

Efficient Decision Procedures for Query Containment and Related Problems

Tanara Lauschner¹, Marco A. Casanova¹, Vânia M.P. Vidal², José Antonio Fernandes de Macêdo²

¹Department of Informatics, Pontifical Catholic University of Rio de Janeiro
Rio de Janeiro, RJ – Brazil

²Department of Computing, Federal University of Ceará
Fortaleza, CE – Brazil

{tanara,casanova}@inf.puc-rio.br, {vvidal,jose.macedo}@lia.ufc.br

Abstract. *The subsumption problem in Description Logics (DL) refers to the question of deciding if a concept description always denotes a subset of the set denoted by another concept description. This paper explores reductions of query containment and other problems to the subsumption problem in DL. It first selects a DL dialect that is expressive enough to cover familiar classes of integrity constraints and query expressions. Then, it describes how to modify the tableau decision procedure for the subsumption problem to account for the classes of integrity constraints considered. Finally, it introduces a fast decision procedure for the subsumption problem for the dialect adopted.*

Resumo. *O problema de subsunção em Lógica de Descrição (LD) refere-se à questão de decidir se uma expressão definindo um conceito sempre denota um subconjunto do conjunto que outra expressão denota. Este trabalho explora reduções do problema de inclusão de consultas e problemas semelhantes ao problema de subsunção em LD. Inicialmente, o trabalho introduz um dialeto de LD que é suficientemente expressivo para cobrir certas classes de restrições de integridade e expressões de consulta. Em seguida, descreve como modificar o procedimento de decisão baseado em tableau para o problema de subsunção de forma a tratar as restrições de integridade consideradas. Por fim, apresenta um procedimento de decisão eficiente para o problema de subsunção no dialeto adotado.*

1. Introduction

Query containment refers to the problem of deciding if a query Q always returns a subset of the result of a second query Q' . An important special case of the query containment problem, that we call the *productive query problem*, covers testing if a query Q always returns an empty result. A third, related problem, that we call the *external schema subset constraint problem*, is to compute which (and how) the subset constraints of a base conceptual schema translate into subset constraints of an external schema, respecting the schema mappings. Any practical decision procedure for such problems has to take into account the integrity constraints of the underlying conceptual schema, and this is the challenge we address in this paper.

In another direction, the subsumption problem in Description Logics (DL) [11] refers to the question of deciding if a concept description always denotes a subset of the set denoted by another concept description. The subsumption problem is decidable for expressive dialects of DL, but typically belongs to hard complexity classes [1], especially in the presence of axioms [7]. For certain dialects of DL, there are polynomial decision procedures for the subsumption problem that explore the structure of the concept descriptions and that are, for this reason, called *structural subsumption procedures* [9][10]. However, such procedures do not take into account axioms. Furthermore, the reductions suggested to encode the axioms lead us back to dialects for which the subsumption problem is hard [7].

The contributions of this paper are threefold. First, we show how to express subset constraints and attribute domain and range definitions in a simple DL dialect, and how to reduce the query containment and related problems directly to the subsumption problem. Second, we show that our careful selection of the DL dialect leads to a simple extension of the familiar tableau decision procedure for the subsumption problem that accounts for the classes of integrity constraints introduced, without resorting to (inefficient) reductions. Third, and more important, we show that we can modify the structural subsumption procedure of [9][10], without impairing its complexity, to account for the classes of integrity constraints considered. Both the tableau and the structural subsumption procedures are of practical significance to the construction of query optimizers, to eliminate redundant subqueries, and of query mediators, to derive the constraints that will be exposed to the users of the mediator, as discussed in [5].

In addition to the references above, related work include the following selected references. Calvanese et al. [4] introduce a Description Logics framework, similar to that in Section 2, to address schema integration and query answering. Curino et al. [6] describe a software tool to support schema evolution that uses mapping invertibility. Fagin et al. [8] study mapping invertibility in the context of source-to-target tuple generating dependencies and formalize the notion of quasi-inverse.

This paper is organized as follows. Section 2 briefly reviews DL, introduces the classes of constraints considered, and defines the database problems we address. Section 3 shows how to modify the tableau decision procedure to account for the classes of integrity constraints introduced. Section 4 describes an efficient structural subsumption procedure that is tuned to the classes of constraints considered. Finally, Section 5 contains the conclusions. The Appendix exhibits proofs for the main results.

2. Basic Concepts

2.1. A Brief Review of Attributive Languages

We adopt the family of attributive languages \mathcal{ALCUE}^+ [1], defined as follows. A language \mathcal{L} in the family is characterized by an *alphabet* \mathcal{A} , consisting of a set of *atomic concepts*, a set of *atomic roles*, the *universal concept* and the *bottom concept*, denoted by \top and \perp , respectively, and a set of *constants*.

The set of *role descriptions* of \mathcal{L} is inductively defined as

- An atomic role is a role description

- If p and q are role descriptions, then the following expression is a role description

$$p \circ q \quad (\text{the composition of } p \text{ and } q)$$

The set of *concept descriptions* of \mathcal{L} is inductively defined as

- An atomic concept and the universal and bottom concepts are concept descriptions
- If a_1, \dots, a_n are constants, $\{a_1, \dots, a_n\}$ is a concept description, called a *set description*
- If e and f are concept descriptions and p is a role description, then the following expressions are concept descriptions

$$\begin{array}{ll} \neg e & (\text{negation}) \\ e \sqcap f & (\text{intersection}) \\ e \sqcup f & (\text{union}) \end{array} \quad \begin{array}{ll} \exists p & (\text{restricted existential quantification}) \\ \exists p.e & (\text{full existential quantification}) \\ \forall p.e & (\text{value restriction}) \end{array}$$

We also introduce a second family of languages, denoted \mathcal{FL} , that considers only set descriptions, intersection, restricted existential quantification and value restriction.

An *interpretation* s for the symbols in the alphabet \mathcal{A} consists of a nonempty set Δ^s , the *domain* of s , whose elements are called *individuals*, and an *interpretation function*, also denoted s , where:

- $s(\top) = \Delta^s$ and $s(\perp) = \emptyset$
- $s(A) \subseteq \Delta^s$, for each atomic concept A of \mathcal{L}
- $s(P) \subseteq \Delta^s \times \Delta^s$, for each atomic role P of \mathcal{L}
- $s(a) \in \Delta^s$, for each constant a of \mathcal{L} , such that distinct constants denote distinct individuals (the *uniqueness assumption*)

The function s is extended to role and concept descriptions of \mathcal{L} as follows:

- $s(p \circ q) = s(p) \circ s(q)$ (the composition of $s(p)$ with $s(q)$)
- $s(\{a_1, \dots, a_n\}) = \{s(a_1), \dots, s(a_n)\}$
- $s(\neg e) = \Delta^s - s(e)$ (the complement of $s(e)$ w.r.t. the domain Δ^s)
- $s(e \sqcap f) = s(e) \cap s(f)$ (the intersection of $s(e)$ and $s(f)$)
- $s(e \sqcup f) = s(e) \cup s(f)$ (the union of $s(e)$ and $s(f)$)
- $s(\exists p) = \{I \in \Delta^s \mid (\exists J \in \Delta^s)((I, J) \in s(p))\}$
(the set of individuals that $s(p)$ relates to some individual in the domain)
- $s(\exists p.e) = \{I \in \Delta^s \mid (\exists J \in \Delta^s)((I, J) \in s(p) \wedge J \in s(e))\}$
(the set of individuals that $s(p)$ relates to some individual in $s(e)$)
- $s(\forall p.e) = \{I \in \Delta^s \mid (\forall J \in \Delta^s)((I, J) \in s(p) \Rightarrow J \in s(e))\}$
(the set of individuals I such that, if $s(p)$ relates I to an individual J , then J is in $s(e)$)

A *formula* of \mathcal{L} is an expression of the form $u \sqsubseteq v$, called an *inclusion*, or of the form $u \equiv v$, called an *equivalence*, where u and v are both concept descriptions or they are both role descriptions of \mathcal{L} . A *definition* is an equivalence of the form $T \equiv u$, where T is an atomic concept and u is a concept description, or T is an atomic role and u is a role description.

An interpretation s for \mathcal{L} *satisfies* $u \sqsubseteq v$ iff $s(u) \subseteq s(v)$, and s *satisfies* $u \equiv v$ iff $s(u) = s(v)$.

We define the *subsumption problem* as ‘‘Given a set Σ of inclusions and an inclusion σ , determine if $\Sigma \models \sigma$ ’’. Special cases of this problem can be defined by restricting Σ or σ to be inclusions written in languages belonging to specific families.

In the rest of the paper, we will use the following notation:

- $s \models \sigma$ indicates that the interpretation s satisfies a formula σ
- $s \models \Sigma$ indicates that the interpretation s satisfies all formulas in a set of formulas Σ
- $\Sigma \models \sigma$ indicates that a set of formulas Σ *logically implies* a formula σ , that is, for any interpretation s , if $s \models \Sigma$, then $s \models \sigma$
- two formulas, φ and σ , are *logically equivalent* iff $\{\sigma\} \models \varphi$ and $\{\varphi\} \models \sigma$

2.2. Ultralite Schemas

We will work with ultralite schemas that, in OWL terminology [2], support classes and properties, and that admit domain and range constraints, and subset constraints, which are some of the most popular constraints in database design.

Formally, an *ultralite schema* is a pair $S=(\mathcal{L},\Sigma)$ such that

- \mathcal{L} is an \mathcal{ALCUE}^+ language with an alphabet \mathcal{A}
 - \mathcal{A} is called the *vocabulary* of S
 - the atomic concepts and atomic roles of \mathcal{A} are called the *classes* and *properties* of S , respectively
- Σ is a set of formulas, called the *constraints* of S , which must be of one the forms
 - *Domain Constraint*: $\exists P \sqsubseteq D$ (property P has *domain* D)
 - *Range Constraint*: $\top \sqsubseteq \forall P \cdot R$ (property P has *range* R)
 - *Subset Constraint*: $C \sqsubseteq D$ (class C is a *subclass* of class D)
- Σ must have exactly one domain and range constraint for each property in \mathcal{A}

From this point on, we will use the terms *class*, *property* and *vocabulary* interchangeably with the terms *atomic concept*, *atomic role* and *alphabet*, respectively.

Subset constraints capture class hierarchies and need no further discussion.

Let s be an interpretation of \mathcal{L} with domain Δ^s . Then, s satisfies a domain constraint $\exists P \sqsubseteq D$ iff, for any $a \in \Delta^s$, if $a \in s(\exists P)$ then $a \in s(D)$. But $a \in s(\exists P)$ iff there is $b \in \Delta^s$ such that $(a,b) \in s(P)$, which means that a is in the domain of $s(P)$. In other words, if a is in the domain of $s(P)$, then $a \in s(D)$. This justifies the formalization of domain constraints.

Likewise, s satisfies a range constraint $\top \sqsubseteq \forall P \cdot R$ iff, for any $a \in \Delta^s$, $a \in s(\forall P \cdot R)$, since $s(\top) = \Delta^s$. But $a \in s(\forall P \cdot R)$ iff, for any $b \in \Delta^s$, if $(a,b) \in s(P)$ then $b \in s(R)$. Thus, for any $a \in \Delta^s$, for any $b \in \Delta^s$, if $(a,b) \in s(P)$ then $b \in s(R)$. Or, equivalently, for any $a \in \Delta^s$, for any $b \in \Delta^s$, either $(a,b) \notin s(P)$, or $(a,b) \in s(P)$ and $b \in s(R)$. In other words, either $s(P)$ does not map a into any individual (that is, $s(P)$ is undefined for a), or $s(P)$ maps a into an individual b and b is in $s(R)$ (that is, if b is in the range of $s(P)$, then $b \in s(R)$). This justifies the formalization of range constraints.

Note that the formalization of ultralite schemas does not distinguish between object and datatype properties, as in OWL. The distinction will be visible in the examples, where the range of an object property will be a class defined in the schema, whereas the range of a datatype property will be an XML Schema type. The formal development does not capture this distinction since the notion of domain does not separate individuals that denote class elements from individuals that correspond to data values.

An *ultralite external schema* over $S=(\mathcal{L}, \Sigma)$ is a pair $\mathcal{E}=(\mathcal{K}, \mathcal{M})$ such that:

- for each atomic concept C of \mathcal{K} , \mathcal{M} contains a definition of the form $C \equiv \mu_C$, where μ_C is a concept description of \mathcal{L}
- for each atomic role P of \mathcal{K} , \mathcal{M} contains a definition of the form $P \equiv \mu_P$, where μ_P is a role description of \mathcal{L}

$S=(\mathcal{L}, \Sigma)$ is called the *base schema* of $\mathcal{E}=(\mathcal{K}, \mathcal{M})$. Note that the integrity constraints of \mathcal{E} are not explicitly defined, but induced by Σ (see Section 2.3).

Let s be an interpretation for \mathcal{L} . The interpretation ν for \mathcal{K} induced by s is such that $\nu(C)=s(\mu_C)$, if \mathcal{M} contains a definition of the form $C \equiv \mu_C$, and $\nu(P)=s(\mu_P)$, if \mathcal{M} contains a definition of the form $P \equiv \mu_P$.

Example 1: Figures 1(a) and 1(c) show an ultralite schema and an ultralite external schema, using an informal notation. We use the namespace prefixes “a:” and “e:” to refer to the vocabularies of the schema and of the external schema. Figures 1(b) and 1(e) formalize the constraints: the first column shows the domain constraints, the second column, the range constraints, and the third column, the subset constraints. Finally, Figure 1(d) shows the schema mappings. \square

a:Product a:title range string a:price range decimal a:currency range string a:Book is-a a:Product a:isbn range string a:author range string a:pub range a:Publ	a:Publ a:name range string a:address range string a:Music is-a a:Product a:Video is-a a:Product a:PC-HW is-a a:Product
--	---

Fig. 1(a). Informal definition of the ultralite schema.

$\exists a:\text{title} \sqsubseteq a:\text{Product}$ $\exists a:\text{price} \sqsubseteq a:\text{Product}$ $\exists a:\text{currency} \sqsubseteq a:\text{Product}$ $\exists a:\text{isbn} \sqsubseteq a:\text{Book}$ $\exists a:\text{author} \sqsubseteq a:\text{Book}$ $\exists a:\text{pub} \sqsubseteq a:\text{Book}$ $\exists a:\text{name} \sqsubseteq a:\text{Publ}$ $\exists a:\text{address} \sqsubseteq a:\text{Publ}$	$\top \sqsubseteq \forall a:\text{title}.\text{string}$ $\top \sqsubseteq \forall a:\text{price}.\text{decimal}$ $\top \sqsubseteq \forall a:\text{currency}.\text{string}$ $\top \sqsubseteq \forall a:\text{isbn}.\text{decimal}$ $\top \sqsubseteq \forall a:\text{author}.\text{string}$ $\top \sqsubseteq \forall a:\text{pub}.\text{a:Publ}$ $\top \sqsubseteq \forall a:\text{name}.\text{string}$ $\top \sqsubseteq \forall a:\text{address}.\text{string}$	$a:\text{Book} \sqsubseteq a:\text{Product}$ $a:\text{Music} \sqsubseteq a:\text{Product}$ $a:\text{Video} \sqsubseteq a:\text{Product}$ $a:\text{PC-HW} \sqsubseteq a:\text{Product}$
---	--	---

Fig. 1(b). Formal definition of the constraints of the ultralite schema.

e:Product e:title range string e:price range double	e:Book e:pub range string
---	------------------------------

Fig. 1(c). Informal definition of the external schema.

$e:\text{Product} \equiv a:\text{Product} \sqcap \exists a:\text{title}$ $e:\text{Book} \equiv a:\text{Book} \sqcap \exists a:\text{title} \sqcap \exists a:\text{pub}$	$e:\text{title} \equiv a:\text{title}$ $e:\text{price} \equiv a:\text{price}$ $e:\text{pub} \equiv a:\text{pub} \circ a:\text{name}$
--	--

Fig. 1(d). Formal definition of the external schema mappings.

$\exists e:\text{title} \sqsubseteq e:\text{Product}$ $\exists e:\text{price} \sqsubseteq e:\text{Product}$ $\exists e:\text{pub} \sqsubseteq e:\text{Book}$	$\top \sqsubseteq \forall e:\text{title}.\text{string}$ $\top \sqsubseteq \forall e:\text{price}.\text{decimal}$ $\top \sqsubseteq \forall e:\text{pub}.\text{string}$	$e:\text{Book} \sqsubseteq e:\text{Product}$
--	--	--

Fig. 1(e). Constraints of the external schema induced by those of the base schema.

2.3. Query Containment and Related Problems

The problems we consider in this paper are defined as follows:

The Query Containment Problem: “Given an ultralite schema $\mathcal{S}=(\mathcal{L},\Sigma)$ and two concept descriptions C and D of \mathcal{L} , determine if $\Sigma \models C \sqsubseteq D$ ”.

Intuitively, C and D capture queries over \mathcal{S} , or fragments of different queries, or even of the same query. If we can prove that $\Sigma \models C \sqsubseteq D$, then query C will always return a subset of query D , in the presence of the integrity constraints Σ .

The Productive Query Problem: “Given an ultralite schema $\mathcal{S}=(\mathcal{L},\Sigma)$ and a concept description C of \mathcal{L} , determine if $\Sigma \models C \sqsubseteq \perp$ ”.

This is a special case of the previous problem. If we can prove that $\Sigma \models C \sqsubseteq \perp$, then query C will always return an empty set (which is the standard interpretation of the bottom concept \perp), in the presence of the integrity constraints Σ .

The External Schema Subset Constraint Problem: “Given an ultralite schema $\mathcal{S}=(\mathcal{L},\Sigma)$, an ultralite external schema $\mathcal{E}=(\mathcal{K},\mathcal{M})$ over \mathcal{S} , and two classes, C and D , of \mathcal{K} , with definitions $C \equiv \mu_C$ and $D \equiv \mu_D$, determine if $\Sigma \models \mu_C \sqsubseteq \mu_D$ ”.

This last problem should be understood as follows. Assume that $\Sigma \models \mu_C \sqsubseteq \mu_D$. Since $C \equiv \mu_C$ and $D \equiv \mu_D$, we have that $\nu \models C \sqsubseteq D$, for any interpretation ν of \mathcal{E} induced by an interpretation of \mathcal{L} that satisfies Σ . Intuitively, users will always observe states of the external schema that satisfy $C \sqsubseteq D$, provided that we always assume that the interpretations of the base schema satisfy Σ .

Any of these problems admit variants by requiring that the concept descriptions pertain to specific DL dialects. This observation is of practical significance since the complexity of the problems depends on the choice of the dialect.

Example 2:

(a) (Query Containment) Consider the ultralite schema $\mathcal{S}=(\mathcal{L},\Sigma)$ defined in Example 1 and the following concept descriptions

- (1) $Q_1 \equiv a:\text{Product}$ (obtain the set of all products)
- (2) $Q_2 \equiv a:\text{Book} \sqcap \exists a:\text{author}$ (obtain the set of books that have known authors)
- (3) $Q_3 \equiv a:\text{Book} \sqcap \forall a:\text{author} . \{ \text{'Shakespeare'} \}$
(obtain the set of books that, if the author is known, he is 'Shakespeare')

Then, we can show that $\Sigma \models Q_2 \sqsubseteq Q_1$ and $\Sigma \models Q_3 \sqsubseteq Q_1$, since $a:\text{Book} \sqsubseteq a:\text{Product}$ is in Σ . However, we cannot show that $\Sigma \models Q_3 \sqsubseteq Q_2$ (unless an axiom in Σ guarantees that all books have known authors, which is not the case for the schema in Example 1).

(b) (*Productive Query*) Consider now the concept descriptions

- (4) $Q_4 \equiv a:\text{Book} \sqcap \neg a:\text{Product}$
- (5) $Q_5 \equiv a:\text{Book} \sqcap \forall a:\text{author} . \{ \text{'Shakespeare'} \} \sqcap \exists a:\text{author} . \{ \text{'Marlowe'} \}$

We can show that $\Sigma \models Q_4 \sqsubseteq \perp$, since $a:\text{Book} \sqsubseteq a:\text{Product}$ is in Σ . We can also show that $\Sigma \models Q_5 \sqsubseteq \perp$, since by definition different constants denote different individuals (which is the uniqueness assumption introduced in Section 2.1).

(c) (*External Schema Subset Constraint*) Recall from Figure 1(d) that $e:\text{Product}$ and $e:\text{Book}$, two concepts of the external schema, are defined as

$$(6) \quad e:\text{Product} \equiv a:\text{Product} \sqcap \exists a:\text{title}$$

$$(7) \quad e:\text{Book} \equiv a:\text{Book} \sqcap \exists a:\text{title} \sqcap \exists a:\text{pub}$$

Then, we have that $\Sigma \models e:\text{Book} \sqsubseteq e:\text{Product}$, which follows from (6), (7) and the fact that $a:\text{Book} \sqsubseteq a:\text{Product}$ is in Σ (see also Example 3). \square

3. Extending Tableau Procedures to Ultralite Schemas

In this section, we discuss how to extend the familiar tableau decision procedure for the subsumption problem [1] to account for the domain, range and subset constraints. The extension is in fact a decision procedure for the problems defined in Section 2.3, when the concept and role descriptions considered may use any of the constructs defined in Section 2.1.

We consider that, in addition to the usual tableau rules [1], we may resort to the new rules defined in Figure 2, which directly encode reasoning with the constraints of an ultralite schema. That is, we do not resort to (unpractical) reductions to encode the integrity constraints. Indeed, as observed in [7], by using union and transitive closure of roles, we can encode a set of inclusions $\Sigma = \{ C_1 \sqsubseteq D_1, \dots, C_n \sqsubseteq D_n \}$ as a set of concept descriptions $\Sigma^* = \{ \neg C_1 \sqcup D_1, \dots, \neg C_n \sqcup D_n \}$ in such a way that a concept description d is satisfiable w.r.t. Σ iff the following concept description is satisfiable:

$$d \sqcap \forall (R_1 \sqcup \dots \sqcup R_m)^* \cdot ((\neg C_1 \sqcup D_1) \sqcap \dots \sqcap (\neg C_n \sqcup D_n))$$

where R_1, \dots, R_m are the role names used in d or Σ . Note that this encoding uses a concept description in a DL dialect decidable in deterministic exponential time.

As for the usual tableau rules, the new rules in Figure 2 are expressed in terms of

- *concept assertions* of the form $C(a)$, where C is an atomic concept and a is a constant
- *role assertions* of the form $P(a,b)$, where P is an atomic role and a and b are constants
- *equality assertions* of the form $a=b$, where a and b are constants

As in [1], we assume that equality assertions are symmetric, in the sense that if a tableau node contains “ $a=b$ ”, it also implicitly contains “ $b=a$ ”. In view of the equality assertions, we also drop the uniqueness assumption for constants (see section 2.1).

The rules in Figure 2 should be understood as follows:

- The Subset Rule encodes subset constraints of the form $C \sqsubseteq D$ in Σ . It says that, if we assert that $C(a)$ holds, then $D(a)$ must also hold, in the presence of a subset constraint $C \sqsubseteq D$ in Σ .
- The Domain Rule encodes domain constraints of the form $\exists P \sqsubseteq D$ in Σ . It says that, if we assert that $P(a,b)$ holds, then a must be in the domain of P , that is, $D(a)$ must also hold.
- The Range Rule encodes range constraints of the form $\top \sqsubseteq \forall P.R$ in Σ . It says that, if we assert that $P(a,b)$ holds, then b must be in the range of P , that is, $R(b)$ must also hold.

- The Set of Constants Rule says that, if we assume that a is in the set of constants $\{c_1, \dots, c_n\}$, which is indicated by the assertion $\{c_1, \dots, c_n\}(a)$, then a must be one of the constants, which is expressed by the equality assertion $a=c_i$.
- The Role Composition Rule encodes role compositions of the form $p \circ q$. It says that, if we assert that $(p \circ q)(a, b)$ holds, then there must be an individual denoted by a new constant c such that $p(a, c)$ and $q(c, b)$ hold.

<i>Subset Rule</i>	if “ $C(a)$ ” is in node N and there is a subset constraint $C \sqsubseteq D$ in Σ then add “ $D(a)$ ” to N
<i>Domain Rule</i>	if “ $P(a, b)$ ” is in node N and $\exists P \sqsubseteq D$ is the domain constraint in Σ for property P then add “ $D(a)$ ” to N
<i>Range Rule</i>	if “ $P(a, b)$ ” is in node N and $\top \sqsubseteq \forall P.R$ is the range constraint in Σ for property P then add “ $R(b)$ ” to N
<i>Set of Constants Rule</i>	if “ $\{c_1, \dots, c_n\}(a)$ ” is in node N then add new children N_1, \dots, N_n of N , where N_i contains $a=c_i$
<i>Equality Rule</i>	if “ $a=b$ ” is in node N and an expression φ is in N , then add “ $\varphi [b/a]$ ” to N , where $\varphi [b/a]$ denotes φ with each occurrence of b replaced by a
<i>Role Composition Rule</i>	if “ $(p \circ q)(a, b)$ ” is in node N then add “ $p(a, c)$ ” and “ $q(c, b)$ ” to N , where c is a new constant

Fig. 2. Additional tableau rules.

We can prove that the extended tableau procedure is a decision procedure for the following special case of the subsumption problem: “Given a set Σ of inclusions of an ultralite schema and an inclusion σ in \mathcal{ALCUE}^+ , determine if $\Sigma \models \sigma$ ”.

Theorem 1: Let $\mathcal{S}=(\mathcal{A}, \Sigma)$ be an ultralite schema and σ be an inclusion in \mathcal{ALCUE}^+ . Then, any complete extended tableau for $\Sigma \cup \{\neg\sigma\}$ is closed iff $\Sigma \models \sigma$. \square

Theorem 2: The special case of the subsumption problem for ultralite schemas and inclusions in \mathcal{ALCUE}^+ is decidable. \square

(The proof of these results is outside the scope of this paper).

We conclude this section with an example that illustrates how to solve the instance of the External Schema Subset Constraint Problem described in Example 2(c).

Example 3: Let $\mathcal{S}=(\mathcal{L}, \Sigma)$ be the ultralite schema and $\mathcal{E}=(\mathcal{K}, \mathcal{M})$ be the external schema in Example 1. Recall that $e:\text{Product}$ and $e:\text{Book}$, two concepts of \mathcal{K} , are defined as

- (1) $e:\text{Product} \equiv a:\text{Product} \sqcap \exists a:\text{title}$
- (2) $e:\text{Book} \equiv a:\text{Book} \sqcap \exists a:\text{title} \sqcap \exists a:\text{pub}$

The tableau in Figure 3 establishes that

- (3) $\Sigma \models a:\text{Book} \sqcap \exists a:\text{title} \sqcap \exists a:\text{pub} \sqsubseteq a:\text{Product} \sqcap \exists a:\text{title}$

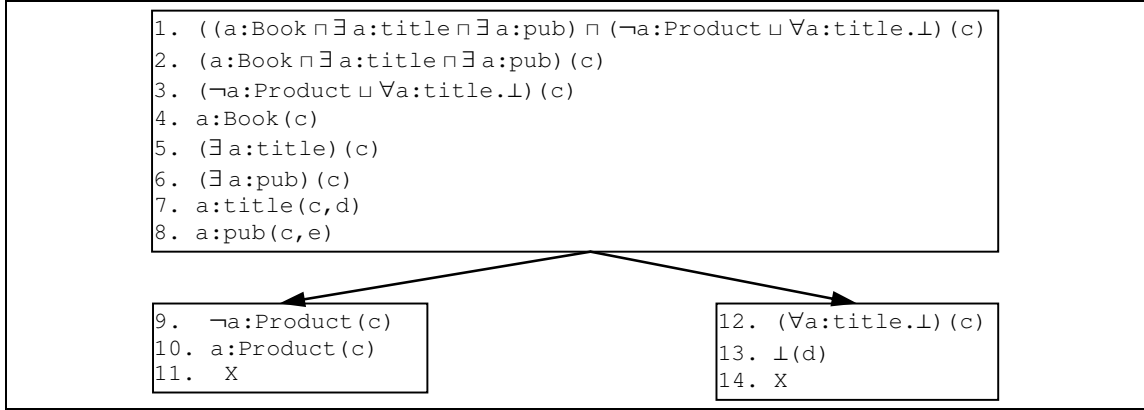


Fig. 3. Example of a tableau.

Briefly, the tableau is constructed as follows. Recall that the tableau procedure is a refutation procedure. Hence, Line 1 asserts that there is an individual, denote c , that is a counter-example to what we want to prove. That is, Line 1 asserts that c belongs to $a:\text{Book} \sqcap \exists a:\text{title} \sqcap \exists a:\text{pub}$ and to the negation of $a:\text{Product} \sqcap \exists a:\text{title}$. Negations are then pushed inside the expressions as much as possible. In particular, $\exists a:\text{title}$ is transformed into $\forall a:\text{title}.\perp$. Lines 2 to 6 result from decomposing the conjunctions by applying the \sqcap -rule (again, see [1] for the tableau rules). For example, Line 4 follows from Line 2 by the \sqcap -rule and asserts that $a:\text{Book}(c)$ holds. Lines 7 and 8 result from applying the \exists -rule to Lines 5 and 6. The tableau then branches into two nodes. Lines 9 and 12 follow from Line 3 by applying the \sqcup -rule. Line 10 follows from Line 4 by applying the Subset Rule, since there is a subset constraint $a:\text{Book} \sqsubseteq a:\text{Product}$ in Σ . Then, Line 11 indicates that Lines 9 and 10 lead to a contradiction. Line 13 follows from Lines 7 and 12 by the \forall -rule. Line 14 indicates that Line 13 leads to a contradiction, since $\perp(d)$ is always false (because \perp denotes the empty set). Hence, the tableau closes. By Theorem 1, we then establish (3). \square

4. A Fast Decision Procedure for Ultralite Schemas and \mathcal{FL} Inclusions

In this section, we first introduce the *extended structural subsumption* procedure (see Figure 4) that embeds reasoning with domain, range and subset constraints. The procedure is a modification of that described in [9][10]. We then prove that extended structural subsumption is a decision procedure for the following special case of the subsumption problem: “Given a set Σ of inclusions of an ultralite schema and an inclusion σ in \mathcal{FL} , determine if $\Sigma \models \sigma$ ”. Hence, it is a decision procedure for the problems defined in Section 2.3, when we consider Σ to be the constraints of an ultralite schema and when we restrict σ to be in \mathcal{FL} , that is, when σ contains only set descriptions, intersections, restricted existential quantifications and value restrictions.

Let $\mathcal{S}=(\mathcal{L},\Sigma)$ be an ultralite schema. The procedure depends on the *extended dependency graph* $\mathbf{G}^*(\Sigma) = (\mathbf{N}^*,\mathbf{E}^*)$ for Σ , defined as follows:

- \mathbf{N}^* is the set of atomic concepts that occur in Σ , extended with all concept descriptions of the form $\exists P$ for which $\exists P \sqsubseteq D$ is a domain constraint in Σ
- \mathbf{E}^* contains an arc (A,B) for each subset constraint $A \sqsubseteq B$ in Σ , and an arc $(\exists P,D)$ for each domain constraint $\exists P \sqsubseteq D$ in Σ

The procedure also depends on the notion of normal formal, defined as follows:

Definition 1: A concept description f of an \mathcal{FL} -language is in *normal form* iff

- (i) f is a set description, an atomic concept or a restricted existential quantification, or
- (ii) f is of the form $e_1 \sqcap \dots \sqcap e_m \sqcap \forall P_1 \cdot g_1 \sqcap \dots \sqcap \forall P_n \cdot g_n$ where
 - e_1 is a set description (the only admissible set description in f , if any), an atomic concept or a restricted existential quantification
 - for all $i \in [2, m]$, e_i is an atomic concept or a restricted existential quantification
 - for all $i, j \in [1, n]$, if $i \neq j$ then $P_i \neq P_j$
 - for all $i \in [1, n]$, g_i is in normal form. \square

Definition 2: An inclusion $e \sqsubseteq f$ of an \mathcal{FL} -language is in *normal form* iff e and f are in normal form.

We can always *normalize* an inclusion σ by: (1) replacing, on both sides of σ , all set descriptions, if any, by a single set description that represents their intersection; (2)

```

IMPLIES( $\Sigma, \sigma$ )
input: an ultralite schema with a set  $\Sigma$  of integrity constraints
         an  $\mathcal{FL}$ -inclusion  $\sigma$ 
output: True,   iff  $\Sigma \models \sigma$ 
         False,  otherwise
begin
  If any of the conjuncts on the left-hand side of  $\sigma$  is the bottom concept  $\perp$ , then return True
  Eliminate parenthesis from the conjunctions on the left- and the right-hand sides of  $\sigma$ 
  Normalize  $\sigma$ 
  Assume that  $\sigma$  is of the form  $e_1 \sqcap \dots \sqcap e_m \sqcap \forall P_1 \cdot g_1 \sqcap \dots \sqcap \forall P_n \cdot g_n \sqsubseteq f_1 \sqcap \dots \sqcap f_r \sqcap \forall Q_1 \cdot h_1 \sqcap \dots \sqcap \forall Q_s \cdot h_s$ 
  if  $f_1$  is a set description
    then begin if  $e_1$  is not a set description that does not define a subset of  $f_1$ 
      then return False /* False implies that  $\Sigma \not\models e_1 \sqsubseteq f_1$ 
       $b = 2$ 
    end
    else  $b = 1$ 
  for  $i = b$  to  $r$  do
    if there is no  $e_j$ , with  $j \in [b, m]$ , such that there is a path in  $G^*(\Sigma)$  from  $e_j$  to  $f_i$ 
      then return False /* False implies that, for each  $j \in [1, m]$ ,  $\Sigma \not\models e_j \sqsubseteq f_i$ 
  for  $i = 1$  to  $s$  do
    begin
      Let  $R_i$  be the range of  $Q_i$ , that is,  $\top \sqsubseteq \forall Q_i \cdot R_i$  is in  $\Sigma$ 
      if IMPLIES( $\Sigma, R_i \sqsubseteq h_i$ ) /* True implies that  $\Sigma \models R_i \sqsubseteq h_i$  and, hence,  $\Sigma \models \forall Q_i \cdot R_i \sqsubseteq \forall Q_i \cdot h_i$ 
        then begin /* since  $\top \sqsubseteq \forall Q_i \cdot R_i$ , we then have  $\Sigma \models \top \sqsubseteq \forall Q_i \cdot h_i$ 
          Replace  $\forall Q_i \cdot h_i$  by  $\top$  in  $\sigma$  /* that is,  $\Sigma \models \top \sqsubseteq \forall Q_i \cdot h_i$ 
          if the right-hand side of  $\sigma$  becomes  $\top \sqcap \dots \sqcap \top$ 
            then return True
          end
        else if there is  $j \in [1, n]$  such that  $P_j = Q_i$ 
          then if NOT IMPLIES( $\Sigma, g_j \sqsubseteq h_i$ ) /* False implies that  $\Sigma \not\models g_j \sqsubseteq h_i$ 
            then return False /* and, hence,  $\Sigma \not\models \forall Q_i \cdot g_j \sqsubseteq \forall Q_i \cdot h_i$ 
          else return False
        end
      end
    return True /* True implies that  $\Sigma \models \sigma$ 
  end

```

Fig. 4. A Polynomial Decision Procedure for Ultralite Schemas and \mathcal{FL} -Inclusions.

repeated replacing, on both sides of σ , “ $\forall p \cdot e_1 \sqcap \forall p \cdot e_2$ ” by “ $\forall p \cdot (e_1 \sqcap e_2)$ ”; (3) reordering the conjuncts. The resulting inclusion is called the *normal form* of σ .

Proposition 1: Let \mathcal{L} be an \mathcal{FL} language. Let σ be inclusion of \mathcal{L} and σ' be its normal form. Then, σ and σ' are logically equivalent. \square

Therefore, in view of Proposition 1, we can always assume that σ is in normal form.

To help understand the *IMPLIES* procedure, we present two very simple examples.

Example 4:

(a) (Same as Example 3) Let $\mathcal{S}=(\mathcal{L},\Sigma)$ be the ultralite schema and $\mathcal{E}=(\mathcal{K},\mathcal{M})$ be the external schema in Example 1. Let $\mathbf{G}^*(\Sigma)$ be the extended dependency graph of Figure 5. Recall that $e:\text{Product}$ and $e:\text{Book}$ are defined as

- (1) $e:\text{Product} \equiv a:\text{Product} \sqcap \exists a:\text{title}$
- (2) $e:\text{Book} \equiv a:\text{Book} \sqcap \exists a:\text{title} \sqcap \exists a:\text{pub}$

IMPLIES establishes that

- (3) $\Sigma \models a:\text{Book} \sqcap \exists a:\text{title} \sqcap \exists a:\text{pub} \sqsubseteq a:\text{Product} \sqcap \exists a:\text{title}$

Indeed, on the first step of the first loop, *IMPLIES* would find a path from $a:\text{Book}$ to $a:\text{Product}$ in $\mathbf{G}^*(\Sigma)$. On the second step, *IMPLIES* would find a trivial path (of length 0) from $\exists a:\text{title}$ to $\exists a:\text{title}$ in $\mathbf{G}^*(\Sigma)$. Then, *IMPLIES* would stop, returning *True*.

(b) This second example illustrates how *IMPLIES* handles sets of constants to solve an instance of the Query Containment Problem. Consider again the ultralite schema $\mathcal{S}=(\mathcal{L},\Sigma)$ of Example 1 and the following two queries:

- (4) $Q_1 \equiv a:\text{Product} \sqcap \forall a:\text{author} \cdot \{ 'Marlowe', 'Shakespeare' \}$
(obtain the set of products that, if the author exists, he is either 'Marlowe' or 'Shakespeare')
- (5) $Q_2 \equiv a:\text{Book} \sqcap \exists a:\text{author} \sqcap \forall a:\text{author} \cdot \{ 'Shakespeare' \}$
(obtain the set of books that the author is known to be 'Shakespeare')

Note that Q_1 is not correctly formulated, since Σ indicates that $a:\text{author}$ has domain $a:\text{Book}$. However, *IMPLIES* would still correctly determine if $\Sigma \models Q_2 \sqsubseteq Q_1$ holds.

IMPLIES would again use the extended dependency graph $\mathbf{G}^*(\Sigma)$ shown in Figure 5. It establishes that $\Sigma \models Q_2 \sqsubseteq Q_1$ as follows. On the first step of the first loop, it would process $a:\text{Product}$ and find a path from $a:\text{Book}$ to $a:\text{Product}$ in $\mathbf{G}^*(\Sigma)$. Passing to the

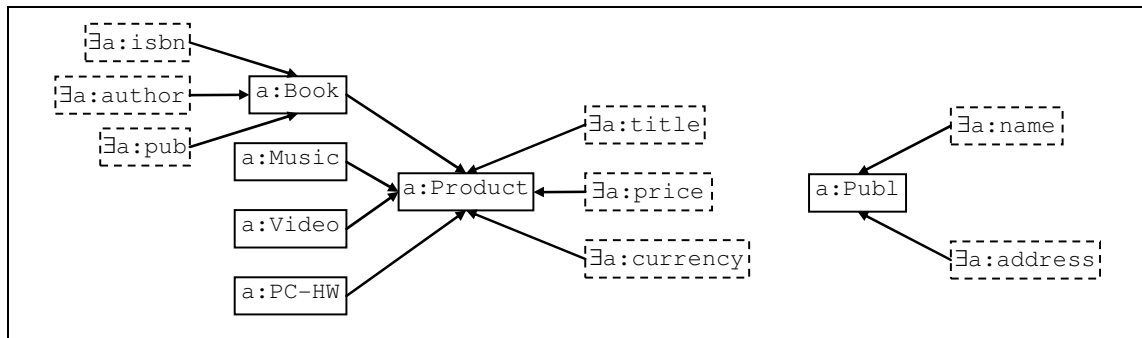


Fig. 5. Dependency graph for the ultralite schema in Example 1.

second loop, on the first step, *IMPLIES* would process $\forall a:\text{author} . \{ 'Marlowe', 'Shakespeare' \}$ and execute a recursive call having as parameter the inclusion $\{ 'Shakespeare' \} \sqsubseteq \{ 'Marlowe', 'Shakespeare' \}$. Since one set is indeed a subset of the other, the recursive call would then return *True*. Since these are the only conjuncts in Q_I , *IMPLIES* would stop, returning *True*. \square

We conclude this section with the main results about the *IMPLIES* procedure.

Theorem 3: Let $\mathcal{S}=(\mathcal{A},\Sigma)$ be an ultralite schema and σ be an inclusion in \mathcal{FL} . Then, *IMPLIES*(Σ,σ) returns *True* iff $\Sigma \models \sigma$, otherwise it returns *False*. \square

(The proof follows from the lemmas shown in the Appendix).

Theorem 4: The special case of the subsumption problem for ultralite schemas and inclusions in \mathcal{FL} is decidable in polynomial time.

Proof

Let $\mathcal{S}=(\mathcal{A},\Sigma)$ be an ultralite schema and σ be an inclusion in \mathcal{FL} . Assume that $\mathbf{G}^*(\Sigma)$ has u nodes and v arcs and that, after normalization, σ is of the form

$$e_1 \sqcap \dots \sqcap e_m \sqcap \forall P_1 \cdot g_1 \sqcap \dots \sqcap \forall P_n \cdot g_n \sqsubseteq f_1 \sqcap \dots \sqcap f_r \sqcap \forall Q_1 \cdot h_1 \sqcap \dots \sqcap \forall Q_s \cdot h_s$$

Assume that the total number of conjuncts on the right-hand side is t , whether the conjuncts occur inside value restrictions or not. Hence, t is the sum of the number of conjuncts which are on the right-hand side of the inclusion in the initial call, and outside value restrictions, plus the number of conjuncts which are on the right-hand side of the inclusions passed as parameters to the recursive calls, and outside value restrictions. For each such conjunct, *IMPLIES* either makes 1 path search in $\mathbf{G}^*(\Sigma)$, or at most 2 recursive calls. Hence, including the recursive calls, the body of *IMPLIES* is executed at most $2 \cdot t$ times. Furthermore, since $\mathbf{G}^*(\Sigma)$ has u nodes and v arcs, each path search in $\mathbf{G}^*(\Sigma)$ requires $O(\text{Max}(u,v))$ steps. Therefore, *IMPLIES* is $O(t \cdot \text{Max}(u,v))$. \square

5. Conclusions

We addressed in this paper three database problems in the context of Description Logics (DL) to take advantage of the precise semantics of DL languages and, more importantly, to leverage on the decision procedures developed for DL dialects. However, right from the onset, it became clear that the integrity constraints present in a database schema had to be encoded directly into the decisions procedures to avoid inefficient reductions.

We showed how to extend the familiar tableau decision procedure for the subsumption problem to take into account subset, domain and range constraints. Such procedure works for expressive DL dialects but is still fairly inefficient. We then proceeded to modify the structural subsumption procedure, without impairing its complexity, to account for the classes of integrity constraints considered.

Finally, we remark that both the extended tableau and the extended structural subsumption procedures are of practical significance to the construction of query optimizers and query mediators, as indicated in section 2. They can be extended to account for concrete domains, such strings with the usual operators, by embedding reasoners for such domains directly into the decision procedures. We are also working on extensions that cope with disjointness constraints and cardinality restrictions.

References

- [1] Baader, F.; Nutt, W. (2003) “Basic Description Logics”. In: Baader, F.; Calvanese, D.; McGuinness, D.L.; Nardi, D.; Patel-Schneider, P.F. (Eds) *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, Cambridge, UK.
- [2] Breitman, K., Casanova, M.A, and Truszkowski, W. (2007) *Semantic web: concepts, technologies, and applications*. Springer, London.
- [3] Calvanese, D.; Lenzerini, M.; Nardi, D. (1998) “Description Logics for Conceptual data modeling”. In: Chomicki, J. and Saake, G. (Eds) *Logics for Databases and Information Systems*. Kluwer Academic Publisher.
- [4] Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; Poggi, A.; Rosati, R.; Ruzzi, M. (2008). “Data Integration through DL-Lite-A Ontologies”. In: Proc. 3rd Int. Workshop on Semantics in Data and Knowledge Bases (SDKB 2008), pp. 26-47.
- [5] Casanova, M.A.; Lauschner, T.; Paes Leme, L.A.; Breitman, K.K; Furtado, A.L. (2009) “A Strategy to Revise the Constraints of the Mediated Schema”. (Accepted to the 28th International Conference on Conceptual Modeling - Gramado, Brasil).
- [6] Curino, C.A.; Moon, H.J.; Zaniolo, C. (2008) “Graceful database schema evolution: the PRISM workbench”. In: Proc. of the VLDB Endowment, v.1, n.1, pp. 761-772.
- [7] Donini, F.M. (2003) “Complexity of Reasoning”. In: Baader, F.; Calvanese, D.; McGuinness, D.L.; Nardi, D.; Patel-Schneider, P.F. (Eds) *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, Cambridge, UK.
- [8] Fagin, R.; Kolaitis, P. G.; Popa, L.; Tan, W.-C. (2007) “Quasi-inverses of schema mappings”. In: Proc. PODS '07, pp. 123–132.
- [9] Franconi, E. (2002) "Structural Description Logics: \mathcal{FL} ". In: Description Logics Course. Available at: <http://www.inf.unibz.it/~franconi/dl/course/slides/struct-DL/flminus.pdf>
- [10] Levesque, H.J.; Brachman, R.J. (1987) “Expressiveness and tractability in knowledge representation and reasoning”. *Computational Intelligence* 3, p. 78-93.
- [11] Nardi, D.; Brachman, R.J. (2003) “An Introduction to Description Logics”. In: Baader, F.; Calvanese, D.; McGuinness, D.L.; Nardi, D.; Patel-Schneider, P.F. (Eds) *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, Cambridge, UK.

Appendix: Proof of the Main Lemmas

Lemma 1: Let $S=(\mathcal{L},\Sigma)$ be an ultralite schema and $e \sqsubseteq f$ be an inclusion in \mathcal{L} . Assume that e and f are either atomic concepts or restricted existential quantifications. Then, $\Sigma \models e \sqsubseteq f$ iff there is a path in the extended dependency graph $G^*(\Sigma)$ from e to f .

Proof

Let $S=(\mathcal{L},\Sigma)$ be an ultralite schema and $e \sqsubseteq f$ be an inclusion in \mathcal{L} . Assume that e and f are either atomic concepts or restricted existential quantifications. If e is equal to f then we trivially have that $\Sigma \models e \sqsubseteq e$ iff there is a path, with length 0, in the extended dependency graph $G^*(\Sigma)$ from e to e . So, assume that e is not equal to f .

(\Leftarrow) Assume that there is a path in the extended dependency graph $G^*(\Sigma)$ from e to f . Then, by construction of $G^*(\Sigma)$, we have that $\Sigma \models e \sqsubseteq f$.

(\Rightarrow) Assume that $\Sigma \models e \sqsubseteq f$. By assumption, e is not equal to f . We proceed by contradiction. Assume that there is no path in the extended dependency graph $\mathbf{G}^*(\Sigma)$ from e to f . There are four cases to consider, depending on whether e and f are atomic concepts or restricted existential quantifications.

Case 1: e is an atomic concept E and f is an atomic concept F .

Let \mathbf{r} be any interpretation for \mathcal{L} such that

$$(1) \quad \mathbf{r} \models \Sigma$$

Let Δ^r be the domain of \mathbf{r} and a be a new individual such that $a \notin \Delta^r$. Let \mathbf{s} be an interpretation for \mathcal{L} , with domain Δ^s , constructed as follows:

$$(2) \quad \Delta^s = \Delta^r \cup \{a\}$$

$$(3) \quad \text{for each atomic concept } C \text{ of } \mathcal{L} \text{ such that there is no path in the extended dependency graph } \mathbf{G}^*(\Sigma) \text{ from } E \text{ to } C, \mathbf{s}(C) = \mathbf{r}(C)$$

$$(4) \quad \text{for each atomic concept } C \text{ of } \mathcal{L} \text{ such that there is a path in the extended dependency graph } \mathbf{G}^*(\Sigma) \text{ from } E \text{ to } C \text{ (including } E), \mathbf{s}(C) = \mathbf{r}(C) \cup \{a\}$$

$$(5) \quad \text{for each atomic role } P \text{ of } \mathcal{L}, \mathbf{s}(P) = \mathbf{r}(P)$$

Then, by (1) to (5), we have

$$(6) \quad \mathbf{s} \models \Sigma$$

Since, by assumption, there is no path in the extended dependency graph $\mathbf{G}^*(\Sigma)$ from E to F , by construction of \mathbf{s} , we have

$$(7) \quad a \in \mathbf{s}(E) \text{ and } a \notin \mathbf{s}(F)$$

Hence, by (7), since e is the atomic concept E and f is the atomic concept F

$$(8) \quad \mathbf{s} \models e \sqsubseteq f \text{ fails}$$

Hence, by (6) and (8), we have constructed an interpretation \mathbf{s} for \mathcal{L} such that

$$(9) \quad \mathbf{s} \models \Sigma \text{ holds, but } \mathbf{s} \models e \sqsubseteq f \text{ fails}$$

Contradiction, since we assumed that $\Sigma \models e \sqsubseteq f$.

(The other cases follow likewise and are omitted for brevity).

Therefore, in all cases, we reach a contradiction. \square

Lemma 2: Let $\mathcal{S}=(\mathcal{L},\Sigma)$ be an ultralite schema and $\forall P \cdot e \sqsubseteq f$ be an inclusion in \mathcal{FL} . Assume that f is an atomic concept or a restricted existential quantification. Then, $\Sigma \not\models \forall P \cdot e \sqsubseteq f$.

Proof

Let $\mathcal{S}=(\mathcal{L},\Sigma)$ be an ultralite schema and $\forall P \cdot e \sqsubseteq f$ be an inclusion in \mathcal{FL} . Assume that f is an atomic concept or a restricted existential quantification. Let \mathbf{r} be an interpretation for \mathcal{L} such that $\mathbf{r} \models \Sigma$. Let Δ^r be the domain of \mathbf{r} and a be a new individual such that $a \notin \Delta^r$. Let \mathbf{s} be an interpretation for \mathcal{L} , with domain Δ^s , constructed as follows:

- $\Delta^s = \Delta^r \cup \{a\}$
- for each atomic concept C of \mathcal{L} , $\mathbf{s}(C) = \mathbf{r}(C)$
- for each atomic role P of \mathcal{L} , $\mathbf{s}(P) = \mathbf{r}(P)$
- for each constant a of \mathcal{L} , $\mathbf{s}(a) = \mathbf{r}(a)$

Then, for any concept or role expression g which does not contain a value restriction, by construction of \mathbf{s} and $a \notin \Delta^r$, we have that $a \notin \mathbf{s}(g) = \mathbf{r}(g)$. Therefore, since f is an atomic concept or a restricted existential quantification, we have that $a \notin \mathbf{s}(f) = \mathbf{r}(f)$. Furthermore, for any value restriction $\forall q \cdot h$, by construction of \mathbf{s} and $\Delta^s = \Delta^r \cup \{a\}$, we have that $a \in \mathbf{s}(\forall q \cdot h)$. Therefore, $a \in \mathbf{s}(\forall P \cdot e)$. Hence, $\mathbf{s} \not\models \forall P \cdot e \sqsubseteq f$.

But, since no constraint in Σ has a value restriction on the left-hand side and since $r \models \Sigma$, by construction of s , we have that $s \models \Sigma$. Therefore, we have $s \models \Sigma$ and $s \not\models \forall P \cdot e \sqsubseteq f$, which implies that $\Sigma \not\models \forall P \cdot e \sqsubseteq f$. \square

Lemma 3: Let $S=(\mathcal{L}, \Sigma)$ be an ultralite schema and σ be an inclusion in \mathcal{FL} . Assume that σ is in normal form and is of the form

$$e_1 \sqcap \dots \sqcap e_m \sqcap \forall P_1 \cdot g_1 \sqcap \dots \sqcap \forall P_n \cdot g_n \sqsubseteq f_1 \sqcap \dots \sqcap f_r \sqcap \forall Q_1 \cdot h_1 \sqcap \dots \sqcap \forall Q_s \cdot h_s$$

Then, $\Sigma \models \sigma$ iff

- (i) for each $i \in [1, r]$, there is $j \in [1, m]$ such that $\Sigma \models e_j \sqsubseteq f_i$
- (ii) for each $i \in [1, s]$, either $\Sigma \models R_i \sqsubseteq h_i$, where Σ has a range constraint of the form $\top \sqsubseteq \forall Q_i \cdot R_i$, or there is $j \in [1, n]$ such that P_j is equal to Q_i and $\Sigma \models g_j \sqsubseteq h_i$

Proof

Let $S=(\mathcal{L}, \Sigma)$ be an ultralite schema and σ be an inclusion in \mathcal{FL} . Assume that σ is in normal form and is of the form $e_1 \sqcap \dots \sqcap e_m \sqcap \forall P_1 \cdot g_1 \sqcap \dots \sqcap \forall P_n \cdot g_n \sqsubseteq f_1 \sqcap \dots \sqcap f_r \sqcap \forall Q_1 \cdot h_1 \sqcap \dots \sqcap \forall Q_s \cdot h_s$.

(\Leftarrow) Assume that the conditions of the Lemma hold. Then, for each $i \in [1, r]$, there is $j \in [1, m]$ such that $\Sigma \models e_j \sqsubseteq f_i$. Furthermore, for each $i \in [1, s]$, there is $j \in [1, n]$ such that P_j is equal to Q_i and $\Sigma \models g_j \sqsubseteq h_i$, which implies that $\Sigma \models \forall P_j \cdot g_j \sqsubseteq \forall Q_i \cdot h_i$. Hence, we have that $\Sigma \models \sigma$.

(\Rightarrow) Assume $\Sigma \models \sigma$. We proceed by contradiction. Assume that the conditions of the lemma fail, that is, one of the two conditions below holds

- (1) there is $i \in [1, r]$ such that, for all $j \in [1, m]$, $\Sigma \not\models e_j \sqsubseteq f_i$, or
- (2) there is $i \in [1, s]$ such that
 - (2.1) $\Sigma \not\models R_i \sqsubseteq h_i$, where Σ has a range constraint of the form $\top \sqsubseteq \forall Q_i \cdot R_i$, and
 - (2.2) for all $j \in [1, n]$, if P_j is equal to Q_i then $\Sigma \not\models g_j \sqsubseteq h_i$

Case 1: there is $i \in [1, r]$ such that, for all $j \in [1, m]$, $\Sigma \not\models e_j \sqsubseteq f_i$.

Let $k \in [1, r]$ be such that, for all $j \in [1, m]$, $\Sigma \not\models e_j \sqsubseteq f_k$. Assume first that f_k is a set description. Since, for all $j \in [1, m]$, $\Sigma \not\models e_j \sqsubseteq f_k$, the left-hand side of σ has no set description that defines a subset of f_k . Then, we immediately conclude that $\Sigma \not\models \sigma$. Contradiction.

Assume now that f_k is not a set description. By Lemma 2, we also have that, for all $j \in [1, n]$, $\Sigma \not\models \forall P_j \cdot g_j \sqsubseteq f_k$. Let $d_j = e_j$, for $j \in [1, m]$, and $d_j = \forall P_j \cdot g_j$, for $j \in [1, n]$. Then, for each $j \in [1, m+n]$, there is an interpretation s_j for \mathcal{L} , with domain Δ^j , such that s_j satisfies all formulas in Σ , but s_j does not satisfy $d_j \sqsubseteq f_k$. Assume without loss of generality that the domains of such interpretations are pairwise disjoint.

For each $j \in [1, m+n]$, since s_j does not satisfy $d_j \sqsubseteq f_k$, there is $a_j \in \Delta^j$ such that $a_j \in s_j(d_j)$ and $a_j \notin s_j(f_k)$. Let a be a new individual such that $a \notin \Delta^j$, for all $j \in [1, m+n]$. Let $s_j[a]$ denote s_j where a_j is uniformly replaced by a in the interpretations of all atomic concepts, atomic roles and constants. Let s be an interpretation for \mathcal{L} , with domain Δ^s , constructed as follows:

- $\Delta^s = \Delta^1 \cup \dots \cup \Delta^{m+n} - \{a_1, \dots, a_{m+n}\} \cup \{a\}$ (union of the domains of s_j , with a_j replaced by a)
- for each atomic concept C of \mathcal{L} , $s(C) = s_1[a](C) \cup \dots \cup s_m[a](C)$
- for each atomic role R of \mathcal{L} , $s(R) = s_1[a](R) \cup \dots \cup s_m[a](R)$
- for each constant c of \mathcal{L} , $s(c) = s_1[a](c)$

Then, by construction of s , we have that s satisfies all formulas in Σ . Furthermore, $a \notin s(f_k)$ and, for each $j \in [1, m+n]$, $a \in s(d_j)$. Therefore, $a \in s(d_1 \sqcap \dots \sqcap d_{m+n})$ and $a \notin s(f_k)$. Hence, we trivially have that $a \notin s(f_1 \sqcap \dots \sqcap f_r \sqcap \forall Q_1 \cdot h_1 \sqcap \dots \sqcap \forall Q_s \cdot h_s)$. Recalling the definitions of σ and d_1, \dots, d_{m+n} , we may conclude that $\Sigma \not\models \sigma$. Contradiction.

Case 2: there is $i \in [1, s]$ such that $\Sigma \models R_i \sqsubseteq h_i$ fails, where Σ has a range constraint of the form $\top \sqsubseteq \forall Q_i \cdot R_i$, and also, for all $j \in [1, n]$, if P_j is equal to Q_i then $\Sigma \models g_j \sqsubseteq h_i$ fails.

(Follows likewise and is omitted for brevity).

Therefore, in both cases, we reach a contradiction. \square