# Towards a Characterization of Counting and Alternating Classes via Discrete Ordinary Differential Equations: Ongoing Research Report

Melissa Antonelli[1], Eduardo Skapinakis[1,2]

[1]Carl Friedrich von Weizsäcker-Zentrum, Universität Tübingen, Germany
[2]Center for Mathematics and Applications (NOVA Math), NOVA FCT, Portugal

**Abstract**

This paper presents a high-level report on an ongoing project aiming to leverage implicit approaches based on discrete ordinary differential equations (ODEs) to study multiple complexity classes, even beyond small circuit and polynomial-time classes. Stimulated by recent ODE-based characterizations of polynomial-time functions ($\mathbf{FP}$) and classes over the reals, the research project outlined here pushes this investigation further into counting and alternation. Specifically, we present a uniform framework, built upon a single base algebra and a unified family of schemas, where complexity levels, such as those of the polynomial and counting hierarchies, are captured simply by the nesting depth of ODE operators. Crucially, our approach starts from a base class much weaker than $\mathbf{FP}$, thus strengthening existing recursion-theoretic treatments and establishing a natural connection to descriptive complexity. Moreover, by isolating three elementary schemas, our framework makes the computational content of linearity restrictions completely transparent while extending ODE-based implicit complexity to previously unaddressed counting classes, such as $\oplus\mathbf{P}$. More generally, this work establishes a clear bridge between differentiation and counting, offering a fresh perspective on the relationships between different complexity classes, which remains the object of ongoing and future research.

**Keywords**

Implicit complexity, function algebras, alternating and counting classes, discrete differential equations

## 1. Introduction

In recent years, the use of ordinary differential equations (ODEs) as a framework to design algorithms and characterize various complexity classes has seen a resurgence of interest. Given their inherent connections to analog computation, ODEs have been used to characterize complexity over the reals [1, 2, 3], and, more recently, their discrete counterparts have also been shown to handle standard computational models [4, 5]. This establishes the ODE framework as a promising and versatile tool for machine-independent complexity, the branch of complexity theory dedicated to characterizing functions without relying on specific machines or cost models (see e.g. [6, 7, 8, 9, 10, 11]).

The seminal work introducing *discrete* ODEs in complexity established a characterization for the class of poly-time computable functions ($\mathbf{FP}$) [4]. In a series of subsequent works [5, 12, 13], this approach has also proven fruitful for studying small circuit classes, revealing one of the key advantages of ODEs: the interplay of multiple parameters (such as discrete vs. continuous inputs, the choice of functions to derive along, the syntactical form of the equations, and constraints on their calls) enables a flexible characterization of (circuit) computational resources. The research project delineated here aims to push this investigation further by exploring the expressive power of discrete ODEs in a different direction, namely by considering classes *above* $\mathbf{FP}$. This proves the versatility of this mathematical tool by showing that they can naturally capture resources related to different machine models, such as nondeterministic and counting machines, providing a comprehensive and new perspective on complexity.

## 2. A Brief Review of Discrete ODEs in Complexity

As is standard, we define ODEs as expressions of the form $\frac{\partial f(x,\mathbf{y})}{\partial x} = h(x, \mathbf{y}, f(x, \mathbf{y}))$, where $\frac{\partial f(x,\mathbf{y})}{\partial x}$ is the partial derivative of $f(x, \mathbf{y})$ with respect to $x$ (being $\mathbf{y}$ fixed). When an initial condition $f(0, \mathbf{y}) =$

*ICTCS 2026: Italian Conference on Theoretical Computer Science, September 7–9, 2026, Udine, Italy*
melissa.antonelli@uni-tuebingen.de (M. Antonelli); eduardo.skapinakis@uni-tuebingen.de (E. Skapinakis)
0009-0006-9072-4847 (M. Antonelli); 0000-0002-5729-1078 (E. Skapinakis)

$g(\mathbf{y})$ is added, we obtain a so-called *initial value problem* (IVP). Here, we focus on the discrete counterpart of ODEs, such that the discrete derivative of $f(x)$ is defined as $f(x+1)-f(x)$. In order to capture *complexity* classes two novel concepts are introduced into the standard setting of difference calculus: (a) the idea of deriving along functions, enabling to control the number of recursion steps and (b) a syntactic form of the equation enabling to control the object size. These two simple features allow for the definition of schemas which do not impose explicit bounds on the recursion (as in [6]) or assign special roles to variables (as in [14, 15, 7]). Formally, (a) is realized through the notion of $\lambda$-ODE: a function $f$ is defined by $\lambda$-ODE from $g$ and $u$ if it is the solution of the IVP with initial value $f(0,\mathbf{y}) = g(\mathbf{y})$ and s.t. $\frac{\partial f(x,\mathbf{y})}{\partial \lambda} = \frac{\partial f(x,\mathbf{y})}{\partial \lambda(x,\mathbf{y})} = u(x,\mathbf{y},f(x,\mathbf{y}))$, being a formal synonym of $f(x+1,\mathbf{y}) = f(x,\mathbf{y}) + \big(\lambda(x+1,\mathbf{y}) - \lambda(x,\mathbf{y})\big) \times u(x,\mathbf{y},f(x,\mathbf{y}))$. This ensures that the value of $f(x,\mathbf{y})$ changes only when the value of $\lambda(x,\mathbf{y})$ does, thereby linking the number of recursion steps directly to the growth rate of the function along which we are deriving. When $\lambda$ is the length function $\ell$ (s.t. $\ell(0)=0$ and $\ell(x) = \lceil log_2(x+1)\rceil$) this is called length-ODE.

To control (b) function growth, we apply linearity-based syntactical restrictions; these ensure that the output of recursive calls is strictly bounded at each step, preventing growth explosions.

**Definition 1** (Linear $\lambda$-ODE)**.** *Given $g,\lambda,h$, the function $f$ is said to be defined by $\lambda$-ODE from $g$ and $h$ if it is the solution of the IVP with initial value $f(0,\mathbf{y}) = g(\mathbf{y})$ and such that: $\frac{\partial f(x,\mathbf{y})}{\partial \lambda} = A(x,\mathbf{y},f(x,\mathbf{y}),h(x,\mathbf{y})) \times f(x,\mathbf{y}) + B(x,\mathbf{y},f(x,\mathbf{y}),h(x,\mathbf{y}))$ where $A$ and $B$ are expressions over the signature $\{+,-,\times,\mathsf{sg}\}$ (being $\mathsf{sg}:\mathbb{Z}\to\mathbb{Z}$ the sign function over $\mathbb{Z}$ taking value 1 for $x>0$ and 0 otherwise) in which $f$ occurs only under the scope of the sign function. If no call to $f$ occurs in $A$ and $B$, the schema is said to be* strict.

So-called linear length ODE, $\ell$-ODE, is defined by putting these two features together. This schema was the crucial novelty introduced to capture poly-time computation, offering the very first characterization of $\mathbf{FP}$ in terms of $\mathbb{LDL}$, an ODE-based function algebra made of standard basic functions $(0,1,\pi_i^p,\ell,\mathsf{sg},+,-,\times)$ and closed under composition ($\circ$) and $\ell$-ODE [4]. Subsequently, it was shown that discrete ODEs also provide a natural tool for expressing core operations of circuit-based computation, such as basic right-shifting or left-shifting with bit addition. This led to multiple original characterizations for classes ranging from those computable by unbounded Boolean circuits of polynomial size and constant depth ($\mathbf{FAC}^0$) to those of logarithmic depth ($\mathbf{FNC}^1$), also including various counting gates [5, 12, 13]; notably, most of these latter classes had hitherto lacked implicit characterizations. Recall in particular that $\mathbf{FAC}^0$ was characterized by a function algebra called $\mathbb{ACDL}$, defined by standard basic functions and closed under composition and under a special schema allowing for iterated left-shifting, with possible bit addition (see [5, 12]).

Motivated by the simplicity with which ODE schemas capture crucial operations like (sharply) bounded search and counting, our project aims to extend this approach to complexity beyond $\mathbf{FP}$, thereby simplifying and unifying the characterizations of these classes as well.

## 3. Uniformly Reasoning about Computation Paths

In recursion theory, complexity classes related to notions of *alternation* and *counting* have been mostly captured by forms of *tree recursion* with parameter substitution (see, e.g. [7, 16, 17, 18, 19]) or restricted forms of *primitive recursion* (see, e.g. [20, 21, 22]). In both cases, one uses recursion schemes that mimic the traversal of a *computation tree*: from its root to its leaves for tree recursion (on notation), and from the last leaf back to the root for primitive recursion.

This approach to recursion can be naturally viewed as a form of discrete differentiation. Thus, our key strategy is to combine the recognition of accepting computation paths (already doable in $\mathbf{FAC}^0$, [23, Th. 1]) with mechanisms for searching and counting over them, tools precisely offered by ODE-schemas (Def. 1). More generally, this framework makes it possible to capture a wide spectrum of classes, ranging from small circuits to polynomial space, by relying on *the same function algebra* and by introducing two simple modifications to the core defining schemas: (A) a restricted form of composition (Def. 2) and (B) specific constraints on the linearity of ODEs.

**A. Restricted composition.** As is standard in recursion theory [6, 24], our characterizations are in terms of function algebras. However, in order to uniformly capture hierarchy levels, we rely on a special form of restricted composition, inspired by [25, 19, 18]. This addresses the fact that classes like $\mathbf{NP}$ are not known to be closed under complement, and thus also not under composition.

**Definition 2.** *Given a (base) algebra $\mathscr{B}$ and a set of operators $\mathcal{O}_1, \dots, \mathcal{O}_k$, the algebra $\mathscr{F} = [\mathscr{B}; \circ_0, \mathcal{O}_1, \dots, \mathcal{O}_k]$ is defined as the standard algebra $[\mathscr{B}; \circ, \mathcal{O}_1, \dots, \mathcal{O}_k]$ but with $f$ being obtained by composition, $f(x, \mathbf{y}) = h(\mathbf{g}(x, \mathbf{y}))$, only when all (inner) functions in $\mathbf{g}$ are in $\mathscr{B}$.*

If the base class $\mathscr{B}$ is itself defined via operators $\mathcal{O}'_1, \cdots, \mathcal{O}'_m$, these cannot be used in $[\mathscr{B}; \circ, \mathcal{O}_1, \dots, \mathcal{O}_k]$, i.e., one has access to functions in $\mathscr{B}$, but not to operators defining them. This restriction allows for fine-grained control over the *nested* application of certain schemas.

**B. Some paradigmatic ODE schemas.** We consider strict ODE-based schemas obtained by imposing various restrictions on the linearity of the defining equation. Specifically, when $A$ takes values only in $\{-1, 0\}$ and $B = 0$, the computation corresponds to universal bounded search; when $A = -B$ and $B$ takes values in $\{0, 1\}$, it corresponds to existential bounded search; if $A = 0$ and $B$ takes values in $\{0, 1\}$, counting is obtained.

**Definition 3.** *Let $g : \mathbb{N}^p \to \{0, 1\}, k_1 : \mathbb{N}^{p+1} \to \{-1, 0\}$ and $k_2 : \mathbb{N}^{p+1} \to \{0, 1\}$. The function $f : \mathbb{N}^{p+1} \to \mathbb{N}$ defined as the solution of the IVP with initial value $f(0, \mathbf{y}) = g(\mathbf{y})$ and s.t.: $\frac{\partial f(x,\mathbf{y})}{\partial x} = f(x, \mathbf{y}) \times k_1(x, \mathbf{y})$ is said to be defined by ODE$_\wedge$; $\frac{\partial f(x,\mathbf{y})}{\partial x} = -f(x, \mathbf{y}) \times k_2(x, \mathbf{y}) + k_2(x, \mathbf{y})$ is said to be defined by ODE$_\vee$; $\frac{\partial f(x,\mathbf{y})}{\partial x} = k_2(x, \mathbf{y})$ is said to be defined by ODE$_\#$.*

It is easy to check that solutions of the corresponding systems precisely correspond to the desired operations: universal and existential bounded search, and counting ([23, Prop. 1]).

**C. Recognizing accepting computation paths.** Typical machine-independent characterizations of non-deterministic classes with polynomial resource restrictions ($\mathbf{NP}$, $\#\mathbf{P}$, ...) are defined having as the base class $\mathbf{P}$ or $\mathbf{FP}$. Thus, having at our disposal ODE schemas which mimic the operations required to define these classes, a natural approach to characterize them would be to close the algebra of functions $\mathbb{LDL}$ [4] under these schemas. Here we show that, in fact, we can use a much weaker class as our basis, namely $\mathbb{ACDL}$ [5]. Indeed, while characterizations in [19, 18] rely on schemas to represent the non-deterministic bits generated by a non-deterministic Turing machine (NTM) $M$, we instead use them to generate the entire computation of $M$. Then, what we need is simply a mechanism allowing us to check whether the generated words correspond to accepting computations of $M$, which can be done already in $\mathbf{FAC}^0$ (for a self-contained proof see [23, Sec. A]; similar formulations of this result have been proved in [26, Th. 3.1] and [9, Th. 7.8]). The core intuition behind this result is that, under a suitable encoding, one can interpret a computation path of a machine $M$ as a binary word. For each machine $M$, we can define the set $\mathtt{acc}_M = \{(z, x) \in \mathbb{N}^2 : z \text{ is an accepting computation of } M \text{ on input } x\}$. Our proof that for any (N)TM $M$ and input $x$, there is an $\mathbf{FAC}^0$ function deciding $\mathtt{acc}_M$ (see [23, Th. 1]) partially strengthens the state of the art [26, 25, 19, 18] and establishes a clear(er) bridge between the ODE-based approach and descriptive complexity, where it is also FO ($= \mathbf{AC}^0$) which serves as the basis for the characterization of complexity classes [8, 27, 9].

## 4. Capturing Classes Beyond $\mathbf{FP}$ via Discrete ODEs: an Overview

Simply combining $\mathbb{ACDL}$ and restricted composition with ODE$_\wedge$ or ODE$_\vee$ is enough to capture the base levels of $\mathbf{FPH}$. For example, the equality $\mathbf{FAC}^0 \cup \mathbf{NP} = [\mathbb{ACDL}; \circ_0, \text{ODE}_\vee]$ is established by relying on the closure properties of $\circ_0$ and ODE$_\vee$ for ($\supseteq$). Conversely, ($\subseteq$) follows from the fact that for any poly-time $\mathbf{NP}$-machine $M$, letting $V(z, x) \in \mathbb{ACDL}$ be the function deciding the set $\mathtt{acc}_M$ and letting

$r$ be a polynomial such that for any $x$, the encoding of an accepting computation of $M(x)$ has *precisely* size $r(\ell(x))$, we can define $f$ via $\text{ODE}_\vee$ from $0$ and $V$ so that $f(1\#r(\ell(x)), x) = 1$ if and only if there exists a word $z$ of size $r(\ell(x))$ for which $V(z, x) = 1$ (see [23, Th. 2]). The class $\mathbf{FAC}^0 \cup \mathbf{coNP}$ is captured in an analogous way, by considering $\text{ODE}_\wedge$, while subsequent levels of $\mathbf{PH}$ are characterized by iteratively considering the previous level as the basic function class (for details see [23, Sec. C]).

A similar approach extends to the study of the counting classes [28] and hierarchy [21] ($\mathbf{FCH}$) and its levels, where the crucial schema is $\text{ODE}_\#$, i.e., a schema that allows one to "count" paths. Intuitively, completeness direction shows that if $f \in \#\mathbf{P}$, i.e. if there is a poly-time NTM $M$ s.t., for every $x$, $f(x) = \#\mathtt{acc}_{M,x}$, we can use $\text{ODE}_\#$ to define $F(r(x), x) = f(x)$ using $g(z) = 0$ and $V(z, x) \in \mathbb{ACDL}$ which returns $1$ iff $z$ is an accepting computation of $M$ on $x$. Notably, the flexibility of this approach allows us to straightforwardly capture other counting classes never characterized before, such as $\oplus\mathbf{P}$ or $\mathbf{FMod}_\text{m}\mathbf{P}$, by considering, again, $\mathbb{ACDL}$, $\circ_0$ but different-counting ODE schemas (e.g., a schema *counting modulo 2*, see [23, Sec. E]).

Other simple restrictions on linearity lead to ODEs with significant computational power. Although this investigation is still in progress, at least two of them seem especially worth mentioning. Endowing $\mathbb{ACDL}$ with an ODE schema that differentiates with respect to $x$ and such that $A = -1$ (or is in $\{0, 1\}$, see [23, Sec. F]) is enough to capture $\mathbf{FPSPACE}$. Observe that this schema closely mirrors the one defining $\mathbf{FNC}^1$, which instead differentiates with respect to $\ell$; a connection which deserves further investigation. Moving to derivations along $\ell$, an alternative characterization for $\mathbf{FP}$ can be obtained by considering a schema with generalized linearity, where $A = 1$ and $B$ takes values in $\{0, 1\}$ rather than $\ell$-ODE [4] (see [23, Sec. G]).

## 5. Perspectives

The paper presents a high-level overview of an ongoing project that aims to advance the study of complexity through the perspective of ODEs and clarify the power of these schemas, for instance, to capture alternating hierarchies. Developing clear ODE instruments and systematically investigating these characterizations is the precise focus of the research sketched here.

This investigation is relevant for multiple reasons. First, it further exemplifies how flexible ODEs can be: besides circuits and dynamical systems, ODEs are able to capture resources associated with non-determinism, relativized computation and counting machines. Additionally, while most of the classes mentioned here have already been characterized using standard tools available in recursion theory, the ODE perspective offers for the first time a *unique* mathematical object capable of characterizing diverse aspects of computation, spanning both the discrete and continuous domains. To our knowledge, such universality is not offered by any previously developed tools in recursion theory. The versatility of this framework is made possible by the variety of features defining ODEs (e.g., strict vs. non-strict, functions to derive along, linearity, etc.). Clarifying the relationship between these constraints remains an open question for future research, potentially bridging other approaches to implicit complexity, e.g., by comparing how computational resources are captured by constraints on ODEs vs. logical features [8, 27, 9]. Finally, far from being just another way to rewrite recursion schemas, this framework is capable of straightforwardly capturing classes never characterized before in recursion theory, as shown in [13] or here for $\oplus\mathbf{P}$. We are thus confident that these tools can also provide original characterizations for other relevant classes (e.g., involving randomness [29] or monotonicity [30]) and offer new means to study the relationships between them.

| | deriving along $x$ | | | deriving along $\ell$ | | |
|---|---|---|---|---|---|---|
| | $\mathbf{FAC}^0 \cup \mathbf{PH}$ | $\mathbf{FCH}$ | $\mathbf{FPSPACE}$ | $\mathbf{FTC}^0$ | $\mathbf{FNC}^1$ | $\mathbf{FP}$ |
| lev. 1 | $\circ_0$ ; $-f \times k_2 + k_2$ | $\circ_0$ ; $k_2$ | | | | $f \times A + B$ |
| lev. $n$ | $\circ_n$ ; $-f \times k_2 + k_2$ | $\circ_n$ ; $k_2$ | $-f + B$ | $k_2$ | $-f + B$ | |
| union | $-f \times k_2 + k_2$ | $k_2$ | | | | $f + K$ |

The table summarizes the defining features of ODE schemas characterizing different classes and hierarchy levels. These characterizations are obtained by closing $\mathbb{ACDL}$ or the preceding hierarchy level (denoted by $\circ_n$) under (possibly restricted, $\circ_0$) composition and the given ODE schema. Whenever composition is not restricted, $\circ$ is simply omitted. We follow the convention that $k_1 : \mathbb{N} \to \{-1, 0\}$, $k_2 : \mathbb{N} \to \{0, 1\}$, and $B$ (resp., $K$) is a (resp, generalized) $\mathsf{sg}$-polynomial expression, s.t. $f$ only occurs under the scope of $\mathsf{sg}$ (resp., takes values in $\{0, 1\}$). The schema for $\mathbf{FNC}^1$ is s.t. $f$ occurs in $B$ only in expressions of the form $\mathsf{sg}(f - c)$, with $c \in \mathbb{N}$ constant, see [12].

## Declaration on Generative AI

The authors have not employed any Generative AI tools.

## References


[1] O. Bournez, D. S. Graça, A. Pouly, Polynomial time corresponds to solutions of polynomial ordinary differential equations of polynomial length: The general purpose analog computer and computable analysis are two efficiently equivalent models of computations, in: I. Chatzigiannakis, M. Mitzenmacher, Y. Rabani, D. Sangiorgi (Eds.), 43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, Rome, Italy, July 11-15, 2016, volume 55 of *LIPIcs*, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016, pp. 109:1–109:15.

[2] O. Bournez, D. S. Graça, A. Pouly, Polynomial time corresponds to solutions of polynomial ordinary differential equations of polynomial length, J. ACM 64 (2017) 38:1–38:76.

[3] M. Blanc, O. Bournez, A characterisation of functions computable in polynomial time and space over the reals with discrete ordinary differential equations: Simulation of turing machines with analytic discrete ODEs, in: J. Leroux, S. Lombardy, D. Peleg (Eds.), 48th International Symposium on Mathematical Foundations of Computer Science, MFCS 2023, Bordeaux, France, August 28 - September 1, 2023, volume 272 of *LIPIcs*, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023, pp. 21:1–21:15.

[4] O. Bournez, A. Durand, Recursion schemes, discrete differential equations and characterization of polynomial time computations, in: P. Rossmanith, P. Heggernes, J. Katoen (Eds.), 44th International Symposium on Mathematical Foundations of Computer Science, MFCS 2019, Aachen, Germany, August 26-30, 2019, volume 138 of *LIPIcs*, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019, pp. 23:1–23:14.

[5] M. Antonelli, A. Durand, J. Kontinen, A new characterization of $\mathrm{fac}^0$ via discrete ordinary differential equations, in: R. Královic, A. Kucera (Eds.), 49th International Symposium on Mathematical Foundations of Computer Science, MFCS 2024, Bratislava, Slovakia, August 26-30, 2024, volume 306 of *LIPIcs*, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024, pp. 10:1–10:18.

[6] A. Cobham, The intrinsic computational difficulty of functions, in: Y. Bar-Hillel (Ed.), Logic, Methodology and Philosophy of Science II, North-Holland, 1965, pp. 24–30.

[7] D. Leivant, J.-Y. Marion, Ramified recurrence and computational complexity II: substitution and poly-space, in: International Workshop on Computer Science Logic, Springer, 1994, pp. 486–500.

[8] R. Fagin, Generalized first-order spectra and polynomial-time recognizable sets, Complexity of computation 7 (1974) 43.

[9] N. Immerman, Descriptive complexity, Springer Science & Business Media, 2012.

[10] S. R. Buss, Bounded Arithmetic, Ph.D. thesis, University of California, Berkeley, 1985.

[11] J.-Y. Girard, A. Scedrov, P. J. Scott, Bounded linear logic: a modular approach to polynomial-time computability, Theoretical Computer Science 97 (1992) 1–66.

[12] M. Antonelli, A. Durand, J. Kontinen, Characterizing small circuit classes from $\mathrm{FAC}^0$ to $\mathrm{FAC}^1$ via discrete ordinary differential equations, in: 50th International Symposium on Mathematical Foundations of Computer Science, 2025.

[13] M. Antonelli, A. Durand, R. Li, Recursion and proof theoretical characterizations of small circuit classes with modulo counting via discrete differential equations, in: Proceedings of the International Colloquium on Automata, Languages, and Programming (ICALP), 2026, accepted.

[14] S. Bellantoni, S. Cook, A new recursion-theoretic characterization of poly-time functions, Computational Complexity 2 (1992) 97–110.

[15] D. Leivant, A foundational delineation of poly-time, Information and Computation 110 (1994) 391–420.

[16] D. Leivant, J.-Y. Marion, A characterization of alternating log time by ramified recurrence, Theoretical Computer Science 236 (2000) 193–208.

[17] I. Oitavem, Characterizing PSPACE with pointers, Mathematical Logic Quarterly 54 (2008) 323–329.
[18] U. Dal Lago, R. Kahle, I. Oitavem, Implicit recursion-theoretic characterizations of counting classes, Archive for Mathematical Logic 61 (2022) 1129–1144.
[19] I. Oitavem, The polynomial hierarchy of functions and its levels, Theoretical Computer Science 900 (2022) 25–34.
[20] D. B. Thompson, Subrecursiveness: Machine-independent notions of computability in restricted time and storage, Mathematical Systems Theory 6 (1972) 3–15.
[21] K. W. Wagner, Some observations on the connection between counting and recursion, Theoretical Computer Science 47 (1986) 131–147.
[22] A. M. Ben-Amram, B. Loff, I. Oitavem, Monotonicity constraints in characterizations of pspace, Journal of Logic and Computation 22 (2012) 179–195.
[23] E. S. Melissa Antonelli, Towards a characterization of counting and alternating classes via discrete ordinary differential equations, ???? Available at https://arxiv.org/abs/2608.05431.
[24] P. G. Clote, G. Takeuti, Bounded arithmetic for NC, ALogTIME, L and NL, Annals of Pure and Applied Logic 56 (1992) 73–117.
[25] I. Oitavem, A recursion-theoretic approach to np, Annals of Pure and Applied Logic 162 (2011) 661–666.
[26] H. Vollmer, K. W. Wagner, Recursion theoretic characterizations of complexity classes of counting functions, Theoretical Computer Science 163 (1996) 245–258.
[27] N. Immerman, Expressibility and parallel complexity, SIAM Journal on Computing 18 (1989) 625–638.
[28] L. G. Valiant, The complexity of computing the permanent, Theoretical Computer Science 8 (1979) 189–201.
[29] J. Gill, Computational complexity of probabilistic Turing machines, SIAM Journal of Computing 6 (1977) 675–695.
[30] M. Grigni, M. Sipser, Monotone complexity, in: Proceedings of the London Mathematical Society Symposium on Boolean Function Complexity, 1992, pp. 57–75.