Description Logics with Abstraction and Refinement

Carsten Lutz[1,2] , **Lukas Schulze**[1]
[1] Department of Computer Science, Leipzig University, Germany
[2]Center for Scalable Data Analytics and Artificial Intelligence (ScaDS.AI)
{clu, lschulze}@informatik.uni-leipzig.de

Abstract

Ontologies often require knowledge representation on multiple levels of abstraction, but description logics (DLs) are not well-equipped for supporting this. We propose an extension of DLs in which abstraction levels are first-class citizens and which provides explicit operators for the abstraction and refinement of concepts and roles across multiple abstraction levels, based on conjunctive queries. We prove that reasoning in the resulting family of DLs is decidable while several seemingly harmless variations turn out to be undecidable. We also pinpoint the precise complexity of our logics and several relevant fragments.

1 Introduction

Abstraction and refinement is an important topic in many subfields of computer science such as systems verification (Dams and Grumberg 2018). The same is true for ontology design because ontologies often refer to different levels of abstraction (or equivalently, levels of granularity). To name only one example, the widely known medical ontology SNOMED CT contains the concepts Arm, Hand, Finger, Phalanx, Osteocyte, and Mitochondrion which intuitively all belong to different (increasingly finer) levels of abstraction (Stearns et al. 2001). Existing ontology languages, however, do not provide explicit support for modeling across different abstraction levels. The aim of this paper is to introduce a family of description logics (DLs) that provide such support in the form of abstraction and refinement operators.

We define the *abstraction DL* $\mathcal{ALCHI}^{\mathsf{abs}}$ as an extension of the familiar description logic $\mathcal{ALCHI}$, which may be viewed as a modest and tame core of the OWL 2 DL ontology language. In principle, however, the same extension can be applied to any other DL, both more and less expressive than $\mathcal{ALCHI}$. Abstraction levels are explicitly named and referred to in the ontology, and we provide explicit operators for the abstraction and refinement of concepts and roles. For example, the concept refinement

$$L_2{:}q_A \ \underline{\mathsf{refines}}\ L_1{:}\mathsf{Arm},$$

where q_A denotes the conjunctive query (CQ)

$$q_A = \mathsf{UArm}(x_1) \wedge \mathsf{LArm}(x_2) \wedge \mathsf{Hand}(x_3) \wedge \mathsf{joins}(x_1, x_2) \wedge \mathsf{joins}(x_2, x_3),$$

expresses that every instance of Arm on the coarser abstraction level L_1 decomposes into an ensemble of three objects on the finer level L_2 as described by q_A. Concept abstractions are dual to refinements and state that every ensemble of a certain kind on a finer level gives rise to an abstracting object on a coarser level. Semantically, there is one classical DL interpretation for each abstraction level and a (partial) refinement function that associates objects on coarser levels with ensembles on finer levels. Every object may participate in at most one ensemble and we require that abstraction levels are organized in the form of a tree. We believe that the abstraction DLs defined along these lines are useful for many application domains, examples are given in the paper. Though not limited to it, our DLs are particularly well-suited for capturing the mereological (part-whole) aspect of abstraction and refinement (Artale et al. 1996).

Base DL	cr	ca	rr	ra	Complexity
$\mathcal{ALCHI}$	X				in EXPTIME
$\mathcal{ALCHI}$	X	X	X	X	in 2EXPTIME
$\mathcal{ALC}$		X			2EXPTIME-hard
$\mathcal{ALC}$			X		2EXPTIME-hard
$\mathcal{ALC}$				X	2EXPTIME-hard

Figure 1: The complexity of satisfiability in abstraction DLs.

Our main technical contribution is to show that adding abstraction and refinement to $\mathcal{ALCHI}$ preserves decidability of the base logic, and to provide a detailed analysis of its complexity, also considering many relevant fragments. It turns out that satisfiability in $\mathcal{ALCHI}^{\mathsf{abs}}$ is 2EXPTIME-complete. Note that this is in line with the fact that CQ evaluation in $\mathcal{ALCHI}$ is 2EXPTIME-complete (Lutz 2008). For the fragments, however, such parallels to evaluation cease to hold. We use $\mathcal{ALCHI}^{\mathsf{abs}}[\text{cr}]$ to denote $\mathcal{ALCHI}^{\mathsf{abs}}$ in which only concept refinement is admitted, and likewise ca denotes concept abstraction and rr, ra denote role refinement and abstraction. We recall that $\mathcal{ALC}$ is $\mathcal{ALCHI}$ without inverse roles and role hierarchies.

We show that satisfiability in the natural fragment $\mathcal{ALCHI}^{\mathsf{abs}}[\text{cr}]$ is only EXPTIME-complete, despite the fact that it still comprises CQs. Moreover, 2EXPTIME-hardness already holds for $\mathcal{ALC}^{\mathsf{abs}}$ in contrast to the fact that CQ evaluation in $\mathcal{ALC}$ is in EXPTIME. There are actually three different sources of complexity as satisfiability is 2ExpTime-hard already in each of the fragments $\mathcal{ALC}^{\mathsf{abs}}$[ca], $\mathcal{ALC}^{\mathsf{abs}}$[rr], and $\mathcal{ALC}^{\mathsf{abs}}$[ra]. In $\mathcal{ALC}^{\mathsf{abs}}$[ra], role abstractions allow us to recover inverse roles. The same is true for $\mathcal{ALC}^{\mathsf{abs}}$[ca] that, however, requires a more subtle reduction relying on the fact that ensembles must not overlap. Finally, $\mathcal{ALC}^{\mathsf{abs}}$[rr] is 2ExpTime-hard because role refinements allow us to generate objects interlinked in a complex way. See Figure 1 for a summary.

We then observe that the decidability of $\mathcal{ALCHI}^{\mathsf{abs}}$ is more fragile than it may seem on first sight and actually depends on a number of careful design decisions. In particular, we consider three natural extensions and variations and show that each of them is undecidable. The first variation is to drop the requirement that abstraction levels are organized in a tree. The second is to add the requirement that ensembles (which are tuples rather than sets) must not contain repeated elements. And the third is to drop the requirement that CQs in abstraction and refinement statements must be full, that is, to admit quantified variables in such CQs.

Proof details are in the appendix, provided in (Lutz and Schulze 2023).

Related Work. A classic early article on granularity in AI is (Hobbs 1985). Granularity has later been studied in the area of foundational ontologies, focussing on a philosophically adequate modeling in first-order logic. Examples include granular partitions (Bittner and Smith 2003), the descriptions and situations framework (Gangemi and Mika 2003), and domain-specific approaches (Fonseca et al. 2002; Schulz, Boeker, and Stenzhorn 2008; Vogt 2019).

Existing approaches to representing abstraction and refinement / granularity in DLs and in OWL are rather different in spirit. Some are based on rough or fuzzy set theory (Klinov, Taylor, and Mazlack 2008; Lisi and Mencar 2018), some provide mainly a modeling discipline (Calegari and Ciucci 2010), some aim at the spatial domain (Hbeich, Roxin, and Bus 2021) or at speeding up reasoning (Glimm, Kazakov, and Tran 2017), and some take abstraction to mean the translation of queries between different data schemas (Cima et al. 2022). We also mention description logics of context (Klarman and Gutiérrez-Basulto 2016); an abstraction level can be seen as a context, but the notion of context is more general and governed by looser principles. A categorization of different forms of granularity is in (Keet 2008).

There is a close connection between DLs with abstraction as proposed in this paper and the unary negation fragment of first-order logic (UNFO). In fact, UNFO encompasses ontologies formulated in DLs such as $\mathcal{ALCI}$ and conjunctive queries. UNFO satisfiability is decidable and 2ExpTime-complete (Segoufin and ten Cate 2013). This does, however, not imply any of the results in this paper due to the use of refinement functions in the semantics of our DLs and the fact that UNFO extended with functional relations is undecidable (Segoufin and ten Cate 2013).

2 Preliminaries

Base DLs. Fix countably infinite sets $\mathbf{C}$ and $\mathbf{R}$ of *concept names* and *role names*. A *role* is a role name or an *inverse role*, that is, an expression r^- with r a role name. If $R = r^-$ is an inverse role, then we set $R^- = r$. *$\mathcal{ALCI}$-concepts* C, D are built according to the syntax rule

$$C, D ::= A \mid \neg C \mid C \sqcap D \mid C \sqcup D \mid \exists R.C \mid \forall R.C$$

where A ranges over concept names and R over roles. We use $\top$ as an abbreviation for $A \sqcup \neg A$ with A a fixed concept name and $\bot$ for $\neg\top$. An *$\mathcal{ALC}$-concept* is an $\mathcal{ALCI}$-concept that does not use inverse roles and an *$\mathcal{EL}$-concept* is an $\mathcal{ALC}$-concepts that uses none of $\neg$, $\sqcup$, and $\forall$.

An *$\mathcal{ALCHI}$-ontology* is a finite set of *concept inclusions (CIs)* $C \sqsubseteq D$ with C and D $\mathcal{ALCI}$-concepts and *role inclusions (RIs)*, $R \sqsubseteq S$ with R, S roles. The letter $\mathcal{I}$ indicates the presence of inverse roles and $\mathcal{H}$ indicates the presence of role inclusions (also called role hierarchies), and thus it should also be clear what we mean e.g. by an $\mathcal{ALCI}$-ontology and an $\mathcal{ALCH}$-ontology. An *$\mathcal{EL}$-ontology* is a finite set of CIs $C \sqsubseteq D$ with C, D $\mathcal{EL}$-concepts.

An *interpretation* is a pair $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ with $\Delta^\mathcal{I}$ a non-empty set (the *domain*) and $\cdot^\mathcal{I}$ an *interpretation function* that maps every concept name $A \in \mathbf{C}$ to a set $A^\mathcal{I} \subseteq \Delta^\mathcal{I}$ and every role name $r \in \mathbf{R}$ to a binary relation $r^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$. The interpretation function is extended to compound concepts as usual, c.f. (Baader et al. 2017). An interpretation $\mathcal{I}$ *satisfies* a CI $C \sqsubseteq D$ if $C^\mathcal{I} \subseteq D^\mathcal{I}$ and an RI $R \sqsubseteq S$ if $R^\mathcal{I} \subseteq S^\mathcal{I}$. It is a *model* of an ontology $\mathcal{O}$ if it satisfies all CIs and RIs in it.

For any syntactic object O such as an ontology or a concept, we use $||O||$ to denote the *size* of O, that is, the number of symbols needed to write O over a suitable alphabet.

Conjunctive Queries. Let $\mathbf{V}$ be a countably infinite set of variables. A *conjunctive query (CQ)* takes the form $q(\bar{x}) = \exists \bar{y}\, \varphi(\bar{x}, \bar{y})$ with φ a conjunction of *concept atoms* $C(x)$ and *role atoms* $r(x, y)$, C a (possibly compound) concept, r a role name, and x, y variables from $\bar{x} \cup \bar{y}$. We may write $\alpha \in q$ to indicate that α is an atom in φ and $r^-(x, y) \in q$ in place of $r(y, x) \in q$. The variables in $\bar{x}$ are the *answer variables* of q. We require that every answer variable x occurs in some atom of q, but omit this atom in writing in case it is $\top(x)$. With $\mathsf{var}(q)$, we denote the set of all (answer and quantified) variables in q. If q has no answer variables then it is *Boolean*. We mostly restrict our attention to CQs q that are *full*, meaning that q has no quantified variables. A CQ q is *connected* if the undirected graph with node set $\mathsf{var}(q)$ and edge set $\{\{v, v'\} \mid r(v, v') \in q \text{ for any } r \in \mathbf{R}\}$ is. A CQ q is a *subquery* of a CQ q' if q can be obtained from q' by dropping atoms.

Let $q(\bar{x}) = \exists \bar{y}\, \varphi(\bar{x}, \bar{y})$ be a CQ and $\mathcal{I}$ an interpretation. A mapping $h : \bar{x} \cup \bar{y} \to \Delta^\mathcal{I}$ is a *homomorphism* from q to $\mathcal{I}$ if $C(x) \in q$ implies $h(x) \in C^\mathcal{I}$ and $r(x, y) \in q$ implies $(h(x), h(y)) \in r^\mathcal{I}$. A tuple $\bar{d} \in (\Delta^\mathcal{I})^{|\bar{x}|}$ is an *answer* to q on $\mathcal{I}$ if there is a homomorphism h from q to $\mathcal{I}$ with $h(\bar{x}) = \bar{d}$. We use $q(\mathcal{I})$ to denote the set of all answers to q on $\mathcal{I}$. If q is Boolean, we write $\mathcal{I} \models q$ to indicate the existence of a homomorphism from q to $\mathcal{I}$.

3 DLs with Abstraction and Refinement

We extend $\mathcal{ALCHI}$ to the DL $\mathcal{ALCHI}^{\mathsf{abs}}$ that supports abstraction and refinement. Fix a countable set $\mathbf{A}$ of *abstrac-*

tion levels. An $\mathcal{ALCHI}^{\mathsf{abs}}$-ontology is a finite set of statements of the following form:

- *labeled concept inclusions* $C \sqsubseteq_L D$,
- *labeled role inclusions* $R \sqsubseteq_L S$,
- *concept refinements* $L{:}q(\bar{x})$ $\underline{\mathsf{refines}}$ $L'{:}C$,
- *concept abstractions* $L'{:}C$ $\underline{\mathsf{abstracts}}$ $L{:}q(\bar{x})$,
- *role refinements* $L{:}q(\bar{x},\bar{y})$ $\underline{\mathsf{refines}}$ $L'{:}q_R(x,y)$,
- *role abstractions* $L'{:}R$ $\underline{\mathsf{abstracts}}$ $L{:}q(\bar{x},\bar{y})$

where L, L' range over $\mathbf{A}$, C, D over $\mathcal{ALCI}$-concepts, R, S over roles, q over full conjunctive queries, and q_R over full conjunctive queries of the form $C_1(x) \wedge R(x,y) \wedge C_2(y)$. In concept and role abstraction statements, we additionally require the CQ q to be connected. We may write $C \equiv_L D$ as shorthand for the two CIs $C \sqsubseteq_L D$ and $D \sqsubseteq_L C$. We underline abstraction and refinement operators to ensure better readability throughout the paper.

Intuitively, a concept refinement $L{:}q(\bar{x})$ $\underline{\mathsf{refines}}$ $L'{:}C$ expresses that any instance of C on abstraction level L' refines into an *ensemble* of $|\bar{x}|$ objects on abstraction level L which satisfies all properties expressed by CQ q. Conversely, a concept abstraction $L'{:}C$ $\underline{\mathsf{abstracts}}$ $L{:}q(\bar{x})$ says that any ensemble of $|\bar{x}|$ objects on abstraction level L that satisfies q abstracts into a single instance of C on abstraction level L'. Role refinements and abstractions can be understood in a similar way, where each of the two elements that participate in a role relationship refines into its own ensemble.

Note that in role refinements, we consider CQs $q_R = C_1(x) \wedge R(x,y) \wedge C_2(y)$ rather than only the role R. This is because roles are often of a general kind such as partOf or $\mathsf{interactsWith}$ and need proper context to be meaningfully refined. This context is provided by the concepts C_1, C_2.

Example 1. *Granularity is important in many domains. Anatomy has already been mentioned in the introduction. The concept refinement given there may be complemented by choosing q_A as in the introduction and adding the concept abstraction*

$$L_1{:}\mathsf{Arm}\ \underline{\mathsf{abstracts}}\ L_2{:}q_A.$$

We next consider bikes as a simple example for a technical domain. Let us first say how wheels refine into components: $L_2{:}q_W$ $\underline{\mathsf{refines}}$ $L_1{:}\mathsf{Wheel}$ *where*

$$q_W = \mathsf{Axle}(x_1) \wedge \mathsf{Spokes}(x_2) \wedge \mathsf{Rim}(x_3) \wedge \mathsf{Tire}(x_4) \wedge \mathsf{join}(x_2,x_1) \wedge \mathsf{join}(x_2,x_3) \wedge \mathsf{carries}(x_3,x_4).$$

We may then use the following role refinement to express how frames connect to wheels:

$$L_2{:}q_{FW}\ \underline{\mathsf{refines}}\ L_1{:}\,\mathsf{Wheel}(x) \wedge \mathsf{connTo}(x,y) \wedge \mathsf{Frame}(y)$$

where, for $\bar{x} = x_1 \cdots x_4$ and $\bar{y} = y_1 \cdots y_7$ (assuming that frames have seven components),

$$q_{FW}(\bar{x},\bar{y}) = \mathsf{Axle}(x_1) \wedge \mathsf{connTo}(x_1,y_1) \wedge \mathsf{Dropout}(y_1).$$

This expresses that if a wheel is connected to a frame, then the axle of the wheel is connected to the dropout of the frame.

Extensions $\mathcal{L}^{\mathsf{abs}}$ of other DLs $\mathcal{L}$ introduced in Section 2, such as $\mathcal{ALC}$ and $\mathcal{ALCH}$, may be defined in the expected way. We also consider various fragments of $\mathcal{ALCHI}^{\mathsf{abs}}$. With $\mathcal{ALCHI}^{\mathsf{abs}}$[cr,rr], for example, we mean the fragment of $\mathcal{ALCHI}^{\mathsf{abs}}$ that admits concept refinement and role refinement, but neither concept abstraction nor role abstraction (identified by ca and ra).

We next define the semantics of $\mathcal{ALCHI}^{\mathsf{abs}}$, based on *A-interpretations* which include one traditional DL interpretation for each abstraction level. Formally, an A-interpretation takes the form $\mathcal{I} = (\mathbf{A}_{\mathcal{I}}, \prec, (\mathcal{I}_L)_{L \in \mathbf{A}_{\mathcal{I}}}, \rho)$, where

- $\mathbf{A}_{\mathcal{I}} \subseteq \mathbf{A}$ is the set of relevant abstraction levels;
- $\prec\ \subseteq \mathbf{A}_{\mathcal{I}} \times \mathbf{A}_{\mathcal{I}}$ is such that the directed graph $(\mathbf{A}_{\mathcal{I}}, \{(L',L) \mid L \prec L'\})$ is a tree; intuitively, $L \prec L'$ means that L is less abstract than L' or, in other words, that the modeling granularity of L is finer than that of L';
- $(\mathcal{I}_L)_{L \in \mathbf{A}_{\mathcal{I}}}$ is a collection of interpretations $\mathcal{I}_L$, one for every $L \in \mathbf{A}_{\mathcal{I}}$, with pairwise disjoint domains; we use $L(d)$ to denote the unique $L \in \mathbf{A}_{\mathcal{I}}$ with $d \in \Delta^{\mathcal{I}_L}$;
- ρ is the *refinement function*, a partial function that associates pairs $(d,L) \in \Delta^{\mathcal{I}} \times \mathbf{A}_{\mathcal{I}}$ such that $L \prec L(d)$ with an *L-ensemble* $\rho(d,L)$, that is, with a non-empty tuple over $\Delta^{\mathcal{I}_L}$. We want every object to participate in only one ensemble and thus require that

($*$) for all $d \in \Delta^{\mathcal{I}}$, there is at most one $e \in \Delta^{\mathcal{I}}$ such that d occurs in $\rho(e, L(d))$.

For readability, we may write $\rho_L(d)$ in place of $\rho(d,L)$.

An A-interpretation $\mathcal{I} = (\mathbf{A}_{\mathcal{I}}, \prec, (\mathcal{I}_L)_{L \in \mathbf{A}_{\mathcal{I}}}, \rho)$ *satisfies* a

- labeled concept or role inclusion $\alpha \sqsubseteq_L \beta$ if $L \in \mathbf{A}_{\mathcal{I}}$ and $\alpha^{\mathcal{I}_L} \subseteq \beta^{\mathcal{I}_L}$;
- concept refinement $L{:}q(\bar{x})$ $\underline{\mathsf{refines}}$ $L'{:}C$ if $L \prec L'$ and for all $d \in C^{\mathcal{I}_{L'}}$, there is an $\bar{e} \in q(\mathcal{I}_L)$ such that $\rho_L(d) = \bar{e}$;
- concept abstraction $L'{:}C$ $\underline{\mathsf{abstracts}}$ $L{:}q(\bar{x})$ if $L \prec L'$ and for all $\bar{e} \in q(\mathcal{I}_L)$, there is a $d \in C^{\mathcal{I}_{L'}}$ s.t. $\rho_L(d) = \bar{e}$;
- role refinement $L{:}q(\bar{x},\bar{y})$ $\underline{\mathsf{refines}}$ $L'{:}q_R(x,y)$ if $L \prec L'$ and for all $(d_1,d_2) \in q_R(\mathcal{I}_{L'})$, there is an $(\bar{e}_1,\bar{e}_2) \in q(\mathcal{I}_L)$ such that $\rho_L(d_1) = \bar{e}_1$ and $\rho_L(d_2) = \bar{e}_2$;
- role abstraction $L'{:}R$ $\underline{\mathsf{abstracts}}$ $L{:}q(\bar{x},\bar{y})$ if $L \prec L'$ and for all $(\bar{e}_1,\bar{e}_2) \in q(\mathcal{I}_L)$, there is a $(d_1,d_2) \in R^{\mathcal{I}_{L'}}$ such that $\rho_L(d_1) = \bar{e}_1$ and $\rho_L(d_2) = \bar{e}_2$.

An A-interpretation is a *model* of an $\mathcal{ALCHI}^{\mathsf{abs}}$-ontology if it satisfies all inclusions, refinements, and abstractions in it.

Example 2. *We consider the domain of (robotic) actions. Assume that there is a* Fetch *action that refines into subactions:* $L_2{:}q_F$ $\underline{\mathsf{refines}}$ $L_1{:}\mathsf{Fetch}$ *where*

$$q_F = \mathsf{Locate}(x_1) \wedge \mathsf{Move}(x_2) \wedge \mathsf{Grasp}(x_3) \wedge \mathsf{precedes}(x_1,x_2) \wedge \mathsf{precedes}(x_2,x_3).$$

We might have a safe version of the fetching action and a two-handed grasping action:

$$\mathsf{SFetch} \sqsubseteq_{L_1} \mathsf{Fetch}$$
$$\mathsf{TwoHandedGrasp} \sqsubseteq_{L_1} \mathsf{Grasp}$$

A safe fetch requires a two-handed grasping subaction: $L_2{:}q_S$ $\underline{\mathsf{refines}}$ $L_1{:}\mathsf{SFetch}$ *where for $\bar{x} = x_1x_2x_3$,*

$$q_S(\bar{x}) = \mathsf{TwoHandedGrasp}(x_3).$$

We remark that abstraction statements need to be used with care since ensembles may not overlap, c.f. Condition $(*)$. For example, the reader may want to verify that the following CI and concept abstraction have no model:

$$\top \sqsubseteq_{L_2} \exists r.\exists r.\top \qquad L_1{:}\top \ \underline{\textsf{abstracts}}\ L_2{:}r(x,y).$$

We are interested in the problem of *(concept) satisfiability* which means to decide, given an $\mathcal{ALCHI}^{\textsf{abs}}$-ontology $\mathcal{O}$, an $\mathcal{ALCI}$-concept C, and an abstraction level $L \in \mathbf{A}$, whether there is a model $\mathcal{I}$ of $\mathcal{O}$ such that $C^{\mathcal{I}_L} \neq \emptyset$. We then say that C is *L-satisfiable w.r.t. $\mathcal{O}$*. As usual, the related reasoning problems of subsumption can be reduced to satisfiability in polynomial time, and vice versa (Baader et al. 2017).

4 Upper Bounds

We prove that satisfiability in $\mathcal{ALCHI}^{\textsf{abs}}$ is decidable in 2ExpTime. Before approaching this general case, however, we consider the fragment $\mathcal{ALCHI}^{\textsf{abs}}[\text{cr}]$ and show that it is only ExpTime-complete.

4.1 $\mathcal{ALCHI}^{\textsf{abs}}[\text{cr}]$ in ExpTime

Our aim is to prove the following.

Theorem 1. *Satisfiability in $\mathcal{ALCHI}^{\textsf{abs}}[\text{cr}]$ is ExpTime-complete.*

The lower bound is inherited from $\mathcal{ALCHI}$ without abstraction and refinement (Baader et al. 2017). We prove the upper bound by a mosaic-based approach, that is, we decide the existence of a model $\mathcal{I}$ by trying to assemble $\mathcal{I}$ from small fragments called mosaics. Essentially, a mosaic describes a single ensemble on a single level of abstraction.

Assume that we are given as input an $\mathcal{ALCHI}^{\textsf{abs}}[\text{cr}]$-ontology $\mathcal{O}$, an $\mathcal{ALCI}$-concept C_0, and an abstraction level L_0. We may assume w.l.o.g. that C_0 is a concept name as we can extend $\mathcal{O}$ with $A_0 \sqsubseteq_{L_0} C_0$ and test satisfiability of the fresh concept name A_0. We also assume w.l.o.g. that $\mathcal{O}$ is in *normal form*, meaning that

1. every CI has one of the forms
$$\top \sqsubseteq_L A \qquad A \sqsubseteq_L \exists R.B \qquad \exists R.B \sqsubseteq_L A$$
$$A_1 \sqcap A_2 \sqsubseteq_L A \qquad A \sqsubseteq_L \neg B \qquad \neg B \sqsubseteq_L A$$
where A, A_1, A_2, B are concept names and R is a role;
2. in every concept refinement $L{:}q(\bar{x})$ $\underline{\textsf{refines}}$ $L'{:}C$, C is a concept name, and so is D in all concept atoms $D(x) \in q$.

It is in fact routine to show that every $\mathcal{ALCHI}^{\textsf{abs}}[\text{cr}]$ ontology $\mathcal{O}$ can be converted in polynomial time into an $\mathcal{ALCHI}^{\textsf{abs}}[\text{cr}]$ ontology $\mathcal{O}'$ in normal form that is a conservative extension of $\mathcal{O}$, see e.g. (Manière 2022). We also assume that (i) $\mathcal{O}$ contains $R \sqsubseteq_L R$ for all roles R and abstraction levels L in $\mathcal{O}$, (ii) $R \sqsubseteq_L S$, $S \sqsubseteq_L T \in \mathcal{O}$ implies $R \sqsubseteq_L T \in \mathcal{O}$, and (iii) $R \sqsubseteq_L S \in \mathcal{O}$ implies $R^- \sqsubseteq_L S^- \in \mathcal{O}$. With $\prec$ we denote the smallest relation on $\mathbf{A}_\mathcal{O}$ such that $L \prec L'$ for all $L{:}q(\bar{x})$ $\underline{\textsf{refines}}$ $L'{:}C$ in $\mathcal{O}$.

Fix a domain Δ of cardinality $||\mathcal{O}||$. A *mosaic* is a pair $M = (L, \mathcal{I})$ where $L \in \mathbf{A}_\mathcal{O}$ is the abstraction level of the mosaic and $\mathcal{I}$ is an interpretation with $\Delta^\mathcal{I} \subseteq \Delta$ such that $\mathcal{I}$ satisfies all CIs $C \sqsubseteq_L D$ in $\mathcal{O}$ and all RIs $R \sqsubseteq_L S$ in $\mathcal{O}$, with the possible exception of CIs of the form $A \sqsubseteq_L \exists r.B$. We may write L^M to denote L, and likewise for $\mathcal{I}^M$. Let $\mathcal{M}$ be a set of mosaics. We say that a mosaic $M = (L, \mathcal{I})$ is *good* in $\mathcal{M}$ if for all $d \in \Delta^\mathcal{I}$ the following hold:

1. if $A \sqsubseteq_L \exists R.B \in \mathcal{O}$, $d \in A^\mathcal{I}$, and $d \notin (\exists R.B)^\mathcal{I}$, then there is an $M' = (L, \mathcal{I}') \in \mathcal{M}$ and a $d' \in \Delta^{\mathcal{I}^{M'}}$ such that
   (a) $d' \in B^{\mathcal{I}'}$,
   (b) if $\exists S.A \sqsubseteq_L B \in \mathcal{O}$, $R \sqsubseteq_L S \in \mathcal{O}$, and $d' \in A^{\mathcal{I}'}$, then $d \in B^\mathcal{I}$;
   (c) if $\exists S.A \sqsubseteq_L B \in \mathcal{O}$, $R^- \sqsubseteq_L S \in \mathcal{O}$, and $d \in A^\mathcal{I}$, then $d' \in B^{\mathcal{I}'}$;
2. for every level $L' \in \mathbf{A}_\mathcal{O}$ such that
$$Q = \{q \mid L'{:}q(\bar{x})\ \underline{\textsf{refines}}\ L{:}A \in \mathcal{O} \text{ and } d \in A^\mathcal{I}\} \neq \emptyset,$$
there is a mosaic $M' \in \mathcal{M}$ with $M' = (L', \mathcal{I}')$ and a tuple $\bar{e}$ over $\Delta^{\mathcal{I}'}$ such that $\bar{e} \in q(\mathcal{I}')$ for all $q \in Q$.

We now formulate the actual decision procedure. If the directed graph $(\mathbf{A}_\mathcal{O}, \{(L', L) \mid L \prec L'\})$ is not a tree, we directly return 'unsatisfiable'. Our algorithm first computes the set $\mathcal{M}_0$ of all mosaics for $\mathcal{O}$ and then repeatedly and exhaustively eliminates mosaics that are not good. Let $\mathcal{M}^*$ denote the set of mosaics at which this process stabilizes.

Lemma 1. *C_0 is L_0-satisfiable w.r.t. $\mathcal{O}$ iff $\mathcal{M}^*$ contains (i) a mosaic M with $L^M = L_0$ and $C_0^{\mathcal{I}^M} \neq \emptyset$ and (ii) a mosaic M with $L^M = L$, for every L in $\mathbf{A}_\mathcal{O}$.*

The algorithm thus returns 'satisfiable' if Conditions (i) and (ii) from Lemma 1 are satisfied and 'unsatisfiable' otherwise. It is easy to see that the algorithm runs in single exponential time.

4.2 $\mathcal{ALCHI}^{\textsf{abs}}$ in 2ExpTime

Our aim is to prove the following.

Theorem 2. *Satisfiability in $\mathcal{ALCHI}^{\textsf{abs}}$ is decidable in 2ExpTime.*

A matching lower bound will be provided later on. We prove Theorem 2 by a mosaic-based approach which is, however, significantly more complex than the one used in the previous section. In particular, a mosaic now represents a 'slice' through an A-interpretation that includes multiple abstraction levels and multiple ensembles.

Assume that we are given as input an $\mathcal{ALCHI}^{\textsf{abs}}$-ontology $\mathcal{O}$, an $\mathcal{ALCI}$-concept C_0, and an abstraction level L_0. We again assume that C_0 is a concept name. and $\mathcal{O}$ is in normal form, defined as in the previous section, but with the obvious counterparts of Point 2 for role refinements and (concept and role) abstractions. We also define the relation $\prec$ on $\mathbf{A}_\mathcal{O}$ as in the previous section, except that we now consider concept and role refinements, as well as concept and role abstractions, in the obvious way. Again, if the directed graph $(\mathbf{A}_\mathcal{O}, \{(L', L) \mid L \prec L'\})$ is not a tree, then we directly return 'unsatisfiable'.

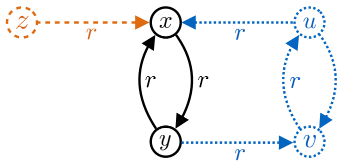
Fix a set Δ of cardinality $||\mathcal{O}||^{||\mathcal{O}||}$. A *mosaic* is a tuple
$$M = ((\mathcal{I}_L)_{L \in \mathbf{A}_\mathcal{O}}, \rho, f_{\textsf{in}}, f_{\textsf{out}})$$
where

- $(\mathcal{I}_L)_{L\in\mathbf{A}_\mathcal{O}}$ is a collection of interpretations and ρ is a partial function such that $(\mathbf{A}_\mathcal{O}, \prec, (\mathcal{I}_L)_{L\in\mathbf{A}_\mathcal{O}}, \rho)$ is an A-interpretation except that some interpretation domains $\Delta^{\mathcal{I}_L}$ may be empty; the length of tuples in the range of ρ may be at most $||\mathcal{O}||$;
- f_{in} and f_{out} are functions that associate every $L \in \mathbf{A}_\mathcal{O}$ with a set of pairs (q, h) where q is a CQ from an abstraction statement in $\mathcal{O}$ or a subquery thereof, and h is a partial function from $\mathsf{var}(q)$ to $\Delta^{\mathcal{I}_L}$; we call these pairs the *forbidden incoming queries* in the case of f_{in} and the *forbidden outgoing queries* in the case of f_{out}.

We may write $\mathcal{I}_L^M$ to denote $\mathcal{I}_L$, for any $L \in \mathbf{A}_\mathcal{O}$, and likewise for ρ^M, f_{in}^M, and f_{out}^M.

Every mosaic has to satisfy several additional conditions. Before we can state them, we introduce some notation. For $V \subseteq \mathsf{var}(q)$, we use $q|_V$ to denote the restriction of q to the variables in V and write $\overline{V}$ as shorthand for $\overline{V} = \mathsf{var}(q) \setminus V$. A *maximally connected component (MCC)* of q is a CQ $q|_V$ that is connected and such that V is maximal with this property. A CQ $p = E \uplus p_0$ is a *component of q w.r.t. $V \subseteq \mathsf{var}(q)$* if p_0 is an MCC of $q|_{\overline{V}}$ and E is the set of all atoms from q that contain one variable from V and one variable from $\overline{V}$.

Example 3. *The following CQ has two components w.r.t. $V = \{x, y\}$, which are displayed in dashed and dotted lines:*



For example, the dotted component is defined by $p = E \uplus p_0$ with $E = r(u, x) \wedge r(y, v)$ and $p_0 = r(u, v) \wedge r(v, u)$.

With these notions at hand, let us explain the intuition of the f_{in} and f_{out} components of mosaics. Our decomposition of A-interpretations into sets of mosaics is such that every ensemble falls within a single mosaic. This means that we must avoid homomorphisms from the CQs in concept abstractions that hit multiple mosaics: such homomorphisms would hit elements from multiple ensembles while also turning the set of all elements that are hit into an ensemble; they thus generate overlapping ensembles which is forbidden. Almost the same holds for role abstractions where however the CQ takes the form $q(\bar{x}, \bar{y})$ with each of $\bar{x}$ and $\bar{y}$ describing an ensemble, and we must only avoid homomorphisms that hit multiple mosaics from the variables in $\bar{x}$, or from the variables in $\bar{y}$.

Query avoidance is implemented by the f_{in} and f_{out} components. In brief and for CQs $q(\bar{x})$ from concept abstractions, we consider any non-empty subset $V \subsetneq \mathsf{var}(q)$ of variables and homomorphism h from $q|_V$ to the current mosaic. We then have to avoid any homomorphism g from $q \setminus q|_V$ that is compatible with h and hits at least one mosaic other than the current one. The choice of V decomposes q into remaining components, which are exactly the components of q w.r.t. V defined above. We choose one such component p and put (p, h') into f_{out}, h' the restriction of h to the variables in p, to 'send' the information to other mosaics that this query is forbidden. The f_{in} component, in contrast, contains forbidden queries that we 'receive' from other mosaics.

We now formulate the additional conditions on mosaics. We require that $M = ((\mathcal{I}_L)_{L\in\mathbf{A}_\mathcal{O}}, \rho, f_{\mathsf{in}}, f_{\mathsf{out}})$ satisfies the following conditions, for all $L \in \mathbf{A}_\mathcal{O}$:

1. the A-interpretation $(\mathbf{A}_\mathcal{O}, \prec, (\mathcal{I}_L)_{L\in\mathbf{A}_\mathcal{O}}, \rho)$ satisfies all inclusions, refinements, and abstractions in $\mathcal{O}$ with the possible exception of CIs the form $A \sqsubseteq_L \exists r.B$;
2. for all concept abstractions $L'{:}A$ $\underline{\text{abstracts}}$ $L{:}q(\bar{x})$ in $\mathcal{O}$, all non-empty $V \subsetneq \mathsf{var}(q)$, and all homomorphisms h from $q|_V$ to $\mathcal{I}_L$: there is a component p of q w.r.t. V such that $(p, h|_{V\cap\mathsf{var}(p)}) \in f_{\mathsf{out}}(L)$
3. for all role abstractions $L'{:}R$ $\underline{\text{abstracts}}$ $L{:}q(\bar{x}, \bar{y})$ in $\mathcal{O}$, all non-empty $V \subsetneq \mathsf{var}(q)$ with $V \neq \bar{x}$ and $V \neq \bar{y}$,[1] and all homomorphisms h from $q|_V$ to $\mathcal{I}_L$: there is a component p of q w.r.t. V such that $(p, h|_{V\cap\mathsf{var}(p)}) \in f_{\mathsf{out}}(L)$;
4. for all $(q, h) \in f_{\mathsf{in}}(L)$, all $V \subseteq \mathsf{var}(q)$, and all homomorphisms g from $q|_V$ to $\mathcal{I}_L$ that extend h, there is a component p of q w.r.t. V such that $(p, g|_{V\cap\mathsf{var}(p)}) \in f_{\mathsf{out}}(L)$.

We next need a mechanism to interconnect mosaics. This is driven by concept names A and elements $d \in A^{\mathcal{I}_L}$ such that $A \sqsubseteq_L \exists R.B \in \mathcal{O}$ and d lacks a witness inside the mosaic. In principle, we would simply like to find a mosaic M' that has some element e on level L such that $e \in B^{\mathcal{I}^{M'}}$ and an R-edge can be put between d and e. The situation is complicated, however, by the presence of role refinements and role abstractions, which might enforce additional edges that link the two mosaics. We must also be careful to synchronize the $f_{\mathsf{in}}, f_{\mathsf{out}}$ components of the two involved mosaics across the connecting edges.

Consider mosaics $M = ((\mathcal{I}_L)_{L\in\mathbf{A}_\mathcal{O}}, \rho, f_{\mathsf{in}}, f_{\mathsf{out}})$ and $M' = ((\mathcal{I}'_L)_{L\in\mathbf{A}_\mathcal{O}}, \rho', f'_{\mathsf{in}}, f'_{\mathsf{out}})$. An *$M, M'$-edge* is an expression $R(d, d')$ such that R is a role, $d \in \Delta^{\mathcal{I}_L}$, and $d' \in \Delta^{\mathcal{I}'_L}$ for some $L \in \mathbf{A}_\mathcal{O}$. A set E of M, M'-edges is an *edge candidate* if the following conditions are satisfied:

1. $R(d, e) \in E$ and $L(d) = L$ implies $S(d, e) \in E$, for all $R \sqsubseteq_L S \in \mathcal{O}$;
2. if $\exists R.A \sqsubseteq_L B \in \mathcal{O}$, $R(d, d') \in E$, and $d' \in A^{\mathcal{I}'_L}$, then $d \in B^{\mathcal{I}_L}$;
3. for all $L \in \mathbf{A}_\mathcal{O}$, all $(q, h) \in f_{\mathsf{out}}(L)$, where $q = E_q \uplus q|_{\overline{V}}$ for $V = \mathsf{dom}(h)$, and all functions g from $\overline{V} \cap \mathsf{var}(E_q)$ to $\Delta^{\mathcal{I}'_L}$ such that $R(h(x), g(y)) \in E$ for all $R(x, y) \in E_q$, we have $(q|_{\overline{V}}, g) \in f'_{\mathsf{in}}(L)$;
4. for all $R(d, d') \in E$ and all $L{:}q(\bar{x}, \bar{y})$ $\underline{\text{refines}}$ $L'{:}q_R(x, y) \in \mathcal{O}$ such that $q = q|_{\bar{x}} \uplus E_q \uplus q|_{\bar{y}}$, $q_R = C_x(x) \wedge R(x, y) \wedge C_y(y)$, $d \in C_x^{\mathcal{I}_{L'}}$, and $d' \in C_y^{\mathcal{I}_{L'}}$:
   (a) $\rho_L(d)$ and $\rho'_L(d')$ are defined;
   (b) $h : \bar{x} \mapsto \rho_L(d)$ is a homomorphism from $q|_{\bar{x}}$ to $\mathcal{I}_L$;
   (c) $h' : \bar{y} \mapsto \rho'_L(d')$ is a homomorphism from $q|_{\bar{y}}$ to $\mathcal{I}'_L$;
   (d) $\{R(h(x), h'(y)) \mid R(x, y) \in E_q\} \subseteq E$;

[1]Here we view $\bar{x}$ and $\bar{y}$ as sets.

5. for all role abstractions $L'{:}R$ abstracts $L{:}q(\bar{x}, \bar{y}) \in \mathcal{O}$, where $q = q|_{\bar{x}} \uplus E_q \uplus q|_{\bar{y}}$, all homomorphisms h from $q|_{\bar{x}}$ to $\mathcal{I}_L$, and all homomorphisms g from $q|_{\bar{y}}$ to $\mathcal{I}'_L$ such that $\{S(h(x), g(y)) \mid S(x, y) \in E_q\} \subseteq E$, there are $d \in \Delta^{\mathcal{I}_{L'}}$ and $d' \in \Delta^{\mathcal{I}'_{L'}}$ with $\rho_L(d) = h(\bar{x})$, $\rho'_L(d') = g(\bar{y})$, and $R(d, d') \in E$;
6. Converses of Conditions 2-5 above that go from M' to M instead of from M to M'; details are in the appendix.

Let $\mathcal{M}$ be a set of mosaics. A mosaic M is *good in* $\mathcal{M}$ if for all $A \sqsubseteq_L \exists R.B \in \mathcal{O}$ and $d \in (A \sqcap \neg\exists R.B)^{\mathcal{I}^M_L}$:

$(*)$ there is a mosaic $M' \in \mathcal{M}$, a $d' \in B^{\mathcal{I}^{M'}_L}$, and an edge candidate E such that $R(d, d') \in E$.

The actual algorithm is now identical to that from the previous section. We first compute the set $\mathcal{M}_0$ of all mosaics and then repeatedly and exhaustively eliminate mosaics that are not good. Let $\mathcal{M}^*$ denote the set of mosaics at which this process stabilizes.

Lemma 2. *C_0 is L_0-satisfiable w.r.t. $\mathcal{O}$ iff $\mathcal{M}^*$ contains (i) a mosaic M with $C_0^{\mathcal{I}^M_{L_0}} \neq \emptyset$ and (ii) a mosaic M with $\Delta^{\mathcal{I}^M_L} \neq \emptyset$, for every L in $\mathbf{A}_{\mathcal{O}}$.*

The algorithm thus returns 'satisfiable' if Conditions (i) and (ii) from Lemma 2 are satisfied and 'unsatisfiable' otherwise. It can be verified that the algorithm runs in double exponential time.

5 Lower Bounds

We have seen that the fragment $\mathcal{ALCHI}^{\mathsf{abs}}[\text{cr}]$ of $\mathcal{ALCHI}^{\mathsf{abs}}$ which focusses on concept refinement is only ExpTime-complete. Here we show that all other fragments that contain only a single form of abstraction/refinement are 2ExpTime-hard, and consequently 2ExpTime-complete. This of course also provides a matching lower bound for Theorem 2—actually three rather different lower bounds, each one exploiting a different effect. All of our lower bounds apply already when $\mathcal{ALCHI}$ is replaced with $\mathcal{ALC}$ as the underlying DL.

5.1 Role Abstraction: $\mathcal{ALC}^{\mathsf{abs}}[\text{ra}]$

The 2ExpTime-hardness of satisfiability in $\mathcal{ALCHI}^{\mathsf{abs}}$ is not entirely surprising given that we have built conjunctive queries into the logic and CQ evaluation on $\mathcal{ALCI}$ knowledge bases is known to be 2ExpTime-hard (Lutz 2008). In fact, this is already the case for the following *simple* version of the latter problem: given an $\mathcal{ALCI}$ ontology $\mathcal{O}$, a concept name A_0, and a Boolean CQ q, decide whether $\mathcal{I} \models q$ for all models $\mathcal{I}$ of $\mathcal{O}$ with $A_0^{\mathcal{I}} \neq \emptyset$. We write $\mathcal{O}, A_0 \models q$ if this is the case.

It is easy to reduce the (complement of the) simple CQ evaluation problem to satisfiability in $\mathcal{ALCI}^{\mathsf{abs}}[\text{ca}]$. Fix two abstraction levels $L \prec L'$, let $\widehat{q}$ be the CQ obtained from q by dequantifying all variables, thus making all variables answer variables, and let $\mathcal{O}'$ be the set of all concept inclusions $C \sqsubseteq_L D$ with $C \sqsubseteq D \in \mathcal{O}$ and the concept abstraction

$$L'{:}\bot \text{ abstracts } L{:}\widehat{q}.$$

It is straightforward to show that A_0 is L-satisfiable w.r.t. $\mathcal{O}'$ iff $\mathcal{O}, A_0 \not\models q$.

Our aim, however, is to prove 2ExpTime-hardness without using inverse roles. The above reduction does not help for this purpose since CQ evaluation on $\mathcal{ALC}$ knowledge bases is in ExpTime (Lutz 2008). Also, we wish to use role abstractions in place of the concept abstraction.

As above, we reduce from the complement of simple CQ evaluation on $\mathcal{ALCI}$ ontologies. Given an input $\mathcal{O}, A_0, q$, we construct an $\mathcal{ALC}^{\mathsf{abs}}[\text{ra}]$ ontology $\mathcal{O}'$ such that A_0 is L_1-satisfiable w.r.t. $\mathcal{O}'$ iff $\mathcal{O}, A_0 \not\models q$. The idea is to use multiple abstraction levels and role abstractions to simulate inverse roles. Essentially, we replace every inverse role r^- in $\mathcal{O}$ with a fresh role name $\widehat{r}$, obtaining an $\mathcal{ALC}$-ontology $\mathcal{O}_{\mathcal{ALC}}$, use three abstraction levels $L_1 \prec L_2 \prec L_3$, and craft $\mathcal{O}'$ so that in its models, the interpretation on level L_1 is a model of $\mathcal{O}_{\mathcal{ALC}}$, the interpretation on level L_2 is a copy of the one on level L_1, but additionally has an r-edge for every $\widehat{r}^-$-edge, and L_3 is an additional level used to check that there is no homomorphism from q, as in the initial reduction above. Details are in the appendix.

Theorem 3. *Satisfiability in $\mathcal{ALC}^{\mathsf{abs}}[\text{ra}]$ is 2ExpTime-hard.*

5.2 Concept Abstraction: $\mathcal{ALC}^{\mathsf{abs}}[\text{ca}]$

When only concept abstraction is available, the simulation of inverse roles by abstraction statements presented in the previous section does not work. We are nevertheless able to craft a reduction from the simple CQ evaluation problem on $\mathcal{ALCI}$ ontologies, though in a slightly different version. What is actually proved in (Lutz 2008) is that simple CQ evaluation is 2ExpTime-hard already in the DL $\mathcal{ALC}^{\mathsf{sym}}$, which is $\mathcal{ALC}$ with a single role name s that must be interpreted as a reflexive and symmetric relation.

We simulate $\mathcal{ALC}^{\mathsf{sym}}$-concepts $\exists s.C$ by $\mathcal{ALCI}$-concepts $\exists r^-.\exists r.C$, and likewise for $\forall s.C$. The reduction then exploits the following effect. Consider an $\mathcal{ALC}^{\mathsf{abs}}[\text{ca}]$-ontology $\mathcal{O}$ that contains the CI $\top \sqsubseteq_L \forall r.\exists \widehat{r}.\top$ and the concept abstractions

$$L'{:}\top \text{ abstracts } L{:}r(x, y) \text{ and } L'{:}\top \text{ abstracts } L{:}\widehat{r}(y, x).$$

Then in any model $\mathcal{I}$ of $\mathcal{O}$, $(d, e) \in r^{\mathcal{I}_L}$ implies $(e, d) \in \widehat{r}^{\mathcal{I}_L}$ and thus we can use $\widehat{r}$ as the inverse of r. This is because the first concept abstraction forces (d, e) to be an ensemble on level L, e must have a $\widehat{r}$-successor f, and the second concept abstraction forces (f, e) to be an ensemble. But since ensembles cannot overlap, this is only possible when $d = f$.

We have to be careful, though, to not produce undesired overlapping ensembles which would result in unsatisfiability. As stated, in fact, the ontology $\mathcal{O}$ does not admit models in which $\mathcal{I}_L$ contains an r-path of length two. This is why we resort to $\mathcal{ALC}^{\mathsf{sym}}$. Very briefly, the r^--part of the role composition $r^-; r$ falls inside an ensemble and is implemented based on the trick outlined above while the r-part connects two different ensembles to avoid overlapping. The details are somewhat subtle and presented in the appendix.

Theorem 4. *Satisfiability in $\mathcal{ALC}^{\mathsf{abs}}[\text{ca}]$ is 2ExpTime-hard.*

5.3 Role Refinement: $\mathcal{ALC}^{\mathsf{abs}}[\text{rr}]$

While concept and role abstractions enable reductions from CQ evaluation, this does not seem to be the case for concept and role refinements. Indeed, we have seen in Section 4.1 that concept refinements do not induce 2EXPTIME-hardness. Somewhat surprisingly, role refinements behave differently and are a source of 2EXPTIME-hardness, though for rather different reasons than abstraction statements.

It is well-known that there is an exponentially space-bounded alternating Turing machine (ATM) that decides a 2EXPTIME-complete problem and on any input w makes at most $2^{|w|}$ steps (Chandra, Kozen, and Stockmeyer 1981). We define ATMs in detail in the appendix and only note here that our ATMs have a one-side infinite tape and a dedicated accepting state q_a and rejecting state q_r, no successor configuration if its state is q_a or q_r, and exactly two successor configurations otherwise.

Let $M = (Q, \Sigma, \Gamma, q_0, \Delta)$ be a concrete such ATM with $Q = Q_\exists \uplus Q_\forall \uplus \{q_a, q_r\}$. We may assume w.l.o.g that M never attempts to move left when the head is positioned on the left-most tape cell. Let $w = \sigma_1 \cdots \sigma_n \in \Sigma^*$ be an input for M. We want to construct an $\mathcal{ALC}^{\mathsf{abs}}[\text{rr}]$-ontology $\mathcal{O}$ and choose a concept name S and abstraction level L_1 such that S is L_1-satisfiable w.r.t. $\mathcal{O}$ iff $w \in L(M)$. Apart from S, which indicates the starting configuration, we use the following concept names:

- A_σ, for each $\sigma \in \Gamma$, to represent tape content;
- A_q, for each $q \in Q$, to represent state and head position;
- $B_{q,\sigma,M}$ for $q \in Q$, $\sigma \in \Gamma$, $M \in \{L, R\}$, serving to choose a transition;
- $H_\leftarrow, H_\rightarrow$ indicating whether a tape cell is to the right or left of the head.

plus some auxiliary concept names whose purpose shall be obvious. We use the role name t for next tape cell and c_1, c_2 for successor configurations.

The ontology $\mathcal{O}$ uses the abstraction levels $\mathbf{A} = \{L_1, \dots, L_n\}$ with $L_{i+1} \prec L_i$ for $1 \leq i < n$. While we are interested in L_1-satisfiability of S, the computation of M is simulated on level L_n. We start with generating an infinite computation tree on level L_1:

$$S \sqsubseteq_{L_1} \exists c_1.N \sqcap \exists c_2.N \quad N \sqsubseteq_{L_1} \exists c_1.N \sqcap \exists c_2.N.$$

In the generated tree, each configuration is represented by a single object. On levels $L_2, \dots, L_n$, we generate similar trees where, however, configurations are represented by t-paths. The length of these paths doubles with every level and each node on a path is connected via c_1 to the corresponding node in the path that represents the first successor configuration, and likewise for c_2 and the second successor configuration. This is illustrated in Figure 2 where for simplicity we only show a first successor configuration and three abstraction levels. We use the following role refinements:

$$L_{i+1}\colon q(\bar{x}, \bar{y}) \ \underline{\text{refines}} \ L_i\colon t(x, y)$$
$$L_{i+1}\colon q_j(\bar{x}, \bar{y}) \ \underline{\text{refines}} \ L_i\colon c_i(x, y)$$

for $0 \leq i < n$ and $j \in \{1, 2\}$, and where $\bar{x} = x_1 x_2$,

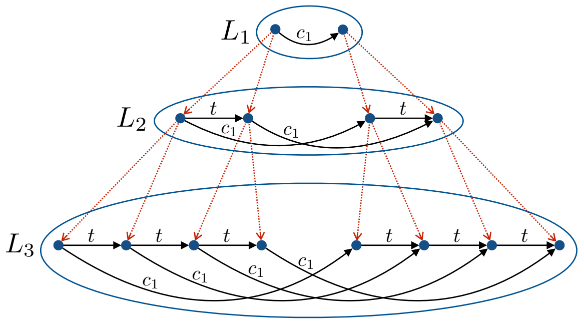


Figure 2: Dotted lines indicate refinement.

$\bar{y} = y_1 y_2$ and

$$q(\bar{x}, \bar{y}) = t(x_1, x_2) \wedge t(x_2, y_1) \wedge t(y_1, y_2)$$
$$q_j(\bar{x}, \bar{y}) = t(x_1, x_2) \wedge t(y_1, y_2) \wedge c_j(x_1, y_1) \wedge c_j(x_2, y_2).$$

To make more precise what we want to achieve, let the m-*computation tree*, for $m > 0$, be the interpretation $\mathcal{I}_m$ with

$$\Delta^{\mathcal{I}_m} = \{c_0, c_1\}^* \cdot \{1, \dots, m\}$$
$$t^{\mathcal{I}_m} = \{(wi, wj) \mid w \in \{c_0, c_1\}^*, 1 \leq i < m, j = i + 1\}$$
$$c_\ell^{\mathcal{I}_m} = \{(wj, wc_i j) \mid w \in \{c_0, c_1\}^*, 1 \leq j \leq m, i \in \{0, 1\}\}$$

for $\ell \in \{1, 2\}$. It can be shown that for any model $\mathcal{I}$ of the $\mathcal{ALC}^{\mathsf{abs}}[\text{rr}]$-ontology $\mathcal{O}$ constructed so far and for all $i \in \{1, \dots, n\}$, we must find a (homomorphic image of a) 2^i-computation tree in the interpretation $\mathcal{I}_{L_i}$. This crucially relies on the fact that ensembles cannot overlap. In Figure 2, for example, the role refinements for t and for c_1 both apply on level L_2, and for attaining the structure displayed on level L_3 it is crucial that in these applications each object on level L_2 refines into the same ensemble on level L_3.

On level L_n, we thus find a 2^n-computation tree which we use to represent the computation of M on input w. To start, the concept name S is copied down from the root of the 1-computation tree on level L_1 to that of the 2^n-computation tree on level L_n. To achieve this, we add a copy of the above role refinements for c_1, but now using the CQ

$$q_1(\bar{x}, \bar{y}) = S(x_1) \wedge c_1(x_1, y_1) \wedge c_1(x_2, y_2).$$

We next describe the initial configuration:

$$S \sqsubseteq_{L_n} A_{q_0} \sqcap A_{\sigma_1} \sqcap \forall t.A_{\sigma_2} \sqcap \cdots \sqcap \forall t^{n-1}.(A_{\sigma_n} \sqcap B_\rightarrow)$$
$$B_\rightarrow \sqsubseteq_{L_n} \forall t.(A_\square \sqcap B_\rightarrow)$$

For existential states, we consider one of the two possible successor configurations:

$$A_q \sqcap A_\sigma \sqsubseteq_{L_n} (\forall c_1.B_{q',\sigma',M'}) \sqcup (\forall c_2.B_{\bar{q},\bar{\sigma},\bar{M}})$$

for all $q \in Q_\exists$ and $\sigma \in \Gamma$ such that $\Delta(q, \sigma) = \{(q', \sigma', M'), (\bar{q}, \bar{\sigma}, \bar{M})\}$. For universal states, we use both successors:

$$A_q \sqcap A_\sigma \sqsubseteq_{L_n} (\forall c_1.B_{q',\sigma',M'}) \sqcap (\forall c_2.B_{\bar{q},\bar{\sigma},\bar{M}})$$

for all $q \in Q_\forall$ and $\sigma \in \Gamma$ such that $\Delta(q, \sigma) = \{(q', \sigma', M'), (\bar{q}, \bar{\sigma}, \bar{M})\}$. We next implement the transitions:

$$B_{q,\sigma,M} \sqsubseteq_{L_n} A_\sigma \quad \exists t.B_{q,\sigma,L} \sqsubseteq_{L_n} A_q$$
$$B_{q,\sigma,R} \sqsubseteq_{L_n} \forall t.A_q$$

for all $q \in Q$, $\sigma \in \Gamma$, and $M \in \{L, R\}$. We mark cells that are not under the head:

$$A_q \sqsubseteq_{L_n} \forall t.H_\leftarrow \qquad \exists t.A_q \sqsubseteq_{L_n} H_\rightarrow$$
$$H_\leftarrow \sqsubseteq_{L_n} \forall t.H_\leftarrow \qquad \exists t.H_\rightarrow \sqsubseteq_{L_n} H_\rightarrow$$

for all $q \in Q$. Such cells do not change:

$$(H_\leftarrow \sqcup H_\rightarrow) \sqcap A_\sigma \sqsubseteq_{L_n} \forall c_i.A_\sigma$$

for all $\sigma \in \Gamma$ and $i \in \{1,2\}$. State, content of tape, and head position must be unique:

$$A_q \sqcap A_{q'} \sqsubseteq_{L_n} \bot \quad A_\sigma \sqcap A_{\sigma'} \sqsubseteq_{L_n} \bot$$
$$(H_\leftarrow \sqcup H_\rightarrow) \sqcap A_q \sqsubseteq_{L_n} \bot$$

for all $q, q' \in Q$ and $\sigma, \sigma' \in \Gamma$ with $q \neq q'$ and $\sigma \neq \sigma'$. Finally, all followed computation paths must be accepting:

$$A_{q_r} \sqsubseteq_{L_n} \bot.$$

This finishes the construction of $\mathcal{O}$.

Lemma 3. *S is L_1-satisfiable w.r.t. $\mathcal{O}$ iff $w \in L(M)$.*

We have thus obtained the announced result.

Theorem 5. *Satisfiability in $\mathcal{ALC}^{\mathsf{abs}}[\mathrm{rr}]$ is 2ExpTime-hard.*

6 Undecidability

One might be tempted to think that the decidability of $\mathcal{ALCHI}^{\mathsf{abs}}$ is clear given that only a finite number of abstraction levels can be mentioned in an ontology. However, achieving decidability of DLs with abstraction and refinement requires some careful design choices. In this section, we consider three seemingly harmless extensions of $\mathcal{ALCHI}^{\mathsf{abs}}$ and show that each of them results in undecidability. This is in fact already the case for $\mathcal{EL}^{\mathsf{abs}}$ where the underlying DL $\mathcal{ALCHI}$ is replaced with $\mathcal{EL}$.

6.1 Basic Observations

We make some basic observations regarding the DL $\mathcal{EL}^{\mathsf{abs}}$ and its fragments. In classical $\mathcal{EL}$, concept satisfiability is not an interesting problem because every concept is satisfiable w.r.t. every ontology. This is not the case in $\mathcal{EL}^{\mathsf{abs}}$ where we can express concept inclusions of the form $C \sqsubseteq_L \bot$ with C an $\mathcal{EL}$-concept, possibly at the expense of introducing additional abstraction levels. More precisely, let L' be the unique abstraction level with $L \prec L'$ if it exists and a fresh abstraction level otherwise. Then $C \sqsubseteq_L \bot$ can be simulated by the following CI and concept abstraction:

$$C \sqsubseteq_L \exists r_C.\exists r_C.\top$$
$$L'{:}\top \;\underline{\mathsf{abstracts}}\; L{:}\, r_C(x,y)$$

where r_C is a fresh role name. Note that this again relies on the fact that ensembles cannot overlap, and thus an r_C-path of length two in level L results in unsatisfiability. The same can be achieved by using a role abstraction in place of the concept abstraction.

To prove our undecidability results, it will be convenient to have available concept inclusions of the form $C \sqsubseteq_L \forall r.D$ with C, D $\mathcal{EL}$-concepts. Let L' be a fresh abstraction level. Then $C \sqsubseteq_L \forall r.D$ can be simulated by the following role refinement and concept abstraction:

$$L'{:}\, r(x,y) \wedge A(y) \;\underline{\mathsf{refines}}\; L{:}\, C(x) \wedge r(x,y)$$
$$L'{:}\, D \;\underline{\mathsf{abstracts}}\; L'{:}\, A(x)$$

where A is a fresh concept name. It is easy to see that the same can be achieved with a role abstraction in place of the concept abstraction.

In the following, we thus use inclusions of the forms $C \sqsubseteq_L \bot$ and $C \sqsubseteq \forall r.D$ in the context of $\mathcal{EL}^{\mathsf{abs}}$ and properly keep track of the required types of abstraction and refinement statements.

6.2 Repetition-Free Tuples

In the semantics of $\mathcal{ALCHI}^{\mathsf{abs}}$ as defined in Section 3, ensembles are tuples in which elements may occur multiple times. It would arguably be more natural to define ensembles to be repetition-free tuples. We refer to this version of the semantics as the *repetition-free* semantics.

If only concept and role refinement are admitted, then there is no difference between satisfiability under the original semantics and under the repetition-free semantics. In fact, any model of $\mathcal{O}$ under the original semantics can be converted into a model of $\mathcal{O}$ under repetition-free semantics by duplicating elements. This gives the following.

Proposition 1. *For every $\mathcal{ALCI}$-concept C, abstraction level L, and $\mathcal{ALCHI}^{\mathsf{abs}}[\mathrm{cr}, \mathrm{rr}]$-ontology $\mathcal{O}$: C is L-satisfiable w.r.t. $\mathcal{O}$ iff C is L-satisfiable w.r.t. $\mathcal{O}$ under the repetition-free semantics.*

The situation changes once we admit abstraction.

Theorem 6. *Under the repetition-free semantics, satisfiability is undecidable in $\mathcal{ALC}^{\mathsf{abs}}[\mathrm{ca}]$, $\mathcal{ALC}^{\mathsf{abs}}[\mathrm{ra}]$, $\mathcal{EL}^{\mathsf{abs}}[\mathrm{rr}, \mathrm{ca}]$, and $\mathcal{EL}^{\mathsf{abs}}[\mathrm{rr}, \mathrm{ra}]$.*

In the following, we prove undecidability for satisfiability in $\mathcal{ALC}^{\mathsf{abs}}[\mathrm{ca}]$. The result for $\mathcal{ALC}^{\mathsf{abs}}[\mathrm{ra}]$ is a minor variation and the results for $\mathcal{EL}^{\mathsf{abs}}$ are obtained by applying the observations from Section 6.1.

We reduce the complement of the halting problem for deterministic Turing machines (DTMs) on the empty tape. Assume that we are given a DTM $M = (Q, \Sigma, \Gamma, q_0, \delta)$. As in Section 5.3, we assume that M has a one-side infinite tape and never attempts to move left when the head is on the left end of the tape. We also assume that there is a dedicated halting state $q_h \in Q$.

We want to construct an $\mathcal{ALC}^{\mathsf{abs}}[\mathrm{ca}]$-ontology $\mathcal{O}$ and choose a concept name S and abstraction level L such that S is L-satisfiable w.r.t. $\mathcal{O}$ iff M does not halt on the empty tape. We use essentially the same concept and role names as in Section 5.3, except that only a single role name c is used for transitions to the (unique) next configuration. Computations are represented in the form of a grid as shown on the left-hand side of Figure 3, where the concept names X_i must be disregarded as they belong to a different reduction (and so do the queries). We use two abstraction levels L and L' with $L \prec L'$. The computation of M is represented on level L.

We first generate an infinite binary tree in which every node has one t-successor and one c-successor:

$$\top \sqsubseteq_L \exists t.\top \sqcap \exists c.\top$$

To create the desired grid structure, it remains to enforce that grid cells close, that is, the t-successor of a c-successor of

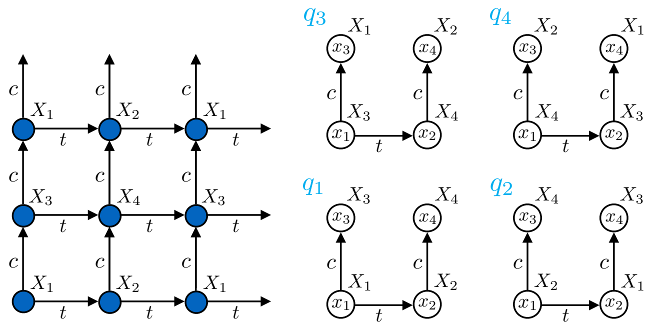


Figure 3: Grid structure and queries for the DAG Semantics.

any node coincides with the c-successor of the t-successor of that node. We add the concept abstraction

$$L'\colon \bot \text{ \underline{abstracts} } L\colon q(\bar{x}) \text{ where}$$
$$q(\bar{x}) = c(x_1,x_2) \wedge t(x_1,x_3) \wedge c(x_3,x_4) \wedge t(x_2,x_4').$$

The idea is that any non-closing grid cell admits a repetition-free answer to q on $\mathcal{I}_L$, thus resulting in unsatisfiability. If all grid cells close, there will still be answers, but all of them are repetitive. The above abstraction alone, however, does not suffice to implement this idea. It still admits, for instance, a non-closing grid cell in which the two left elements have been identified. We thus need to rule out such unintended identifications and add the concept abstraction $L'\colon \bot$ abstracts $L\colon q$ for the following six CQs q:

$$t(x_1,x_2) \wedge c(x_1,x_2) \qquad t(x_1,x_2) \wedge c(x_2,x_1)$$
$$t(x_1,x_2) \wedge c(x_1,x_3) \wedge t(x_3,x_2) \qquad c(x_1,x_1)$$
$$t(x_1,x_2) \wedge c(x_1,x_3) \wedge c(x_2,x_3) \qquad t(x_1,x_1)$$

The rest of the reduction is now very similar to that given in Section 5.3, details are in the appendix.

6.3 DAG Semantics

Our semantics requires abstraction levels to be organized in a tree. While this is very natural, admitting a DAG structure might also be useful. This choice, which we refer to as the *DAG semantics*, leads to undecidability.

Theorem 7. *Under the DAG semantics, satisfiability is undecidable in $\mathcal{ALC}^{\mathsf{abs}}$[ca,cr] and $\mathcal{EL}^{\mathsf{abs}}$[ca, cr, rr].*

The result is again proved by a reduction from (the complement of) the halting problems for DTMs. In fact, the reduction differs from that in Section 6.2 only in how the grid is constructed and thus we focus on that part. We present the reduction for $\mathcal{ALC}^{\mathsf{abs}}$[ca, cr].

Assume that we are given a DTM $M = (Q, \Sigma, \Gamma, q_0, \delta)$. We want to construct an ontology $\mathcal{O}$ and choose a concept name S and abstraction level L such that S is L-satisfiable w.r.t. $\mathcal{O}$ iff M does not halt on the empty tape. We use abstraction levels L, L_1, L_2, L_3, L_4 with $L \prec L_i$ for all $i \in \{1,\dots,4\}$. The computation of M is simulated on level L. We start with generating an infinite t-path with outgoing infinite c-paths from every node:

$$S \sqsubseteq_L \exists t.A_t \qquad A_t \sqsubseteq_L \exists t.A_t$$
$$A_t \sqsubseteq_L \exists c.A_c \qquad A_c \sqsubseteq_L \exists c.A_c.$$

In principle, we would like to add the missing t-links using the following concept abstraction and refinement:

$$L_1\colon U_1 \text{ \underline{abstracts} } L\colon q$$
$$L\colon q \wedge t(x_3,x_4) \text{ \underline{refines} } L_1\colon U_1 \text{ where}$$
$$q = c(x_1,x_3) \wedge t(x_1,x_2) \wedge c(x_2,x_4).$$

This would then even show undecidability under the original semantics, but it does not work because it creates overlapping ensembles and thus simply results in unsatisfiability. We thus use the four abstraction levels $L_1,\dots,L_4$ in place of only L_1. This results in different kinds of ensembles on level L, one for each level L_i, and an L_i-ensemble can overlap with an L_j-ensemble if $i \neq j$. We label the grid with concept names $X_1,\dots,X_4$ as shown in Figure 3, using CIs

$$X_1 \sqsubseteq_L \forall c.X_3 \sqcap \forall t.X_2 \qquad X_2 \sqsubseteq_L \forall c.X_4 \sqcap \forall t.X_1$$
$$X_3 \sqsubseteq_L \forall c.X_1 \qquad X_4 \sqsubseteq_L \forall c.X_2 \qquad S \sqsubseteq X_1.$$

We define four variations $q_1,\dots,q_4$ of the above CQ q, as shown on the right-hand side of Figure 3, and use the following concept abstraction and refinement, for $i \in \{1,\dots,4\}$:

$$L_i\colon U_i \text{ \underline{abstracts} } L\colon q_i$$
$$L\colon q_i \wedge t(x_3,x_4) \text{ \underline{refines} } L_i\colon U_i.$$

It can be verified that this eliminates overlapping ensembles and indeed generates a grid.

6.4 Quantified Variables

The final variation that we consider is syntactic rather than semantic: we admit quantified variables in CQs in abstraction and refinement statements.

Theorem 8. *In the extension with quantified variables, satisfiability is undecidable in $\mathcal{ALC}^{\mathsf{abs}}$[ca,cr] and $\mathcal{EL}^{\mathsf{abs}}$[ca, cr, rr].*

We use a DTM reduction that follows the same lines as the previous reduction and only explain how to generate a grid. We again start with an infinite t-path with outgoing infinite c-paths from every node. In the previous reduction, the main issue when adding the missing t-links was that a naive implementation creates overlapping ensembles. It is here that quantified variables help since they allow us to speak about elements without forcing them to be part of an ensemble. We use the following concept abstraction and refinement:

$$L_1\colon U_1 \text{ \underline{abstracts} } L\colon q$$
$$L\colon q \wedge t(x_3,x_4) \text{ \underline{refines} } L_1\colon U_1 \text{ where}$$
$$q = \exists x_1 \exists x_2\, c(x_1,x_3) \wedge t(x_1,x_2) \wedge c(x_2,x_4).$$

7 Conclusion

We have introduced DLs that support multiple levels of abstraction and include operators based on CQs for relating these levels. As future work, it would be interesting to analyse the complexity of abstraction DLs based on Horn DLs such as $\mathcal{EL}$, $\mathcal{ELI}$ and Horn-$\mathcal{ALCI}$. It would also be interesting to design an ABox formalism suitable for abstraction DLs, and to use such DLs for ontology-mediated querying. Finally, our work leaves open some decidability questions such as for $\mathcal{ALC}^{\mathsf{abs}}$[cr] and $\mathcal{EL}^{\mathsf{abs}}$[cr, ca] under the DAG semantics and with quantified variables.

Acknowledgments

The research reported in this paper has been supported by the German Research Foundation DFG, as part of Collaborative Research Center (Sonderforschungsbereich) 1320 Project-ID 329551904 “EASE - Everyday Activity Science and Engineering”, University of Bremen (http://www.ease-crc.org/). The research was conducted in subproject “P02 - Ontologies with Abstraction”.

This work is partly supported by BMBF (Federal Ministry of Education and Research) in DAAD project 57616814 (SECAI, School of Embedded Composite AI) as part of the program Konrad Zuse Schools of Excellence in Artificial Intelligence.

References

Artale, A.; Franconi, E.; Guarino, N.; and Pazzi, L. 1996. Part-whole relations in object-centered systems: An overview. *Data & Knowledge Engineering* 20(3):347–383.

Baader, F.; Horrocks, I.; Lutz, C.; and Sattler, U. 2017. *An Introduction to Description Logic*. Cambridge University Press.

Bittner, T., and Smith, B. 2003. A theory of granular partitions. In *Foundations of Geographic Information Science*. Taylor & Francis. 117–149.

Calegari, S., and Ciucci, D. 2010. Granular computing applied to ontologies. *Int. J. Approx. Reason.* 51(4):391–409.

Chandra, A. K.; Kozen, D. C.; and Stockmeyer, L. J. 1981. Alternation. *Journal of the ACM (JACM)* 28(1):114–133.

Cima, G.; Console, M.; Lenzerini, M.; and Poggi, A. 2022. Monotone abstractions in ontology-based data management. In *Proc. of AAAI*, 5556–5563. AAAI Press.

Dams, D., and Grumberg, O. 2018. Abstraction and abstraction refinement. In *Handbook of Model Checking*. Springer. 385–419.

Fonseca, F. T.; Egenhofer, M. J.; Jr., C. A. D.; and Câmara, G. 2002. Semantic granularity in ontology-driven geographic information systems. *Ann. Math. Artif. Intell.* 36(1-2):121–151.

Gangemi, A., and Mika, P. 2003. Understanding the semantic web through descriptions and situations. In *Proc. of OTM*, volume 2888 of *LNCS*, 689–706. Springer.

Glimm, B.; Kazakov, Y.; and Tran, T. 2017. Ontology materialization by abstraction refinement in horn SHOIF. In *Proc. of AAAI*, 1114–1120. AAAI Press.

Hbeich, E.; Roxin, A.; and Bus, N. 2021. Connecting granular and topological relations through description logics (short paper). In *Proc. of the GeoLD Workshop*, volume 2977 of *CEUR Workshop Proceedings*, 97–104. CEUR-WS.org.

Hobbs, J. R. 1985. Granularity. In *Proc. of IJCAI*, 432–435. Morgan Kaufmann.

Keet, C. M. 2008. *A Formal Theory of Granularity*. Ph.D. Dissertation, KRDB Research Centre, Faculty of Computer Science, Free University of Bozen-Bolzano.

Klarman, S., and Gutiérrez-Basulto, V. 2016. Description logics of context. *J. Log. Comput.* 26(3):817–854.

Klinov, P.; Taylor, J. M.; and Mazlack, L. J. 2008. Interval rough mereology and description logic: An approach to formal treatment of imprecision in the semantic web ontologies. *Web Intell. Agent Syst.* 6(2):157–174.

Lisi, F. A., and Mencar, C. 2018. A granular computing method for OWL ontologies. *Fundam. Informaticae* 159(1-2):147–174.

Lutz, C., and Schulze, L. 2023. Description logics with abstraction and refinement. *CoRR* abs/2306.03717.

Lutz, C. 2008. The complexity of conjunctive query answering in expressive description logics. In *Proc. of IJCAR*, volume 5195 of *LNCS*, 179–193. Springer.

Manière, Q. 2022. *Counting queries in ontology-based data access*. Ph.D. Dissertation, Université de Bordeaux.

Schulz, S.; Boeker, M.; and Stenzhorn, H. 2008. How granularity issues concern biomedical ontology integration. In *Proc. of MIE*, volume 136 of *Studies in Health Technology and Informatics*, 863–868. IOS Press.

Segoufin, L., and ten Cate, B. 2013. Unary negation. *Log. Methods Comput. Sci.* 9(3).

Stearns, M. Q.; Price, C.; Spackman, K. A.; and Wang, A. Y. 2001. SNOMED clinical terms: overview of the development process and project status. In *Proc. of AMIA*.

Vogt, L. 2019. Levels and building blocks - toward a domain granularity framework for the life sciences. *J. Biomed. Semant.* 10(1):4:1–4:29.