The Data Complexity of the Syllogistic Fragments of English

Camilo Thorne and Diego Calvanese

KRDB Research Centre
Free University of Bozen-Bolzano
3 Piazza Domenicani, 39100, Bolzano, Italy
{cthorne,calvanese}inf.unibz.it

Abstract. Pratt and Third's syllogistic fragments of English can be used to capture, in addition to syllogistic reasoning, many other kinds of common sense reasoning, and, in particular *(i)* knowledge base consistency and *(ii)* knowledge base query answering, modulo their FO semantic representations. We show how difficult, in terms of semantic (computational) complexity and data complexity (i.e., computational complexity w.r.t. the number of instances declared in a knowledge base), such reasoning problems are. In doing so, we pinpoint also those fragments for which the reasoning problems are tractable (in **PTime**) or intractable (**NP**-hard or **coNP**-hard).

1 Introduction

Natural logic is concerned with capturing formal logic and common sense reasoning, by means of natural language insofar as we can ascribe to the latter a formal, compositional semantics. Montague, back in the 1970's [8] showed how to define a compositional, formal semantics for fragments of English by means of *compositional translations* $\tau(\cdot)$ that recursively assign to each English syntactic constituent a HO or FO *meaning representation*[1], and that gives rise to the notion of *semantic complexity* (see Pratt in [11,10]), viz., the computational properties of those meaning representations, which help in measuring the ease or the difficulty of the cognitive and/or computational processing involved in natural language understanding and reasoning [15,11,10,7]. Easy fragments are believed to exhibit "good" (tractable, i.e., in **PTime**) computational properties and difficult ones "bad" (intractable, i.e., **NP**-hard or **coNP**-hard) properties [15,11,10,7].

In this paper we study the *syllogistic fragments* (FOEs) of Pratt [11,10], who proposed them as a means of capturing in English syllogistic entailment and satisfiability. However, entailment and satisfiability are not the only relevant reasoning problems a fragment, let alone the FOEs, that is not "booleanly closed" (i.e., cannot express complete sets of boolean functions) captures. Instead, the common-sense reasoning problems we study in this paper are *(i)* identifying inconsistencies in known information and *(ii)* posing questions to known information. Such reasoning problems can be modelled in logic quite naturally via ontology and knowledge base reasoning tasks, resp., by the

[1] HO can be conceived of as the extension of FO with the λ-abstraction, λ-application, β-normalization and, eventually, the types of the simply-typed λ-calculus.

Table 1. Coverage of the FOEs and of TSQs. See [11,10,16]

COP	Copula, common and proper nouns, negation, universal, existential quantifiers
COP+Rel	COP plus relative pronouns
COP+TV	COP plus transitive verbs
COP+TV+DTV	COP+TV plus ditransitive verbs
COP+Rel+TV	COP+Rel plus transitive verbs
COP+Rel+TV+DTV	COP+Rel+TV plus ditransitive verbs
TSQs	Copula, common and proper nouns, existential quantifiers, transitive verbs, noun and verb phrase coordination, relative pronouns, passives, query words

knowledge base satisfiability problem and the query answering problem. An ontology is a set $\mathcal{O}$ of FO axioms and a knowledge base is a pair $(\mathcal{O}, \mathcal{D})$, where $\mathcal{O}$ is an ontology and $\mathcal{D}$ is a database containing data about a given domain of interest [14,4]. In this context it is assumed that the size of the data dominates that of the ontology and hence the main focus is in the *data complexity* of reasoning, i.e., in inferring computational properties w.r.t. the size of $\mathcal{D}$ alone [17,4].

The FOEs capture problems *(i)* and *(ii)* when combined with suitable interrogative fragments such as *tree shaped questions* (TSQs) [16], an interrogative fragment with which a wide variety of information requests can be expressed. Since $\tau(\cdot)$ allows us to assign to the FOEs knowledge bases as meaning representations, by combining the FOEs and TSQs and studying their data complexity, we can infer for which FOEs data complexity is tractable or intractable. Or, more precisely, which combinations of function words in the fragments give rise to such computational properties.

To infer data complexity bounds we adopt as main strategy *resolution-based saturation decision procedures* for fragments of FO as outlined by Joyner in [6]. We show that, in general, intractability w.r.t. data arises whenever fragments, alone or in combination with TSQs, are "booleanly closed". Otherwise, we obtain fragments where reasoning is tractable w.r.t. data.

2 The Syllogistic Fragments and Tree-Shaped Questions

The FOEs are defined incrementally. The idea is to start with a FOE, called COP, that covers: *(i)* copula ("is"), *(ii)* verb-phrase negation ("is not"), *(ii)* the determiners "some", "every" and "no", together with common and proper nouns. The fragment and the translation $\tau(\cdot)$ are defined at the same time, by means of a semantically annotated context-free grammar. Standard HO meaning representations are used. Thereafter, by extending coverage to a new English construct, viz., transitive verbs (e.g., "likes"), ditransitive verbs (e.g., "gives"), relatives (e.g., "that") and anaphors (e.g., "him"), the other members of the family are defined. See Table 1. For the detailed definition of the fragments, we refer to [11,10]. See Table 2 for their meaning representations.

Table 2. The meaning representations generated by the FOEs and TSQs: φ_l (resp. φ_r) stands for the meaning representation of the subject (resp. predicate) constituent of a main or subordinated sentence, whereas φ_{tv} (resp. φ_{dtv}) denotes the meaning representation of a verb pharse containing a transitive (resp. ditransitive) verb; $\psi(x, y)$ (resp. $\chi(x, y, z)$) stands for some binary (resp. ternary) atom, while $\pm$ means that a formula may or may not be negated

COP	$\varphi_l(x) \to A(x)$	$\forall x(\varphi_l(x) \Rightarrow \pm\varphi_r(x))$	No student failed.
	$\varphi_r(x) \to \pm\varphi_l(x)$	$\exists x(\varphi_l(x) \land \varphi_r(x))$	A student failed.
COP+TV	$\varphi_l(x) \to A(x)$	$\forall x(\varphi_l(x) \Rightarrow \pm\varphi_r(x))$	No student failed.
	$\varphi_r(x) \to \pm\varphi_l(x) \mid \forall y(A(x) \Rightarrow \pm\psi(x,y))$	$\exists x(\varphi_l(x) \land \varphi_r(x))$	Some student follows
	$\mid \exists y(A(x) \land \psi(x,y))$		every course.
COP+TV+DTV	$\varphi_l(x) \to A(x)$	$\forall x(\varphi_l(x) \Rightarrow \pm\varphi_r(x))$	Every student
	$\varphi_{tv}(x) \to \pm\varphi_l(x) \mid \forall y(A(x) \Rightarrow \pm\psi(x,y))$		gives no credit
	$\mid \exists y(A(x) \land \psi(x,y))$		to some student.
	$\varphi_{dtv}(x,y) \to \forall z(A(x) \Rightarrow \pm\chi(x,y,z))$	$\exists x(\varphi_l(x) \land \varphi_r(x))$	A student
	$\mid \exists z(A(x) \land \chi(x,y,z))$		borrowed a book
	$\varphi_r(x) \to \varphi_{tv}(x) \mid \forall y(A(x) \Rightarrow \pm\varphi_{dtv}(x,y))$		from some library.
	$\mid \exists y(A(x) \land \varphi_{dtv}(x,y))$		
COP+Rel	$\varphi_l(x) \to A(x) \mid \pm\varphi_l(x) \land \pm\varphi_l(x)$	$\forall x(\pm\varphi_l(x) \Rightarrow \pm\varphi_r(x))$	Every student who is not
	$\varphi_r(x) \to \varphi_l(x)$	$\exists x(\pm\varphi_l(x) \land \pm\varphi_r(x))$	dum is smart.
COP+TV+Rel	$\varphi_l(x) \to A(x)$	$\forall x(\varphi_l(x) \Rightarrow \pm\varphi_r(x))$	No student failed.
	$\varphi_r(x) \to \pm\varphi_l(x) \mid \forall y(A(x) \Rightarrow \pm\psi(x,y))$	$\exists x(\varphi_l(x) \land \varphi_r(x))$	Some student studies
	$\mid \exists y(A(x) \land \psi(x,y))$		every course.
COP+Rel+TV+DTV	$\varphi_l(x) \to A(x) \mid \pm\varphi_r \land \pm\varphi_r$	$\forall x(\varphi_l(x) \Rightarrow \pm\varphi_r(x))$	Every helpful student
	$\varphi_{tv}(x) \to \pm\varphi_l(x) \mid \forall y(A(x) \Rightarrow \pm\psi(x,y))$		gives some aid
	$\mid \exists y(A(x) \land \psi(x,y))$		to some student.
	$\varphi_{dtv}(x,y) \to \forall z(A(x) \Rightarrow \pm\chi(x,y,z))$	$\exists x(\varphi_l(x) \land \varphi_r(x))$	Some diligent student
	$\mid \exists z(A(x) \land \chi(x,y,z))$		borrowed every book
	$\varphi_r(x) \to \varphi_{tv}(x) \mid \forall y(A(x) \Rightarrow \pm\varphi_{dtv}(x,y))$		from every library.
	$\mid \exists y(A(x) \land \varphi_{dtv}(x,y))$		
TSQs	$\varphi_l(x) \to A(x) \mid \exists y R(x,y) \mid \varphi_r(x) \land \varphi_r(x)$	$\varphi_l(x) \land \varphi_r(x)$	Which student who attends
	$\mid \exists y(R(x,y) \land \varphi_r(y))$		some course is diligent?
	$\varphi_r(x) \to \varphi_l(x)$		

The information that we can express/store in the FOEs can be queried/accessed by TSQs[2], which are built through query words (e.g., "who"), relatives, transitive verbs, copula, common nouns, the determiner "some", the pronoun "somebody", passives (e.g., "is loved by") and conjunction ("and"). See Table 1. For their formal definition we refer to [16]. See Table 2 for their meaning representations.

We intend to understand the computational properties of the FOEs *in the size of the data*. We consider sets $\mathcal{S}$ of quantified and $\mathcal{F}$ of ground sentences. We overload the notion of knowledge base to cover linguistic knowledge bases, i.e., pairs $(\mathcal{S}, \mathcal{F})$: since modulo $\tau(\cdot)$, $\mathcal{S}$ maps into ("expresses") an ontology $\mathcal{O}$ and $\mathcal{F}$ into ("expresses") a database $\mathcal{D}$, reasoning over $(\mathcal{S}, \mathcal{F})$ reduces to reasoning over $(\mathcal{O}, \mathcal{D})$.

We study two decision problems. On the one hand, *(i)* knowledge base satisfiability (KB-SAT): **Given:** $(\mathcal{S}, \mathcal{F})$. **Check:** is $\tau(\mathcal{S}) \cup \tau(\mathcal{F})$ satisfiable? And, on the other hand, *(ii)* query answering (KB-QA): **Given:** $(\mathcal{S}, \mathcal{F})$, a question Q and (possibly) a constant c. **Check:** does $\tau(\mathcal{S}) \cup \tau(\mathcal{F}) \models \tau(Q)\{x \mapsto c\}$? Where $\tau(Q)$ is a formula of (possibly) free variable x. By analogy to [17], we define the *data complexity* of KB-SAT and KB-QA as their computational complexity measured in the size of $\mathcal{F}$ only ($\mathcal{S}$ and possibly $\mathcal{F}$ are considered constant). The *size* $\#(\mathcal{F})$ of $\mathcal{F}$ is defined as the number of proper names occurring in $\mathcal{F}$. This measure can be extended to $\tau(\mathcal{F})$ by denoting with $\#(\tau(\mathcal{F}))$ the number of individual constants in $\tau(\mathcal{F})$.

Note that as soon as we add relative pronouns, the FOEs become closed under negation (see Table 2). Note also that TSQs are strictly contained in the FOEs with transitive verbs and relatives (see Table 2). However, corpora analysis [2] show that while negation and universal quantification are rare in questions, relatives and existential quantifiers are common and that TSQs are a fragment that naturally expresses queries to knowledge bases.

Example 1. Suppose we want to reason about our (previous) knowledge regarding university students. We can capture such knowledge via a COP+TV knowledge base. Moreover, we can check whether such knowledge is consistent and, thus, meaningful, and pose to it TSQs:

$$\mathcal{S}_s := \begin{cases} \text{Every student attends some course.} & \forall x(\mathit{Student}(x) \Rightarrow \exists y(\mathit{attends}(x,y) \wedge \mathit{Course}(y))) \\ \text{Every bachelor student is a student.} & \forall x(\mathit{BachelorStudent}(x) \Rightarrow \mathit{Student}(x)) \end{cases}$$

$$\mathcal{F}_s := \begin{cases} \text{John is a bachelor student.} & \mathit{BachelorStudent}(\text{John}) \\ \vdots & \vdots \end{cases}$$

$$Q_s := \{ \text{Which student attends some course?} \quad \mathit{Student}(x) \wedge \exists y(\mathit{attends}(x,y) \wedge \mathit{Course}(y))$$

Clearly, "John" is the answer to our question, and the knowledge base is consistent. But how difficult are these forms of reasoning over, ultimately, $\tau(\mathcal{S}_s) \cup \tau(\mathcal{F}_s)$ and $\tau(Q_s)$ (viz., KB-SAT and KB-QA) in computational terms?

3 Resolution Saturations and Data Complexity

Resolution decision procedures. A *term* t is *(i)* a variable x or a constant c or *(ii)* an expression $f(t_1, \ldots, t_n)$ where f is a function symbol and $t_1, \ldots, t_n$ are terms. In the

[2] TSQs express a significant subset of database SQL `SELECT-PROJECT-JOIN` queries [1].

latter case, we speak about *function terms*. A *literal* L is a FO atom $P(t_1, \ldots, t_n)$. By a *clause* we understand a disjunction $L_1 \vee \cdots \vee L_n \vee \overline{N}_{n+1} \vee \cdots \vee \overline{N}_{n+m}$ of positive and negative literals. The *empty* clause or *falsum* is denoted $\perp$. By $V(t)$, $V(L)$ and $V(C)$ we denote the sets of variables of, resp., term t, literal L and clause C. A term, literal, clause or set of clauses is said to be *ground* if it contains no free variables. A *substitution* σ is a function from variables to terms. It is called a *renaming* when it is a function from variables to variables. A *unifier* is a substitution σ s.t., given two terms t and t', $t\sigma = t'\sigma$. A *most general unifier* is a unifier σ s.t. for every other unifier σ' there exists a renaming σ'' with $\sigma' = \sigma\sigma''$.

The *depth* of a term is defined by *(i)* $d(x) := d(c) := 0$ and *(ii)* $d(f(t_1, \ldots, t_n)) := \max\{d(t_i) \mid i \in [1, n]\} + 1$. The *depth* $d(L)$ of a literal L or $d(\Gamma)$ of a set of clauses Γ is the maximal depth of their terms. The *relative depth* of a variable x in a term is defined by *(i)* $d(x, y) := d(x, c) := 0$ and *(ii)* $d(x, f(t_1, \ldots, t_n)) := \max\{d(x, t_i) \mid i \in [1, n]\} + 1$. The *relative depth* $d(x, L)$ of a variable x in a literal L is its maximal relative depth among L's terms.

We consider the so-called *saturation-based* version of the resolution calculus in which we iteratively (monotonically w.r.t. $\subseteq$) generate the set of all possible clauses derived from Γ using the rules

$$\textit{res}\,\frac{C \vee \overline{L} \qquad C \vee L'}{(C \vee C')\sigma} \quad \text{and} \quad \textit{fact}\,\frac{C \vee L \vee L'}{(C \vee L)\sigma},$$

where σ is a most general unifier of L and L', until either *(i)* $\perp$ is derived or *(ii)* all possible clauses are generated (fixpoint computation). Formally, consider a function $\rho(\cdot)$ over sets of clauses, defined in terms of *res* and *fact*. A *resolution calculus* is a function $\mathcal{R}(\cdot)$ s.t. $\mathcal{R}(\Gamma) := \Gamma \cup \rho(\Gamma)$. A *derivation* δ from Γ is defined by putting *(i)* $\mathcal{R}^0(\Gamma) := \Gamma$ and $\mathcal{R}^{i+1}(\Gamma) := \mathcal{R}(\mathcal{R}^i(\Gamma))$, for $i > 0$. Thereafter the *saturation* of Γ is defined as $\Gamma^\infty := \bigcup\{\mathcal{R}^i(\Gamma) \mid i \geq 0\}$. The positive integer i is called the *depth* or *rank* of δ. The set(s) of clauses derived at each rank $i \geq 0$ of δ is (are) called the *state(s)* of δ. The *size* of δ is defined as its total number of states. Resolution is sound and complete w.r.t. (un)satisfiability: Γ is unsatisfiable iff $\perp \in \Gamma^\infty$. Moreover, if Γ is satisfiable, we can build out of Γ^∞ a herbrand model of Γ [5].

Resolution saturations are not computable in general (they may not converge finitely). However, Joyner in [6] showed that finite convergence can be achieved provided that two conditions are met: *(i)* that the depth of literals does not grow beyond a certain bound $d \geq 0$ and *(ii)* that the length of clauses (the number of disjunctions) does not grow beyond a bound $l \geq 0$. Several *refinements* can be used to ensure the existence of such bounds and a fortiori finite convergence for several fragments of FO.

To control depth, *acceptable orderings* (A-orderings), that is, well-founded and substitution-invariant partial orders on clause literals and sets thereof, can be used (which force resolution on literals that are maximal w.r.t. the ordering). The best known is the $\prec_d$ ordering defined by

$$\begin{aligned} L \prec_d L' \ \text{ iff } \ & d(L) < d(L'), V(L) \subseteq V(L') \\ & \text{and, for all } x \in V(L), d(x, L) < d(x, L), \end{aligned}$$

a refinement sound and complete w.r.t. satisfiability. To control length the splitting rule

$$\textit{split}\ \frac{C\vee L\vee L' \qquad \begin{array}{c} C\vee L \\ \vdots \\ C'\sigma\end{array} \qquad \begin{array}{c} C\vee L' \\ \vdots \\ C'\sigma\end{array}}{C'\sigma}\ (V(L)\cap V(L')=\emptyset)$$

can be used (it is sound and complete w.r.t. satisfiability). These refinements are guaranteed to work the way we want them to in case they are applied to covering clauses. A literal L is said to be *covering* whenever *(i)* $d(L) = 0$ or *(ii)* for every functional term t in L, $V(t) = V(L)$. If all the literals of a clause C are covering, so is C. This property is not, however, closed under resolution or its refinements: applying them to covering clauses may result in non-covering clauses. To prevent this from happening, a further refinement is required: *monadization* [6]. Intuitively, what this does is to reduce the (un)satisfiability of non-covering clauses, satisfying some structural properties, into that of a set of covering clauses. The applicability of the refinements thus depends on the FO fragments such clauses are drawn from, but, whenever *all* are applicable, saturations finitely converge [5].

The different systems arising from the different combinations of rules, orderings and refinements are summarized below:

		split	*mon*	*split* *mon*
	$\mathcal{R}_{1,1}$	$\mathcal{R}_{1,2}$	$\mathcal{R}_{1,4}$	$\mathcal{R}_{1,5}$
$\prec_d$	$\mathcal{R}_{2,1}$	$\mathcal{R}_{2,2}$	$\mathcal{R}_{2,4}$	$\mathcal{R}_{2,5}$

In particular, the $\mathcal{R}_{2,5}$ calculus decides the $\mathcal{S}^+$ class of clauses [5]. The class $\mathcal{S}^+$ is the class where every clause C satisfies: *(i)* $V(C) = V(t)$, for every functional term t in C, and *(ii)* either L has at most one variable or $V(L) = V(C)$, for every literal L in C. Note that saturations exhibit the shape of a tree (of branching factor 2) or of a sequence, depending on whether the calculi make use or not of the splitting rule.

Data Complexity of KB-QA and KB-SAT. In this section we study the data complexity of KB-SAT and KB-QA by applying resolution decision procedures to the FOEs. We apply data complexity arguments to sets $\Sigma \cup \Delta$ of non-ground and ground clauses. This makes sense, because, modulo $\tau(\cdot)$ and clausification, FOE sentences $\mathcal{S}$ map to sets Σ of non-ground clauses, FOE facts $\mathcal{F}$ map to sets Δ of ground clauses, and, in general, knowledge bases $(\mathcal{S}, \mathcal{F})$ to sets $\Sigma \cup \Delta$ of clauses.

We do as follows. For the tractable FOEs we rely on the "separation" property of resolution saturations [5] (resolution of ground clauses can be delayed to the end). For the intractable, on the "monadic reducibility" property [10] that enforces a reduction to $\mathcal{S}^+$ clauses for the fragments involved; this we combine with a data complexity of the $\mathcal{S}^+$ class (and saturations).

– **Separation:** $\bot \in (\Sigma \cup \Delta)^\infty$ iff there exists a set $\Sigma' \subseteq \Sigma^\infty$ s.t. *(i)* $d(\Sigma') \leq d(\Delta)$, *(ii)* $\bot \in (\Sigma' \cup \Delta)^\infty$ and *(iii)* Σ' is finite.

- **Monadic reducibility:** every set Γ of COP+TV+DTV+Rel clausified meaning reprentations (or any fragment thereof) can be polynomially (in the size of Γ) transformed into a set Γ_u of unary clauses s.t. Γ is satisfiable iff Γ_u is satisfiable.

Lemma 1. *Let $(\mathbf{C}, \mathbf{F}, \mathbf{R})$ be a finite FO signature, where $\mathbf{C}$ is a (finite) set of constants, $\mathbf{F}$ a (finite) set of function symbols and $\mathbf{R}$ a (finite) set of predicate symbols. Consider a clause set Γ over such a signature and suppose that there exist both a term depth bound $d \geq 0$ and a clause length bound $k \geq 0$. Then, in the worst case,*

1. *the number of clauses derivable by the saturation is (i) exponential in the number of constants in $\mathbf{C}$ if we use the splitting rule or (ii) polynomial in the number of constants in $\mathbf{C}$ otherwise, and*
2. *the depth of the saturation is polynomial in the number of constants in $\mathbf{C}$.*

Proof. Assume that a depth bound d and a length bound l exist. Let c be the number of constant symbols in $\mathbf{C}$, v the number of variables in $\mathbf{V}$, f the number of function symbols in $\mathbf{F}$, p the number of predicate symbols in $\mathbf{R}$, ar_f the maximum arity of the function symbols, and ar_p the maximum arity of the predicate symbols. We can define the number te_i of terms of depth $i \geq 0$ inductively by setting *(i)* $te_0 := v + c$, *(ii)* $te_{i+1} := f \cdot te_n^{ar_f}$. Thus, the number *te* of terms of depth $\leq d$ is

$$te \leq \sum_{i=0}^{d} te_i = f^0 \cdot (v+c)^{ar_f^0} + ... + f^d \cdot (v+c)^{ar_f^d} := \mathsf{p}_{\mathsf{te}}(c) \tag{1}$$

which defines a polynomial $\mathsf{p}_{\mathsf{te}}(c)$. This in its turn yields as upper bound to the number *li* of positive and negative literals

$$li \leq 2 \cdot p \cdot te^{ar_p} = 2 \cdot p \cdot \mathsf{p}_{\mathsf{te}}(c)^{ar_p} := \mathsf{p}_{\mathsf{li}}(c) \tag{2}$$

thus defining a polynomial $\mathsf{p}_{\mathsf{li}}(c)$. Finally, from *li* we derive an upper bound to the number *cl* of clauses of length $\leq l$

$$cl \leq li^l = \mathsf{p}_{\mathsf{li}}(c)^l := \mathsf{p}_{\mathsf{cl}}(c) \tag{3}$$

which again defines a polynomial $\mathsf{p}_{\mathsf{cl}}(c)$. The splitting rule splits saturations into two, yielding a (saturation) tree of worst-case size $\leq 2^{\mathsf{p}_{\mathsf{cl}}(c)}$, largest (derived) state of size $\leq \mathsf{p}_{\mathsf{cl}}(c)$ and that will converge after $\leq \mathsf{p}_{\mathsf{cl}}(c)$ iterations. □

Theorem 1. KB-SAT *is in* **NP** *in data complexity for $\mathcal{S}^+$.*

Proof. Let $\Sigma \cup \Delta$ be a set of $\mathcal{S}^+$ clauses. Consider now a $\mathcal{R}_{2,5}$-saturation. Calculus $\mathcal{R}_{2,5}$ decides $\mathcal{S}^+$ and saturations finitely converge. Assume w.l.o.g. that Σ contains no constants and that Δ is of depth $d(\Delta) = 0$ and has c distinct constants (where $c \geq 0$). By Lemma 1 we know that the saturation will be tree-shaped, of rank $\leq \mathsf{p}(c)$, of size $\leq 2^{\mathsf{p}(c)}$ and of maximal state of size $\leq \mathsf{p}(c)$.

Outline a non-deterministic algorithm for KB-SAT as follows. Start with $\Sigma \cup \Delta$. For each rank $i \in [0, \mathsf{p}(c)]$ of the saturation, guess/choose a state $j \in [0, 2^i]$. Notice that the algorithm will make polynomially many choices on c. Finally, check, in time polynomial in c whether $\perp$ is in the resulting state, and, if no, compute, in time polynomial in c, a herbrand model of $\Sigma \cup \Delta$. □

Theorem 2 (KB-SAT). *The data complexity for* KB-SAT *is*

1. *in* **LSpace** *for COP, COP+TV, COP+TV+DTV and COP+Rel, and*
2. **NP**-*complete for COP+Rel+TV, COP+Rel+TV and COP+Rel+TV+DTV.*

Proof. (Sketch.) For COP, COP+TV and COP+TV+DTV we reason as follows. Let $(\mathcal{S}, \mathcal{F})$ be a knowledge base and consider its meaning representations $\tau(\mathcal{S})$ and $\tau(\mathcal{F})$ (which can be computed in space logarithmic in $\#(\mathcal{F})$). Computing their skolemization and clausification does not affect data complexity, since it is the identity for $\tau(\mathcal{F})$. By inspecting the resulting clauses we can observe that they are covering: using A-ordered resolution prevents clause depth from growing beyond a bound d. Furthermore, it can be proven that applying *res* and *fact*, does not increase clause length beyond a bound l, nor does it result in non-covering clauses. Thus, the A-ordered resolution calculi without splitting decide the satisfiability of $\tau(\mathcal{S}) \cup \tau(\mathcal{F})$. We also know by the "separation" property that we can "separate" data from facts provided $\tau(\mathcal{S})$ is satisfiable.

Sketch a decision algorithm for KB-SAT as follows. Check whether $\tau(\mathcal{S})$ is satisfiable, i.e., whether $\perp \in \tau(\mathcal{S})^{\infty}$, computation that does not depend on $\#(\mathcal{F})$ (or $\#(\tau(\mathcal{F}))$). If the answer is negative, return "no". If the answer is positive: *(i)* Compute a (finite) model/database $\mathcal{D}_{\mathcal{F}}$ from $\tau(\mathcal{F})$ (i.e., the herbrand model defined from $\tau(\mathcal{F})$). *(ii)* Compute the FO formula $\varphi_{\mathcal{S}} := \bigwedge\{C \mid C$ clause of $\tau(\mathcal{S})^{\infty}\}$. Then,

$$\tau(\mathcal{S}) \cup \tau(\mathcal{F}) \text{ is satisfiable iff } \mathcal{D}_{\mathcal{F}} \models \varphi_{\mathcal{S}},$$

which outlines a reduction to relational database query answering, known to be in **LSpace** [1]. Membership in **LSpace** follows. The argument for COP+Rel is similar.

Membership in **NP** for COP+Rel+TV and COP+Rel+TV+DTV is derived as follows. Consider a knowledge base $(\mathcal{S}, \mathcal{F})$. Consider now the resulting meaning representations, $\tau(\mathcal{S})$ and $\tau(\mathcal{F})$. Clausifying such meaning representations can be done in time constant in $\#(\tau(\mathcal{F}))$. By Pratt and Third's "monadic reducibility" property, we know that we can reduce, in time polynomial in $\#(\tau(\mathcal{F}))$ their satisfiability to that of a set $\tau(\mathcal{S})_u \cup \tau(\mathcal{F})_u$ of monadic clauses. By inspection, we can, moreover, observe that such classes belong to the $\mathcal{S}^+$ class. We can now apply Lemma 1, whence it follows that KB-SAT is in **NP**.

Finally, **NP**-hardness for COP+Rel+TV and COP+Rel+TV+DTV can be inferred by a reduction from the **NP**-complete satisfiability problem for 2+2 clauses [12]. A 2+2 clause is a clause $L_1 \vee L_2 \vee \overline{L}_3 \vee \overline{L}_4$ containing two positive literals and two negative literals. □

Theorem 3 (KB-QA). *If we consider TSQs, then the data complexity of* KB-QA *is*

1. *in* **LSpace** *for COP and in* **PTime** *for COP+TV,*
2. *in* **coNP** *for COP+TV+DTV, and* **coNP**-*complete for COP+Rel, COP+Rel+TV and COP+Rel+TV+DTV.*

Proof. (Sketch.) KB-QA for COP is in **LSpace** in data complexity, because it can be shown that its meaning representations are contained in the description logic *DL-Lite*, for which such result holds [3]. Similarly, it can be shown that COP+TV Kb-Qa reduces to Datalog Kb-Qa. Furthemore, given a COP+TV knowledge base $(\mathcal{S}, \mathcal{F})$ and a TSQ Q, such reduction proceeds in space logarithmic in $\#(\mathcal{F})$. It thus preserves data complexity. Since Datalog Kb-Qa is in **PTime** [1], the result follows.

The **coNP** upper bound for COP+Rel and COP+Rel+TV follows from the **coNP**-completeness for data complexity of Kb-Qa for the two-variable fragment of FO [9]. Regarding COP+TV+DTV and COP+Rel+TV+DTV, we observe that: *(i)* TSQs can be expressed quite easily by COP+Rel+TV+DTV, by extending this FOE with grammar rules accounting for wh- and y/n-questions. *(ii)* COP+Rel+TV+DTV is closed under negation. We can thus reduce Kb-Qa (again, by a reduction space logarithmic in the size of the data) to coKb-Sat (i.e., the complement of Kb-Sat) and apply Theorem 2.

Finally, **coNP**-hardness derives from the fact that we can again reduce the satisfiability of 2+2 clauses to COP+Rel coKb-Qa (i.e., the complement of Kb-Qa). This lower bound then propagates to COP+Rel+TV and COP+Rel+TV. □

4 Conclusions

We have studied the data complexity of Pratt's FOEs w.r.t. Kb-Sat (viz., knowledge base satisfiability) and Kb-Qa (viz., answering TSQs over knowledge bases). In so doing, we have assessed their semantic complexity w.r.t. the common-sense problems of checking for the consistency of and posing questions to known information, which the aforementioned decision problems formalize.

Our results show that the data complexity of the fragments without relative clauses, namely, COP, COP+TV and COP+TV+DTV, are grosso modo, tractable (the upper bound for Kb-Qa for COP+TV+DTV is not tight), and that data complexity is grosso modo, intractable, when relatives are added (the upper bound for Kb-Sat for COP+Rel is not tight either). The following table summarizes the computational properties associated to each such combination of function words:

Tractable	Negation in the predicates of declarations.	Relatives and conjunction everywhere but no negation in questions.
Intractable	Relatives and negation in the subject of declarations.	Relatives, conjunctions and transitive verbs in questions.

Table 3. Data complexity of Kb-Qa and Kb-Sat for the FOEs and TSQs and complexity of satisfiability for the FOEs

	Kb-Qa	**Kb-Sat**	**Satisfiability** [11]
COP	in **LSpace** [Th 3]	in **LSpace** [Th 2]	in **NLSpace**
COP+TV	in **PTime** [Th 3]	in **LSpace** [Th 2]	**NLSpace**-complete
COP+TV+DTV	in **coNP** [Th 3]	in **LSpace** [Th 2]	in **PTime**
COP+Rel	**coNP**-complete [9]	in **LSpace** [Th 2]	**NP**-complete
COP+Rel+TV	**coNP**-complete [9]	**NP**-complete [7,Th 2]	**ExpTime**-complete
COP+Rel+DTV	**coNP**-complete [Th 3]	**NP**-complete [Th 2]	**NExpTime**-complete
COP+Rel+DTV+TV	**coNP**-complete [Th 3]	**NP**-complete [Th 2]	**NExpTime**-complete

Intractability arises when the combination is ”booleanly closed”. Relatives express modulo $\tau(\cdot)$ logical conjunction, hence, when combined with negation, yield intractability (we can encode propositional satisfiability). Note that the complexity of satisfiability (and entailment), and hence the *combined* complexity of KB-SAT and KB-QA, is significantly higher: it is, e.g., **ExpTime**-complete for COP+Rel+TV and **NExpTime**-complete for COP+Rel+TV+DTV (see Table 3).

As related work we must mention the results by Slavkovic [13] regarding approximate reasoning techniques for fragments that map into the two variable fragment of FO and Pratt’s already cited results [11,10] regarding the (combined) computational complexity for satisfiability and entailent of the FOEs. In [15], on the other hand, the data complexity of model checking for several fragments is studied.

References

1. Abiteboul, S., Hull, R., Vianu, V.: Foundations of Databases. Addison-Wesley, Reading (1995)
2. Bernardi, R., Bonin, F., Carbotta, D., Calvanese, D., Thorne, C.: English querying over ontologies: E-QuOnto. In: Proc. of the 10th Congress of the Italian Association for Artificial Intelligence AI*IA 2007 (2007)
3. Bernardi, R., Calvanese, D., Thorne, C.: Expressing DL-Lite ontologies with controlled English. In: Proc. of the 20th Int. Workshop on Description Logics, DL 2007 (2007)
4. Calvanese, D., de Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: Efficiently managing data intensive ontologies. In: Proc. of the 2nd Italian Semantic Web Workshop: Semantic Web Applications and Perspectives, SWAP 2005 (2005)
5. Fermüller, C.G., Leitsch, A., Hustadt, U., Tammet, T.: Resolution decision procedures. In: Robinson, A., Voronkov, A. (eds.) Handbook of Automated Reasoning, vol. 2, ch. 2, 1791–1849. Elsevier - The MIT Press (2001)
6. Joyner, W.H.: Resolution strategies as decision procedures. Journal of the ACM 23(3), 398–417 (1976)
7. Jurafsky, D., Martin, J.: Speech and Language Processing. Prentice-Hall, Englewood Cliffs (2000)
8. Montague, R.: Universal grammar. Theoria 36(3), 373–398 (1970)
9. Pratt, I.: Data complexity of the two-variable fragment with counting quantifiers. Information and Computation 207(8), 867–888 (2008)
10. Pratt, I., Third, A.: More fragments of language. Notre Dame Journal of Formal Logic 47(2), 151–177 (2006)
11. Pratt-Hartmann, I.: Computational complexity of controlled natural languages. In: Proc. of the Workshop in Controlled Languages CNL 2009 (2009)
12. Schaerf, A.: On the complexity of the instance checking problem in concept languages with existential quantification. Journal of Intelligent Information Systems 2(3), 265–278 (1993)
13. Slavkovik, M.: Deep analysis for an interactive question answering system. Master’s thesis, Free University of Bozen-Bolzano (2007)
14. Staab, S., Studer, R. (eds.): Handbook on Ontologies. Int. Handbooks on Information Systems. Springer, Heidelberg (2004)
15. Szymanik, J.: The computational complexity of quantified reciprocals. In: Proc. of the 2007 Tbilisi Colloquium, TbiLLC 2007 (2007)
16. Thorne, C., Calvanese, D.: Tree shaped aggregate queries over ontologies. In: Proc. of the 8th Int. Conf. on Flexible Query Answering Systems FQAS 2009 (2009)
17. Vardi, M.: The complexity of relational query languages. In: Proc. of the 14th Annual ACM Symposium on Theory of Computing, STOC 1982 (1982)