The Existential Theory of the Reals with Summation Operators

Markus Bläser
Saarland University, Saarbrücken, Germany

Julian Dörfler
Saarland University, Saarbrücken, Germany

Maciej Liśkiewicz
University of Lübeck, Germany

Benito van der Zander
University of Lübeck, Germany

Abstract

To characterize the computational complexity of satisfiability problems for probabilistic and causal reasoning within Pearl's Causal Hierarchy, van der Zander, Bläser, and Liśkiewicz [IJCAI 2023] introduce a new natural class, named succ-$\exists\mathbb{R}$. This class can be viewed as a succinct variant of the well-studied class $\exists\mathbb{R}$ based on the Existential Theory of the Reals (ETR). Analogously to $\exists\mathbb{R}$, succ-$\exists\mathbb{R}$ is an intermediate class between NEXP and EXPSPACE, the exponential versions of NP and PSPACE.

The main contributions of this work are threefold. Firstly, we characterize the class succ-$\exists\mathbb{R}$ in terms of nondeterministic real Random-Access Machines (RAMs) and develop structural complexity theoretic results for real RAMs, including translation and hierarchy theorems. Notably, we demonstrate the separation of $\exists\mathbb{R}$ and succ-$\exists\mathbb{R}$. Secondly, we examine the complexity of model checking and satisfiability of fragments of existential second-order logic and probabilistic independence logic. We show succ-$\exists\mathbb{R}$-completeness of several of these problems, for which the best-known complexity lower and upper bounds were previously NEXP-hardness and EXPSPACE, respectively. Thirdly, while succ-$\exists\mathbb{R}$ is characterized in terms of ordinary (non-succinct) ETR instances enriched by exponential sums and a mechanism to index exponentially many variables, in this paper, we prove that when only exponential sums are added, the corresponding class $\exists\mathbb{R}^{\Sigma}$ is contained in PSPACE. We conjecture that this inclusion is strict, as this class is equivalent to adding a VNP-oracle to a polynomial time nondeterministic real RAM. Conversely, the addition of exponential products to ETR, yields PSPACE. Furthermore, we study the satisfiability problem for probabilistic reasoning, with the additional requirement of a small model, and prove that this problem is complete for $\exists\mathbb{R}^{\Sigma}$.

2012 ACM Subject Classification Theory of computation → Complexity classes

Keywords and phrases Existential theory of the real numbers, Computational complexity, Probabilistic logic, Models of computation, Existential second order logic

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2024.13

Related Version *Full Version*: https://arxiv.org/abs/2405.04697

Funding *Benito van der Zander*: Work supported by the Deutsche Forschungsgemeinschaft (DFG) grant 471183316 (ZA 1244/1-1).

1 Introduction

The existential theory of the reals, ETR, is the set of all true sentences of the form

$$\exists x_0 \dots \exists x_n \ \varphi(x_0, \dots, x_n), \tag{1}$$

where φ is a quantifier-free Boolean formula over the basis $\{\vee, \wedge, \neg\}$, variables $x_0, \dots, x_n$, and a signature consisting of the constants 0 and 1, the functional symbols $+$ and $\cdot$, and the relational symbols $<$, $\leq$, and $=$. The sentences are interpreted over the real numbers in the standard way. The significance of this theory lies in its exceptional expressiveness, enabling the representation of numerous natural problems across computational geometry [1, 20, 8], Machine Learning and Artificial Intelligence [2, 21, 31], game theory [4, 12], and various other domains. Consequently, a complexity class, $\exists\mathbb{R}$, has been introduced to capture the computational complexity associated with determining the truth within the existential theory of the reals. This class is formally defined as the closure of ETR under polynomial-time many-one reductions [14, 7, 25]. For a comprehensive compendium on $\exists\mathbb{R}$, see [26].

Our study focuses on ETR which extends the syntax of formulas to allow the use of summation operators in addition to the functional symbols $+$ and $\cdot$. This research direction, initiated in [31], was motivated by an attempt to accurately characterize the computational complexity of satisfiability problems for probabilistic and causal reasoning across "Pearl's Causal Hierarchy" (PCH) [28, 23, 3].

In [31], the authors introduce a new natural class, named $\mathsf{succ}\text{-}\exists\mathbb{R}$, which can be viewed as a succinct variant of $\exists\mathbb{R}$. Perhaps, one of the most notable complete problems for the new class is the problem, called Σ_{vi}-ETR ("*vi*" stands for variable indexing). It is defined as an extension of ETR by adding to the signature an additional summation operator[1] $\sum_{x_j=0}^{1}$ which can be used to *index* the quantified variables x_i used in Formula (1). To this end, the authors define variables of the form $x_{\langle x_{j_1},\dots,x_{j_m}\rangle}$, which represent indexed variables with the index given by $x_{j_1},\dots,x_{j_m}$ interpreted as a number in binary. They can only be used when variables $x_{j_1},\dots,x_{j_m}$ occur in the scope of summation operators with range $\{0,1\}$. E.g., $\exists x_0 \dots \exists x_{2^N-1} \sum_{e_1=0}^{1} \cdots \sum_{e_N=0}^{1} (x_{\langle e_1,\dots,e_N\rangle})^2 = 1$ is a Σ_{vi}-ETR sentence[2] encoding a unit vector in $\mathbb{R}^{2^N}$. Note that sentences of Σ_{vi}-ETR allow the use of exponentially many variables. Another example sentence is $\sum_{x_1=0}^{1}\sum_{x_2=0}^{1}(x_1+x_2)(x_1+(1-x_2))(1-x_1)=0$ that models the co-SAT instance $(p \vee q) \wedge (p \vee \overline{q}) \wedge \overline{p}$. It shows that the summation operator can also be used in Σ_{vi}-ETR formulas in a standard way.

Analogously to $\exists\mathbb{R}$, $\mathsf{succ}\text{-}\exists\mathbb{R}$ is an intermediate class between the exponential versions of NP and PSPACE:

$$\mathsf{NP} \subseteq \exists\mathbb{R} \subseteq \mathsf{PSPACE} \subseteq \mathsf{NEXP} \subseteq \mathsf{succ}\text{-}\exists\mathbb{R} \subseteq \mathsf{EXPSPACE}. \tag{2}$$

An interesting challenge, in view of the new class, is to determine whether it contains harder problems than NEXP and to examine the usefulness of $\mathsf{succ}\text{-}\exists\mathbb{R}$-completeness as a tool for understanding the apparent intractability of natural problems. A step in these directions, that we take in this work, is to express $\mathsf{succ}\text{-}\exists\mathbb{R}$ in terms of machine models over the reals, which in the case of $\exists\mathbb{R}$ yield an elegant and useful characterization by $\mathsf{NP}_{\text{real}}$ [10].

In our work, we study also the different restrictions on which the summation operators in ETR are allowed to be used and the computational complexity of deciding the resulting problems. In particular, we investigate $\exists\mathbb{R}^{\Sigma}$ – the class based on ETR enriched with standard summation operator, and $\mathsf{succ}\text{-}\exists\mathbb{R}_{\text{poly}}$ which is based on Σ_{vi}-ETR with the restriction that only polynomially many variables can be used.

In this paper, we employ a family of satisfiability problems for probabilistic reasoning, which nicely demonstrates the expressiveness of the ETR variants under consideration and illustrates the natural necessity of introducing the summation operator.

---

[1] In [31], the authors assume arbitrary integer lower and upper bound in $\sum_{x_j=a}^{b}$. It is easy to see that, w.l.o.g., one can restrict a and b to binary values.

[2] We represent the instances in Σ_{vi}-ETR omitting the (redundant) block of existential quantifiers, so the encoding of the example instance has length polynomial in N.

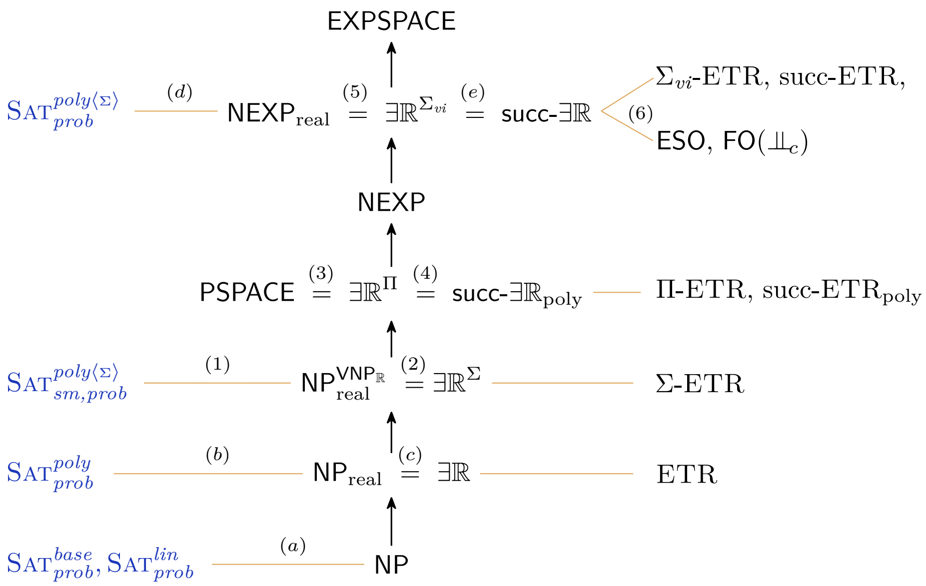

Figure 1 The landscape of complexity classes of the existential theories of the reals and the satisfiability problems for the probabilistic languages (to the left-hand side) complete for the corresponding classes. Arrows "→" denote inclusions ⊆ and the earth-yellow labeled lines "–" connect complexity classes with complete problems for those classes. The completeness results (a), (b), and (d) were proven by Fagin et al. [11], Mossé et al. [21], resp. van der Zander et al. [31]. The characterization (c) is due to Erickson et al. [10] and (e) is proven in [31]. The references to our results are as follows: (1) Theorem 26, (2) Theorem 24, (3) and (4) Theorem 19, (5) Lemma 2, and (6) Theorem 12.

Related Work to our Study

In a pioneering paper in this field, Fagin, Halpern, and Megiddo [11] consider the probabilistic language consisting of Boolean combinations of (in)equalities of *basic* and *linear* terms, like $\mathbb{P}((X{=}0 \vee Y{=}1) \wedge (X{=}0 \vee Y{=}0)){=}1 \wedge (\mathbb{P}(X{=}0){=}0 \vee \mathbb{P}(X{=}0){=}1) \wedge (\mathbb{P}(Y{=}0){=}0 \vee \mathbb{P}(Y{=}0){=}1)$, over binary variables X, Y (which can be seen as a result of reduction from the satisfied Boolean formula $(\overline{a} \vee b) \wedge (\overline{a} \vee \overline{b})$). The authors provide a complete axiomatization for the used logic and investigate the complexity of the probabilistic satisfiability problems SAT^{base}_{prob} and SAT^{lin}_{prob}, which ask whether there is a joint probability distribution of $X, Y, \ldots$ that satisfies a given Boolean combination of (in)equalities of basic, respectively linear, terms (for formal definitions, see Sec. 2). They show that both satisfiability problems are NP-complete (cf. Fig. 1). Thus, surprisingly, the complexity is no worse than that of propositional logic. Fagin et al. extend then the language to (in)equalities of *polynomial* terms, with the goal of reasoning about conditional probabilities. They prove that there is a PSPACE algorithm, based on Canny's decision procedure [7], for deciding if such a formula is satisfiable but left the exact complexity open. Recently, Mossé, Ibeling, and Icard, [21] have solved this problem, showing that deciding the satisfiability (SAT^{poly}_{prob}) is $\exists\mathbb{R}$-complete. In [21], the authors also investigate the satisfiability problems for the higher, more expressive PCH layers – which are not the subject of our paper – and prove an interesting result, that for (in)equalities of polynomial terms both at the interventional and the counterfactual layer the decision problems still remain $\exists\mathbb{R}$-complete.

The languages used in [11, 21] and also in other relevant works as, e.g., [22, 13, 17], can only represent *marginalization* as an expanded sum since they lack a unary summation operator Σ. Thus, for instance, to express the marginal distribution of a random variable Y over a subset of (binary) variables $\{Z_1, \ldots, Z_m\}$ as $\sum_{z_1,\ldots,z_m} \mathbb{P}(y, z_1, \ldots, z_m)$, an encoding without summation requires an extension $\mathbb{P}(y, Z_1{=}0, \ldots, Z_m{=}0) + \ldots + \mathbb{P}(y, Z_1{=}1, \ldots, Z_m{=}1)$ of exponential size. Thus to analyze the complexity aspects of the standard notation of probability theory, one requires an encoding that directly represents marginalization. In a recent paper [31], the authors introduce the class succ-∃ℝ, and show that the satisfiability ($\textsc{Sat}^{poly\langle\Sigma\rangle}_{prob}$) for the (in)equalities of *polynomial* terms involving probabilities is succ-∃ℝ-complete.

Thus, succ-∃ℝ-completeness seems to be a meaningful yardstick for measuring computational complexity of decision problems. An interesting task would be to investigate problems involving the reals that have been shown to be in EXPSPACE, but not to be EXPSPACE-complete, which are natural candidates for succ-∃ℝ-complete problems.

Contributions and Structure of the Paper

Below we highlight our main contributions, partially summarized also in Fig. 1.

- We provide the characterization of succ-ETR in terms of nondeterministic real RAMs of exponential time respectively (Sec. 3). Moreover, for the classes over the reals in the sequence of inclusions (2), an upward translation result applies, which implies, e.g., NEXP $\subsetneq$ succ-∃ℝ unless NP = ∃ℝ which is widely disbelieved (Sec. 4).
- We strength slightly the completeness result (marked as (*d*) in Fig. 1) of [31] and prove the problem $\textsc{Sat}^{poly\langle\Sigma\rangle}_{prob}$ remains succ-∃ℝ-complete even if we disallow the basic terms to contain conditional probabilities (Sec. 5).
- We show that existential second order logic of real numbers is complete for succ-∃ℝ(Sec. 6).
- PSPACE has natural characterizations in terms of ETR; It coincides both with $\exists\mathbb{R}^{\Pi}$ – the class based on ETR enriched with standard product operator, and with succ-$\exists\mathbb{R}_{\text{poly}}$, defined in terms of the succinct variant of ETR with polynomially many variables (Sec. 7).
- $\exists\mathbb{R}^{\Sigma}$ – defined similar to $\exists\mathbb{R}^{\Pi}$, but with the addition of a unary summation operator instead – is contained in PSPACE = $\exists\mathbb{R}^{\Pi}$. We conjecture that this inclusion is strict, as the class is equivalent to $\mathsf{NP}_{\text{real}}^{\mathsf{VNP}_{\mathbb{R}}}$, machine to be an $\mathsf{NP}_{\text{real}}$ model with a $\mathsf{VNP}_{\mathbb{R}}$ oracle, where $\mathsf{VNP}_{\mathbb{R}}$ denotes Valiant's NP over the reals (Sec. 8.1).
- Unlike the languages devoid of the marginalization operator, the crucial small-model property is no longer satisfied. This property says that any satisfiable formula has a model of size bounded polynomially in the input length. Satisfiability with marginalization and with the additional requirement that there is a small model is complete for $\exists\mathbb{R}^{\Sigma}$ at the probabilistic layer (Sec. 8.2).

2 Preliminaries

Complexity Classes Based on the ETR

The problem succ-ETR and the corresponding class succ-∃ℝ are defined in [31] as follows. succ-ETR is the set of all Boolean circuits C that encode a true sentence φ as in Equation (1) as follows: Assume that C computes a function $\{0,1\}^N \to \{0,1\}^M$. Then φ is a tree consisting of 2^N nodes, each node being labeled with a symbol of $\{\vee, \wedge, \neg, +, \cdot, <, \leq, =\}$, a constant 0 or 1, or a variable $x_0, \ldots x_{2^N-1}$. For the node $i \in \{0,1\}^N$, the circuit computes an encoding $C(i)$ of the description of node i, consisting of the label of i, its parent, and its two children. The tree represents a true sentence, if the value at the root node would become true after applying the operator of each node to the value of its children, whereby the value of constants and variables is given in the obvious way. As in the case of $\exists\mathbb{R}$, to succ-$\exists\mathbb{R}$ belong all languages which are polynomial time many-one reducible to succ-ETR.

Besides succ-ETR, [31] introduce more complete problems for succ-$\exists\mathbb{R}$ as intermediate problems in the hardness proof. Of particular importance is the problem Σ_{vi}-ETR that we already discussed in the Introduction. Formally, the problem is defined as an extension of ETR by adding to the signature an additional summation operator $\sum_{x_j=0}^{1}$ with the following semantics[3]: If an arithmetic term is given by a tree with the top gate $\sum_{x_j=0}^{1}$ and $t(x_1,\dots,x_n)$ is the term computed at the child of the top gate, then the new term computes $\sum_{e=0}^{1} t(x_1,\dots,x_{j-1},e,x_{j+1},\dots,x_n)$, that is, we replace the variable x_j by a summation variable e, which then runs from 0 to 1. By nesting the summation operator, we are able to produce a sum with an exponential number of summands. The main reason why the new summation variables are introduced is due to the fact they can be used to *index* the quantified variables x_i used in Formula (1). Similarly as in succ-ETR, sentences of Σ_{vi}-ETR allow the use of exponentially many variables, however, the formulas are given directly and do not require any succinct encoding.

Probabilistic Languages

We always consider discrete distributions in the probabilistic languages studied in this paper. We represent the values of the random variables as $Val = \{0, 1, \dots, c-1\}$ and denote by $X_1, X_2, \dots, X_n$ the random variables used in the input formula. We assume, w.l.o.g., that they all share the same domain Val. A value of X_i is often denoted by x_i or a natural number. In this section, we describe syntax and semantics of the probabilistic languages.

By an *atomic* event, we mean an event of the form $X = x$, where X is a random variable and x is a value in the domain of X. The language $\mathcal{E}$ of propositional formulas over atomic events is the closure of such events under the Boolean operators $\wedge$ and $\neg$: $\mathbf{p} ::= X = x \mid \neg\mathbf{p} \mid \mathbf{p} \wedge \mathbf{p}$. The probability $\mathbb{P}(\delta)$ for formulas $\delta \in \mathcal{E}$ is called *primitive* or *basic term*, from which we build the probabilistic languages. The expressive power and computational complexity of the languages depend on the operations applied to the primitives.Allowing gradually more complex operators, we describe the languages which are the subject of our studies below. We start with the description of the languages $\mathcal{T}^*$ of terms, using the grammars given below.[4]

$$
\begin{array}{ll}
\mathcal{T}^{base} & \mathbf{t} ::= \mathbb{P}(\delta) \\
\mathcal{T}^{lin} & \mathbf{t} ::= \mathbb{P}(\delta) \mid \mathbf{t} + \mathbf{t} \\
\mathcal{T}^{poly} & \mathbf{t} ::= \mathbb{P}(\delta) \mid \mathbf{t} + \mathbf{t} \mid -\mathbf{t} \mid \mathbf{t} \cdot \mathbf{t} \\
\mathcal{T}^{poly\langle\Sigma\rangle} & \mathbf{t} ::= \mathbb{P}(\delta) \mid \mathbf{t} + \mathbf{t} \mid -\mathbf{t} \mid \mathbf{t} \cdot \mathbf{t} \mid \sum_x \mathbf{t}
\end{array}
$$

In the summation operator $\sum_x$, we have a dummy variable x which ranges over all values $0, 1, \dots, c-1$. The summation $\sum_x \mathbf{t}$ is a purely syntactical concept which represents the sum $\mathbf{t}[{}^0/_x] + \mathbf{t}[{}^1/_x] + \dots + \mathbf{t}[{}^{c-1}/_x]$, where by $\mathbf{t}[{}^v/_x]$, we mean the expression in which all occurrences of x are replaced with value v. For example, for $Val = \{0, 1\}$, the expression

[3] Recall, in [31], the authors assume arbitrary integer lower and upper bound in $\sum_{x_j=a}^{b}$. But it is easy to see that, w.l.o.g., one can restrict a and b to binary values.

[4] In the given grammars we omit the brackets for readability, but we assume that they can be used in a standard way.

$\sum_x \mathbb{P}(Y{=}1, X{=}x)$ semantically represents $\mathbb{P}(Y{=}1, X{=}0) + \mathbb{P}(Y{=}1, X{=}1)$. We note that the dummy variable x is not a (random) variable in the usual sense and that its scope is defined in the standard way.

The polynomial calculus $\mathcal{T}^{poly}$ was originally introduced by Fagin, Halpern, and Megiddo [11] to be able to express conditional probabilities by clearing denominators. While this works for $\mathcal{T}^{poly}$, this does not work in the case of $\mathcal{T}^{poly\langle\Sigma\rangle}$, since clearing denominators with exponential sums creates expressions that are too large. But we could introduce basic terms of the form $\mathbb{P}(\delta'|\delta)$ with $\delta, \delta' \in \mathcal{E}$ explicitly. All our hardness proofs work without conditional probabilities but all our matching upper bounds are still true with explicit conditional probabilities. For example, expression as $\mathbb{P}(X{=}1) + \mathbb{P}(Y{=}2) \cdot \mathbb{P}(Y{=}3)$ is a valid term in $\mathcal{T}^{poly}$.

Now, let $Lab = \{\text{base}, \text{lin}, \text{poly}, \text{poly}\langle\Sigma\rangle\}$ denote the labels of all variants of languages. Then for each $* \in Lab$ we define the languages $\mathcal{L}^*$ of Boolean combinations of inequalities in a standard way: $\mathbf{f} ::= \mathbf{t} \leq \mathbf{t}' \mid \neg\mathbf{f} \mid \mathbf{f} \wedge \mathbf{f}$, where $\mathbf{t}, \mathbf{t}'$ are terms in $\mathcal{T}^*$.

Although the language and its operations may appear rather restricted, all the usual elements of probabilistic formulas can be encoded. Namely, equality is encoded as greater-or-equal in both directions, e.g. $\mathbb{P}(x) = \mathbb{P}(y)$ means $\mathbb{P}(x) \geq \mathbb{P}(y) \wedge \mathbb{P}(y) \geq \mathbb{P}(x)$. The number 0 can be encoded as an inconsistent probability, i.e., $\mathbb{P}(X{=}1 \wedge X{=}2)$. In a language allowing addition and multiplication, any positive integer can be easily encoded from the fact $\mathbb{P}(\top) \equiv 1$, e.g. $4 \equiv (1+1)(1+1) \equiv (\mathbb{P}(\top)+\mathbb{P}(\top))(\mathbb{P}(\top)+\mathbb{P}(\top))$. If a language does not allow multiplication, one can show that the encoding is still possible. Note that these encodings barely change the size of the expressions, so allowing or disallowing these additional operators does not affect any complexity results involving these expressions.

We define the semantics of the languages as follows. Let $\mathfrak{M} = (\{X_1,\dots,X_n\}, P)$ be a tuple, where P is the joint probability distribution of variables $X_1,\dots,X_n$. For values $x_1,\dots,x_n \in Val$ and $\delta \in \mathcal{E}$, we write $x_1,\dots,x_n \models \delta$ if δ is satisfied by the assignment $X_1{=}x_1,\dots,X_n{=}x_n$. Denote by $S_\delta = \{x_1,\dots,x_n \mid x_1,\dots,x_n \models \delta\}$. We define $[\![\mathbf{e}]\!]_{\mathfrak{M}}$, for some expression $\mathbf{e}$, recursively in a natural way, starting with basic terms as follows $[\![\mathbb{P}(\delta)]\!]_{\mathfrak{M}} = \sum_{x_1,\dots,x_n \in S_\delta} P(X_1{=}x_1,\dots,X_n{=}x_n)$ and $[\![\mathbb{P}(\delta|\delta')]\!]_{\mathfrak{M}} = [\![\mathbb{P}(\delta \wedge \delta')]\!]_{\mathfrak{M}}/[\![\mathbb{P}(\delta')]\!]_{\mathfrak{M}}$, assuming that the expression is undefined if $[\![\mathbb{P}(\delta')]\!]_{\mathfrak{M}} = 0$. For two expressions $\mathbf{e}_1$ and $\mathbf{e}_2$, we define $\mathfrak{M} \models \mathbf{e}_1 \leq \mathbf{e}_2$, if and only if, $[\![\mathbf{e}_1]\!]_{\mathfrak{M}} \leq [\![\mathbf{e}_2]\!]_{\mathfrak{M}}$. The semantics for negation and conjunction are defined in the usual way, giving the semantics for $\mathfrak{M} \models \varphi$ for any $\varphi \in \mathcal{L}^*$.

Existential Second Order Logic of Real Numbers

We follow the definitions of [16]. Let A be a non-empty finite set and $\mathfrak{A} = (A, \mathbb{R}, f_1^{\mathfrak{A}},\dots, f_r^{\mathfrak{A}}, g_1^{\mathfrak{A}},\dots, g_t^{\mathfrak{A}})$, with $f_i^{\mathfrak{A}} : A^{ar(f_i)} \to \mathbb{R}$ and $g_i^{\mathfrak{A}} \subseteq A^{ar(g_i)}$, be a structure. Each $g_i^{\mathfrak{A}}$ is an $ar(g_i)$-ary relation on A and each $f_i^{\mathfrak{A}}$ is a weighted real function on $A^{ar(f_i)}$. The term $\mathbf{t}$ is generated by the following grammar: $\mathbf{t} ::= c \mid f(\vec{x}) \mid \mathbf{t} + \mathbf{t} \mid \mathbf{t} - \mathbf{t} \mid \mathbf{t} \times \mathbf{t} \mid \sum_x \mathbf{t}$, where $c \in \mathbb{R}$ is a constant (denoting itself), f is a function symbol, and $\vec{x}$ is a tuple of first-order variables. An assignment s is a total function that assigns a value in A for each first-order variable. The numerical value of $\mathbf{t}$ in a structure $\mathfrak{A}$ under an assignment s, denoted by $[\![\mathbf{t}]\!]^s_{\mathfrak{A}}$, is defined recursively in a natural way, starting with $[\![f_i(\vec{x})]\!]^s_{\mathfrak{A}} = f_i^{\mathfrak{A}}(s(\vec{x}))$ and applying the standard rules of real arithmetic.

For operators $O \subseteq \{+, \times, \Sigma, -\}$, (in-)equality operators $E \subseteq \{\leq, <, =\}$, and constants $C \subseteq \mathbb{R}$, the grammar of $\mathsf{ESO}_{\mathbb{R}}(O, E, C)$ sentences is given by $\phi ::= x = y \mid \neg(x = y) \mid i\ e\ j \mid \neg(i\ e\ j) \mid R(\vec{x}) \mid \neg R(\vec{x}) \mid \phi \wedge \phi \mid \phi \vee \phi \mid \exists x\ \phi \mid \forall x\ \phi \mid \exists f\ \phi$, where $x, y \in A$ are first order variables, i, j are real terms constructed using operations from O and constants from C, $e \in E$, and R denotes a relation symbol of a finite relational vocabulary[5] $g_1, \dots, g_t$.

[5] The grammar of [16] does not allow quantification over relations, e.g. $\exists R$, as these relations can be replaced by functions, e.g. chosen by $\exists f$.

The semantics of $\mathsf{ESO}_{\mathbb{R}}(O, E, C)$ is defined via $\mathbb{R}$-structures and assignments analogous to first-order logic with additional semantics for second order existential quantifier $\exists f$. That is, a structure $\mathfrak{A}$ satisfies a sentence ϕ under an assignment s, i.e., $\mathfrak{A} \models_s \phi$, according to the following cases of the grammar: $\mathfrak{A} \models_s x = y$, iff $s(x)$ equals $s(y)$; $\mathfrak{A} \models_s \neg(\phi)$ iff $\mathfrak{A} \not\models_s \phi$; $\mathfrak{A} \models_s i\ e\ j$ iff $[\![i]\!]^s_{\mathfrak{A}}\ e\ [\![j]\!]^s_{\mathfrak{A}}$ where $[\![i]\!]^s_{\mathfrak{A}}$ is the numerical value of i as defined above; $\mathfrak{A} \models_s R(\vec{x})$ iff $g_i^{\mathfrak{A}}(s(\vec{x}))$ is true for the $g_i^{\mathfrak{A}}$ corresponding to R in the model $\mathfrak{A}$; $\mathfrak{A} \models_s \phi \wedge \phi'$ iff $\mathfrak{A} \models_s \phi$ and $\mathfrak{A} \models_s \phi'$; $\mathfrak{A} \models_s \phi \vee \phi'$ iff $\mathfrak{A} \models_s \phi$ or $\mathfrak{A} \models_s \phi'$; $\mathfrak{A} \models_s \exists x \phi$ iff $\mathfrak{A} \models_{s[a/x]} \phi$ for some $a \in A$ where $s[a/x]$ means the assignment s modified to assign a to x; $\mathfrak{A} \models_s \forall x \phi$ iff $\mathfrak{A} \models_{s[a/x]} \phi$ for all $a \in A$; and $\mathfrak{A} \models_s \exists f \phi$ iff $\mathfrak{A}[h/f] \models_s \phi$ for some[6] function $h : A^{ar(f)} \to \mathbb{R}$ where $\mathfrak{A}[h/f]$ is the expansion of $\mathfrak{A}$ that interprets f as h.

For a set $S \subseteq \mathbb{R}$, we consider the restricted logic $\mathsf{ESO}_S(O, E, C)$ and $\mathsf{L\text{-}ESO}_S(O, E, C)$. There only the operators and constants of $O \cup E \cup C$ are allowed and all functions f are maps into S, i.e. $f : A^{ar(f)} \to S$. In the loose fragment $\mathsf{L\text{-}ESO}_S(O, E, C)$, negations $\neg(i\ e\ j)$ on real terms are also disallowed.

Probabilistic independence logic $\mathsf{FO}(\perp\!\!\!\perp_c)$ is defined as the extension of first-order logic with probabilistic independence atoms $\vec{x} \perp\!\!\!\perp_{\vec{z}} \vec{y}$ whose semantics is the standard semantics of conditional independence in probability distributions [9, 16].

Known Completeness and Complexity Results

The decision problems $\textsc{Sat}^*_{prob}$, with $* \in Lab$, take as input a formula φ in the languages $\mathcal{L}^*$ and ask whether there exists a model $\mathfrak{M}$ such that $\mathfrak{M} \models \varphi$. The computational complexity of probabilistic satisfiability problems has been a subject of intensive studies for languages which do not allow explicitly marginalization via summation operator Σ. Very recently [31] addressed the problem for polynomial languages.

Below, we summarize these results[7], informally presented in the Introduction:

- $\textsc{Sat}^{base}_{prob}$ and $\textsc{Sat}^{lin}_{prob}$ are NP-complete, [11],
- $\textsc{Sat}^{poly}_{prob}$ is $\exists\mathbb{R}$-complete [21], and
- $\textsc{Sat}^{poly\langle\Sigma\rangle}_{prob}$ is $\mathsf{succ}\text{-}\exists\mathbb{R}$-complete [31].

For a logic L, the satisfiability problem $\textsc{Sat}(L)$ is defined as follows: given a formula $\varphi \in L$, decide whether φ is satisfiable. For the model checking problem of a logic L, we consider the following variant: given a sentence $\varphi \in L$ and a structure $\mathfrak{A}$, decide whether $\mathfrak{A} \models \varphi$. For model checking of $\mathsf{FO}(\perp\!\!\!\perp_c)$, the best-known complexity lower and upper bounds are NEXP-hardness and $\mathsf{EXPSPACE}$, respectively [15].

3 NEXP over the Reals

In [10], Erickson, van Der Hoog, and Miltzow extend the definition of word RAMs to real computations. In contrast to the so-called BSS model of real computation [5], the real RAMs of Erickson et al. provide integer and real computations at the same time, allowing for instance indirect memory access to the real registers and other features that are important to implement algorithms over the reals. The input to a real RAM is a pair of vectors, the first one is a vector of real numbers, the second is a vector of integers. Real RAMs have two types of registers, word registers and real registers. The word registers can store integers

[6] Note that h might be an arbitrary function and is not restricted to the functions $f_i^{\mathfrak{A}}$ of the model.
[7] In the papers [21] and [31] the authors show even stronger results, namely that the completeness results also hold for causal satisfiability problems.

with w bits, where w is the word size. The total number of registers is 2^w for each of the two types. Real RAMs perform arithmetic operations on the word registers, where words are interpreted as integers between 0 and $2^w - 1$, and bitwise Boolean operations. On the real registers, only arithmetic operations are allowed. Word registers can be used for indirect addressing on both types of registers and the control flow is established by conditional jumps that depend on the result of a comparison of two word registers or of a real register with the constant 0. For further details we refer to the original paper [10].

The real RAMs of [10] characterize the existential theory of the reals. The authors prove that a problem is in $\exists\mathbb{R}$ iff there is a polynomial time real verification algorithm for it. In this way, real RAMs are an "easy to program" mechanism to prove that a problem is contained in $\exists\mathbb{R}$. Beside the input I, which is a sequence of words, the real verification algorithm also gets a certificate consisting of a sequence of real numbers x and a further sequence of words z. I is in the language if there is a pair (x, z) that makes the real verification algorithm accept. I is not in the language if for all pairs (x, z), the real verification rejects.

Instead of using certificates and verifiers, we can also define nondeterministic real RAMs that can guess words and real numbers on the fly. Like for classical Turing machines, it is easy to see that these two definitions are equivalent (when dealing with time bounded computations).

▶ **Definition 1.** *Let $t : \mathbb{N} \to \mathbb{N}$ be a function. We define $\mathsf{NTime}_{real}(t)$ to be the set of all languages $L \subseteq \{0,1\}^\star$, such that there is a constant $c \in \mathbb{N}$ and a nondeterministic real word-RAM M that recognizes L in time t for all word-sizes $w \geq c \cdot \log(t(n)) + c$.*

For any set of functions T, we define $\mathsf{NTime}_{real}(T) = \bigcup_{t \in T} \mathsf{NTime}_{real}(t)$. We define our two main classes of interest, NP_{real} and NEXP_{real} as follows:

$$\mathsf{NP}_{real} = \mathsf{NTime}_{real}(\mathrm{poly}(n)), \qquad \mathsf{NEXP}_{real} = \mathsf{NTime}_{real}(2^{\mathrm{poly}(n)}).$$

Note that the word size needs to be at least logarithmic in the running time, to be able to address a new register in each step.

One of the main results of Erickson et al. (Theorem 2 in their paper) can be rephrased as $\exists\mathbb{R} = \mathsf{NP}_{\mathrm{real}}$. Their techniques can be extended to prove that $\mathsf{succ}\text{-}\exists\mathbb{R} = \mathsf{NEXP}_{\mathrm{real}}$.

We get the following in analogy to the well-known results that the succinct version of 3-Sat is NEXP-complete.

▶ **Lemma 2.** succ-*ETR is NEXP_{real}-complete and thus $\mathsf{NEXP}_{real} = \mathsf{succ}\text{-}\exists\mathbb{R}$.*

Proof idea. For the one direction, one carefully has to analyze the construction by Erickson et al. and show that the simulation there can also be implemented succinctly. The reverse direction simply follows from expanding the succinct ETR instance and use the fact that nondeterministic real word-RAMs can solve ETR in polynomial time. Along the way, we also obtain a useful normalization procedure for succinct ETR instances. While for normal ETR instances, it is obvious that one can always push negations down, it is not clear for succinct instances. We describe the details in the full version. ◀

4 The Relationships between the Boolean Classes and Classes over the Reals

Now we study the new class $\mathsf{NEXP}_{\mathrm{real}} = \mathsf{succ}\text{-}\exists\mathbb{R}$ from a complexity theoretic point of view.

$$\mathsf{NP} \subseteq \exists\mathbb{R} \subseteq \mathsf{PSPACE}; \qquad \mathsf{NEXP} \subseteq \mathsf{succ}\text{-}\exists\mathbb{R} \subseteq \mathsf{EXPSPACE}. \tag{3}$$

The left side of (3) is well-known. The first inclusion on the right side is obvious, since a real RAM simply can ignore the real part. The second inclusion follows from expanding the succinct instance into an explicit formula (which now has exponential size) and simply using the known PSPACE-algorithm.

We prove two translation results, that is, equality of one of the inclusion in the left equation of (3) implies the equality of the corresponding inclusion in the right equation of (3).

▶ **Theorem 3.** *If* $\exists\mathbb{R} = \mathsf{NP}$*, then* $\mathsf{succ}\text{-}\exists\mathbb{R} = \mathsf{NEXP}$*.*

▶ **Theorem 4.** *If* $\exists\mathbb{R} = \mathsf{PSPACE}$*, then* $\mathsf{succ}\text{-}\exists\mathbb{R} = \mathsf{EXPSPACE}$*.*

Further we prove a nondeterministic time hierarchy theorem (see the full version for the details) for real word RAMs. Using the characterization of $\exists\mathbb{R}$ and $\mathsf{succ}\text{-}\exists\mathbb{R}$ in terms of real word RAMs, in particular, we get that succ-ETR is strictly more expressive than ETR.

▶ **Corollary 5.** $\exists\mathbb{R} = \mathsf{NP}_{real} \subsetneq \mathsf{NEXP}_{real} = \mathsf{succ}\text{-}\exists\mathbb{R}$*.*

5 Hardness of Probabilistic Satisfiability without Conditioning

To prove that $\text{SAT}^{poly\langle\Sigma\rangle}_{prob}$ is $\mathsf{succ}\text{-}\exists\mathbb{R}$-complete, van der Zander, Bläser and Liśkiewicz [31] show the hardness part for the variant of the probabilistic language where the primitives are also allowed to be conditional probabilities. A novel contribution of our work is to extend this completeness result to our version for languages which disallow conditional probabilities:

▶ **Theorem 6.** *The problem* $\text{SAT}^{poly\langle\Sigma\rangle}_{prob}$ *remains* $\mathsf{succ}\text{-}\exists\mathbb{R}$*-complete even without conditional probabilities.*

In the rest of this section, we will give the proof of the theorem.

In [31] the authors have already shown that Σ_{vi}-ETR is $\mathsf{succ}\text{-}\exists\mathbb{R}$-complete. We define Σ_{vi}-ETR$_1$ in the same way as Σ_{vi}-ETR, but asking the question whether there is a solution where the sum of the absolute values (ℓ_1 norm) is bounded by 1. Then we can reduce Σ_{vi}-ETR$_1$ to $\text{SAT}^{poly\langle\Sigma\rangle}_{prob}$ without the need for conditional probabilities (Lemma 9). The proof that Σ_{vi}-ETR$_1$ is hard for $\mathsf{succ}\text{-}\exists\mathbb{R}$ (Lemma 8) depends on a result of Grigoriev and Vorobjov [14] who showed that the solution to an ETR instance can be bounded by a constant that only depends on the bitsize of the instance. Thus the solution can be scaled to fit into a probability distribution. This completes the proof of Theorem 6.

▶ **Theorem 7** (Grigoriev and Vorobjov [14])**.** *Let* $f_1, \dots, f_k \in \mathbb{R}[X_1, \dots, X_n]$ *be polynomials of total degree* $\leq d$ *with coefficients of bit size* $\leq L$*. Then every connected component of* $\{x \in \mathbb{R}^n \mid f_1(x) \geq 0 \wedge \cdots \wedge f_k(x) \geq 0\}$ *contains a point of distance less than* $2^{Ld^{cn}}$ *from the origin for some absolute constant c. The same is true if some of the inequalities are replaced by strict inequalities.*

▶ **Lemma 8.** Σ_{vi}*-ETR* $\leq_P$ Σ_{vi}*-ETR*$_1$*.*

Proof. Let ϕ be an instance of Σ_{vi}-ETR. We will transform it into a formula φ such that φ has a solution with ℓ_1 norm bounded by 1 iff ϕ has any solution.

Let S be the bit length of ϕ. The number n of variables in ϕ is bounded by 2^S. The degree of all polynomials is bounded by S. Note that the exponential sums do not increase the degree at all. Finally, all coefficients have bit size $O(S)$. Note that one summation operator doubles the coefficients at most. By Theorem 7, if ϕ is satisfiable, then there is a solution with entries bounded by $T := 2^{2^{2^{cS}}}$ for some constant c.

In our new instance φ first creates a small constant $d \leq 1/((2^m+n)T)$ for some m polynomial in S defined below. This can be done using Tseitin's trick: We take 2^m many fresh variables t_i and start with $(2^m+2^S)t_1 = 1$ and then iterate by adding the equation $\sum_{i=1}^{2^m-1}(t_i^2 - t_{i+1})^2 = 0$, i.e. forcing $t_{i+1} = t_i^2$. To implement the first equation we replace 2^m by $\sum_{e_1=0}^{1} \cdots \sum_{e_m=0}^{1} 1$ and similarly replace 2^S. To implement the second equation we replace $\sum_{i=1}^{2^m-1}(t_i^2 - t_{i+1})^2$ by $\sum_{e_1=0}^{1} \cdots \sum_{e_m=0}^{1} \sum_{f_1=0}^{1} \cdots \sum_{f_m=0}^{1} (t_{e_1,\ldots,e_m}^2 - t_{f_1,\ldots,f_m})^2 \cdot A(e_1,\ldots,e_m,f_1,\ldots,f_m)$ where A is an arithmetic formula returning 1 iff the binary number represented by $f_1,\ldots,f_m$ is the successor of the binary number represented by $e_1,\ldots,e_m$. The number m is polynomial in S. The unique satisfying assignment to the t_i has its entries bounded by $1/(2^m+2^S)$. Let $d := t_{2^m}$ be the last variable.

Now in ϕ we replace every occurrence of x_i by x_i/d and then multiple each (in-)equality by an appropriate power of d to remove the divisions in order to obtain φ. In this way, from every solution to ϕ, we obtain a solution to φ by multiplying the entries by d and vice versa. Whenever ϕ has a solution, then it has one with entries bounded by T. By construction φ then has a solution with entries bounded by $1/(2^m+2^S)$. Since each entry of the solution is bounded by $1/(2^m+2^S)$, the ℓ_1 norm is bounded by $1/2$. ◀

▶ **Lemma 9.** *Σ_{vi}-$ETR_1 \leq_P \mathrm{SAT}_{prob}^{poly\langle\Sigma\rangle}$ via a reduction without the need for conditional probabilities.*

Proof. Let X_0 be a random variable with range $\{-1,0,1\}$ and let $X_1,\ldots,X_N$ be binary random variables. We replace each real variable $x_{e_1,\ldots,e_N}$ in the Σ_{vi}-ETR$_1$ formula as follows:

$$x_{e_1,\ldots,e_N} := \mathbb{P}(X_0{=}1 \wedge X_1{=}e_1 \wedge \ldots \wedge X_N{=}e_N) - \mathbb{P}(X_0{=}-1 \wedge X_1{=}e_1 \wedge \ldots \wedge X_N{=}e_N)$$

This guarantees that $x_{e_1,\ldots,e_N} \in [-1,1]$. The existential quantifiers now directly correspond to the existence of a probability distribution $P(X_0,\ldots,X_N)$, where each variable corresponds to an different set of two entries of P.

Let $P(X_0,\ldots,X_N)$ be a solution to the constructed $\mathrm{SAT}_{prob}^{poly\langle\Sigma\rangle}$ instance. Then clearly setting $x_{e_1,\ldots,e_N} = P(1,e_1,\ldots,e_N) - P(-1,e_1,\ldots,e_N)$ satisfies the original Σ_{vi}-ETR$_1$ instance. Furthermore it has an ℓ_1 norm bounded by 1:

$$\begin{aligned}\sum_{e_1=0}^{1} \cdots \sum_{e_N=0}^{1} |x_{e_1,\ldots,e_N}| &= \sum_{e_1=0}^{1} \cdots \sum_{e_N=0}^{1} |P(1,e_1,\ldots,e_N) - P(-1,e_1,\ldots,e_N)| \\ &\leq \sum_{e_1=0}^{1} \cdots \sum_{e_N=0}^{1} (P(1,e_1,\ldots,e_N) + P(-1,e_1,\ldots,e_N)) \\ &\leq 1\,.\end{aligned}$$

Vice-versa, let the original Σ_{vi}-ETR$_1$ be satisfied by some choice of the $x_{e_1,\ldots,e_N}$ with ℓ_1 norm α bounded by 1. We define the probability distribution

$$P(X_0,X_1,\ldots,X_N) = \begin{cases} \frac{1-\alpha}{2^N} & \text{if } X_0 = 0 \\ \max(x_{X_1,\ldots,X_N},0) & \text{if } X_0 = 1 \\ \max(-x_{X_1,\ldots,X_N},0) & \text{if } X_0 = -1 \end{cases}$$

Every entry of P is non-negative since $\alpha \leq 1$. Furthermore the sum of all entries is exactly 1, the entries with $X_0 \in \{-1,1\}$ contribute exactly α total and the 2^N entries with $X_0 = 0$ contribute $1-\alpha$ total. Since P fulfills the equation $x_{e_1,\ldots,e_N} = P(1,e_1,\ldots,e_N) - P(-1,e_1,\ldots,e_N)$, it is a solution to the constructed $\mathrm{SAT}_{prob}^{poly\langle\Sigma\rangle}$ instance. ◀

6 Correspondence to Existential Second Order Logic and $\mathsf{FO}(\perp\!\!\!\perp_c)$

In this section, we investigate the complexity of existential second order logics and the probabilistic independence logic $\mathsf{FO}(\perp\!\!\!\perp_c)$.

▶ **Lemma 10.** *Model checking of* $\mathsf{ESO}_{\mathbb{R}}(\Sigma, +, \times, \leq, <, =, \mathbb{Q})$ *is in* $\mathsf{succ}\text{-}\exists\mathbb{R}$.

Proof. In model checking, the input is a finite structure $\mathfrak{A}$ and a sentence ϕ, and we need to decide whether $\mathfrak{A} \models \phi$. $\mathfrak{A}$ includes a domain A for the existential/universal quantifiers over variables. Any function (relation) of arity k can be represented as a (Boolean) table of size $|A|^k$. Some of these tables might be given in the input. The remaining tables of functions chosen by quantifiers $\exists f$ can simply be guessed by a $\mathsf{NEXP}_{\text{real}}$ machine in non-deterministic exponential time. Then all possible values for the quantifiers of the finite domain can be enumerated and all sentences can be evaluated. This completes the proof, due to the characterization given in Lemma 2. ◀

▶ **Proposition 11.** *Model checking of* $\mathsf{L}\text{-}\mathsf{ESO}_{[0,1]}(+, \times, \leq, 0, 1)$ *is* $\mathsf{succ}\text{-}\exists\mathbb{R}$*-hard.*

Proof. We start with the following equivalences relating the logics:

$$\mathsf{L}\text{-}\mathsf{ESO}_{[0,1]}(+, \times, \leq, 0, 1) \equiv \mathsf{L}\text{-}\mathsf{ESO}_{[-1,1]}(+, \times, \leq, 0, 1) \equiv \mathsf{L}\text{-}\mathsf{ESO}_{[-1,1]}(+, \times, -, =, \leq, 0, 1/8, 1).$$

The first equivalence has been shown by Hannula et al. [16]. To see the second one, note that we can replace operator $=$ using $a = b$ as $a \leq b \wedge b \leq a$. The negative one -1 can be defined by a function -1 of arity 0 using $\exists(-1) : (-1) + 1 = 0$. Then any subtraction $a - b$ can be replaced with $a + (-1) \times b$. Finally, the fraction $1/8$ is a function given by $\exists 1/8 : 1/8 + 1/8 + 1/8 + 1/8 + 1/8 + 1/8 + 1/8 + 1/8 = 1$. These equivalence reductions can be performed in polynomial time.

In the rest of the proof, we show the hardness, reducing the problems in $\mathsf{succ}\text{-}\exists\mathbb{R}$ to the existential second order logic $\mathsf{L}\text{-}\mathsf{ESO}_{[-1,1]}(+, \times, -, =, \leq, 0, 1/8, 1)$. To this aim, we use a $\mathsf{succ}\text{-}\exists\mathbb{R}$-complete problem which is based on a problem given by Abrahamsen et al. [1], who have shown that an equation system consisting of only sentences of the form $x_i = 1/8$, $x_i + x_j = x_k$, and $x_i \cdot x_j = x_k$ is $\exists\mathbb{R}$-complete. As shown in [31], this can be turned into a $\mathsf{succ}\text{-}\exists\mathbb{R}$-complete problem, denoted as $\text{succETR}^{1/8,+,\times}_{[-1/8,1/8]}$, by replacing the explicit indices i, j, k with circuits that compute the indices for an exponential number of these three equations. The circuits can be encoded with arithmetic operators, which allows us to encode all equations in existential second order logic in a polynomial time reduction.

The instances of $\text{succETR}^{1/8,+,\times}_{[-1/8,1/8]}$ are represented as seven Boolean circuits $C_0, C_1, \ldots, C_6 : \{0,1\}^M \to \{0,1\}^N$ such that $C_0(j)$ gives the index of the variable in the jth equation of type $x_i = 1/8$, $C_1(j), C_2(j), C_3(j)$ give the indices of variables in the jth equation of the type $x_{i_1} + x_{i_2} = x_{i_3}$, and $C_4(j), C_5(j), C_6(j)$ give the indices of variables in the jth equation of the type $x_{i_1} x_{i_2} = x_{i_3}$. Without loss of generality, we can assume that each type has the same number 2^M of equations. An instance of the problem $\text{succETR}^{1/8,+,\times}_{[-1/8,1/8]}$ is satisfiable if and only if:

$$\begin{aligned} &\exists x_0, \ldots, x_{2^N-1} \in [-1/8, 1/8] : \forall j \in [0, 2^M - 1] : \\ &\qquad x_{C_0(j)} = 1/8, \quad x_{C_1(j)} + x_{C_2(j)} = x_{C_3(j)}, \quad \text{and} \quad x_{C_4(j)} \cdot x_{C_5(j)} = x_{C_6(j)}. \end{aligned} \tag{4}$$

Below, we prove that

$$\text{succETR}^{1/8,+,\times}_{[-1/8,1/8]} \leq_P \mathsf{L}\text{-}\mathsf{ESO}_{[-1,1]}(+, \times, -, =, \leq, 0, 1/8, 1). \tag{5}$$

Let the instance of $\mathrm{succETR}^{1/8,+,\times}_{[-1/8,1/8]}$ be represented by seven Boolean circuits $C_0, C_1, \ldots, C_6 : \{0,1\}^M \to \{0,1\}^N$ as described above. Let the variables of the instance be indexed as $x_{e_1,\ldots,e_N}$, with $e_i \in \{0,1\}$ for $i \in [N]$. We will identify the bit sequence $\vec{b} = b_1, \ldots, b_M$ by an integer j, with $0 \le j \le 2^M - 1$, the binary representation of which is $b_1 \ldots b_M$ and vice versa.

We construct sentences in the logic $\mathsf{L\text{-}ESO}_{[-1,1]}(+, \times, -, =, \le, 0, 1/8, 1)$ and prove that a binary model satisfies the sentences if and only if the formula (4) is satisfiable.

Let q be an N-ary function where $q(e_1, \ldots, e_N)$ should encode the value of variable $x_{e_1,\ldots,e_N}$. For the range, we require $\forall \vec{x} : 0 - 1/8 \le q(\vec{x}) \wedge q(\vec{x}) \le 1/8$.

For each circuit C_i, we define a function y_i whose value $y_i(\vec{b})$ is $x_{C_i(j)}$, i.e., $q(C_i(j))$. Then y_i can directly be inserted in the equation system (4). For this, we need to encode the circuit as logical sentences and relate y and q.

To model a Boolean formula encoded by a node of C_i, with $i = 0, 1, \ldots, 6$, we use one step of arithmetization to go from logical formulas to calculations on real numbers, where $0 \in \mathbb{R}$ means false and $1 \in \mathbb{R}$ means true. While negation is not allowed directly in $\mathsf{L\text{-}ESO}$, on the real numbers we can simulate negation by subtraction.

For each node v of each circuit C_i, we need a function $c_{i,v}$ of arity M, such that $c_{i,v}(\vec{b})$ is the value computed by the node if the circuit is evaluated on input $j = b_1 \ldots b_M$.

If v is an input node, the node only reads one bit $u_{i,k}$ from the input, so let $\forall \vec{b} : c_{i,v}(\vec{b}) = id(b_k)$, where id is a function that maps $0, 1$ from the finite domain to $0, 1 \in \mathbb{R}$.

For each internal node v of C_i, we proceed as follows.

If v is labeled with $\neg$ and u is a child of v, then we require $\forall \vec{b} : c_{i,v}(\vec{b}) = 1 - c_{i,u}(\vec{b})$.

If v is labeled with $\wedge$ and u and w are children of v, then we require $\forall \vec{b} : c_{i,v}(\vec{b}) = c_{i,u}(\vec{b}) \times c_{i,w}(\vec{b})$.

Finally, if v is labeled with $\vee$ and u and w are children of v, then we require $\forall \vec{b} : c_{i,v}(\vec{b}) = 1 - (1 - c_{i,u}(\vec{b})) \times (1 - c_{i,w}(\vec{b}))$.

Thus, if v is an output node of a circuit C_i, then, for C_i fed with input $j = b_1 \ldots b_M \in \{0,1\}^M$, we have that v evaluates to true if and only if $c_{i,v}(\vec{b}) = 1$.

Next, we need an $(N + M)$-arity selector function $s_i(\vec{b}, \vec{e})$ which returns 1 iff the output of circuit C_i on input $\vec{b}$ is $\vec{e}$. It can be defined as:

$$\forall \vec{b}, \vec{e} : \quad s_i(\vec{b}, \vec{e}) = \prod_{k=1}^{N} (c_{i,v_k}(\vec{b}) \times id(e_k) + (1 - c_{i,v_k}(\vec{b})) \times (1 - id(e_k))).$$

Each factor of the product is 1 iff $c_{i,v_k}(\vec{b}) = e_k$. It has constant length, so it can be expanded using the multiplication of the logic.

We express each $q(C_i(j))$ as a function $y_i(j)$, where $\vec{b}$ is the binary representation of j:

$$\forall \vec{b}, \vec{e} : \quad y_i(\vec{b}) \times s_i(\vec{b}, \vec{e}) = q(\vec{e}) \times s_i(\vec{b}, \vec{e}).$$

The above equation is trivially satisfied for $s_i(\vec{b}, \vec{e}) = 0$, thus it enforces equality of $y_i(\vec{b})$ and $q(\vec{e})$ only in the case $s_i(\vec{b}, \vec{e}) = 1$. Inserting y_i in the equation system (4) gives us the last $\mathsf{L\text{-}ESO}$ formula:

$$\forall \vec{b} : \quad y_0(\vec{b}) = 1/8, \quad y_1(\vec{b}) + y_2(\vec{b}) = y_3(\vec{b}), \quad \text{and} \quad y_4(\vec{b}) \times y_5(\vec{b}) = y_6(\vec{b}),$$

which, combining with the previous formulas and preceded by second order existential quantifiers $\exists y_i, \exists s_i, \exists c_{i,v}, \exists id$, with $i = 0, \ldots, 6$, is satisfiable if and only if the formula (4) are satisfiable.

Obviously, the size of the resulting sentences are polynomial in the size $|C_0|+|C_1|+\ldots+|C_6|$ of the input instance and the sentences can be computed in polynomial time.

This completes the construction of reduction (5) and the proof of the proposition. ◀

As $\mathsf{L\text{-}ESO}_{[0,1]}(+,\times,\leq,0,1)$ is weaker than $\mathsf{ESO}_{\mathbb{R}}(\Sigma,+,\times,\leq,<,=,\mathbb{Q})$, it follows:

▶ **Theorem 12.** *Let $S=\mathbb{R}$ or $S=[a,b]$ with $[0,1]\subseteq S$, $\{0,1\}\subseteq C\subseteq\mathbb{Q}$, $\{\times\}\subseteq O\subseteq\{+,\times,\Sigma\}$ with $|O|\geq 2$, and $E\subseteq\{\leq,<,=\}$ with $\{\leq,=\}\cap E\neq\emptyset$. Model checking of*

- $\mathsf{L\text{-}ESO}_S(O,E,C)$ *and*
- $\mathsf{ESO}_S(O,E,C)$

is succ-$\exists\mathbb{R}$*-complete.*

Proof. We start with the following equivalence, which follows from the fact that a comparison $a\leq b$ can be replaced by $\exists\epsilon,x: a\epsilon+x=b\epsilon$:

$$\mathsf{L\text{-}ESO}_{[0,1]}(+,\times,\leq,0,1)\equiv\mathsf{L\text{-}ESO}_{[0,1]}(+,\times,=,0,1). \tag{6}$$

The next fact has been used by [16], but without proof. Perhaps the authors thought it to be too trivial to mention. But it is not obvious, since the standard technique of replacing $a\leq b$ with $\exists x: a+x^2=b$ does not work here when x is restricted to $[0,1]$.

In some sense, $\mathsf{L\text{-}ESO}_{[0,1]}(+,\times,\leq,0,1)$ is the weakest logic one can consider in this context:

▶ **Fact 13.** *Let $S=\mathbb{R}$ or $S=[a,b]$ with $[0,1]\subseteq S$, $\{0,1\}\subseteq C\subseteq\mathbb{Q}$, $\{\times\}\subseteq O\subseteq\{+,\times,\Sigma\}$ with $|O|\geq 2$, and $E\in\{\leq,=\}$.*

$\mathsf{L\text{-}ESO}_{[0,1]}(+,\times,\leq,0,1)\leq\mathsf{L\text{-}ESO}_S(O,E,C)\leq\mathsf{ESO}_S(O,E,C)$.

Proof of Fact 13. Relation $=$ subsumes $\leq$ due to (6).

If $+\in O$, the remaining statements are trivial. Otherwise, we need to express $+$ using Σ.

If $S=\mathbb{R}$, $x+y$ can be written as $\Sigma_t c(t)$ where $c(0)=x, c(1)=y$. (we consider model checking problems, where the finite domain can be set to binary)

If $S=[a,b]$, x or y might be outside the range. But the total weight of any k-arity function is $(b-a)^k$ and each term has a maximal polynomial degree D, so x and y are bounded by $O((b-a)^{kD})$. So all expressions can be scaled to fit in the range (Lemma 6.4. Step 3 proves this for functions that are probability distributions in [16]). ◀

All of this combined shows the theorem. ◀

Hannula et. al [16] and Durand et. al [9] have shown the following relationships between expressivity of the logics: $\mathsf{L\text{-}ESO}_{[0,1]}(+,\times,=,0,1)\leq\mathsf{L\text{-}ESO}_{d[0,1]}(\Sigma,\times,=)\equiv\mathsf{FO}(\perp\!\!\!\perp_c)$. $\mathsf{L\text{-}ESO}_{d[0,1]}[O,E,C]$ means a variant of $\mathsf{L\text{-}ESO}_{[0,1]}(O,E,C)$ where all functions are required to be distributions, that is $f^{\mathfrak{A}}: A^{ar(f)}\to[0,1]$ and $\sum_{\vec{a}\in A^{ar(f)}}f^{\mathfrak{A}}(\vec{a})=1$. From the proof for the translation from L-ESO to $\mathsf{FO}(\perp\!\!\!\perp_c)$ in [9], it follows that the reduction can be done in polynomial time. Moreover, it is easy to see that model checking of $\mathsf{FO}(\perp\!\!\!\perp_c)$ can be done in $\mathsf{NEXP}_{\text{real}}$. Thus we get

▶ **Corollary 14.** *Model checking of $\mathsf{FO}(\perp\!\!\!\perp_c)$ is* succ-$\exists\mathbb{R}$*-complete.*

This corollary answers the question asked in [15] for the exact complexity of $\mathsf{FO}(\perp\!\!\!\perp_c)$ and confirms their result that the complexity lies between NEXP and EXPSPACE.

7 Succinct ETR of Polynomially Many Variables

The key feature that makes the language Σ_{vi}-ETR defined in [31] very powerful is the ability to index the quantified variables in the scope of summation. Nesting the summations allows handling an exponential number of variables. Thus, similarly as in succ-ETR, sentences of Σ_{vi}-ETR allow the use of exponentially many variables, however, the formulas are given directly and do not require any succinct encoding. Due to the fact that variable indexing is possible, [31] show that Σ_{vi}-ETR is polynomial time equivalent to succ-ETR.

Valiant's class VNP [6, 19] is also defined in terms of exponential sums (we recall the definition of VNP and related concepts in the full version). However, we cannot index variables as above, therefore, the overall number of variables is always bounded by the length of the defining expression. It is natural to extend ETR with a summation operator, but without variable indexing as was allowed in Σ_{vi}-ETR. In this way, we can have exponential sums, but the number of variables is bounded by the length of the formula. Instead of a summation operator, we can also add a product operator, or both.

▶ **Definition 15.**
1. Σ-*ETR is defined as ETR with the addition of a unary summation operator* $\sum_{x_i=0}^{1}$.
2. Π-*ETR is defined similar to* Σ-*ETR, but with the addition of a unary product operator* $\prod_{x_i=0}^{1}$ *instead.*
3. $\Sigma\Pi$-*ETR is defined similar to* Σ-*ETR or* Π-*ETR, but including both unary summation and product operators.*

In the three problems above, the number of variables is naturally bounded by the length of the instance, since the problems are not succinct. For example, the formula $\sum_{x_1=0}^{1}\sum_{x_2=0}^{1}(x_1+x_2)(x_1+(1-x_2))(1-x_1)=0$ explained in the introduction is also in Σ-ETR and $\Sigma\Pi$-ETR, but not in Π-ETR. The formula $\sum_{e_1=0}^{1}\cdots\sum_{e_N=0}^{1}(x_{\langle e_1,\dots,e_N\rangle})^2=1$ is in neither of these three classes since it uses variable indexing.

To demonstrate the expressiveness of Π-ETR, we will show that the PSPACE-complete problem QBF can be reduced to it.

▶ **Lemma 16.** QBF $\leq_P$ Π-*ETR.*

Proof. Let $Q_1x_1Q_2x_2\dots Q_nx_n\varphi(x_1,\dots,x_n)$ be a quantified Boolean formula with $Q_1,\dots,Q_n\in\{\exists,\forall\}$. We arithmetize φ as $A(\varphi)$ inductively using the following rules:

φ is a variable x_i: We construct $A(\varphi)=x_i$.

φ is $\neg\varphi_1$: We construct $A(\varphi)$ as $1-A(\varphi_1)$.

φ is $\varphi_1\wedge\varphi_2$: We construct $A(\varphi)$ as $A(\varphi_1)\cdot A(\varphi_2)$.

φ is $\varphi_1\vee\varphi_2$: We construct $A(\varphi)$ as $1-(1-A(\varphi_1))\cdot(1-A(\varphi_2))$ via De Morgan's law and the previous two cases.

The special treatment of the $\vee$ operator ensures that whenever $x_1,\dots,x_n\in\{0,1\}$, then $A(\varphi)$ evaluates to 1 iff $x_1,\dots,x_n$ satisfy φ and 0 otherwise. We then arithmetize the quantifiers $Q_1,\dots,Q_n$ in a similar way, but using the unary product operator.

$Q_i=\forall$: We construct $A(\forall x_iQ_{i+1}x_{i+1}\dots Q_nx_n\varphi)$ as $\prod_{x_i=0}^{1}A(Q_{i+1}x_{i+1}\dots Q_nx_n\varphi)$

$Q_i=\exists$: We construct $A(\exists x_iQ_{i+1}x_{i+1}\dots Q_nx_n\varphi(x_1,\dots,x_n)$ as $1-\prod_{x_i=0}^{1}(1-A(Q_{i+1}x_{i+1}\dots Q_nx_n\varphi))$, again using De Morgan's law.

The final Π-ETR formula is then just $A(Q_1x_1Q_2x_2\dots Q_nx_n\varphi)=1$.

The correctness of the construction follows because a formula of the form $\forall\psi(x)$ is true over the Boolean domain iff $\psi(0)\wedge\psi(1)$ is true. The unary product together with arithmetization allows us to write the whole formula down without an exponential blow-up. ◀

We also consider the succinct version of ETR with only a polynomial number of variables.

▶ **Definition 17.** succ-ETR_{poly} *is defined similar to* succ-*ETR, but variables are encoded in unary instead of binary, thus limiting the amount of variables to a polynomial amount of variables. (Note that the given input circuit succinctly encodes an ETR formula and not an arbitrary circuit.)*

For succ-ETR, it does not matter whether the underlying structure of the given instance is a formula or an arbitrary circuit, since we can transform the circuit into a formula using Tseitin's trick. This, however, requires a number of new variables that is proportional to the size of the circuit, which is exponential.

Like for ETR, we can now define corresponding classes by taking the closure of the problems defined above. It turns out that we get meaningful classes in this way, however, for some unexpected reason. Except for Σ-ETR, all classes coincide with PSPACE, which we will see below. By restricting the number of variables to be polynomial, the complexity of succ-ETR reduces considerably, from being $\mathsf{NEXP}_{\text{real}}$-complete, which contains NEXP, to PSPACE. On the other hand, the problems are most likely more powerful than ETR, assuming that $\exists\mathbb{R}$ is a proper subset of PSPACE, which is believed by at least some researchers.

▶ **Definition 18.** *Let* succ-$\exists\mathbb{R}_{\text{poly}}$ *be the closure of* succ-ETR_{poly} *under polynomial time many one reductions.*

▶ **Theorem 19.** PSPACE $=$ succ-$\exists\mathbb{R}_{\text{poly}}$ *and the problems* Π-*ETR,* ΣΠ-*ETR, and* succ-ETR_{poly} *are* PSPACE*-complete.*

Proof idea. To show that succ-ETR_{poly} is in PSPACE, we rely on results by [24]. One of the famous consequences of Renegar's work is that ETR $\in$ PSPACE. But Renegar shows even more, because he can handle an exponential number of arithmetic terms of exponential size with exponential degree as long as the number of variables is polynomially bounded. For the completeness of Π-ETR, it turns out that an unbounded product is able to simulate an arbitrary number of Boolean quantifier alternations, in constrast to an unbounded sum. So, as shown in Lemma 16, we can reduce QBF to it. ◀

8 ETR with the Standard Summation Operator

In the previous section, we have seen that ETR with a unary product operator (Π-ETR) is PSPACE-complete. Moreover, allowing both unary summation and product operators does not lead to an increase in complexity. In this section, we investigate the complexity of Σ-ETR, ETR with only unary summation operators.

▶ **Definition 20.** *Let* $\exists\mathbb{R}^{\Sigma}$ *be the closure of* Σ-*ETR under polynomial time many one reductions. Moreover, for completeness, let* $\exists\mathbb{R}^{\Pi}$ *be the closure of* Π-*ETR under polynomial time many one reductions.*

8.1 Machine Characterization of $\exists\mathbb{R}^{\Sigma}$

By Theorem 19, we have $\exists\mathbb{R}^{\Pi} = \mathsf{PSPACE}$. For $\exists\mathbb{R}^{\Sigma}$, we can conclude: $\mathsf{NP}_{\text{real}} = \exists\mathbb{R} \subseteq \exists\mathbb{R}^{\Sigma} \subseteq$ PSPACE. We conjecture that all inclusions are strict. In this section, we will provide some arguments in favor of this.

We first observe that using summations we can quite easily solve PP-problems. In particular, we have:

▶ **Lemma 21.** $\mathsf{NP}^{\mathsf{PP}} \subseteq \exists\mathbb{R}^{\Sigma}$.

Proof. The canonical $\mathsf{NP}^{\mathsf{PP}}$-complete problem E-MajSat is deciding the satisfiability of a formula

$$\psi : \exists x_1, \ldots, x_n : \#\{(y_1, \ldots, y_n) \in \{0,1\}^n \mid \phi(x, y) = 1\} \geq 2^{n-1},$$

i.e., deciding whether there is an assignment to the x-variables such that the resulting formula is satisfied by at least half of the assignments to the y-variables [18].

Let $X_1 \ldots X_n, Y_1 \ldots Y_n$ be real variables and $\phi^{\mathbb{R}}$ the arithmetization of ϕ. We build an equivalent $\exists\mathbb{R}^{\Sigma}$ instance as follows:

1. $X_i = 0 \vee X_i = 1$, $1 \leq i \leq n$, and
2. $\sum_{Y_1=0}^{1} \cdots \sum_{Y_n=0}^{1} \phi^{\mathbb{R}}(X, Y) \geq 2^{n-1}$.

Then this instance is satisfiable iff ψ is satisfiable because X_i are existentially chosen and constraint to be Boolean and $\sum_{Y_1=0}^{1} \cdots \sum_{Y_n=0}^{1} \phi^{\mathbb{R}}(X, Y)$ is exactly the number of satisfying assignments to the Y-variables. ◀

Similarly to $\exists\mathbb{R} = \mathsf{NP}_{\text{real}}$, we can also characterize $\exists\mathbb{R}^{\Sigma}$ using a machine model instead of a closure of a complete problem under polynomial time many one reductions. For this we define a $\mathsf{NP}_{\text{real}}^{\mathsf{VNP}_{\mathbb{R}}}$ machine to be an $\mathsf{NP}_{\text{real}}$ machine with a $\mathsf{VNP}_{\mathbb{R}}$ oracle, where $\mathsf{VNP}_{\mathbb{R}}$ denotes Valiant's NP over the reals. Since $\mathsf{VNP}_{\mathbb{R}}$ is a family of polynomials, the oracle allows us to evaluate a family of polynomials, for example the permanent, at any real input[8]. The two lemmas below demonstrate that $\mathsf{NP}_{\text{real}}^{\mathsf{VNP}_{\mathbb{R}}}$ coincides with $\exists\mathbb{R}^{\Sigma}$ which strengthens Lemma 21 that $\mathsf{NP}^{\mathsf{PP}} \subseteq \exists\mathbb{R}^{\Sigma}$ and characterizes Σ-ETR in terms of complexity classes over the reals.

▶ **Lemma 22.** *Σ-ETR $\in \mathsf{NP}_{real}^{\mathsf{VNP}_{\mathbb{R}}}$. This also holds if the NP_{real} machine is only allowed to call its oracle once.*

▶ **Lemma 23.** *Σ-ETR is hard for $\mathsf{NP}_{real}^{\mathsf{VNP}_{\mathbb{R}}}$.*

▶ **Theorem 24.** *Σ-ETR is complete for $\mathsf{NP}_{real}^{\mathsf{VNP}_{\mathbb{R}}}$. Thus, $\exists\mathbb{R}^{\Sigma} = \mathsf{NP}_{real}^{\mathsf{VNP}_{\mathbb{R}}}$.*

Proof idea. To prove Lemma 22, we show a normal form for Σ-ETR instances such that all polynomials contained in it are of the form $\sum_{Y \in \{0,1\}^m} p(X, Y)$ where p does not contain any unary sums. Then we show how to translate formulas in this normal form into a real word-RAM with oracle access. For the hardness results of Lemma 23, we encode the real word-RAM computations into an ETR-instance, where the oracle calls (which w.l.o.g. can be assumed to be calls to the permanent) are simulated by the summation operator. ◀

8.2 Reasoning about Probabilities in Small Models

In this section, we employ the satisfiability problems for languages of the causal hierarchy. The problem $\text{SAT}_{sm,prob}^{poly\langle\Sigma\rangle}$ is defined like $\text{SAT}_{prob}^{poly\langle\Sigma\rangle}$, but in addition we require that a satisfying distribution has only polynomially large support, that is, only polynomially many entries in the exponentially large table of probabilities are nonzero. Formally we can achieve this by extending an instance with an additional unary input $p \in \mathbb{N}$ and requiring that the satisfying distribution has a support of size at most p. The membership proofs of SAT_{prob}^{poly} in NP and

[8] See the full version for an overview of the relevant definitions.

in $\exists\mathbb{R}$, respectively, by [11], [17], and [21] rely on the fact that the considered formulas have the small model property: If the instance is satisfiable, then it is satisfiable by a small model. For $\text{SAT}^{poly\langle\Sigma\rangle}_{prob}$, this does not seem to be true because we can directly force any model to be arbitrarily large, e.g., by encoding the additional parameter p above in binary or by enforcing a uniform distribution using $\sum_{x_1}\cdots\sum_{x_n}(P(X_1{=}x_1,\ldots,X_n{=}x_n)-P(X_1{=}0,\ldots,X_n{=}0))^2=0$. Thus, we have to explicitly require that the models are small, yielding the problem $\text{SAT}^{poly\langle\Sigma\rangle}_{sm,prob}$. Formally, we use the following:

▶ **Definition 25.** *The decision problems* $\text{SAT}^{poly\langle\Sigma\rangle}_{sm,prob}$ *take as input a formula* $\varphi\in\mathcal{L}^{poly\langle\Sigma\rangle}$ *and a unary encoded number* $p\in\mathbb{N}$ *and ask whether there exists a model* $\mathfrak{M}=(\{X_1,\ldots,X_n\},P)$ *such that* $\mathfrak{M}\models\varphi$ *and* $\#\{(x_1,\ldots,x_n):P(X_1{=}x_1,\ldots,X_n{=}x_n)>0\}\le p$.

It turns out that $\text{SAT}^{poly\langle\Sigma\rangle}_{sm,prob}$ is a natural complete problem for $\exists\mathbb{R}^{\Sigma}$:

▶ **Theorem 26.** *The decision problem* $\text{SAT}^{poly\langle\Sigma\rangle}_{sm,prob}$ *is complete for* $\exists\mathbb{R}^{\Sigma}$.

Proof idea. To show the containment of $\text{SAT}^{poly\langle\Sigma\rangle}_{sm,prob}$ in $\exists\mathbb{R}^{\Sigma}$, we first show a normal form that every probability occurring in the input instance contains all variables. Then we have to use the exponential sum and the polynomially many variables to "built" a probability distribution with polynomial support. The lower bound follows from reducing from a restricted Σ-ETR-instance. ◀

9 Discussion

Traditionally, ETR has been used to characterize the complexity of problems from geometry and real optimization. It has recently been used to characterize probabilistic satisfiability problems, which play an important role in AI, see e.g. [21, 31]. We have further investigated the recently defined class succ-$\exists\mathbb{R}$, characterized it in terms of real word-RAMs, and shown the existence of further natural complete problems. Moreover, we defined a new class $\exists\mathbb{R}^{\Sigma}$ and also gave natural complete problems for it.

The studied summation operators allow the encoding of exponentiation, but only with integer bases, so they do not affect the decidability, unlike Tarski's exponential function [29].

Schäfer and Stefankovic [27] consider extensions of ETR where we have a constant number of alternating quantifiers instead of just one existential quantifier. By the work of Grigoriev and Vorobjov [14], these classes are all contained in PSPACE. Can we prove a real version of Toda's theorem [30]? Are these classes contained in $\exists\mathbb{R}^{\Sigma}$?

References

1 Mikkel Abrahamsen, Anna Adamaszek, and Tillmann Miltzow. The art gallery problem is ∃ℝ-complete. In *Proc. of the 50th ACM SIGACT Symposium on Theory of Computing (STOC 2018)*, pages 65–73, 2018. doi:10.1145/3188745.3188868.

2 Mikkel Abrahamsen, Linda Kleist, and Tillmann Miltzow. Training neural networks is ∃ℝ-complete. In *Proc. Advances in Neural Information Processing Systems (NeurIPS 2021)*, pages 18293–18306, 2021.

3 Elias Bareinboim, Juan D. Correa, Duligur Ibeling, and Thomas Icard. *On Pearl's Hierarchy and the Foundations of Causal Inference*, pages 507–556. Association for Computing Machinery, New York, NY, USA, 2022. doi:10.1145/3501714.3501743.

4 Vittorio Bilò and Marios Mavronicolas. ∃ℝ-complete decision problems about symmetric Nash equilibria in symmetric multi-player games. In *Proc. 34th Symposium on Theoretical Aspects of Computer Science (STACS 2017)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017.

5 Lenore Blum, Mike Shub, and Steve Smale. On a theory of computation and complexity over the real numbers: NP-completeness, recursive functions and universal machines. *Bulletin of the American Mathematical Society*, 21(1):1–46, 1989.

6 Peter Bürgisser. *Completeness and reduction in algebraic complexity theory*, volume 7. Springer Science & Business Media, 2000.

7 John Canny. Some algebraic and geometric computations in PSPACE. In *Proc. of the 20th ACM Symposium on Theory of Computing (STOC 1988)*, pages 460–467. ACM, 1988. doi:10.1145/62212.62257.

8 Jean Cardinal. Computational geometry column 62. *ACM SIGACT News*, 46(4):69–78, 2015. doi:10.1145/2852040.2852053.

9 Arnaud Durand, Miika Hannula, Juha Kontinen, Arne Meier, and Jonni Virtema. Probabilistic team semantics. In *Foundations of Information and Knowledge Systems: 10th International Symposium, FoIKS 2018, Proceedings 10*, pages 186–206. Springer, 2018. doi:10.1007/978-3-319-90050-6_11.

10 Jeff Erickson, Ivor Van Der Hoog, and Tillmann Miltzow. Smoothing the gap between NP and ER. *SIAM Journal on Computing*, pages FOCS20–102, 2022.

11 Ronald Fagin, Joseph Y Halpern, and Nimrod Megiddo. A logic for reasoning about probabilities. *Information and Computation*, 87(1-2):78–128, 1990. doi:10.1016/0890-5401(90)90060-U.

12 Jugal Garg, Ruta Mehta, Vijay V Vazirani, and Sadra Yazdanbod. ∃ℝ-completeness for decision versions of multi-player (symmetric) Nash equilibria. *ACM Transactions on Economics and Computation (TEAC)*, 6(1):1–23, 2018. doi:10.1145/3175494.

13 George Georgakopoulos, Dimitris Kavvadias, and Christos H. Papadimitriou. Probabilistic satisfiability. *Journal of Complexity*, 4(1):1–11, 1988. doi:10.1016/0885-064X(88)90006-4.

14 Dima Grigoriev and Nicolai N. Vorobjov Jr. Solving systems of polynomial inequalities in subexponential time. *J. Symb. Comput.*, 5(1/2):37–64, 1988. doi:10.1016/S0747-7171(88)80005-1.

15 Miika Hannula, Minna Hirvonen, Juha Kontinen, Yasir Mahmood, Arne Meier, and Jonni Virtema. Logics with probabilistic team semantics and the boolean negation. *arXiv preprint arXiv:2306.00420*, 2023. doi:10.48550/arXiv.2306.00420.

16 Miika Hannula, Juha Kontinen, Jan Van den Bussche, and Jonni Virtema. Descriptive complexity of real computation and probabilistic independence logic. In *Proc. of the 35th ACM/IEEE Symposium on Logic in Computer Science (LICS 2020)*, pages 550–563, 2020. doi:10.1145/3373718.3394773.

17 Duligur Ibeling and Thomas Icard. Probabilistic reasoning across the causal hierarchy. In *Proc. 34th AAAI Conference on Artificial Intelligence (AAAI 2020)*, pages 10170–10177. AAAI Press, 2020. doi:10.1609/AAAI.V34I06.6577.

18 Michael L. Littman, Judy Goldsmith, and Martin Mundhenk. The computational complexity of probabilistic planning. *Journal of Artificial Intelligence Research*, 9:1–36, 1998. doi:10.1613/JAIR.505.

19 Meena Mahajan. Algebraic complexity classes. *Perspectives in Computational Complexity: The Somenath Biswas Anniversary Volume*, pages 51–75, 2014.

20 Colin McDiarmid and Tobias Müller. Integer realizations of disk and segment graphs. *Journal of Combinatorial Theory, Series B*, 103(1):114–143, 2013. doi:10.1016/J.JCTB.2012.09.004.

21 Milan Mossé, Duligur Ibeling, and Thomas Icard. Is causal reasoning harder than probabilistic reasoning? *The Review of Symbolic Logic*, pages 1–26, 2022.

22 Nils J. Nilsson. Probabilistic logic. *Artificial Intelligence*, 28(1):71–87, 1986. doi:10.1016/0004-3702(86)90031-7.

23 Judea Pearl. *Causality.* Cambridge University Press, 2009.

24 James Renegar. On the computational complexity and geometry of the first-order theory of the reals. Part I: Introduction. Preliminaries. The geometry of semi-algebraic sets. The decision problem for the existential theory of the reals. *Journal of symbolic computation*, 13(3):255–299, 1992. doi:10.1016/S0747-7171(10)80003-3.

25 Marcus Schaefer. Complexity of some geometric and topological problems. In *Proc. International Symposium on Graph Drawing (GD 2009)*, pages 334–344. Springer, 2009. doi:10.1007/978-3-642-11805-0_32.

26 Marcus Schaefer, Jean Cardinal, and Tillmann Miltzow. The existential theory of the reals as a complexity class: A compendium. *arXiv preprint*, 2024. doi:10.48550/arXiv.2407.18006.

27 Marcus Schaefer and Daniel Štefankovič. Beyond the existential theory of the reals. *Theory of Computing Systems*, 68(2):195–226, 2024. doi:10.1007/S00224-023-10151-X.

28 Ilya Shpitser and Judea Pearl. Complete identification methods for the causal hierarchy. *Journal of Machine Learning Research*, 9:1941–1979, 2008. doi:10.5555/1390681.1442797.

29 Alfred Tarski. A decision method for elementary algebra and geometry. *Journal of Symbolic Logic*, 14(3), 1949.

30 Seinosuke Toda. PP is as hard as the Polynomial-Time Hierarchy. *SIAM Journal on Computing*, 20(5):865–877, 1991. doi:10.1137/0220053.

31 Benito van der Zander, Markus Bläser, and Maciej Liśkiewicz. The hardness of reasoning about probabilities and causality. In *Proc. Joint Conference on Artificial Intelligence (IJCAI 2023)*, 2023.