Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI-20)
Journal Track

Proving Semantic Properties as First-Order Satisfiability (Extended Abstract)*

Salvador Lucas†
Valencian Research Institute for Artificial Intelligence (VRAIN)
Universitat Politècnica de València, Spain
slucas@dsic.upv.es

Abstract

The semantics of computational systems (e.g., relational and knowledge data bases, query-answering systems, programming languages, etc.) can often be expressed as (the specification of) a logical theory Th. Queries, goals, and claims about the behavior or features of the system can be expressed as formulas φ which should be checked with respect to the *intended model* of Th, which is often huge or even incomputable. In this paper we show how to prove such *semantic* properties φ of Th by just *finding a model* $\mathcal{A}$ of $\mathsf{Th} \cup \{\varphi\} \cup \mathcal{Z}_\varphi$, where $\mathcal{Z}_\varphi$ is an appropriate (possibly empty) theory depending on φ only. Applications to relational and deductive databases, rewriting-based systems, logic programming, and answer set programming are discussed.

1 Introduction

Logic is often used to implement the expected functionality of databases (queries) or programs (executing them). In this setting, the following approach is naturally adopted [Green and Raphael, 1968]:

$$\text{can } \varphi \text{ be } \textit{proved} \text{ from Th? (written } \mathsf{Th} \vdash \varphi) \tag{1}$$

where Th is a logical theory representing the computational system and φ represents a computation.

Example 1 (Running example) *Consider the following relational database about teachers* a, b, c, d *(of sort* Teacher*) and students* p, q, r *(of sort* Student*) together with a relation* teach *giving information about the students taught by each teacher, according to the* table *[Reiter, 1977, page 59]*

teach	*Teacher*	*Student*
	a	p
	a	q
	b	q
	c	r

This table describes a (many-sorted) first-order theory Teach *consisting of the following* facts*:*

$$\mathtt{teach}(\mathtt{a},\mathtt{p}),\quad \mathtt{teach}(\mathtt{b},\mathtt{q}),\quad \mathtt{teach}(\mathtt{a},\mathtt{q}),\quad \mathtt{teach}(\mathtt{c},\mathtt{r})$$

where teach *is a binary predicate with arguments of sort* Teacher *and* Student*, respectively. A query asking for the students taught by teacher* a *would be encoded as:*

$$(\exists y : \mathit{Student})\ \mathtt{teach}(\mathtt{a}, y) \tag{2}$$

McCune's (first-order) theorem prover Prover9[1] *[McCune, 2005 2010] obtains a proof of* (2).

Logic can also be used to prove *properties* of computational systems. For instance, knowledge stored in databases or computational properties of programs.

Example 2 *The claim "*every student is taught by some teacher*" is expressed as follows:*

$$(\forall y : \mathit{Student})(\exists x : \mathit{Teacher})\ \mathtt{teach}(x, y) \tag{3}$$

Prover9 fails to prove (3). This is not surprising as (3) is *not* a logical consequence of Teach. Example 2 suggests that the *logical-reasoning-as-logical-consequence* approach (1) is "too strong" in some cases: our intuition that (3) *should* hold is *not* supported by (1).

As remarked by Clark, sentences expressing properties should be checked *with respect to a given canonical model only* [Clark, 1980, Chapter 4]. Similarly, in the logical approach to relational databases [Nicolas and Gallaire, 1977], solving queries and checking functional dependencies and integrity constraints is thought as the *evaluation* of logical formulas φ with respect to the facts stored in the database which are considered as a logical *interpretation* (or canonical, Herbrand model). Thus, we naturally consider the following:

$$\text{is } \varphi \text{ satisfiable in } \mathcal{I}_{\mathsf{Th}}\text{? (written } \mathcal{I}_{\mathsf{Th}} \models \varphi) \tag{4}$$

where $\mathcal{I}_{\mathsf{Th}}$ represents such a canonical model. Note that φ in (4) does *not* need to be a *theorem* of Th (as in (1)); thus, we better generically call φ in (4) a *semantic property* of Th. As we prove in Example 14, (3) *is* a semantic property of Teach, thus recovering our intuition.

Thus, given a first-order theory Th, the following questions arise: (i) How to provide a generic notion of "*canonical*

†Partially supported by the EU (FEDER), projects RTI2018-094403-B-C32, PROMETEO/2019/098, and SP20180225.

*Extended abstract of an article entitled *Proving semantic properties as first-order satifiability*, published in the *Artificial Intelligence Journal* [Lucas, 2019].

[1] http://www.cs.unm.edu/~mccune/prover9/.

model" for Th? (ii) Accordingly, given a logical statement φ, what does it mean that φ "is a *property*" of Th? (iii) How to *prove* and *disprove* such properties? (iv) How to use the standard notions of provability and satisfiability for this purpose? (v) What kind of problems can be investigated in this way? Providing answers to these questions and related issues was one of the main goals of [Lucas, 2019] (although (i)–(v) are not explicitly formulated there). In the following, we provide answers to these questions.

Along the paper, we use standard notions of many-sorted first order-logic (MS-FOL), see [Goguen and Meseguer, 1987; Hodges, 1993; Mendelson, 1997].

2 Semantic Properties of FO Theories

A natural candidate to play the role of *canonical model* for a theory Th is the set $\mathsf{Th}^{\vdash}$ of *ground atoms* which can be proved from Th. Such a set of atoms can be seen as a *Herbrand interpretation* $\mathcal{I}_{\mathsf{Th}}$: each function symbol f is interpreted as a mapping which, given a tuple $t_1, \ldots, t_k$ of ground terms returns a ground term $f(t_1, \ldots, t_k)$; each predicate symbol P is interpreted as the set of atoms $P(t_1, \ldots, t_n)$ in $\mathsf{Th}^{\vdash}$ (more precisely, as the set of tuples $(t_1, \ldots, t_n)$ of ground terms such that $P(t_1, \ldots, t_n) \in \mathsf{Th}^{\vdash}$). This answers question (i) above: $\mathcal{I}_{\mathsf{Th}}$ is intended to be the canonical model for Th. In general, though, $\mathcal{I}_{\mathsf{Th}}$ may fail to be a model of Th [van Emden and Kowalski, 1976]: for $\mathsf{Th} = \{p(a) \vee p(b)\}$, we have $\mathsf{Th}^{\vdash} = \emptyset$; thus, $\mathcal{I}_{\mathsf{Th}}$ does *not* satisfy Th. However, for Horn theories[2] Th, $\mathcal{I}_{\mathsf{Th}} \models \mathsf{Th}$ holds [van Emden and Kowalski, 1976]. Question (ii) is answered with the following definition.

Definition 3 (Semantic property) *A formula φ is a* semantic property *of a theory* Th *iff* $\mathcal{I}_{\mathsf{Th}} \models \mathsf{Th} \cup \{\varphi\}$.

Remark 4 *If $\mathcal{I}_{\mathsf{Th}} \models \mathsf{Th}$, then either φ or $\neg\varphi$ is a semantic property of* Th. *If $\mathcal{I}_{\mathsf{Th}} \models \neg\varphi$ holds, then $\mathcal{I}_{\mathsf{Th}} \models \varphi$ does not hold and we say that φ is* not *a semantic property of* Th. *In contrast, there are theories* Th *and formulas φ where neither φ nor $\neg\varphi$ can be proved in* Th *(which is often said to be an* incomplete *theory).*

Regarding questions (iii) and (iv), for Horn theories Th, if φ can be proved from Th, then φ is a semantic property of Th. We can use any proof calculus (see [Fitting, 1996] for instance).

Remark 5 *Theories usually specify true instances of relations only [Clark, 1977] and no negative information can be derived from them by logical reasoning. This means that using theorem proving to verify a semantic property $\neg A$ for a given atom A is often hopeless.*

We show that semantic properties φ of theories Th can be proved as *satisfiability* of an extended theory $\mathsf{Th} \cup \{\varphi\} \cup \mathcal{Z}_\varphi$ where $\mathcal{Z}_\varphi$ is an auxiliary (possibly empty) theory which depends on the shape of φ only (see Corollary 13). For this purpose, we use the results in Section 3, based on the notion of *preservation* of logic formulas [Hodges, 1993].

[2] A *clause* is a disjunction $L_1 \vee \cdots \vee L_n$ of *literals* L_i (which can be atoms A_i or negations of atoms $\neg A_i$). A Horn theory is a set of clauses each of them having at most one atom.

Examples of use. Finally, regarding question (v), computational systems that can benefit from our approach, involving the specification of clausal or even Horn theories, include *Relational and Deductive Databases*; *Logic and Answer Set Programming*; and *Rewriting-Based Systems*. See [Lucas, 2019, Section 4] and also Section 6 below.

In this section we have advanced some answers to questions (i)–(v) above. The next sections provide further details.

3 Preservation of MSFO Sentences

In the following, Th is a theory and we consider sentences φ in *prenex* form (5) as follows:

$$(Q_1 x_1 : s_1) \cdots (Q_k x_k : s_k) \bigvee_{i=1}^{m} \bigwedge_{j=1}^{n_i} L_{ij} \tag{5}$$

where (a) for all $1 \leq i \leq m$ and $1 \leq j \leq n_i$, L_{ij} are *literals* $L_{ij} = A_{ij}$ or $L_{ij} = \neg A_{ij}$ for some atom A_{ij} (in the first case, we say that (the *sign* of) L_{ij} is *positive*; otherwise, it is *negative* [van Emden and Kowalski, 1976]) (b) $x_1, \ldots, x_k$ for some $k \geq 0$ are the variables occurring in those literals (of sorts $s_1, \ldots, s_k$, respectively), and (c) $Q_1, \ldots, Q_k$ are universal/existential quantifiers. A sentence (5) is said to be *positive* if all literals are; if, additionally, $Q_q = \exists$ for all $1 \leq q \leq k$, then it is an *Existentially Closed Boolean Combination of Atoms* (ECBCA).

Theorem 6 *Let $\mathcal{A}$ be a model[3] of* Th*, and φ as (5) be such that (a) for all q, $1 \leq q \leq k$, if $Q_q = \forall$ then h_{s_q} is surjective[4] and (b) for all negative literals $\neg P(t_1, \ldots, t_n)$ in φ and ground substitutions σ, if $(h(\sigma(t_1)), \ldots, h(\sigma(t_n))) \in P^{\mathcal{A}}$ then $(\sigma(t_1), \ldots, \sigma(t_n)) \in P^{\mathcal{I}_{\mathsf{Th}}}$. If $\mathcal{A} \models \neg\varphi$, then $\mathcal{I}_{\mathsf{Th}} \models \neg\varphi$.*

Remark 7 (Verification as satisfiability) *We can verify whether φ is a semantic property of a Horn theory* Th *by satisfiability as follows: (i) let $\overline{\varphi}$ be $\neg\varphi$; then (ii) find a structure $\mathcal{A}$ satisfying (a) and (b) with regard to $\overline{\varphi}$ and such that (iii) $\mathcal{A} \models \mathsf{Th} \cup \{\varphi\}$ holds (note that $\neg\overline{\varphi}$ and φ coincide). By Theorem 6, $\mathcal{I}_{\mathsf{Th}} \models \varphi$ holds. Since $\mathcal{I}_{\mathsf{Th}} \models \mathsf{Th}$ holds, φ is a semantic property of* Th *(Definition 3).*

Models $\mathcal{A}$ required in Theorem 6 can be obtained from $\mathsf{Th} \cup \{\varphi\}$ by using model generators like AGES[5] [Gutiérrez and Lucas, 2019a] or Mace4[6] [McCune, 2005 2010].

Example 8 *The question "*is d teaching someone?*" can be expressed as an ECBCA:*

$$(\exists y : \mathit{Student})\ \mathtt{teach}(\mathtt{d}, y) \tag{6}$$

With AGES*, we obtain a model of* $\mathsf{Teach} \cup \{\neg(6)\}$*, i.e., of*

$$\mathsf{Teach} \cup \{\neg(\exists y : \mathit{Student})\ \mathtt{teach}(\mathtt{d}, y)\} \tag{7}$$

[3] A (many-sorted) *structure* $\mathcal{A}$ is an interpretation of sorts s as sets $\mathcal{A}_s$ (the *carriers*), function symbols $f : s_1 \cdots s_k \to s$ as mappings $f^{\mathcal{A}} : \mathcal{A}_{s_1} \times \cdots \times \mathcal{A}_{s_k} \to \mathcal{A}_s$, and predicate symbols P as relations $P^{\mathcal{A}} \subseteq \mathcal{A}_{s_1} \times \cdots \times \mathcal{A}_{s_n}$. If $\mathcal{A}$ satisfies all formulas in a theory Th, we say that $\mathcal{A}$ is a *model* of Th.

[4] A mapping $f : A \to B$ is *surjective* if for all $b \in B$ there is $a \in A$ such that $f(a) = b$.

[5] http://zenon.dsic.upv.es/ages/

[6] https://www.cs.unm.edu/~mccune/prover9/

where $\mathcal{A}_{Student} = \{-1, 0, 1\}$ *and* $\mathcal{A}_{Teacher} = \mathbb{N} - \{0\}$. *For constants of sort Teacher,* $\mathsf{a}^{\mathcal{A}} = \mathsf{b}^{\mathcal{A}} = \mathsf{c}^{\mathcal{A}} = 1$, *and* $\mathsf{d}^{\mathcal{A}} = 6$*; for all constants of sort Student,* $\mathsf{p}^{\mathcal{A}} = \mathsf{q}^{\mathcal{A}} = \mathsf{r}^{\mathcal{A}} = 1$. *Predicate* teach *is interpreted by* $\mathtt{teach}^{\mathcal{A}}(x, y) \Leftrightarrow x \leq 3$. *By Theorem 6, (6) is* not *a semantic property of* Teach.

The treatment of condition (a) in Theorem 6 (surjectivity of h_s for a given sort s) is considered in Section 4. Regarding condition (b) (if negative literals $\neg P(t_1, \ldots, t_n)$ are used in φ), given a predicate $P : w$ where $w = s_1 \cdots s_n$ for sorts $s_1, \ldots, s_n$, let $N_{\mathsf{Th}}(P) = \mathcal{I}_w - P^{\mathcal{I}_{\mathsf{Th}}}$ be the tuples $(t_1, \ldots, t_n)$ (of terms $t_1, \ldots, t_n$ of sorts $s_1, \ldots, s_n$) which are obtained as the complement of the) interpretation of P in $\mathcal{I}_{\mathsf{Th}}$. Let $\mathcal{N}_{\mathsf{Th}}(P) = \{\neg P(t_1, \ldots, t_n) \mid (t_1, \ldots, t_n) \in N_{\mathsf{Th}}(P)\}$ be these tuples viewed as negative literals.

Remark 9 *In general,* $\mathcal{N}_{\mathsf{Th}}(P)$ *is infinite and incomputable. A* finite description *of* $\mathcal{N}_{\mathsf{Th}}(P)$ *can be given if the set* $\mathcal{T}_{\Sigma s_i}$ *of terms of sort* s_i *is finite for all* $1 \leq i \leq n$.

With $\mathsf{N}_{\mathsf{Th},\varphi} = \bigcup_{\neg P(\vec{t}) \text{ in } \varphi} \mathcal{N}_{\mathsf{Th}}(P)$, we recast Theorem 6:

Corollary 10 *Let* $\mathcal{A}$ *be a model of* $\mathsf{Th} \cup \mathsf{N}_{\mathsf{Th},\varphi}$ *such that for all* q, $1 \leq q \leq k$, *if* $Q_q = \forall$ *then* h_{s_q} *is surjective. If* $\mathcal{A} \models \neg\varphi$, *then* $\mathcal{I}_{\mathsf{Th}} \models \neg\varphi$.

Example 11 *According to Remark 7, we prove that (3) is a semantic property of* Teach *by first obtaining* $\neg(3)$, *i.e.,*

$$(\exists y : Student)(\forall x : Teacher)\ \neg\mathtt{teach}(x, y) \qquad (8)$$

and then applying Corollary 10 with $\varphi = (8)$. *Since*

$$N(\mathtt{teach}) = \{(t, s) \mid t \in \{\mathsf{a}, \mathsf{b}, \mathsf{c}, \mathsf{d}\}, s \in \{\mathsf{p}, \mathsf{q}, \mathsf{r}\}\} - \{(\mathsf{a}, \mathsf{p}), (\mathsf{a}, \mathsf{q}), (\mathsf{b}, \mathsf{q}), (\mathsf{c}, \mathsf{r})\}$$

we have to consider $\mathsf{N}_{\mathsf{Teach},\neg(3)} = \mathsf{N}_{\mathsf{Teach}(8)}$, *which is*

$\{\neg\mathtt{teach}(\mathsf{a}, \mathsf{r}), \neg\mathtt{teach}(\mathsf{b}, \mathsf{p}), \neg\mathtt{teach}(\mathsf{b}, \mathsf{r}), \neg\mathtt{teach}(\mathsf{c}, \mathsf{p}), \neg\mathtt{teach}(\mathsf{c}, \mathsf{q}), \neg\mathtt{teach}(\mathsf{d}, \mathsf{p}), \neg\mathtt{teach}(\mathsf{d}, \mathsf{q}), \neg\mathtt{teach}(\mathsf{d}, \mathsf{r})\}$

The proof finishes in Example 14 below.

4 Surjective Homomorphisms

In Theorem 6 and Corollary 10, ensuring *surjectivity* of $h_s : \mathcal{T}_{\Sigma s} \to \mathcal{A}_s$ is important to use them. We guarantee *surjectivity* of h_s by satisfiability of an appropriate *theory*. Given a set $T \subseteq \mathcal{T}_{\Sigma} s$ of ground terms of sort s, let

$$\mathsf{SuH}^T_s = (\forall x : s) \bigvee_{t \in T} x = t$$

Example 12 *For* $T = \{\mathsf{a}, \mathsf{b}, \mathsf{c}, \mathsf{d}\}$ *(containing all terms of sort Teacher),* $\mathsf{SuH}^T_{Teacher}$ *is*

$$(\forall x : Teacher)\ x = \mathsf{a} \vee x = \mathsf{b} \vee x = \mathsf{c} \vee x = \mathsf{d}$$

If $\mathcal{A} \models \mathsf{Th} \cup \{\mathsf{SuH}^T_s\}$ holds, then h_s is surjective. Expression SuH^T_s is first-order for *finite* sets T only. Moreover, not only the *number* but also the specific *shape* of terms in T can be important for SuH^T_s to work [Lucas, 2019, Remark 44]. Thus, a complementary approach is necessary, see [Lucas, 2019, Section 6.2] for details and examples. Overall, given a sentence φ, let H_{φ} and N_{φ} be appropriate (maybe empty) versions of SuH and $\mathsf{N}_{\mathsf{Th},\varphi}$ as discussed above to be used with φ. We finally recast Corollary 10 as follows:

Corollary 13 (Satisfiability criterion) *Let* Th *be a theory,* φ *be as (5), and* $\mathcal{A}$ *be a structure.*

If $\mathcal{A} \models \mathsf{Th} \cup \mathsf{H}_{\neg\varphi} \cup \mathsf{N}_{\neg\varphi} \cup \{\varphi\}$, *then* $\mathcal{I}_{\mathsf{Th}} \models \varphi$.

Example 14 *For* $\mathsf{SuH}^T_{Teacher}$ *in Example 12 and* $\mathsf{N}_{\mathsf{Teach},\neg(3)}$ *in Example 11, a model* $\mathcal{A}$ *of*

$$\mathsf{Teach} \cup \{\mathsf{SuH}^T_{Teacher}\} \cup \mathsf{N}_{\mathsf{Teach},\neg(3)} \cup \{(\forall y : Student)(\exists x : Teacher)\ \mathtt{teach}(x, y)\}$$

is obtained using Mace4[7] *where* $\mathcal{A}_{Teacher} = \mathcal{A}_{Student} = \{0, 1, 2, 3\}$*;* $\mathsf{a}^{\mathcal{A}} = \mathsf{p}^{\mathcal{A}} = 0$, $\mathsf{b}^{\mathcal{A}} = \mathsf{q}^{\mathcal{A}} = 1$, $\mathsf{c}^{\mathcal{A}} = \mathsf{r}^{\mathcal{A}} = 2$, *and* $\mathsf{d}^{\mathcal{A}} = 3$*; finally* $\mathtt{teach}^{\mathcal{A}} = \{(0, 0), (0, 1), (1, 1), (2, 2)\}$. *By Corollary 13, (3) is a semantic property of* Teach*, as expected.*

5 Refutation Witnesses

A semantic value b such that $\neg\varphi^{\mathcal{A}}(b)$ holds is a *counterexample* to $(\forall x)\ \varphi(x)$. However, it is often desirable to provide *terms* t such that $t^{\mathcal{A}} = b$ for a better understanding. Then, we say that $\neg\varphi(t)$ is a *refutation witness* [Lucas, 2019, Definition 53]. In order to obtain them, consider the negation of (5), i.e.,

$$(\overline{Q}_1 x_1 : s_1) \cdots (\overline{Q}_k x_k : s_k) \bigwedge_{i=1}^{m} \bigvee_{j=1}^{n_i} \neg L_{ij}(x_1, \ldots, x_k) \qquad (9)$$

where $\overline{Q}_i$ is $\forall$ whenever Q_i is $\exists$, and $\exists$ whenever Q_i is $\forall$.

Example 15 *The claim "*all students are taught by at least two teachers*" is expressed as follows:*

$$(\forall z : Student)(\exists x : Teacher)(\exists y : Teacher)\ \mathtt{teach}(x, z) \wedge \mathtt{teach}(y, z) \wedge \neg(x = y) \qquad (10)$$

Below, we prove that (10) is not *a semantic property of* Teach *and show a refutation witness. First, consider its negation*

$$(\exists z : Student)(\forall x : Teacher)(\forall y : Teacher)\ \neg\mathtt{teach}(x, z) \vee \neg\mathtt{teach}(y, z) \vee x = y \qquad (11)$$

in the normalized format (5).

Model generators often transform sentences into universally quantified formulas by *Skolemization* (see, e.g., [Kim and Zhang, 1994]). If (9) contains existential quantifiers, we remove them by introducing appropriate *Skolem function symbols* $sk : w \to s$ where w is a (possibly empty) sequence of sorts. Thus, our satisfiability techniques would be applied to the Skolemized version $(\neg\varphi)^{sk}$ of the negation of the target formula φ.

Example 16 *The Skolem normal form of (11) is*

$$(\forall x : Teacher)(\forall y : Teacher)\ \neg\mathtt{teach}(x, sk_z) \vee \neg\mathtt{teach}(y, sk_z) \vee x = y \qquad (12)$$

where sk_z *is a new Skolem constant of sort Student. Thus,* $(\neg(10))^{sk}$ *is (12). According to Corollary 13, we would prove that* (10) *is* not *a semantic property of* Teach *by obtaining a*

[7] For model generators, like Mace4, which do *not* support sorts, all sort information is removed before model generation, see (Sect. 5.2) for technical details.

model of $\mathsf{Teach} \cup \mathsf{H}_{(10)} \cup \mathsf{N}_{(10)} \cup \{\neg(10)\}$. *Using* (12) *instead of* $\neg(10)$, *we rather seek a model of*

$$\mathsf{Teach} \cup \mathsf{H}_{(10)} \cup \mathsf{N}_{(10)} \cup \{(12)\} \tag{13}$$

Since universally quantified variables in (10) are of sort $Student$, $\mathsf{H}_{(10)} = \mathsf{SuH}^{T}_{Student}$ *for* $T = \{\mathsf{p}, \mathsf{q}, \mathsf{r}\}$. *Since the only negative literal in (10) concerns the equality, we let*

$$\mathsf{N}_{(10)} = \{\mathsf{a} \neq \mathsf{b}, \mathsf{a} \neq \mathsf{c}, \mathsf{a} \neq \mathsf{d}, \mathsf{b} \neq \mathsf{c}, \mathsf{b} \neq \mathsf{d}, \mathsf{c} \neq \mathsf{d}\}$$

using $x \neq y$ *instead of* $\neg(x = y)$, *for short. A model* $\mathcal{A}$ *of (13) is obtained with* AGES, *where* $\mathcal{A}_{Student} = \{-1, 0, 1\}$ *and* $\mathcal{A}_{Teacher} = \mathbb{N} \cup \{-1\}$. *For constants of sort* $Teacher$, $\mathsf{a}^{\mathcal{A}} = 0$, $\mathsf{b}^{\mathcal{A}} = 1$, $\mathsf{c}^{\mathcal{A}} = -1$, *and* $\mathsf{d}^{\mathcal{A}} = 2$; *for constants of sort* $Student$, $\mathsf{p}^{\mathcal{A}} = 0$, $\mathsf{q}^{\mathcal{A}} = -1$, *and* $\mathsf{r}^{\mathcal{A}} = 1$. *Also,* $sk_z^{\mathcal{A}} = 1$, *and* $\mathsf{teach}^{\mathcal{A}}(x, y) \Leftrightarrow x + y \leq 0$.

Constant sk_z in Example 16 represents the existential quantification in (11) (i.e., in $(\neg 10)$). Value $sk_z^{\mathcal{A}} = 1$ represents the desired counterexample to (10). As discussed above, returning "1" is meaningless to the user, as it has to do with our proof-as-satisfiability approach. Returning "sk_z" is not helpful either, as symbol sk_z is not part of the original user-defined signature of the problem, being introduced during the model generation process. Our solution invoves the use of the *inverse* of h_s for sorts s with Skolem symbols $sk : w \to s$ to retrieve a term of sort s with the *same* interpretation as sk. For this purpose, (i) surjectivity of h_s is required (so that the existence of terms t of sort s such that $h_s(t) = sk^{\mathcal{A}}$ is guaranteed), and then (ii) we *choose* among terms t satisfying this condition. In this way, we obtain refutation witnesses built from symbols in the original signature only.

Example 17 *For (10), since* $sk_x^{\mathcal{A}} = 1$, *the only term* $t \in \mathcal{T}_{\Sigma Student}$ *such that* $t^{\mathcal{A}} = sk_x^{\mathcal{A}} = 1$ *is* r. *We obtain the following refutation witness:*

$$(\forall x, y : Teacher)\ \neg\mathtt{teach}(x, \mathsf{r}) \vee \neg\mathtt{teach}(y, \mathsf{r}) \vee x = y$$

which, in the following, maybe clearer, equivalent version

$$\neg(\exists x, y : Teacher)\ \mathtt{teach}(x, \mathsf{r}) \wedge \mathtt{teach}(y, \mathsf{r}) \wedge x \neq y$$

expresses that r *is taught by a single teacher.*

6 Related Work and Applications

In the following, we summarize some related work and applications concerning our approach.

Verification of concurrent systems. Lisitsa investigates reachability problems for transition systems represented as finite automata (with the possibility of using some arithmetic operators together with standard arithmetic properties encoded as equations) by using a logic-based approach similar to ours [Lisitsa, 2013]. Only monadic predicates are considered, though. Furthermore, only ECBCA sentences are used as semantic properties.

Protocol verification. Selinger shows how to encode cryptographic protocols as first-order formulas so that a proof of correctness of a given protocol can be pursued by just finding a model of a set of axioms representing properties of cryptographic properties together with a description of some particular protocol, and (the negation of) a formula Ψ which represents a violation of secrecy [Selinger, 2001]. Goubault-Larrecq shows that formally checkable proofs of correctness of protocols can be derived from the obtained models [Goubault-Larrecq, 2010]. He also investigates the specific interest (concerning decidability issues) of using *finite* models in protocol verification (he also shows that some protocols require infinite models). In these two papers, the considered formulas Ψ are ECBCAs. Jürgens and Weber use a similar technique, where theories are restricted to *limit sentences*, which are universally quantified implications where some variables in the consequent can be quantified with the *uniqueness quantifier* $\exists!$ instead [Jürjens and Weber, 2009]. Sentences to be refuted (called *conjectures*) must be universally quantified conjunctions of atoms $(\forall \vec{x}) \bigwedge_{i=1}^{n} A_i$. They consider finite models only, although no strong technical reason is apparently given to justify this restriction (which is not imposed in [Selinger, 2001], for instance).

Proof by consistency. The idea of proving logic formulas φ with respect to a theory Th by showing that $\mathsf{Th} \cup \{\varphi\}$ is not contradictory is called *proof by consistency* in [Kapur and Musser, 1987]. The authors point that this proof method is, in general, unsound, but can be faithfully used with *strongly complete* proof systems, which are those guaranteeing that, whenever a formula φ can be added as an axiom to the proof system without introducing inconsistencies, then φ is also a theorem of the system. Kapur and Musser focus on many-sorted *equational* proof systems consisting of a subset C of ground terms and a set of equations E where equational logic is used as proof system. Their proof by consistency method is used to prove equations (with implicit universal quantification) with respect to the considered systems. Furthermore, additional requirements like *unambiguity* are required on the considered systems (i.e., theories, see [Kapur and Musser, 1987, Theorem 9.2], for instance. Moreover, Kapur and Musser do not use satisfiability to show consistency of the theory (although both notions are equivalent) Instead, proof-theoretic (rewriting-based) methods are used to prove consistency.

Feasibility framework. The framework in [Gutiérrez and Lucas, 2019b] has been recently extended to automatically prove and disprove ECBCAs (as *feasibility goals*). The system infChecker[8] provides a first implementation of the framework which has been shown successful in the *infeasibility competition* of the 2019 Confluence Competition [Middeldorp *et al.*, 2019, Section 5.2] for Conditional Term Rewriting Systems. By *infeasibility* it is meant the absence of a substitution σ which makes a boolean combination $\bigvee_{i=1}^{m} \bigwedge_{j=1}^{n_i} A_{ij}$ of atoms A_{ij} provable after instantiation with σ. As shown in [Lucas, 2020], infeasibility of first-order formulas is also useful in proofs of *operational termination* of programs in general logics [Meseguer, 1989].

[8] http://zenon.dsic.upv.es/infChecker/

References

[Clark, 1977] Keith L. Clark. Negation as failure. In Hervé Gallaire and Jack Minker, editors, *Logic and Data Bases, Symposium on Logic and Data Bases, Centre d'études et de recherches de Toulouse, France, 1977*, Advances in Data Base Theory, pages 293–322, New York, 1977. Plemum Press.

[Clark, 1980] Keith L. Clark. *Predicate logic as a computational formalism*. PhD thesis, Queen Mary University of London, UK, 1980.

[Fitting, 1996] Melvin Fitting. *First-Order Logic and Automated Theorem Proving, Second Edition*. Graduate Texts in Computer Science. Springer, 1996.

[Goguen and Meseguer, 1987] Joseph A. Goguen and José Meseguer. Models and equality for logical programming. In Hartmut Ehrig, Robert A. Kowalski, Giorgio Levi, and Ugo Montanari, editors, *TAPSOFT'87: Proceedings of the International Joint Conference on Theory and Practice of Software Development, Pisa, Italy, March 23-27, 1987, Volume 2: Advanced Seminar on Foundations of Innovative Software Development II and Colloquium on Functional and Logic Programming and Specifications (CFLP)*, volume 250 of *Lecture Notes in Computer Science*, pages 1–22. Springer, 1987.

[Goubault-Larrecq, 2010] Jean Goubault-Larrecq. Finite models for formal security proofs. *Journal of Computer Security*, 18(6):1247–1299, 2010.

[Green and Raphael, 1968] C. Cordell Green and Bertram Raphael. The use of theorem-proving techniques in question-answering systems. In *Proceedings of the 1968 23rd ACM National Conference*, ACM '68, page 169–181, New York, NY, USA, 1968. Association for Computing Machinery.

[Gutiérrez and Lucas, 2019a] Raúl Gutiérrez and Salvador Lucas. Automatic generation of logical models with AGES. In Pascal Fontaine, editor, *Automated Deduction - CADE 27 - 27th International Conference on Automated Deduction, Natal, Brazil, August 27-30, 2019, Proceedings*, volume 11716 of *Lecture Notes in Computer Science*, pages 287–299. Springer, 2019.

[Gutiérrez and Lucas, 2019b] Raúl Gutiérrez and Salvador Lucas. infChecker, a tool for Checking Infeasibility . In María Alpuente, Julia Sapiña, and Roberto Rodríguez-Echeverría, editors, *Actas de las XIX Jornadas de Programación y Lenguajes (PROLE 2019)*, page 12. SISTEDES, 2019.

[Hodges, 1993] Wilfrid Hodges. *Model theory*, volume 42 of *Encyclopedia of mathematics and its applications*. Cambridge University Press, 1993.

[Jürjens and Weber, 2009] Jan Jürjens and Tjark Weber. Finite models in fol-based crypto-protocol verification. In Pierpaolo Degano and Luca Viganò, editors, *Foundations and Applications of Security Analysis, Joint Workshop on Automated Reasoning for Security Protocol Analysis and Issues in the Theory of Security, ARSPA-WITS 2009, York, UK, March 28-29, 2009, Revised Selected Papers*, volume 5511 of *Lecture Notes in Computer Science*, pages 155–172. Springer, 2009.

[Kapur and Musser, 1987] Deepak Kapur and David R. Musser. Proof by consistency. *Artif. Intell.*, 31(2):125–157, 1987.

[Kim and Zhang, 1994] Sun Kim and Hantao Zhang. Modgen: Theorem proving by model generation. In Barbara Hayes-Roth and Richard E. Korf, editors, *Proceedings of the 12th National Conference on Artificial Intelligence, Seattle, WA, USA, July 31 - August 4, 1994, Volume 1*, pages 162–167. AAAI Press / The MIT Press, 1994.

[Lisitsa, 2013] Alexei Lisitsa. Finite reasons for safety - parameterized verification by finite model finding. *J. Autom. Reasoning*, 51(4):431–451, 2013.

[Lucas, 2019] Salvador Lucas. Proving semantic properties as first-order satisfiability. *Artif. Intell.*, 277, 2019.

[Lucas, 2020] Salvador Lucas. Using well-founded relations for proving operational termination. *J. Autom. Reasoning*, 64(2):167–195, 2020.

[McCune, 2005 2010] William McCune. Prover9 & Mace4. Technical report, 2005–2010.

[Mendelson, 1997] Elliott Mendelson. *Introduction to mathematical logic (4. ed.)*. Chapman and Hall, 1997.

[Meseguer, 1989] José Meseguer. General logics. In H.-D. Ebbinghaus, J. Fernandez-Prida, M. Garrido, D. Lascar, and M. Rodriquez Artalejo, editors, *Logic Colloquium'87*, volume 129 of *Studies in Logic and the Foundations of Mathematics*, pages 275 – 329. Elsevier, 1989.

[Middeldorp *et al.*, 2019] A. Middeldorp, J. Nagele, and K. Shintani. Confluence Competition 2019. In D. Beyer, M. Huisman, F. Kordon, and B. Steffen, editors, *Proc. of Tools and Algorithms for the Construction and Analysis of Systems, TACAS'19*, pages 25–40. Springer, 2019.

[Nicolas and Gallaire, 1977] Jean-Marie Nicolas and Hervé Gallaire. Data base: Theory vs. interpretation. In Hervé Gallaire and Jack Minker, editors, *Logic and Data Bases, Symposium on Logic and Data Bases, Centre d'études et de recherches de Toulouse, France, 1977*, Advances in Data Base Theory, pages 33–54, New York, 1977. Plemum Press.

[Reiter, 1977] Raymond Reiter. On closed world data bases. In Hervé Gallaire and Jack Minker, editors, *Logic and Data Bases, Symposium on Logic and Data Bases, Centre d'études et de recherches de Toulouse, France, 1977*, Advances in Data Base Theory, pages 55–76, New York, 1977. Plemum Press.

[Selinger, 2001] Peter Selinger. Models for an adversary-centric protocol logic. *Electron. Notes Theor. Comput. Sci.*, 55(1):69–84, 2001.

[van Emden and Kowalski, 1976] Maarten H. van Emden and Robert A. Kowalski. The semantics of predicate logic as a programming language. *J. ACM*, 23(4):733–742, 1976.