Description Logics with Aggregates and Concrete Domains

F. Baader, U. Sattler,*
RWTH Aachen, 52074 Aachen, Germany
baader@informatik.rwth-aachen.de, uli@cantor.informatik.rwth-aachen.de

Abstract

We show that extending description logics by simple aggregation functions as available in database systems may lead to undecidability of inference problems such as satisfiability and subsumption.

1 Motivation

Aggregation is a very useful mechanism available in many expressive representation formalisms such as database schema and query languages. Most systems provide for a fixed set of aggregation functions like sum, min, max, average, count, which can be used over a given built-in domain, like the integers or the reals. In this paper, the generic Description Logic $\mathcal{ALC}(\mathcal{D})$, as introduced in [Baader & Hanschke1991], is extended by aggregation. $\mathcal{ALC}(\mathcal{D})$ is an extension of the well-known description language $\mathcal{ALC}$ (see [Schmidt-Schauß & Smolka1991; Hollunder *et al.*1990; Donini *et al.*1991]) by a so-called *concrete domain*. In the basic language $\mathcal{ALC}$, concepts can be built using propositional operators, (i.e., *and* ($\sqcap$), *or* ($\sqcup$), and *not* ($\neg$)), and value restrictions on those individuals associated to an individual via a certain role. These are *existential* restrictions like in ($\exists$has_child.Girl) as well as *universal* restrictions like ($\forall$ has_child.Human). Additionally, in $\mathcal{ALC}(\mathcal{D})$, abstract individuals, which are described using $\mathcal{ALC}$, can be related to values in a *concrete domain* (e.g., the integers or strings) via *features*, i.e., functional roles. This allows us to describe managers that spend more money than they earn by Manager $\sqcap$ (less(income, expenses)). In our extension of $\mathcal{ALC}(\mathcal{D})$, aggregation is viewed as a means to define new features. In Figure 1, a person, Josie, is given who spends, in some months, more money than she earns, and in others less. If we want to know the difference between income and expenses for a whole year, we have to consider the sum over all months. Then we can state that or ask whether Josie is an instance of

$$\mathsf{Human} \sqcap (\exists \mathtt{year}.\mathsf{less}(\,\mathsf{sum}(\mathtt{month} \circ \mathtt{income}),\ \mathsf{sum}(\mathtt{month} \circ \mathtt{expenses}))),$$

where the complex feature sum(month $\circ$ income) relates an individual to the sum over all values reachable over month followed by income. This new, complex feature is built using the aggregation function sum, the role name month, and the feature income.

In this paper, we present a generic extension of $\mathcal{ALC}(\mathcal{D})$ by aggregation that is based on this idea of introducing new "aggregated features." Unfortunately, it turns out that, given a concrete domain together with aggregation functions satisfying some very weak conditions, this extension has an undecidable satisfiability problem. Moreover, this result can even be tightened: extending $\mathcal{FL}_0$, a very weak Description Logic allowing for conjunction and universal value restrictions only, by a weak form of aggregation already leads to undecidability of satisfiability and subsumption.

For database research, these results are, for example, of interest in the context of intensional reasoning in the presence of aggregation, as considered in [Ross *et al.*1998; Gupta *et al.*1995; Mumick & Shmueli1995; Levy & Mumick1996; Srivastava *et al.*1996]. They are not comparable with the undecidability results presented in [Mumick & Shmueli1995] since our prerequisites are weaker and no recursion mechanisms are used. Neither are they contained in the undecidability results in [Ross *et al.*1998]: the results presented there concern constraints involving multiplication and addition as well as rather complex aggregation functions like average or count—in contrast to the results presented here.

2 The basic Description Logic $\mathcal{ALC}(\mathcal{D})$

Before we can introduce $\mathcal{ALC}(\mathcal{D})$, as defined in [Baader & Hanschke1991], we must specify the notion of a concrete domain.

Definition 1
A *concrete domain* $\mathcal{D} = (\mathsf{dom}(\mathcal{D}), \mathsf{pred}(\mathcal{D}))$ consists of

*supported by the DFG under Grant No. Sp 230\ 6–6 and by the EU Working Group DWQ

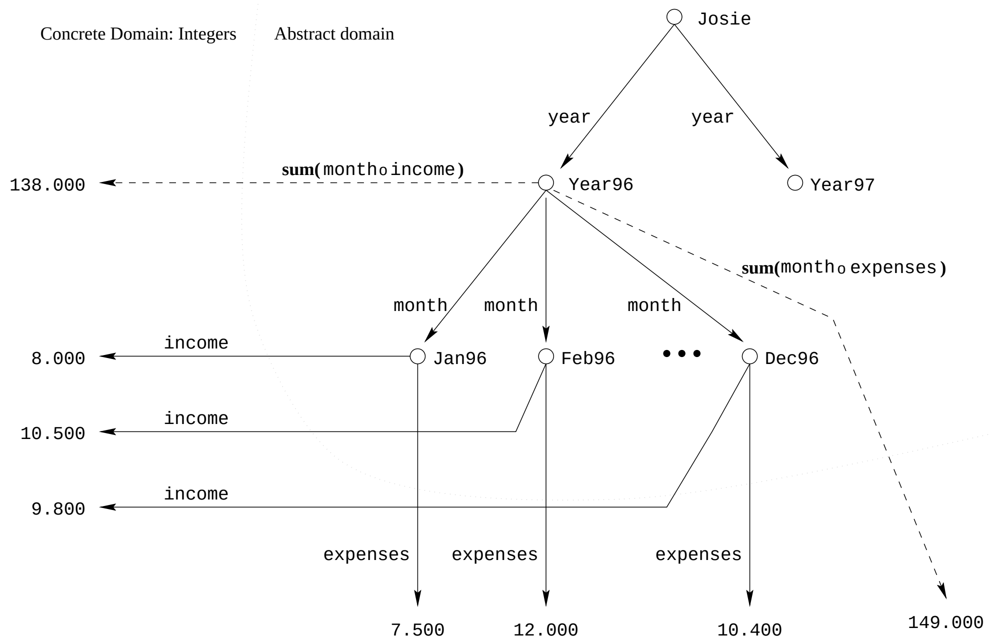

Figure 1: Example for aggregation

a set $\mathsf{dom}(\mathcal{D})$ (the domain) and a set of predicate symbols $\mathsf{pred}(\mathcal{D})$. Each predicate symbol $P \in \mathsf{pred}(\mathcal{D})$ is associated with an arity n and an n-ary relation $P^{\mathcal{D}} \subseteq \mathsf{dom}(\mathcal{D})^n$.

In [Baader & Hanschke1991], concrete domains are restricted to so-called *admissible* concrete domains in order to keep the inference problems of this extension decidable. We recall that, roughly spoken, a concrete domain $\mathcal{D}$ is called *admissible* iff (a) $\mathsf{pred}(\mathcal{D})$ is closed under negation and contains a unary predicate name $\top$ for $\mathsf{dom}(\mathcal{D})$, and (b) satisfiability in $\mathcal{D}$ of finite conjunctions over $\mathsf{pred}(\mathcal{D})$ is decidable. The syntax of $\mathcal{ALC}(\mathcal{D})$-concepts is now defined as follows:

Definition 2 Let N_C, N_R, and N_F be disjoint sets of *concept*, *role*, and *feature names*. The set of $\mathcal{ALC}(\mathcal{D})$-*concepts* is the smallest set such that

1. every concept name is a concept and
2. if C, D are concepts, R is a role or a feature name, $P \in \mathsf{pred}(\mathcal{D})$ is a predicate name, and $u_1, \ldots, u_n$ are feature chains,[1] then $(C \sqcap D)$, $(C \sqcup D)$, $(\neg C)$, $(\forall R.C)$, $(\exists R.C)$, and $P(u_1, \ldots, u_n)$ are concepts.

In order to fix the exact meaning of these concepts, their semantics is defined in the usual model-theoretic way.

Definition 3 An *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consists of a set $\Delta^{\mathcal{I}}$ disjoint from $\mathsf{dom}(\mathcal{D})$, called the *domain* of $\mathcal{I}$, and a function $\cdot^{\mathcal{I}}$ that maps every concept to a subset of $\Delta^{\mathcal{I}}$, every role to a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, and every feature name $f \in N_F$ to a partial function $f^{\mathcal{I}} : \Delta^{\mathcal{I}} \rightarrow \Delta^{\mathcal{I}} \cup \mathsf{dom}(\mathcal{D})$. Furthermore, $\mathcal{I}$ has to satisfy the following properties

$$\begin{aligned}
(C \sqcap D)^{\mathcal{I}} &= C^{\mathcal{I}} \cap D^{\mathcal{I}}, \\
(C \sqcup D)^{\mathcal{I}} &= C^{\mathcal{I}} \cup D^{\mathcal{I}}, \\
\neg C^{\mathcal{I}} &= \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}, \\
(\exists R.C)^{\mathcal{I}} &= \{d \in \Delta^{\mathcal{I}} \mid \text{Exists } e \in \Delta^{\mathcal{I}} \text{ with} \\
&\qquad (d,e) \in R^{\mathcal{I}} \text{ and } e \in C^{\mathcal{I}}\}, \\
(\forall R.C)^{\mathcal{I}} &= \{d \in \Delta^{\mathcal{I}} \mid \text{For all } e \in \Delta^{\mathcal{I}}, \\
&\qquad \text{if } (d,e) \in R^{\mathcal{I}}, \text{ then } e \in C^{\mathcal{I}}\}, \\
P(u_1, \ldots, u_n)^{\mathcal{I}} &= \{x \in \Delta^{\mathcal{I}} \mid (u_1^{\mathcal{I}}(x), \ldots, u_n^{\mathcal{I}}(x)) \in P^{\mathcal{D}}\},
\end{aligned}$$

where $(f_1 \circ \ldots \circ f_m)^{\mathcal{I}}(x) := f_1^{\mathcal{I}}(f_2^{\mathcal{I}}(\ldots(f_m^{\mathcal{I}}(x)\ldots)$. A concept C is called *satisfiable* iff there is some interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$. A concept D is said to *subsume* another concept C (written $C \sqsubseteq D$) iff all interpretations $\mathcal{I}$ satisfy $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. For an interpretation $\mathcal{I}$, an individual $x \in \Delta^{\mathcal{I}}$ is called an *instance* of a concept C iff $x \in C^{\mathcal{I}}$.

From the results presented in [Baader & Hanschke 1991], it follows immediately that subsumption and satisfiability are decidable for $\mathcal{ALC}(\mathcal{D})$ concepts—given that $\mathcal{D}$ is admissible. The authors present a tableau-based procedure that decides these and other problems.

3 Extension of $\mathcal{ALC}(\mathcal{D})$ by aggregation

In order to define aggregation appropriately, first, we will introduce the notion of *multisets*: in contrast to

[1] A feature chain $u = f_1 \circ \ldots \circ f_m$ is a sequence of features.

simple sets, in a multiset an individual can occur more than once; for example, the multiset $\{1\}$ is different from the multiset $\{1, 1\}$. Multisets are needed to ensure, e.g., that `Josie`'s income is calculated correctly in the case she earns the same amount of money in more than one month.

Definition 4 Let S be a set. A *multiset* M over S is a mapping $M : S \to \mathbb{N}$, where $M(s)$ denotes the number of occurrences of s in M. We write $s \in M$ as shorthand for $M(s) \geq 1$. A multiset M is said to be *finite* iff $\{s \mid M(s) \neq 0\}$ is a finite set.

As the aggregation functions depend strongly on the specific concrete domains, the notion of a *concrete domain* is extended accordingly. Furthermore, the notion of *concrete features* is introduced. These are features which can be built using aggregation over roles followed by features. Then $\mathcal{ALC}(\mathcal{D}+\Sigma)$-concepts are defined.

Definition 5 The notion of a concrete domain $\mathcal{D}$ as introduced in Definition 1 is extended by a set of aggregation functions $\mathsf{agg}(\mathcal{D})$, where each $\Sigma \in \mathsf{agg}(\mathcal{D})$ is a partial function from the set of multisets over $\mathsf{dom}(\mathcal{D})$ into $\mathsf{dom}(\mathcal{D})$.

The set of *concrete features* is inductively defined as follows:

- Each feature name $f \in N_F$ is a concrete feature,
- feature chains are concrete features,
- if $R \in N_R$ is a role, f is a concrete feature, and $\Sigma \in \mathsf{agg}(\mathcal{D})$ is an aggregation function, then $\Sigma(R\circ f)$ is a concrete feature.

Finally, $\mathcal{ALC}(\mathcal{D}+\Sigma)$-concepts are obtained from $\mathcal{ALC}(\mathcal{D})$-concepts by allowing, additionally, the use of concrete features f_i in predicate restrictions $P(f_1, \ldots, f_n)$ (recall that in $\mathcal{ALC}(\mathcal{D})$, only feature chains were allowed).

It remains to extend the semantics of $\mathcal{ALC}(\mathcal{D})$ to the new feature forming operator:

Definition 6 An $\mathcal{ALC}(\mathcal{D}+\Sigma)$-interpretation $\mathcal{I}$ is an $\mathcal{ALC}(\mathcal{D})$-interpretation $\mathcal{I}$ which additionally satisfies

$$(\Sigma(R\circ f))^{\mathcal{I}} = \{(x, y) \in \Delta^{\mathcal{I}} \times \mathsf{dom}(\mathcal{D}) \mid \Sigma^{\mathcal{D}}(M_x^{R\circ f}) = y\}$$

where, for $x \in \Delta^{\mathcal{I}}$, a concrete feature f, and a role R, $M_x^{R\circ f}$ denotes the multiset over $\mathsf{dom}(\mathcal{D})$ where the number of occurrences of $z \in \mathsf{dom}(\mathcal{D})$ is determined by the number of $R^{\mathcal{I}}$-successors y of x with $f^{\mathcal{I}}(y) = z$, i.e. for $z \in \mathsf{dom}(\mathcal{D})$ we have

$$M_x^{R\circ f}(z) := \#\{y \in \Delta^{\mathcal{I}} \mid (x, y) \in R^{\mathcal{I}} \text{ and } f^{\mathcal{I}}(y) = z\}.$$

We point out two consequences of this definition, which might not be obvious at first sight: (a) if $(R\circ f)^{\mathcal{I}}(x)$ contains individuals in $\Delta^{\mathcal{I}}$, then these individuals have no influence on $M_x^{R\circ f}$: it is defined in such a way that it takes only into account $(R\circ f)^{\mathcal{I}}$-successors of x in the concrete domain $\mathsf{dom}(\mathcal{D})$; (b) if $M_x^{R\circ f}$ is not finite, then the outcome depends on $\mathcal{D}$ and Σ: for example, the minimum of a (possibly infinite) subset of the positive integers is always defined, whereas the sum is undefined for infinite subsets.

Unfortunately, the following theorem shows that admissibility of a concrete domain does no longer guarantee decidability of the interesting inference problems:

Theorem 7 For a concrete domain $\mathcal{D}$ where

- $\mathsf{dom}(\mathcal{D})$ includes the non-negative integers $\mathbb{N}$,
- $\mathsf{pred}(\mathcal{D})$ contains a (unary) predicate $P_{=1}$ that tests for equality with 1, and the (binary) equality predicate $P_=$,
- $\mathsf{agg}(\mathcal{D})$ contains $\mathsf{min}, \mathsf{max}, \mathsf{sum}$,

satisfiability and subsumption of $\mathcal{ALC}(\mathcal{D}+\Sigma)$-concepts is undecidable.

Remarks: (a) The aggregation functions $\mathsf{min}, \mathsf{max}, \mathsf{sum}$ are supposed to be defined as usual, i.e., "$\geq$" is an extension of "$\geq$" on $\mathbb{N}$, and

$$\mathsf{sum}(M) = \begin{cases} \sum\limits_{y\in M} M(y)\cdot y & \text{if M is finite} \\ \text{undefined} & \text{otherwise} \end{cases}$$

$$\mathsf{min}(M) = \begin{cases} m & \text{if there exists m} \in \text{M such that n} \geq \text{m for all n} \in \text{M} \\ \text{undefined} & \text{if such an m does not exist} \end{cases}$$

$$\mathsf{max}(M) = \begin{cases} m & \text{if there exists m} \in \text{M such that n} \leq \text{m for all n} \in \text{M} \\ \text{undefined} & \text{if such an m does not exist} \end{cases}$$

(b) At first sight, this undecidability result may seem to be rather restricted. Note, however, that it just requires that $\mathsf{dom}(\mathcal{D})$ *contains* the non-negative integers. Furthermore, the aggregation functions $\mathsf{min}, \mathsf{max}, \mathsf{sum}$ are among those normally considered as built-in functions in databases (see, for example, [Gupta *et al.*1995; Mumick & Shmueli1995; Levy & Mumick1996; Srivastava *et al.*1996]). Finally, to test whether a certain value equals 1 or whether two values are equal is possible in all database systems with built-in predicates.

Proof of Theorem 7: The proof is by reduction of Hilbert's 10th problem [Davis1973] to the satisfiability of concepts, i.e., for polynomials $P, Q \in \mathbb{N}[x_1, \ldots, x_m]$, one can construct an $\mathcal{ALC}(\mathcal{D}+\Sigma)$-concept $C_{P,Q}$ that is satisfiable iff the polynomial equation

$$P(x_1, \ldots, x_m) = Q(x_1, \ldots, x_m) \tag{1}$$

has a solution in $\mathbb{N}^m$. When building the reduction concept $C_{P,Q}$, one encounters three major problems: (a) We only know that $\mathsf{dom}(\mathcal{D})$ *contains* $\mathbb{N}$, but the solution of Equation 1 has to be in $\mathbb{N}^m$, and $\mathcal{D}$ need not provide for a predicate that tests for being a non-negative integer. (b) The reduction asks for the simulation of calculations such as addition, multiplication, and exponentiation. (c) It has to be assured that (the representation of) each variable x_i is associated with the same non-negative integer wherever it occurs in a model of $C_{P,Q}$.

In the following, we sketch how these problems can be solved—details and the definition of $C_{P,Q}$ can be found in [Baader & Sattler1997]: (a) is solved by making use of the concept

$$E_g^R := (\forall R.(P_{=1}(f))) \sqcap P_{=}(\mathsf{sum}(R \circ f), g),$$

whose instances have as g-successor the number of their R-successors. Hence their g-successor is in $\mathbb{N}$ or undefined (if there are infinitely many R-successors). (b) Addition can be realized by the aggregation function sum, and multiplication (and hence exponentiation) can be reduced to addition. (c) This problem is solved by introducing features $\mathtt{x}_i$ for each variable x_i and by making strong use of the concept Inv. All R-successors of an instance a of Inv have the same $\mathtt{x}_i$-successor, which equals the $\mathtt{x}_i$-successor of a.

$$\begin{aligned}\mathsf{Inv} := \bigsqcap_{1 \le i \le m} (\forall R.\top(\mathtt{x}_i) \sqcap\ & P_{=}(\mathsf{min}(R \circ \mathtt{x}_i), \mathsf{max}(R \circ \mathtt{x}_i)) \sqcap \\ & P_{=}(\mathtt{x}_i, \mathsf{max}(R \circ \mathtt{x}_i))).\end{aligned}$$

This concept can be used to guarantee that all "relevant" individuals in a model of $C_{P,Q}$ have the same $\mathtt{x}_i$-successor for each variable x_i.

Then the idea of the reduction is to represent the (sub)term structure of the polynomial P (resp. Q) as a tree which is related to an instance of $C_{P,Q}$ via the feature P (resp. Q). Each leaf of these trees stands for one of the variables x_i, whose value is "spread" over the whole structure using the concept Inv described above.

We want to emphasize that $C_{P,Q}$ does not make any use of the possibility to apply aggregation functions to feature chains, i.e., wherever a subconcept of $C_{P,Q}$ contains $\Sigma(R \circ f)$ for some aggregation function Σ, f is a feature name (and not a complex feature chain or concrete feature).

A closer investigation of the concept $C_{P,Q}$ reveals that (a) negation does not occur, (b) no concept of the form $\exists R.C$ is used, and (c) the only place where disjunction $\sqcup$ occurs is in concepts E_n^R describing individuals having exactly n R-successors (which are used to represent the coefficients of the polynomials):

$$\begin{aligned} E_n^R \quad := \quad & \forall R.\left(\bigsqcup_{1 \le i \le n} P_{=1}(f_i)\right) \sqcap \\ & \forall R.\left(\bigsqcap_{1 \le i \le n} (P_{=1}(f_i) \Rightarrow (\bigsqcap_{j \ne i} \bot(f_j)))\right) \sqcap \\ & \bigsqcap_{1 \le i \le n} P_{=1}(\mathsf{sum}(R \circ f_i)).\end{aligned}$$

For an instance a of E_n^R, every R-successor has an f_i-successor for exactly one $i, 1 \le i \le n$, and this f_i-successor has value 1 (first two lines). The constraint on the concrete feature $\mathsf{sum}(R \circ f_i)$ (third line) makes sure that there is exactly one R-successor with an f_i-successor for each i, which implies that a has exactly n R-successors. In $\mathcal{ALC}(\mathcal{D}+\Sigma)$, with $\mathcal{D}$ as described in the preconditions of Theorem 7, it seems to be impossible to describe the fact that an individual has exactly n R-successors without using union. However, given a concrete domain $\mathcal{D}$ that provides, in addition to what was required in Theorem 7, for all non-negative integers n a unary predicate $P_{=n}$ that test for equality with n, then the following concept $E_n^{R'}$ can be used to describe those individuals having exactly n R-successors:

$$E_n^{R'} := \forall R.P_{=1}(f) \sqcap P_{=n}(\mathsf{sum}(R \circ f)).$$

Hence, the reduction concept $C_{P,Q}$ can be rewritten using only conjunction $\sqcap$ and universal value restriction $\forall R.C$. As introduced in [Baader1990], let $\mathcal{FL}_0$ denote the set of those concepts that are built using conjunction and universal value restriction only, and let $\mathcal{FL}_0(\mathcal{D}+\Sigma)$ denote the extension of this language by concrete domains and aggregation. Then the following corollary is an immediate consequence of the remarks made above.

Corollary 8 For a concrete domain $\mathcal{D}$ where

- $\mathsf{dom}(\mathcal{D})$ includes the non-negative integers $\mathbb{N}$,
- $\mathsf{pred}(\mathcal{D})$ contains, for all non-negative integers n, (unary) predicates $P_{=n}$ that test for equality with n, and the (binary) equality predicate $P_{=}$,
- $\mathsf{agg}(\mathcal{D})$ contains $\mathsf{min}, \mathsf{max}, \mathsf{sum}$,

satisfiability and subsumption of $\mathcal{FL}_0(\mathcal{D}+\Sigma)$-concepts is undecidable.

Undecidability of satisfiability is shown, as sketched above, by a reduction of Hilbert's 10th problem. From this, undecidability of subsumption follows because a concept C is satisfiable iff it is not subsumed by an unsatisfiable concept, and because the $\mathcal{FL}_0(\mathcal{D}+\Sigma)$-concept $C_\bot := \top(f) \sqcap \bot(f)$ is such an unsatisfiable concept.

4 Conclusion

Reasoning with constraints involving aggregation functions is a crucial task for many advanced information systems like decision support and on-line-analytical processing systems, data warehouses, and (statistical) databases [Ross *et al.*1998; Gupta *et al.*1995; Mumick & Shmueli 1995; De Giacomo & Naggar1996; Levy & Mumick1996; Srivastava *et al.*1996]. The more the amount of data grows that are processed by these systems, the more important become aggregation functions for summarizing, consolidating and analyzing these large amounts of data.

Hence, traditional techniques for query rewriting, query optimization, view maintenance, etc. must be extended such that they are able to cope with aggregation functions.

The two undecidability results presented in this paper indicate that this task will be difficult. The aggregation functions $\mathsf{min}, \mathsf{max}, \mathsf{sum}$ that suffice to obtain undecidability are the most "well-behaved" ones: aggregation functions like count or $\mathsf{average}$ are much more difficult to handle. For example, $\mathsf{min}, \mathsf{max}, \mathsf{sum}$ are monotonic, i.e., if $S \subseteq S'$, then

$$\begin{aligned} \mathsf{min}(S) &\geq \mathsf{min}(S'), \\ \mathsf{max}(S) &\leq \mathsf{max}(S'), \\ \mathsf{sum}(S) &\leq \mathsf{sum}(S'), \end{aligned}$$

whereas these relations cannot be established for count or $\mathsf{average}$. Furthermore, they are "compositional" in the sense that the aggregation $f \in \{\mathsf{min}, \mathsf{max}, \mathsf{sum}\}$ of two disjoint multisets S, S' can be computed using $f, f(S), f(S')$ only—which does not hold, for example, for $\mathsf{average}$. Hence, our undecidability result cannot be said to be caused by using a too powerful set of aggregation functions.

Arguing from another perspective, $\mathcal{ALC}(\mathcal{D}+\Sigma)$ is a rather expressive Description Logic and it might not be very surprising that adding aggregation to $\mathcal{ALC}(\mathcal{D})$ leads to undecidability. In contrast, $\mathcal{FL}_0$ is, to our knowledge, the weakest Description Logic ever considered. It is of such a low expressive power that subsumption between two $\mathcal{FL}_0$-concepts can be reduced to answering conjunctive queries: given two $\mathcal{FL}_0$-concepts C_1 and C_2, C_1 subsumes C_2 if and only if an individual x of an extensional database $\mathsf{edb}_{C_1(x)}$ constructed from C_1 is in the answer set of a conjunctive query q_{C_2} constructed from C_2. This reduction is, for several reasons, not possible for $\mathcal{FL}_0(\mathcal{D}+\Sigma)$-concepts. However, it leads to the speculation that (intensional) reasoning for conjunctive queries with (simple) aggregation functions and built-in predicates is of high computational complexity.

References

[Baader 1990] F. Baader. Terminological cycles in KL-ONE-based knowledge representation languages. Technical Report RR-90-01, Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI), Kaiserslautern, Germany, 1990. An abridged version appeared in *Proc. of the 8th Nat. Conf. on Artificial Intelligence AAAI-90*, pp. 621–626.

[Baader & Hanschke 1991] F. Baader and P. Hanschke. A schema for integrating concrete domains into concept languages. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence (IJCAI-91)*, pages 452–457, Sydney, 1991.

[Baader & Sattler 1997] F. Baader and U. Sattler. Description logics with aggregates and concrete domains. Technical report, LuFg Theoretical Computer Science, RWTH Aachen, 1997. In preparation.

[Davis 1973] M. Davis. Hilbert's tenth problem is unsolvable. *American Mathematical Monthly*, 80:233–269, 1973.

[De Giacomo & Naggar 1996] G. De Giacomo and P. Naggar. Conceptual data model with structured objects for statistical databases. In *Proceedings of the Eighth International Conference on Statistical Database Management Systems (SSDBM'96)*, pages 168–175. IEEE Computer Society Press, 1996.

[Donini *et al.* 1991] F. Donini, M. Lenzerini, D. Nardi, and W. Nutt. The complexity of concept languages. In *Proceedings of the Second International Conference on the Principles of Knowledge Representation and Reasoning (KR-91)*, Boston (USA), 1991.

[Gupta *et al.* 1995] A. Gupta, V. Harinarayan, and D. Quass. Aggregate-query processing in data warehousing environments. In *Proceedings of the 21. International Conference on Very Large Data Bases (VLDB-95)*, 1995.

[Hollunder *et al.* 1990] B. Hollunder, W. Nutt, and M. Schmidt-Schauss. Subsumption algorithms for concept description languages. In *ECAI-90*, Pitman Publishing, London, 1990.

[Levy & Mumick 1996] A. Y. Levy and I. S. Mumick. Reasoning with aggregation constraints. In *Proceedings of the International Conference on Extending Database Technology (EDBT-96)*, Avignon, France, 1996.

[Mumick & Shmueli 1995] I. S. Mumick and O. Shmueli. How expressive is stratified aggregation. *Annals of Mathematics and Artificial Intelligence*, 15(3-4), 1995.

[Ross *et al.* 1998] K. Ross, D. Srivastava, P. J. Stuckey, and S. Sudarshan. Foundations of aggregation constraints. *Theoretical Computer Science*, 1998. To appear.

[Schmidt-Schauß & Smolka 1991] M. Schmidt-Schauß and G. Smolka. Attributive concept descriptions with complements. *Artificial Intelligence*, 48(1):1–26, 1991.

[Srivastava *et al.* 1996] D. Srivastava, S. Dar, H. V. Jagadish, and A. Y. Levy. Answering queries with aggregation using views. In *Proceedings of the 22. International Conference on Very Large Data Bases (VLDB-96)*, Bombay, India, 1996.