The Hardness of Reasoning about Probabilities and Causality

Benito van der Zander[1] , **Markus Bläser**[2] and **Maciej Liśkiewicz**[1]
[1]Institute of Theoretical Computer Science, University of Lübeck, Germany
[2]Saarland University, Saarland Informatics Campus, Saarbrücken, Germany
benito@tcs.uni-luebeck.de, mblaeser@cs.uni-saarland.de, liskiewi@tcs.uni-luebeck.de

Abstract

We study formal languages which are capable of fully expressing quantitative probabilistic reasoning and do-calculus reasoning for causal effects, from a computational complexity perspective. We focus on satisfiability problems whose instance formulas allow expressing many tasks in probabilistic and causal inference. The main contribution of this work is establishing the exact computational complexity of these satisfiability problems. We introduce a new natural complexity class, named $\mathtt{succ}\exists\mathbb{R}$, which can be viewed as a succinct variant of the well-studied class $\exists\mathbb{R}$, and show that these problems are complete for $\mathtt{succ}\exists\mathbb{R}$. Our results imply even stronger limitations on the use of algorithmic methods for reasoning about probabilities and causality than previous state-of-the-art results that rely only on the NP- or $\exists\mathbb{R}$-completeness of the satisfiability problems for some restricted languages.

1 Introduction

Satisfiability problems play a key role in a broad range of research fields, including AI, since many real-life tasks can be reduced in a natural and efficient way to instances of satisfiability expressed in a suitable formal language. A prominent example is the Boolean satisfiability problem (SAT) whose instances represent Boolean formulas of propositional logic and ask whether there exists an assignment that satisfies a given formula. It is well known that any decision problem in the complexity class NP can be reduced in polynomial time to the SAT problem, see [Garey and Johnson, 1979].

In this work, we investigate satisfiability problems (and their validity counterparts) whose instance formulas allow expressing many tasks in probabilistic and causal inference. The formulas are specified in languages commonly used in pure probabilistic and interventional reasoning. In particular, the probabilistic language, called in Pearl's Causal Hierarchy[1] the *associational* one, consists of Boolean combinations of (in)equalities involving pure probabilities as, e.g., $\mathbb{P}(y|x)$ which[2] can ask *how likely is $Y = y$ given that observing $X = x$?* An example formula in this language is the following single equality

$$\sum_{u,x,z,y} (\mathbb{P}(u,x,z,y) - \mathbb{P}(u)\mathbb{P}(x|u)\mathbb{P}(z|x)\mathbb{P}(y|z,u))^2 = 0 \qquad (1)$$

which expresses the fact that the joint distribution can be factorized as $\mathbb{P}(u,x,z,y) = \mathbb{P}(u)\mathbb{P}(x|u)\mathbb{P}(z|x)\mathbb{P}(y|z,u)$, for all values u,x,z,y. The causal language extends the probabilistic language by allowing additionally to use terms involving Pearl's do-operator [Pearl, 2009] as, e.g., $\mathbb{P}(y|do(x))$ which can ask hypothetical questions such as *how likely would $Y = y$ be given an intervention setting X to x?* An example formula in this language is the single equality

$$\mathbb{P}(y|do(x)) = \textstyle\sum_z \mathbb{P}(z|x) \sum_{x'} \mathbb{P}(y|x',z)\mathbb{P}(x') \qquad (2)$$

which allows estimating the (total) causal effect of the intervention $X = x$ on outcome variable $Y = y$ via the prominent front-door adjustment [Pearl, 1995]. It is well known that this formalism enables inference of properties that are impossible to reason about from correlational data using a purely probabilistic framework [Shpitser and Pearl, 2008; Bareinboim *et al.*, 2022].

Similarly as in the classical SAT, both languages allow natural, polynomial-time reductions of many tasks in probabilistic and causal inference to the satisfiability (or validity) problems. Prominent examples can be establishing properties of *structural causal models* [Glymour *et al.*, 2014; Pearl, 2009; Koller and Friedman, 2009; Elwert, 2013] which endow researchers with graphical structures (also called causal Bayesian networks) to encode conditional independences and causal assumptions as a directed acyclic graph (DAG). For instance, the problem to decide, for a given DAG $\mathcal{G}$ and variables X, Y, Z, whether in every structural causal model compatible with the structure $\mathcal{G}$, the causal effect of X on Y can be inferred via front-door adjustment (2) can be reduced in polynomial time to the validity problem. Indeed, for any $\mathcal{G}, X, Y, Z$, one can compute a formula φ whose size is bounded by a polynomial in the number of nodes of $\mathcal{G}$, such that the front-door adjustment is applicable, if and only if, φ is valid. E.g., for the DAG $\mathcal{G}$:

[1]See [Pearl and Mackenzie, 2018] for a first-time introduction to the topic.

[2]In our paper, by $\mathbb{P}(y|x)$, $\mathbb{P}(y|do(x))$, etc., we mean $\mathbb{P}(Y{=}y|X{=}x)$, $\mathbb{P}(Y{=}y|do(X{=}x))$, etc.

$$\begin{tikzcd}
 & U \arrow[dl] \arrow[dr] & \\
X \arrow[r] & Z \arrow[r] & Y
\end{tikzcd}$$

the formula φ looks as follows $(\psi_1 \wedge \psi_2) \Rightarrow \varphi_{\mathit{fda}}$, where φ_{fda} is the equality (2) representing the front-door adjustment, ψ_1 is the equality (1) encoding the probability factorization, and ψ_2 is a conjunction of equalities, whose conjuncts encode edge orientations. E.g., $X \to Z$ can be expressed as

$$\textstyle\sum_{x,z}(\mathbb{P}(x|do(z)) - \mathbb{P}(x))^2 = 0. \qquad (3)$$

The correctness of this reduction follows from the fact, that a structural causal model over X, Y, Z, U is supplemented with the above DAG $\mathcal{G}$ if $\psi_1 \wedge \psi_2$ encodes $\mathcal{G}$. Interestingly, using do-calculus, one can prove that the front-door adjustment is applicable for the discussed instance which means that the formula $(\psi_1 \wedge \psi_2) \Rightarrow \varphi_{\mathit{fda}}$ is valid.

1.1 Our Contribution

Despite its importance, computational complexity aspects of the satisfiability problems for the general probabilistic and causal languages remain unexplored. In our work, we investigate these issues and establish, for the first time, the exact computational complexity of the satisfiability problem for probabilistic languages, denoted as $\text{SAT}_{\mathit{prob}}$, and for the interventional level of the causal hierarchy, denoted as $\text{SAT}_{\mathit{interven}}$. They simultaneously indicate the complexity of the validity problems for these languages.

Our results are based on a novel extension of the well-studied complexity class $\exists\mathbb{R}$ which, loosely speaking, consists of problems that are polynomial time reducible to deciding whether a system of polynomial (in)equalities over real unknowns has a solution. The class $\exists\mathbb{R}$ includes, in an obvious way, NP and, what is highly nontrivial, it is contained in PSPACE [Grigoriev and Vorobjov, 1988; Canny, 1988; Schaefer, 2009]. However, none of the inclusions $\mathtt{NP} \subseteq \exists\mathbb{R} \subseteq \mathtt{PSPACE}$ are known to be strict. Our new complexity class, named $\mathtt{succ}\exists\mathbb{R}$, can be viewed as a succinct variant of $\exists\mathbb{R}$ and it is an intermediate class between the exponential versions of NP and PSPACE:

$$\mathtt{NEXPTIME} \subseteq \mathtt{succ}\exists\mathbb{R} \subseteq \mathtt{EXPSPACE}. \qquad (4)$$

The main contribution of this paper is to show that deciding the satisfiability of both probabilistic reasoning and reasoning about causal interventions is complete for $\mathtt{succ}\exists\mathbb{R}$:

Theorem 1.1 (Main). *The satisfiability problems* $\text{SAT}_{\mathit{prob}}$ *and* $\text{SAT}_{\mathit{interven}}$ *are* $\mathtt{succ}\exists\mathbb{R}$*-complete.*

Since $\mathtt{succ}\exists\mathbb{R}$ contains all problems decidable in nondeterministic exponential time, this shows a significant jump in complexity compared to the classic SAT problem for propositional logic which is known to be NP-complete.

1.2 The Existential Theory of the Reals

The Existential Theory of the Reals (ETR) asks, given a Boolean combination of (in)equalities of multivariate polynomials, called a sentence, whether there exists an assignment of real numbers to the variables of the polynomials that satisfies the combination of (in)equalities? The example

$$\exists x \exists y \quad x^2 - y = 0 \,\wedge\, x^3 - x = 0 \,\wedge\, (x > 0 \,\vee\, y \leq -1)$$

illustrates a *yes* instance of ETR because $(x, y) = (1, 1)$ is a solution of the sentence. The theory forms its own complexity class $\exists\mathbb{R}$ which is defined as the closure of the ETR under polynomial time many-one reductions. The importance of $\exists\mathbb{R}$ is underlined by the fact, that many meaningful problems including algebraic, geometric, graph- and game-theory, machine learning, and continuous constraint satisfaction problems, have been classified as complete for $\exists\mathbb{R}$, see e.g. [Schaefer and Štefankovič, 2017; Abrahamsen *et al.*, 2018; Garg *et al.*, 2018; Abrahamsen *et al.*, 2021; Miltzow and Schmiermann, 2022]. Moreover, assuming $\mathtt{NP} \neq \exists\mathbb{R}$ which is widely believed, the $\exists\mathbb{R}$-completeness of a problem reflects the limitation on the use of algorithmic approaches to solve the problem, as, e.g., dynamic programming, divide and conquer, tree-width based algorithms, or SAT-solver approaches which are applicable for instances of NP-complete problems. The $\mathtt{succ}\exists\mathbb{R}$-completeness of $\text{SAT}_{\mathit{prob}}$ and $\text{SAT}_{\mathit{interven}}$ (Theorem 1.1) imply even stronger algorithmic limitations than could be deduced from the $\exists\mathbb{R}$-completeness results of [Mossé *et al.*, 2022] (discussed below) which concerns, however, some restricted variants of the languages.

1.3 Previous Work on the Hardness of Satisfiability Problems

In the past decades, several research groups have studied the computational complexity aspects of satisfiability problems for some limited languages of probabilistic and causal inference. In their pioneering work, Fagin *et al.* [1990] consider a language for reasoning about pure probabilities, which consists of Boolean combinations of *linear* (in)equalities over probabilities, like $\mathbb{P}(X{=}1 \vee X{=}2) + 2\mathbb{P}(Y{=}1) \leq 1/3 \,\wedge\, \mathbb{P}(Y{=}1) \geq 2/3$. The authors provide a complete axiomatization for the logic and show that the problem of deciding satisfiability is NP-complete which, surprisingly, is no worse than that of propositional logic. Later on, further studies explored the complexity aspects of probability logics [Abadi and Halpern, 1994; Speranski, 2013] and reasoning about causal models [Halpern, 2000; Eiter and Lukasiewicz, 2002; Aleksandrowicz *et al.*, 2017].

Most germane to our work are the recent studies of Ibeling and Icard [2020] and Mossé *et al.* [2022] which extended the formalism of [Fagin *et al.*, 1990], providing the more expressive languages for the second (*interventional*) and third (*counterfactual*) level of Pearl's Causal Hierarchy. The languages consist of Boolean combinations of *polynomial* (in)equalities over pure probabilities or over probabilities involving the do-operator, respectively. In particular, they allow to express *conditioning*, as, e.g., $\mathbb{P}(y|x) = \mathbb{P}(y, x)/\mathbb{P}(x)$. But the languages limit the use of *marginalization* since the summation operator Σ over the domain of random variables, as used, e.g., in Eq. (1)–(3), is not allowed. Consequently, to express the marginal distribution of a variable Y over a subset of variables $\{Z_1, \ldots, Z_m\} \subseteq \{X_1, \ldots, X_n\}$, as $\sum_{z_1,\ldots,z_m} \mathbb{P}(y, z_1, \ldots, z_m)$, in the language 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 in m. (In the expression, we assume that the Z_i are binary variables). Similarly, $\mathbb{P}(y|do(x)) = \sum_z \mathbb{P}(y|x, z)\mathbb{P}(z)$ is expressed equivalently as $\mathbb{P}(y|do(x)) = \mathbb{P}(y|x, Z{=}0)\mathbb{P}(Z{=}0) +$

$\mathbb{P}(y|x, Z{=}1)\mathbb{P}(Z{=}1)$. While this might be acceptable for one variable Z, the expansion of the sums grows exponentially when we have several variables. Therefore, having an explicit summation operator is highly desirable.

Ibeling and Icard [2020] and Mossé *et al.* [2022] give complete axiomatizations of these languages and investigate the computational complexity of the satisfiability problem for each language. In particular, they prove that, although the languages for interventional and counterfactual inference are more expressive than the ones for probabilistic reasoning, the computational complexities of the satisfiability problem for the logics are both $\exists\mathbb{R}$-complete. Combining these results with ours, we obtain a precise complexity theoretic classification of the satisfiability problems for all three levels of Pearl's Causal Hierarchy combined with various languages for terms involving probabilities (proven by †: Fagin *et al.* [1990], ‡: Mossé *et al.* [2022]):

Terms	*prob.*	*interven.*	*counterfact.*	*Source*
lin	NP-complete			†, ‡
poly	$\exists\mathbb{R}$-complete			‡
poly & Σ	succ$\exists\mathbb{R}$-complete		succ$\exists\mathbb{R}$-hard	Thm. 1.1

While the languages of [Mossé *et al.*, 2022] are capable of fully expressing quantitative probabilistic reasoning, respectively, do-calculus reasoning for causal effects, as discussed above, due to the lack of the summation operator Σ, they do not capture the standard notation used commonly in probabilistic and causal inference. Consequently, to analyze the computational complexity aspects of the inference, languages that exploit the standard notation need to be used. Indeed, for the computational complexity, the key issue is how the instances are represented since the time or space complexities are functions in the length of the input. For example, if the language as in [Mossé *et al.*, 2022] would be used (instead of the succinct one with summation operator), then the time complexity of many algorithms, including, e.g., the seminal Shpitser and Pearl [2006] algorithm to estimate the interventional distribution, would jump from polynomial to exponential. Another problem would also be to construct polynomial time reductions to the SAT problems since using the succinct encodings as, e.g., in Eq. (1)–(3), would not be allowed.

Structure of this Paper. The remainder of this work is dedicated to proving Theorem 1.1. After providing preliminaries (Sec. 2 and 3), we introduce the class succ$\exists\mathbb{R}$ in Sec. 4 and provide the first complete problems for succ$\exists\mathbb{R}$ in Sec. 5. In Sec. 6, we prove the membership of SAT$_{prob}$ and SAT$_{interven}$ in the class succ$\exists\mathbb{R}$ and their succ$\exists\mathbb{R}$-hardness. The omitted proofs can be found in the appendix of a companion paper [van der Zander *et al.*, 2023].

2 Preliminaries

2.1 Syntax of Probabilistic and Causal Languages

The syntax used in this paper extends the definitions in [Ibeling and Icard, 2020; Mossé *et al.*, 2022] for the languages of the first two levels of Pearl's Causal Hierarchy. We consider discrete distributions and represent the values of the random variables as $Val = \{0, 1, \ldots, c-1\}$ and denote by $\mathbf{X}$ the set of all (endogenous) random variables. By capital letters $X_1, X_2, \ldots$, we denote the individual variables and assume, w.l.o.g., that they all share the same domain Val. A value of X_i is denoted by x_i or a natural number.

In order to reason about the (in)equalities of arithmetic terms involving probability expressions, we need to define languages describing probabilities, languages describing arithmetic terms, and languages describing (in)equalities. The first languages characterize probabilistic and causal events as Boolean conditions over the values of (endogenous) random variables:

$$\begin{aligned}\mathcal{L}_{prop} &::= X = x \mid \neg\mathcal{L}_{prop} \mid \mathcal{L}_{prop} \wedge \mathcal{L}_{prop}\\ \mathcal{L}_{int} &::= \top \mid X = x \mid \mathcal{L}_{int} \wedge \mathcal{L}_{int}\\ \mathcal{L}_{post\text{-}int} &::= [\mathcal{L}_{int}]\mathcal{L}_{prop}\end{aligned}$$

where $X \in \mathbf{X}$, and x is either in Val or is a summation variable as defined below.

To make the notation more convenient for our analysis, we use $[\mathcal{L}_{int}]$ in the syntax above to describe an *intervention* on certain variables. The operator $[\cdot]\delta$ creates a new *post-intervention* model and the assigned propositional formula δ only applies to the new model. Thus, for example, $\mathbb{P}([X{=}x]Y{=}y)$, which, using do-notation can be expressed as $\mathbb{P}(Y{=}y|do(X{=}x))$, denotes the probability that the variable Y equals y in the model after the intervention $do(X{=}x)$. Since $\top$ means that no intervention has been applied, we can assume that $\mathcal{L}_{prop} \subseteq \mathcal{L}_{post\text{-}int}$.

Inserting the primitives into arithmetic expressions, we get the language $T(\mathcal{L})$, where, for $\mathcal{L} \in \{\mathcal{L}_{prop}, \mathcal{L}_{post\text{-}int}\}$, $\delta \in \mathcal{L}$, and $\delta' \in \mathcal{L}_{prop}$, any $\mathbf{e} \in T(\mathcal{L})$ is formed by the grammar[3]:

$$\mathbf{e} ::= \mathbb{P}(\delta|\delta') \mid \mathbf{e} + \mathbf{e}' \mid \mathbf{e} \cdot \mathbf{e}' \mid \textstyle\sum_x \mathbf{e}. \qquad (5)$$

The probabilities of the form $\mathbb{P}(\delta)$ or $\mathbb{P}(\delta|\delta')$, are called *primitives* or *(atomic) terms*.

In the summation operator $\sum_x$ in definition (5), we have a dummy variable x which ranges over all values $0, 1, \ldots, c-1$. The summation $\sum_x \mathbf{e}$ is a purely syntactical concept which represents the sum $\mathbf{e}[^0/_x] + \mathbf{e}[^1/_x] + \ldots + \mathbf{e}[^{c-1}/_x]$, where, by $\mathbf{e}[^v/_x]$, we mean the expression in which all occurrences of x are replaced with value v. E.g., for $Val = \{0, 1\}$, the expression[4] $\sum_x \mathbb{P}(Y{=}1, X{=}x)$ 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.

Finally, we define the languages of Boolean combinations of inequalities, following the grammar, where $\mathbf{e}, \mathbf{e}'$ are expressions in $T(\mathcal{L}_{prop})$ for *prob* and $T(\mathcal{L}_{post\text{-}int})$ for *causal*, respectively:

$$\begin{aligned}\mathcal{L}_{prob} &::= \mathbf{e} \le \mathbf{e}' \mid \neg\mathcal{L}_{prob} \mid \mathcal{L}_{prob} \wedge \mathcal{L}_{prob}\\ \mathcal{L}_{interven} &::= \mathbf{e} \le \mathbf{e}' \mid \neg\mathcal{L}_{interven} \mid \mathcal{L}_{interven} \wedge \mathcal{L}_{interven}.\end{aligned}$$

Together with conjunctions, the comparisons yield an equality relation. Moreover, for the atomic formula $X{=}x$ in

[3]This is a recursive definition where, e.g., $\mathbf{e} + \mathbf{e}'$ means, for $e, e' \in T(\mathcal{L})$, the expression $e + e'$ is also in $T(\mathcal{L})$.

[4]As usually, $\mathbb{P}(Y{=}1, X{=}x)$, etc., means $\mathbb{P}(Y{=}1 \wedge X{=}x)$, etc.

probabilities as, e. g. $\mathbb{P}(X{=}x)$, we use a common abbreviation $\mathbb{P}(x)$ if this does not lead to confusion. Thus, for example, Eq. (1) builds the correct formula in the language $\mathcal{L}_{prob}$ and Eq. (2) and (3) are the correct formulas in $\mathcal{L}_{interven}$.

Although the language and its operations can appear rather restricted, all the usual elements of probabilistic and causal formulas can be encoded. Namely, equality is encoded as greater-or-equal in both directions, e.g. $\mathbb{P}(x) = \mathbb{P}(y)$ means $\mathbb{P}(x) \geq \mathbb{P}(y) \wedge \mathbb{P}(y) \geq \mathbb{P}(x)$. Subtraction or divisions can be encoded by moving a term to the other side of the equation, e.g., $\mathbb{P}(x) - \mathbb{P}(y) = \mathbb{P}(z)$ means $\mathbb{P}(x) = \mathbb{P}(z) + \mathbb{P}(y)$, and $\mathbb{P}(x)/\mathbb{P}(y) = \mathbb{P}(z)$ means $\mathbb{P}(x) = \mathbb{P}(z)\mathbb{P}(y)$. Any positive integer can be encoded from the fact $\mathbb{P}(\top) \equiv 1$, e.g. $4 \equiv (1+1)(1+1) \equiv (\mathbb{P}(\top)+\mathbb{P}(\top))(\mathbb{P}(\top)+\mathbb{P}(\top))$. The number 0 can be encoded as inconsistent probability, i.e., $\mathbb{P}(X{=}1 \wedge X{=}2)$. 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.

2.2 Semantics

We define a structural causal model (SCM) as in [Pearl, 2009, Sec. 3.2]. An SCM is a tuple $\mathfrak{M} = (\mathcal{F}, P, \mathbf{U}, \mathbf{X})$, such that $\mathbf{V} = \mathbf{U} \cup \mathbf{X}$ is a set of variables partitioned into exogenous (unobserved) variables $\mathbf{U} = \{U_1, U_2, \ldots\}$ and endogenous variables $\mathbf{X}$. The tuple $\mathcal{F} = \{F_1, \ldots, F_n\}$ consists of functions such that function F_i calculates the value of variable X_i from the values $(\mathbf{x}, \mathbf{u})$ of other variables in $\mathbf{V}$ as $F_i(x_{i_1}, \ldots, x_{i_k}, \mathrm{u}_i)$ [5]. P specifies a probability distribution of all exogenous variables $\mathbf{U}$. Since variables $\mathbf{X}$ depend deterministically on the exogenous variables via functions F_i, $\mathcal{F}$ and P define the obvious joint probability distribution of $\mathbf{X}$.

For any atomic $\mathcal{L}_{int}$-formula $X_i{=}x_i$ (which, in our notation, means $do(X_i{=}x_i)$), we denote by $\mathcal{F}_{X_i=x_i}$ the functions obtained from $\mathcal{F}$ by replacing F_i with the constant function $F_i(\mathbf{v}) := x_i$. We generalize this definition for any interventions specified by $\alpha \in \mathcal{L}_{int}$ in a natural way and denote as $\mathcal{F}_\alpha$ the resulting functions.

For any $\varphi \in \mathcal{L}_{prop}$, we write $\mathcal{F}, \mathbf{u} \models \varphi$ if φ is satisfied for values of $\mathbf{X}$ calculated from the values $\mathbf{u}$. For $[\alpha]\varphi \in \mathcal{L}_{post\text{-}int}$, we write $\mathcal{F}, \mathbf{u} \models [\alpha]\varphi$ if $\mathcal{F}_\alpha, \mathbf{u} \models \varphi$. Finally, for $\psi \in \mathcal{L}_{post\text{-}int}$, let $S_{\mathfrak{M}}(\psi) = \{\mathbf{u} \mid \mathcal{F}, \mathbf{u} \models \psi\}$. We assume, as is standard, the measurability of $\mathfrak{M}$ which guarantees that $S_{\mathfrak{M}}(\psi)$ is always measurable.

We define $[\![\mathbf{e}]\!]_{\mathfrak{M}}$ recursively in a natural way, starting with atomic terms as follows: $[\![\mathbb{P}(\psi)]\!]_{\mathfrak{M}} = P(S_{\mathfrak{M}}(\psi))$, resp. $[\![\mathbb{P}([\alpha]\varphi|\varphi')]\!]_{\mathfrak{M}} = P(S_{\mathfrak{M}}([\alpha](\varphi \wedge \varphi')))/P(S_{\mathfrak{M}}([\alpha]\varphi'))$. 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 formula.

2.3 Complexity Notation

We use the well-known complexity classes NP, PSPACE, NEXPTIME, EXPSPACE, and $\exists\mathbb{R}$ [Arora and Barak, 2009]. For two computational problems A, B, we will write $A \leq_p B$ if A can be reduced to B in polynomial time, which means A is not harder to solve than B. A problem A is complete for a complexity class C, if $A \in$ C and, for every other problem $B \in$ C, it holds $B \leq_p A$. By co-C we denote the class of all problems A such that its complements $\overline{A}$ belong to C.

[5] We consider recursive models, that is, we assume the endogenous variables are ordered such that variable X_i (i.e. function F_i) is not affected by any X_j with $j > i$.

3 Satisfiability Problems

The decision problems SAT$_{interven}$ and SAT$_{prob}$ takes as input a formula φ in language $\mathcal{L}_{interven}$ and in $\mathcal{L}_{prob}$, respectively, and asks whether there exists a model $\mathfrak{M}$ such that $\mathfrak{M} \models \varphi$. Analogously, we define the validity problems for languages $\mathcal{L}_{interven}$ and $\mathcal{L}_{prob}$ of deciding whether, for a given φ, $\mathfrak{M} \models \varphi$ holds for all models $\mathfrak{M}$. From the definitions, it is obvious that *causal* variants of the problems are at least as hard as the *prob* counterparts.

To give a first intuition about the expressive power of the SAT$_{prob}$ problem, we prove the following result which shows that SAT$_{prob}$ can encode any problem in NEXPTIME efficiently.

Proposition 3.1. *The* SAT$_{prob}$ *problem is* NEXPTIME*-hard.*

The remaining part of the paper is devoted to proving the main result (Theorem 1.1) showing that the satisfiability problems are complete for the new class succ$\exists\mathbb{R}$. Since a formula φ is *not* valid, if and only if $\neg\varphi$ is satisfiable and $\neg\varphi$ is in $\mathcal{L}_{prob}$ or in $\mathcal{L}_{interven}$, respectively, we can conclude from Theorem 1.1:

Corollary 3.2. *The validity problems for the languages $\mathcal{L}_{prob}$ and $\mathcal{L}_{interven}$ are complete for* co-succ$\exists\mathbb{R}$*, which is related to standard classes as follows:* co-NEXPTIME $\subseteq$ co-succ$\exists\mathbb{R}$ $\subseteq$ EXPSPACE.

4 The Existential Theory of the Reals, Succinctly

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

$$\exists x_1 \ldots \exists x_n \varphi(x_1, \ldots, x_n). \tag{6}$$

φ is a quantifier-free Boolean formula over the basis $\{\vee, \wedge, \neg\}$ and a signature consisting of the constants 0 and 1, the functional symbols $+$ and $\cdot$, and the relational symbols $<$, $\leq$, and $=$. The sentence is interpreted over the real numbers in the standard way.

All operations have an arity of at most two, so we can represent φ as a binary tree. The leaves of the tree are labeled with variables or constants, the inner nodes are labeled with $+, \cdot, =, <, \leq, \wedge, \vee$, or $\neg$. $\neg$-nodes have in-degree one, and all other inner nodes have in-degree two.

We now define succinct encodings of instances for ETR. Succinct encodings have been studied for problems in NP, see e.g. [Papadimitriou, 1994]. The input is now a Boolean circuit C computing a function $\{0,1\}^N \to \{0,1\}^M$, which encodes the input instance. For example, an instance of the well-known 3-SAT problem is some standard encoding of a given 3-CNF formula. In the succinct version of 3-SAT, we are given a circuit C. C encodes a 3-CNF formula ψ in the following way: $C(i)$ is an encoding of the ith clause of ψ. In

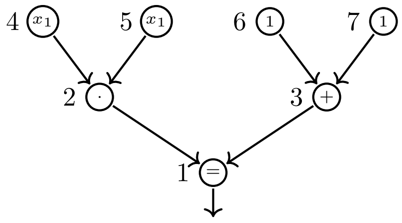

Figure 1: A tree representing the ETR instance $\exists x_1 : x_1^2 = 1 + 1$. The nodes set is $\{1, \dots, 7\}$, which we can represent by bit strings of length 3. The labels of the nodes are drawn in the circles.

this way, one can encode an exponentially large 3-SAT formula ψ (with up to 2^N clauses) . The succinct encoding typically induces a complexity jump: The succinct version of the 3-SAT problem is NEXPTIME-complete. The same is true for many other NP-complete problems [Papadimitriou, 1994].

In the case of ETR, the formula φ is now given succinctly by a Boolean circuit C. The circuit C computes a function $\{0,1\}^N \to \{0,1\}^M$. The input is (an encoding of) a node v of the formula and the output $C(v)$ contains the label of the node as well as its parent and its children (encoded as a string in binary). In this way, we can represent a formula that is exponentially large in the size of the circuit.

Example 4.1. Consider the formula $\exists x_1 : x_1^2 = 1 + 1$ ("2 has a square root over the reals"). As an instance of ETR, this formula would be encoded as a binary string in some standard way. Figure 1 shows the binary tree underlying the formula. There are seven nodes, which we can represent as integers $1, \dots, 7$. As a succETR instance, the formula will be given by a circuit C, computing a function $\{1, \dots, 7\} \to \{=, +, \cdot, 0, 1, x_1\} \times \{0, 1, \dots, 7\}^3$ describing the tree locally. For instance, $C(1) = (=, 0, 2, 3)$, means the following: the label of the node 1 is $=$. It has no parent, that is, it is the root, which is indicated by giving 0 as the parent. The left child is 2 and the right child is 3. In the same way $C(4) = (x_1, 2, 0, 0)$, which means that the node 4 is a leaf with parent 2 and it is labeled with the variable x_1. We can represent the integers ≤ 7 as bit strings of length $N = 3$ and the tuples $C(i)$, e.g., $(=, 0, 2, 3)$ or $(x_1, 2, 0, 0)$, can be encoded as bit strings of some length M. Thus, C can be considered a function $C : \{0,1\}^N \to \{0,1\}^M$. Note that we cannot use more variables than the formula has nodes, thus to encode the tuple $C(i)$, we need at most $M \leq (3 + N) + 3N$ bits: in the first entry, we need $3 + N$ bits to encode a label $+$, $\cdot$, $=$, $<$, $\leq$, $\wedge$, $\vee$, and $\neg$ or an index j of a variable x_j; to encode each of the remaining components, N bits suffice. It is clear that one can construct such a circuit C. When the formula is large but also structured, then the circuit C can be much smaller than the formula itself.

Definition 4.2 (succETR)**.** succETR is the set of all Boolean circuits C that encode a true sentence φ as in (6) as follows. Assume that C computes a function $\{0,1\}^N \to \{0,1\}^M$. Then $\{0,1\}^N$ is the node set of the tree underlying φ and $C(i)$ is the encoding of the description of node i, consisting of the label of i, its parent, and its two children. The variables in the formula are $x_1, \dots, x_{2^N}$.

The number of variables is formally always a power of 2. If we need fewer variables, we simply do not use them in the formula. We still quantify over all of them, but this is no problem. In the same way, the number of nodes is always a power of 2. If we need fewer nodes, we can define a special value for $C(i)$, which marks node i as unused.

We will always assume that all negations are pushed to the lowest level of the formula and then they are absorbed in the arithmetic terms, i.e., $\neg(s = t)$ becomes $(s < t) \vee (t < s)$, $\neg(s < t)$ becomes $t \leq s$ and $\neg(s \leq t)$ becomes $t < s$. Furthermore, we can remove the comparisons $\leq$, since we can replace $s \leq t$ by $(s < t) \vee (s = t)$.

Remark 4.3. In the non-succinct case, eliminating the negations by pushing them to the bottom is no problem, as it can be done explicitly. In the succinct case, it seems that we need to assume that already the given input has this property.

Finally, based on the succETR problem, we define the succinct version of $\exists\mathbb{R}$ as follows:

Definition 4.4. succ$\exists\mathbb{R}$ is the class of all problems that are polynomial time reducible to succETR.

succ$\exists\mathbb{R}$ is related to known complexity classes as follows.

Theorem 4.5. NEXPTIME $\subseteq$ succ$\exists\mathbb{R}$ $\subseteq$ EXPSPACE.

5 First Complete Problems for succ$\exists\mathbb{R}$

In this section, we present the first problems which are complete for succ$\exists\mathbb{R}$. The key role plays the problem Σ_{vi}-ETR which we need to prove the membership of SAT$_{interven}$ in the class succ$\exists\mathbb{R}$ in Proposition 6.1.

QUAD is the problem to decide whether a given family of quadratic multivariate polynomials has a common root. Here the polynomials are given as lists of monomials. The coefficient is represented as a quotient of two integers. QUAD is a complete problem for the class $\exists\mathbb{R}$ of all problems that are reducible to ETR [Schaefer and Štefankovič, 2017].

The succinct version succQUAD is defined as follows: We get a Boolean circuit C as input computing a function $\{0,1\}^K \times \{0,1\}^N \to \{0,1\}^M$. $C(x, y)$ is an encoding of the yth monomial of the xth equation.

We allow that a monomial appears several times in these lists and the coefficients all add up, so the final polynomials are the sum of all monomials in the list.

Lemma 5.1. succETR $\leq_p$ succQUAD. *Furthermore, the reduction only creates quadratic polynomials with at most four monomials and coefficients* ± 1.

Σ_{vi}-ETR is defined like ETR, but we add to the signature an additional summation operator. This is a unary operator $\sum_{x_j=a}^{b}$. Consider an arithmetic term given by a tree with the top gate $\sum_{x_j=a}^{b}$. Let $t(x_1, \dots, x_n)$ be the term computed at the child of the top gate. Then the new term computes $\sum_{e=a}^{b} t(x_1, \dots, x_{j-1}, e, x_{j+1}, \dots, x_n)$, that is, we replace the variable x_j by a summation variable e, which then runs from a to b.

Furthermore, we allow a new form of variable indexing in Σ_{vi}-ETR, namely variables of the form $x_{n(x_{j_1}, \dots, x_{j_m})}$: This can only be used when variables $x_{j_1}, \dots, x_{j_m}$ occur in the scope of summation operators and are replaced by summation variables $e_1, \dots, e_m$ with summation range $\{0, 1\}$.

$x_{n(x_{j_1},\dots,x_{j_m})}$ is interpreted as the variable with index given by $e_1,\dots,e_m$ interpreted as a number in binary.

Instances of Σ_{vi}-ETR are not given succinctly by a circuit. However, the ability to write down exponential sums succinctly makes the problem as hard as succETR. We get

Lemma 5.2. $\text{succQUAD} \le_p \Sigma_{vi}\text{-ETR} \le_p \text{succETR}$

which yields:

Theorem 5.3. *The problems* succQUAD *and* Σ_{vi}-ETR *are* $\texttt{succ}\exists\mathbb{R}$*-complete.*

6 Proof of the Main Theorem

Equipped with the tools provided in the previous sections, we are ready to prove our main result (Theorem 1.1) by showing:

$$\text{SAT}_{prob}, \text{SAT}_{interven} \in \texttt{succ}\exists\mathbb{R} \text{ and} \tag{7}$$

$$\text{SAT}_{prob}, \text{SAT}_{interven} \text{ are } \texttt{succ}\exists\mathbb{R}\text{-hard.} \tag{8}$$

We first show the membership of SAT_{prob} in the class $\texttt{succ}\exists\mathbb{R}$ proving the following reduction:

Proposition 6.1. $\text{SAT}_{interven} \le_p \Sigma_{vi}\text{-ETR}$.

Due to the transitivity of $\le_p$ and by Theorem 5.3, we get that $\text{SAT}_{interven} \le_p \text{succETR}$. Moreover, since $\text{SAT}_{prob} \le_p \text{SAT}_{interven}$, (7) follows.

For (8), it is sufficient to show the $\texttt{succ}\exists\mathbb{R}$-hardness of SAT_{prob}. We prove this in Proposition 6.5. To this aim, we introduce the problem $\text{succETR}^{1/8,+,\times}_{[-1/8,1/8]}$ and we prove the $\texttt{succ}\exists\mathbb{R}$-hardness of this problem.

Definition 6.2 (Abrahamsen *et al.*, 2017)**.** In the problem $\text{ETR}^{c,+,\times}$, where $c \in \mathbb{R}$, we are given a set of real variables $\{x_1,\dots,x_n\}$ and a set of equations of the form $x_i = c,\ x_{i_1} + x_{i_2} = x_{i_3},\ x_{i_1}x_{i_2} = x_{i_3}$, for $i, i_1, i_2, i_3 \in [n]$. The goal is to decide whether the system of equations has a solution. The problem, where we also require that $x_1,\dots,x_n \in [a,b]$, for some $a, b \in \mathbb{R}$, is denoted by $\text{ETR}^{c,+,\times}_{[a,b]}$.

Lemma 6.3 (Abrahamsen *et al.*, 2017)**.** $\text{ETR}^{1,+,\times}$ *and* $\text{ETR}^{1/8,+,\times}_{[-1/8,1/8]}$ *are* $\exists\mathbb{R}$*-complete.*

Let $\text{succETR}^{c,+,\times}$ and $\text{succETR}^{c,+,\times}_{[a,b]}$ denote the succinct versions of $\text{ETR}^{c,+,\times}$ and $\text{ETR}^{c,+,\times}_{[a,b]}$, respectively. We assume that their instances are represented as seven Boolean circuits $C_0, C_1,\dots, C_6 : \{0,1\}^M \to \{0,1\}^N$ such that $C_0(j)$ gives the index of the variables 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 an instance has the same number 2^M of equations of each type; If not, one of the equations can be duplicated as many times as needed.

Lemma 6.4. $\text{succETR}^{1/8,+,\times}_{[-1/8,1/8]}$ *is* $\texttt{succ}\exists\mathbb{R}$*-complete.*

To establish the lemma, we first show that $\text{succQUAD} \le_p \text{succETR}^{1,+,\times}$ and then prove that $\text{succETR}^{1,+,\times} \le_p \text{succETR}^{1/8,+,\times}_{[-1/8,1/8]}$ (see [van der Zander *et al.*, 2023]).

Now we are ready to prove the $\texttt{succ}\exists\mathbb{R}$-hardness of SAT_{prob} through the following reduction:

Proposition 6.5. $\text{succETR}^{1/8,+,\times}_{[-1/8,1/8]} \le_p \text{SAT}_{prob}$.

Proof. Let us assume that the instance of $\text{succETR}^{1/8,+,\times}_{[-1/8,1/8]}$ is represented by seven Boolean circuits $C_0, C_1,\dots, C_6 : \{0,1\}^M \to \{0,1\}^N$ as described above. Let the variables of the instance be indexed as $x_{e_1,\dots,e_N}$, with $e_i \in \{0,1\}$ for $i \in [N]$. Below, we often identify the bit sequence $b_1,\dots,b_L$ by an integer j, with $0 \le j \le 2^L - 1$, the binary representation of which is $b_1\dots b_L$ and vice versa.

The instance of the problem $\text{succETR}^{1/8,+,\times}_{[-1/8,1/8]}$ is satisfiable if and only if:

$$\exists x_0,\dots,x_{2^N-1} \in [-1/8, 1/8]$$
$$\sum_{j=0}^{2^M-1} \Big((x_{C_0(j)} - 1/8)^2 + (x_{C_1(j)} + x_{C_2(j)} - x_{C_3(j)})^2 + (x_{C_4(j)} \cdot x_{C_5(j)} - x_{C_6(j)})^2 \Big) = 0. \tag{9}$$

We construct a system of equations in the language $\mathcal{L}_{prob}$ and prove that there exists a model which satisfies the equations if and only if the formula (9) is satisfiable.

Let $X_0, X_1,\dots, X_N$ be binary random variables. We will model each real variable $x_{e_1,\dots,e_N}$ as a term involving the conditional probability $\mathbb{P}(X_0{=}0 \mid X_1{=}e_1,\dots, X_N{=}e_N)$ as follows:

$$q_{e_1,\dots,e_N} := 2/8 \cdot \mathbb{P}(X_0{=}0 \mid X_1{=}e_1,\dots, X_N{=}e_N) - 1/8.$$

This guarantees that $q_{e_1,\dots,e_N} \in [-1/8, 1/8]$. In our construction, the existential quantifiers in the formula (9) correspond to the existence of a probability distribution $P(X_0, X_1,\dots, X_N)$ which determines the values $q_{e_1,\dots,e_N}$. This means that the formula (9) is satisfiable if and only if there exists a model for the equation:

$$\sum_{j=0}^{2^M-1} \Big((q_{C_0(j)} - 1/8)^2 + (q_{C_1(j)} + q_{C_2(j)} - q_{C_3(j)})^2 + (q_{C_4(j)} \cdot q_{C_5(j)} - q_{C_6(j)})^2 \Big) = 0. \tag{10}$$

The challenging task which remains to be solved is to express the terms $q_{C_i(j)}$ in the language $\mathcal{L}_{prob}$. Below we provide a system of equations that achieves this goal.

To model a Boolean formula encoded by a node of C_i, with $i = 0, 1,\dots, 6$, we use arithmetization to go from logical formulas to polynomials over terms involving probabilities of the form $\mathbb{P}(\delta)$. We start with input nodes $v \in \{0,1\}$ and model them as events δ_v such that $\mathbb{P}(\delta_v) = 1$ if and only if $v = 1$. For every 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 define an event δ_v such that $\mathbb{P}(\delta_v) = 1 - \mathbb{P}(\delta_u)$. If v is labeled with $\wedge$ and u and w are children of v, then we specify an event δ_v such that $\mathbb{P}(\delta_v) = \mathbb{P}(\delta_u)\mathbb{P}(\delta_w)$. Finally, if v is labeled with $\vee$ and u and w are children of v, then $\mathbb{P}(\delta_v) = 1 - (1 - \mathbb{P}(\delta_u))(1 - \mathbb{P}(\delta_w))$. Thus, if v is an output node of a circuit C_i, then, for C_i fed with input $j = b_1\dots b_M \in \{0,1\}^M$, we have $v = 1$ if and only if $\mathbb{P}(\delta_v) = 1$. We define the events as follows.

Let $u_{i,1},\dots,u_{i,M}$ be the input nodes of C_i and let, for every node v of C_i, the sequence $\ell_1^v,\dots,\ell_{k_v}^v$ denotes the indices of the leaves $u_{i,\ell_1^v},\dots,u_{i,\ell_{k_v}^v}$ of the sub-circuit with root v. Note that, for an input $u_{i,k}$, the index $\ell_1^{u_{i,k}} = k$ and $k_{u_{i,k}} = 1$. To define the events, for every node v, we introduce a sequence of M new binary random variables $X_{v,1}\dots,X_{v,M}$ and provide an equation that enforces the properties described above. The probabilities used in the equations involve, for a node v, only the variables indexed with $\ell_1^v,\dots,\ell_{k_v}^v$; The remaining variables are irrelevant.

For the kth input node $u_{i,k}$ and the variable $X_{u_{i,k},k}$, we require

$$\mathbb{P}(X_{u_{i,k},k}{=}0) = 0 \text{ and } \mathbb{P}(X_{u_{i,k},k}{=}1) = 1. \tag{11}$$

For every internal node v of C_i, if v is labeled with $\neg$ and u is a child of v, then we define the equation:

$$\sum_{b_1,\dots,b_{k_v}} \big(\mathbb{P}(X_{v,\ell_1^v}{=}b_1,\dots,X_{v,\ell_{k_v}^v}{=}b_{k_v}) - (1-\mathbb{P}(X_{u,\ell_1^v}{=}b_1,\dots,X_{u,\ell_{k_v}^v}{=}b_{k_v}))\big)^2 = 0. \tag{12}$$

If v has label $\wedge$ and w and z are children of v, then we define

$$\sum_{b_1,\dots,b_{k_v}} \big(\mathbb{P}(X_{v,\ell_1^v}{=}b_1,\dots,X_{v,\ell_{k_v}^v}{=}b_{k_v}) - \mathbb{P}(X_{w,\ell_1^w}{=}b_{j_1},\dots,X_{w,\ell_{k_w}^w}{=}b_{j_{k_w}}) \times \mathbb{P}(X_{z,\ell_1^z}{=}b_{j'_1},\dots,X_{z,\ell_{k_z}^z}{=}b_{j'_{k_z}})\big)^2 = 0, \tag{13}$$

where the indices $j_k, j'_{k'} \in \{1,\dots k_v\}$ are defined as $j_k := i$ for $\ell_k^w = \ell_i^v$ and $j'_{k'} := i$ for $\ell_{k'}^z = \ell_i^v$.

If v is labeled with $\vee$, we encode it as a negation of $\wedge$ after negating the children of v.

Let $\hat{u}_{i,1},\dots,\hat{u}_{i,N}$ be the output nodes of C_i. For the kth output $v = \hat{u}_{i,k}$ and for an C_i's input $j = b_1\dots b_M$, we denote by $o_{i,k}(j)$ the expression

$$o_{i,k}(j) := \mathbb{P}(X_{v,\ell_1^v}{=}b_{\ell_1^v},\dots,X_{v,\ell_{k_v}^v}{=}b_{\ell_{k_v}^v})$$

which represents the value of the kth output bit of $C_i(j)$ in such a way that the bit is equal to e_k if and only if the probability $o_{i,k}(j) = e_k$. We illustrate this concept in Example 6.6.

Finally, for $i = 0,1,\dots,6$ and $j = b_1\dots b_M \in \{0,1\}^M$, let

$$\alpha(i,j) := \sum_{e_1,\dots,e_N\in\{0,1\}} q_{e_1,\dots,e_N} \times \prod_{k=1}^{N} (o_{i,k}(j)\mathbb{P}(E{=}e_k) + (1-o_{i,k}(j))(1-\mathbb{P}(E{=}e_k))),$$

where E is a new binary random variable with $\mathbb{P}(E{=}0) = 0$ and $\mathbb{P}(E{=}1) = 1$. The crucial property of this encoding is that: For all $i = 0,1,\dots,6$ and $j = b_1\dots b_M \in \{0,1\}^M$, it is true that $\alpha(i,j) = q_{C_i(j)}$.

Now, we replace every term $q_{C_i(j)}$ in the Eq. (10) by $\alpha(i,j)$ and get the following final equation in the language $\mathcal{L}_{prob}$:

$$\sum_{j=0}^{2^M-1} \Big((\alpha(0,j) - {}^1\!/\!_8)^2 + (\alpha(1,j) + \alpha(2,j) - \alpha(3,j))^2 + (\alpha(4,j)\cdot\alpha(5,j) - \alpha(6,j))^2\Big) = 0. \tag{14}$$

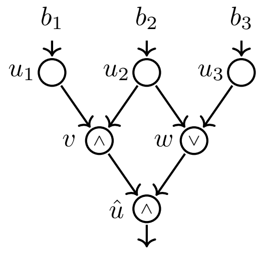

Figure 2: An example Boolean circuit with input nodes u_1, u_2, u_3, internal nodes v, w, and one output node $\hat{u}$.

This completes the proof since it is true that the formula (9) is satisfiable if and only if there exists a model which satisfies Eq. (11)–(13) and Eq. (14). Obviously, the size of the resulting system of equations is polynomial in the size $|C_0| + |C_1| + \dots + |C_6|$ of the input instance and the system can be computed in polynomial time. $\square$

Example 6.6. Consider a Boolean circuit C with three input nodes u_1, u_2, u_3, internal nodes v, w, and one output node $\hat{u}$ as shown in Fig. 2. The relevant variables used to encode the output bit are: $X_{u_1,1}, X_{u_2,2}, X_{u_3,3}$, assigned to the input nodes, as well as $X_{v,1}, X_{v,2}$ and $X_{w,2}, X_{w,3}$ and $X_{\hat{u},1}, X_{\hat{u},2}, X_{\hat{u},3}$, assigned to v, to w, and $\hat{u}$, respectively. Then, assuming the constraints expressed in Eq. (11)–(13) are satisfied, for any $b_1, b_2, b_3 \in \{0,1\}$, the probability $o(j = b_1b_2b_3) := \mathbb{P}(X_{\hat{u},1}{=}b_1, X_{\hat{u},2}{=}b_2, X_{\hat{u},3}{=}b_3)$ can be estimated as follows:

$$\begin{aligned} o(j) &= \mathbb{P}(X_{\hat{u},1}{=}b_1, X_{\hat{u},2}{=}b_2, X_{\hat{u},3}{=}b_3) \\ &= \mathbb{P}(X_{v,1}{=}b_1, X_{v,2}{=}b_2)\cdot\mathbb{P}(X_{w,2}{=}b_2, X_{w,3}{=}b_3) \\ &= \mathbb{P}(X_{u_1,1}{=}b_1)\cdot\mathbb{P}(X_{u_2,2}{=}b_2)\cdot \\ &\qquad \big(1-(1-\mathbb{P}(X_{u_2,2}{=}b_2))\cdot(1-\mathbb{P}(X_{u_3,3}{=}b_3))\big) \\ &= b_1\cdot b_2\cdot(1-(1-b_2)(1-b_3)), \end{aligned}$$

which is 0 if $C(b_1,b_2,b_3) = (b_1 \wedge b_2) \wedge (b_2 \vee b_3) = 0$, and it is equal to 1 if $C(b_1,b_2,b_3) = 1$.

7 Discussion

We have analyzed the complexity of deciding whether a system of (in)equalities involving probabilistic and causal formulas is satisfiable using standard languages, allowing formulas with summations. We have shown that the problems are complete for a new complexity class succ$\exists\mathbb{R}$. Using these results, we could conclude that the complexity of the validity problem, asking whether a Boolean combination of (in)equalities involving pure probabilities and interventional expressions is valid, is complete for the class co-succ$\exists\mathbb{R}$.

Any $\exists\mathbb{R}$-complete problem can probably be turned into a succ$\exists\mathbb{R}$-complete one by encoding the input succinctly. Although we do not know other published succ$\exists\mathbb{R}$-complete problems, the situation is similar to the relation of NP and NEXPTIME. The succinct versions of almost all NP-complete problems are NEXPTIME-complete. We think that succ$\exists\mathbb{R}$ is so natural that further complete problems not coming from $\exists\mathbb{R}$-complete ones will be found in the future.

Acknowledgments

This work was supported by the Deutsche Forschungsgemeinschaft (DFG) grant 471183316 (ZA 1244/1-1).

Contribution Statement

Markus Bläser and Maciej Liśkiewicz are equally last authors.

References

[Abadi and Halpern, 1994] Martin Abadi and Joseph Y Halpern. Decidability and expressiveness for first-order logics of probability. *Information and computation*, 112(1):1–36, 1994.

[Abrahamsen *et al.*, 2017] Mikkel Abrahamsen, Anna Adamaszek, and Tillmann Miltzow. The art gallery problem is $\exists\mathbb{R}$-complete. *arXiv preprint arXiv:1704.06969*, 2017.

[Abrahamsen *et al.*, 2018] Mikkel Abrahamsen, Anna Adamaszek, and Tillmann Miltzow. The art gallery problem is $\exists\mathbb{R}$-complete. In *Proc. of the 50th ACM SIGACT Symposium on Theory of Computing*, pages 65–73, 2018.

[Abrahamsen *et al.*, 2021] Mikkel Abrahamsen, Linda Kleist, and Tillmann Miltzow. Training neural networks is $\exists\mathbb{R}$-complete. *Advances in Neural Information Processing Systems*, 34:18293–18306, 2021.

[Aleksandrowicz *et al.*, 2017] Gadi Aleksandrowicz, Hana Chockler, Joseph Y Halpern, and Alexander Ivrii. The computational complexity of structure-based causality. *Journal of Artificial Intelligence Research*, 58:431–451, 2017.

[Arora and Barak, 2009] Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009.

[Bareinboim *et al.*, 2022] 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.

[Canny, 1988] John Canny. Some algebraic and geometric computations in PSPACE. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 460–467. ACM, 1988.

[Eiter and Lukasiewicz, 2002] Thomas Eiter and Thomas Lukasiewicz. Complexity results for structure-based causality. *Artificial Intelligence*, 142(1):53–89, 2002.

[Elwert, 2013] Felix Elwert. *Graphical Causal Models*, pages 245–273. Handbooks of Sociology and Social Research. Springer, 2013.

[Fagin *et al.*, 1990] Ronald Fagin, Joseph Y Halpern, and Nimrod Megiddo. A logic for reasoning about probabilities. *Information and computation*, 87(1-2):78–128, 1990.

[Garey and Johnson, 1979] Michael Garey and David Johnson. *Computers and intractability: a guide to the theory of NP-completeness*. WH Freeman & Co., San Francisco, 1979.

[Garg *et al.*, 2018] Jugal Garg, Ruta Mehta, Vijay V Vazirani, and Sadra Yazdanbod. $\exists\mathbb{R}$-completeness for decision versions of multi-player (symmetric) nash equilibria. *ACM Transactions on Economics and Computation (TEAC)*, 6(1):1–23, 2018.

[Glymour *et al.*, 2014] Clark Glymour, Richard Scheines, and Peter Spirtes. *Discovering causal structure: Artificial intelligence, philosophy of science, and statistical modeling*. Academic Press, 2014.

[Grigoriev and Vorobjov, 1988] Dima Grigoriev and Nicolai Vorobjov. Solving systems of polynomial inequalities in subexponential time. *J. Symb. Comput.*, 5(1/2):37–64, 1988.

[Halpern, 2000] Joseph Y Halpern. Axiomatizing causal reasoning. *Journal of Artificial Intelligence Research*, 12:317–337, 2000.

[Ibeling and Icard, 2020] Duligur Ibeling and Thomas Icard. Probabilistic reasoning across the causal hierarchy. In *The 34th AAAI Conference on Artificial Intelligence, AAAI 2020*, pages 10170–10177. AAAI Press, 2020.

[Koller and Friedman, 2009] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[Miltzow and Schmiermann, 2022] Tillmann Miltzow and Reinier F Schmiermann. On classifying continuous constraint satisfaction problems. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 781–791. IEEE, 2022.

[Mossé *et al.*, 2022] Milan Mossé, Duligur Ibeling, and Thomas Icard. Is causal reasoning harder than probabilistic reasoning? *The Review of Symbolic Logic*, pages 1 – 24, 2022.

[Papadimitriou, 1994] Christos Papadimitriou. *Computational Complexity*. Addison Welsey, 1994.

[Pearl and Mackenzie, 2018] Judea Pearl and Dana Mackenzie. *The book of why: the new science of cause and effect*. Basic books, 2018.

[Pearl, 1995] Judea Pearl. Causal diagrams for empirical research. *Biometrika*, 82(4):669–688, 1995.

[Pearl, 2009] Judea Pearl. *Causality*. Cambridge University Press, 2009.

[Schaefer and Štefankovič, 2017] Marcus Schaefer and Daniel Štefankovič. Fixed points, Nash equilibria, and the existential theory of the reals. *Theory Comput. Syst.*, 60(2):172–193, 2017.

[Schaefer, 2009] Marcus Schaefer. Complexity of some geometric and topological problems. In *International Symposium on Graph Drawing*, pages 334–344. Springer, 2009.

[Shpitser and Pearl, 2006] Ilya Shpitser and Judea Pearl. Identification of conditional interventional distributions. In *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence*, pages 437–444. AUAI Press, 2006.

[Shpitser and Pearl, 2008] Ilya Shpitser and Judea Pearl. Complete identification methods for the causal hierarchy. *Journal of Machine Learning Research*, 9(Sep):1941–1979, 2008.

[Speranski, 2013] Stanislav O Speranski. Complexity for probability logic with quantifiers over propositions. *Journal of Logic and Computation*, 23(5):1035–1055, 2013.

[van der Zander *et al.*, 2023] Benito van der Zander, Markus Bläser, and Maciej Liśkiewicz. The hardness of reasoning about probabilities and causality. *arXiv preprint arXiv:2305.09508*, 2023.