Querying the Fragments of English

Camilo Thorne

KRDB Research Centre for Knowledge and Data
3 Piazza Domenicani, 39100, Bolzano, Italy
cthorne@inf.unibz.it

Abstract. Controlled languages are fragments of natural languages stripped clean of lexical, structural and semantic ambiguity. They have been proposed as a means for providing natural language front-ends to access structured knowledge sources, given that they compositionally and deterministically translate into the (logic-based) formalisms such back-end systems support. An important issue that arises in this context is the semantic data complexity of accessing such information (i.e., the computational complexity of querying measured w.r.t. the number of instances declared in the back-end knowledge base or database). In this paper we study the semantic data complexity of a distinguished family of context-free controlled fragments, viz., Pratt and Third's fragments of English. In doing so, we pinpoint those fragments for which the reasoning problems are tractable (in PTIME) or intractable (NP-hard or CONP-hard).

Keywords. Controlled language interfaces, computational semantics, semantic data complexity, resolution proof procedures, knowledge base query answering and satisfiability.

1 Introduction

Natural language can be considered the ultimate knowledge representation language, due to its expressiveness and the little prior training required. It is also, arguably, the language casual users prefer when interacting with structured data management systems, i.e., when declaring a data constraint, adding a piece of data or querying such information [11]. These considerations have motivated, from the 1970's and 80's [18] to this day [5, 13] a big body of research on natural language interfaces to structured data management systems, such as knowledge-based systems, ontology-based systems and relational databases. But any system that aims at understanding natural language has to deal with the computational costs of semantic processing, viz., with the *semantic complexity* of natural language [16].

Traditionally, the main challenge that semantic processing faces is *ambiguity*, viz., the fact that the same natural language surface form may stand for different deep forms or semantic representations. Ambiguity arises at different levels: at a lexical level (the same word may have different meanings), a structural level (the same sentence may admit difference parses) and a semantic level (the same sentence may "underspecify" several semantic interpretations). Ambiguity induces a very high semantic complexity: in general, the number of readings of an utterance will grow exponentially in its size [10]. Worse still, since natural language semantics is captured, modulo so-called

compositional translations $\tau(\cdot)$, by higher order logic (HO) and its fragments, notably, first oder logic (FO), the problem of ambiguity is, in general, undecidable.

To tackle the high semantic complexity of ambiguity it has been proposed to trade off expressiveness, by considering *controlled languages*, viz., ambiguity-free fragments of natural languages which allow for efficient (polynomial) deterministic parsing and semantic interpretation [7], thus guaranteeing high or absolute precision and recall at the cost of constraining natural language input.

Ambiguity unfortunately, is not the only factor inducing prohibitive semantic complexity. In general, we are also interested in *reasoning* over semantic representations. In knowledge-based systems, for instance, we might want to leverage on intensional information (domain knowledge and constraints) to refine queries or questions. Such refinement can be arguably considered part of an utterance's semantic processing. Thus, controlled language coverage (i.e., its verbs, nouns, adjectives, relative clauses, coordinating particles, quantification, etc.) can also impact on semantic complexity. For instance, function words which map to logical operations such as Boolean conjunction and negation can give rise to "Boolean-closed" fragments for which reasoning is intractable [17]. Similar results hold for fragments covering some distinguished classes of (generalized) quantifiers, which are intractable w.r.t. to FO finite model checking [21].

In this paper we study the computational *data complexity* of answering queries over controlled English knowledge bases, viz., the computational complexity of this problem (a FO entailment problem) measured in the number of facts stored in the knowledge base. Data complexity is an important (theoretical) measure of query evaluation efficiency in knowledge-based systems [2]. We consider for this purpose the so-called *fragments of English*, a family of very simple controlled fragments proposed by Pratt and Third in [17]. We show for which fragments data complexity is tractable, for which it is intractable and for which query answering is undecidable. As the reader shall see, bad computational properties arise even in the presence of fragments of very little expressiveness.

2 The Fragments of English and Semantic Complexity

In this section we present a generic methodology for defining controlled fragments, proposed by Pratt and Third in [17], the *fragments of English*. These fragments are defined using context-free semantically enriched grammars and has the advantage of generating, alongside the set of grammatical utterances of the fragment, their logical (HO and FO) meaning representations.

Their semantics follows the Montague semantics paradigm outlined in [14], in which English utterances S and sets $\mathcal{S}$ thereof are mapped into into a HO or FO *meaning representation* $\tau(S)$ or set $\tau(\mathcal{S})$ of formulas, leveraging on the λ-abstraction, application and normalization of type-theory. Semantic actions recursively define the compositional translation $\tau(\cdot)$ on a fragment's sentence constituents.

As with all controlled languages, ambiguity is ruled out: complete utterances admit one and only one parse tree and one and only one semantic representation. Consequently, each fragment of English expresses a unique FO fragment, whose computa-

tional properties can be easily studied. In addition, the definition of $\tau(\cdot)$ over grammar lexicons gives rise to their partition into:

- An arbitrarily large *content lexicon* denoting *individuals* with proper nouns ($\mathbf{Pn}$s like "Max"), and unary, binary and ternary *relations*, with, resp., nouns ($\mathbf{N}$s like "beer"), transitive verbs ($\mathbf{TV}$s like "drinks") and ditransitive verbs ($\mathbf{DTV}$s like "gives").
- A finite *function lexicon* denoting *logical operations* over individuals and relations. Relative pronouns ($\mathbf{Relp}$s like "who") and coordinators ($\mathbf{Crd}$s like "and") denote logical conjunction. Determiners ($\mathbf{Det}$s like "some") denote quantification. Negation ($\mathbf{Ng}$s like "not") denotes logical negation. Anaphoric pronouns express co-references between logical expressions.

The fragments themselves are defined incrementally, starting from a base fragment, COP (for "copula"), whose coverage is subsequently expanded to $\mathbf{TV}s$, $\mathbf{DTV}$s, $\mathbf{Relp}$s and (restricted) anaphora, as summarized by Table 1. In what follows we will describe in detail only COP. The other fragments are defined similarly:

Function Lexicon	
$\mathbf{Det} \rightarrow$ every	$\tau(\mathbf{Det}) := \lambda P^{e\rightarrow t}.\lambda Q^{e\rightarrow t}.\forall x^{e}(P(x) \Rightarrow Q(x))$
$\mathbf{Det} \rightarrow$ some	$\tau(\mathbf{Det}) := \lambda P^{e\rightarrow t}.\lambda Q^{e\rightarrow t}.\exists x^{e}(P(x) \wedge Q(x))$
$\mathbf{Det} \rightarrow$ no	$\tau(\mathbf{Det}) := \lambda P^{e\rightarrow t}.\lambda Q^{e\rightarrow t}.\forall x^{e}(P(x) \Rightarrow \neg Q(x))$
$\mathbf{Ng} \rightarrow$ not	$\tau(\mathbf{Ng}) := \lambda P^{e\rightarrow t}.\lambda x^{e}.\neg P(x)$

Phrase Structure Rules		Content Lexicon	
$\mathbf{S}\rightarrow\mathbf{NP\ VP}$	$\tau(\mathbf{S}){:=}\tau(\mathbf{NP})(\tau(\mathbf{VP}))$	$\mathbf{N}\rightarrow$woman	$\tau(\mathbf{N}){:=}\lambda x^{e}.\mathit{Woman}(x)$
$\mathbf{VP}\rightarrow$ is a $\mathbf{N}$	$\tau(\mathbf{VP}){:=}\tau(\mathbf{N})$	$\mathbf{N}\rightarrow$man	$\tau(\mathbf{N}){:=}\lambda x^{e}.\mathit{Man}(x)$
$\mathbf{VP}\rightarrow$ is $\mathbf{Ng}$ a $\mathbf{N}$	$\tau(\mathbf{VP}){:=}\tau(\mathbf{Ng})(\tau(\mathbf{N}))$	$\mathbf{N}\rightarrow$person	$\tau(\mathbf{N}){:=}\lambda x^{e}.\mathit{Person}(x)$
$\mathbf{NP}\rightarrow\mathbf{Pn}$	$\tau(\mathbf{NP}){:=}\tau(\mathbf{Pn})$	$\mathbf{N}\rightarrow$human	$\tau(\mathbf{N}){:=}\lambda x^{e}.\mathit{Human}(x)$
$\mathbf{NP}\rightarrow\mathbf{Det\ N}$	$\tau(\mathbf{NP}){:=}\tau(\mathbf{Det})(\tau(\mathbf{N}))$	$\mathbf{Pn}\rightarrow$Mary	$\tau(\mathbf{Pn}){:=}\lambda P^{e\rightarrow t}.P(\text{Mary})$

The function lexicon express FO universal and existential quantification, in addition to a form of negation. The content lexicon may contain an arbitrarily large stock of common and proper nouns specifying an arbitrarily large FO signature of unary predicates and individual constants.

To each lexical entry and each grammar rewriting rule a semantic action is associated and the computation of $\tau(\cdot)$ can be done on the fly, i.e., side-by-side with the computation of the parse tree. See Figure 1. In COP we can express FO sentences of these forms:

$\mathit{Woman}(\text{Mary})$	Mary is a woman.	$\forall x(\mathit{Man}(x) \Rightarrow \mathit{Person}(x))$	Every man is a person.
$\neg\mathit{Man}(\text{Mary})$	Mary is not a man.	$\forall x(\mathit{Woman}(x) \Rightarrow \neg\mathit{Man}(x))$	No woman is a man.
		$\forall x(\mathit{Person}(x) \Rightarrow \mathit{Human}(x))$	Every person is a human.
		$\exists x(\mathit{Person}(x) \wedge \mathit{Woman}(x))$	Some person is a woman
		$\exists x(\mathit{Person}(x) \wedge \neg\mathit{Woman}(x))$	Some person is not a woman.

More in general, by suitably modifying the content lexicon, we can express *any* FO sentence of these forms. This defines a proper fragment of FO, which we may denote by

COP	copula, common and proper nouns, negation universal and existential quantifiers
COP+TV	COP + transitive verbs (“loves”)
COP+TV+DTV	COP + transitive and ditransitive verbs (“gives”)
COP+Rel	COP + relative pronouns (“who”, “that”)
COP+Rel+TV	COP+Rel + transitive verbs
COP+Rel+TV+DTV	COP+Rel + transitive and ditransitive verbs
COP+Rel+TV+RA	COP+Rel+TV + restricted anaphora (“he”, “it”, “herself”)

Table 1. The fragments of English.

abuse of notation COP as well [17]. Semantic complexity will be, in general, correlated to the fragment’s function lexicon.

Modulo compositionality and the ensuing logic meaning representations, the fragments of English can be said to be subsumed by several known fragments of FO: the monadic fragment of FO (COP and COP+Rel), the 2-variable fragment of FO (due to transitive verbs: COP+TV, COP+Rel+TV and COP+Rel+TV+RA) and the 3-variable fragment of FO (due to ditransitive verbs: COP+TV+DTV and COP+Rel+TV+DTV). They are known to be subsumed or to overlap with several knowledge representation logics such as description logics [4] and have been shown to be decidable for satiafiability [17]. It is perhaps interesting to note that the non-monadic fragments are incomparable to other known fragments such as the guarded fragment[1].

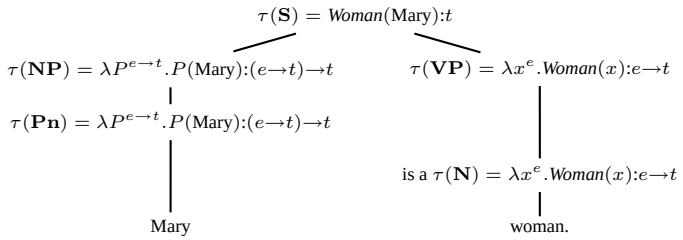

Fig. 1. Parse tree for the COP sentence “Mary is a woman.”.

In general, controlled fragments designed with practical applications in mind tend to be very expressive and are not easily amenable to a computational complexity analysis. The best-known controlled fragment of English, Attempto Controlled English, ACE [7] (that contains all of the fragments discussed here), is for instance more expressive than FO itself, thus being undecidable for satisfiability. Pratt and Third’s fragments by contrast, by being simple and decidable (in most cases), allow to pinpoint those com-

[1] A COP+TV sentence like “Every liar knows every trick.”, yields a semantic representation, $\forall x(\mathit{Liar}(x) \Rightarrow \forall y(\mathit{Trick}(y) \Rightarrow \mathit{Knows}(x,y)))$, that is not guarded [8].

binations of English constructs that give rise to tractable and intractable computational complexity.

3 Querying the Fragments of English

In the remainder of this paper, we will focus on one particular kind of data access and storage system, an ontology-based system or knowledge base, in which data, stored as instances or as records in a (relational) database is managed and queried through an intensional middle layer of *domain constraints* or *ontology* Σ (a set of FO sentences), expressed by a set $\mathcal{S}$ of controlled English sentences.

A *knowledge base* is a pair (Σ, Δ), with Σ an ontology and Δ a a set of facts or FO relational ground atoms. The integer $\#(\Delta)$ denotes the *size* of Δ, the number facts it contains. The usual FO notions of truth and satisfaction apply to ontologies, databases and knowledge bases, and to their constraints and facts.

A *conjunctive query* is an existentially quantified conjunction of positive FO relational atoms

$$\varphi(\boldsymbol{x}) := \exists \boldsymbol{y} \varphi(\boldsymbol{x}, \boldsymbol{y})$$

over variables $\boldsymbol{x}$ and $\boldsymbol{y}$, where the free variables $\boldsymbol{x}$ are known also as the CQ's *distinguished variables*. A *union of conjunctive queries* is a finite disjunction

$$\varphi(\boldsymbol{x}) := \exists \boldsymbol{y}_1 \varphi_1(\boldsymbol{x}, \boldsymbol{y}_1) \vee \cdots \vee \exists \boldsymbol{y}_k \varphi_k(\boldsymbol{x}, \boldsymbol{y}_k)$$

of conjunctive queries. A *tree-shaped query* (TCQ) is a CQ with one distinguished variable x, called *root*, defined inductively by

$$\varphi(x) \longrightarrow A(x) \mid \exists y R(x, y) \mid \exists y R(x, y) \wedge \varphi(y) \mid \varphi(x) \wedge \varphi'(x)$$

The *length* $|\boldsymbol{x}|$ of the sequence of distinguished variables is known as the *arity* of the U(T)CQ. U(T)CQs are said to be *Boolean* when they contain no free variables.

Note that U(T)CQs are positive existential FO formulas. U(T)CQs have a practical motivation: they correspond to the SELECT-PROJECT-JOIN fragment of SQL, the most frequently used class of queries in relational databases [1].

A *substitution* σ is a function from variables to terms. It is called a *renaming* when it is a function from variables to variables and a *grounding* when it is a function from variables to constants. Substitutions can be extended to complex syntactic objects in the standard way.

A knowledge base (Σ, Δ) is said to entail a U(T)CQ φ w.r.t. a substitution σ, in symbols, $\Sigma \cup \Delta \models \varphi\sigma$, if every model of $\Sigma \cup \Delta$ is a model of $\varphi\sigma$ (i.e., plain FO entailment).

Two reasoning tasks are of relevance: *(i)* Consistency: we want to know whether the data being stored is not in conflict with the constraints and is meaningful. *(ii)* Query answering: we want to retrieve information. In both cases, it is customary to understand the data complexity of the tasks, viz., the computational complexity of their associated recognition decision problems measured w.r.t. the number of instances or records stored in the system. This idea was first proposed by Vardi in [23] for relational databases, and

is based on the observation that the main issue in real-world applications is scalability to massive datasets.

Given a knowledge base (Σ, Δ), the knowledge base *satisfiability* problem (KBQA) is the decision problem defined by: **Input:** the set Δ of facts. **Question:** does there exist a model for $\Sigma \cup \Delta$?

Given a knowledge base (Σ, Δ), a U(T)CQ φ of distinguished variables $\boldsymbol{x}$, a sequence of constants $\boldsymbol{c}$ and a substitution σ s.t., $\sigma(\boldsymbol{x}) = \boldsymbol{c}$, the knowledge base *query answering* problem (KBQA) is the decision problem defined by: **Input:** the set Δ of facts. **Question:** does $\Sigma \cup \Delta \models \varphi\sigma$ hold?

4 Resolution Saturations and Data Complexity

The resolution calculus was first proposed by Robinson in [19] as a sound and complete, but not necessarily terminating machine-oriented calculus for FO. Later, Joyner in [9] showed that resolution can be turned into a decision procedure for (un)satisfiability for a number of distinguished fragments of FO, by proposing several saturation- or forward-chaining-based *refinings* of resolution. In [22], we showed how such refinements can be used to provide upper data complexity bounds for KBSAT for several fragments of English. In this section, we extend such results to all the (decidable) fragments of English and generalize the technique to KBQA.

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 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 $\bot$. By *Var*(t), *Var*(L) and *Var*(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 variables.

A *unifier* of two terms t and t' is a substitution σ s.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 recursively 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 t is defined by *(i)* $d(x, t) = 0$ if $x \notin$ *Var*(t), otherwise *(ii)* $d(x, x) := 1$ and $d(x, f(t_1, \ldots, t_n)) := \max\{d(x, t_i) \mid i \in [1, n]\} + 1$. The *relative depth* $d(x, L)$ of a variable x in a literal L is its maximal relative depth among L's terms.

A clause C is "well behaved" whenever *(i)* *Var*$(L) \leq 1$ and *(ii)* either for every functional term t in C, *Var*$(t) =$ *Var*(C), or *Var*$(L) =$ *Var*(C), for all literals in C[2].

The saturation-based versions of the (ordered) resolution calculus iteratively (monotonically w.r.t. $\subseteq$) generate the set of all possible clauses derived from Γ, until either *(i)* $\bot$ is derived or *(ii)* all possible clauses are generated (fixpoint computation). Such

[2] Clauses that are "well-behaved" correspond to the clauses from the S^+ class studied in [6].

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

Table 2. Resolution calculi. Calculus $\mathcal{R}_{2,5}$ is a decision procedure for "well-behaved clauses [6].

clauses are obtained using *res* (resolution), *fact* (factoring) and *split* (splitting):

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

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

where σ is a most general unifier of L and L', the *acceptable* depth ordering (well-founded and substitution-invariant partial order on clause literals and sets thereof) $\prec_d$ defined by

$$L \prec_d L' \quad \text{iff} \quad \textit{(i)}\ d(L) < d(L'),\ \textit{(ii)}\ \textit{Var}(L) \subseteq \textit{Var}(L'),\ \textit{(iii)}\ d(x, L) < d(x, L'),$$

for all $x \in \textit{Var}(L)$, and the *mon* (monadization) rule. Orderings give rise to the ordered versions of *res*, *fact* and *split*. Note that *split* introduces branching in saturations.

The $\prec_d$ ordering prevents clause depth from "well-behaved" clauses from growing beyond a finite bound $d \geq 0$, whereas splitting prevents their length l from growing arbitrarily. Now, ordered *res*, on "well-behaved" clauses may generate non-"well-behaved" clauses, whose depth may grow arbitrarily. Given a set Γ of clauses derived by ordered resolution from "well-behaved" clauses, *mon* substitutes, in each literal L in Γ, all the variables $x \in \textit{Var}(L)$ that *do not* occur in L's functional terms t, with either t or the constants c that occur in Γ. Thus, monadization, which is satisfiability-preserving for such clauses (but not necessarily in the general case), ensures that "well-behaved" clauses are closed under the ordered rules, guaranteeing the termination of saturations [9, 6].

Formally, consider a derivation function $\rho(\cdot)$ over sets of clauses, defined in terms of rules stated above. A *resolution calculus* is a function $\mathcal{R}(\cdot)$ s.t. $\mathcal{R}(\Gamma) := \Gamma \cup \rho(\Gamma)$. A *saturation* is defined as the limit Γ^∞ of the sequence defined by *(i)* $\mathcal{R}^0(\Gamma) := \Gamma$ and *(ii)* $\mathcal{R}^{i+1}(\Gamma) := \mathcal{R}(\mathcal{R}^i(\Gamma))$, for $i > 0$. The positive integer i is called the *depth* of the saturation. Different calculi arise from the different combinations of rules, orderings and refinements, as summarized by Table 4.

Resolution is sound and complete w.r.t. (un)satisfiability, in the sense that Γ is unsatisfiable iff $\bot \in \Gamma^\infty$, and $\mathcal{R}_{2,5}$ in particular terminates if Γ is a set of "well behaved" clauses. Moreover, it is immediate to see that, in general, resolving upon ground clauses can be delayed to the last steps of the saturation:

Proposition 1. *Given a terminating saturation for a set Γ of clauses containing ground atoms, one can delay deriving upon those ground atoms to the last steps of the saturation.*

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. See Figure 2. Due to this observation, resolution decision procedures can be used to provide upper data complexity bounds.

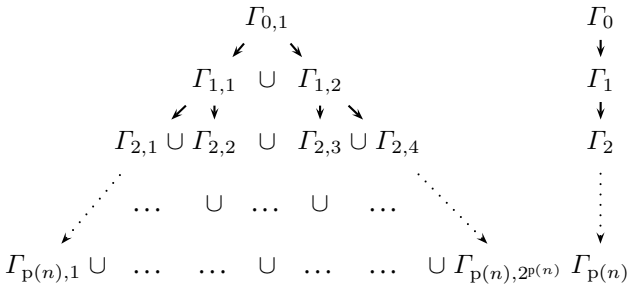

Fig. 2. Terminating saturations of a set Γ of clauses using (tree-shaped) and not using (linear-shaped) the splitting rule. In the figure, directed edges denote derivation steps and $\mathrm{p}(\cdot)$ denotes a polynomial on the number n of constants in Γ, that bounds the depth of saturation.

Lemma 1 (from [22], Lem. 1). *Let Γ be a set of clauses containing n constants (and finitely many function and relation symbols) for which there exist both a term depth bound $d \geq 0$ and a clause length bound $k \geq 0$. In the worst case: (i) the number of clauses derivable by the saturation is (a) exponential in n if we use the splitting rule or (b) polynomial in n otherwise, and (ii) the depth of the saturation is polynomial in n.*

4.1 Tractable Fragments

In this section we show that the data tractable fragments for KBQA are those involving no relative pronouns and that are not Boolean-closed. Similar results hold for KBSAT, provided relatives and transitive verbs are not covered simulaneously by the same fragment(s). For the PTIME upper bound below we use saturations to define a reduction to the well-known (un)satisfiability problem for the HORN fragment (the fragment of FO clauses in which at most one literal is positive), which is known to be in PTIME in the ground case (see, e.g., [8]).

Theorem 1. *The data complexity of* KBSAT *is in* PTIME *for COP+Rel.*

Proof. Let Σ be a set of COP+Rel meaning representations and Δ a database, where Σ is fixed and let $\Sigma^{cl} \cup \Delta^{cl}$ be their clausification and Skolemization. $\Sigma^{cl} \cup \Delta^{cl}$ can

be computed in $\mathbf{O}(\log \#(\Delta))$ space. This gives way to polynomially many constants in $\#(\Delta)$. Now, the clauses in $\Sigma^{cl} \cup \Delta^{cl}$ are *(i)* monadic, and *(ii)* if in Σ^{cl}, Boolean combinations of unary atoms over the single variable x, and containing no function symbols. Clearly, the clauses in $\Sigma^{cl} \cup \Delta^{cl}$ are "well-behaved" clauses of the kind for which resolution saturations can be used as a decision procedure. Furthermore, the *split* rule is not applicable. Use therefore the calculus $\mathcal{R}_{2,4}$: by Lemma 1, we know that such a procedure will run in time polynomial in the number constants of $\Sigma^{cl} \cup \Delta^{cl}$ and hence in time polynomial in $\#(\Delta)$. □

Theorem 2 (from [22], Th. 1). *The data complexity for* KBSAT *is in* LSPACE *for COP, COP+TV and COP+TV+DTV.*

Theorem 3 (from [22], Th. 2). *The data complexity of* KBQA *for COP and UCQs is in* LSPACE.

Theorem 4. *The data complexity of* KBQA *for the fragments COP+TV+DTV and UCQs is in* PTIME.

Proof. By reduction to HORN satisfiability. Let Σ be a set of fixed COP+TV+DTV meaning representations, Δ a database and φ a fixed UCQ of distinguished variables $\boldsymbol{x}$. Let $\boldsymbol{c}$ be a tuple (a sequence of $|\boldsymbol{x}|$ domain constants) and σ a substitution mapping $\boldsymbol{x}$ to $\boldsymbol{c}$. To know whether $\boldsymbol{c}$ is an answer to φ over (Σ, Δ), it is sufficient to check if $\Sigma \cup \Delta \models \varphi\sigma$.

Consider now the sets Σ^{cl}, Δ^{cl} and $(\neg\varphi\sigma)^{cl}$, where by $\cdot^{cl}$ we denote the (satisfiability-preserving) Skolemization and clausification of Σ, Δ and $\neg\varphi\sigma$. Skolemization and clausification are constant in $\#(\Delta)$: Δ is already a set of (ground) clauses. We claim that Σ^{cl}, Δ^{cl} and $(\neg\varphi\sigma)^{cl}$ are sets of HORN clauses. Clearly, Δ^{cl} is a set of HORN ground clauses. On the other hand, the clauses in Σ^{cl} are all of the form(s) [17]:

$$\neg P(x) \vee \pm Q(x),\ \neg P(x) \vee \pm L(x),\ \neg P(x) \vee Q(f(x)),\ \neg P(x) \vee \neg Q(y) \vee \pm L(x,y),$$
$$\pm L,\ \neg P(x) \vee \neg Q(y) \vee \neg N(z) \pm L(x,y,z),\ \neg P(x) \vee \neg Q(y) \vee N(g(x,y)),$$

where $L(\boldsymbol{x})$ stands for a binary or ternary literal with free variables $\boldsymbol{x}$ and possibly functional terms, which are also HORN. Since φ is a UCQ, this means that $\varphi\sigma$ is of the form $\exists y_1 \psi(\boldsymbol{c}, \boldsymbol{y}_1) \vee \cdots \vee \exists y_k \psi(\boldsymbol{c}, \boldsymbol{y}_k)$, with each $\psi_i(\boldsymbol{c}, \boldsymbol{y}_i)$ of the form $L'_{i1}(\boldsymbol{t}_{i1}) \wedge \cdots \wedge L'_{im}(\boldsymbol{t}_{im})$, where $\boldsymbol{c} \cup \boldsymbol{y}_i \subseteq \boldsymbol{t}_{i1} \cup \cdots \cup \boldsymbol{t}_{im}$ and L'_{ij}, for $1 \leq j \leq m$, is a relational symbol of arity ≤ 2. Hence, $(\neg\varphi\sigma)^{cl}$ is the set of (two variable) HORN clauses $\{\neg L'_{11}(\boldsymbol{t}_{11}) \vee \cdots \vee \neg L'_{1m}(\boldsymbol{t}_{1m})\} \cup \cdots \cup \{\neg L'_{k1}(\boldsymbol{t}_{k1}) \vee \cdots \vee \neg L'_{km}(\boldsymbol{t}_{km})\}$.

The clauses in Σ^{cl}, Δ^{cl} and $(\neg\varphi\sigma)^{cl}$ are "well-behaved" clauses, viz., clauses where either all literals are essentially monadic, or covering, or whose functional terms are covering. Hence, resolution saturations (e.g., calculus $\mathcal{R}_{2,5}$) can be used to decide their (un)satisfiability. Furthermore, by Proposition 1, we know that we can "separate" facts from non-ground clauses. Finally, by grounding the saturation (following the Herbrand theorem, cf. [12], Prop. IV-5), we can reduce answering a question to checking for the satisfiability of a set of HORN (propositional) clauses, since

$$\begin{array}{lll} \Sigma \cup \Delta \models \varphi\sigma & \text{iff} & \Sigma^{cl} \cup \Delta^{cl} \cup (\neg\varphi\sigma)^{cl} \text{ is unsatisfiable} \\ & \text{iff} & \bot \in (\Sigma^{cl} \cup \Delta^{cl} \cup (\neg\varphi\sigma)^{cl})^{\infty} \\ & \text{iff} & \bot \in ((\Sigma^{cl} \cup \Delta^{cl})^{\infty} \cup (\neg\varphi\sigma)^{cl})^{\infty} \\ & \text{iff} & \mathrm{GR}((\Sigma^{cl} \cup (\neg\varphi\sigma)^{cl})^{\infty}) \cup \Delta^{cl} \text{ is unsatisfiable,} \end{array} \qquad (\dagger)$$

where GR$(\cdot)$ denotes the operation that grounds the clauses in $(\Sigma^{cl} \cup (\neg\varphi\sigma)^{cl})^{\infty})$ (of which there are finitely many) with constants taken from $adom(\Delta)$, holds.

By (†) we know that the reduction is sound and complete. We already saw that the clausification procedure is constant in $\#(\Delta)$. In addition, by the separation property, saturating, while exponential in general, can be done independently from the data and thus does not affect data complexity and is thus constant in $\#(\Delta)$ We also know that there are polynomially many groundings in $\#(\Delta)$. Finally, checking for the satisfiability of GR$((\Sigma^{cl} \cup (\neg\varphi\sigma)^{cl})^{\infty}) \cup \Delta^{cl}$, which are HORN, can be done in time polynomial in $\#(\Delta)$. □

Corollary 1. *The data complexity of* KBQA *for the controlled fragments COP+TV, and COP+TV+DTV and UCQs is in* PTIME.

4.2 Intractable Fragments

In this section we show that the data intractable fragments for KBQA are those covering relative pronouns and that are Boolean-closed. This is shown by a reduction from an NP-complete propositional satisfiability problem. For KBSAT, the boundary lies in COP+Rel+TV that covers both relatives and transitive verbs. Membership in, resp., NP and CONP is derived using saturations.

Theorem 5 (from [22], Th. 1). *The data complexity of* KBSAT *is* NP*-complete for COP+Rel+TV and COP+Rel+TV+DTV.*

The satisfiability problem for *propositional 2+2 formulas* (2+2-SAT) is the decision problem defined by: **Input:** a conjunction of clauses of the form form $\psi := \psi_1 \wedge \cdots \wedge \psi_k$ where each clause $\psi_i := p_{i1} \vee p_{i2} \vee \neg n_{i1} \vee \neg n_{i2}$ is a disjunction of two positive and two negative literals. **Question:** does there exist a truth assignment $\delta(\cdot)$ s.t. $\delta(\psi) = 1$? This problem was shown to be NP-complete by Schaerf in [20].

Lemma 2. KBQA *is* CONP*-hard in data complexity for COP+Rel and TCQs.*

Proof. We define a reduction from 2+2-SAT. Let $\psi := \psi_1 \wedge \cdots \wedge \psi_k$ be a 2+2-formula over the literals $At(\psi) := \{l_1, ..., l_m\}$ where, for $i \in [1, k]$, $\psi_i := p_{i1} \vee p_{i2} \vee \neg n_{i1} \vee \neg n_{i2}$ is a disjunction of two positive and two negative literals.

Let N_1 ("has as first negative literal"), N_2 ("has as second negative literal"), P_1 ("has as first positive literal") and P_2 ("has as second positive literal") four binary relation symbols/transitive verbs. Let A ("literal"), A_f ("false literal") and A_t ("true literal") be three unary relation symbols/nouns. For each clause ψ_i in ψ, for $i \in [1, k]$, introduce an individual constant/proper name c_i and similarly a constant/proper name l for every $l \in At(\psi)$. Encode ψ with the facts

$$\Delta_\psi := \left\{ \begin{array}{c} A(p_{11}), A(p_{12}), A(n_{11}), A(n_{12}), \ldots, A(p_{k1}), A(p_{k2}), A(n_{k1}), A(n_{k2}), \\ P_1(c_1, p_{11}), P_2(c_1, p_{12}), N_1(c_1, n_{11}), P_2(c_1, n_{12}), \ldots, P_1(c_k, p_{k1}), \\ P_2(c_k, p_{k2}), N_1(c_k, n_{k1}), N_2(c_k, n_{k2}) \end{array} \right\},$$

consider, resp., the declarative sentences $\mathcal{S}$

"No A is not an A_t that is not an A_f", "No A_t is an A_f." and "No A_f is an A_t.",

and the TCQ

$$\varphi := \exists x \exists y (P_1(x,y) \land A_f(y)) \land \exists z (P_2(x,z) \land A_f(z)) \land \\ \exists w (N_1(x,w) \land A_t(w)) \land \exists v (N_2(x,v) \land A_t(v))),$$

and claim that

$$\psi \text{ is satisfiable} \quad \text{iff} \quad \tau(\mathcal{S}) \cup \Delta_\psi \not\models \varphi \tag{†}$$

($\Rightarrow$) Suppose that ψ is satisfiable and let $\delta(\cdot)$ be a truth assignment. Then, for all $i \in [1,k]$, $\delta(\psi_i) = 1$, i.e., $\delta(p_{i1}) = 1$ or $\delta(p_{i2}) = 1$ or $\delta(n_{i1}) = 0$ or $\delta(n_{i2}) = 0$. Given this, we can construct an interpretation $\mathcal{I} = (\mathbb{D}_\mathcal{I}, \cdot^\mathcal{I})$ s.t. $\mathcal{I} \models \tau(\mathcal{S}) \cup \Delta_\psi$ but $\mathcal{I} \not\models \varphi$ as follows:

- $\mathbb{D}_\mathcal{I} := \{c_i, p_{ij}, n_{ij} \mid i \in [1,k], j = 1,2\}$, $A^\mathcal{I} := \{l \in \mathit{At}(\psi) \mid A(l) \in \tau(\mathcal{F}_\psi)\}$,
- $A_f^\mathcal{I} := \{l \in A^\mathcal{I} \mid \delta(l) = 0\}$, $A_t^\mathcal{I} := \{l \in A^\mathcal{I} \mid \delta(l) = 1\}$,
- $P_j^\mathcal{I} := \{(c_i, p_{ij}) \mid P_j(c_i, p_{ij}) \in \Delta_\psi, i \in [1,k]\}$, and
- $N_j^\mathcal{I} := \{(c_i, n_{ij}) \mid N_j(c_i, n_{ij}) \in \Delta_\psi, i \in [1,k]\}$,

($\Leftarrow$) Let $\mathcal{I}$ be a model $\tau(\mathcal{S}) \cup \Delta_\psi$ of s.t. $\mathcal{I}, \gamma \not\models \varphi$ for all γ. We want to show that there exists a truth assignment $\delta(\cdot)$ s.t. $\delta(\psi) = 1$. Let $\delta \colon \mathit{At}(\psi) \to \{0,1\}$ be the truth assignment s.t.

$$\delta(l) = 1 \quad \text{iff} \quad l \in A_t^\mathcal{I}.$$

Now, by assumption $\mathcal{I}, \gamma \not\models \varphi$, for all γ. This implies, for all $i \in [1,k]$, that either $p_{i1} \notin A_f^\mathcal{I}$ or $p_{i2} \notin A_f^\mathcal{I}$ or $n_{i1} \notin A_t^\mathcal{I}$ or $n_{i2} \notin A_t^\mathcal{I}$. Now, recall that $\mathcal{I} \models \tau(\mathcal{S})$, where $\tau(\mathcal{S})$ contains the axioms $\forall x(A(x) \Rightarrow A_t(x) \lor A_f(x))$, $\forall x(A_t(x) \Rightarrow \neg A_f(x))$ and $\forall x(A_f(x) \Rightarrow \neg A_t(x))$, that "say" that a literal is either true or false, but not both. Hence if $p_{i1} \notin A_f^\mathcal{I}$, then, by definition of $\delta(\cdot)$, $\delta(p_{i1}) = 1$ and similarly for the other cases. Therefore, $\delta(\psi_i) = 1$, for all $i \in [1,k]$, and thus $\delta(\psi) = 1$. □

Lemma 3 (from [22], Th. 2). *If we consider TCQs, the data complexity of* KBQA *is in* CONP *for COP+Rel, COP+Rel+TV and COP+Rel+TV+DTV.*

Theorem 6. *The data complexity of* KBQA *and TCQs is* CONP*-complete for COP+Rel, COP+Rel+TV and COP+Rel+TV+DTV.*

Proof. Follows from Lemma 2 and Lemma 3. □

5 An Undecidable Fragment

As we have seen, adding relatives to COP makes answering U(T)CQs hard w.r.t. data complexity. In this section we will see that when we consider also transitive verbs and restricted anaphoric pronouns (that co-refer with their closest antecedent noun), KBQA becomes undecidable. We will consider a slightly enriched version of the fragment COP+Rel+TV+DTV+RA, COP+Rel+TV+DTV+RA$^+$, that covers verbs in *passive* form (e.g., "is loved by") and *indeterminate pronouns* (e.g., "somebody"). This we prove by a reduction from the unbounded tiling problem.

A *tiling grid* or *grid* is a tuple $\mathcal{T} = (\mathbb{T}, \mathbb{V}, \mathbb{H})$, where $\mathbb{T} := \{c_0, \ldots, c_k\}$ is a finite set of k *tiles* and $\mathbb{V}$ and $\mathbb{H}$ are binary relations over $\mathbb{T}$, called, resp., the *vertical* and *horizontal* relations. A *tiling* is a function $t \colon \mathbb{N} \times \mathbb{N} \to \mathbb{T}$ that satisfies the *horizontal*

and *vertical adjacency constraints*, i.e., s.t., for all $i, j \in \mathbb{N}$, $(t(i,j), t(i,j+1)) \in \mathbb{H}$ and $(t(i,j), t(i+1,j)) \in \mathbb{V}$. The *unbounded tiling problem* (TP) is the decision problem defined by: **Input:** a grid $\mathcal{T} = (\mathbb{T}, \mathbb{V}, \mathbb{H})$. **Question:** does a tiling t exist for $\mathcal{T}$? It was shown to be undecidable by Berger in [3].

Theorem 7. KBQA *is undecidable for COP+Rel+TV+RA$^+$ and arbitrary CQs.*

Proof. By reduction from TP to the *complement* of KBQA for COP+Rel+TV+RA$^+$ and arbitrary CQs. Let $\mathcal{T} = (\mathbb{T}, \mathbb{V}, \mathbb{H})$ be a tiling grid, with $\mathbb{T} := \{c_0, \ldots, c_k\}$ being its tiles and $\mathbb{H}$ and $\mathbb{V}$ the, resp., horizontal and vertical relations over $\mathbb{T}$. We can encode $\mathcal{T}$ with a set $\mathcal{S}_\mathcal{T}$ of COP+Rel+TV+RA sentences and a set of facts $\Delta_\mathcal{T}$ as follows.

Let the transitive verbs/binary relation symbols H and V encode, resp., the horizontal $\mathbb{H}$ and vertical $\mathbb{V}$ relations. Let the noun/unary relation C_i encode a tile $c_i \in \mathbb{T}$, for $i \in [0,k]$. Let finally $\bar{H}$ be a "fresh" transitive verb/binary relation. We start by defining the set $\mathcal{S}_\mathcal{T}$ that encodes the structure of the grid:

$$\text{Everything } H\text{s something.} \quad \text{Everything } V\text{s something.} \tag{1}$$

$$\text{For all } i \in [0,k]\text{:} \quad \text{Anything that is not a } C_i \text{ is a } C_{i+1}. \quad \text{No } C_i \text{ is a } C_{i+1}. \tag{2}$$

$$\begin{array}{r} \text{For all } (c,c') \notin \mathbb{V}\text{:} \quad \text{Everybody who } H\text{s somebody is a } C. \\ \text{Everybody who is } H\text{d by somebody is a } C'. \end{array} \tag{3}$$

$$\begin{array}{r} \text{For all } (c,c') \notin \mathbb{H}\text{:} \quad \text{Everybody who } V\text{s somebody is a } C. \\ \text{Everybody who is } V\text{d by somebody is a } C'. \end{array} \tag{4}$$

$$\begin{array}{c} \text{Everybody who } \bar{H}\text{s somebody does not } H \text{ him.} \\ \text{Everybody who does not } H \text{ somebody } \bar{H}\text{'s him.} \end{array} \tag{5}$$

Next, we define $\Delta_\mathcal{T}$, that encodes the initial state of the grid:

$$\Delta_\mathcal{T} := \left\{ C_0(c_0), \ldots, C_k(c_k), H(c_0,c_1), \ldots, H(c_{k-1},c_k) \right\}.$$

Finally, we consider the CQ

$$\varphi := \exists x \exists y \exists z \exists w (H(x,y) \wedge V(y,w) \wedge V(x,z) \wedge \bar{H}(z,w)),$$

and claim that

$$\text{there exists a tiling } t \text{ for } \mathcal{T} \quad \text{iff} \quad \tau(\mathcal{S}_\mathcal{T}) \cup \Delta_\mathcal{T} \not\models \varphi. \tag{$\dagger$}$$

Modulo $\tau(\cdot)$, the sentences from (1) express $\forall x \exists y H(x,y)$ and $\forall x \exists y V(x,y)$, which "say" that the grid is unbounded. Those from (2), express, resp., $\forall x (C_0(x) \vee \cdots \vee C_k(x))$ and $\forall x \neg (C_0(x) \wedge \cdots \wedge C_k(x))$, viz., that tiles are pairwise distinct. Sentences from (3)–(4), expressing, resp., the FO formulas $\bigwedge \{ \forall x (\exists y H(x,y) \Rightarrow C(x)) \wedge \forall x (\exists y H(y,x) \Rightarrow C'(x))) \mid (c,c') \notin \mathbb{V} \}$ and $\bigwedge \{ \forall x (\exists y V(x,y) \Rightarrow C(x)) \wedge \forall x (\exists y V(y,x) \Rightarrow C'(x))) \mid (c,c') \notin \mathbb{H} \}$, capture the adjacency constraints, thus ensuring that tiles which are not vertically (resp., horizontally) adjacent, are instead horizontally (resp., vertically) adjacent. For the reduction to proceed, we must, in addition, ensure that the grid is "closed",

i.e., that models exhibit a grid structure and no tiles fail to satisfy the adjacency constraints. This we obtain by combining φ with the sentences in (5). Such sentences express, modulo $\tau(\cdot)$, $\forall x, y(\bar{H}(x,y) \Rightarrow \neg H(x,y))$ and $\forall x, y(\bar{H}(x,y) \Rightarrow \neg H(x,y))$, resp., i.e., an (explicit) definition for $\bar{H}$, the formula $\xi := \forall x, y(\bar{H}(x,y) \Leftrightarrow \neg H(x,y))$: φ is false exactly when the formula $\chi := \forall x, y, z, w(H(x,y) \wedge V(x,z) \wedge V(y,w) \Rightarrow H(z,w))$, that enforces grid closure, is true, in every model of ξ. We now prove (†).

($\Leftarrow$) Let $\mathcal{I} = (\mathbb{D}_\mathcal{I}, \cdot^\mathcal{I})$ be a model of $\tau(\mathcal{S}_\mathcal{T}) \cup \mathcal{A}_\mathcal{T}$ s.t. $\mathcal{I} \not\models \varphi$ (and hence s.t., $\mathcal{I} \models \chi$). Define a mapping $f\colon \mathbb{N} \times \mathbb{N} \to \mathbb{D}_\mathcal{I}$ recursively as follows:

- For $i \in [0,k]$, $f(i,0) := c_i^\mathcal{I}$.
- For $i \geq k$, $f(i+1,0) :=$ some c s.t. $(f(i,0), c) \in H^\mathcal{I}$.
- For $i \geq k, j \geq 0$, $f(i+1,j+1) :=$ some c s.t. $(f(i,j), c) \in H^\mathcal{I}$.

Now, f is well-defined since *(i)* any grid point has always an $H^\mathcal{I}$-successor (since $\mathcal{I} \models \tau((1)))$, *(ii)* the grid is always closed (since $\mathcal{I} \not\models \varphi$) and *(iii)* $H^\mathcal{I}$ is non-empty (since $\mathcal{I} \models \mathcal{A}_\mathcal{T}$). Furthermore, by observing that $\mathcal{I}$ is modulo $\tau(\cdot)$ both a a model of (2)–(5) but not φ, one can prove by double induction on $(i,j) \in \mathbb{N} \times \mathbb{N}$ that

$$(f(i,j), f(i,j+1)) \in H^\mathcal{I} \quad \text{and} \quad (f(i,j), f(i+1,j)) \in V^\mathcal{I},$$

that is, f satisfies the horizontal and vertical constraints. Finally, to define the tiling $t\colon \mathbb{N} \times \mathbb{N} \to \mathbb{T}$ we put, for all $(i,j) \in \mathbb{N} \times \mathbb{N}$,

$$t(i,j) := c \quad \text{iff} \quad f(i,j) \in C^\mathcal{I}.$$

($\Rightarrow$) For the converse let t be a tiling for $\mathcal{T} = (\mathbb{T}, \mathbb{V}, \mathbb{H})$. We have to build a model $\mathcal{I}$ s.t. $\mathcal{I} \models \tau(\mathcal{S}_\mathcal{T}) \cup \mathcal{A}_\mathcal{T}$ and $\mathcal{I} \not\models \varphi$. Define $\mathcal{I}$ as follows:

- $\mathbb{D}_\mathcal{I} := \mathbb{N} \times \mathbb{N}$, $c_i^\mathcal{I} := (i,0)$, for $i \in [0,n]$, $\bar{H}^\mathcal{I} := (\mathbb{D}_\mathcal{I} \times \mathbb{D}_\mathcal{I}) \setminus H^\mathcal{I}$,
- $C_i^\mathcal{I} := \{(i,0) \in \mathbb{D}_\mathcal{I} \mid c_i \text{ is a } C_i \in \mathcal{F}_\mathcal{T}\}$, for $i \in [0,n]$,
- $H^\mathcal{I} := \{((i,j),(i,j+1)) \in \mathbb{D}_\mathcal{I} \times \mathbb{D}_\mathcal{I} \mid ((i,j),(i,j+1)) \in \mathbb{H}\}$, and
- $V^\mathcal{I} := \{((i,j),(i+1,j)) \in \mathbb{D}_\mathcal{I} \times \mathbb{D}_\mathcal{I} \mid ((i,j),(i+1,j)) \in \mathbb{V}\}$.

Clearly, $\mathcal{I}$ is the model we are looking for. □

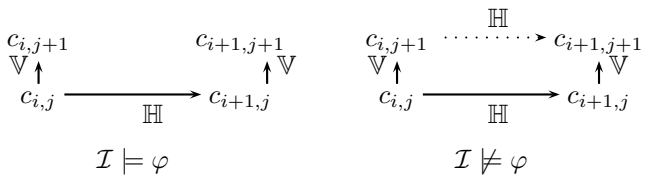

Fig. 3. If φ is satisfied, the grid stays open, closed otherwise.

	KBQA-UCQs	KBSAT
COP	in LSPACE	in LSPACE
COP+TV	in PTIME	in LSPACE
COP+TV+DTV	in PTIME	in LSPACE

	KBQA-CQs
COP+Rel+TV+RA$^+$	Undecidable

	KBQA-TCQs	KBSAT
COP+Rel	CONP-complete	in PTIME
COP+Rel+TV	CONP-complete	NP-complete
COP+Rel+DTV+TV	CONP-complete	NP-complete

Table 3. KBQA and KBSAT for the fragments of English and the positive fragments.

6 Conclusions and Related Work

We have studied the semantic data complexity of Pratt and Third's fragments of English in the contexts of structured data access (KBQA) and to a lesser degree, of declaring or updating information (KBSAT). Table 3 summarizes the data complexity results presented in this paper.

Regarding data access, we can observe that as soon as function word combinations, in either the declarations alone or the declarations and queries, expressing full Boolean conjunction and negation, yielding a so-called "Boolean-closed" fragment (i.e., expressing Boolean functions), as is the case with COP+Rel are covered by the language, the task becomes intractable. Such combinations are: either *(i)* negation in subject **NP**s or *(ii)* relatives and negation in subject **NP**s and predicate **VP**s. When, in addition to these intractable constructs, transitive verbs (expressing binary relations) and anaphoric pronouns are covered, undecidability ensues. The latter observation is particularly interesting as the fragment COP+Rel+TV+RA is decidable for plain satisfiability (Pratt and Third in [17] show that this problem is NEXPTIME-complete). See Table 3. It is also interesting to note that this result strengthens Pratt's result in [15], Th. 3, stating that KBQA for the 2-variable fragment of FO and CQs is undecidable (COP+Rel+TV+RA$^+$ is strictly subsumed by the 2-variable fragment).

Also, if membership in NP of (and a fortiori for COP+Rel+TV+RA and the fragments it subsumes) was known (cf. [15], Th. 1), it turns out that the same result can be shown for restricted 3-variable fragments such as COP+Rel+TV+DTV via resolution saturations. It can also be observed that the boundary between tractability and intractability in this case lies on whether the fragment is *both* "Boolean-closed" and expresses (via transitive verbs) binary relations.

In so doing we have identified maximal and minimal fragments (and English constructs) of known (and very expressive) controlled fragments English, tailored specifically for data management tasks, such as ACE [7], for which consistency and query answering are, resp., tractable and intractable w.r.t. data complexity.

Aknowledgements. I would wish to thank Ian Pratt-Hartmann and Diego Calvanese for the discussions that originated this paper, and the anonymous reviewers, whose comments resulted in substantial improvements.

References

1. S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison-Welsey, 1995.
2. F. Baader, D. Calvanese, D. Nardi, P. Patel-Schneider, and Deborah McGuinness. *The Description Logic Handbook*. Cambridge University Press, 2003.
3. R. Berger. *The Undecidability of the Domino Problem*. Memoirs of the American Mathematical Society. American Mathematical Society, 1966.
4. R. Bernardi, D. Calvanese, and C. Thorne. Expressing *DL-Lite* ontologies with controlled English. In *Proceedings of the 20th International Workshop on Description Logics (DL 2007)*, 2007.
5. P. Cimiano, P. Haase, J. Heizmann, M. Mantel, and R. Studer. Towards portable natural language interfaces to knowledge bases - The case of the ORAKEL system. *Data and Knowledge Engineering*, 65(2):325–354, 2008.
6. C. G. Fermüller, A. Leitsch, U. Hustadt, and T. Tammet. Resolution decision procedures. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume 2, chapter 2, pages 1791–1849. Elsevier - The MIT Press, 2001.
7. N. E. Fuchs, K. Kaljurand, and G. Schneider. Attempto Controlled English meets the challenges of knowledge representation, reasoning, interoperability and user interfaces. In *Proceedings of the 19th International Florida Artificial Intelligence Research Society Conference (FLAIRS 2006)*, 2005.
8. Y. Gurevitch, E. Grädel, and E. Börger. *The Classical Decision Problem*. Springer, 2001.
9. W. H. J. Jr. Resolution strategies as decision procedures. *Journal of the ACM*, 23(3):398–417, 1976.
10. D. Jurafsky and J. Martin. *Speech and Language Processing*. Prentice Hall, 2nd edition, 2009.
11. E. Kaufmann and A. Bernstein. How useful are natural language interfaces to the semantic web for casual end-users? In *Proceedings of the 6th International Web Conference and the 2nd Asian Web Conference (ISWC/ASWC 2007)*, pages 281–294, 2007.
12. R. Lalément. *Logique, réduction, résolution*. Dunod, 1997.
13. M. Minock, P. Olofson, and A. Näslund. Towards building robust natural language interfaces to databases. In *Proceedings of 13th International Conference on Applications of Natural Language to Information Systems (NLDB 2008)*, 2008.
14. R. Montague. Universal grammar. *Theoria*, 36(3):373–398, 1970.
15. I. Pratt-Hartmann. Data complexity of the two-variable fragment with counting quantifiers. *Information and Computation*, 207(8):867–888, 2008.
16. I. Pratt-Hartmann. Computational complexity in natural language. In *Handbook of Computational Linguistics and Natural Language Processing*, chapter 2, pages 43–73. Wiley-Blackwell, 2010.
17. I. Pratt-Hartmann and A. Third. More fragments of language. *Notre Dame Journal of Formal Logic*, 47(2):151–177, 2006.
18. E. Rich. Natural-language interfaces. *Computer*, 19(9):39–47, 1984.
19. A. Robinson. A machine-oriented logic based on the resolution principle. *Journal of the ACM*, 12(1), 1965.
20. A. Schaerf. On the complexity of the instance checking problem in concept languages with existential quantification. *Journal of Intelligent Information Systems*, 2(3):265–278, 1993.
21. J. Szymanik. The computational complexity of quantified reciprocals. In *Proceedings of the 2007 Tbilisi Colloquium (TbiLLC 2007)*, 2007.
22. C. Thorne and D. Calvanese. The data complexity of the syllogistic fragments of English. In *Proceedings of the 2009 Amsterdam Colloquium (AC 2009)*, 2010.
23. M. Vardi. The complexity of relational query languages. In *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing*, 1982.