Parameterised Counting in Logspace

Anselm Haak
Institut für Theoretische Informatik, Leibniz Universität Hannover, Germany

Arne Meier
Institut für Theoretische Informatik, Leibniz Universität Hannover, Germany

Om Prakash
Department of Computer Science and Engineering, IIT Madras, Chennai, India

Raghavendra Rao B. V.
Department of Computer Science and Engineering, IIT Madras, Chennai, India

Abstract

Logarithmic space bounded complexity classes such as **L** and **NL** play a central role in space bounded computation. The study of counting versions of these complexity classes have lead to several interesting insights into the structure of computational problems such as computing the determinant and counting paths in directed acyclic graphs. Though parameterised complexity theory was initiated roughly three decades ago by Downey and Fellows, a satisfactory study of parameterised logarithmic space bounded computation was developed only in the last decade by Elberfeld, Stockhusen and Tantau (IPEC 2013, Algorithmica 2015).

In this paper, we introduce a new framework for parameterised counting in logspace, inspired by the parameterised space bounded models developed by Elberfeld, Stockhusen and Tantau (IPEC 2013, Algorithmica 2015). They defined the operators $\mathbf{para_W}$ and $\mathbf{para}_\beta$ for parameterised space complexity classes by allowing bounded nondeterminism with multiple-read and read-once access, respectively. Using these operators, they characterised the parameterised complexity of natural problems on graphs. In the spirit of the operators $\mathbf{para_W}$ and $\mathbf{para}_\beta$ by Stockhusen and Tantau, we introduce variants based on tail-nondeterminism, $\mathbf{para_{W[1]}}$ and $\mathbf{para}_{\beta\mathbf{tail}}$. Then, we consider counting versions of all four operators applied to logspace and obtain several natural complete problems for the resulting classes: counting of paths in digraphs, counting first-order models for formulas, and counting graph homomorphisms. Furthermore, we show that the complexity of a parameterised variant of the determinant function for $(0,1)$-matrices is $\#\mathbf{para}_{\beta\mathbf{tail}}\mathbf{L}$-hard and can be written as the difference of two functions in $\#\mathbf{para}_{\beta\mathbf{tail}}\mathbf{L}$. These problems exhibit the richness of the introduced counting classes. Our results further indicate interesting structural characteristics of these classes. For example, we show that the closure of $\#\mathbf{para}_{\beta\mathbf{tail}}\mathbf{L}$ under parameterised logspace parsimonious reductions coincides with $\#\mathbf{para}_\beta\mathbf{L}$, that is, modulo parameterised reductions, tail-nondeterminism with read-once access is the same as read-once nondeterminism.

Initiating the study of closure properties of these parameterised logspace counting classes, we show that all introduced classes are closed under addition and multiplication, and those without tail-nondeterminism are closed under parameterised logspace parsimonious reductions.

Also, we show that the counting classes defined can naturally be characterised by parameterised variants of classes based on branching programs in analogy to the classical counting classes.

Finally, we underline the significance of this topic by providing a promising outlook showing several open problems and options for further directions of research.

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

Keywords and phrases Parameterized Complexity, Counting Complexity, Logspace

Digital Object Identifier 10.4230/LIPIcs.STACS.2021.40

Related Version *Technical report at arXiv.*: `https://arxiv.org/abs/1904.12156` [36]

Funding Indo-German co-operation grant: DAAD (57388253), DST (INT/FRG/DAAD/P-19/2018).
Arne Meier: Funded by the German Research Foundation (DFG), project ME4279/1-2.

Acknowledgements The authors thank the anonymous referees for their valuable feedback.

38th International Symposium on Theoretical Aspects of Computer Science (STACS 2021).
Editors: Markus Bläser and Benjamin Monmege; Article No. 40; pp. 40:1–40:17
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

Parameterised complexity theory, introduced by Downey and Fellows [25], takes a two-dimensional view on the computational complexity of problems and has revolutionised the algorithmic world. Two-dimensional here refers to the fact that the complexity of a parameterised problem is analysed with respect to the input size n and a parameter k associated with the given input as two independent quantities. The notion of fixed-parameter tractability is the proposed notion of efficient computation. A problem with parameter k is fixed-parameter tractable (fpt, or in the class **FPT**) if there is a deterministic $f(k) \cdot n^{O(1)}$ time algorithm for deciding it, where f is a computable function. The primary notion of intractability is captured by the **W**-hierarchy in this setting.

Since its inception, the focus of parameterised complexity theory has been to identify parameterisations of **NP**-hard problems that allow for efficient parameterised algorithms, and to address structural aspects of the classes in the **W**-hierarchy and related complexity classes [33]. This led to the development of machine-based and logical characterisations of parameterised complexity classes (see the book by Flum and Grohe [33] for more details). While the structure of classes in hierarchies such as the **W**- and **A**-hierarchy is well understood, a parameterised view of parallel and space-bounded computation lacked attention.

In 2013, Elberfeld et al. [43, 28] focused on parameterised space complexity classes and initiated the study of parameterised circuit complexity classes. In fact, they introduced parameterised analogues of deterministic and nondeterministic logarithmic space-bounded classes. The machine-based characterisation of $\mathbf{W}[\mathbf{P}]$ (the class of problems that are fpt-reducible to a weighted circuit satisfiability question), and the type of access to nondeterministic choices (multi-read or read-once) led to two different variants of parameterised logspace (para-logspace), namely, $\mathbf{para_W L}$ and $\mathbf{para}_\beta\mathbf{L}$. Elberfeld et al. [28] obtained several natural complete problems for these classes, such as parameterised variants of reachability in graphs.

Bannach, Stockhusen and Tantau [6] further studied parameterised parallel algorithms. They used colour coding techniques [4] to obtain efficient parameterised parallel algorithms for several natural problems. A year later, Chen and Flum [15, 16] proved parameterised lower bounds for $\mathbf{AC^0}$ by adapting circuit lower bound techniques.

Apart from decision problems, counting problems have found a prominent place in complexity theory. Valiant [46] introduced the notion of counting complexity classes that capture natural counting problems such as counting the number of perfect matchings in a graph, or counting the number of satisfying assignments of a CNF formula. Informally, #**P** (resp., #**L**) consists of all functions $F\colon \{0,1\}^* \to \mathbb{N}$ such that there exists an nondeterministic Turing machine (NTM) running in polynomial time (resp., logarithmic space) in the input length whose number of accepting paths on every input $x \in \{0,1\}^*$ is equal to $F(x)$. Valiant's theory of #**P**-completeness led to several structural insights into complexity classes around **NP** and interactive proof systems, as well as to the seminal result of Toda [45].

While counting problems in #**P** stayed in the focus of research for long, the study of the determinant by Damm [23], Vinay [47], and Toda [44] established that the complexity of computing the determinant of an integer matrix characterises the class #**L** up to a closure under subtraction. Allender and Ogihara [3] analysed the structure of complexity classes based on #**L**. The importance of counting classes based on logspace-bounded Turing machines (TMs) was further established by Allender, Beals and Ogihara [2]. They characterised the complexity of testing feasibility of linear equations by a class which is based on #**L**. Beigel and Fu [7] then showed that small depth circuits built with oracle access to #**L** functions lead to a hierarchy of languages which can be seen as the logspace version of the counting hierarchy. In a remarkable result, Ogihara [40] showed that this hierarchy collapses to the first level. Further down the complexity hierarchy, Caussinus et al. [12] introduced counting versions of $\mathbf{NC^1}$ based on various characterisations of $\mathbf{NC^1}$. The counting and probabilistic analogues of $\mathbf{NC^1}$ exhibit properties similar to their logspace counterparts [24]. Moreover, counting and gap variants of the class $\mathbf{AC^0}$ were defined by Agrawal et al. [1].

The theory of parameterised counting classes was pioneered by Flum and Grohe [32] as well as McCartin [39]. The class $\#\mathbf{W}[1]$ consists of all parameterised counting problems that reduce to the problem of counting k-cliques in a graph. Flum and Grohe [32] proved that counting cycles of length k is complete for $\#\mathbf{W}[1]$. Curticapean [18] further showed that counting matchings with k edges in a graph is also complete for $\#\mathbf{W}[1]$. These results led to several remarkable completeness results and new techniques (see, e.g., the works of Curticapean [19, 20], Curticapean, Dell and Marx [21], Jerrum and Meeks [37], Brand and Roth [10]).

Motivation. Given the rich structure of logspace-bounded counting complexity classes, the study of parameterised variants of these classes is vital to obtain a finer classification of counting problems.

A theory on para-logspace counting did not exist before. We wanted to overcome this defect to further understand the landscape of counting problems with decision versions in para-logspace-based classes. Our new framework allows us to classify many of these problems more precisely. In this article, we define counting variants inspired by the parameterised space complexity classes introduced by Elberfeld et al. [43, 28].

In the realm of space-bounded computation, different manners in which nondeterministic bits are accessed lead to different complexity classes. For example, the standard definition of **NL** implicitly gives the corresponding NTMs only read-once access to their nondeterministic bits [5]: nondeterminism is given only in the form of choices between different transitions. This means that nondeterministic bits are not re-accessible by the machine later in the computation. When instead using an auxiliary read-only tape for these bits and allowing for multiple passes on it, one obtains the class **NP**. This is due to the fact that **SAT** is **NP**-complete with respect to logspace many-one reductions [5], and that one can evaluate a CNF formula in deterministic logspace even when the assignment is given on a read-only tape. However, polynomial time bounded NTMs still characterise **NP** even when the machine is allowed to do only one pass on the nondeterministic bits as they can simply store all nondeterministic bits on the work-tape. So, it is very natural to investigate whether the differentiation from above leads to new insights in our setting.

With parameterisation as a means for a finer classification, Stockhusen and Tantau [43] defined nondeterministic logarithmic space-bounded computation based on *how* (unrestricted or read-once) the nondeterministic bits are accessed. Based on this distinction, they defined two operators: $\mathbf{para_W}$ (unrestricted) and $\mathbf{para}_\beta$ (read-once). Their study led to many compelling natural problems that are complete for logspace-bounded nondeterministic computations with suitable parameters. Thereby, a rich structure of computational power based on the restrictions on the number of reads of the nondeterministic bits was exhibited. In this article, we additionally differentiate based on *when* (unrestricted or tail access) the nondeterministic bits are accessed. The classes $\mathbf{W}[1]$ and $\mathbf{W}[\mathbf{P}]$ are the two most prominent nondeterministic classes in the parameterised world which is why we wanted to see the effect of such a restriction on the rather small classes in our setting. This leads to the new operators $\mathbf{para_{W[1]}}$ and $\mathbf{para_{\beta tail}}$. The concept of tail-nondeterminism allowed to capture the parameterised complexity class $\mathbf{W}[1]$ – via tail-nondeterministic, k-bounded machines – and thereby relates to many interesting problems such as searching for cliques, independent sets, or homomorphism, and evaluating conjunctive queries [33]. Intuitively, tail-nondeterminism means that all nondeterministic bits are read at the end of the computation, and k-boundedness limits the number of these nondeterministic bits to $f(k) \cdot \log |x|$ for all inputs (x, k).

Studying counting complexity often improves the understanding of related classical problems and classes (e.g., Toda's theorem [45]). With regard to space-bounded complexity, there are several characterisations of logspace-bounded counting classes in terms of natural problems. For example, counting paths in directed graphs is complete for $\#\mathbf{L}$, and checking if an integer matrix is singular or not is complete for the class $\mathbf{C_{=}L}$ (see Allender et al. [2]). Furthermore, testing if a system of linear equations is feasible or not can be done in $\mathbf{L}$ with queries to any complete language for $\mathbf{C_{=}L}$. Moreover, two hierarchies built over counting classes for logarithmic space collapse either to the first level [40] or to the second level [2]. Apart from this, the separation of various counting classes over logarithmic space remains widely open. For example, it is not known if the class $\mathbf{C_{=}L}$ is closed under complementation.

We consider different parameterised variants of the logspace-bounded counting class $\#\mathbf{L}$ to give a new perspective on its fine structure.

Results. We introduce the counting variants of parameterised space-bounded computation and show that each of the parameterised logspace complexity classes, defined by Stockhusen and Tantau [43], has a natural counting counterpart. Moreover, by considering also tail-nondeterminism with respect to their classes, we obtain four different variants of parameterised logspace counting classes, namely, $\#\mathbf{para_W L}$, $\#\mathbf{para}_\beta\mathbf{L}$, $\#\mathbf{para_{W[1]} L}$, and $\#\mathbf{para}_{\beta\mathbf{tail}}\mathbf{L}$. We show that $\#\mathbf{para_W L}$ and $\#\mathbf{para}_\beta\mathbf{L}$ are closed under para-logspace parsimonious reductions and that all of our new classes are closed under addition and multiplication.

Furthermore, we develop a complexity theory by obtaining natural complete problems for these new classes. We introduce variants of the problem of counting walks of parameter-bounded length that are complete for the classes $\#\mathbf{para}_\beta\mathbf{L}$ (Theorems 14, 15 and 18), $\#\mathbf{para}_{\beta\mathbf{tail}}\mathbf{L}$ (Theorem 16) and $\#\mathbf{para_W L}$ (Theorem 19). Since the same problem is shown to be complete for both, $\#\mathbf{para}_\beta\mathbf{L}$ and $\#\mathbf{para}_{\beta\mathbf{tail}}\mathbf{L}$, we get the somewhat surprising result that the closure of $\#\mathbf{para}_{\beta\mathbf{tail}}\mathbf{L}$ under para-logspace parsimonious reductions coincides with $\#\mathbf{para}_\beta\mathbf{L}$ (Corollary 17). Also, we show that a parameterised version of the problem of counting homomorphisms from coloured path structures to arbitrary structures is complete for $\#\mathbf{para}_\beta\mathbf{L}$ with respect to para-logspace parsimonious reductions (Theorem 28).

Afterwards, we study variants of the problem of counting assignments to free first-order variables in a quantifier-free FO formula. We identify complete problems for the classes $\#\mathbf{para}_\beta\mathbf{L}$ and $\#\mathbf{para_{W[1]} L}$ in this context. More specifically, counting assignments to free first-order variables in a quantifier-free formula with relation symbols of bounded arity and the syntactical locality of the variables in the formula being restricted (p-$\#\mathrm{MC}(\Sigma_0^{r\text{-local}})_a$) is shown to be complete for the classes $\#\mathbf{para}_{\beta\mathbf{tail}}\mathbf{L}$ and $\#\mathbf{para}_\beta\mathbf{L}$ with respect to para-logspace parsimonious reductions (Theorem 22). When there is no restriction on the arity of relational symbols or on the locality of the variables, counting the number of satisfying assignments to free first-order variables in a quantifier-free formula in a given structure (p-$\#\mathrm{MC}(\Sigma_0)$) is complete for $\#\mathbf{para_{W[1]} L}$ with respect to para-logspace parsimonious reductions (Theorem 23).

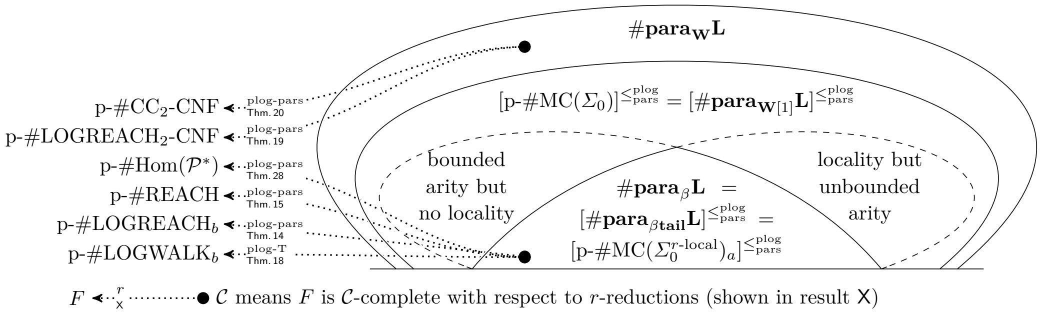

Figure 1 Diagram assuming pair-wise difference of studied classes with list of complete problems.

Finally, we consider a parameterised variant of the determinant function (p-det) introduced by Chauhan and Rao [13]. By adapting the arguments of Mahajan and Vinay [38], we show that p-det on $(0, 1)$-matrices can be expressed as the difference of two functions in $\#\mathbf{para}_\beta\mathbf{L}$, and is $\#\mathbf{para}_{\beta\mathbf{tail}}\mathbf{L}$-hard with respect to para-logspace many-one reductions (Theorem 33).

Figure 1 shows a class diagram with complete problems.

Main Techniques. Our primary contribution is laying foundations for the study of parameterised logspace-bounded counting complexity classes. The completeness results in Theorems 15 and 23 required a quantised normal form for k-bounded nondeterministic Turing Machines (NTMs) (Lemma 8). This normal form quantises the nondeterministic steps of a k-bounded NTM into chunks of $\log n$-many steps such that the total number of accepting paths remains the same. We believe that the normal form given in Lemma 8 will be useful in the structural study of parameterised counting classes. The study of p-det involved definitions of so-called parameterised clow sequences generalising the classical notion [38]. Besides, a careful assignment of signs to clow sequences was necessary for our complexity analysis of p-det.

Related Results. Chen and Müller [14] studied the parameterised complexity of counting homomorphisms and divided the problems into four equivalence classes. However, their equivalence is only based on reductions among variants of counting homomorphisms but not in terms of concrete complexity classes. In this context, Dalmau and Johnson [22] investigated the complexity of counting homomorphisms as well, and provided generalisations of results by Grohe [34] to the counting setting. A similar classification regarding our classes can give new insights into the complexity of the homomorphism problem (Open Problem 29). The behaviour of our classes with respect to reductions is similar to the one observed for $\mathbf{W}[1]$ by Bottesch [8, 9].

Outline. In Section 2, we introduce the considered machine model, as well as needed foundations of parameterised complexity theory, and logic. Section 3 presents structural results regarding our introduced notions in the parameterised counting context. Afterwards, in Section 4, our main results on counting walks, FO-assignments, homomorphisms as well as regarding the determinant are shown. Finally, we conclude in Section 5.
Due to space limitations, all proof details can be found in the technical report [36].

2 Preliminaries

In this section, we describe the computational models and complexity classes that are relevant for parameterised complexity theory. We use standard notions and notations from parameterised complexity theory [25, 33]. Without loss of generality, we restrict the input alphabet to be $\{0, 1\}$.

Turing Machines (TMs) with Random Access to the Input. We consider an intermediate model between TMs and Random Access Machines (RAMs) on words. Particularly, we make use of TMs that have random access to the input tape and can query relations in input structures in constant time. This can be achieved with two additional tapes of logarithmic size (in the input length), called the *random access tape* and the *relation query tape*. On the former, the machine can write the index of an input position to get the value of the respective bit of the input. On the relation query tape, the machine can write a tuple t of the input structure together with a relation identifier R to get the bit stating whether t is in the relation specified by R. Note that our model achieves linear speed-up for accessing the input compared to the standard TM model. (This is further justified by Remark 6.) For convenience, in the following, whenever we speak about TMs we mean the TM model with random access to the input. Denote by $\mathbf{SPACETIME}(s, t)$ ($\mathbf{NSPACETIME}(s, t)$) with $s, t\colon \mathbb{N} \to \mathbb{N}$ the class of languages that are accepted by (nondeterministic) TMs with space-bound $O(s(n))$ and time-bound $O(t(n))$. A $\mathcal{C}$-machine for $\mathcal{C} = \mathbf{SPACETIME}(s, t)$ ($\mathcal{C} = \mathbf{NSPACETIME}(s, t)$) is a (nondeterministic) TM that is $O(s(n))$ space-bounded and $O(t(n))$ time-bounded.

NTMs are a generalisation of TMs where multiple transitions from a given configuration are allowed. This can be formalised by allowing the transition to be a relation rather than a function. Sometimes, it is helpful to view NTMs as deterministic TMs with an additional tape, called the (nondeterministic) choice tape which is read-only. Let M be a deterministic TM with a choice tape. A nondeterministic step in the computation of M is a step where M moves the head on the choice tape to a cell that was not visited before. The *language accepted by* M, $L(M)$ is defined as

$$\{\, x \in \{0, 1\}^* \mid \exists y \in \{0, 1\}^* \text{ s.t. } M \text{ accepts } x \text{ when the choice tape is initialised with } y \,\}.$$

Notice that in this framework the machine M has two-way access to the choice tape. Furthermore, resource bounds are with respect to the input only (the content of the choice tape is not part of the input) and the choice tape is not counted for space bounds. In this paper, we regard nondeterministic TMs as deterministic ones with a choice tape.

Before we proceed to the definition of parameterised complexity classes, a clarification on the choice of the model is due. Note that RAMs and NRAMs are often appropriate in the parameterised setting as exhibited by several authors (see, e.g., the textbook of Flum and Grohe [33]). They allow to define bounded nondeterminism quite naturally. On the other hand, in the classical setting, branching programs (BPs) are one of the fundamental models that represent space bounded computation, in particular logarithmic space. Since BPs inherently use bit access, this relationship suggests the use of a bit access model. Consequently, we consider a hybrid computational model: Turing machines with random access to the input. While the computational power of this model is the same as that of Turing machines and RAMs, it seems to be a natural choice to guarantee a certain robustness, allowing for desirable characterisations of our classes.

Parameterised Complexity Classes. Let **FPT** denote the set of parameterised problems that can be decided by a deterministic TM running in time $f(k) \cdot p(|x|)$ for any input (x, k) where f is a computable function and p is a polynomial. Two central classes in parameterised complexity theory are $\mathbf{W}[1]$ and $\mathbf{W}[\mathbf{P}]$ which were originally defined via special types of circuit satisfiability [33]. Flum, Chen and Grohe [17] obtained a characterisation of these two classes using the following notion of k-bounded NTMs.

▶ **Definition 1** (k-bounded TMs)**.** *An NTM M, working on inputs of the form (x, k) with $x \in \{0,1\}^*, k \in \mathbb{N}$, is said to be k-*bounded *if for all inputs (x, k) it reads at most $f(k) \cdot \log |x|$ bits from the choice tape on input (x, k), where f is a computable function.*

Here, we will work with the following characterisation of $\mathbf{W}[\mathbf{P}]$. The characterisation for $\mathbf{W}[1]$ needs another concept that will be defined on the next page.

▶ **Proposition 2** ([17, 33])**.** *$\mathbf{W}[\mathbf{P}]$ is the set of all parameterised problems that can be accepted by k-bounded* **FPT***-machines with a choice tape.*

Now, we recall three complexity theoretic operators that define parameterised complexity classes from an arbitrary classical complexity class, namely $\mathbf{para}, \mathbf{para_W}$ and $\mathbf{para}_\beta$, following the notation of Stockhusen [42].

▶ **Definition 3** ([31])**.** *Let $\mathcal{C}$ be any complexity class. Then $\mathbf{para}\mathcal{C}$ is the class of all parameterised problems $P \subseteq \{0,1\}^* \times \mathbb{N}$ for which there is a computable function $\pi \colon \mathbb{N} \to \{0,1\}^*$ and a language $L \in \mathcal{C}$ with $L \subseteq \{0,1\}^* \times \{0,1\}^*$ such that for all $x \in \{0,1\}^*$, $k \in \mathbb{N}$: $(x, k) \in P \Leftrightarrow (x, \pi(k)) \in L$.*

Notice that $\mathbf{paraP} = \mathbf{FPT}$ is the standard precomputation characterisation of **FPT** [31]. A $\mathbf{para}\mathcal{C}$-machine for $\mathcal{C} = \mathbf{SPACETIME}(s, t)$ ($\mathcal{C} = \mathbf{NSPACETIME}(s, t)$) is a (nondeterministic) TM, working on inputs of the form (x, k), that is $O(s(|x| + f(k)))$ space-bounded and $O(t(|x| + f(k)))$ time-bounded where f is a computable function.

The class **XP** (problems accepted in time $|x|^{f(k)}$ for a computable function f) and the **W**-hierarchy [33] capture intractability of parameterised problems. Though the **W**-hierarchy was defined using the weighted satisfiability of formulas with bounded weft, which is the number of alternations between gates of high fan-in, Flum and Grohe [31] characterised central classes in this context using bounded nondeterminism. Stockhusen and Tantau [43, 42] considered space-bounded and circuit-based parallel complexity classes with bounded nondeterminism.

The following definition is a more formal version of the one given by Stockhusen and Tantau [43, Def. 2.1]. They use $\text{para}\exists^{\leftrightarrow}_{f \log}\mathcal{C}$ instead of $\mathbf{para_W}\mathcal{C}$ for a complexity class $\mathcal{C}$.

▶ **Definition 4.** *Let $\mathcal{C} = \mathbf{SPACETIME}(s, t)$ for some $s, t \colon \mathbb{N} \to \mathbb{N}$. Then, $\mathbf{para_W}\mathcal{C}$ is the class of all parameterised problems Q that can be accepted by a k-bounded $\mathbf{para}\mathcal{C}$-machine with a choice tape.*

For example, $\mathbf{para_W L}$ denotes the parameterised version of **NL** with k-bounded nondeterminism. One can also restrict this model by only giving one-way access to the nondeterministic tape. The following definition is a more formal version of the one of Stockhusen and Tantau [43, Def. 2.1] who use the symbol $\text{para}\exists^{\rightarrow}_{f \log}$ instead.

▶ **Definition 5.** *Let $\mathcal{C} = \mathbf{SPACETIME}(s, t)$ for some $s, t \colon \mathbb{N} \to \mathbb{N}$. Then $\mathbf{para}_\beta\mathcal{C}$ denotes the class of all parameterised problems Q that can be accepted by a k-bounded $\mathbf{para}\mathcal{C}$-machine with a choice tape with one-way read access to the choice tape.*

As there is only read-once access to the nondeterministic bits, $\mathbf{para}_\beta\mathcal{C}$ can be equivalently defined via nondeterministic transitions and without using a choice tape.

Another notion studied in parameterised complexity is tail-nondeterminism. A k-bounded machine M is *tail-nondeterministic* if there exists a computable function g such that on all inputs (x, k), M makes at most $g(k) \cdot \log n$ further steps in the computation, after its first nondeterministic step. The value of this concept is evidenced by the machine characterisation of $\mathbf{W}[1]$ (Chen et al. [17]). We hope to get new insights by transferring this concept to space-bounded computation. In consequence, we introduce the tail-nondeterministic versions of $\mathbf{para}_{\mathbf{W}}\mathcal{C}$ and $\mathbf{para}_\beta\mathcal{C}$ which are denoted by $\mathbf{para}_{\mathbf{W}[1]}\mathcal{C}$ and $\mathbf{para}_{\beta\mathbf{tail}}\mathcal{C}$.

▶ Remark 6. Note that it is important to have random access to the input tape in the case of tail-nondeterminism. Without random access to input bits and input relations, a TM cannot even make reasonable queries to the input in time $g(k) \cdot \log(n)$.

Logic. We assume basic familiarity with first-order logic (FO). A *vocabulary* is a finite ordered set of relation symbols and constants. Each relation symbol R has an associated *arity* $\operatorname{arity}(R) \in \mathbb{N}$. Let τ be a vocabulary. A *τ-structure* $\mathbf{A}$ consists of a nonempty finite set $\operatorname{dom}(\mathbf{A})$ (its *universe*), and an *interpretation* $R^{\mathbf{A}} \subseteq \operatorname{dom}(\mathbf{A})^{\operatorname{arity}(R)}$ for every relation symbol $R \in \tau$. Syntax and semantics are defined as usual (see, e.g., the textbook of Ebbinghaus et al. [27]). Let $\mathbf{A}$ be a structure with universe A. We denote by $|\mathbf{A}|$ the *size of a binary encoding of* $\mathbf{A}$, i.e., the number of bits required to represent the universe and relations as lists of tuples. For example, if R is a relation of arity 3, then $R^{\mathbf{A}}$ is represented as a subset of A^3, i.e., a set of triples over A. This requires $O(|R^{\mathbf{A}}| \cdot \operatorname{arity}(R)) \cdot \log |A|)$ bits to represent the relation $R^{\mathbf{A}}$, assuming $\log |A|$ bits to represent an element in A. As analysed by Flum et al. [30, Sect. 2.3], this means that $|\mathbf{A}| \in \Theta((|A| + |\tau| + \sum_{R\in\tau} |R^{\mathbf{A}}| \cdot \operatorname{arity}(R)) \cdot \log |A|)$. Also recall that the fragment Σ_i (for $i \in \mathbb{N}$) refers to the class of FO-formulas with i quantifier blocks alternating between existential and universal quantifiers and the outermost quantifier being existential.

3 Parameterised Counting in Logarithmic Space

Now, we define the counting counterparts based on the parameterised classes defined using bounded nondeterminism. The definitions of the decision classes based on tail-nondeterminism can be found in the technical report [36]. A *parameterised function* is a function $F\colon \{0,1\}^* \times \mathbb{N} \to \mathbb{N}$. For an input (x, k) of F with $x \in \{0,1\}^*$, $k \in \mathbb{N}$, we call k the *parameter* of that input. If $\mathcal{C}$ is a complexity class and a parameterised function F belongs to $\mathcal{C}$, we say that F is $\mathcal{C}$-computable. Let M be a TM. We denote by $\operatorname{acc}_M(x)$ the number of accepting paths of M on input x, and similarly, $\operatorname{acc}_M(x, k)$, for parameterised inputs of the form (x, k).

▶ **Definition 7.** *Let $\mathcal{C} = \mathbf{SPACETIME}(s, t)$ for some $s, t\colon \mathbb{N} \to \mathbb{N}$. Then a parameterised function F is in $\#\mathbf{para}_{\mathbf{W}}\mathcal{C}$ if there is a k-bounded nondeterministic $\mathbf{para}\mathcal{C}$-machine M such that for all inputs (x, k), we have that $\operatorname{acc}_M(x, k) = F(x, k)$. Furthermore, F is in*

- *$\#\mathbf{para}_\beta\mathcal{C}$ if there is such an M with read-once access to its nondeterministic bits,*
- *$\#\mathbf{para}_{\mathbf{W}[1]}\mathcal{C}$ if there is such an M that is tail-nondeterministic, and*
- *$\#\mathbf{para}_{\beta\mathbf{tail}}\mathcal{C}$ if there is such an M with read-once access to its nondeterministic bits that is tail-nondeterministic.*

By definition, we get $\#\mathbf{para}_{\beta\mathbf{tail}}\mathbf{L} \subseteq \mathcal{C} \subseteq \#\mathbf{para}_{\mathbf{W}}\mathbf{L}$ for $\mathcal{C} \in \{\#\mathbf{para}_\beta\mathbf{L}, \#\mathbf{para}_{\mathbf{W}[1]}\mathbf{L}\}$. Note that the restriction of the above classes to k-boundedness is crucial. If we drop this restriction, the machines are able to access $2^{f(k)+log|x|}$, so fpt-many, nondeterministic bits.

Regarding multiple-read access, this allows for solving SAT (with constant parameterisation). So this class then would contain a **paraNP**-complete problem. For read-once access, we expect a similar result for **paraNL**. When adding tail-nondeterminism, we implicitly get k-boundedness again, so this does not lead to new classes.

The following lemma shows that **paraL**-machines can be normalised in a specific way. This normalisation will play a major role in Section 4.

▶ **Lemma 8.** *For any k-bounded nondeterministic **paraL**-machine M there exists a k-bounded nondeterministic **paraL**-machine M' with $\#\mathrm{acc}_M(x,k) = \#\mathrm{acc}_{M'}(x,k)$ for all inputs (x,k) such that M' has the following properties:*
(1) *M' has a unique accepting configuration,*
(2) *on any input (x,k), every computation path of M' accesses exactly $g(k) \cdot \log|x|$ nondeterministic bits (for some computable function g), and M' counts on an extra tape (tape S) the number of nondeterministic steps, and*
(3) *M' has an extra tape (tape C) on which it remembers previous nondeterministic bits, resetting the tape after every $\log|x|$-many nondeterministic steps.*
Additionally, if M has read-once access to its nondeterministic bits, or is tail-nondeterministic, or both, then M' also has these properties.

The following result follows from a simple simulation of nondeterministic machines by deterministic ones. Let **FFPT** be the class of functions computable by **FPT**-machines with output.

▶ **Theorem 9.** $\#\mathbf{para}_\beta\mathbf{L} \subseteq \mathbf{FFPT}$.

Using the notion of oracle machines (see, e.g., [41]), we define Turing, metric, and parsimonious reductions computable in **paraL**. For our purposes, the oracle tape is always exempted from space restrictions which is often the case in the context of logspace Turing reductions [11]. A study on the effect of changing this assumption might be interesting.

▶ **Definition 10** (Reducibilities)**.** *Let $F, F' \colon \{0,1\}^* \times \mathbb{N} \to \mathbb{N}$ be two functions. Then, F is* para-logspace Turing reducible *to F', $F \leq_T^{\mathrm{plog}} F'$, if there is a **paraL** oracle TM M that computes F with oracle F' and the parameter of any oracle query of M is bounded by a computable function in the parameter. If there is such an M that uses only one oracle query, then F is* para-logspace metrically reducible *to F', $F \leq_{\mathrm{met}}^{\mathrm{plog}} F'$. If there is such an M that returns the answer of the first oracle query, then F is* para-logspace parsimoniously reducible *to F', $F \leq_{\mathrm{pars}}^{\mathrm{plog}} F'$.*

Note that the definition of parsimonious reductions ensures that the size of the witness set is preserved by the fact that M immediately returns the answer of its only oracle query (without further computations). For any reducibility relation $\preccurlyeq$ and any complexity class $\mathcal{C}$, $[\mathcal{C}]^{\preccurlyeq} := \{\, A \mid \exists C \in \mathcal{C} \text{ such that } A \preccurlyeq C \,\}$ is the *$\preccurlyeq$-closure of $\mathcal{C}$*.

Next, we show that both new classes without tail-nondeterminism are closed under $\leq_{\mathrm{pars}}^{\mathrm{plog}}$.

▶ **Lemma 11.** *The classes $\#\mathbf{para_W L}$ and $\#\mathbf{para}_\beta\mathbf{L}$ are closed under $\leq_{\mathrm{pars}}^{\mathrm{plog}}$.*

For the tail-classes, such a closure property is not obvious. Corollary 16 will show that closing the class with read-once access and tail-nondeterminism under these reductions gives the full power of the class without tail-nondeterminism. Open Problem 24 on page 12 asks what class is obtained when closing the class without read-once access and with tail-nondeterminism.

Another important question is whether classes are closed under certain arithmetic operations. We show that all newly introduced classes are closed under addition and multiplication.

▶ **Theorem 12.** *For any $o \in \{\mathbf{W}, \mathbf{W}[1], \beta, \beta\text{-}\mathbf{tail}\}$, the class $\#\mathbf{para}_o\text{-}\mathbf{L}$ is closed under addition and multiplication.*

▶ **Open Problem 13.** *Which of the classes are closed under* monus*, that is,* $\max\{F - G, 0\}$*?*

4 Complete Problems

This section studies complete problems for the previously defined classes: counting problems in the context of walks in directed graphs, model-checking problems for FO formulas, and homomorphisms between FO-structures as well as a parameterised version of the determinant.

4.1 Counting Walks

We start with parameterised variants of counting walks in directed graphs which will be shown to be complete for the introduced classes.

Problem:	p-#LOGREACH$_b$
Input:	directed graph $\mathfrak{G} = (V, E)$ with out-degree b, $s, t \in V$ and $a, k \in \mathbb{N}$.
Parameter:	k.
Output:	number of s-t-walks of length a if $a \leq k \cdot \log |V|$, 0 otherwise.

▶ **Theorem 14.** *For every $b \geq 2$, p-#LOGREACH$_b$ is $\#\mathbf{para}_\beta\mathbf{L}$-complete with respect to $\leq^{\text{plog}}_{\text{pars}}$-reductions.*

Proof Idea. For the upper bound, guess a path of length exactly a. The number of nondeterministic bits is bounded by $O(k \cdot \log |V|)$ since successors can be referenced by a number in $\{0, \ldots, b-1\}$.

For the lower bound, using Lemma 8, construct the configuration graph $\mathfrak{G}$ restricted to nondeterministic configurations and the unique accepting configuration C_{acc}, where the edge relation expresses whether a configuration is reachable with exactly one nondeterministic, but an arbitrary number of deterministic steps. Accepting computations of the machine correspond to paths from the first nondeterministic configuration to C_{acc} of length $f(k) \cdot \log |\mathfrak{G}|$ in $\mathfrak{G}$. ◀

Now consider the problem p-#REACH, defined as follows.

Problem:	p-#REACH
Input:	directed graph $\mathfrak{G} = (V, E)$, $s, t \in V$, $k \in \mathbb{N}$.
Parameter:	k.
Output:	number of s-t-walks of length exactly k.

The difference to the previous problem is the unbounded out-degree of nodes and the length of counted walks being k instead of $a \leq k \cdot \log |x|$. Note that the analogue problem for counting paths is $\#\mathbf{W}[1]$-complete [32]. However, we will see now that the problem for walks is $\#\mathbf{para}_\beta\mathbf{L}$-complete.

▶ **Theorem 15.** *p-#REACH is $\#\mathbf{para}_\beta\mathbf{L}$-complete with respect to $\leq^{\text{plog}}_{\text{pars}}$.*

As the length of paths that are counted in p-#REACH is k, the runtime of the whole algorithm used to prove membership in the previous theorem is actually bounded by $k \cdot \log |x|$ on input (x, k). This means that the computation is tail-nondeterministic.

▶ **Theorem 16.** p-#REACH *is* $\#\mathbf{para}_{\beta\mathbf{tail}}\mathbf{L}$*-complete with respect to* $\leq^{\text{plog}}_{\text{pars}}$.

The previous results together with the fact that $\#\mathbf{para}_\beta\mathbf{L}$ is closed under $\leq^{\text{plog}}_{\text{pars}}$ yield the following surprising collapse (a similar behaviour was observed by Bottesch [8, 9]).

▶ **Corollary 17.** $[\#\mathbf{para}_{\beta\mathbf{tail}}\mathbf{L}]^{\leq^{\text{plog}}_{\text{pars}}} = \#\mathbf{para}_\beta\mathbf{L}$.

We continue with another variant of p-#LOGREACH$_b$, namely p-#LOGWALK$_b$. Here, all walks of length a are counted, if $a \leq k \cdot \log|x|$ (and s, t are not part of the input).

▶ **Theorem 18.** p-#LOGWALK$_b$ *is* $\#\mathbf{para}_\beta\mathbf{L}$*-complete with respect to* $\leq^{\text{plog}}_{T}$.

Now, consider a problem that combines a reachability problem with model-checking for propositional logic, that is, it only counts walks that are models of a propositional formula (see Haak et al. [35]). Let $\mathfrak{G} = (V, E)$ be a DAG, $(e_1, \dots, e_n)$ be a walk in $\mathfrak{G}$ with $e_i \in E$ for $1 \leq i \leq n$, and $P = \{e_1, \dots, e_n\}$. Define the function $c_P : E \to \{0, 1\}$ to be the characteristic function of P with respect to E: $c_P(e) = 1$ iff $e \in P$.

Problem:	p-#LOGREACH$_2$-CNF
Input:	directed graph $\mathfrak{G} = (V, E)$ of out-degree 2, $s, t \in V$, CNF formula φ with $\text{Vars}(\varphi) \subseteq E$, $a, k \in \mathbb{N}$.
Parameter:	k.
Output:	Number of s-t-walks ($s = e_1, \dots, e_a = t$) such that $c_P \models \varphi$, where $P = \{e_1, \dots, e_a\}$, if $a \leq k \cdot \log(\|V\| + \|\varphi\|)$, 0 otherwise.

▶ **Theorem 19.** p-#LOGREACH$_2$-CNF *is* $\#\mathbf{para}_W\mathbf{L}$*-complete with respect to* $\leq^{\text{plog}}_{\text{pars}}$.

Proof Idea. Regarding membership, we can first use the algorithm outlined in the proof idea of Theorem 14 to nondeterministically guess a path, and then use the standard logspace model-checking algorithm for propositional formulas. Since edges in the graph are associated with variables of the formula, whenever we need the value of an edge variable e, we run the original algorithm re-using nondeterministic bits to determine it.

For the lower bound, we use the same graph as in Theorem 14, and the formula is used to express consistency of the re-used nondeterministic bits in the configuration graph. ◀

Similarly, define the problem p-#CC$_2$-CNF: Given a graph $\mathfrak{G} = (V, E)$ of bounded out-degree 2, a CNF-formula φ with $\text{Vars}(\varphi) \subseteq E$ and $a, k \in \mathbb{N}$, with k as the parameter and $a \leq \log(|G| + |\varphi|)$, output the number of cycle covers $D \subseteq E$ in which the number of non-selfloop-cycles is $\leq k$, exactly $k \cdot a$ vertices are covered non-trivially and $\text{Vars}(D) \models \varphi$.

▶ **Theorem 20.** p-#CC$_2$-CNF *is* $\#\mathbf{para}_W\mathbf{L}$*-complete with respect to* $\leq^{\text{plog}}_{\text{pars}}$.

4.2 Counting FO-Assignments

Let $\mathcal{F}$ be a class of well-formed formulas. The problem of counting satisfying assignments to free FO-variables in $\mathcal{F}$-formulas, *p*-#MC($\mathcal{F}$), is defined as follows.

Problem:	*p*-#MC($\mathcal{F}$)
Input:	formula $\varphi \in \mathcal{F}$, structure $\mathbf{A}$, $k \in \mathbb{N}$.
Parameter:	$\|\varphi\|$.
Output:	$\|\varphi(\mathbf{A})\|$ if $k = \|\varphi\|$, 0 otherwise.

Here, $\varphi(\mathbf{A})$ is the set of satisfying assignments of φ in $\mathbf{A}$:

$$\varphi(\mathbf{A}) = \{\, (a_1,\dots,a_n) \mid (a_1,\dots,a_n) \in \mathrm{dom}(\mathbf{A})^n, \mathbf{A} \models \varphi(a_1,\dots,a_n) \,\}.$$

Denote by $p\text{-}\#\mathrm{MC}(\mathcal{F})_a$ the variant where for all relations the arity is at most $a \in \mathbb{N}$. We investigate parameterisations that yield complete problems for some of the new classes in this setting.

In particular, we consider a fragment of FO obtained by restricting the occurrence of variables in the syntactic tree of a formula in a purely syntactic manner. Formally, the *syntax tree* of a quantifier-free FO-formula φ is a tree with edge-ordering whose leaves are labelled by atoms of φ and whose inner vertices are labelled by Boolean connectives.

▶ **Definition 21.** *Let $r \in \mathbb{N}$ and φ be a quantifier-free* FO*-formula. Let $\theta_1,\dots,\theta_m$ be the atoms of φ in the order of their occurrence in the order-respecting depth-first run through the syntax tree of φ. We say that φ is r*-local *if for any θ_i, θ_j that involve the same variable, we have $|i-j| \le r$. We define $\Sigma_0^{r\text{-}local} := \{\, \varphi \in \Sigma_0 \mid \varphi \text{ is } r\text{-local} \,\}$.*

Using this syntactic notion, we obtain a complete problem for our classes with read-once access to nondeterministic bits in the setting of first-order model-checking.

▶ **Theorem 22.** *For $a \ge 2, r \ge 1$,* $\mathrm{p}\text{-}\#\mathrm{MC}(\Sigma_0^{r\text{-}local})_a$ *is $\#\mathbf{para}_\beta\mathbf{L}$-complete and $\#\mathbf{para}_{\beta\mathbf{tail}}\mathbf{L}$-complete with respect to $\le^{\mathrm{plog}}_{\mathrm{pars}}$.*

Proof Idea. Regarding membership, we evaluate the given φ in $\mathbf{A}$ top to bottom using the locality of φ by storing assignments to variables until we encountered r more atoms. As a result, at most $a \cdot r$ assignments to variables are simultaneously stored and each one needs $\log|\mathbf{A}|$ space. Moreover, the runtime of the whole procedure is bounded by $f(|\varphi|) \cdot \log|\mathbf{A}|$ for some computable function f and thereby the procedure is tail-nondeterministic.

Regarding the lower bound, we reduce from p-#REACH and use the formula

$$\varphi_k(x_1,\dots,x_k) := (x_1 = s) \wedge E(x_1,x_2) \wedge E(x_2,x_3) \wedge \ldots \wedge E(x_{k-1},x_k) \wedge x_k = t$$

expressing that a tuple of vertices $(v_1,\dots,v_k)$ is an $s^{\mathbf{A}}$-$t^{\mathbf{A}}$-walk in an (E,s,t)-structure $\mathbf{A}$. ◀

Note that the decision version of $p\text{-}\#\mathrm{MC}(\Sigma_0)$ is equivalent to parameterised model-checking for Σ_1-sentences, as we count assignments to free variables. This problem characterises tail-nondeterministic para-logspace with read-once access to nondeterministic bits.

▶ **Theorem 23.** *$p\text{-}\#\mathrm{MC}(\Sigma_0)$ is $\#\mathbf{para}_{\mathbf{W}[1]}\mathbf{L}$-complete with respect to $\le^{\mathrm{plog}}_{\mathrm{pars}}$.*

The complexity status of counting assignments to free first-order variables in a Σ_0 formula with unbounded arity or without the local restrictions is not known. In particular, it is not clear if the restriction on the arity or the locality property of the formula can be removed while preserving completeness. Finally, we close this section with three open questions.

▶ **Open Problem 24.** *What is the complexity of $[\mathrm{p}\text{-}\#\mathrm{MC}(\Sigma_0)_a]^{\le^{\mathrm{plog}}_{\mathrm{pars}}}$ for fixed $a \in \mathbb{N}$? What is the complexity of $[\mathrm{p}\text{-}\#\mathrm{MC}(\Sigma_0^{r\text{-}local})]^{\le^{\mathrm{plog}}_{\mathrm{pars}}}$ for fixed $r \in \mathbb{N}$?*

▶ **Open Problem 25.** *Is the class $[\#\mathbf{para}_{\mathbf{W}[1]}\mathbf{L}]^{\le^{\mathrm{plog}}_{\mathrm{pars}}}$ equivalent to some known class?*

4.3 Counting Homomorphisms

This subsection is devoted to the study of the problem of counting homomorphisms between two structures in the parameterised setting. Typically, the size of the universe of the first structure is considered as the parameter. The complexity of counting homomorphisms has been intensively investigated for almost two decades [26, 34, 22, 14].

▶ **Definition 26** (Homomorphism). *Let* $\mathbf{A}$ *and* $\mathbf{B}$ *be structures over some vocabulary* τ *with universes* A *and* B*, respectively. A function* $h\colon A \to B$ *is a* homomorphism from $\mathbf{A}$ to $\mathbf{B}$ *if for all* $R \in \tau$ *and for all tuples* $(a_1, \dots, a_{\text{arity}(R)}) \in R^{\mathbf{A}}$*, we have* $(h(a_1), \dots, h(a_{\text{arity}(R)})) \in R^{\mathbf{B}}$*.*

A bijective homomorphism h between two structures $\mathbf{A}, \mathbf{B}$ such that the inverse of h is also a homomorphism is called an *isomorphism*. If there is an isomorphism between $\mathbf{A}$ and $\mathbf{B}$, then $\mathbf{A}$ is said to be *isomorphic* to $\mathbf{B}$.

▶ **Definition 27.** *Let* $\mathbf{A}$ *be a structure with universe* A*. Then denote by* $\mathbf{A}^*$ *the extension of* $\mathbf{A}$ *by a fresh unary relation symbol* C_a *interpreted as* $C_a^{\mathbf{A}} = \{a\}$ *for each* $a \in \text{dom}(\mathbf{A})$*. Analogously, denote by* $\mathcal{A}^*$ *for a class of structures* $\mathcal{A}$ *the class* $\{\, \mathbf{A}^* \mid \mathbf{A} \in \mathcal{A} \,\}$*.*

Define p-#Hom($\mathcal{A}$) as the following problem. Given a pair of structures $(\mathbf{A}, \mathbf{B})$ where $\mathbf{A} \in \mathcal{A}$, and parameter k, output the number of homomorphisms from $\mathbf{A}$ to $\mathbf{B}$, if $|\text{dom}(\mathbf{A})| \le k$, and 0 otherwise.

Problem:	p-#Hom($\mathcal{A}$)
Input:	A pair of structures $(\mathbf{A}, \mathbf{B})$ where $\mathbf{A} \in \mathcal{A}$.
Parameter:	$\lvert\mathbf{A}\rvert$, $k \in \mathbb{N}$.
Output:	the number of homomorphisms from $\mathbf{A}$ to $\mathbf{B}$ if $\lvert\text{dom}(\mathbf{A})\rvert \le k$, 0 otherwise.

Notice that $\mathbf{B}$ can be any structure. For $n \ge 2$, let P_n be the canonical undirected path of length n, that is, the (E)-structure with universe $\{1, \dots, n\}$ and $E^{P_n} = \{\, (i, i+1), (i+1, i) \mid 1 \le i < n \,\}$. Let $\mathcal{P}$ be the class of structures isomorphic to some P_n. For the next theorem, reduce to p-#REACH for membership, and from a normalised, coloured variant of p-#REACH for hardness.

▶ **Theorem 28.** p-#Hom($\mathcal{P}^*$) *is* $\#\mathbf{para}_\beta\mathbf{L}$*-complete with respect to* $\le^{\text{plog}}_{\text{pars}}$*.*

▶ **Open Problem 29.** *Is there a natural class of structures* $\mathcal{A}$ *such that* p-#Hom($\mathcal{A}$) *is* $\#\mathbf{para}_{\mathbf{W}[1]}\mathbf{L}$*-complete with respect to* $\le^{\text{plog}}_{\text{pars}}$*?*

4.4 The Parameterised Complexity of the Determinant

In this section, we consider a parameterised variant of the determinant function introduced by Chauhan and Rao [13]. For $n > 0$ let $\mathcal{S}_n$ denote the set of all permutations of $\{1, \dots, n\}$. For $k \le n$, let $\mathcal{S}_{n,k}$ denote the following subset of $\mathcal{S}_n$: $\mathcal{S}_{n,k} = \{\, \pi \mid \pi \in S_n, |\{\, i : \pi(i) \ne i \,\}| = k \,\}$.

We define the parameterised determinant function of an $n \times n$ square matrix $A = (a_{i,j})_{1 \le i,j \le n}$ as $p\text{-det}(A, k) = \sum_{\pi \in S_{n,k}} \text{sign}(\pi) \prod_{i:\pi(i) \ne i} a_{i,\pi(i)}$.

Using an interpolation argument, it can be shown that p-det is in **FP** when k is part of the input and thereby in **FFPT** [13], the functional counterpart of **FPT**. In fact, the same interpolation argument can be used to show that p-det is in **GapL** (the class of functions $f(x)$ such that for some **NL**-machine, $f(x)$ is the number of accepting minus the number of rejecting paths). However, this does not give a space efficient algorithm for p-det in the sense of parameterised classes. The **GapL** algorithm may require a large number of nondeterministic steps and accordingly is not k-bounded. We show that the space efficient algorithm for the determinant given by Mahajan and Vinay [38] can be adapted to the parameterised setting, proving that p-det can be written as a difference of two $\#\mathbf{para}_\beta\mathbf{L}$ functions. Recall the notion of a *clow sequence* introduced by Mahajan and Vinay [38].

▶ **Definition 30** (Clow). *Let $\mathfrak{G} = (V, E)$ be a directed graph with $V = \{1, \ldots, n\}$ for some $n \in \mathbb{N}$. A* clow *in $\mathfrak{G}$ is a walk $C = (w_1, \ldots, w_{r-1}, w_r = w_1)$ where w_1 is the minimal vertex among $w_1, \ldots, w_{r-1}$ with respect to the natural ordering of V and $w_1 \neq w_j$ for all $1 < j < r$. Node w_1 is called the* head *of C, denoted by* $\text{head}(C)$.

▶ **Definition 31** (Clow sequence). *A* clow sequence *of a graph $\mathfrak{G} = (\{1, \ldots, n\}, E)$ is a sequence $W = (C_1, \ldots, C_k)$ such that C_i is a clow of $\mathfrak{G}$ for $1 \leq i \leq k$ and*

- *the heads of the sequence are in ascending order* $\text{head}(C_1) < \cdots < \text{head}(C_k)$*, and*
- *the total number of edges that appear in some C_i (including multiplicities) is exactly n.*

For a clow sequence W of some graph $\mathfrak{G} = (\{1, \ldots, n\}, E)$ with r clows the *sign of* W, $\text{sign}(W)$, is defined as $(-1)^{n+r}$. Note that, if the clow sequence is a cycle cover σ, then $(-1)^{n+r}$ is equal to the sign of the permutation represented by σ (that is, $(-1)^{\#\text{ inversions in }\sigma}$). Mahajan and Vinay came up with this sign-function to derive their formula for the determinant.

For an $(n \times n)$-matrix A, $\mathfrak{G}_A$ is the weighted directed graph with vertex set $\{1, \ldots, n\}$ and weighted adjacency matrix A. For a clow (sequence) W, $\text{weight}(W)$ is the product of weights of the edges (clows) in w. For any $\mathfrak{G}$ as above, $\mathcal{W}_\mathfrak{G}$ is the set of all clow sequences of $\mathfrak{G}$. Mahajan and Vinay [38] proved that $\det(A) = \sum_{W \in \mathcal{W}_{\mathfrak{G}_A}} \text{sign}(W) \cdot \text{weight}(W)$.

We adapt these notions to the parameterised setting. First observe that for a permutation $\sigma \in S_{n,k}$, we have that $\text{sign}(\sigma) = (-1)^{n+r}$, where r is the number of cycles in the permutation. However, the number of cycles in σ is $n - k + r'$, where r' is the number of cycles of length at least two in σ. Accordingly, we have $\text{sign}(\sigma) = (-1)^{2n-k+r'}$. Adapting the definition of a clow sequence, for $k \geq 0$, define a *k-clow sequence* to be a clow sequence where the total number of edges (including multiplicity) in the sequence is exactly k, every clow has at least two edges, and no self loop edge of the form (i, i) occurs in any of the clows. For any graph $\mathfrak{G}$ with vertex set $\{1, \ldots, n\}$ for $n \in \mathbb{N}$, $\mathcal{W}_{\mathfrak{G},k}$ is the set of all k-clow sequences of $\mathfrak{G}$. For a k-clow sequence $W \in \mathcal{W}_{\mathfrak{G},k}$, $\text{sign}(W)$ is $(-1)^{2n-k+r'}$, where r' is the number of clows in W. Mahajan and Vinay [38, Theorem 1] showed that the signed sum of the weights of all clow sequences is equal to the determinant. At the outset, this is a bit surprising, since the determinant is equal to the signed sum of weights of cycle covers, whereas there are clow sequences that are not cycle covers. Mahajan and Vinay [38] observed that every clow sequence that is not a cycle cover can be associated with a unique clow sequence of opposite sign, and thereby all clow sequences cancel out. We observe a parameterised version of the above result [38, Theorem 1].

▶ **Lemma 32.** $p\text{-det}(A, k) = \sum_{W \in \mathcal{W}_{\mathfrak{G}_A,k}} \text{sign}(W) \cdot \text{weight}(W)$*, for $\{0, 1\}$-matrix A, $k \in \mathbb{N}$.*

Using this characterisation, the upper bound in the following theorem can be obtained. For hardness a reduction from p-#REACH suffices.

▶ **Theorem 33.** *The problem p-det for $(0, 1)$-matrices can be written as a difference of two functions in $\#\mathbf{para}_{\beta\mathbf{tail}}\mathbf{L}$, and is $\#\mathbf{para}_{\beta\mathbf{tail}}\mathbf{L}$-hard with respect to $\leq^{\text{plog}}_{\text{met}}$.*

5 Conclusions and Outlook

We developed foundations for the study of parameterised space complexity of counting problems. Our results show interesting characterisations for classes defined in terms of k-bounded para-logspace NTMs. We believe that our results will lead to further research of parameterised logspace counting complexity. Notice, that the studied walk problems in Section 4.1 can be considered restricted to DAGs yielding the same completeness results.

Branching programs are immanent for the study of space-bounded and parallel complexity classes. Languages accepted by polynomial-size logspace-uniform branching programs characterise **NL**. In fact, this result carries forward to the counting versions. Motivated by this, one can consider parameterised counting classes based on deterministic branching programs (DBPs) and nondeterministic branching programs (BPs). It can be shown that for any $o \in \{\mathbf{W}, \mathbf{W}[1], \beta, \beta\text{-}\mathbf{tail}\}$, $\#\mathbf{para}_o\text{-}\mathbf{L}$ and $\#\mathbf{para}_o\text{-}\mathbf{NL}$, can be characterised in terms of an adequate parameterised counting version of DBPs and BPs, respectively (see the technical report [36]).

Comparing our newly introduced classes with the **W**-hierarchy (which is defined in terms of weighted satisfiability problems for circuits of a so-called bounded weft), one might ponder whether there is an alternative definition of our classes with such circuit problems. Though in this article we did not explore the weighted satisfiability, the closely related problem p-MC(Σ_0) sheds some light on this. Theorem 23 shows that p-MC(Σ_0) is complete for $\mathbf{para}_{\mathbf{W}[1]}\mathbf{L}$ (in fact, we show this for their counting versions) under $\leq_m^{\text{plog}}$-reductions. However, if we take **FPT**-reductions, p-MC(Σ_0) is complete for $\mathbf{W}[1]$. Though we could not prove it so far, we believe this is a general phenomenon: Any $\mathbf{W}[1]$-complete problem is complete for $\mathbf{para}_{\mathbf{W}[1]}\mathbf{L}$ under $\leq_m^{\text{plog}}$-reductions. More generally, there is a possibility that the FPT-closure of $\mathbf{para}_W\mathbf{L}$-classes is equal to the corresponding class in the **W**-hierarchy.

One might also ask the question if $\mathbf{para}_W\mathbf{L}$ is contained in **FFPT**. This is unlikely based on the view expressed above. For example, p-MC(Σ_0) is complete for both $\mathbf{para}_{\mathbf{W}[1]}\mathbf{L}$ and $\mathbf{W}[1]$ but under two different reductions. As a result, $\mathbf{para}_W\mathbf{L} \subseteq \mathbf{FFPT}$ would imply that p-MC(Σ_0) $\in$ **FPT** and, accordingly, $\mathbf{FPT} = \mathbf{W}[1]$ as **FPT** is closed under **FPT**-reductions. We close with interesting tasks for further research:

- Study further closure properties of the new classes (e.g., Open Problem 13).
- Improve the understanding of the influence of syntactic locality, resp., bounded arity in the setting of p-#MC(Σ_0) (Open Problem 24).
- Find a characterisation of the $\leq_{\text{pars}}^{\text{plog}}$-closure of $\#\mathbf{para}_{\mathbf{W}[1]}\mathbf{L}$ (Open Problem 25).
- Identify a natural class of structures for which the homomorphism problem is $\#\mathbf{para}_{\mathbf{W}}\mathbf{L}$-complete (Open Problem 29).
- Establish a broader spectrum of complete problems for the classes $\mathbf{para}_\beta\mathbf{L}$ and $\mathbf{para}_W\mathbf{L}$, e.g., in the realm of satisfiability questions.
- Identify further characterisations of the introduced classes, e.g., in the vein of descriptive complexity, which could highlight their robustness.
- Study gap classes [29] based on our classes. This might help improve Theorem 33.

References

1 Manindra Agrawal, Eric Allender, and Samir Datta. On TC^0, AC^0, and arithmetic circuits. *J. Comput. Syst. Sci.*, 60(2):395–421, 2000.

2 Eric Allender, Robert Beals, and Mitsunori Ogihara. The complexity of matrix rank and feasible systems of linear equations. *Computational Complexity*, 8(2):99–126, 1999.

3 Eric Allender and Mitsunori Ogihara. Relationships among PL, #L, and the determinant. *ITA*, 30(1):1–21, 1996.

4 Noga Alon, Raphael Yuster, and Uri Zwick. Color-coding. *J. ACM*, 42(4):844–856, 1995.

5 Sanjeev Arora and Boaz Barak. *Computational Complexity - A Modern Approach*. Cambridge University Press, 2009.

6 Max Bannach, Christoph Stockhusen, and Till Tantau. Fast parallel fixed-parameter algorithms via color coding. In *IPEC*, volume 43 of *LIPIcs*, pages 224–235. Schloss Dagstuhl – Leibniz-Zentrum fuer Informatik, 2015.

7 Richard Beigel and Bin Fu. Circuits over PP and PL. *J. Comput. Syst. Sci.*, 60(2):422–441, 2000.

8 Ralph Bottesch. Relativization and interactive proof systems in parameterized complexity theory. In *IPEC*, volume 89 of *LIPIcs*, pages 9:1–9:12. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017.

9 Ralph Christian Bottesch. On W[1]-hardness as evidence for intractability. In *MFCS*, volume 117 of *LIPIcs*, pages 73:1–73:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018.

10 Cornelius Brand and Marc Roth. Parameterized counting of trees, forests and matroid bases. In *CSR*, volume 10304 of *Lecture Notes in Computer Science*, pages 85–98. Springer, 2017.

11 Jonathan F. Buss. Relativized alternation and space-bounded computation. *J. Comput. Syst. Sci.*, 36(3):351–378, 1988.

12 Hervé Caussinus, Pierre McKenzie, Denis Thérien, and Heribert Vollmer. Nondeterministic NC^1 computation. *J. Comput. Syst. Sci.*, 57(2):200–212, 1998.

13 Ankit Chauhan and B. V. Raghavendra Rao. Parameterized analogues of probabilistic computation. In *CALDAM*, volume 8959 of *Lecture Notes in Computer Science*, pages 181–192. Springer, 2015.

14 Hubie Chen and Moritz Müller. The fine classification of conjunctive queries and parameterized logarithmic space. *TOCT*, 7(2):7:1–7:27, 2015.

15 Yijia Chen and Jörg Flum. Some lower bounds in parameterized AC^0. In *MFCS*, volume 58 of *LIPIcs*, pages 27:1–27:14. Schloss Dagstuhl – Leibniz-Zentrum fuer Informatik, 2016.

16 Yijia Chen and Jörg Flum. Some lower bounds in parameterized AC^0. *CoRR*, abs/1606.08014, 2016. `arXiv:1606.08014`.

17 Yijia Chen, Jörg Flum, and Martin Grohe. Bounded nondeterminism and alternation in parameterized complexity theory. In *Computational Complexity Conference*, pages 13–29. IEEE Computer Society, 2003.

18 Radu Curticapean. Counting matchings of size k is W[1]-hard. In *ICALP (1)*, volume 7965 of *Lecture Notes in Computer Science*, pages 352–363. Springer, 2013.

19 Radu Curticapean. *The simple, little and slow things count: on parameterized counting complexity*. PhD thesis, Saarland University, 2015.

20 Radu Curticapean. Block interpolation: A framework for tight exponential-time counting complexity. *Inf. Comput.*, 261(Part):265–280, 2018.

21 Radu Curticapean, Holger Dell, and Dániel Marx. Homomorphisms are a good basis for counting small subgraphs. In *STOC*, pages 210–223. ACM, 2017.

22 Víctor Dalmau and Peter Jonsson. The complexity of counting homomorphisms seen from the other side. *Theor. Comput. Sci.*, 329(1-3):315–323, 2004.

23 Carsten Damm. DET = $L^{\#L}$? Technical report, Informatik-Preprint 8, Fachbereich Informatik der Humboldt-Universitlt zu Berlin, 1991.

24 Samir Datta, Meena Mahajan, B. V. Raghavendra Rao, Michael Thomas, and Heribert Vollmer. Counting classes and the fine structure between NC^1 and L. *Theor. Comput. Sci.*, 417:36–49, 2012.

25 Rodney G. Downey and Michael R. Fellows. *Parameterized Complexity*. Monographs in Computer Science. Springer, 1999.

26 Martin E. Dyer and Catherine S. Greenhill. The complexity of counting graph homomorphisms. *Random Struct. Algorithms*, 17(3-4):260–289, 2000.

27 Heinz-Dieter Ebbinghaus, Jörg Flum, and Wolfgang Thomas. *Mathematical logic (2. ed.)*. Undergraduate texts in mathematics. Springer, 1994.

28 Michael Elberfeld, Christoph Stockhusen, and Till Tantau. On the space and circuit complexity of parameterized problems: Classes and completeness. *Algorithmica*, 71(3):661–701, 2015.

29 Stephen A. Fenner, Lance Fortnow, and Stuart A. Kurtz. Gap-definable counting classes. *J. Comput. Syst. Sci.*, 48(1):116–148, 1994.

30 Jörg Flum, Markus Frick, and Martin Grohe. Query evaluation via tree-decompositions. *J. ACM*, 49(6):716–752, 2002.

31 Jörg Flum and Martin Grohe. Describing parameterized complexity classes. *Inf. Comput.*, 187(2):291–319, 2003.

32 Jörg Flum and Martin Grohe. The parameterized complexity of counting problems. *SIAM J. Comput.*, 33(4):892–922, 2004.

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

34 Martin Grohe. The complexity of homomorphism and constraint satisfaction problems seen from the other side. *J. ACM*, 54(1):1:1–1:24, 2007.

35 Anselm Haak, Juha Kontinen, Fabian Müller, Heribert Vollmer, and Fan Yang. Counting of teams in first-order team logics. In *MFCS*, volume 138 of *LIPIcs*, pages 19:1–19:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019.

36 Anselm Haak, Arne Meier, Om Prakash, and B. V. Raghavendra Rao. Parameterised counting in logspace. *CoRR*, abs/1904.12156, 2019. `arXiv:1904.12156`.

37 Mark Jerrum and Kitty Meeks. The parameterised complexity of counting even and odd induced subgraphs. *Combinatorica*, 37(5):965–990, 2017.

38 Meena Mahajan and V. Vinay. Determinant: Combinatorics, algorithms, and complexity. *Chicago J. Theor. Comput. Sci.*, 1997, 1997.

39 Catherine McCartin. Parameterized counting problems. In *MFCS*, volume 2420 of *Lecture Notes in Computer Science*, pages 556–567. Springer, 2002.

40 Mitsunori Ogihara. The PL hierarchy collapses. *SIAM J. Comput.*, 27(5):1430–1437, 1998.

41 Michael Sipser. *Introduction to the theory of computation*. PWS Publishing Company, 1997.

42 Christoph Stockhusen. *On the space and circuit complexity of parameterized problems*. PhD thesis, University of Lübeck, Germany, 2017. URL: `http://www.zhb.uni-luebeck.de/epubs/ediss1780.pdf`.

43 Christoph Stockhusen and Till Tantau. Completeness results for parameterized space classes. In *IPEC*, volume 8246 of *Lecture Notes in Computer Science*, pages 335–347. Springer, 2013.

44 Seinosuke Toda. Counting problems computationally equivalent to the determinant. Technical Report CSIM 91-07, Dept. Comp. Sci. and Inf. Math., Univ. of Electro-Communications, Tokyo, 1991. URL: `http://perso.ens-lyon.fr/natacha.portier/papers/toda91.pdf`.

45 Seinosuke Toda. PP is as hard as the polynomial-time hierarchy. *SIAM J. Comput.*, 20(5):865–877, 1991.

46 Leslie G. Valiant. The complexity of computing the permanent. *Theor. Comput. Sci.*, 8:189–201, 1979.

47 V. Vinay. Counting auxiliary pushdown automata and semi-unbounded arithmetic circuits. In *Proceedings of 6th Structure in Complexity Theory Conference*, pages 270–284, 1991.