Controlled English Ontology-Based Data Access

Camilo Thorne and Diego Calvanese

KRDB Research Centre
Free University of Bozen-Bolzano
Via della Mostra, 4, 39100 Bolzano, Italy
{calvanese,cthorne}@inf.unibz.it

Abstract. As it is well-known, querying and managing structured data in natural language is a challenging task due to its ambiguity (syntactic and semantic) and its expressiveness. On the other hand, querying, e.g., a relational database or an ontology-based data access system is a well-defined and unambigous task, namely, the task of evaluating a formal query (e.g., an SQL query) of a limited expressiveness over such database. However these formal query languages may be difficult to learn and use for the casual user and ambiguity may compromise the interface. To bridge this gap, the use of controlled language interfaces has been proposed. As a measure of their efficiency for data access, we propose to consider *data complexity*, which is the complexity of query evaluation measured *in the size of the data*. We study a familiy of controlled languages that express several fragments of OWL, ranging from tractable (**LogSpace** and **PTime**) to intractable (**coNP**-hard) in data complexity, singling out which constructs give rise to each computational property.

1 Introduction

Controlled languages (CLs) are subsets of natural language (NL) with minimal ambiguity tailored to fullfil data management tasks. Among these tasks, an important one is to provide ”lightweight” front-end languages for non-expert users of information systems [30, 6, 7]. Data in information systems such as relational databases (DBs) is inserted, updated, deleted and queried using expressive formal query and data management languages such as, e.g., SQL, which are difficult to handle (let alone learn) for casual users. Language technologies have endeavoured to fill this gap by proposing wide coverage NL interfaces (NLIs) that allow the user to use, say, everyday English, at the cost of system accuracy [2]. This is because NL is ridden with ambiguity (while formal languages are not) and thus NLIs have to choose between using complex heuristics or blowing up the translation process.

CL interfaces constitute a trade-off between the intuitive appeal of NLIs and the rigor of formal data query and management languages. Typically, CL expressions are translated symbolically into expressions belonging either to those formal languages or to some intermediate formalism (or logic) conveying a (formal) representation of its meaning (MR). Many different techniques, ranging from shallow (e.g., finite-state transducers) to deep (e.g., parsing), are used for defining such translations. Deep translations, in particular, have the property of inducing absolute accuracy without any significant

loss in translation efficiency. Indeed, deep translations are compositional, viz., the MR of a complete utterance is a function of the MRs of its (syntactic) constituents, i.e., semantics mirrors syntax. Hence, no information is lost in the translation [14]. When, in addition, the CL is context-free, compositional translations can be computed quite efficiently, i.e., we "compile" the CL in time polynomial in the length $|s|$ of the input string s.

Recently [6, 19, 25, 27, 26, 28], CL interfaces to ontology-based data access systems (OBDASs) centered mostly around the W3C standard ontology language OWL[1] (Web Ontology Language) [11, 16] have been proposed. They have given rise to a number of applications and implementations [6, 19], among which ACE-OWL [11, 16], which maps to OWL DL and fragments of it. The formal underpinning of OWL DL is provided by description logics (DLs) [3, 13].

Given that OBDASs may contain large amounts of data, we are interested in knowing whether querying such systems through CL interfaces scales to data. An ODBAS can be modelled by a DL *knowledge base* (KB), whose information is accessed through queries belonging to some fragment of SQL. A fragment that is considered sufficiently expressive but still computationally manageable (since query answering is decidable in significant cases) is that of conjunctive queries [1]. The basic problem in this setting is *(conjunctive) query answering* (QA) [8], which is a form of logical entailment [3]. To evaluate the efficiency of querying through CL interfaces, we focus on the so-called *data complexity* of QA, i.e., the computational complexity of the problem measured in terms of the size of the data only [33]. Crucially, we want to know whether CL interfaces give way to *tractable* or *intractable* QA and, more in general, whether they affect positively or negatively the performance of OBDASs.

Tractable data complexity (viz., when the data complexity of QA is in **PTime**, that is, can be decided in at most polynomial time in the size of the data) indicates good scalability to data of query answering and CL interfaces. Intractable data complexity (viz., when the data complexity of QA is at least **coNP**-hard, hence beyond **PTime**) indicates, on the other hand, low scalability. The data complexity of query answering in OWL is known to be intractable [21]. Hence, CLs like ACE-OWL do not scale to data, although they contain fragments that do. This computational behaviour depends on the language constructs they cover. It would be of interest, therefore, for CL designers working with OBDASs to know which NL constructs (and in which combination) give rise to different computational properties.

In this paper we pinpoint some of the English constructs and a fortiori of ACE, that give rise to these computational properties by studying a space of CLs that translate into several fragments of OWL. In doing so, we make the following contributions:

- We study DL-English, a CL that maps into $\mathcal{ALCI}$, an intractable DL contained in OWL that is the smallest DL closed under boolean operations on concepts [3].
- By constraining DL-English constituents, we define a family of fragments thereof, viz., the $\{\text{IS-A}_i\}_{i\in[0,7]}$ family, that are either *(i) minimal* w.r.t. intractability or *(ii) maximal* w.r.t. tractability, mirroring Pratt and Third in [23] (cf. also [29]).
- We pinpoint the NL/CL constructs that are responsible for their different computational properties, in particular, for maximal tractability and minimal intractability.

[1] `http://www.w3.org/TR/owl-ref/`

Table 1. Semantics of the DL $\mathcal{ALCI}$.

Syntax	Semantics
c	$c^{\mathcal{I}} := c$
A	$A^{\mathcal{I}} \subseteq \Delta$
$\top$	$\top^{\mathcal{I}} := \Delta$
$\exists R{:}C$	$(\exists R{:}C)^{\mathcal{I}} := \{d \mid \text{exists } d' \text{ s.t. } \langle d, d'\rangle \in R^{\mathcal{I}} \text{ and } d' \in C^{\mathcal{I}}\}$
$\neg C$	$(\neg C)^{\mathcal{I}} := \Delta \setminus C^{\mathcal{I}}$
$C \sqcap C'$	$(C \sqcap C')^{\mathcal{I}} := C^{\mathcal{I}} \cap C'^{\mathcal{I}}$
P	$P^{\mathcal{I}} \subseteq \Delta \times \Delta$
P^-	$(P^-)^{\mathcal{I}} := \{\langle d, d'\rangle \mid \langle d', d\rangle \in R^{\mathcal{I}}\}$
$C_l \sqsubseteq C_r$	$\mathcal{I} \models C_l \sqsubseteq C_r$ iff $C_l^{\mathcal{I}} \subseteq C_r^{\mathcal{I}}$
$A(c)$	$\mathcal{I} \models A(c)$ iff $c^{\mathcal{I}} \in A^{\mathcal{I}}$
$P(c, c')$	$\mathcal{I} \models P(c, c')$ iff $\langle c^{\mathcal{I}}, c'^{\mathcal{I}}\rangle \in A^{\mathcal{I}}$
$\mathcal{O}$	$\mathcal{I} \models \mathcal{O}$ iff for all $\alpha \in \mathcal{O}, \mathcal{I} \models \alpha$
$\mathcal{D}$	$\mathcal{I} \models \mathcal{D}$ iff for all $\alpha \in \mathcal{D}, \mathcal{I} \models \alpha$
$\langle \mathcal{O}, \mathcal{D}\rangle$	$\mathcal{I} \models \langle \mathcal{O}, \mathcal{D}\rangle$ iff $\mathcal{I} \models \mathcal{O}$ and $\mathcal{I} \models \mathcal{D}$

2 Ontologies and Query Answering

2.1 Ontologies, Knowledge Bases and Queries

In an OBDAS, an ontology provides a conceptual view on the data stored in a relational database. OBDASs are formally underpinned by DL knowledge bases [8, 3]. DLs are logics which structure the domain of discourse in terms of concepts (representing classes, i.e., sets of objects) and roles (representing binary relations between objects). We are interested in DLs of different expressiveness, viz., DL $\mathcal{ALCI}$ and its fragments (such as *DL-Lite* [8]). In $\mathcal{ALCI}$, *concepts* C and *roles* R are formed according to the following syntax:

$$R \longrightarrow P \mid P^- \qquad\qquad C \longrightarrow \top \mid A \mid \exists R{:}C \mid \neg C \mid C \sqcap C$$

where A stands for a concept name (a unary predicate), P for a role name (a binary predicate), and P^- for its inverse. We can enrich the set of $\mathcal{ALCI}$ concepts, modulo the following (explicit) definitions: *(i)* $\forall R{:}C := \neg\exists R{:}\neg C$, *(ii)* $C \sqcup C' := \neg(\neg C \sqcap \neg C')$, *(iii)* $\bot := \neg\top$, and *(iv)* $\exists R := \exists R{:}\top$.

In a DL ontology $\mathcal{O}$, intensional knowledge is specified by means of a set of (concept inclusion) *assertions* of the form $C_l \sqsubseteq C_r$, stating inclusion (or IS-A) between the instances of the concept C_l on the *left* and those of the concept C_r on the *right*. In $\mathcal{ALCI}$, C_l and C_r may be arbitrary concepts, while fragments of $\mathcal{ALCI}$, such as *DL-Lite* [8], can be obtained by suitably restricting the syntax for C_l and for C_r. A *database* (DB), expressing extensional knowledge, is a finite set $\mathcal{D}$ of unary and binary ground atoms of the form $A(c)$, $P(c, c')$. A *knowledge base* (KB) is a pair $\langle \mathcal{O}, \mathcal{D}\rangle$, where $\mathcal{O}$ is an ontology and $\mathcal{D}$ a DB.

The semantics of DL concepts, assertions, ontologies, and KBs is specified by considering FO *interpretations* $\mathcal{I} := \langle \Delta, \cdot^{\mathcal{I}} \rangle$, where Δ is the *domain*, a possibly countably infinite set of constants, and $\cdot^{\mathcal{I}}$ is an *interpretation* function that maps *(i)* each concept name A to a subset of the domain, *(ii)* each role name P to a binary relation over the domain, and *(iii)* each constants c to itself. It can be extended to complex concepts C and roles R by structural recursion, as shown in Table 1.

An interpretation $\mathcal{I}$ is said to be a *model* of an inclusion assertion $C_l \sqsubseteq C_r$, in symbols $\mathcal{I} \models C_l \sqsubseteq C_r$, if $C_l^{\mathcal{I}} \subseteq C_r^{\mathcal{I}}$. Similarly, $\mathcal{I}$ is said to be a model of a ground atom $A(c)$ (or $P(c, c')$), in symbols $\mathcal{I} \models A(c)$ (resp. $\mathcal{I} \models P(c, c')$), if $c \in A^{\mathcal{I}}$ (resp. $\langle c, c' \rangle \in P^{\mathcal{I}}$). It is said to be a *model* of an ontology $\mathcal{O}$, in symbols $\mathcal{I} \models \mathcal{O}$ (resp. a DB $\mathcal{D}$, in symbols $\mathcal{I} \models \mathcal{D}$) if it is a model of all of $\mathcal{O}$'s assertions (resp. $\mathcal{D}$'s ground atoms). It is said to be a *model* of a KB $\langle \mathcal{O}, \mathcal{D} \rangle$, in symbols $\mathcal{I} \models \langle \mathcal{O}, \mathcal{D} \rangle$, if it is a model of $\mathcal{O}$ and $\mathcal{D}$. Two concepts C and C' are said to be *equivalent* iff, for all $\mathcal{I}$, $C^{\mathcal{I}} = C'^{\mathcal{I}}$. Two assertions α and α' are said to be *equivalent* iff, for all $\mathcal{I}$, $\mathcal{I} \models \alpha$ iff $\mathcal{I} \models \alpha'$.

We use queries to retrieve information from KBs. As query languages, we consider *conjunctive queries* (CQs) [1], i.e., SELECT-PROJECT-JOIN SQL queries and *tree shaped conjunctive queries* (TCQs) [21], which are those CQs built using only unary and binary relations and that are tree-isomorphic.

A *conjunctive query* (CQ) q is an expression of the form

$$q(\bar{x}) \leftarrow \Phi(\bar{x}, \bar{y}),$$

where $q(\bar{x})$ is the *head*, $\bar{x}$ denotes a (finite) sequence of variables of *length* $|\bar{x}|$, the query's *distinguished variables*, and $\Phi(\bar{x}, \bar{y})$ is the *body*, a conjunction of FO relational atoms over the variables $\bar{x}$ and $\bar{y}$.

Intuitively, non-distinguished variables combined with the relational conjunctions stand for relational DB table joins and selections, and distinguished variables for the information we want to project in the result: CQs thus constitute a declarative specification of a SQL SELECT-PROJECT-JOIN query result table [1].

A *tree-shaped condition* (TSC) of root z_i, for $i \in \mathbb{N}$, is a FO formula $\varphi(z_i)$ inductively defined as follows:

- every atom $A(z_i)$ or $R(z_i, z_{i+1})$ is a TSC,
- if $\varphi(z_{i+1})$ is a TSC, so is $R(z_i, z_{i+1}) \wedge \varphi(z_{i+1})$, and
- if $\varphi(z_i)$ and $\varphi'(z_i)$ are TSCs, so is $\varphi(z_i) \wedge \varphi'(z_i)$.

Note that, for all $i \in \mathbb{N}$, $z_i \neq z_{i+1}$, i.e., each time we apply the second rule we have to introduce a new variable. Furthermore, whenever $R(z_j, z_{j+1})$, occurs in $\varphi(z_i)$, for $j \geq i$, we say that z_{j+1} is *connected* to z_j.

Every TSC $\varphi(z_i)$ rooted in z_i and variables $z_{i+1}, \ldots, z_{i+m}$ can be (bijectively) mapped to a directed adorned tree T_φ: each variable z_j, for $j \in [i, i+m]$, gives rise to a node z_j, each atom $R(z_j, z_{j+1})$ to an edge $\langle z_j, z_{j+1} \rangle$ with tag R and each atom $A(z_j)$ to tag A over node z_j [3].

A *tree-shaped conjunctive query* (TCQ) q [21] is a CQ with one distinguished variable x and non distinguished variables $y_1, \ldots, y_n$ whose body $\Phi(x, y_1, \ldots, y_n)$ can be written as a TSC $\varphi(x)$ of root x wherein y_1 is connected to x and each y_{i+1} is connected to y_i.

Let $q(\bar{x}) \leftarrow \Phi(\bar{x}, \bar{y})$ be a (T)CQ. A *grounded substitution* or grounding of q is a (not necessarily total) function $\sigma(\cdot)$ that maps the (free) variables $\bar{x} \cup \bar{y}$ of q to Δ. Groundings are extended to complex syntactic objects in the usual way. By $\sigma\theta$ we will denote the composition of grounding σ with grounding θ. We denote by $q\sigma$ the *grounding* of q's body by $\sigma(\cdot)$, i.e., $\Phi(\bar{x}, \bar{y})\sigma$. Notice that (T)CQ bodies are positive FO formulas. An interpretation $\mathcal{I}$ is said to be a *model* of $q\sigma$ whenever $\mathcal{I} \models \Phi(\bar{x}, \bar{y})\sigma$ (following standard FO semantics, based on truth and satisfaction).

We say that a sequence $\bar{c}$ of constants is an *answer* to a (T)CQ q with distinguished variables $\bar{x}$ over a KB $\langle \mathcal{O}, \mathcal{D} \rangle$, whenever there exists a grounding σ, mapping $\bar{x}$ to $\bar{c}$, s.t. $q\sigma$ is *logically entailed* by $\langle \mathcal{O}, \mathcal{D} \rangle$, viz., whenever, for all $\mathcal{I}$, $\mathcal{I} \models \langle \mathcal{O}, \mathcal{D} \rangle$ implies $\mathcal{I} \models q\sigma$; in symbols: $\langle \mathcal{O}, \mathcal{D} \rangle \models q\sigma$.

2.2 Query Answering and Data Complexity

We first recall some basic notions of complexity theory that we use in the following, and refer to [22] for more details. A *decision problem* P is a pair constituted by a set of *inputs* and a *question*, viz., a property to be verified for the inputs. Computational complexity refers to the (worst case) space and time resources an algorithm (i.e., a Turing machine) uses to solve a decision problem P, i.e., to determine whether an arbitrary input satisfies or not the problem. A *complexity class* or *computational property* is a class of decision problems. **LogSpace** denotes the class of problems P solvable by a deterministic algorithm using $\mathbf{O}(\log n)$ space, where n denotes the size of P's input(s). **NLogSpace** denotes the class of problems solvable by a non-deterministic algorithm in, again, $\mathbf{O}(\log n)$ space. **PTime** denotes the class of problems solvable by a deterministic algorithm in $\mathbf{O}(p(n))$ time, where $p(\cdot)$ denotes a polynomial function. **coNP** denotes the class of problems whose complement is solvable by a non-deterministic algorithm in $\mathbf{O}(p(n))$ time. We have that **LogSpace** $\subseteq$ **NLogSpace** $\subseteq$ **PTime** $\subseteq$ **coNP**. Higher complexity classes are defined similarly.

Given a class **C**, a problem P is said to be **C**-*hard* whenever for each problem $P' \in$ **C** there exists an algorithm, called *reduction*, that embeds P' into P and that runs in $\mathbf{O}(\log n)$ space in the size n of the input for P'. A **C**-hard problem P is as hard as any problem in **C**, possibly harder, i.e., reductions make **C** a complexity lower bound for P. If, in addition, **C** is a complexity upper bound for P, i.e., if P can be shown to be in **C**, P is said to be **C**-*complete*. Since log-space reductions are closed under composition, to show that a problem P is **C**-hard it suffices to reduce to P a problem P' that is already known to be **C**-hard.

Problems in **PTime** are said to be *tractable* and problems beyond **PTime** (e.g., **coNP**-hard problems) *intractable*, insofar as polynomial time traces the boundary up to which problems can be efficiently solved.

The *query answering* (QA) decision problem for (T)CQs and KBs is the entailment problem stated as follows:

- **Input:** a KB $\langle \mathcal{O}, \mathcal{D} \rangle$, a (T)CQ q with distinguished variables $\bar{x}$, and a sequence $\bar{c}$ of $|\bar{x}|$ constants,
- **Question:** does there exist a grounded substitution $\sigma(\cdot)$ s.t. *(i)* $\sigma(\bar{x}) = \bar{c}$ and *(ii)* $\langle \mathcal{O}, \mathcal{D} \rangle \models q\sigma$?

Table 2. An overview of some CLs.

CL (English)	Compositional	Maps to	Parser	Goal
ACE [12]	yes	FO	APE	KR/User specifications
ACE-OWL [15]	yes	OWL-DL	APE	Ontology authoring + querying
PENG [27]	yes	OWL-DL	ECOLE	Ontology authoring + querying
SOS [26]	N.A.	OWL-DL	N.A.	Ontology authoring + querying
CLCE [31]	yes	FO	compiler	KR
English Query [7]	N.A.	SQL	N.A.	DB querying/management
OWL-CNL [28]	yes	OWL-DL	DCG parser	Ontology authoring
Lite English[5]	yes	*DL-Lite*	CG parser	KR/Ontology authoring
λ-SQL [19]	yes	SQL	compiler	DB querying
nRQL [25]	yes	FO queries	DCG parser	Ontology querying
Rabbit [26]	no	OWL	GATE	Ontology authoring
ACE-PQL [6]	yes	PQL	DCG parser	Ontology querying

We are interested in the *data complexity* of QA, namely, in its computational complexity when we consider $\mathcal{D}$ as the only input of this problem [33]. The optimal data complexity for QA (w.r.t. (T)CQs) lies in **LogSpace**, which is roughly the complexity of answering SQL queries over plain DBs[2][1]. Such complexity is attained w.r.t. KBs from the *DL-Lite* fragment of the DL $\mathcal{ALCI}$ [8]. OWL and $\mathcal{ALCI}$ are, on the other hand, **coNP**-hard and **coNP**-complete, respectively [21]. Tractable data complexity implies OBDAS scalability (to data).

3 Controlled Languages

CLs have been proposed as a means of overcoming the ambiguity problem inherent to NLIs to information systems, but this is not the only knowledge representation (KR) or information management task CLs have been developed for. Other important tasks have been, e.g., *(i)* ontology authoring and *(ii)* declaring (and reasoning about) formal specifications. In such a setting, we are required to declare with CL complex structured information, such as OWL ontology assertions, or FO axiomatics. Expressive and compositional CLs such as ACE, ACE-OWL, PENG, SOS, etc., [25, 15, 27, 26, 28] have been developed with such purpose in mind. See Table 2 for an overview (CG stands for "categorial grammar", DCG for "definite clause grammar" and the other acronyms for known English parsers and/or parser APIs such as GATE).

ACE-OWL, for instance, supports English verb phrase, nominal/noun and noun pharse-coordination, (full) negation, universal, existential, and counting quantifiers, and gap-filler dependencies induced by relative sentences (although quantification, mirroring quantification in OWL, is somewhat restricted [15, 3]). Other constructs, inherently ambiguous in English, such as prepositional phrases (e.g., "John saw a man with the telescope") are, on the other hand, not supported. Nor are adjectives (attributes are expressed by ad hoc transitive verbs, as is common in DLs), intransitive verbs (atomic

[2] Data complexity for DB query evaluation is actually in $\mathbf{AC^0}$, a circuit-based complexity class that strictly includes **LogSpace** [1, 22].

concepts are expressed by means of common nouns) or query words (”who”, ”which, ”what”, etc.), although ACE-OWL can be extended to an interrogative fragments that translates into TSQs [15].

But if syntactic (i.e., parsing) complexity or ambiguity are not an issue for CLs, *semantic complexity*, however, is. *(i)* The formal model of a compositional translation for any fragment of NL, such as CLs, are formal semantics compositional translations $\tau(\cdot)$ [14, 10] wherein NL utterances are mapped to formal logic MRs, typically higher order logic (HO) or first order logic (FO) MRs. To be more precise, they are mapped to FO extended with the types, the λ-abstraction, the application and β-normalization of the simply-typed lambda calculus [10]. *(ii)* Such a translation is semantics-preserving. Observations *(i)* and *(ii)* together imply that the NL fragment will inherit both the reasoning problems and computational properties of its MR logic. Such decision problems and computational properties have been called by Pratt & Third in [23] the *semantic complexity* of an NL fragment/CL. QA is one such decision problem and data complexity one such computational property.

As a result, whenever a (logical) reasoning problem is invoked in the back-end, the semantic complexity of the (otherwise ambiguity-free and efficiently processable) front-end CL can still affect the overall performance of a information system that, like OBDASs, relies heavily on FO reasoning. Modulo $\tau(\cdot)$ such impact can be studied *construct by construct*.

4 Expressing Ontologies with Controlled English

We make use of the standard formal semantics analysis for fragments of English as developed in [10, 4, 20]. We recall the so-called HO *typing rule*, which allows us to discard (when parsing) constituents that do not yield well-typed expressions. In particular, this rule fails whenever the types of the MRs acting as premises do not unify [10]:

$$app\ \frac{\Gamma \vdash u\colon \tau \qquad \Gamma' \vdash v\colon \tau \to \tau'}{\Gamma, \Gamma' \vdash u(v)\colon \tau'}$$

where τ denotes an arbitrary type, u, v arbitrary HO expressions and Γ, Γ' typing *contexts* (sets of possibly empty variable typings). An HO expression is said to be *well-typed* whenever *(i)* it is typed and *(ii)* $\Gamma = \emptyset$.

We use *definite clause grammars* (DCGs), albeit written in phrase structure grammar style for simplicity, as grammar formalism for defining our CLs. Parsing amounts to walking or searching (depth or breadth first) a tree-shaped space of *parse states* $\langle \alpha, \varphi, \tau, \Gamma \rangle$, where α stands for a CL syntactic constituent, φ for its MR, τ for its type and Γ for its typing context. Transition between states is based on unification and type-checking [14], and is fired by the grammar rules. MRs are computed on the fly, during parsing. The states may also encode morphosyntactic information.

We consider, when defining our CLs, the following syntactic constituents. We consider as content word categories proper nouns (**Pn**s), common nouns (**N**s), transitive and intransitive verbs (**TV**s and **IV**s) and adjectives (**Adj**s). By way of function word categories, we consider determiners (**Det**s), (indeterminate) pronouns (**Pro**s), relative

pronouns (**Pro**s), conjunctions (**Crd**s) and negations (**Neg**s). Finally, regarding non-primitive (whether recursive or not) constituents, we consider verb phrases (**VP**s), noun phrases (**NP**s) nominals (**Nom**s) and complete sentences (**S**s). For simplicity we have barred out of the language, subordinate clauses, which are captured, basically, by **VP**s combining with **Relp**s and **Nom**s.

Given a constituent γ, parsing yields a unique DL MR up to *structural equivalence*. A concept C (resp. an assertion α) is said to be *structurally equivalent* to (a HO formula) $\varphi{:}e \to t$ iff C is equivalent to the DL concept φ. Following DL conventions [3], we associate (and map) the non-recursive word categories **N**, **Adj**, and **IV** to atomic concepts. Category **TV** is associated to role names. Recursive constituents, by contrast, are associated to arbitrary concepts.

We have shown elsewhere [5] how to express the optimal (for data complexity) fragment of $\mathcal{ALCI}$, *DL-Lite*. In this section we *(i)* express the **coNP**-complete (in data complexity for QA) $\mathcal{ALCI}$ and then we show *(ii)* how to define and express in CL the maximal tractable fragments of $\mathcal{ALCI}$ and the minimal intractable fragments of $\mathcal{ALCI}$. This will lead us to define in the first place the CL DL-English, expressing $\mathcal{ALCI}$, and by restricting DL-English's grammar the IS-A$_{i \in [0,7]}$ family of CLs and EL-English, viz., the minimal intractable and maximal tractable fragments of DL-English.

4.1 DL-English

Figure 1 introduces DL-English's grammar G_{DL}. For reasons of simplicity and space, we disregard morphology and polarity issues. We also omit specifying the (open) class of content words.

Lemma 1. *For all sentences S in $L(G_{DL})$, there exists an assertion α in $\mathcal{ALCI}$ s.t. $\tau(S) \equiv \alpha$.*

Proof. In order to prove this lemma we will prove something more general, namely, that,

$$\begin{array}{c}\text{for each } \mathbf{VP} \text{ or } \mathbf{Nom} \text{ constituent, there exists a concept } C \text{ in } \mathcal{ALCI} \\ \text{s.t. } \tau(\mathbf{VP}) \equiv C \text{ (resp. } \tau(\mathbf{Nom}) \equiv C).\end{array} \quad (\dagger)$$

We prove (†) by mutual induction on the length n (for $n \geq 1$) of G_{DL} derivations rooted in a **VP** or a **Nom**. We make use of unification to prune away undesired parse (sub)trees when walking through the space of parsing states, whenever (semantic) types and (morphosyntactic) features fail to unify. See Figures 2 and 3 for examples.

– ($n = 1$). We consider in this case the derivations

$$\mathbf{VP} \Rightarrow \mathbf{IV} \Rightarrow A, \quad \mathbf{Nom} \Rightarrow \text{is a } \mathbf{N} \Rightarrow \text{is a } A, \quad \mathbf{VP} \Rightarrow \text{is } \mathbf{Adj} \Rightarrow A,$$

which are in D_{DL} provided that we consider as part of our content lexicon the productions

$$\mathbf{IV} \to A, \tau(\mathbf{IV}) := A{:}e \to t, \qquad \mathbf{N} \to A, \tau(\mathbf{N}) := A{:}e \to t,$$
$$\mathbf{Adj} \to A, \tau(\mathbf{Adj}) := A{:}\ e \to t,$$

whence $\tau(\mathbf{TV}) = \tau(\mathbf{Nom}) \equiv A$, where A is an atomic concept.

(Prase structure rules)

S → **NP VP**			**NP** → **Det Nom**
VP → **TV NP**	**VP** → is a **Nom**	**VP** → is **TV** by **NP**	**NP** → **Pro Relp VP**
VP → is **Adj**	**VP** → **IV**	**VP** → is **Neg TV** by **NP**	**NP** → **Pro**
VP → does **Neg IV**	**VP** → is **Neg** a **Nom**	**Nom** → **Nom Relp VP**	**Nom** → **Adj Nom**
VP → is **Neg Adj**	**VP** → **VP Crd VP**	**Nom** → **Nom Crd Nom**	**Nom** → **N**

(Semantic actions)

$$\tau(\mathbf{S}) := \tau(\mathbf{NP})(\tau(\mathbf{VP}))$$

$\tau(\mathbf{VP}) := \tau(\mathbf{NP})(\tau(\mathbf{TV}))$	$\tau(\mathbf{VP}) := \tau(\mathbf{Crd})(\tau(\mathbf{VP}))(\tau(\mathbf{VP}))$
$\tau(\mathbf{VP}) := \tau(\mathbf{Neg})(\tau(\mathbf{NP})(\tau(\mathbf{TV})))$	$\tau(\mathbf{VP}) := \tau(\mathbf{Neg})(\tau(\mathbf{Adj}))$
$\tau(\mathbf{VP}) := \tau(\mathbf{Adj})$	$\tau(\mathbf{VP}) := \tau(\mathbf{Nom})$
$\tau(\mathbf{VP}) := \tau(\mathbf{Neg})(\tau(\mathbf{Nom}))$	$\tau(\mathbf{VP}) := \tau(\mathbf{Neg})(\tau(\mathbf{IV}))$
$\tau(\mathbf{NP}) := \tau(\mathbf{Pro})$	$\tau(\mathbf{VP}) := \tau(\mathbf{IV})$
$\tau(\mathbf{NP}) := \tau(\mathbf{Det})(\tau(\mathbf{Nom}))$	$\tau(\mathbf{NP}) := \tau(\mathbf{Pro})(\tau(\mathbf{Relp})(\tau(\mathbf{VP})))$
$\tau(\mathbf{Nom}) := \tau(\mathbf{Nom})(\tau(\mathbf{Relp})(\tau(\mathbf{VP})))$	$\tau(\mathbf{Nom}) := \tau(\mathbf{Crd})(\tau(\mathbf{Nom}))(\tau(\mathbf{Nom}))$
$\tau(\mathbf{Nom}) := \tau(\mathbf{Adj})(\tau(\mathbf{Nom}))$	$\tau(\mathbf{Nom}) := \tau(\mathbf{N})$

(Function lexicon)

Pro → anybody	$\tau(\mathbf{Pro}) := \lambda C.\lambda C'.C \sqsubseteq C'$	$(e \to t) \to ((e \to t) \to t)$
Pro → somebody	$\tau(\mathbf{Pro}) := \lambda R.\exists R$	$(e \to (e \to t)) \to (e \to t)$
Pro → nobody	$\tau(\mathbf{Pro}) := \lambda C.\lambda C'.C \sqsubseteq \neg C'$	$(e \to t) \to ((e \to t) \to t)$
Pro → nobody	$\tau(\mathbf{Pro}) := \lambda R.\neg\exists R$	$(e \to (e \to t)) \to (e \to t)$
Crd → and	$\tau(\mathbf{Crd}) := \lambda C.\lambda C'.C \sqcap C'$	$(e \to t) \to ((e \to t) \to (e \to t))$
Crd → or	$\tau(\mathbf{Crd}) := \lambda C.\lambda C'.C \sqcup C'$	$(e \to t) \to ((e \to t) \to (e \to t))$
Relp → who	$\tau(\mathbf{Relp}) := \lambda C.C$	$(e \to t) \to (e \to t)$
Relp → who	$\tau(\mathbf{Relp}) := \lambda C.\lambda C'.C : C'$	$(e \to t) \to ((e \to t) \to (e \to t))$
Neg → not	$\tau(\mathbf{Neg}) := \lambda C.\neg C$	$(e \to t) \to (e \to t)$
Pro → only	$\tau(\mathbf{Pro}) := \lambda C.\lambda R.\forall R{:}C$	$(e \to t) \to ((e \to (e \to t) \to (e \to t)$
Pro → everybody	$\tau(\mathbf{Pro}) := \lambda C.\top \sqsubseteq C$	$(e \to t) \to t$
Pro → nobody	$\tau(\mathbf{Pro}) := \lambda C.C \sqsubseteq \bot$	$(e \to t) \to t$
Det → some	$\tau(\mathbf{Det}) := \lambda C.\lambda R.\exists R{:}C$	$(e \to t) \to ((e \to (e \to t) \to (e \to t)$
Det → every	$\tau(\mathbf{Det}) := \lambda C.\lambda C'.C \sqsubseteq C'$	$(e \to t) \to ((e \to t) \to t)$
Det → no	$\tau(\mathbf{Det}) := \lambda C.\lambda C'.C \sqsubseteq \neg C'$	$(e \to t) \to ((e \to t) \to t)$

Fig. 1. The grammar G_{DL} of DL-English.

– ($n = k + 1$). By induction hypothesis, for every derivation of length $i \leq k$ rooted in **VP** or **Nom**, there exists a concept C s.t.

$$\textbf{VP} \Rightarrow_i \gamma \text{ and } \tau(\textbf{VP}) \equiv C \quad \text{or} \quad \textbf{Nom} \Rightarrow_i \gamma \text{ and } \tau(\textbf{Nom}) \equiv C, \qquad \text{(IH)}$$

where γ stands for a component derived (in G_{DL}) from **VP** (resp. **Nom**) in $i \leq k$ steps. We want to prove that the property holds for **VP** $\Rightarrow_{k+1} \gamma$ and **Nom** $\Rightarrow_{k+1} \gamma$. We have several cases to consider, namely as many as there are recursive rules for **VP** and **Nom** in G_{DL}.

- **Nom** $\Rightarrow$ **Adj Nom** $\Rightarrow_k \gamma\gamma'$. By induction hypothesis, there exists a concept C' s.t. $\tau(\gamma') \equiv C'$. Now, **Adj** is a *qualificative* adjective and we know from G_{DL} that in this case **Adj** $\Rightarrow A$ with MR $\tau(\textbf{Adj}) := \lambda D^{e\rightarrow t}.(A \sqcap D)\colon (e \rightarrow t) \rightarrow (e \rightarrow t)$. Thus,

$$\begin{aligned} \tau(A\gamma') &=_{df} \lambda D^{e\rightarrow t}.(A \sqcap D)(\tau(\gamma'))\colon e \rightarrow t \\ &=_{ih} \lambda D^{e\rightarrow t}.(A \sqcap D)(C')\colon e \rightarrow t \\ &\rhd_\beta A \sqcap C'\colon e \rightarrow t, \end{aligned}$$

  and $A \sqcap C'$ is the concept we were looking for.
- **VP** $\Rightarrow$ **TV NP** $\Rightarrow$ **TV Det Nom** $\Rightarrow_{k-1} \gamma\gamma'\gamma''$. By induction hypothesis, there exists a concept C'' s.t. $\tau(\gamma'') \equiv C''$. We know that in G_{DL} **TV** $\Rightarrow P$, with $\tau(\textbf{TV}) \equiv P$. There are only two possibilities for **Det**:

$$\textbf{Det} \Rightarrow \text{only} \qquad \text{or} \qquad \textbf{Det} \Rightarrow \text{some}.$$

  Let us focus, w.l.o.g. on the former. We know that in such a case it holds that $\tau(\textbf{Det}) := \lambda D^{e\rightarrow t}.\lambda R^{e\rightarrow(e\rightarrow t)}.(\forall R{:}C)\colon (e \rightarrow (e \rightarrow t)) \rightarrow ((e \rightarrow t) \rightarrow (e \rightarrow t))$. Therefore,

$$\begin{aligned} \tau(P \text{ only } \gamma'') &=_{df} \lambda D^{e\rightarrow t}.\lambda R^{e\rightarrow(e\rightarrow t)}.(\forall R{:}D)(\tau(\gamma''))(\tau(P))\colon e \rightarrow t \\ &=_{ih} \lambda D^{e\rightarrow t}.\lambda R^{e\rightarrow(e\rightarrow t)}.(\forall R{:}D)(C'')(P)\colon e \rightarrow t \\ &\rhd_\beta \forall P{:}C''\colon e \rightarrow t, \end{aligned}$$

  and, clearly, $\tau(\textbf{VP}) \equiv \forall P{:}C''$. Notice that any other choice for **Det** would prevent any derivation of the whole constituent. For instance, while **Det** $\Rightarrow$ every, constituent of (partial) MR $\tau(\textbf{Det}) := \lambda D^{e\rightarrow t}.\lambda E^{e\rightarrow t}.D \sqsubseteq E\colon t$, we cannot apply $\lambda E^{e\rightarrow t}.C'' \sqsubseteq E\colon e \rightarrow t$ to $P\colon e \rightarrow (e \rightarrow t)$, due to our HO type system. See Figure 3.
- All the other cases are dealt with analogously.

We now turn to complete utterances, viz. to DL-English sentences. There are three different ways in which a sentence S can be generated in our CL, namely:

– **S** $\Rightarrow$ **NP VP** $\Rightarrow$ **Det Nom VP** $\Rightarrow_* \gamma\gamma'\gamma''$ (with $S = \gamma\gamma'\gamma''$). We know that $\tau(\gamma') \equiv C'$ and that $\tau(\gamma'') \equiv C''$. Due to the typing constraints, the only possibilities for **Det** are:

$$\textbf{Det} \Rightarrow \text{every} \qquad \text{or} \qquad \textbf{Det} \Rightarrow \text{no},$$

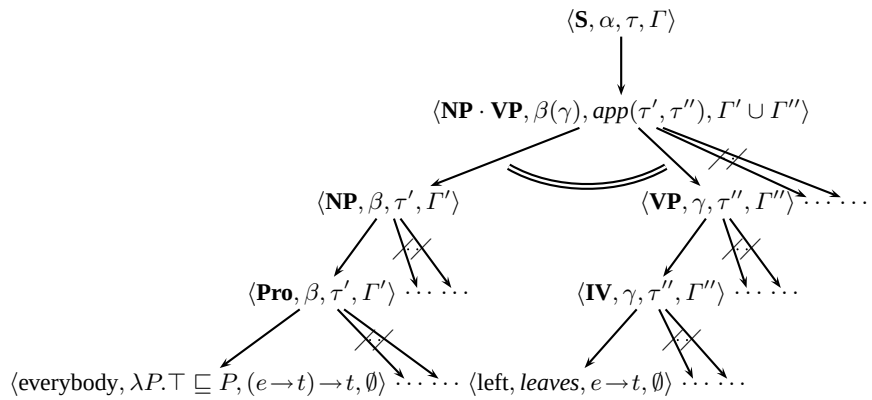

Fig. 2. A succesful derivation for "everybody left". The dots indicate transitions to failing states. The information propagated from leaves to root via unification yields, ultimately, $\langle$everybody left, $\top \sqsubseteq$ *leaves*, $t, \emptyset\rangle$, i.e., a well-typed string and MR of type sentence (or t).

with MR $\tau(\textbf{Det}) := \lambda D^{e\to t}.\lambda E^{e\to t}.D \sqsubseteq E\colon (e \to t) \to ((e \to t) \to t)$ (resp. $\tau(\textbf{Det}) := \lambda D^{e\to t}.\lambda E^{e\to t}.D \sqsubseteq \neg E\colon (e\to t)\to((e\to t)\to t)$). Hence,

$$\begin{aligned}\tau(\text{every}\gamma'\gamma'') &=_{df} \lambda D^{e\to t}.\lambda E^{e\to t}.D \sqsubseteq E(\tau(\gamma'))(\tau(\gamma''))\colon t\\ &=_{\dagger} \lambda D^{e\to t}.\lambda E^{e\to t}.D \sqsubseteq E(C')(C'')\colon t\\ &\rhd_{\beta} C' \sqsubseteq C''\colon t.\end{aligned}$$

- $\textbf{S} \Rightarrow \textbf{NP VP} \Rightarrow \textbf{Pro Relp VP VP} \Rightarrow_* \gamma\gamma'\gamma''\gamma'''$ (with $S = \gamma\gamma'\gamma''\gamma'''$). Similar argument.
- $\textbf{S} \Rightarrow \textbf{NP VP} \Rightarrow \textbf{Pro VP} \Rightarrow_* \gamma\gamma'$ (with $S = \gamma\gamma'$). Similar argument.

Therefore, for each S in $L(G_{DL})$ there exists an assertion α s.t. $\tau(S) \equiv \alpha$. This completes the proof. □

The *negation normal form* (NNF) of an $\mathcal{ALCI}$ concept C is defined by pushing negation down to atomic concepts using the De Morgan laws [3]. For every concept C in $\mathcal{ALCI}$ there exists an equivalent $\mathcal{ALCI}$ concept C' in NNF.

Lemma 2. *For each assertion α in $\mathcal{ALCI}$, there exists a sentence S in $L(G_{DL})$ s.t. (i) $\tau(S) \equiv \alpha'$, where α' is an $\mathcal{ALCI}$ assertion, and (ii) α' is equivalent to α.*

Proof. Again, in order to prove this lemma, we prove a more general claim, namely, that

$$\begin{gathered}\text{for each concept } C \text{ in NNF, there exists either a } \textbf{VP} \text{ or a } \textbf{Nom}\\ \text{s.t. } \tau(\textbf{VP}) \equiv C' \text{ or } \tau(\textbf{Nom}) \equiv C',\end{gathered} \qquad (\dagger)$$

where C' is equivalent to C. This we prove by induction on C.

- (Basis). There are two cases to consider, given that C is in NNF.
  - $C := A$. Notice that A is already in NNF. Include in the function lexicon of G_{DL} the (terminal) production $\textbf{N} \to A$ with lexical semantics $\tau(\textbf{N}) := A\colon e \to t$. There are two possibilities:

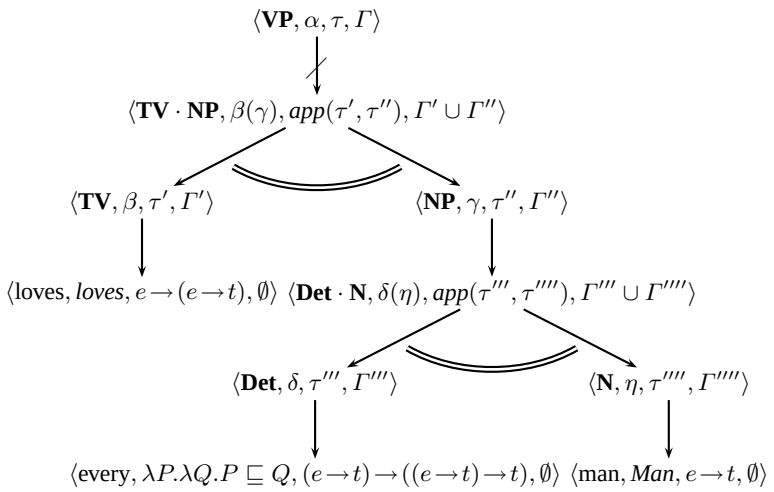

Fig. 3. A failed derivation for the **VP** "loves every man", since $app(e \to (e \to t), e \to t) = \uparrow$, whence the string is not well-typed and is devoid of a (partial) MR.

  - ∗ either **Nom** ⇒ **N** ⇒ A,
  - ∗ or **VP** ⇒ is a **Nom** ⇒ is a **N** ⇒ is a A

  Notice, furthermore that we can express $\top$ with the ad hoc **N** "thing", which is the usual DL convention [3].
- $C := \neg A$. then **VP** ⇒ is **Neg** a **Nom** ⇒ is not a **N** ⇒ is not a A, with $\tau(\mathbf{VP}) \equiv \neg A$, by the same argument as before, which is in NNF.

– (Inductive step). By inductive hypothesis we know that, for all subconcepts C' of C, there exists a component γ rooted in a **VP** or **Nom** s.t.:

$$\mathbf{VP} \Rightarrow_* \gamma \text{ and } \tau(\mathbf{VP}) \equiv C'' \quad \text{or} \quad \mathbf{Nom} \Rightarrow_* \gamma \text{ and } \tau(\mathbf{Nom}) \equiv C'', \qquad \text{(IH)}$$

where C'' is a concept equivalent to C'. This leaves two cases to consider, namely:

- $C := \exists P{:}C'$. By (IH), there exists a γ' s.t. either **VP** $\Rightarrow_* \gamma'$ and $\tau(\mathbf{VP}) \equiv C''$, or **Nom** $\Rightarrow_* \gamma'$, $\tau(\mathbf{Nom}) \equiv C''$, and C'' is equivalent to C'. This gives us two possibilities for $\exists P{:}C'$, namely:
  - ∗ **VP** ⇒ **TV NP** ⇒ **TV Det Nom** $\Rightarrow_*$ P some γ', with $\tau(\mathbf{VP}) \equiv \exists P{:}C'$, which is equivalent to $\exists P{:}C$, or
  - ∗ **VP** ⇒ **TV NP** ⇒ **TV Pro Relp VP** $\Rightarrow_*$ P somebody who γ', where $\tau(\mathbf{VP}) \equiv \exists P{:}C'$, which is equivalent to $\exists P{:}C$.
- $C := C' \sqcap C''$. Similar argument.

Let now $C \sqsubseteq C'$ be an $\mathcal{ALC}$ assertion with C, C' in NNF. We can capture this assertion in one of two ways in DL-English:

– either **S** $\Rightarrow_*$ every **Nom VP** $\Rightarrow_*$ every $\gamma\gamma'$,
– or **S** $\Rightarrow_*$ everybody who **VP VP** $\Rightarrow_*$ everybody who $\gamma\gamma'$,

for some (two) components γ' and γ' whose existence is guaranteed by (†). Clearly, in both cases $\tau(\mathbf{S}) \equiv C'' \sqsubseteq C'''$. Moreover, it is evident that $C'' \sqsubseteq C'''$ is equivalent to $C \sqsubseteq C'$. This completes the proof. □

From Lemmas 1 and 2, it follows immediately that DL-English expresses $\mathcal{ALCI}$ (up to equivalence).

Theorem 1 (DL-English). *DL-English expresses $\mathcal{ALCI}$.*

Corollary 1. *QA for DL-English and (T)CQs is **coNP**-complete in data complexity.*

Example 1. Examples of sentences in DL-English (we spell out the MRs underneath) are:

No man who runs some business that does not make some money is shrewd.
$$\textit{Man} \sqcap \exists \textit{run}{:}(\textit{Business} \sqcap \neg(\exists \textit{make}{:}\textit{Money})) \sqsubseteq \neg \textit{Shrewd} \quad (1)$$

Nobody eats only apples.
$$\forall \textit{eats}{:}\textit{Apple} \sqsubseteq \bot \quad (2)$$

Everybody sleeps.
$$\top \sqsubseteq \textit{Sleep} \quad (3)$$

Every married man has some wife.
$$\textit{Man} \sqcap \textit{Married} \sqsubseteq \exists \textit{has}{:}\textit{Wife} \quad (4)$$

Anybody who has some car drives some new car or old car.
$$\exists \textit{has}{:}\textit{Car} \sqsubseteq \exists \textit{drives}{:}((\textit{Car} \sqcap \textit{New}) \sqcup ((\textit{Car} \sqcap \textit{Old}) \quad (5)$$

4.2 The family $\{\text{IS-A}_i\}_{i\in[0,7]}$ of CLs

We now turn to the computational properties of each of the constructs of DL-English *in isolation*. We do it by essentially restricting the kind of *right* (i.e., C_r) and *left* (i.e., C_l) concepts we may express. All utterances comply with the sentence patterns

"every γ_l γ_r" and "everybody who γ_l γ_r".

The constituents γ_l and γ_r map to, respectively, left and right concepts, while sentences map to IS-A assertions of the form $C_l \sqsubseteq C_r$. We consider in this paper only 8 out of all possible combinations obtained by allowing in C_l and C_r some subset of the DL constructs in Table 3, giving rise to the family $\{\text{IS-A}_i\}_{i\in[0,7]}$ of CLs shown in Table 4. The basic kind of assertion they all express is IS-A among atomic concepts, viz., $A \sqsubseteq A'$, captured by IS-A_0. Notice that this can be easily achieved by, so to speak, merging the grammars from Table 3 expressing (in isolation) the C_ls and C_rs occuring in the DL assertions of Table 4 into a grammar for each CL IS-A_i.

Theorem 2 (IS-A_is). *For each $i \in [0, 7]$ and each sentence S_i in IS-A_i, there exists an assertion α_i s.t. $\tau(S_i) \equiv \alpha_i$. Conversely, for each assertion α_i there exists a sentence S_i in IS-A_i s.t. $\tau(S_i) \equiv \alpha_i'$ and α_i' is equivalent to α_i.*

Table 3. Expressing concepts C_f, for $f \in \{l, r\}$, and assertions $C_l \sqsubseteq C_r$, by restricting and subcategorizing rules in G_{DL}.

$$\mathbf{S} \longrightarrow \mathbf{NP}_l\,\mathbf{VP}_r \quad \mathbf{NP}_l \longrightarrow \mathbf{Pro}_l\,\mathbf{Relp}_l\,\mathbf{VP}_l \quad \mathbf{NP}_l \longrightarrow \mathbf{Det}_l\mathbf{Nom}_l$$
$$\mathbf{Pro}_l \longrightarrow \text{anybody} \quad \mathbf{Relp}_l \longrightarrow \text{who} \quad \mathbf{Det}_l \longrightarrow \text{every}$$

Concept C_f	**Constituent** γ_f	**Grammar Rules**
$\exists P{:}A$	**TV** some $\mathbf{Nom}_f$ **TV** somebody who $\mathbf{VP}_f$	$\mathbf{VP}_f \longrightarrow$ is a $\mathbf{Nom}_f$ | **IV** | is **Adj** $\mathbf{Nom}_f \longrightarrow$ **N**
$\exists P^{-}{:}A$	is **TV** by some $\mathbf{Nom}_f$ is **TV** by somebody who $\mathbf{VP}_f$	$\mathbf{VP}_f \longrightarrow$ is a $\mathbf{Nom}_f$ | **IV** | is **Adj** | **TV** $\mathbf{NP}_f$ $\mathbf{Nom}_f \longrightarrow$ **N** $\mathbf{NP}_f \longrightarrow \mathbf{Det}_f\,\mathbf{Nom}_f$ | $\mathbf{Pro}_f\,\mathbf{Relp}_f\,\mathbf{VP}_f$ $\mathbf{Det}_f \longrightarrow$ some $\mathbf{Relp}_f \longrightarrow$ who $\mathbf{Pro}_f \longrightarrow$ somebody
$\forall P{:}A$	**TV** only $\mathbf{VP}_f$ **TV** only who $\mathbf{VP}_f$	$\mathbf{VP}_f \longrightarrow$ is a $\mathbf{Nom}_f$ | **IV** | is **Adj** | **TV** $\mathbf{NP}_f$ $\mathbf{Nom}_f \longrightarrow$ **N** $\mathbf{NP}_f \longrightarrow \mathbf{Det}_f\,\mathbf{Nom}_f$ | $\mathbf{Pro}_f\,\mathbf{Relp}_f\,\mathbf{VP}_f$ $\mathbf{Det}_f \longrightarrow$ only $\mathbf{Relp}_f \longrightarrow$ who $\mathbf{Pro}_f \longrightarrow$ only
A	$\mathbf{Nom}_f$ $\mathbf{VP}_f$	$\mathbf{VP}_f \longrightarrow$ is a $\mathbf{Nom}_f$ | **IV** | is **Adj** $\mathbf{Nom}_f \longrightarrow$ **N**
$A_1 \sqcap \ldots \sqcap A_n$	**Adj** $\mathbf{Nom}_f$ $\mathbf{Nom}_f$ who $\mathbf{VP}_f$ $\mathbf{Nom}_f$ and $\mathbf{Nom}_f$ $\mathbf{VP}_f$ and $\mathbf{VP}_f$	$\mathbf{VP}_f \longrightarrow$ is a $\mathbf{Nom}_f$ | **IV** | is **Adj** | $\mathbf{VP}_f\,\mathbf{Crd}_f\,\mathbf{VP}_f$ $\mathbf{Nom}_f \longrightarrow$ **N** | **Adj** $\mathbf{Nom}_f$ | $\mathbf{Nom}_f\,\mathbf{Crd}_f\,\mathbf{Nom}_f$ | $\mathbf{Nom}_f\mathbf{Relp}_f\,\mathbf{VP}_f$ $\mathbf{Relp}_f \longrightarrow$ who $\mathbf{Crd}_f \longrightarrow$ and
$\exists P$	**TV** something **TV** somebody	$\mathbf{VP}_f \longrightarrow$ **TV** $\mathbf{Pro}_f$ $\mathbf{Pro}_f \longrightarrow$ somebody | something
$A_1 \sqcup \ldots \sqcup A_n$	$\mathbf{VP}_f$ or $\mathbf{VP}_f$	$\mathbf{VP}_f \longrightarrow$ is a $\mathbf{Nom}_f$ | **IV** | is **Adj** | $\mathbf{VP}_f$ and $\mathbf{VP}_f$ $\mathbf{Nom}_f \longrightarrow$ **N** | $\mathbf{Nom}_f$ and $\mathbf{Nom}_f$
$\neg A$	is not **Adj** does not **IV** is not a $\mathbf{Nom}_f$	$\mathbf{Nom}_f \longrightarrow$ **N** $\mathbf{VP}_f \longrightarrow$ does not **IV** | is not **Adj** | is not a $\mathbf{Nom}_f$

Table 4. Defining the $\{\text{IS-A}_i\}_{i\in[0,7]}$ CLs. IS-A_i, for all $i > 0$, contains also the assertions of IS-A_0.

	Assertions α_i	**Example(s)**
IS-A_0	$A \sqsubseteq A_1 \sqcap \ldots \sqcap A_n$	Every businessman is a cunning man is a cunning man.
IS-A_1	$A \sqsubseteq \forall P{:}A$	Every herbivorous eats only herbs eats only herbs.
IS-A_2	$A_1 \sqcap \ldots \sqcap A_n \sqsubseteq \forall P{:}(A_1 \sqcap \ldots \sqcap A_k)$	Every Italian man drinks only strong coffee.
IS-A_3	$\exists P{:}A \sqsubseteq A_1 \sqcap \cdots \sqcap A_n$ $\exists P^{-}{:}A \sqsubseteq A_1 \sqcap \cdots \sqcap A_n$ $A \sqsubseteq \exists P$	Anybody who murders some person is a heartless killer. Anybody who is loved by some person is a happy person. Every driver drives something.
IS-A_4	$A_1 \sqcap \ldots \sqcap A_n \sqsubseteq A_1 \sqcap \ldots \sqcap A_k$ $\exists P{:}(A_1 \sqcap \ldots \sqcap A_n) \sqsubseteq A_1 \sqcap \ldots \sqcap A_k$	Every cruel man is a bad man. Anybody who runs some bankrupt company is a bad businessman.
IS-A_5	$\forall P{:}A \sqsubseteq A_1 \sqcap \ldots \sqcap A_n$	Anybody who values only money is a greedy person.
IS-A_6	$A \sqsubseteq A_1 \sqcup \ldots \sqcup A_n$	Every mammal is male or is female.
IS-A_7	$\neg A \sqsubseteq A_1 \sqcap \ldots \sqcap A_n$	Anybody who is not selfish is a reasonable person.

Proof. The theorem can be proved for each fragment in a manner analogous to DL-English. Basically, we consider two cases both on the "$\Rightarrow$" and the "$\Leftarrow$" directions of the proof, viz., a (mutual) induction on either *(i)* $\mathbf{Nom}_l$ and $\mathbf{VP}_l$ constituents or *(ii)* on the $\mathbf{Nom}_r$ and $\mathbf{VP}_r$ constituents for the if direction and a (mutual) induction over *(i)* a C_l or *(ii)* a C_r concept for the only if direction. With some routine adjustments to the specific syntax of the fragments and their MRs, we adapt each time the proof of Theorem 1.

4.3 Data Complexity of the IS-A_is

In [5], we have shown how to express the DL *DL-Lite* for which QA (w.r.t. (T)CQs) is in **LogSpace** [8], with the CLs Lite-English. We want now to know which fragments of ACE-OWL and DL-English are *(i) maximal* w.r.t. tractable data complexity (i.e., in **PTime**), and hence scale to data, and *(ii) minimal* w.r.t. intractable data complexity (i.e., **coNP**-hard), and hence do not scale.

Theorem 3. *The data complexity of QA for (T)CQs is (i) in* ***LogSpace*** *for IS-A_0, (ii)* ***PTime****-complete for IS-A_2, IS-A_3, and IS-A_4 and (iii)* ***coNP****-complete for IS-A_5, IS-A_6, and IS-A_7.*

Proof. The CL IS-A_0 is subsumed by the CL Lite-English, which as we have shown elsewhere [5] expresses the DL *DL-Lite* for which QA w.r.t. CQs is in **LogSpace** in data complexity ([8], Theorem 2).

The lower bounds for IS-A_2, IS-A_3, and IS-A_4 follow from the results in [8]. For IS-A_2 the result is derived from Theorem 7, case 2. For IS-A_3, it is derived from Theorem 6, case 1. Finally, the lower bound for IS-A_4 follows from Theorem 7, case 3. Basically, this is because our CLs subsume the DLs for which those theorems hold. **PTime**-hardness in all three cases holds already for atomic queries. The complexity upper bounds, on the other hand, follow from results in [18] for the DL $\mathcal{EL}$, which subsumes the DL assertions IS-A_2, IS-A_3 and IS-A_4.

The lower bounds for IS-A_5, IS-A_6, and IS-A_7 follow also from [8]: for IS-A_5, we apply Theorem 8, case 3; for IS-A_6, we apply Theorem 8, case 2; and for IS-A_7, Theorem 8, case 1. In these three cases, TCQs are used to define a reduction from the **NP**-complete satisfiability problem for 2+2-clauses (defined any studied by [24]). The **coNP** upper bounds for these fragments, on the other hand, derive from the **coNP** data complexity upper bounds for QA over expressive DLs (containing $\mathcal{ALCI}$) shown in [21] and hold, again, for CQs. □

Theorem 4 (Data complexity of IS-A_1). *QA for IS-A_1 and (T)CQs is **NLogSpace**-complete w.r.t. data complexity.*

Proof. (**Hardness**) Calvanese et al. show in [8] (by reduction from the reachability problem for directed graphs) that any DL capable of expressing assertions of the form $A \sqsubseteq \forall P.A'$, or, equivalently, of the form $\exists P^{-}.A \sqsubseteq A'$, is **NLogSpace**-hard for QA. This result holds already for atomic queries. Note that such assertions are expressed in our fragments by sentences of the form "Every A Ps only A's", rather than by sentences like "Every A Ps every A'".

(**Membership**) Let $q(\bar{x}) \leftarrow \Phi(\bar{x}, \bar{y})$ be a *fixed* CQ, $\bar{c}$ a *fixed* tuple, $\mathcal{O}$ a *fixed* set of universally quantified IS-A_1 MRs and $\mathcal{D}$ a set of facts s.t. $\#(\mathcal{D}) = n$. We will reduce QA for IS-A_1 to QA for linear Datalog, which is known to be in **NLogSpace** in data complexity [1]. The only inclusion assertions expressible in our fragment are $A \sqsubseteq B$ and $\exists R.A \sqsubseteq B$, which can be transformed into an (equivalent) set $\mathcal{P}_\mathcal{O}$ of clauses $\neg A(x) \vee B(x)$ and $\neg R(x, y) \vee \neg A(x) \vee B(y)$, called a linear Datalog *program*. On the other hand, CQ q might not be a linear Datalog goal, but its body $\Phi(\bar{x}, \bar{y})$ [1] consists of a conjunction of m atoms $A_1(\bar{z}_1) \wedge \cdots \wedge A_m(\bar{z}_m)$, where $\bar{x} \cup \bar{y} = \bar{z}_1 \cup \cdots \cup \bar{z}_m$. Note that, since q is fixed, m is constant (a fixed positive integer), and so are $|\bar{y}|$ and $|\bar{z}_i|$, for $i \in [1, m]$. If we were to transform such atoms into a family of atomic queries (which are linear Datalog goals), by means of some satisfaction-preserving reduction that requires only $\mathbf{O}(\log n)$ space, the data complexity upper bound would immediately follow.

Start by computing the program $\mathcal{P}_\mathcal{O}$ as described above. Since $\mathcal{O}$ is fixed, transforming it into $\mathcal{P}_\mathcal{O}$ does not affect data complexity. We transform now $\Phi(\bar{x}, \bar{y})$, in space logarithmic in n, into a family of linear Datalog goals, thus reducing answering q over $\mathcal{O}$ and $\mathcal{D}$ to answering a family of atomic goals over $\mathcal{P}_\mathcal{O}$ and $\mathcal{D}$. Let $[\bar{c}/\bar{z}]$ denote the grounding mapping $\bar{z}$ to the constants $\bar{c}$. Ground q by $\sigma := [\bar{c}/\bar{x}]$. Grounding q by σ, which returns the CQ of body $\Psi(\bar{c}, \bar{y})$, does not affect, once again, data complexity. Next, consider all the possible groundings $[\bar{c}'/\bar{y}]$ with $\bar{c}' \in \mathit{adom}(\mathcal{D})^{|\bar{y}|}$ and apply them to $\Psi(\bar{c}, \bar{y})$. There are $\mathbf{O}(n^{|\bar{y}|})$ such groundings. This yields a family of CQs of body $\Psi(\bar{c}, \bar{c}')$, whose atoms can be stored in a registry of $\mathbf{O}(\log n)$ size (we can encode such

Table 5. Summmary of data complexity results. For a comparison, note that ACE-OWL is **coNP**-hard (upper bounds for OWL-DL are not, as yet, known) [21].

CL	Data Complexity ((T)CQs)	CL	Data Complexity ((T)CQs)
IS-A_0	**LogSpace**	EL-English	**PTime**-complete
IS-A_1	**NLogSpace**-complete	IS-A_5	**coNP**-complete
IS-A_2	**PTime**-complete	IS-A_6	**coNP**-complete
IS-A_3	**PTime**-complete	IS-A_7	**coNP**-complete
IS-A_4	**PTime**-complete	DL-English	**coNP**-complete

grounded atoms using $\mathbf{O}(\log n)$ bits). This reduction is sound and complete. Indeed

$$\langle \mathcal{O}, \mathcal{D} \rangle \models \Psi(\bar{c}, \bar{y}) \text{ iff } \mathcal{P}_{\mathcal{O}} \cup \mathcal{D} \models A_i(\bar{c}''), \text{ for all } i \in [1,k] \text{ and some } \bar{c}'' \in \mathit{adom}(\mathcal{D})^{|\bar{z}_i|} \text{ "compatible" with } \bar{c}, \quad (\dagger)$$

where by "compatible" we mean that $\bar{c}''$ coincides with $\bar{c}$ on the distinguished variables (note that $\bar{z}_i$ may contain *both* distinguished and non-distinguished variables).

The "$\Rightarrow$" direction is immediate. To prove the "$\Leftarrow$" direction, we reason as follows. Assume for contradiction that there exists an interpretation $\mathcal{I}$ s.t. $\mathcal{I} \models \mathcal{P}_{\mathcal{O}} \cup \mathcal{D}$ but $\mathcal{I} \not\models A_i(\bar{c}'')$, for some $i \in [1,k]$ and every $\bar{c}'' \in \mathit{adom}(\mathcal{D})^{|\bar{z}_i|}$ "compatible", again, with $\bar{c}$. Since $\mathcal{I} \models \mathcal{P}_{\mathcal{O}} \cup \mathcal{D}$, we have that $\mathcal{I} \models \langle \mathcal{O}, \mathcal{D} \rangle$ and $\mathcal{I} \models \Phi(\bar{c}, \bar{y})$. Therefore, for some grounding θ from $\bar{y}$ into $\mathit{adom}(\mathcal{D})$, $\mathcal{I} \models A_i(\bar{z}_i)\sigma\theta$. Now, clearly, $\bar{c}'' = \bar{c}''_1 \cup \bar{c}''_2$ with $\bar{c}''_2 \subseteq \bar{c}$, and $\bar{z}_i = \bar{z}_{i1} \cup \bar{z}_{i2}$ with $\bar{z}_{i2} \subseteq \bar{x}$. Therefore, $\theta(\bar{z}_{i2}) \in \mathit{adom}(\mathcal{D})^{|\bar{z}_{i2}|}$. On the other hand, $\mathcal{I} \not\models A_i(\bar{c}'')$, for all $\bar{c}''$. Hence, $\theta(\bar{z}_{i2}) \neq \bar{c}''_2$ and $\theta(\bar{z}_{i2}) \notin \mathit{adom}(\mathcal{D})^{|\bar{z}_{i2}|}$. Contradiction.

The algorithm then proceeds by looping $\mathbf{O}(n^{|\bar{y}|})$ times over the $A(c'')$s (stored in the $\mathbf{O}(\log n)$ registry), checking each time whether, for all $i \in [1,k]$, there exists some "compatible" $\bar{c}''$ s.t. $\mathcal{P}_{\mathcal{O}} \cup \mathcal{D} \models A_i(\bar{c}'')$ holds. For each atom the algorithm runs a linear Datalog non-deterministic check that uses at most $\mathbf{O}(\log n)$ space. Clearly, such a non-deterministic algorithm decides QA using, overall, at most $\mathbf{O}(\log n)$ space. □

We can now individuate the constructs of DL-English, and a fortiori of any CL expressing a **coNP**-hard ontology language that negatively affect the scalability of CL interfaces to ODBASs, namely:

- "only" in subject position (**coNP**-hardness of IS-A_5),
- disjunction in predicate position (**coNP**-hardness of IS-A_6),
- negation in subject position (**coNP**-hardness of IS-A_7).

4.4 EL-English: A Maximal Tractable Fragment of DL-English

The constructs from Fig 3 also allow us to identify *maximal* CLs contained in DL-English (and a fortiori ACE-OWL) w.r.t. scalability (i.e., tractable data complexity). By merging the (tractable) fragments IS-A_i, for $i \leq 4$, we essentially express the $\mathcal{ELI}$ ontology language, with syntax (notice that the assertion $A \sqsubseteq \forall P\colon A'$ is equivalent to $\exists P^-\colon A \sqsubseteq A'$):

$$R \rightarrow P \mid P^- \qquad\qquad C \rightarrow \top \mid A \mid \exists R\colon C \mid C \sqcap C$$

That is, the DL where negation- and disjunction free existential concepts are allowed to arbitrarily nest on *both* sides of $\sqsubseteq$. QA for $\mathcal{ELI}$ is **PTime**-complete in data complexity for (T)CQs [18] and thus induces a **PTime**-complete fragment of DL-English, that we term EL-English, which captures most of the constraints and axioms of real-world large-scale biomedical ontologies such as GALEN or SNOWMED [18]. EL-English is defined top-down by removing from G_{DL} the grammar rules for negation, disjunction, and universal quantification, and the negative function words. Whence:

Proposition 1. *QA for EL-English and (T)CQs is **PTime**-complete in data complexity.*

In such a CL arbitrary sentence subordination (and relatives), in combination with, existential quantification and conjunction among **VP**s or **Nom**s is allowed. Universal quantification is highly controlled and negation and disjunction are ruled out.

4.5 Discussion

In general, intractability (w.r.t. data complexity) will arise in every CL that induces a partitioning of the data of the OBDAS's DB. This will happen whenever their MRs can simulate full negation, conjunction and disjunction: the technical **coNP**-hardness results from [8], on which we ground our own work, are based on this intuition. We can thus say that CLs like IS-A_5, IS-A_6, IS-A_7, and a fortiori DL-English (which contains them all) are "booleanly closed". EL-English, on the other hand, by being a negation-free fragment of DL-English, remains tractable.

5 Conclusions

In this paper we have studied the data complexity of querying OBDASs in controlled English. We have shown that optimal (i.e., **LogSpace**) and tractable (i.e., **PTime**) data complexity (w.r.t. (T)CQs) is basically attained by constraining the behaviour of certain function words, namely those expressing universal restrictions, negations and disjunctions of concepts and roles in the underlying DL ontologies. They must not be allowed to occur *both* in the subject and predicate of CL sentences. Moreover, they can only occur in carefully sought positions of CL sentences. This is the case with the CLs IS-A_0, IS-A_1, IS-A_2, IS-A_3, IS-A_4, and EL-English. Relaxing the constraints on negation, disjunction, universal determiners, and pronouns, as in IS-A_5, IS-A_6 and IS-A_7, until the components in sentence subjects and predicates become symmetrical, as in DL-English (or more expressive CLs), yields **coNP**-hardness. Table 6 summarizes the computational properties associated to each such combination of (controlled) English function words.

Computational complexity will be higher if we consider *combined complexity* i.e., when we consider as inputs not only the data but also the ontology (and possibly the query). Satisfiability for $\mathcal{ALCI}$ is **ExpTime**-complete (**ExpTime** is the class of problems solvable by a deterministic algorithm in exponential time) [3]. Since DL-English expresses $\mathcal{ALCI}$ and for this DL QA of TCQs (but not CQs) reduces to unsatisfiability, this entails that QA for TCQs is **ExpTime**-complete. Instead, QA for CQs is 2**ExpTime**-complete [9, 17] in $\mathcal{ALCI}$, and hence in DL-English.

Table 6. Data complexity of CL constructs.

Tractable (**PTime** or less)	negation in predicate **VP**s, relatives in predicate **VP**s, conjunction in predicate **VP**s	relatives and conjunction in subject **NP**s and predicate **VP**s, but no negation
Intractable (**coNP**-hard)	relatives and negation in subject **NP**s and in predicate **VP**s	negation in subject **NP**s "only" in subject **NP**s

As related work we must mention the computational complexity results regarding the satisfiability problem for several fragments of (and CLs obtained from) English by Slavkovic [29] and Pratt and Third [23], from which upper and lower complexity bounds for the combined complexity of QA can be derived in some cases. Such fragments and CLs are orthogonal in expressiveness to the CLs studied in this paper. In the case of Pratt and Third's declarative fragments, termed fragments of English (FOEs), we have provided in [32] *data complexity* results for QA (albeit w.r.t. TCQs only). In particular, we show that "booleanly closed" FOEs are intractable not only w.r.t. combined complexity and/or satisfiability, but exhibit intractable data complexity for QA.

Acknowledgements. We thank Raffaella Bernardi and Norbert Fuchs for discussions and criticism regarding earlier versions of this paper.

References

1. S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison-Welsey, 1995.
2. I. Androutsopoulos, G. D. Ritchie, and P. Thanisch. Natural language interfaces to databases - An introduction. *Journal of Natural Language Engineering*, 1(1):29–81, 1995.
3. F. Baader, D. Calvanese, D. Nardi, P. Patel-Schneider, and Deborah McGuinness. *The Description Logic Handbook*. Cambridge University Press, 2003.
4. J. Barwise and R. Cooper. Generalized quantifiers and natural language. *Linguistics and Philosophy*, 4(2):159–219, 1980.
5. R. Bernardi, D. Calvanese, and C. Thorne. Lite natural language. In *Proc. of the 7th Int. Workshop on Computational Semantics (IWCS-7)*, 2007.
6. A. Bernstein, E. Kaufman, A. Göring, and C. Kiefer. Querying ontologies: A controlled English interface for end-users. In *Proc. of the 4th Int. Semantic Web Conf. (ISWC 2005)*, 2005.
7. A. Blum. Microsoft English Query 7.5. automatic extraction of semantics from relational databases and OLAP cubes. In *Proc. of the 25th Int. Conf. on Very Large Databases (VLDB 1999)*, 1999.
8. D. Calvanese, G. de Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Data complexity of query answering in description logics. In *Proc. of the 10th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2006)*, 2006.
9. D. Calvanese, G. De Giacomo, and M. Lenzerini. On the decidability of query containment under constraints. In *Proc. of the 17th ACM Symposium on Principles of Database Systems (PODS 1998)*, pages 149–158, 1998.
10. B. Carpenter. *Type-Logical Semantics*. The MIT Press, 1997.
11. N. E. Fuchs and K. Kaljurand. Mapping Attempto Controlled English to OWL-DL. In *Demos and Posters of the 3rd European Semantic Web Conf. (ESWC 2006)*, 2006.

12. N. E. Fuchs, K. Kaljurand, and G. Schneider. Attempto Controlled English meets the challenges of knowledge representation, reasoning, interoperability and user interfaces. In *Proc. of the 19th Int. Florida Artificial Intelligence Research Society Conf. (FLAIRS 2006)*, 2005. Available at `http://www.ifi.unizh.ch/attempto/publications/`.
13. I. Horrocks, P. F. Patel-Schneider, and F. van Harmelen. From $\mathcal{SHIQ}$ and RDF to OWL: The making of a web ontology language. *Journal on Web Semantics*, 1(1):7–26, 2003.
14. D. Jurafsky and J. Martin. *Speech and Language Processing*. Prentice Hall, 2000.
15. K. Kaljurand. *Attempto Controlled English as a Semantic Web Language*. PhD thesis, University of Tartu, 2007. Available at `http://attempto.ifi.uzh.ch/site/pubs/`.
16. K. Kaljurand and N. E. Fuchs. Verbalizing OWL in Attempto Controlled English. In *Proc. of 3rd OWL Experiences and Directions Workshop (OWLED 2007)*, 2007.
17. C. Lutz. The complexity of conjunctive query answering in expressive description logics. In *Proc. of the 4th Int. Joint Conf. on Automated Reasoning (IJCAR 2008)*, volume 5195 of *Lecture Notes in Artificial Intelligence*, pages 179–193. Springer, 2008.
18. C. Lutz and A. Krisnadhi. Data complexity in the $\mathcal{EL}$ family of DLs. In *Proc. of the 20th Int. Workshop on Description Logics (DL 2007)*, 2007.
19. S. Mador-Haim, Y. Winter, and A. Braun. Controlled language for geographical information system queries. In *Proc. of Inference in Computational Semantics 2006*, 2006.
20. R. Montague. Universal grammar. *Theoria*, 36(3):373–398, 1970.
21. M. Ortiz, D. Calvanese, and T. Eiter. Data complexity of query answering in expressive description logics. *Journal of Automated Reasoning*, 41(1):61–98, 2008.
22. C. Papadimitrou. *Computational Complexity*. Addison Wesley - Longman, 1994.
23. I. Pratt and A. Third. More fragments of language. *Notre Dame Journal of Formal Logic*, 47(2):151–177, 2006.
24. 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.
25. R. Schwitter. Creating and querying linguistically motivated ontologies. In *Proc. of the 2008 Knowledge Representation Workshop (KROW 2008)*, 2008.
26. R. Schwitter, K. Kaljurand, A. Cregan, C. Dolbear, and G. Hart. A comparison of three controlled natural languages for OWL 1.1. In *Proc. of the 4th Int. Workshop on OWL: Experiences and Directions (OWLED 2008)*, Washington, Apr. 2008.
27. R. Schwitter, A. Ljungberg, and D. Hood. ECOLE - A look-ahead editor for a controlled language. In *Proc. of the 8th Int. Workshop of the European Association for Machine Translation and the 4th Controlled Language Applications Workshop (EAMT/CLAW 2003)*, 2003.
28. R. Schwitter and M. Tilbrook. Let's talk in description logic via controlled language. In *Proc. of the 3rd Int. Workshop on Logic and Engineering of Natural Language Semantics (LENLS 2006)*, 2006.
29. M. Slavkovik. Deep analysis for an interactive question answering system. Master's thesis, Free University of Bozen-Bolzano, 2007.
30. J. Sowa. *Knowledge Representation: Logical, Philosophical and Computational Foundations*. Brooks Cole Publishing Co., 1999.
31. J. Sowa. Common Logic Controlled English (draft). Available at `http://www.jfsowa.com/clce/specs.htm`, 2004.
32. C. Thorne and D. Calvanese. The data complexity of the syllogistic fragments of english. In *Proc. of the 2009 Amsterdam Colloquium (AC 2009)*, 2009.
33. M. Vardi. The complexity of relational query languages. In *Proc. of the 14th Annual ACM Symposium on Theory of Computing (STOC 1982)*, 1982.