Binding Forms in First-Order Logic

Fabio Mogavero[*1,2] **and Giuseppe Perelli**[†1]

1 University of Oxford, U.K.
2 Università degli Studi di Napoli Federico II, Italy

Abstract

Aiming to pinpoint the reasons behind the decidability of some complex extensions of *modal logic*, we propose a new *classification criterion* for sentences of *first-order logic*, which is based on the kind of *binding forms* admitted in their expressions, *i.e.*, on the way the arguments of a relation can be bound to a variable. In particular, we describe a hierarchy of four fragments focused on the Boolean combinations of these forms, showing that the less expressive one is already incomparable with several first-order restrictions proposed in the literature, as the *guarded* and *unary negation* fragments. We also prove, via a novel model-theoretic technique, that our logic enjoys the finite-model property, Craig's interpolation, and Beth's definability. Furthermore, the associated model-checking and satisfiability problems are solvable in PTime and Σ_3^{P}, respectively.

1998 ACM Subject Classification F.4.1 Mathematical Logic

Keywords and phrases First-Order Logic, Decidable Fragments, Satisfiability, Model Checking

Digital Object Identifier 10.4230/LIPIcs.CSL.2015.648

1 Introduction

Since from the revolutionary negative solutions owed to Church and Turing, Hilbert's original "*Entscheidungsproblem*" [27] turned into a vast *classification process* looking for all those classes of *first-order sentences* having a *decidable satisfiability* [10]. Depending upon the syntactic criteria used to identify the particular classes of interest [23], this process was declined into several research programs, among which we can mention, on one side, those limiting *relation arities* [35] or the total *number of variables* [40, 24] and, on the other one, those classifying sentences in *prenex normal form* based on their *prefix vocabulary* [11].

The research in this field has had a considerable impact, from both a theoretical and practical point of view, in a variety of areas on the edge between mathematics and computer science, *e.g.*, *reverse mathematics* [48], *descriptive complexity* [33], *database theory* [51, 1], and *formal verification*, just to mention a few. However, Vardi observed that almost all of the classic approaches did not shed any satisfactory light on why *modal logic* and derived frameworks, like the ones featuring fixpoint constructs, are so robustly decidable [57, 21].

Trying to find a plausible answer, Andréka, van Benthem, and Németi introduced the *guarded fragment* of first-order logic [3], which generalizes the modal framework by essentially retaining several of its model-theoretic and algorithmic properties. This work started a completely new research program based on the way quantifications can be *relativised* to atoms, avoiding the usual syntactic restrictions on quantifier patterns, number of variables, and relation arities. Pushing forward the idea that robust fragments of first-order logic

* Research supported by the INdAM-GNCS Funding "Giovani Ricercatori 2014".
† Work partially done when the author was a Ph.D. student at the Università degli Studi di Napoli Federico II. Research partially supported by the ERC Advanced Grant RACE (291528) at Oxford.

24th EACSL Annual Conference on Computer Science Logic (CSL 2015).
Editor: Stephan Kreutzer; pp. 648–665

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

owe their nice properties to some sort of guarded quantification, several extensions along this line of research were proposed in the literature, such as the *loosely guarded* [52], the *clique guarded* [19, 22], the *action guarded* [53, 18], and the *guarded fixpoint logic* [25]. This classification program has also important applications in database theory and description logic, where it is relevant to evaluate a query against guarded first-order theories [5].

Only recently, ten Cate and Segoufin observed that the first-order translation of modal logic presents, besides the guarded nature of quantifications, another important peculiarity: negation is only applied to sentences or monadic formulas, *i.e.*, formulas with a single free variable. Exploiting this observation, they introduced a new robust fragment of first-order logic, called *unary negation* [49, 50], which extends modal logic, as well as other formalisms, like Boolean conjunctive queries, that cannot be expressed in terms of guarded quantifications. Since this new restriction is incomparable with the guarded fragments, right after the original work, another formalism was proposed, called *guarded negation* [7], which unifies the two approaches. Syntactically, there is no primary universal quantifier and the use of negation is only allowed if guarded by an atom. In terms of expressive power, this fragment forms a strict extension of both the logics on which it is based, while preserving the same desirable properties of the modal framework. However, it has to be noted that it is still incomparable with more complex extensions of the guarded fragment, such as the clique guarded one. This way of analysing formulas focusing on the guarded nature of negation has also important applications to database theory, where it is well-known that the operation of complementation makes queries hard to evaluate [6].

Although these two innovative classification programs really succeeded in the original task to explain the nice properties enjoyed by modal logic, we cannot consider them completely satisfactory with respect to the more general intent of identifying the reasons why some of its complex extensions are so well-behaved. In particular, based only on the resulting model-theoretic and algorithmic features, we are not able to answer the question about the decidability of several multi-agent logics for strategic abilities, such as the *Alternating Temporal Logics* [2] ATL [58, 47] and ATL* [46] and the *one-goal fragment* SL[1G] [37] of *Strategy Logic* [12, 39, 38], which do not intrinsically embed such kinds of relativisation. For example, consider the ATL* formula $[\![\mathtt{a}, \mathtt{b}, \mathtt{c}]\!]\neg\psi$ over a game structure with $\mathtt{a}$, $\mathtt{b}$, $\mathtt{c}$, and $\mathtt{d}$ as the only agents. Intuitively, it asserts that agent $\mathtt{d}$ has a strategy, which depends upon those chosen by the other ones, ensuring that the LTL property ψ does not hold. Now, observe that the underlying strategic reasoning can be represented by the first-order sentence $\forall\mathtt{a}\forall\mathtt{b}\forall\mathtt{c}\exists\mathtt{d}\neg\mathtt{r}_\psi(\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d})$ having a prefix of the form $\forall^3\exists$ coupled with the quaternary atomic relation $\mathtt{r}_\psi$ in place of the temporal requirement ψ, which absorbs and hides the intrinsic second-order flavour of the original ATL* formula. It is evident that this sentence belongs neither to a decidable prefix-vocabulary class nor to the two-variable fragment. Moreover, quantifications are not guarded and negation is applied to a formula that is neither monadic nor guarded. Another explicit example is given by the SL[1G] sentence $[\![\mathtt{x}]\!]\langle\!\langle\mathtt{y}\rangle\!\rangle[\![\mathtt{z}]\!](\mathtt{a}, \mathtt{x})(\mathtt{b}, \mathtt{x})(\mathtt{c}, \mathtt{y})(\mathtt{d}, \mathtt{z})\psi$ asserting that, once $\mathtt{a}$ and $\mathtt{b}$ have chosen the common strategy $\mathtt{x}$, agent $\mathtt{c}$ can select its better response $\mathtt{y}$ to ensure ψ, in a way that is independent of the behaviour $\mathtt{z}$ of $\mathtt{d}$. In this case, the associated first-order sentence $\forall\mathtt{ab}\exists\mathtt{c}\forall\mathtt{d}\,\mathtt{r}_\psi(\mathtt{ab}, \mathtt{ab}, \mathtt{c}, \mathtt{d})$ has a prefix of the form $\forall\exists\forall$ coupled with the atomic relation $\mathtt{r}_\psi$, whose first two arguments are bound to the same variable. Again, we cannot cast this sentence in any of the decidable restrictions previously described. In particular, it is neither unary negation nor guarded negation, since universal quantifications are used as primary construct, which is not allowed in either of them.

Table 1 Notable algorithmic and model-theoretic properties for some fragments of FOL (M: *monadic*, 2VAR: 2-*variables*, GF: *guarded fragment*, CG: *clique guarded*, UN: *unary negation*, GN: *guarded negation*, FL: *fluted logic*, UF_1: *uniform one-dimensional fragment*, 1B: *one binding*, CB: *conjunctive binding*, DB: *disjunctive binding*, BB: *Boolean binding*).

	MC	SAT	FMP	CI	BD
FOL[M]	PSPACE-C	NEXPTIME-C [11]	✓ [11]	✓	✓
FOL[2VAR]	PTIME [56]	NEXPTIME-C [24]	✓ [40]	× [45]	× [45]
FOL[GF]	PTIME-C [9]	2EXPTIME-C [20]	✓ [20]	× [29]	✓ [29]
FOL[CG]	PSPACE-C [9]	2EXPTIME-C [19]	✓ [28]	× [29]	✓ [29]
FOL[UN]	$\Delta_2^{\text{P}}(\text{O}(\log^2 n))$-C [50]	2EXPTIME-C [50]	✓ [50]	✓ [50]	✓ [50]
FOL[GN]	$\Delta_2^{\text{P}}(\text{O}(\log^2 n))$-C [7]	2EXPTIME-C [7]	✓ [7]	✓ [4]	✓ [4]
FOL[FL]	PSPACE	NEXPTIME-C [42]	✓ [42]	✓ [42]	✓ [42]
FOL[UF_1]	?	NEXPTIME-C [34]	✓ [26]	?	?
FOL[1B]	PTIME [Thm.3.12]	Σ_3^{P}-C [Thm.3.13]	✓ [Thm.3.11]	✓ [Thm.3.11]	✓ [Thm.3.11]
FOL[CB]	PSPACE-C [Thm.3.8]	? [Con.3.10]	? [Con.3.10]	?	?
FOL[DB]	PSPACE-C [Thm.3.8]	Undecidable [Thm.3.9]	× [Thm.3.9]	?	?
FOL[BB]	PSPACE-C [Cor.3.6]	Undecidable [Cor.3.6]	× [Cor.3.6]	✓ [Cor.3.6]	✓ [Cor.3.6]

At this point, a question naturally arises: what are the *syntactic constraints* on the first-order representations of these logics of strategies that ensure their decidability? After a careful analysis, one can observe that such representations are always composed by a Boolean combination of sentences in prenex normal form, whose matrices are Boolean combinations of relations over the same arguments, which denote the agents of the game under analysis.

In this paper, trying to lay the foundation for a more thorough understanding of these decidability questions, we exploit the above observation to devise a new classification program based on the *binding forms* admitted in a sentence, *i.e.*, on the way the arguments of a relation can be bound to a variable. Indeed, inspired by the decoupling between agents and variables in SL, we define a syntactic variant of first-order logic in which arguments are bound to variables by means of an appropriate *binding construct*. To support this, similarly to the treatment of the attributes of a table in database theory [16], we describe a generalization of standard notions of *language signature* and *relational structure* in which arguments are explicit. With more detail, every relation r is associated with a set of arguments $\{a_1, \ldots, a_n\}$, which are bound to the variables via a binding form $(a_1, x_1) \cdots (a_n, x_n)r$ that replaces the standard writing $r(x_1, \ldots, x_n)$. So, for instance, a formula like $\mathtt{r}_1(\mathtt{x}_1, \mathtt{x}_2) \wedge \mathtt{r}_2(\mathtt{x}_2, \mathtt{x}_3) \rightarrow \mathtt{r}_3(\mathtt{x}_1, \mathtt{x}_3)$ would be written as $(\mathtt{a}_1, \mathtt{x}_1)(\mathtt{a}_2, \mathtt{x}_2)(\mathtt{a}_3, \mathtt{x}_3)(\mathtt{r}_1 \wedge \mathtt{r}_2 \rightarrow \mathtt{r}_3)$, assuming $\{\mathtt{a}_1, \mathtt{a}_2\}$, $\{\mathtt{a}_2, \mathtt{a}_3\}$, and $\{\mathtt{a}_1, \mathtt{a}_3\}$ as the arguments of $\mathtt{r}_1$, $\mathtt{r}_2$, and $\mathtt{r}_3$, respectively. Our notation, although perfectly equivalent to the classic one, allows to introduce and analyse, in a natural way, a hierarchy of four fragments of first-order logic based on the Boolean combinations of these forms. In particular, the simplest one, called *one binding*, is already incomparable with the clique guarded and guarded negation restrictions, as well as, with the *fluted logic* introduced by Quine [44] and the *uniform one-dimensional fragment* recently proposed by Hella, Kieronski, and Kuusisto [26, 34]. Examples of one-binding sentences result from the translation of the game properties described above, namely $\forall\mathtt{x}\forall\mathtt{y}\forall\mathtt{z}\exists\mathtt{w}(\mathtt{a}, \mathtt{x})(\mathtt{b}, \mathtt{y})(\mathtt{c}, \mathtt{z})(\mathtt{d}, \mathtt{w})\neg\mathtt{r}_\psi$ and $\forall\mathtt{x}\exists\mathtt{y}\forall\mathtt{z}(\mathtt{a}, \mathtt{x})(\mathtt{b}, \mathtt{x})(\mathtt{c}, \mathtt{y})(\mathtt{d}, \mathtt{z})\mathtt{r}_\psi$, where we assume that the relation $\mathtt{r}_\psi$ has the agents $\mathtt{a}$, $\mathtt{b}$, $\mathtt{c}$, and $\mathtt{d}$ as arguments. Via a novel model-theoretic technique exploiting the peculiarity of binding forms, we prove that our logic enjoys the *finite-model property*, both *Craig's interpolation* and *Beth's definability*, a PTIME *model checking* and a Σ_3^{P}-COMPLETE *satisfiability*.

In Table 1, we summarize results and open problems about classic and new fragments, from which we can immediately observe that the one-binding restriction is the simplest one with respect to the algorithmic point of view.

The main aim of this work is to describe a new criterion to classify formulas in order to better explore the boundary between *nice/non-nice* [3], *tame/untamed* [42], *easy/hard* [56], and *decidable/undecidable* [11] fragments of FOL, which is probably quite wider, but hopefully less irregular, than it was considered before. In particular, thanks to the introduced syntax, it is easier to discover connections with other languages, as those derived from *algebras* [13] and *calculi* [14] for relational databases. Moreover, we think it can help in the quest for the ultimate reasons behind tractability and decidability of a logic. Last but not least, we provide model-theoretic results that may be used as technical tools in other contexts, as well. Indeed, going back to the logics for strategic abilities, we claim that the *bounded-tree model property* of ATL [47], ATL* [46], and SL[1G] [37], shown to be crucial for their decidability, can be proved by means of the finite-model property of the one-binding fragment. This result is the focus of a dedicated work [36].

2 Signatures and Structures

Since from Codd's pioneering work on the definition of *relational databases* [13], several kinds of first-order languages have been used to describe databases queries [14]. In particular, *first-order logic* (FOL, for short) has been established as the main theoretical framework in which to prove results about properties of query languages [31, 55, 32]. In such a context, a table is usually represented as a mathematical relation between elements of a given domain, where its attributes are mapped to the indexes of that relation in a predetermined fixed way. Hence, attributes do not have any explicit matching element in the syntax of the language.

To introduce the *binding-form fragments* of FOL, we need to reformulate, instead, both the syntax and semantics of the logic in a way that is much closer to database theory. In particular, we explicitly associate a finite non-empty set of arguments to each relation [16], which are handled in the syntax via corresponding symbols. To do this, in the following, we introduce an alternative version of classic *language signatures* and *relational structures*.

Language Signatures. A *language signature* is a mathematical object describing the form of all non-logical symbols composing a formula. The typology we introduce here is purely relational, since we do not make use of constant or function symbols. Also, in our reasonings, we do not explicitly consider distinguished relations as equivalences, orders, or the equality.

▶ **Definition 2.1** (Language Signature). A *language signature* (LS, for short) is a tuple $\mathcal{L} \triangleq \langle \mathrm{Ar}, \mathrm{Rl}, \mathsf{ar} \rangle$, where Ar and Rl are the finite non-empty sets of *argument* and *relation* symbols and $\mathsf{ar} : \mathrm{Rl} \to 2^{\mathrm{Ar}} \setminus \{\emptyset\}$ is the *argument function* mapping every relation $r \in \mathrm{Rl}$ to its non-empty set of arguments $r^{\mathcal{L}} \triangleq \mathsf{ar}(r) \subseteq \mathrm{Ar}$.

Suppose we want to describe the schema of a genealogy database containing the relations *isFather*(*father*, *child*), *isMother*(*mother*, *child*), and *areParents*(*father*, *mother*, *child*). We can do this by means of the simple LS $\mathcal{L}_G = \langle \mathrm{Ar}, \mathrm{Rl}, \mathsf{ar} \rangle$, whose elements are set as follows: $\mathrm{Ar} = \{\mathtt{c} : \mathit{child}, \mathtt{f} : \mathit{father}, \mathtt{m} : \mathit{mother}\}$; $\mathrm{Rl} = \{\mathtt{Ft} : \mathit{isFather}, \mathtt{Mt} : \mathit{isMother}, \mathtt{Pr} : \mathit{areParents}\}$; $\mathtt{Ft}^{\mathcal{L}_G} = \{\mathtt{c}, \mathtt{f}\}$, $\mathtt{Mt}^{\mathcal{L}_G} = \{\mathtt{c}, \mathtt{m}\}$, and $\mathtt{Pr}^{\mathcal{L}_G} = \mathrm{Ar}$. From now on, we may put a superscript on a relation containing a list of its argument, *e.g.*, $\mathtt{Pr}^{\mathtt{cfm}}$, to help the reader to keep track of them.

Relational Structures. Given a language signature, we define the interpretation of all symbols by means of a *relational structure*, *i.e.*, a carrier domain together with an association

of each relation with a set of suitable tuples assuming values in that domain. Since relations with the same arity may have different arguments, it is not sufficient to work with elements of a Cartesian product as components of their interpretation. Therefore, we assign to every relation a set of *tuple functions* mapping arguments in their support to values of the carrier domain. Note that, under this definition, an order among arguments is not required.

▶ **Definition 2.2** (Relational Structure). A *relational structure* over an LS $\mathcal{L} = \langle \mathrm{Ar}, \mathrm{Rl}, \mathsf{ar} \rangle$ ($\mathcal{L}$-RS, for short) is a tuple $\mathcal{R} \triangleq \langle \mathrm{Dm}, \mathsf{rl} \rangle$, where Dm is the non-empty set of arbitrary objects named *domain* and $\mathsf{rl} : \mathrm{Rl} \to_r 2^{\mathsf{ar}(r) \to \mathrm{Dm}}$ is the *relation function* mapping every relation $r \in \mathrm{Rl}$ to the set $r^{\mathcal{R}} \triangleq \mathsf{rl}(r) \subseteq \mathsf{ar}(r) \to \mathrm{Dm}$ of *tuple functions* $\mathtt{t} \in \mathsf{rl}(r)$ from the arguments $a \in \mathsf{ar}(r)$ of r to values $\mathtt{t}(a) \in \mathrm{Dm}$ of the domain.

The *order* (*resp.*, *size*) of an $\mathcal{L}$-RS $\mathcal{R}$ is given by the cardinality $|\mathcal{R}| \triangleq |\mathrm{Dm}|$ (*resp.*, $\|\mathcal{R}\| \triangleq |\bigcup_{r \in \mathrm{Rl}} r^{\mathcal{R}}|$) of its domain set (*resp.*, relation function). A relational structure is *finite* if it has finite order and, so, a finite size.

Consider the LS $\mathcal{L}_G$ previously described. In Figure 1, we depict an $\mathcal{L}_G$-RS $\mathcal{R}_G$ containing part of Simpson Family's genealogy tree, where the two binary relations $\mathtt{Ft}^{\mathtt{cf}}$ and $\mathtt{Mt}^{\mathtt{cm}}$ are indicated through the edges labelled by their own names, while the ternary relation $\mathtt{Pr}^{\mathtt{cfm}}$ is described via the three hyperedges represented by the gray areas.

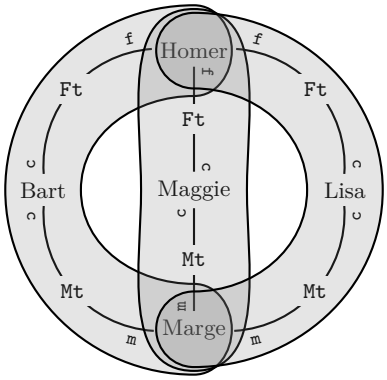

Figure 1 A relational structure.

3 First-Order Logic

We start by describing a slightly different but equivalent formalization of both syntax and semantics of FOL according to the explained alternatives of the language signature and relational structure. Then, we introduce a new family of fragments based on the kinds of binding forms allowed in a formula, *i.e.*, on the ways arguments can be bound to variables.

From now on, unless stated otherwise, we use $\mathcal{L} = \langle \mathrm{Ar}, \mathrm{Rl}, \mathsf{ar} \rangle$ to denote an *a priori* fixed LS. Also, Vr represents an enumerable non-empty set of *variables*. For the sake of succinctness, to indicate the extension of $\mathcal{L}$ with Vr, we adopt the composed symbol $\mathcal{L}(\mathrm{Vr})$.

Syntax. As far as the syntax of FOL is concerned, the novelty of our setting resides in the decoupling between variables and arguments, which implies an explicit occurrence of the latter as atomic components of a formula. Indeed, a variables x is not directly applied to the index associated with an argument a of a relation r, as in the usual writing $r(\ldots, x, \ldots)$, but an appropriate construct $(a, x)\varphi$, called *binding*, is required to bind a to x in the formula φ.

▶ **Definition 3.1** (FOL Syntax). FOL *formulas* over $\mathcal{L}(\mathrm{Vr})$ are built by means of the following context-free grammar, where $a \in \mathrm{Ar}$, $r \in \mathrm{Rl}$, and $x \in \mathrm{Vr}$:

$$\varphi := \bot \mid \top \mid r \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid \exists x.\varphi \mid \forall x.\varphi \mid (a, x)\varphi.$$

$\mathcal{L}(\mathrm{Vr})$-FOL denotes the set of all formulas over $\mathcal{L}(\mathrm{Vr})$ generated by the above grammar.

Consider again the LS $\mathcal{L}_G$ of Section 2 and suppose we want to formalize the fact that father and mother of a person are her parents and *vice versa*. This can be done via the $\mathcal{L}_G(\{\mathtt{x}, \mathtt{y}, \mathtt{z}\})$-FOL formula $\varphi_1 = \forall\mathtt{x}\forall\mathtt{y}\forall\mathtt{z}(\mathtt{f}, \mathtt{x})(\mathtt{m}, \mathtt{y})(\mathtt{c}, \mathtt{z})((\mathtt{Ft}^{\mathtt{cf}} \wedge \mathtt{Mt}^{\mathtt{cm}}) \leftrightarrow \mathtt{Pr}^{\mathtt{cfm}})$, where by $\varphi_1 \leftrightarrow \varphi_2$ we denote, as usual, the conjunction $(\varphi_1 \to \varphi_2) \wedge (\varphi_2 \to \varphi_1)$ with $\varphi_i \to \varphi_j$ in

place of $\neg\varphi_i \vee \varphi_j$. Now, imagine we need to determine which people have a father. To this aim, we can employ the formula $\varphi_2 = \exists \mathtt{x}(\mathtt{f},\mathtt{x})\mathtt{Ft}^{\mathtt{cf}}$, whose argument $\mathtt{c}$ is associated with the result. Finally, to query all pairs $(\mathtt{x},\mathtt{y})$ of grandfather and grandchild, we can use the formula $\varphi_3 = \exists \mathtt{z}((\mathtt{f},\mathtt{x})(\mathtt{c},\mathtt{z})\mathtt{Ft}^{\mathtt{cf}} \wedge (\mathtt{c},\mathtt{y})((\mathtt{f},\mathtt{z})\mathtt{Ft}^{\mathtt{cf}} \vee (\mathtt{m},\mathtt{z})\mathtt{Mt}^{\mathtt{cm}}))$. By fixing a linear order on the arguments, it is possible to rewrite every statement in the classic syntax. For instance, φ_1 can be expressed by $\forall \mathtt{x}\forall \mathtt{y}\forall \mathtt{z}((\mathtt{Ft}(\mathtt{x},\mathtt{z}) \wedge \mathtt{Mt}(\mathtt{y},\mathtt{z})) \leftrightarrow \mathtt{Pr}(\mathtt{x},\mathtt{y},\mathtt{z}))$, once it is assumed that $\mathtt{f} < \mathtt{m} < \mathtt{c}$. Conversely, every formula in the standard syntax can be translated into our syntax, by means of numeric arguments representing the positions in the relations. For example, the transitivity property $\forall \mathtt{x}\forall \mathtt{y}\forall \mathtt{z}((\mathtt{R}(\mathtt{x},\mathtt{y}) \wedge \mathtt{R}(\mathtt{y},\mathtt{z})) \rightarrow \mathtt{R}(\mathtt{x},\mathtt{z}))$ can be rewritten as $\forall \mathtt{x}\forall \mathtt{y}\forall \mathtt{z}(((1,\mathtt{x})(2,\mathtt{y})\mathtt{R}^{12} \wedge (1,\mathtt{y})(2,\mathtt{z})\mathtt{R}^{12}) \rightarrow (1,\mathtt{x})(2,\mathtt{z})\mathtt{R}^{12})$.

Usually, predicative logics, *i.e.*, languages having explicit quantifiers, need a concept of free or bound *placeholder* to formally evaluate the meaning of their formulas. The placeholders are used, in fact, to identify particular positions in a syntactic expression that are crucial for the definition of its semantics. Classic formalizations of FOL just require one kind of placeholder represented by the variables on which the formulas are built. In our new setting, instead, also the arguments have this fundamental role, as they are used to decouple variables from their association with a relation. As a consequence, we need a way to check whether a variable is quantified or an argument is bound. To do this, for every formula φ, we compute its set of *free arguments/variables* $\mathsf{free}(\varphi)$, as the subset of $\mathrm{Ar} \cup \mathrm{Vr}$ containing all arguments that are free from binding together with all variables occurring in some binding that are not quantified. Formally, we have the following definition.

▶ **Definition 3.2** (Free Placeholders). The set of *free arguments/variables* of an $\mathcal{L}(\mathrm{Vr})$-FOL formula can be computed via the function $\mathsf{free} : \mathcal{L}(\mathrm{Vr})\text{-FOL} \to 2^{\mathrm{Ar}\cup\mathrm{Vr}}$ defined as follows:

1. $\mathsf{free}(\bot) = \mathsf{free}(\top) \triangleq \emptyset$;
2. $\mathsf{free}(r) \triangleq \mathsf{ar}(r)$, where $r \in \mathrm{Rl}$;
3. $\mathsf{free}(\neg\varphi) \triangleq \mathsf{free}(\varphi)$;
4. $\mathsf{free}(\varphi_1 \mathtt{Op} \varphi_2) \triangleq \mathsf{free}(\varphi_1) \cup \mathsf{free}(\varphi_2)$, where $\mathtt{Op} \in \{\wedge, \vee\}$;
5. $\mathsf{free}(\mathtt{Qn}x.\varphi) \triangleq \mathsf{free}(\varphi) \setminus \{x\}$, where $\mathtt{Qn} \in \{\exists, \forall\}$;
6. $\mathsf{free}((a,x)\varphi) \triangleq \begin{cases} (\mathsf{free}(\varphi) \setminus \{a\}) \cup \{x\}, & \text{if } a \in \mathsf{free}(\varphi); \\ \mathsf{free}(\varphi), & \text{otherwise.} \end{cases}$

A formula φ without free arguments (*resp.*, variables), *i.e.*, $\mathsf{ar}(\varphi) \triangleq \mathsf{free}(\varphi) \cap \mathrm{Ar} = \emptyset$ (*resp.*, $\mathsf{vr}(\varphi) \triangleq \mathsf{free}(\varphi) \cap \mathrm{Vr} = \emptyset$), is named *argument* (*resp.*, *variable*) *closed*. If φ is both argument and variable closed, it is referred to as a *sentence*. Consider the three formulas φ_1, φ_2, and φ_3 given above. We have that $\mathsf{free}(\varphi_1) = \emptyset$, $\mathsf{free}(\varphi_2) = \mathsf{ar}(\varphi_2) = \{\mathtt{c}\}$, and $\mathsf{free}(\varphi_3) = \mathsf{vr}(\varphi_3) = \{\mathtt{x},\mathtt{y}\}$. Thus, φ_1 is a sentence, φ_2 is variable closed, and φ_3 is argument closed.

One may observe that the proposed syntax has some similarities with the relational calculus introduced by Codd [14, 16] as a logic counterpart of standard relational algebra [13], since in this language the attributes may be identified by name rather than position. However, its notation is tuple-centric, thus, it necessarily requires the use of the equality relation even to express very simple properties that do not intrinsically need it. For example, to describe in that calculus the left-totality of a binary relation $\mathtt{r}$ over the arguments $\mathtt{a}$ and $\mathtt{b}$, we have to write $\forall \mathtt{t}\exists \mathtt{t}'(\mathtt{t}[\mathtt{b}] = \mathtt{t}'[\mathtt{a}] \wedge \mathtt{r}(\mathtt{t}'))$ (unrestricted semantics) or $\forall \mathtt{t}(\mathtt{r}(\mathtt{t}) \rightarrow \exists \mathtt{t}'(\mathtt{t}[\mathtt{b}] = \mathtt{t}'[\mathtt{a}] \wedge \mathtt{r}(\mathtt{t}')))$ (active-domain semantics). In our syntax, instead, we just write $\forall \mathtt{x}\exists \mathtt{y}(\mathtt{a},\mathtt{x})(\mathtt{b},\mathtt{y})\mathtt{r}$ (unrestricted semantics) or $\forall \mathtt{z}\forall \mathtt{x}((\mathtt{a},\mathtt{z})(\mathtt{b},\mathtt{x})\mathtt{r} \rightarrow \exists \mathtt{y}(\mathtt{a},\mathtt{x})(\mathtt{b},\mathtt{y})\mathtt{r})$ (active-domain semantics).

Semantics. The semantics of FOL described here is defined, as usual, *w.r.t.* an RS. As a matter of fact, the peculiarities of our setting only concern the interpretation of binding constructs and the non-standard evaluation of relations.

In order to formalize the meaning of a formula, we first need to describe the concept of *assignment*, *i.e.*, a partial function $\chi \in \mathrm{Asg}_{\mathrm{D}} \triangleq (\mathrm{Ar} \cup \mathrm{Vr}) \rightharpoonup \mathrm{D}$ mapping each placeholder in its domain to a value of an arbitrary set D, which is used to define a valuation of all the free arguments and variables. For a given placeholder $p \in \mathrm{Ar} \cup \mathrm{Vr}$ and a value $d \in \mathrm{D}$, the notation $\chi[p \mapsto d]$ represents the assignment defined on $\mathsf{dom}(\chi[p \mapsto d]) \triangleq \mathsf{dom}(\chi) \cup \{p\}$ that returns d on p and is equal to χ on the remaining part of its domain, *i.e.*, $\chi[p \mapsto d](p) \triangleq d$ and $\chi[p \mapsto d](p') \triangleq \chi(p')$, for all $p' \in \mathsf{dom}(\chi) \setminus \{p\}$.

▶ **Definition 3.3** (FOL Semantics)**.** Let $\mathcal{R}$ be an $\mathcal{L}$-RS and φ an $\mathcal{L}(\mathrm{Vr})$-FOL formula. Then, for all assignments $\chi \in \mathrm{Asg}_{\mathrm{Dm}}$ with $\mathsf{free}(\varphi) \subseteq \mathsf{dom}(\chi)$, the relation $\mathcal{R}, \chi \models \varphi$ is inductively defined on the structure of φ as follows.

1. Boolean values and connectives are interpreted as usual.
2. $\mathcal{R}, \chi \models r$ if $\chi_{\restriction r^{\mathcal{L}}} \in r^{\mathcal{R}}$, for every relation $r \in \mathrm{Rl}$.
3. For each variable $x \in \mathrm{Vr}$, it is set that:
   a. $\mathcal{R}, \chi \models \exists x.\varphi$ if there exists a value $d \in \mathrm{Dm}$ such that $\mathcal{R}, \chi[x \mapsto d] \models \varphi$;
   b. $\mathcal{R}, \chi \models \forall x.\varphi$ if, for all values $d \in \mathrm{Dm}$, it holds that $\mathcal{R}, \chi[x \mapsto d] \models \varphi$.
4. $\mathcal{R}, \chi \models (a, x)\varphi$ if $\mathcal{R}, \chi[a \mapsto \chi(x)] \models \varphi$, for each argument $a \in \mathrm{Ar}$ and variable $x \in \mathrm{Vr}$.

Intuitively, Condition 2 states that a relation r is satisfied by an assignment χ whenever the tuple function $\chi_{\restriction r^{\mathcal{L}}}$ obtained by the restriction of χ to the arguments $r^{\mathcal{L}}$ of r is included in the interpretation $r^{\mathcal{R}}$. Condition 4, instead, interprets the binding construct (a, x) by associating the argument a with the value of the variable x contained inside the assignment.

Consider again the formulas φ_1, φ_2, and φ_3 and the $\mathcal{L}_G$-RS $\mathcal{R}_G$ of Figure 1. We have that $\mathcal{R}_G, \varnothing \models \varphi_1$. Moreover, $\mathcal{R}_G, \varnothing[\mathtt{c} \mapsto \mathrm{Lisa}] \models \varphi_2$ and $\mathcal{R}_G, \varnothing[\mathtt{x} \mapsto \mathrm{Homer}, \mathtt{y} \mapsto \mathrm{Bart}] \not\models \varphi_3$.

To complete the description of the semantics, we give the notions of *model* and *satisfiability*. For an $\mathcal{L}$-RS $\mathcal{R}$ and an $\mathcal{L}(\mathrm{Vr})$-FOL sentence φ, we say that $\mathcal{R}$ is a *model* of φ, in symbols $\mathcal{R} \models \varphi$, iff $\mathcal{R}, \varnothing \models \varphi$, where $\varnothing \in \mathrm{Asg}_{\mathrm{Dm}}$ simply denotes the empty assignment. We also say that φ is *satisfiable* iff there exists a model for it. Given two $\mathcal{L}(\mathrm{Vr})$-FOL formulas φ_1 and φ_2, we say that φ_1 *implies* φ_2, in symbols $\varphi_1 \Rightarrow \varphi_2$, iff $\mathcal{R}, \chi \models \varphi_1$ implies $\mathcal{R}, \chi \models \varphi_2$, for each $\mathcal{L}$-RS $\mathcal{R}$ and assignment $\chi \in \mathrm{Asg}_{\mathrm{Dm}}$ with $\mathsf{free}(\varphi_1), \mathsf{free}(\varphi_2) \subseteq \mathsf{dom}(\chi)$. Moreover, we say that φ_1 is *equivalent* to φ_2, in symbols $\varphi_1 \equiv \varphi_2$, iff both $\varphi_1 \Rightarrow \varphi_2$ and $\varphi_2 \Rightarrow \varphi_1$ hold.

Fragments. We now introduce a family of syntactic fragments of FOL by means of a special *normal form*, where relations over the same set of arguments may be clustered together by a unique sequence of bindings. With more detail, we consider Boolean combinations of sentences in prenex normal form in which quantification prefixes are coupled with Boolean combinations of these relation clusters called *binding forms*. Each fragment is then characterized by a specific constraint on the possible combinations of these forms.

A *quantification prefix* $\wp \in \mathrm{Qn} \subseteq \{\exists x, \forall x : x \in \mathrm{Vr}\}^*$ is a finite sequence of quantifiers, in which each variable occurs at most once. Similarly, a *binding prefix* $\flat \in \mathrm{Bn} \subseteq \{(a, x) : a \in \mathrm{Ar} \wedge x \in \mathrm{Vr}\}^*$ is a finite sequence of bindings, in which each argument occurs at most once. For example, $\wp = \forall\mathtt{x}\forall\mathtt{y}\forall\mathtt{z}$ and $\flat = (\mathtt{f}, \mathtt{x})(\mathtt{m}, \mathtt{y})(\mathtt{c}, \mathtt{z})$ are the quantification and binding prefixes occurring in the formula $\varphi_1 = \wp\flat((\mathtt{Ft}^{\mathtt{cf}} \wedge \mathtt{Mt}^{\mathtt{cm}}) \leftrightarrow \mathtt{Pr}^{\mathtt{cfm}})$ previously described. Finally, a *derived relation* $\widehat{r} \in \widehat{\mathrm{Rl}}$ is a Boolean combination of relations in Rl all having the same arguments, while a *binding form* $\flat\widehat{r} \in \mathrm{BF}$ is an argument-closed formula obtained by the juxtaposition of a binding prefix to a derived relation. Note that $\flat((\mathtt{Ft}^{\mathtt{cf}} \wedge \mathtt{Mt}^{\mathtt{cm}}) \leftrightarrow \mathtt{Pr}^{\mathtt{cfm}})$ is not a binding form, as the three relations have different arguments.

▶ **Definition 3.4** (Binding-Form Fragments)**.** *Boolean-binding formulas* over $\mathcal{L}(\mathrm{Vr})$ are built by means of the following context-free grammar, where $\wp \in \mathrm{Qn}$ and $\flat\widehat{r} \in \mathrm{BF}$:

$$\varphi := \bot \mid \top \mid \wp\psi \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi); \qquad \psi := \flat\widehat{r} \mid (\psi \wedge \psi) \mid (\psi \vee \psi).$$

$\mathcal{L}$(Vr)-FOL[BB] denotes the enumerable set of all formulas over $\mathcal{L}$(Vr) generated by the principal rule φ. Moreover, the *conjunctive*, *disjunctive*, and *one binding* fragments of FOL (FOL[CB], FOL[DB], and FOL[1B], for short) are obtained, respectively, by weakening the secondary rule ψ as follows: $\psi := \flat\widehat{r} \mid (\psi \wedge \psi)$, $\psi := \flat\widehat{r} \mid (\psi \vee \psi)$, and $\psi := \flat\widehat{r}$.

As an example, consider the FOL[BB] sentence $\forall \mathtt{x}\forall \mathtt{y}\forall \mathtt{z}(((\mathtt{c},\mathtt{x})(\mathtt{f},\mathtt{y})\mathtt{Ft}^{\mathtt{cf}} \wedge (\mathtt{c},\mathtt{x})(\mathtt{m},\mathtt{z})\mathtt{Mt}^{\mathtt{cm}}) \leftrightarrow (\mathtt{c},\mathtt{x})(\mathtt{f},\mathtt{y})(\mathtt{m},\mathtt{z})\mathtt{Pr}^{\mathtt{cfm}})$. It is easy to see that this is equivalent to the formula φ_1 given above. In general, by applying a simple generalization of the classic procedure used to obtain a prenex normal form, which further pushes the bindings inside as much as possible, we can always transform a FOL formula into an equivalent FOL[BB] one, with only a linear blow-up.

▶ **Theorem 3.5** (Binding Normal Form). *For each* $\mathcal{L}$(Vr)-FOL *formula, there exists an equivalent* $\mathcal{L}$(Vr)-FOL[BB] *one.*

An immediate consequence of the previous theorem is that FOL[BB] inherits all computational and model-theoretic properties of FOL. Therefore, the following holds.

▶ **Corollary 3.6** (FOL[BB] Properties). FOL[BB] *does enjoy both Craig's interpolation and Beth's definability, but not the finite-model property. Also, it has a* PSPACE-COMPLETE *model-checking problem and an undecidable satisfiability problem.*

At this point, we describe some meaningful example to illustrate the expressive power of the other binding-form fragments. First extend the LS $\mathcal{L}_G$ of Section 2, by adding the two arguments $\mathtt{p}_1$ and $\mathtt{p}_2$, standing for "*person*", and the three binary relations $\mathtt{Sb}$, $\mathtt{Mr}$, and $\mathtt{Lv}$ with $\mathtt{Sb}^{\mathcal{L}_G} = \mathtt{Mr}^{\mathcal{L}_G} = \mathtt{Lv}^{\mathcal{L}_G} = \{\mathtt{p}_1, \mathtt{p}_2\}$, in place of "*Sibling*", "*Married*", and "*inLove*", respectively. By definition, two siblings share the same parents. This can be expressed by the FOL[DB] sentence $\forall \mathtt{x}\forall \mathtt{y}\forall \mathtt{z}\forall \mathtt{w}(((\mathtt{p}_1,\mathtt{x})(\mathtt{p}_2,\mathtt{y})\mathtt{Sb}^{\mathtt{p}_1\mathtt{p}_2} \wedge (\mathtt{c},\mathtt{x})(\mathtt{f},\mathtt{z})(\mathtt{m},\mathtt{w})\mathtt{Pr}^{\mathtt{cfm}}) \rightarrow (\mathtt{c},\mathtt{y})(\mathtt{f},\mathtt{z})(\mathtt{m},\mathtt{w})\mathtt{Pr}^{\mathtt{cfm}})$, where, for the sake of readability, we use the form $(\psi_1 \wedge \psi_2) \rightarrow \psi_3$ to represent $\neg\psi_1 \vee \neg\psi_2 \vee \psi_3$. In the romantic literature it is common to find an unrequited love, whose scenario may be represented with the FOL[CB] sentence $\exists \mathtt{x}\exists \mathtt{y}\exists \mathtt{z}((\mathtt{p}_1,\mathtt{x})(\mathtt{p}_2,\mathtt{y})\mathtt{Lv}^{\mathtt{p}_1\mathtt{p}_2} \wedge (\mathtt{p}_1,\mathtt{y})(\mathtt{p}_2,\mathtt{x})\neg\mathtt{Lv}^{\mathtt{p}_1\mathtt{p}_2} \wedge (\mathtt{p}_1,\mathtt{y})(\mathtt{p}_2,\mathtt{z})\mathtt{Lv}^{\mathtt{p}_1\mathtt{p}_2}))$. Usually, two married people cannot be sibling and should be in love. The FOL[1B] sentence $\forall \mathtt{x}\forall \mathtt{y}(\mathtt{p}_1,\mathtt{x})(\mathtt{p}_2,\mathtt{y})(\mathtt{Mr}^{\mathtt{p}_1\mathtt{p}_2} \rightarrow \neg\mathtt{Sb}^{\mathtt{p}_1\mathtt{p}_2} \wedge \mathtt{Lv}^{\mathtt{p}_1\mathtt{p}_2})$ precisely expresses this fact. To conclude, consider the FOL[DB] sentence $\forall \mathtt{x}\forall \mathtt{y}((\mathtt{p}_1,\mathtt{x})(\mathtt{p}_2,\mathtt{y})\mathtt{Mr}^{\mathtt{p}_1\mathtt{p}_2} \rightarrow (\mathtt{p}_1,\mathtt{y})(\mathtt{p}_2,\mathtt{x})\mathtt{Mr}^{\mathtt{p}_1\mathtt{p}_2})$ stating that the relation $\mathtt{Mr}$ is symmetric. The only reason why we cannot express it in FOL[1B] is that the two bindings are permutations of each other. Therefore, if we allow the use of permutations in the definition of derived relations, we obtain a strictly more expressive FOL[1B] fragment able to describe the symmetric property as follows: $\forall \mathtt{x}\forall \mathtt{y}(\mathtt{p}_1,\mathtt{x})(\mathtt{p}_2,\mathtt{y})(\mathtt{Mr}^{\mathtt{p}_1\mathtt{p}_2} \rightarrow \{\mathtt{p}_1\mathtt{p}_2 \mapsto \mathtt{p}_2\mathtt{p}_1\}\mathtt{Mr}^{\mathtt{p}_1\mathtt{p}_2})$. By using techniques similar to those developed for FOL[1B], one can prove that such an extension retains exactly the same model-theoretic and algorithmic properties. However, for the sake of simplicity, we prefer to postpone this study to the extended version of the paper.

Before continuing, we want to point out an interesting connection between FOL[1B] and the language par excellence for relational databases. Indeed, without considering negation, it is not hard to see that sentences of our logic correspond to expressions of relational algebra of the form unions/natural joins of projections/divisions of positive Boolean combinations of atomic relations. The connection is still valid in the presence of negation, but it requires a deeper analysis, as observed in Section 4. Unfortunately, this correspondence is not helpful in the derivation of the model-theoretic and satisfiability results described in the following.

An expert reader might also have noticed a significant similarity between the concept of binding form and the notion of uniformity formalized in [26], which is used to introduce

the uniform one-dimensional fragment of FOL. Nevertheless, the syntax of the four binding fragments does not comply with the further required one-dimensional restriction. Therefore, these logics should be orthogonal *w.r.t.* the expressive power. However, a further analysis of connections and differences is in order.

Results. The negative features concerning FOL[BB] have spurred us to investigate the three simpler fragments FOL[CB], FOL[DB], and FOL[1B], to which we devote the remaining part of this section by giving an overview of the obtained model-theoretic and algorithmic results.

As far as the expressiveness is concerned, we show that FOL[1B] is strictly less expressive than FOL[CB] and FOL[DB]. This is done by means of a suitable concept of bisimulation under which the first fragment is proved to be invariant. Also, FOL[1B] is incomparable with other logics studied in the literature. Indeed, on the one hand, by means of the sentence $\forall \mathtt{x} \forall \mathtt{y} \forall \mathtt{z} (\mathtt{a}_1, \mathtt{x})(\mathtt{a}_2, \mathtt{y})(\mathtt{a}_3, \mathtt{z}) \mathtt{r}^{\mathtt{a}_1 \mathtt{a}_2 \mathtt{a}_3}$, we can state the completeness of a ternary relation $\mathtt{r}$, which is not possible to express in any of the fragments FOL[2VAR], FOL[CG], and FOL[GN]. Moreover, FOL[FL] cannot express the reflexivity of a binary relation $\mathtt{r}$ [43], which is easily described by the FOL[1B] sentence $\forall \mathtt{x} (\mathtt{a}_1, \mathtt{x})(\mathtt{a}_2, \mathtt{x}) \mathtt{r}^{\mathtt{a}_1 \mathtt{a}_2}$. On the other hand, due to the invariance under the particular bisimulation referred to above, we have shown that the simple modal logic formula $\Box \Diamond \mathtt{p}$, does not have an equivalent formulation in FOL[1B]. Finally, it is possible to observe that both FOL[CB] and FOL[DB] are not closed under negation and the former, differently from the latter, strictly subsumes the conjunctive query fragment of FOL.

▶ **Theorem 3.7** (Expressiveness). FOL[1B] *is strictly less expressive than* FOL[CB] *and* FOL[DB] *and incomparable with* FOL[2VAR], FOL[CG], FOL[GN], FOL[FL].

Although FOL[CB] and FOL[DB] have a more constrained syntax than FOL[BB], they do not have an easier model-checking problem. We can prove this by employing the classic reduction from the satisfiability problem of QBF, which is known to be PSPACE-COMPLETE.

▶ **Theorem 3.8** (FOL[CB] & FOL[DB] MC). *Both* FOL[CB] *and* FOL[DB] *have a* PSPACE-COMPLETE *model-checking problem.*

For FOL[DB], the situation is even worse. In fact, it is not hard to see that this fragment does not enjoy the finite-model property, as we can express the existence of an unbounded strict partial order [15]. Moreover, we can show that it is a conservative reduction class [11].

▶ **Theorem 3.9** (FOL[DB] FMP & SAT). FOL[DB] *does not enjoy the finite-model property and has an undecidable satisfiability problem.*

Once observed that the negation of a FOL[CB] formula is equivalent to a FOL[DB] one and *vice versa*, we immediately derive that the validity problem for FOL[CB] is undecidable. Nevertheless, we conjecture that this logic is model-theoretically and algorithmically well-behaved, as we think that the techniques developed for FOL[1B] can be suitably adapted to work with the more expressive fragment as well.

▶ **Conjecture 3.10** (FOL[CB] FMP & SAT). FOL[CB] *does enjoy the finite-model property and has a decidable satisfiability problem.*

We now focus on FOL[1B]. First of all, we prove that it is model-theoretically well-behaved, as it enjoys the finite-model property and both Craig's interpolation and Beth's definability, which are considered as mandatory properties for a "nice" FOL fragment [3]. To do this, we devise a novel technique that allows us to determine which subsentences of a given sentence might prevent its satisfiability. In other words, we propose a criterion that

identifies a set of *templates* of a formula, *i.e.*, pairs of quantification and binding prefixes of its subformulas, that may enforce inconsistent requirements on part of the underlying model. Such a set of templates is said to be *overlapping*. For example, consider the sentence $\forall \mathtt{x} \forall \mathtt{y} (\mathtt{a}_1, \mathtt{x})(\mathtt{a}_2, \mathtt{y}) \mathtt{r}^{\mathtt{a}_1 \mathtt{a}_2} \wedge \exists \mathtt{x} (\mathtt{a}_1, \mathtt{x})(\mathtt{a}_2, \mathtt{x}) \neg \mathtt{r}^{\mathtt{a}_1 \mathtt{a}_2}$. It is immediate to see that it is not satisfiable, as there is an assignment of the arguments $\mathtt{a}_1$ and $\mathtt{a}_2$, shared by both the templates $(\forall \mathtt{x} \forall \mathtt{y}, (\mathtt{a}_1, \mathtt{x})(\mathtt{a}_2, \mathtt{y}))$ and $(\exists \mathtt{x}, (\mathtt{a}_1, \mathtt{x})(\mathtt{a}_2, \mathtt{x}))$, which leads to the inconsistent formula $\mathtt{r}^{\mathtt{a}_1 \mathtt{a}_2} \wedge \neg \mathtt{r}^{\mathtt{a}_1 \mathtt{a}_2}$. These templates are, in fact, overlapping. By exploiting this criterion, we are able to build a particular kind of a finite Herbrand structure [8, 54] that satisfies the formula iff the overlapping templates only require consistent properties on the shared assignments. This is the basis for the finite-model property. The same tool is also used to find a particular normal form for FOL[1B] that allows us to prove Craig's interpolation, from which Beth's definability immediately follows. With more detail, we reduce the search for an interpolant between two FOL[1B] sentences φ_1 and φ_2 to the same problem between two propositional formulas η_1 and η_2 suitably obtained from the derived relations composing the matrices of the former. As for the standard syntax, in the new one, an interpolant is a sentence φ over the signature common to φ_1 and φ_2 such that $\varphi_1 \Rightarrow \varphi \Rightarrow \varphi_2$ holds. For instance, consider the implication $\exists \mathtt{x} \forall \mathtt{y} \flat (\mathtt{p} \wedge (\mathtt{q} \rightarrow \mathtt{r})) \wedge \forall \mathtt{x} \exists \mathtt{y} \flat (\mathtt{p} \wedge (\mathtt{r} \rightarrow \mathtt{q})) \Rightarrow \exists \mathtt{x} \exists \mathtt{y} \flat (\mathtt{s} \rightarrow (\mathtt{q} \leftrightarrow \mathtt{r}))$, where $\flat = (\mathtt{x}, \mathtt{a}_1)(\mathtt{y}, \mathtt{a}_2)$. Now, it is not hard to see that $\exists \mathtt{x} \exists \mathtt{y} \flat (\mathtt{q} \leftrightarrow \mathtt{r})$ is the associated FOL[1B] interpolant, where $\mathtt{q} \leftrightarrow \mathtt{r}$ is obtained as the propositional one for $(\mathtt{p} \wedge (\mathtt{q} \rightarrow \mathtt{r})) \wedge (\mathtt{p} \wedge (\mathtt{r} \rightarrow \mathtt{q})) \Rightarrow (\mathtt{s} \rightarrow (\mathtt{q} \leftrightarrow \mathtt{r}))$.

▶ **Theorem 3.11** (FOL[1B] FMP, CI & BD). FOL[1B] *does enjoy the finite-model property and both Craig's interpolation and Beth's definability.*

FOL[1B] is very well-behaved from the algorithmic point of view too. Indeed, we can provide a PTIME model-checking procedure *w.r.t.* the combined complexity, which is based on a linear reduction to a two-player reachability game, whose solution is known to be in PTIME [30]. It remains open whether the problem is also hard for this class.

▶ **Theorem 3.12** (FOL[1B] MC). FOL[1B] *has a* PTIME *model-checking problem.*

We finally discuss the satisfiability problem for FOL[1B], which we prove to be complete for the third level of the polynomial hierarchy, *i.e.*, $\Sigma^{\mathrm{P}}_3 = \mathrm{NPTIME}^{\mathrm{CONPTIME}^{\mathrm{NPTIME}}}$. In other words, it is solvable by an NPTIME Turing machine having access to a CONPTIME oracle that, in turn, can exploit an NPTIME advice. For the upper bound, thanks to the characterization of satisfiability via the overlapping templates mentioned above, we are able to reduce the problem to a three-round two-player game, in which each player has either NPTIME or CONPTIME computational power. The lower bound follows from a reduction from the satisfiability problem of QBF sentences with alternation 2. Intuitively, we transform a formula of the form $\exists^* \forall^* \exists^* \psi$ into an equisatisfiable conjunction $\varphi_\exists \wedge \varphi_\forall \wedge \varphi_\psi$ of FOL[1B] sentences such that the first two take care of the initial existential and universal quantifications, while the last handles the matrix. Observe that, since FOL[1B] is closed under negation, its *validity* and *implication problems* inherit the same complexity.

▶ **Theorem 3.13** (FOL[1B] SAT). FOL[1B] *has a* Σ^{P}_3-COMPLETE *satisfiability problem.*

For the sake of space, we exclusively devote the remaining part of this work to sketching the proofs of the two algorithmic results concerning FOL[1B].

4 Model Checking

We now describe a PTIME model-checking procedure for FOL[1B] sentences of unbounded width based on a reduction to a reachability game [30] over a tree arena, whose depth and

size are bound by, respectively, the number of quantifiers in the sentence and the size of the underlying RS. It is worth noting that the verification of a sentence φ can be reduced to a linear number of checks of its subsentences of the form $\wp\flat\widehat{r}$. Thus, we just focus on the latter.

Matrix Normalization. Before starting, we need to introduce the concept of restricted Boolean combination, *i.e.*, a Boolean formula in which the negation is replaced by the difference connective $\setminus$, whose semantics is set as follows: $\phi_1 \setminus \phi_2 \triangleq \phi_1 \wedge \neg\phi_2$. Now, we can show that, for each derived relation $\widehat{r}$, there exists a restricted Boolean combination $\widehat{r}^\star$ with $|\widehat{r}^\star| = \mathrm{O}(|\widehat{r}|)$ such that either $\widehat{r} \equiv \widehat{r}^\star$ or $\widehat{r} \equiv \neg\widehat{r}^\star$. This can be proved by induction via De Morgan's laws and the substitutions of $\varphi_1 \wedge \neg\varphi_2$ and $\varphi_1 \vee \neg\varphi_2$ with $\varphi_1 \setminus \varphi_2$ and $\neg(\varphi_2 \setminus \varphi_1)$, respectively. In case $\widehat{r} \equiv \widehat{r}^\star$, it holds that $\wp\flat\widehat{r} \equiv \wp\flat\widehat{r}^\star$. If $\widehat{r} \equiv \neg\widehat{r}^\star$, instead, we have that $\wp\flat\widehat{r} \equiv \neg\overline{\wp}\flat\widehat{r}^\star$, where the quantification prefix $\overline{\wp}$ is the dual of $\wp$, *i.e.*, every existential (*resp.*, universal) quantifier in $\wp$ is replaced by a universal (*resp.*, existential) one in $\overline{\wp}$. Hence, *w.l.o.g.*, we can assume $\widehat{r}$ to be a restricted Boolean combination of relations.

Model-Checking Procedure. The first step in our procedure is to transform the restricted derived relation $\widehat{r}$ into a fresh atomic relation $r^\star$ containing all and only the assignments satisfying $\widehat{r}$. From the original LS $\mathcal{L}{=}\langle \mathrm{Ar}, \mathrm{Rl}, \mathsf{ar}\rangle$ and $\mathcal{L}$-RS $\mathcal{R}{=}\langle \mathrm{Dm}, \mathsf{rl}\rangle$, we build the new LS $\mathcal{L}^\star \triangleq \langle \mathsf{ar}(\widehat{r}), \{r^\star\}, \mathsf{ar}^\star\rangle$ with $\mathsf{ar}^\star(r^\star) \triangleq \mathsf{ar}(\widehat{r})$ and the $\mathcal{L}^\star$-RS $\mathcal{R}^\star \triangleq \langle \mathrm{Dm}, \mathsf{rl}^\star\rangle$ with $\mathsf{rl}^\star(r^\star) \triangleq \mathsf{v}(\widehat{r})$, where the valuation function v is set as follows: *(i)* $\mathsf{v}(r) \triangleq \mathsf{rl}(r)$; *(ii)* $\mathsf{v}(\phi_1 \wedge \phi_2) \triangleq \mathsf{v}(\phi_1) \cap \mathsf{v}(\phi_2)$; *(iii)* $\mathsf{v}(\phi_1 \vee \phi_2) \triangleq \mathsf{v}(\phi_1) \cup \mathsf{v}(\phi_2)$; *(iv)* $\mathsf{v}(\phi_1 \setminus \phi_2) \triangleq \mathsf{v}(\phi_1) \setminus \mathsf{v}(\phi_2)$. It is easy to observe that $\mathcal{R} \models \wp\flat\widehat{r}$ iff $\mathcal{R}^\star \models \wp\flat r^\star$. Moreover, such a construction can be done in time $\mathrm{O}(|\widehat{r}| \cdot \|\mathcal{R}\|)$. Note that, in this step, we are just employing the classic set operations of relational algebra.

The second step is devoted to the transformation of the binding form $\flat r^\star$ into an injective one $\flat^\natural r^\natural$ having $\flat^\natural \triangleq \prod_{x \in \mathsf{free}(\flat r^\star)}(x, x)$, *i.e.*, a formula in which every argument is bound to a different variable. From the previous LS $\mathcal{L}^\star$ and $\mathcal{L}^\star$-RS $\mathcal{R}^\star$, we build the new LS $\mathcal{L}^\natural \triangleq \langle \mathsf{free}(\flat r^\star), \{r^\natural\}, \mathsf{ar}^\natural\rangle$, where $\mathsf{ar}^\natural(r^\natural) \triangleq \mathsf{free}(\flat r^\star)$, and the $\mathcal{L}^\natural$-RS $\mathcal{R}^\natural \triangleq \langle \mathrm{Dm}, \mathsf{rl}^\natural\rangle$ with $\mathsf{rl}^\natural(r^\natural) \triangleq \{\chi \in \mathrm{Asg}_{\mathrm{Dm}} : \mathcal{R}^\star, \chi \models \flat r^\star\}$. Now, we immediately obtain that $\mathcal{R}^\star \models \wp\flat r^\star$ iff $\mathcal{R}^\natural \models \wp\flat^\natural r^\natural$. Moreover, the construction can be done in time $\mathrm{O}(|\flat| \cdot \|\mathcal{R}\|)$. In this case, we are making use of the three relational algebra operations of selection, projection, and renaming.

With the third and final step, we reduce the verification of $\mathcal{R}^\natural \models \wp\flat^\natural r^\natural$ to the win of the first player in a suitably constructed reachability game. Firstly, fix an ordering $<\subseteq \mathrm{Dm}\times\mathrm{Dm}$ among the values of the carrier domain Dm. Then, let $<_\wp$ be the ordering on variables induced by the quantification prefix $\wp$. Now, we can sort in the lexicographic order $(<, <_\wp)$ all tuple functions contained into the interpretation $\mathsf{rl}^\natural(r^\natural)$ of the relation $r^\natural$. At this point, from these tuples, we can construct a prefix-tree data structure, *a.k.a.* trie [17], taking values in the set $\{\bot, \top\}$. In particular, to each tuple in $\mathsf{rl}^\natural(r^\natural)$ we assign $\top$, while to every minimal partial tuple that does not have an extension in the interpretation we assign $\bot$. As an example, in Figure 2, we depict the trie corresponding to $\mathrm{Dm} = \{1, 2, 3\}$, $\mathsf{ar}^\natural(r^\natural) = \{\mathtt{x}, \mathtt{y}, \mathtt{z}\}$, $\mathtt{x}{<_\wp}\mathtt{y}{<_\wp}\mathtt{z}$, and $\mathsf{rl}^\natural(r^\natural) = \{\mathtt{xyz} \mapsto 111, \mathtt{xyz} \mapsto 123, \mathtt{xyz} \mapsto 132, \mathtt{xyz} \mapsto 333\}$. Note that, as there are no tuple functions in $\mathsf{rl}^\natural(r^\natural)$ having the variable $\mathtt{x}$ set to 2, it follows that the corresponding node of the tree has a $\bot$-child. Similarly, the second node corresponding to $\mathtt{y}$ has a $\bot$-child, since

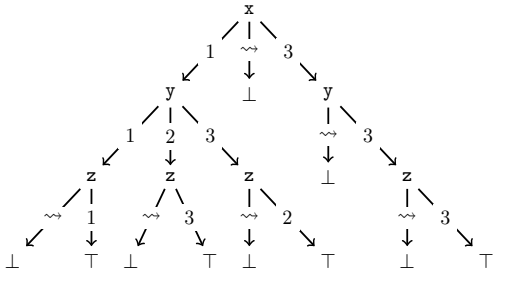

Figure 2 A prefix-tree reachability game arena.

there are no tuples in the interpretation of $r^\natural$ mapping $\mathtt{x}$ to 3 and $\mathtt{y}$ to 1 or 2. To complete the construction of the arena we only need to assign the nodes to the players. If a variable x is existentially (*resp.*, universally) quantified in $\wp$, the corresponding node of the tree is assigned to the first (*resp.*, second) player. Now, it is not hard to prove that $\mathcal{R}^\natural \models \wp \flat^\natural r^\natural$ iff the first player wins the reachability game to $\top$ on this trie. For instance, consider again the above arena and suppose that $\wp = \exists\mathtt{x}\forall\mathtt{y}\exists\mathtt{z}$. Then, $\mathcal{R}^\natural \models \wp \flat^\natural r^\natural$, as the first player has a winning strategy on the nodes $\mathtt{x}$ and $\mathtt{z}$. On the contrary, assume $\wp = \forall\mathtt{x}\exists\mathtt{y}\exists\mathtt{z}$. In this case $\mathcal{R}^\natural \not\models \wp \flat^\natural r^\natural$, since the second player can choose to reach the $\bot$-child of $\mathtt{x}$.

Summing up, since the problem of determining the winner of a turn-based reachability game is solvable in PTime [30], the complexity of our procedure immediately follows.

5 Satisfiability

We finally come to the more technical part of the paper, in which we provide the satisfiability procedure for FOL[1B]. In addition we describe the key model-theoretic tool at its basis. As mentioned before, this is focused on the concept of overlapping templates, *i.e.*, intuitively, a set $\{(\wp_i, \flat_i)\}$ of quantification and binding prefixes for which *(i)* there is a strict total order between the arguments that agrees, via $\flat_i$, with the functional dependences of all $\wp_i$ and *(ii)* each argument is bound to at most one existential variable. For the sake of space, here we just give the basic definitions that are necessary to formalize this concept and state the main characterization theorem. All proofs together with further notions and details will be reported in the extended version of this work.

In order to have a deeper intuition on the novel model-theoretic tool, we first describe four examples, built on the LS $\mathcal{L} = \langle\{\mathtt{a},\mathtt{b},\mathtt{c}\},\{\mathtt{q},\mathtt{r}\},\mathsf{ar}\rangle$ with $\mathsf{ar}(\mathtt{q}) = \mathsf{ar}(\mathtt{r}) = \{\mathtt{a},\mathtt{b},\mathtt{c}\}$, which cover some interesting cases of correlation between the satisfiability of a sentence and the overlapping property of its templates. Consider the sentence $\varphi_1 = \wp_1\flat_1\mathtt{q} \wedge \wp_2\flat_2\neg\mathtt{r} \wedge \wp_3\flat_3(\mathtt{q} \leftrightarrow \mathtt{r})$, where $\wp_1 = \forall\mathtt{x}\exists\mathtt{y}\forall\mathtt{z}$, $\wp_2 = \forall\mathtt{y}\exists\mathtt{z}\forall\mathtt{x}$, $\wp_3 = \forall\mathtt{x}\forall\mathtt{y}\forall\mathtt{z}$, and $\flat_1 = \flat_2 = \flat_3 = (\mathtt{a},\mathtt{x})(\mathtt{b},\mathtt{y})(\mathtt{c},\mathtt{z})$. It is not hard to see that φ_1 is unsatisfiable, since it requires the inconsistent derived relations $\mathtt{q}$, $\neg\mathtt{r}$, and $\mathtt{q} \leftrightarrow \mathtt{r}$ to hold on the same tuple function $\mathtt{t}$. Indeed, $\wp_1\flat_1\mathtt{q}$ forces $\mathtt{q}$ to hold on all tuples $\mathtt{t}_1$ satisfying the constraint $\mathtt{t}_1(\mathtt{b}) = \mathtt{f}_1(\mathtt{t}_1(\mathtt{a}))$, where $\mathtt{f}_1$ is a Skolem function for $\mathtt{y}$ in $\wp_1$. Similarly, $\wp_2\flat_2\neg\mathtt{r}$ demands $\neg\mathtt{r}$ on all tuples $\mathtt{t}_2$ with $\mathtt{t}_2(\mathtt{c}) = \mathtt{f}_2(\mathtt{t}_2(\mathtt{b}))$, where $\mathtt{f}_2$ is a Skolem function for $\mathtt{z}$ in $\wp_2$. Finally, $\wp_3\flat_3(\mathtt{q} \leftrightarrow \mathtt{r})$ enforces $\mathtt{q} \leftrightarrow \mathtt{r}$ on every possible tuple. Hence, the sentence φ_1 requires $\psi = \mathtt{q} \wedge \neg\mathtt{r} \wedge (\mathtt{q} \leftrightarrow \mathtt{r})$ on all tuples $\mathtt{t}$ with $\mathtt{t}(\mathtt{b}) = \mathtt{f}_1(\mathtt{t}(\mathtt{a}))$ and $\mathtt{t}(\mathtt{c}) = \mathtt{f}_2(\mathtt{t}(\mathtt{b}))$, which leads to an inconsistency. Observe that we can find such tuples because of the dependency order $\mathtt{a} \rightsquigarrow \mathtt{b} \rightsquigarrow \mathtt{c}$ among the arguments, which is compatible with the functional dependences of $\wp_1$ and $\wp_2$ via the bindings $\flat_1$ and $\flat_2$. Also, there are no arguments associated to more than one existential variable. Consequently the set of templates $\{(\wp_1,\flat_1),(\wp_2,\flat_2),(\wp_3,\flat_3)\}$ is said to be overlapping. Now, let φ_2 be the sentence obtained from φ_1 by replacing the prefixes $\wp_3$ and $\flat_3$ with $\forall\mathtt{x}\forall\mathtt{y}$ and $(\mathtt{a},\mathtt{x})(\mathtt{b},\mathtt{y})(\mathtt{c},\mathtt{y})$. We show that φ_2 is satisfiable on an RS of order $|\mathrm{Dm}| = 2$. First note that $\wp_3\flat_3(\mathtt{q} \leftrightarrow \mathtt{r})$ enforces $\mathtt{q} \leftrightarrow \mathtt{r}$ only on tuples $\mathtt{t}_3$ with $\mathtt{t}_3(\mathtt{b}) = \mathtt{t}_3(\mathtt{c})$. Moreover, choose a Skolem function $\mathtt{f}_2$ for $\mathtt{z}$ in $\wp_2$ such that $\mathtt{f}_2(\alpha) \neq \alpha$, for every $\alpha \in \mathrm{Dm}$, and suppose that φ_2 requires ψ on a tuple $\mathtt{t}$. In this case, we have that $\mathtt{t}(\mathtt{c}) = \mathtt{f}_2(\mathtt{t}(\mathtt{b}))$ and $\mathtt{t}(\mathtt{b}) = \mathtt{t}(\mathtt{c})$, *i.e.*, $\mathtt{t}(\mathtt{c}) = \mathtt{f}_2(\mathtt{t}(\mathtt{c}))$, which is impossible. Observe that there are no such tuples $\mathtt{t}$ because of a cyclic dependence $\mathtt{c} \rightsquigarrow \mathtt{c}$ caused by the second and third sentence. Therefore, the set of templates $\{(\wp_2,\flat_2),(\wp_3,\flat_3)\}$ is not overlapping. Similarly, consider the sentence φ_3 drawn from φ_1 by substituting $\forall\mathtt{z}\exists\mathtt{x}\forall\mathtt{y}$ for the prefix $\wp_3$. Also in this case, φ_3 is satisfiable on an RS of order 2. Indeed, suppose it requires ψ on a tuple $\mathtt{t}$. Then, we have that $\mathtt{t}(\mathtt{b}) = \mathtt{f}_1(\mathtt{t}(\mathtt{a}))$, $\mathtt{t}(\mathtt{c}) = \mathtt{f}_2(\mathtt{t}(\mathtt{b}))$, and $\mathtt{t}(\mathtt{a}) = \mathtt{f}_3(\mathtt{t}(\mathtt{c}))$, where $\mathtt{f}_3$ is a Skolem

function for $\mathtt{x}$ in $\wp_3$. Thus, $\mathsf{t}(\mathtt{a}) = \mathsf{f}_3(\mathsf{f}_2(\mathsf{f}_1(\mathsf{t}(\mathtt{a}))))$. Now, it is not hard to find f_1, f_2, and f_3 in such a way that $\mathsf{f}_3(\mathsf{f}_2(\mathsf{f}_1(\alpha))) \neq \alpha$, for every $\alpha \in \mathrm{Dm}$. So, the tuple t cannot exist, due to the cyclic dependence $\mathtt{a} \rightsquigarrow \mathtt{b} \rightsquigarrow \mathtt{c} \rightsquigarrow \mathtt{a}$. Here, $\{(\wp_1, \flat_1), (\wp_2, \flat_2), (\wp_3, \flat_3)\}$ is a set of non overlapping templates. Finally, derive the sentence φ_4 from φ_1, by setting the prefix $\wp_3$ to $\exists \mathtt{z} \forall \mathtt{x} \forall \mathtt{y}$. Again, φ_4 is satisfiable, as the argument $\mathtt{c}$ is existential in both $(\wp_2, \flat_2)$ and $(\wp_3, \flat_3)$. So, one can find a Skolem constant for $\mathtt{z}$ in $\wp_3$ that is different from all possible values assumed by the Skolem function f_2 for $\mathtt{z}$ in $\wp_2$.

Satisfiability Characterization. In this technical subsection, we state the fundamental characterization theorem connecting the satisfiability of a sentence with the overlapping property of its templates. To do this, we first give the formalization of the latter concept together with some auxiliary definition. Then, we introduce two suitable graphs defined on the structural features of a sets of templates, which are used to define the notion of overlapping.

A *template* $\tau \triangleq (\wp, \flat) \in \mathrm{Tem}$ is a pair of a quantification and a binding prefix on the set of arguments $\mathsf{ar}(\tau) \subseteq \mathrm{Ar}$ and variables $\mathsf{vr}(\tau) \subseteq \mathrm{Vr}$. By $\mathrm{Tem}(\mathrm{A})$ we denote the subset of templates having argument set $\mathrm{A} \subseteq \mathrm{Ar}$. For a given template $\tau = (\wp, \flat)$, by $\exists(\tau)$ (*resp.*, $\forall(\tau)$) we denote the set of arguments that are associated with an existential (*resp.*, universal) variable in $\wp$ via $\flat$. Moreover, $\cong_\tau \subseteq \mathsf{ar}(\tau) \times \mathsf{ar}(\tau)$ and $\rightsquigarrow_\tau \subseteq \forall(\tau) \times \exists(\tau)$ represent, respectively, the *collapsing equivalence* and the *functional dependence* induced by τ, *i.e.*, for all $a_1, a_2 \in \mathsf{ar}(\tau)$, $a_1 \cong_\tau a_2$ iff $\flat(a_1) = \flat(a_2)$ and, for all $a_1 \in \forall(\tau)$ and $a_2 \in \exists(\tau)$, $a_1 \rightsquigarrow_\tau a_2$ iff the variable $\flat(a_1)$ occurs before the variable $\flat(a_2)$ in $\wp$. For instance, for the template $\tau = (\forall \mathtt{x} \exists \mathtt{y}, (\mathtt{a}, \mathtt{x})(\mathtt{b}, \mathtt{y})(\mathtt{c}, \mathtt{x}))$, it holds that $\exists(\tau) = \{\mathtt{b}\}$, $\forall(\tau) = \{\mathtt{a}, \mathtt{c}\}$, $\mathtt{a} \rightsquigarrow_\tau \mathtt{b}$, $\mathtt{c} \rightsquigarrow_\tau \mathtt{b}$, and $\mathtt{a} \cong_\tau \mathtt{c}$.

As set of vertexes of the above mentioned graphs, we use the set $\mathrm{Ar}^{\mathrm{A}}_{\mathrm{S}} \subseteq \mathrm{S} \times \mathrm{Ar}$ of *extended arguments* $e = (\tau, a)$, *i.e.*, pairs of a template $\tau(e) = \tau$ and one of its arguments $a(e) = a \in \mathsf{ar}(\tau) \cap \mathrm{A}$. Also, $\exists^{\mathrm{A}}_{\mathrm{S}}$ (*resp.*, $\forall^{\mathrm{A}}_{\mathrm{S}}$) represents the set of existential (*resp.*, universal) extended arguments, *i.e.*, the elements e such that $a(e) \in \exists(\tau(e))$ (*resp.*, $a(e) \in \forall(\tau(e))$).

The *collapsing graph* for S over A is the symmetric directed graph $\mathcal{C}^{\mathrm{A}}_{\mathrm{S}} \triangleq \langle \mathrm{Ar}^{\mathrm{A}}_{\mathrm{S}}, \cong^{\mathrm{A}}_{\mathrm{S}} \rangle$ with the extended arguments as vertexes and the edge relation given by $\cong^{\mathrm{A}}_{\mathrm{S}} \triangleq \{(e_1, e_2) \in \mathrm{Ar}^{\mathrm{A}}_{\mathrm{S}} \times \mathrm{Ar}^{\mathrm{A}}_{\mathrm{S}} : (\tau(e_1) = \tau(e_2) \Rightarrow a(e_1) \cong_{\tau(e_1)} a(e_2)) \wedge (\tau(e_1) \neq \tau(e_2) \Rightarrow a(e_1) = a(e_2))\}^+$. Intuitively, $\cong^{\mathrm{A}}_{\mathrm{S}}$ is the least equivalence relation on $\mathrm{Ar}^{\mathrm{A}}_{\mathrm{S}}$ that identifies the arguments bound to the same variable in some template. This graph is used to take into account the multiple possible associations of an argument with the existential variables, in order to formalize the property *(ii)* described at the beginning of the section. In Figure 3, we depict the collapsing graphs $\mathcal{C}_i = \mathcal{C}^{\mathrm{A}}_{\mathrm{S}_i}$ for the set of templates $\mathrm{S}_i = \{\tau_1 = (\wp_1, \flat_1), \tau_2 = (\wp_2, \flat_2), \tau_3 = (\wp_3, \flat_3)\}$ and arguments $\mathrm{A} = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$ associated with every single sentences φ_i given above, where $i \in \{1, 3, 4\}$. The dots simply represent the extended arguments obtained by intersecting rows and columns. In Figure 4, we report the collapsing graph $\mathcal{C}_2 = \mathcal{C}^{\mathrm{A}}_{\mathrm{S}_2}$ for the sentence φ_2, where the edge between the vertexes $(\tau_3, \mathtt{b})$ and $(\tau_3, \mathtt{c})$ is due to the binding $\flat_3$. Note that, since $\cong^{\mathrm{A}}_{\mathrm{S}}$ is an equivalence relation, we are omitting the transitive closure from the graphs.

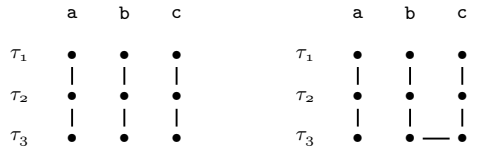

Figure 3 Collapsing graphs $\mathcal{C}_1$, $\mathcal{C}_3$, and $\mathcal{C}_4$.

Figure 4 Collapsing graph $\mathcal{C}_2$.

We can now define a property describing the case in which two existential arguments are forced to assume the same value. A collapsing graph $\mathcal{C}^{\mathrm{A}}_{\mathrm{S}}$ is *conflicting* iff there are two existential extended arguments $e_1, e_2 \in \exists^{\mathrm{A}}_{\mathrm{S}}$ such that $e_1 \cong^{\mathrm{A}}_{\mathrm{S}} e_2$ and, if $\tau(e_1) = \tau(e_2)$ then

$a(e_1) \not\cong_{\tau(e_1)} a(e_2)$. Intuitively, this property ensures the existence of two arguments that, at the same time, need to assume the same value, due to the collapsing equivalence on some template, and are both existentially quantified in possibly different templates. As an example, the collapsing graph $\mathcal{C}_2$ of φ_2 is conflicting, since $(\tau_1, \mathsf{b}) \cong^{\mathrm{A}}_{\mathrm{S}} (\tau_2, \mathsf{c})$. The same holds for the collapsing graph $\mathcal{C}_4$ of φ_4, since $(\tau_2, \mathsf{c}) \cong^{\mathrm{A}}_{\mathrm{S}} (\tau_3, \mathsf{c})$.

The second graph we introduce keeps track of the functional dependences between arguments that cross all the templates. This property has been informally described in point *(i)* at the beginning of this section. The *dependence graph* for S over A is the directed graph $\mathcal{D}^{\mathrm{A}}_{\mathrm{S}} \triangleq \langle \mathrm{Ar}^{\mathrm{A}}_{\mathrm{S}}, \rightsquigarrow^{\mathrm{A}}_{\mathrm{S}} \rangle$ with the extended arguments as vertexes and the edge relation given by $\rightsquigarrow^{\mathrm{A}}_{\mathrm{S}} \triangleq \cong^{\mathrm{A}}_{\mathrm{S}} \circ \{(e_1, e_2) \in \mathrm{Ar}^{\mathrm{A}}_{\mathrm{S}} \times \mathrm{Ar}^{\mathrm{A}}_{\mathrm{S}} : \tau(e_1) = \tau(e_2) \wedge a(e_1) \rightsquigarrow_{\tau(e_1)} a(e_2)\}$. In Figures 5, 6, and 7, we report the dependence graphs $\mathcal{D}_i = \mathcal{D}^{\mathrm{A}}_{\mathrm{S}_i}$ corresponding to the sentences φ_1, φ_2, and φ_3, respectively. The black arrows represent the functional dependences inside a single template, while the gray ones are obtained by the composition with the collapsing relation. Note that $\mathcal{D}_1$ is acyclic, so, we can build an order among the extended arguments that agrees with all functional dependences of the templates. In $\mathcal{D}_2$, instead, there is a loop on (τ_2, c) due to the structures of τ_2 and τ_3. Finally, $\mathcal{D}_3$ contains a cycle among (τ_3, a), (τ_1, b), and (τ_2, c).

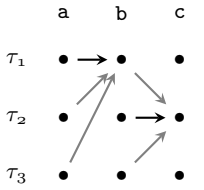

Figure 5 Dependence graph $\mathcal{D}_1$.

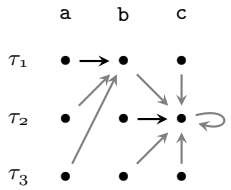

Figure 6 Dependence graph $\mathcal{D}_2$.

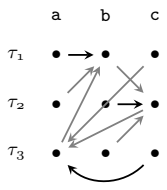

Figure 7 Dependence graph $\mathcal{D}_3$.

▶ **Definition 5.1** (Overlapping Templates). A set of templates $\mathrm{S} \subseteq \mathrm{Tem}(\mathrm{A})$ over a set of arguments $\mathrm{A} \subseteq \mathrm{Ar}$ is *overlapping* iff the collapsing graph $\mathcal{C}^{\mathrm{A}}_{\mathrm{S}}$ is not conflicting and the dependence graph $\mathcal{D}^{\mathrm{A}}_{\mathrm{S}}$ is acyclic.

Observe that there are similarities between the introduced concept of overlapping templates and notion of *weakly acyclic dependencies* of TGDs [41], which is known to be a sufficient property for the termination of the chase algorithm [1], one of the most useful tools in database theory to test query containment under constraints. This connection will be analysed in the journal version of this article.

We can finally state the characterization theorem. To do this, we first recall that, for a given set X and a Boolean formula η over X, we denote by $\mathsf{wit}(\eta) \subseteq 2^{\mathrm{X}}$ the set of *witnesses* of η, *i.e.*, the set of all possible subsets of X satisfying η.

▶ **Theorem 5.2** (Satisfiability Characterization). *A given* FOL[1B] *sentence φ is satisfiable iff there exists a witness $\mathrm{F} \in \mathsf{wit}(\varphi)$ such that, for all overlapping templates $\mathrm{S} \subseteq \mathsf{dom}(\mathsf{fr}) \cap \mathrm{Tem}(\mathrm{A})$ with $\mathrm{A} \subseteq \mathrm{Ar}$, it holds that $\bigwedge_{\tau \in \mathrm{S}} \mathsf{fr}(\tau)$ is a satisfiable Boolean combination of relations, where $\mathsf{fr} = \{(\wp, \flat) \in \mathrm{Tem} \mapsto \widehat{r} \in \widehat{\mathrm{Rl}} : \wp \flat \widehat{r} \in \mathrm{F}\}$ is the function associating each template with the corresponding derived relation in* F [1].

Satisfiability Procedure. We finally provide an algorithm for the solution of the satisfiability problem for FOL[1B], which can be interpreted as a satisfiability-modulo-theory procedure.

[1] Recall that a FOL[1B] sentence can be seen as a Boolean combination of sentences of the form $\wp \flat \widehat{r}$. Moreover, we are assuming that each quantification/binding prefix $\wp \flat$ in φ occurs only once, thus, fr is indeed a function (this can be ensured by a standard renaming of the variables).

Indeed, by means of a syntactic preprocessing based on the concept of overlapping templates, the search for a model of a FOL[1B] sentence is reduced to that of a sequence of Boolean formulas over the set of derived relations. The correctness of such an approach is crucially based on the fundamental characterization of Theorem 5.2. It is also interesting to observe that the procedure is independent from the size of the finite model derived in the proof of Theorem 3.11.

To understand the main idea behind the algorithm, it is useful to describe it through a simple three-round two-player turn-based game between the existential player, called Eloise, willing to show that a sentence φ is satisfiable, and the universal player, called Abelard, trying to do exactly the opposite. First, Eloise chooses a witness $\mathrm{F} \in \mathsf{wit}(\varphi)$ for φ seen as a Boolean combination of simpler subsentences of the form $\wp\flat\widehat{r}$. In this way, she identifies a formula function $\mathsf{fr} = \{(\wp,\flat) \in \mathrm{Tem} \mapsto \widehat{r} \in \widehat{\mathrm{Rl}} : \wp\flat\widehat{r} \in \mathrm{F}\}$ that describes F by associating each derived relation with the corresponding template. Then, Abelard chooses a subset of overlapping templates $\mathrm{S} \subseteq \mathsf{dom}(\mathsf{fr}) \cap \mathrm{Tem(A)}$ over a set of arguments $\mathrm{A} \subseteq \mathrm{Ar}$. At this point, Eloise wins the play iff the Boolean formula $\psi = \bigwedge_{\tau \in \mathrm{S}} \mathsf{fr}(\tau)$, obtained as the conjunction of all the derived relations associated with the templates in S, is satisfiable. If this is not the case, Abelard has spotted a subset $\{\wp\flat\mathsf{fr}(\tau) : \tau = (\wp,\flat) \in \mathrm{S}\}$ of the witness F that requires to verify the unsatisfiable property ψ on a certain valuation of arguments in A. Thus, F cannot have a model. Consequently, φ is satisfiable iff Eloise has a winning strategy for this game.

Algorithm 1: FOL[1B] Satisfiability Checker.

```
   signature sat : L(Vr)-FOL[1B] → {⊤, ⊥}
   function sat(φ)
1      foreach F ∈ wit(φ) do
2          fr ← {(℘, ♭) ∈ Tem ↦ r̂ ∈ R̂l : ℘♭r̂ ∈ F}
3          i ← ⊥
4          foreach S ⊆ dom(fr) ∩ Tem(A) with A ⊆ Ar do
5              if S is-overlapping-over A then
6                  if wit(⋀_{τ∈S} fr(τ)) = ∅ then
7                      i ← ⊤
8          if i = ⊥ then
9              return ⊤
10     return ⊥
```

We can now describe the pseudo-code of Algorithm 1. The deterministic counterpart of Eloise's choice is the selection of a witness $\mathrm{F} \in \mathsf{wit}(\varphi)$ in the loop at Line 1, which is followed by the computation of the corresponding formula function fr. At each iteration, a flag i is also set to $\bot$, with the aim to indicate that F is not inconsistent (a witness is consistent until proven otherwise). After this, we find the deterministic counterpart of Abelard's choice, implemented by the combination of a loop and a conditional statement at Lines 4 and 5, where a subset of overlapping templates $\mathrm{S} \subseteq \mathsf{dom}(\mathsf{fr}) \cap \mathrm{Tem(A)}$ over a set of arguments $\mathrm{A} \subseteq \mathrm{Ar}$ is selected. This is done in order to verify the inconsistency of the conjunction $\bigwedge_{\tau \in \mathrm{S}} \mathsf{fr}(\tau)$ at Line 6. If this check is positive then the flag i is switched to $\top$. Once all choices for Abelard are analysed, the computation reaches Line 8, where it is verified whether an inconsistency was previously found. In the negative case, the algorithm terminates by returning $\top$, with the aim to indicate that a good guess for Eloise is possible. In the case all witnesses are analysed, finding for each of them an inconsistency, Eloise has no winning strategy. Thus, the algorithm ends by returning $\bot$.

It remains to evaluate the complexity of the algorithm *w.r.t.* the length of the sentence φ. The verification at Lines 5-7 of Abelard's universal guess of Line 4 can be done in PTime with an NPTime advice for the Boolean satisfiability problem of Line 6. Thus, the check at Lines 2-

9 of Eloise's existential guess of Line 1 can be done in PTime with a CoNPTime$^{\text{NPTime}}$ advice for Line 4. Hence, the overall complexity is $\Sigma_3^{\text{P}} = \text{NPTime}^{\text{CoNPTime}^{\text{NPTime}}}$, *i.e.*, the problem belongs to the third level of the polynomial hierarchy.

6 Discussion

Trying to understand the reasons why some powerful extensions of modal logic are decidable, we introduced and studied a new family of FOL fragments based on the combinations of binding forms admitted in their formulas. In other words, we provided a novel criterion to classify FOL sentences focused on the associations between arguments and variables. A main features of this classification is to avoid the usual syntactic restrictions on quantifier patterns, number of variables, relation arities, and relativisation of quantifiers or negations. Therefore, it represents a suitable framework in which to study model-theoretic and algorithmic properties of extensions of modal logic, such as ATL* and SL[1G]. We analysed the expressiveness of the introduced fragments, showing that the simplest one, called *one binding* (FOL[1B]), is already incomparable with other important restrictions of FOL, such as the *clique guarded* (FOL[CG]) [19, 22], the *guarded negation* (FOL[GN]) [7], and the *fluted logic* (FOL[FL]) [44]. Moreover, we proved that it enjoys the finite-model property, a PTime model checking, a Σ_3^{P}-complete satisfiability, and a constructive version of both Craig's interpolation and Beth's definability, which are indirectly derived from the same properties of propositional logic. As future work, besides a deeper study of the *conjunctive* and *disjunctive* fragments (FOL[CB] and FOL[DB]), it is important to verify which of the stated results lift to the extensions of the introduced logics that comprise distinguished relations representing equivalences, orders, or the equality. Finally, it would be interesting to come up with a wider framework, which can accommodate the incomparable languages FOL[CG], FOL[GN], FOL[FL], and FOL[1B].

Acknowledgements We are very grateful to M. Benerecetti, L. Sauro, and M.Y. Vardi for many useful comments and discussions on a first draft of this work.

References

1 S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases.* Addison-Wesley, 1995.
2 R. Alur, T.A. Henzinger, and O. Kupferman. Alternating-Time Temporal Logic. *JACM*, 49(5):672–713, 2002.
3 H. Andréka, J. van Benthem, and I. Németi. Modal Languages And Bounded Fragments Of Predicate Logic. *JPL*, 27(3):217–274, 1998.
4 V. Bárány, M. Benedikt, and B. ten Cate. Rewriting Guarded Negation Queries. In *MFCS'13*, LNCS 8087, pages 98–110. Springer, 2013.
5 V. Bárány, G. Gottlob, and M. Otto. Querying the Guarded Fragment. In *LICS'10*, pages 1–10. IEEE Computer Society, 2010.
6 V. Bárány, B. ten Cate, and M. Otto. Queries with Guarded Negation. *PVLDB*, 5(11):1328–1339, 2012.
7 V. Bárány, B. ten Cate, and L. Segoufin. Guarded Negation. In *ICALP'11*, LNCS 6756, pages 356–367. Springer, 2011.
8 M. Ben-Ari. *Mathematical Logic for Computer Science (3rd ed.).* Springer, 2012.
9 D. Berwanger and E. Grädel. Games and Model Checking for Guarded Logics. In *LPAR'01*, LNCS 2250, pages 70–84. Springer, 2001.
10 E. Börger. Decision Problems in Predicate Logic. In *LC'82*, volume 112, pages 263–301. Elsevier, 1984.

11 E. Börger, E. Grädel, and Y. Gurevich. *The Classical Decision Problem.* Perspectives in Mathematical Logic. Springer, 1997.

12 K. Chatterjee, T.A. Henzinger, and N. Piterman. Strategy Logic. In *CONCUR'07*, LNCS 4703, pages 59–73. Springer, 2007.

13 E.F. Codd. A Relational Model of Data for Large Shared Data Banks. *CACM*, 13(6):377–387, 1970.

14 E.F. Codd. Relational Completeness of Data Base Sublanguages. *DS*, pages 65–98, 1972.

15 H.-D. Ebbinghaus and J. Flum. *Finite Model Theory.* Perspectives in Mathematical Logic. Springer, 1995.

16 R. Fagin and M.Y. Vardi. The Theory of Data Dependencies – A Survey. *MIP*, 34:19–71, 1972.

17 E. Fredkin. Trie Memory. *CACM*, 3(9):490–499, 1960.

18 E. Goncalves and E. Grädel. Decidability Issues for Action Guarded Logics. In *DL'00*, pages 123–132, 2000.

19 E. Grädel. Decision Procedures for Guarded Logics. In *CADE'99*, LNCS 1632, pages 31–51. Springer, 1999.

20 E. Grädel. On The Restraining Power of Guards. *JSL*, 64(4):1719–1742, 1999.

21 E. Grädel. Why are Modal Logics so Robustly Decidable? In *CTTCS'01*, pages 393–408. World Scientific, 2001.

22 E. Grädel. Guarded Fixed Point Logics and the Monadic Theory of Countable Trees. *TCS*, 288(1):129–152, 2002.

23 E. Grädel. Decidable Fragments of First-Order and Fixed-Point Logic. In *KWLCS'03*, 2003.

24 E. Grädel, P.G. Kolaitis, and M.Y. Vardi. On the Decision Problem for Two-Variable First-Order Logic. *BSL*, 3(1):53–69, 1997.

25 E. Grädel and I. Walukiewicz. Guarded Fixed Point Logic. In *LICS'99*, pages 45–54. IEEE Computer Society, 1999.

26 Lauri Hella and A. Kuusisto. One-Dimensional Fragment of First-Order Logic. In *AIML'14*, pages 274–293. College Publications, 2014.

27 D. Hilbert and W. Ackermann. *Grundzüge der Theoretischen Logik.* Die Grundlehren der Mathematischen Wissenschaften. Springer, 1928.

28 I. Hodkinson. Loosely Guarded Fragment of First-Order Logic has the Finite Model Property. *SL*, 70(2):205–240, 2002.

29 E. Hoogland and M. Marx. Interpolation and Definability in Guarded Fragments. *SL*, 70(3):373–409, 2002.

30 N. Immerman. Number of Quantifiers is Better Than Number of Tape Cells. *JCSS*, 22(3):384–406, 1981.

31 N. Immerman. Relational Queries Computable in Polynomial Time (Extended Abstract). In *STOC'82*, pages 147–152. Association for Computing Machinery, 1982.

32 N. Immerman. Relational Queries Computable in Polynomial Time. *IC*, 68(1-3):86–104, 1986.

33 N. Immerman. *Descriptive Complexity.* Graduate Texts in Computer Science. Springer, 1999.

34 E. Kieronski and A. Kuusisto. Complexity and Expressivity of Uniform One-Dimensional Fragment with Equality. In *MFCS'14*, LNCS 8634, pages 365–376. Springer, 2014.

35 L. Löwenheim. Über Möglichkeiten im Relativkalkül. *MA*, 76(4):447–470, 1915.

36 F. Mogavero, A. Murano, G. Perelli, and M.Y. Vardi. Reasoning About Strategies: On the Satisfiability Problem. Under submission.

37 F. Mogavero, A. Murano, G. Perelli, and M.Y. Vardi. What Makes ATL* Decidable? A Decidable Fragment of Strategy Logic. In *CONCUR'12*, LNCS 7454, pages 193–208. Springer, 2012.

38 F. Mogavero, A. Murano, G. Perelli, and M.Y. Vardi. Reasoning About Strategies: On the Model-Checking Problem. *TOCL*, 15(4):34:1–42, 2014.

39 F. Mogavero, A. Murano, and M.Y. Vardi. Reasoning About Strategies. In *FSTTCS'10*, LIPIcs 8, pages 133–144. Leibniz-Zentrum fuer Informatik, 2010.

40 M. Mortimer. On Languages with Two Variables. *MLQ*, 21(1):135–140, 1975.

41 A. Onet. The Chase Procedure and its Applications in Data Exchange. In *Data Exchange, Integration, and Streams.*, pages 1–37. Leibniz-Zentrum fuer Informatik, 2013.

42 W.C. Purdy. Complexity and Nicety of Fluted Logic. *SL*, 71(2):177–198, 2002.

43 W.C. Purdy. Inexpressiveness of First-Order Fragments. *AJL*, 4:1–12, 2006.

44 W.V. Quine. Variables Explained Away. *PAPS*, 104(3), 1960.

45 I. Sain. Beth's and Craig's Properties via Epimorphisms and Amalgamation in Algebraic Logic. In *ALUACS'88*, LNCS 425, pages 209–225. Springer, 1990.

46 S. Schewe. ATL* Satisfiability is 2ExpTime-Complete. In *ICALP'08*, LNCS 5126, pages 373–385. Springer, 2008.

47 S. Schewe and B. Finkbeiner. Satisfiability and Finite Model Property for the Alternating-Time muCalculus. In *CSL'06*, pages 591–605. Springer, 2006.

48 S.G. Simpson. Partial Realizations of Hilbert's Program. *JSL*, 53(2):349–363, 1988.

49 B. ten Cate and L. Segoufin. Unary Negation. In *STACS'11*, LIPIcs 9, pages 344–355. Leibniz-Zentrum fuer Informatik, 2011.

50 B. ten Cate and L. Segoufin. Unary Negation. *LMCS*, 9(3):1–46, 2013.

51 J.D. Ullman. *Principles of Database Systems.* Computer Science Press, 1982.

52 J. van Benthem. Dynamic Bits And Pieces. Technical report, University of Amsterdam, Amsterdam, Netherlands, 1997.

53 J. van Benthem. Modal Logic In Two Gestalts. In *AIML'98*, pages 91–118. CSLI Publications, 2000.

54 D. van Dalen. *Logic and Structure (3rd ed.).* Universitext. Springer, 1994.

55 M.Y. Vardi. The Complexity of Relational Query Languages (Extended Abstract). In *STOC'82*, pages 137–146. Association for Computing Machinery, 1982.

56 M.Y. Vardi. On the Complexity of Bounded-Variable Queries. In *PODS'95*, pages 266–276. Association for Computing Machinery, 1995.

57 M.Y. Vardi. Why is Modal Logic So Robustly Decidable? In *DCFM'96*, pages 149–184. American Mathematical Society, 1996.

58 D. Walther, C. Lutz, F. Wolter, and M. Wooldridge. ATL Satisfiability is Indeed ExpTime-Complete. *JLC*, 16(6):765–787, 2006.