigcup_{\bar{a}\in\varphi(\mathcal{A})}}A_m(\bar{a}). \tag{22}$$

Using the following Claim 2 we get what we want:

$$\Pr\left[f(mB_1\cdots B_{k+2})=1\right]=\sum_{\bar{a}\in\varphi(\mathcal{A})}\Pr[A_m(\bar{a})]>\sum_{\bar{a}\in\varphi(\mathcal{A})}\frac{p-1}{p^{m+3}}\geq\frac{p^m(p-1)}{p^{m+3}}\geq\frac{1}{2p^2}.$$

Here, the equality is due to (22), the first inequality to Claim 2 below and the second inequality to m being good. $\dashv$

We are thus left to verify:

Claim 2: If m is good, then $\Pr[A_m(\bar{a})]>(p-1)/p^{m+3}$ *for all* $\bar{a}\in\varphi(\mathcal{A})$.

Proof of Claim 2: Let m be good, $\bar{a} \in \varphi(\mathcal{A})$ and write $\bar{B} := (B_1, \dots, B_{m+2})$. By (18)

$$\Pr\left[\bar{B} \in H_{\bar{a}}^{m+2}\right] = \prod_{i \in [m+2]} \Pr\left[B_i \in H_{\bar{a}}\right] = 1/p^{m+2}. \tag{23}$$

For any $\bar{a}' \in \varphi(\mathcal{A}) \setminus \{\bar{a}\}$ we have by (19) (note $\bar{a}, \bar{a}' \neq \bar{0}$)

$$\Pr\left[\bar{B} \in (H_{\bar{a}} \cap H_{\bar{a}'})^{m+2}\right] = \prod_{i \in [m+2]} \Pr\left[B_i \in H_{\bar{a}} \cap H_{\bar{a}'}\right] \leq 1/p^{2(m+2)}. \tag{24}$$

By (23) and (24)

$$\Pr\left[\bar{B} \in H_{\bar{a}'}^{m+2} \mid \bar{B} \in H_{\bar{a}}^{m+2}\right] \leq 1/p^{m+2}. \tag{25}$$

Letting $\bar{a}'$ range over $\varphi(\mathcal{A}) \setminus \{\bar{a}\}$ we conclude

$$\begin{aligned}
\Pr\left[A_m(\bar{a})\right] &= \Pr\left[\bar{B} \in H_{\bar{a}}^{m+2}\right] \cdot \Pr\left[\bigcap_{\bar{a}'} \bigcup_{i \in [m+2]} \{B_i \notin H_{\bar{a}'}\} \;\middle|\; \bar{B} \in H_{\bar{a}}^{m+2}\right] \\
&\geq \Pr\left[\bar{B} \in H_{\bar{a}}^{m+2}\right] \cdot \Big(1 - \sum_{\bar{a}'} \Pr\left[\bar{B} \in H_{\bar{a}'}^{m+2} \;\middle|\; \bar{B} \in H_{\bar{a}}^{m+2}\right]\Big) \\
&\geq \frac{1}{p^{m+2}} \cdot \Big(1 - \frac{|\varphi(\mathcal{A})| - 1}{p^{m+2}}\Big) \\
&> \frac{1}{p^{m+2}} \cdot \Big(1 - \frac{p^{m+1}}{p^{m+2}}\Big) = \frac{p-1}{p^{m+3}}.
\end{aligned}$$

The equality follows from definition (21), the first inequality from the union bound, the second inequality from (23) and (25), and the third from m being good. $\square$

8.4. **Proofs of the corollaries.** To prove the results announced in Section 8.2, we need to make two things precise (Definitions 8.10 and 8.11 below), namely, we have to define the randomized reductions to promise problems as mentioned in Theorems 8.5 and 8.6 and we have to define what it means to decide a problem with uniqueness promise in fpt time (as mentioned in Corollary 8.7).

Usually, if (F, κ) is the naturally associated counting version of a decision problem (Q, κ), then $Q = \{F > 0\} (= \{x \in \Sigma^* \mid F(x) > 0\})$. But, of course $\{F > 0\}$ does not determine F, so the modes of speech we are going to introduce are applicable only within a context determining F. This obligation vanishes when talking about decision problems having – by agreement – a certain naturally associated counting problem. This is the case for model-checking problems.

Definition 8.10. Let (Q, κ) be a parameterized decision problem and let (F, κ') be a parameterized counting problem. A (W[1]-, W[P]-)*randomized reduction with one-sided fpt error from (Q, κ) to $(\{F > 0\}, \kappa')$ with uniqueness promise* is a (W[1]-, W[P]-)randomized program $\mathbb{P}$ such that there are computable $f, g : \mathbb{N} \to \mathbb{N}$ and a $c \in \mathbb{N}$ such that for all $x \in \Sigma^*$:

(1) if $x \in Q$, then $\Pr[F(\mathbb{P}(x)) = 1] \geq 1/(f(\kappa(x)) \cdot |x|^c)$;
(2) if $x \notin Q$, then $\Pr[F(\mathbb{P}(x)) = 0] = 1$;
(3) $\Pr[\kappa'(\mathbb{P}(x)) \leq g(\kappa(x))] = 1$.

Definition 8.11. Given a parameterized counting problem (F, κ), we say that $(\{F > 0\}, \kappa)$ *with uniqueness promise is fixed-parameter tractable* if there are an algorithm $\mathbb{A}$ and a computable $f : \mathbb{N} \to \mathbb{N}$ such that for all $x \in \Sigma^*$ with $F(x) \in \{0, 1\}$:

(1) $\mathbb{A}$ runs in time $f(\kappa(x)) \cdot |x|^{O(1)}$;
(2) $\mathbb{A}$ accepts $x \iff x \in \{F > 0\}$.

Remark 8.12. One could equivalently demand $\mathbb{A}$ to be fpt time bounded: equip an algorithm as in Definition 8.11 with a clock; if time runs out the input x does not satisfy $F(x) \in \{0, 1\}$ and we can answer arbitrarily.

Proposition 8.13. *Let (Q, κ) be a parameterized decision problem outside* RFPT *and let (F, κ') be a parameterized counting problem. Assume there is a* W[P]*-randomized reduction with one-sided fpt error from (Q, κ) to $(\{F > 0\}, \kappa')$ with uniqueness promise. Then $(\{F > 0\}, \kappa')$ with uniqueness promise is not fixed-parameter tractable.*

Proof. First apply the reduction from (Q, κ) to $(\{F > 0\}, \kappa')$ and then an fpt time bounded (Remark 8.12) algorithm witnessing that $(\{F > 0\}, \kappa')$ with uniqueness promise is fixed-parameter tractable. This decides (Q, κ) in fpt time with one-sided error at most $1-1/(h(\kappa(x))\cdot|x|^c)$ for some computable $h : \mathbb{N} \to \mathbb{N}$ and some constant $c \in \mathbb{N}$. In particular, Q is decidable. But the error $1 - 1/(h(\kappa(x)) \cdot |x|^c)$ is at most $1-|x|^{-(c+1)}$ on those instances x of length at least $h(\kappa(x))$. On other inputs we can run some decision procedure for Q; this needs time effectively bounded in $\kappa(x)$. We conclude that (Q, κ) can be decided by a W[P]-randomized program with one-sided error $1 - |x|^{-(c+1)}$. Thus $(Q, \kappa) \in$ RFPT by Theorem 4.1 (and Remark 4.2). □

To prove Theorem 8.6 we use the following straightforward lemma:

Lemma 8.14. *Let (F, κ'') be a parameterized counting problem and (Q, κ), (Q', κ') be parameterized decision problems such that (Q, κ) is fpt reducible to (Q', κ'). If there is a* W[1]*-randomized reduction with one-sided fpt error from (Q', κ') to $(\{F > 0\}, \kappa'')$ with uniqueness promise, then also from (Q, κ) to $(\{F > 0\}, \kappa'')$.*

Proof of Theorem 8.6: To prove (1) or (2) or (3) we can, by Lemma 8.14, assume that $(Q, \kappa) = (\mathrm{MC}(\Phi), \kappa)$ where Φ is some suitable class of LFP formulas and and κ is either the parameterization by the length of the input formula or the number of its variables. A program computing the desired reduction works as follows: given an instance $(\mathcal{A}, \varphi(\bar{x}))$ of MC(Φ) first compute deterministically $\mathcal{A}^*$ and $(\varphi^{\dot{A}}(\bar{x}) \wedge \rho(\bar{x}\bar{y}\bar{z}))$ from Theorem 8.3 in polynomial time. Then roll $|\bar{y}|$ many $|A^*|$-sided dice to get $\bar{b} \in (A^*)^{|\bar{y}|}$ and output $\big(\mathcal{A}^*, (\varphi^{\dot{A}} \wedge \rho)\frac{\bar{b}}{\bar{y}}\big)$.

Note $(\varphi^{\dot{A}} \wedge \rho) \in \Phi$, whenever $\varphi \in \Phi$ modulo an easy polynomial time modification. As the length of ρ and hence also $|\bar{y}|$ is polynomially bounded in $|\bar{x}|$, our program is W[1]-randomized and satisfies Definition 8.10 (3). Definition 8.10 (2) is satisfied too, because in case $\varphi(\mathcal{A}) = \emptyset$ we have $(\varphi^{\dot{A}} \wedge \rho)\frac{\bar{b}}{\bar{y}}(\mathcal{A}^*) = \emptyset$ for all $\bar{b}$. Otherwise $|(\varphi^{\dot{A}} \wedge \rho)\frac{\bar{b}}{\bar{y}}(\mathcal{A}^*)| = 1$ with probability at least $1/|A^*|^2$. As $|A^*|$ is polynomially bounded in the input size, Definition 8.10 (1) is satisfied. □

Remark 8.15. The argument given shows more than stated. It proves analogous statements also for the classes of the W*-hierarchy, the classes of the $\mathrm{W}^{\mathrm{func}}$-hierarchy and the classes AW[$*$], AW[SAT] and AW[P]. See [37, Chapter 8] for suitable model-checking characterizations of these classes except AW[P]. For AW[P] see [21, Theorem 33(2)].

The following is easy to see.

Lemma 8.16. *Let (F, κ) and (F', κ') be parameterized counting problems such that (F, κ) is fpt reducible to (F', κ'). If $(\{F' > 0\}, \kappa')$ with uniqueness promise is fixed-parameter tractable, then $(\{F > 0\}, \kappa)$ with uniqueness promise is fixed-parameter tractable.*

Theorem 8.17. *Let $\ell, t \geq 1$ and let (F, κ) be a $\#\mathrm{A}[\ell, t]$-hard parameterized counting problem. Then $(\{F > 0\}, \kappa)$ with uniqueness promise is not fixed-parameter tractable unless* $\mathrm{A}[\ell, t] \subseteq \mathrm{RFPT}$.

Proof. Let $\ell, t \geq 1$ and (F, κ) be as stated. If $(\{F > 0\}, \kappa)$ with uniqueness promise is fixed-parameter tractable, then so is p-$\mathrm{MC}(\Pi_{\ell-1,t-1,1})$ by Lemma 8.16. Then $\mathrm{A}[\ell, t] \subseteq \mathrm{RFPT}$ by Proposition 8.13 and Theorem 8.6. □

Concerning 'uniqueness variants' of decision problems this yields:

Theorem 8.18. *Let $\ell, t \geq 1$ and let (F, κ) be a $\#\mathrm{A}[\ell, t]$-hard parameterized counting problem. Then $(\{F = 1\}, \kappa) \notin \mathrm{FPT}$ unless* $\mathrm{A}[\ell, t] \subseteq \mathrm{RFPT}$.

Proof. By Theorem 8.17 it suffices to note that an fpt algorithm for $(\{F = 1\}, \kappa)$ witnesses that $(\{F > 0\}, \kappa)$ with uniqueness promise is fixed-parameter tractable. □

Since p-#CLIQUE is #W[1]-complete, the two claims of Corollary 8.7 follow from the previous two theorems.

9. Questions

We saw some first steps of parameterized random complexity theory and many question are open. Answers to the following ones may advance more generally our understanding of structural parameterized complexity theory.

Section 3 (Parameterized randomization: basic observations and techniques). The section defined two modes of parameterized randomization corresponding to taking W[1] or W[P] as parameterized analogue of NP. Concerning the characterization of randomized parameterized tractability by randomized kernelizations it would be interesting to know whether randomized kernelizations can beat deterministic ones with respect to kernel size.

Section 4 (Deterministic probability amplification). The section established strong probability amplification for W[P]-randomization. Can you amplify W[1]-randomized two-sided error from $\frac{1}{2} - \frac{1}{|x|}$ to $\frac{1}{4}$ or from $\frac{1}{4}$ to $\frac{1}{|x|}$? We are not aware of amplifications methods that are useful in this respect, so the question may be a starting point to find new such methods.

Section 5 (Parameterized derandomization). The section characterized ("black-box") derandomization of BPFPT in terms of classical derandomization. What does BPFPT[1] = FPT mean in such terms? The question points to the problem that W[1] does not have a nice characterization by Turing machines (see however [11, 29]). Or can you prove BPFPT[1] = FPT unconditionally?

Section 6 (Upper bounds). The section gave analogues of the Sipser-Gács Theorem for both W[P]- and W[1]-randomization. Is there an analogue of Adleman's Theorem, that is, does parameterized randomized tractability imply some form of non-uniform parameterized tractability? Adleman's line of argument breaks at the point where the analogy of the parameterized class XP and the classical class EXP is flawed: there are more than $n^{f(k)}$ instances of size n with parameter k. A structural question seemingly of relevance here is: what kind a problem (Q, κ) has an fpt reduction r to itself such that the range of r is polynomial time decidable and contains at most $n^{f(k)}$ instances of size n with parameter k?

Does W[1] = FPT imply BPFPT[1] = FPT? The structural problem behind this question is the lack of a collapse theorem for the A-hierarchy.

Section 7 (Probably almost correct counting). The section gave analogues of a Theorem of Stockmeyer. Under what conditions do we have these theorems 'problem-wise'? That is: when approximating a counting problem coming from a decision problem (Q, κ), say $(Q, \kappa) \in$ W[1], are we really in need of an oracle for the full class W[1] or could we do with (Q, κ)? Trivially, the answer is yes for W[1]-hard (Q, κ). But the question is interesting for problems in FPT with a #W[1]-hard counting version. An answer would constitute a step towards a theoretical understanding of what it is that makes certain hard counting problems have fast randomized approximations.

Another open question is whether one can derandomize the result of V.Arvind and V.Raman (see page 25) to W[P]-fptrases.

Section 8 (Uniqueness promises). We proved a general 'logical' version of the Valiant-Vazirani Lemma applicable to all classes of the A-matrix. An important question is whether we can achieve a better success probability, say, of at least $1/k$? This question comes from the struggle for a parameterized analogue of Toda's Theorem, repeatedly asked for in [29, 28, 35, 37]. The first author obtained some conditional results [50]. Such an analogue would state that $\mathrm{FPT}^{\#\mathrm{W}[1]}$ contains the A-hierarchy or at least the W-hierarchy, or that $\mathrm{FPT}^{\#\mathrm{W}[\mathrm{P}]}$ contains the AWP-hierarchy (cf. Definition 6.1) or at least the A-hierarchy.

References

[1] M.Ajtai, J.Komlós, and E.Szemerédi, *Deterministic simulation in LOGSPACE*, Proceedings of the 9th annual ACM Symposium on Theory of Computing (STOC'87), pp.132-140, 1987.

[2] M.Alekhnovich and A.A.Razborov, *Resolution is not automatizable unless W[P] is tractable*, Proceedings of the 41th IEEE Symposium on Foundations of Computer Science (FOCS'01), pp- 210-219, 2001.

[3] N.Alon, R.Yuster, and U.Zwick, *Color-coding*, Journal of the ACM 42, pp.844-856,1995.

[4] S.Arora and B.Barak, **Computational Complexity: A Modern Perspective**, Cambridge University Press, 2009.

[5] V.Arvind and V.Raman, *Approximation algorithms for some parameterized counting problems.* In I.Bose and P.Morin, ed., Proceedings of the 13th International Symposium on Algorithms and Computation, LNCS 2518, pp.453-464. Springer, 2002.

[6] A.Blass and Y.Gurevich, *On the unique satisfiability problem*, Information and Computation 55, pp.80-88, 1982.

[7] R.P.Brent, D.J.Kuck and K.Maruyama, *The parallel evaluation of arithmetic expressions without division*, IEEE Transactions on Computers C-22, pp.532-534, 1973.

[8] R.P.Brent, *The parallel evaluation of general arithmetic expressions*, Journal of the ACM 21(2), pp.201-206, 1974.

[9] P.Bremaud, **Markov Chains, Gibbs Fields, Monte Carlo Simulation and Qeues**, Springer Texts in Applied Mathematics 31, 1999.

[10] L.Cai, *Random separation: a new method for solving fixed-cardinality optimization problems*, Proceedings of the 2nd International Workshop on Parameterized and Exact Computation (IWPEC'06), LNCS 4169, pp.239-250, 2006.

[11] L.Cai, J.Chen, R.G.Downey and M.R.Fellows, *On the structure of parameterized problems in NP*, Information and Computation 123, pp.38-49, 1995.

[12] C.Calabro, R.Impagliazzo, V.Kabanets and R.Paturi, *The complexity of unique k-SAT: an isolation lemma for k-CNFs*, Proceedings of the 18th IEEE Conference on Computational Complexity (CCC'03),pp.135-141, 2003.

[13] R.Chang and J.Kadin, *On computing boolean connectives of characteristic functions*, Theory of Computing Systems 28 (3), pp.173-198 Springer, 1995.

[14] R.Chang, J.Kadin, and P.Rohatgi, *On unique satisfiability and the threshold behavior of randomized reductions*, Journal of Computer and System Sciences, 50(3), pp.359–373, 1995.

[15] S.Chari, P.Rohatgi and A.Srinivasan, *Randomness-optimal unique element isolation, with applications to perfect matching and related problems*, Proceedings of the twenty-fifth ACM Symposium on Theory of Computing (STOC'93), pp.458-467, 1993.

[16] J.Chen, S.Lu, S.-H.Sze and F.Zhang, *Improved algorithms for path, matching and packing problems*, Proceedings of the 18th ACM-SIAM Symposium on Discrete Algorithms (SODA'07), pp.298-307, 2007.

[17] J.Chen, *Randomized disposal of unknowns and implicitly enforced bounds on parameters*, Proceedings of the 3rd International Workshop on Parameterized and Exact Computation (IWPEC'08) in Victoria, LNCS 5018, pp.1-8, 2008.

[18] Y.Chen, **Model-Checking Problems, Machines and Parameterized Complexity**, Dissertation, Albert-Ludwigs-Universität Freiburg i.Br., 2004.

[19] Y.Chen and J.Flum, *The parameterized complexity of maximality and minimality problems*, Annals of Pure and Applied Logic 151(1), pp.22-61, 2008.

[20] Y. Chen and J. Flum, *Subexponential time and fixed-parameter tractability: exploiting the miniaturization mapping*, In: Proceedings of the 21st International Workshop on Computer Science Logic (CSL'07), LNCS 4646, pp.389-404, 2007.

[21] Y.Chen, J.Flum and M.Grohe, *Bounded nondeterminism and alternation in parameterized complexity theory*, In Proceedings of the 18th IEEE Conference on Computational Complexity (CCC'03), pp.13-29, 2003.

[22] Y.Chen, J.Flum and M.Grohe, *Machine-based methods in parameterized complexity theory*, Theoretical Computer Science 339, pp.167-199, 2005.

[23] Y.Chen, J.Flum and M.Müller, *Lower bounds for kernelizations and other preprocessing procedures*, Theory of Computing Systems, 48(4):803-839, 2011.

[24] Y.Chen and M.Grohe, *An isomorphism between subexponential and parameterized complexity theory*, SIAM Journal on Computing, 37(4), pp.1228-1258, 2007.

[25] S.A.Cook and R.A.Reckhoff, *Time bounded random access machines*, Journal of Computer and System Sciences 7, pp.354-375, 1973.

[26] R.G.Downey, V.Estivill-Castro, M.R.Fellows, E.Prietoc and F.A.Rosamond, *Cutting up is hard to do: the parameterised complexity of k-cut and related problems*, in J.Harland (ed.), Proceeding of the Australasian Theory Symposium (CATS'03), Electronic Notes in Theoretical Computer Science 78, pp.209-222, 2003.

[27] R.G.Downey and M.R.Fellows, **Parameterized Complexity**, Springer, 1999.

[28] R.G.Downey and M.R.Fellows, *Parameterized complexity after almost ten years: review and open questions*, Proceedings of Combinatorics, Computation and Logic, DMTCS'99 and CATS'99, Australian Computer Science Communications 21, Springer, pp.1-33, 1999.

[29] R.G.Downey, M.R.Fellows and K.W.Regan, *Parameterized Circuit Complexity and the W Hierarchy*, Theoretical Computer Science 191(1-2), pp.97-115, 1998.

[30] H.D.Ebbinghaus, *Extended Logics: The general framework*, in: Model-Theoretical Logics, ed. J.Barwise, S.Feferman, Springer-Verlag, pp.25-76, 1985.

[31] K.Eickmeyer and M.Grohe, *Randomisation and derandomisation in descriptive complexity theory*, Proceedings of the 24th International Workshop Computer Science Logic (CSL'10), 2010.

[32] K.Eickmeyer, M.Grohe and M.Grübner, *Approximisation of W[P]-complete minimisation problems is hard*, Proceedings 23rd IEEE Conference on Computational Complexity (CCC'08), pp.8-18, 2008.

[33] M.R.Fellows and N.Koblitz, *Fixed-parameter complexity and cryptography*, Proceedings of the 10th International Symposium on Applied Algebra, Algebraic Algorithms and Error-Correcting Codes (AAECC'93), LNCS 673, pp.121-131, Springer, 1993.

[34] J.Flum and M.Grohe, *Fixed-parameter tractability, definability, and model checking*, SIAM Journal on Computing 31, pp.113-145, 2001.

[35] J.Flum and M.Grohe, *The parameterized complexity of counting problems*, SIAM Journal on Computing 33(4), pp.892-922, 2004.

[36] J.Flum and M.Grohe, *Model-checking problems as a basis for parameterized intractability*, Logical Methods in Computer Science 1(1), 2005. Conference version in Proceedings of the 19th IEEE Symposium on Logic in Computer Science (LICS'04), pp.388-397, 2004.

[37] J.Flum and M.Grohe, **Parameterized Complexity Theory**, Springer, 2006.

[38] J.Flum, M.Grohe and M.Weyer, *Bounded fixed-parameter tractability and* $\log^2 n$ *nondeterministic bits*, Journal of Computer and System Sciences 72, pp.34-71, 2006.

[39] O.Goldreich, *Introduction to complexity theory - Lecture Notes*, 2001; available at `http://www.wisdom.weizmann.ac.il/~oded/homepage.html`.

[40] O.Goldreich, *Randomized methods in computation - Lecture Notes*, 2001; available at `http://www.wisdom.weizmann.ac.il/~oded/homepage.html`.

[41] O.Goldreich, **Computational Complexity: A Conceptual Perspective**, Cambridge University Press, 2008.

[42] E.Grandjean and H.Kleine-Büning, *SAT-problems and reductions with respect to the number of variables*, Journal of Logic and Computation 7(4), pp.457-471, 1997.

[43] M.Grohe, *The Complexity of generalized model-checking problems*, unpublished manuscript from 2001.

[44] S.Hoory, N.Linial and A.Wigderson, *Expander graphs and their applications*, Bulletin of the American Mathematical Society 43(4), pp.439-561, 2006.

[45] M.Jerrum, **Counting, Sampling and Intergrating: Algorithms and Complexity**, Birkhäuser, 2003.

[46] D.S.Johnson, C.H.Papadimitriou and M.Yannakakis, *On generating all maximal independent sets*, Information Processing Letters 27, pp.119-123, 1988.

[47] Y.Liu, S.Lu, J.Chen and S.-H.Sze, *Greedy localization and color-coding: improved matching and packing algorithms*, Proceedings of the 2nd International Workshop on Parameterized and Exact Computation (IWPEC'06), LNCS 4169, pp.84-95, 2006.

[48] M.Luby and A.Wigderson, *Pairwise independence and derandomization*, Foundation and Trends in Theoretical Computer Science 1(4), pp.237-301, 2005.

[49] D.Marx, *Parameterized complexity of constraint satisfaction problems*, Computational Complexity 14(2), pp.153-183, 2005.

[50] J.A.Montoya, **On Parameterized Counting**, Dissertation, Albert-Ludwigs-Universität Freiburg i.Br, 2008.

[51] J.A.Montoya, *The Parameterized Complexity of Probability Amplification*, Information Processing Letters 109(1), pp.46-53, 2008.

[52] J.A.Montoya, *On the parameterized complexity of approximate counting*, RAIRO - Theoretical Informatics and Applications, doi:10.1051/ita/2011007.

[53] M.Müller, *Randomized approximations of parameterized counting problems*, Proceedings of the 2nd International Workshop on Parameterized and Exact Computation (IWPEC'06), LNCS 4169, pp.50-59, 2006.

[54] M.Müller, *Parameterized derandomization*, Proceedings of the 3rd International Workshop on Parameterized and Exact Computation (IWPEC'08), LNCS 5018, pp.148-159, 2008.

[55] M.Müller, *Valiant-Vazirani lemmata for various logics*, Electronic Colloquium on Computational Complexity (ECCC'08), Report TR08-063, 2008; available at `http://eccc.hpi-web.de/eccc-reports/2008/TR08-063/index.html`

[56] M.Müller, **Parameterized Randomization**, Dissertation, Albert-Ludwigs-Universität Freiburg i.Br., 2009.

[57] J.Naor and M.Naor, *Small-bias probability spaces: efficient constructions and applications*, SIAM Journal on Computing 22, pp.213-223, 1993.

[58] M.Naor, L.Schulman and A.Srinivasan, *Splitters and near-optimal derandomization*, Proceedings of the 39th IEEE Symposium on Foundations of Computer Science (FOCS'95), pp.182-190, 1995.

[59] M.Ogiwara and S.Toda, *Counting classes are at least as hard as the polynomial-time hierarchy*, SIAM Journal on Computing 21(2), pp.316-328, 1992.

[60] K.W.Regan, *Efficient reductions from NP to parity using error-correcting codes (preliminary version)*, State University of New York at Buffalo, Technial Report 93-24, 1993; available at `http://www.cse.buffalo.edu/tech-reports/`

[61] O.Reingold, S.Vadhan, A.Wigderson, *Entropy waves, the zig-zag graph product, and new constant degree expanders*, Annals of Mathematics, 155, pp.157-187, 2002.

[62] M.Sipser, *A complexity theoretic approach to randomness*, Proceedings of the 15. ACM Symposium on Theory of Computing (STOC'83), pp.330-335, 1983.

[63] L.Stockmeyer, *On approximation algorithms for #P*, SIAM Journal on Computing 14(4), pp.849-861,1985.

[64] S.Toda, *PP is as hard as the polynomial hierarchy*, SIAM Journal on Computing 20(5) , pp.865-877, 1991.

[65] L.G.Valiant, *The complexity of enumeration and reliability problems*, SIAM Journal on Computing 8(3), pp.410-421, 1979.

[66] L.G.Valiant, V.V.Vazirani, *NP is as easy as detecting unique solutions*, Proceedings of the 17th ACM Symposium on Theory of Computing (STOC'85), pp.458-463, 1985.

Escuela de Matemáticas, Universidad industrial de Santander, Bucaramanga, Colombia.
E-mail address: `jmontoya@matematicas.uis.edu.co`

Kurt Gödel Research Center, University of Vienna, Austria.
E-mail address: `moritz.mueller@univie.ac.at`