TLP **10** (3): 331–359, 2010. © Cambridge University Press 2010
doi:10.1017/S1471068410000116

Inductive Logic Programming in Databases: From DATALOG *to* $\mathcal{DL}$+LOG$^{\neg\vee}$

FRANCESCA A. LISI
Dipartimento di Informatica, Università degli Studi di Bari "Aldo Moro", Italy
(*e-mail:* lisi@di.uniba.it)

submitted 24 April 2009; revised 17 October 2009; accepted 2 February 2010

Abstract

In this paper we address an issue that has been brought to the attention of the database community with the advent of the Semantic Web, i.e., the issue of how ontologies (and semantics conveyed by them) can help solving typical database problems, through a better understanding of Knowledge Representation (KR) aspects related to databases. In particular, we investigate this issue from the ILP perspective by considering two database problems, (i) the definition of views and (ii) the definition of constraints, for a database whose schema is represented also by means of an ontology. Both can be reformulated as ILP problems and can benefit from the expressive and deductive power of the KR framework $\mathcal{DL}$+LOG$^{\neg\vee}$. We illustrate the application scenarios by means of examples.

KEYWORDS: Inductive Logic Programming, relational databases, ontologies, description logics, hybrid Knowledge Representation and reasoning systems

1 Motivation

Inductive Logic Programming (ILP) has been historically concerned with the induction of rules from examples for classification purposes (Nienhuys-Cheng and de Wolf 1997). Due to the close relation between Logic Programming and Relational Databases (Ceri *et al.* 1990), ILP has established itself as a major approach to Relational Data Mining (Džeroski and Lavrač 2001). Indeed, DATALOG (Ceri *et al.* 1989) is the most widely used Knowledge Representation (KR) framework in ILP. Conversely, interesting extensions of DATALOG such as DATALOG$^{\neg\vee}$ (Eiter *et al.* 1997) have attracted very little attention in ILP. Some effort has been made also at making ILP more able to face the challenges posed by Relational Data Mining applications, e.g., scalability (Blockeel *et al.* 1999). However, the actual added value of ILP with respect to far more efficient approaches still remains the use of prior conceptual knowledge (also known as *background knowledge*, or shortly BK) during the learning process which enables the induction of conceptually meaningful rules. Yet, the BK in ILP is often not organized around a well-formed conceptual model. This practice seems to ignore the latest achievements in conceptual modeling such as ontologies.

In Artificial Intelligence, an *ontology* refers to an engineering artifact (more precisely, produced according to the principles of *Ontological Engineering*, Gómez-Pérez *et al.* 2004), constituted by a specific vocabulary used to describe a certain reality, plus a set of explicit assumptions regarding the intended meaning of the vocabulary words. This set of assumptions has usually the form of a first-order logical (FOL) theory, where vocabulary words appear as unary or binary predicate names, respectively called concepts and relations. More formally, an ontology is a formal explicit specification of a shared conceptualization for a domain of interest (Gruber 1993). Among the other things, this definition emphasizes the fact that an ontology has to be specified in a language that comes with a formal semantics. Only by using such a formal approach ontologies provide the machine interpretable meaning of concepts and relations that is expected when using an ontology-based approach. Among the formalisms proposed by Ontological Engineering, the most currently used are *Description Logics* (DLs) (Baader *et al.* 2007). In particular, the advent of the Semantic Web (Berners-Lee *et al.* 2001) has given a tremendous impulse to research on DL-based ontology languages. Indeed the DL $\mathcal{SHIQ}$ (Horrocks *et al.* 2000) has been the starting point for the definition of the W3C standard mark-up language OWL (Horrocks *et al.* 2003). Note that DLs are decidable fragments of FOL that are incomparable with Clausal Logics (CLs) as regards the expressive power (Borgida 1996) and the semantics (Rosati 2005b). Yet, DLs and CLs can be combined according to some limited forms of hybridization. For example, $\mathcal{DL}+\textsc{log}^{\neg\vee}$ is a general KR framework that allows for the tight integration of DLs and $\textsc{Datalog}^{\neg\vee}$ by imposing the condition of weak $\mathcal{DL}$-safeness on hybrid rules (Rosati 2006)[1]. We argue that the adoption of such hybrid KR systems can help overcoming the current difficulties in accommodating ontologies in ILP.

In this paper we address an issue that has been brought to the attention of the database community with the advent of the Semantic Web, i.e., the issue of how ontologies (and semantics conveyed by them) can help solving typical database problems, through a better understanding of KR aspects related to databases. In particular, we investigate this issue from the ILP perspective by considering two database problems:

- the definition of views
- the definition of constraints

for a database whose schema is represented also by means of an ontology. Both can be reformulated as ILP problems and can benefit from the expressive and deductive power of the KR framework $\mathcal{DL}+\textsc{log}^{\neg\vee}$, mainly from its nonmonotonic (NM) features. We illustrate the application scenarios by means of examples.

The paper is organized as follows. Section 2 provides basic notions on DLs, a short summary of KR research on the integration of DLs and CLs, and a brief introduction to ILP. Section 3 introduces syntax, semantics, and reasoning of $\mathcal{DL}+\textsc{log}^{\neg\vee}$. Sections 4 and 5 define the ILP proposals for inducing database views

[1] We prefer to use the name $\mathcal{DL}+\textsc{log}^{\neg\vee}$ instead of the original one $\mathcal{DL}+\textsc{log}$ in order to emphasize the $\textsc{Datalog}^{\neg\vee}$ component of the framework.

Table 1. *Syntax and semantics of some typical DL constructs*

Bottom (resp. top) concept	$\bot$ (resp. $\top$)	$\emptyset$ (resp. $\Delta^{\mathcal{I}}$)
Atomic concept	A	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
(Abstract) simple role	S	$S^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
(Abstract) individual	a	$a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$
Concept	C	
Role	R	
Concept negation	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
Concept intersection	$C_1 \sqcap C_2$	$C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}$
Concept union	$C_1 \sqcup C_2$	$C_1^{\mathcal{I}} \cup C_2^{\mathcal{I}}$
Value restriction	$\forall R.C$	$\{x \in \Delta^{\mathcal{I}} \mid \forall y\ (x,y) \in R^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}}\}$
Existential restriction	$\exists R.C$	$\{x \in \Delta^{\mathcal{I}} \mid \exists y\ (x,y) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$
At least number restriction	$\geqslant nR$	$\{x \in \Delta^{\mathcal{I}} \mid \lvert\{y \mid (x,y) \in R^{\mathcal{I}}\}\rvert \geqslant n\}$
At most number restriction	$\leqslant nR$	$\{x \in \Delta^{\mathcal{I}} \mid \lvert\{y \mid (x,y) \in R^{\mathcal{I}}\}\rvert \leqslant n\}$
At least qualif. number restriction	$\geqslant nR.C$	$\{x \in \Delta^{\mathcal{I}} \mid \lvert\{y \in C^{\mathcal{I}} \mid (x,y) \in R^{\mathcal{I}}\}\rvert \geqslant n\}$
At most qualif. number restriction	$\leqslant nR.C$	$\{x \in \Delta^{\mathcal{I}} \mid \lvert\{y \in C^{\mathcal{I}} \mid (x,y) \in R^{\mathcal{I}}\}\rvert \leqslant n\}$
Role inversion	R^-	$\{(x,y) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \mid (y,x) \in R^{\mathcal{I}}\}$
Role intersection	$R_1 \sqcap R_2$	$R_1^{\mathcal{I}} \cap R_2^{\mathcal{I}}$

and database constraints, respectively, within the $\mathcal{DL}$+LOG$^{\neg\vee}$ framework. Section 6 surveys related work. Section 7 concludes the paper with final remarks.

2 Background

2.1 Representing ontologies

DLs are a family of decidable FOL fragments that allow for the specification of knowledge in terms of classes (*concepts*), instances (*individuals*), and binary relations between instances (*roles*) (Borgida 1996). Complex concepts can be defined from atomic concepts and roles by means of constructors. Syntax and semantics of some typical DL constructs are reported in Table 1. For example, concept descriptions in the basic DL $\mathcal{AL}$ are formed according to only the constructors of atomic negation, concept conjunction, value restriction, and limited existential restriction. The DLs $\mathcal{ALC}$ and $\mathcal{ALN}$ are members of the $\mathcal{AL}$ family. The former extends $\mathcal{AL}$ with (arbitrary) concept negation (also called complement and equivalent to having both concept union and full existential restriction), whereas the latter with number restriction. The DL $\mathcal{ALCNR}$ adds to the constructors inherited from $\mathcal{ALC}$ and $\mathcal{ALN}$ a further one: role intersection. Conversely, in the DL $\mathcal{SHIQ}$ (Horrocks *et al.* 2000) it is allowed to invert roles and to express qualified number restrictions of the form $\geqslant nR.C$ and $\leqslant nR.C$ where R is a simple role. Also transitivity holds for roles. A role (expression) is called complex if it contains any role operations other than inversion, e.g., role intersection.

A DL knowledge base (KB) Σ can state both is-a relations between concepts (*axioms*) and instance-of relations between individuals (resp. couples of individuals) and concepts (resp. roles) (*assertions* or *facts*). Axioms form the so-called *terminological box* (TBox) $\mathcal{T}$ whereas facts are contained in the so-called *assertional box*

Table 2. *Syntax and semantics of DL KBs*

Concept equivalence axiom	$C_1 \equiv C_2$	$C_1^{\mathcal{I}} = C_2^{\mathcal{I}}$
Concept subsumption axiom	$C_1 \sqsubseteq C_2$	$C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$
Role equivalence axiom	$R_1 \equiv R_2$	$R_1^{\mathcal{I}} = R_2^{\mathcal{I}}$
Role inclusion axiom	$R_1 \sqsubseteq R_2$	$R_1^{\mathcal{I}} \subseteq R_2^{\mathcal{I}}$
Concept assertion	$C(a)$	$a^{\mathcal{I}} \in C^{\mathcal{I}}$
Role assertion	$R(a,b)$	$(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$
Individual equality assertion	$a \approx b$	$a^{\mathcal{I}} = b^{\mathcal{I}}$
Individual inequality assertion	$a \not\approx b$	$a^{\mathcal{I}} \neq b^{\mathcal{I}}$

(ABox) $\mathcal{A}$. A $\mathcal{SHIQ}$ KB encompasses also a role box (RBox) $\mathcal{R}$ which consists of a finite set of role equivalence and role inclusion axioms. Therefore hierarchies can be defined over not only concepts but also roles. Transitivity of roles is also specified by means of axioms. Thus, when a DL-based ontology language is adopted, an ontology is nothing else than a TBox, possibly together with a RBox. If the ontology is populated, it corresponds to a whole DL KB, i.e., encompassing also an ABox. The semantics of DLs can be defined directly with set-theoretic formalizations as shown in Table 2 or through a mapping to FOL as shown in Borgida (1996). An *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ for a DL KB consists of a domain $\Delta^{\mathcal{I}}$ and a mapping function $\cdot^{\mathcal{I}}$. Under the *Unique Names Assumption* (UNA) (Reiter 1980), individuals are mapped to elements of $\Delta^{\mathcal{I}}$ such that $a^{\mathcal{I}} \neq b^{\mathcal{I}}$ if $a \neq b$. Yet UNA does not hold by default in DLs. Thus individual equality (inequality) assertions may appear in a DL KB (see Table 2). An interpretation $\mathcal{I}$ is a *model* of a KB $\Sigma = (\mathcal{T}, \mathcal{A})$ iff it satisfies all axioms and assertions in $\mathcal{T}$ and $\mathcal{A}$. Also the KB represents many different interpretations, i.e., all its models. This is coherent with the *Open World Assumption* (OWA) that holds in FOL semantics. A DL KB is *satisfiable* if it has at least one model. An ABox assertion α is a *logical consequence* of a KB Σ, written $\Sigma \models \alpha$, if all models of Σ are also models of α.

The main reasoning task for a DL KB Σ is the *consistency check* which tries to prove the satisfiability of Σ. The consistency check is performed by applying decision procedures mostly based on tableau calculus. Another well-known reasoning service in DLs is *instance check*, i.e., the check of whether an ABox assertion is a logical implication of a DL KB. A more sophisticated version of instance check, called *instance retrieval*, retrieves, for a DL KB Σ, all (ABox) individuals that are instances of the given (possibly complex) concept expression C, i.e., all those individuals a such that Σ entails that a is an instance of C. In data-intensive applications, querying KBs plays a central role. Instance retrieval is, in some aspects, a rather weak form of querying: although possibly complex concept expressions are used as queries, we can only query for tree-like relational structures, i.e., a DL concept cannot express arbitrary cyclic structures. The possibility of expressing *conjunctive queries* (CQ) and *unions of conjunctive queries* (UCQ) is widely studied in DLs. Let $P_{\mathcal{C}}$ and $P_{\mathcal{R}}$ be the alphabets of concept names and role names, respectively. A *Boolean UCQ* over the alphabet $P_{\mathcal{C}} \cup P_{\mathcal{R}}$ is a FOL sentence of the form $q_1 \vee \ldots \vee q_n$, where each q_i is a conjunction $\exists \vec{X} conj_i(\vec{X})$ of atoms whose predicates are in $P_{\mathcal{C}} \cup P_{\mathcal{R}}$ and whose arguments are either constants or variables from the tuple $\vec{X}$. A *Boolean CQ* corresponds to a Boolean UCQ in the case when $n = 1$. The *Boolean UCQ entailment problem* in DLs is defined as follows: A KB Σ entails a UCQ $Q = q_1 \vee \ldots \vee q_n$, written as $\Sigma \models q_1 \vee \ldots \vee q_n$, if, for every model $\mathcal{I}$ of Σ, there is some i such that q_i is satisfied in $\mathcal{I}$ and $1 \leqslant i \leqslant n$. Note that instance check can be expressed as the problem of query entailment problem of a Boolean CQs constituted by just one ground atom. The *Boolean CQ/UCQ containment problem*[2] in DLs is defined as follows: Given a $\mathcal{DL}$-TBox $\mathcal{T}$, a Boolean CQ Q_1 and a Boolean UCQ Q_2 over the alphabet $P_{\mathcal{C}} \cup P_{\mathcal{R}}$, Q_1 is contained in Q_2 with respect to $\mathcal{T}$, denoted by $\mathcal{T} \models Q_1 \subseteq Q_2$, iff, for every model $\mathcal{I}$ of $\mathcal{T}$, if Q_1 is satisfied in $\mathcal{I}$ then Q_2 is satisfied in $\mathcal{I}$. This problem has been proved decidable for many DLs, notably for the very expressive $\mathcal{SHIQ}$ (Glimm *et al.* 2008) and $\mathcal{SHOQ}$ (Glimm *et al.* 2008). Finally, when the UNA does not hold, it can be immediately reduced to the Boolean UCQ entailment problem (Calvanese *et al.* 2008). In the rest of the paper we shall consider DLs without UNA.

2.2 *Integrating ontologies and relational databases*

The integration of ontologies and relational databases follows the tradition of KR research on *hybrid systems*, i.e., those systems which are constituted by two or more subsystems dealing with distinct portions of a single KB by performing specific reasoning procedures (Frisch and Cohn 1991). The motivation for investigating and developing such systems is to improve on two basic features of KR formalisms, namely *representational adequacy* and *deductive power*, by preserving the other crucial feature, i.e., *decidability*. Those KR systems that integrate ontologies and relational databases will be referred to as DL-CL hybrid KR systems in the rest of the paper. They implement different solutions to the problem of combining DLs and CLs. Indeed DLs and CLs are FOL fragments incomparable as for the expressiveness (Borgida 1996) and the semantics (Rosati 2005a) but combinable at different degrees of integration. The integration is said to be *tight* when a model of the hybrid KB is defined as the union of two models, one for the DL part and one for the CL part, which share the same domain. In particular, combining DLs with CLs in a tight manner can easily yield to undecidability if the interaction scheme between the DL and the CL part of a hybrid KB does not fulfill some condition of *safeness*(Rosati 2005b). Indeed safeness allows to solve the semantic mismatch between DLs and CLs, namely the OWA for DLs and the CWA for CLs[3]. In the following we shall briefly describe two exemplary cases of tightly integrated DL-CL hybrid KR systems: $\mathcal{AL}$-log (Donini *et al.* 1998) and CARIN (Levy and Rousset 1998). The former is safe whereas the latter is not.

$\mathcal{AL}$-log (Donini *et al.* 1998) is a hybrid KR system that integrates $\mathcal{ALC}$ (Schmidt-Schauss and Smolka 1991) and DATALOG (Ceri *et al.* 1989). In particular, variables occurring in the body of rules may be constrained with $\mathcal{ALC}$ concept assertions to be used as "typing constraints." This makes rules applicable only to explicitly

[2] This problem was called *existential entailment* in Levy and Rousset (1998).

[3] Note that the OWA and CWA have a strong influence on the results of reasoning.

named objects. A further restriction is that only Datalog atoms are allowed in rule heads. Reasoning for $\mathcal{AL}$-log knowledge bases is based on *constrained SLD-resolution*, i.e., an extension of SLD-resolution with a tableau calculus for $\mathcal{ALC}$ to deal with constraints. Constrained SLD-resolution is *decidable* and runs in single nondeterministic exponential time. Constrained SLD-refutation is a complete and sound method for answering *ground* queries, i.e., conjunctions of ground Datalog atoms and $\mathcal{ALC}$ concept assertions.

A comprehensive study of the effects of combining DLs and CLs can be found in Levy and Rousset (1998). Here the family Carin of hybrid languages is presented. Special attention is devoted to the DL $\mathcal{ALCNR}$. The results of the study can be summarized as follows: (i) answering CQs over $\mathcal{ALCNR}$ TBoxes is decidable, (ii) query answering in a logic obtained by extending $\mathcal{ALCNR}$ with nonrecursive Datalog rules, where both concepts and roles can occur in rule bodies, is also decidable, as it can be reduced to answering a UCQ, (iii) if rules are recursive, query answering becomes undecidable, (iv) decidability can be regained by disallowing certain combinations of constructors in the logic, and (v) decidability can be regained by requiring rules to be *role-safe*, where at least one variable from each role literal must occur in some non-DL-atom. As in $\mathcal{AL}$-log, query answering is decided using constrained resolution and a modified version of tableau calculus.

2.3 Learning rules with ILP

Inductive Logic Programming was born at the intersection between Logic Programming and Concept Learning (Muggleton 1990). From Logic Programming it has borrowed the KR framework, i.e., Horn Clausal Logic (HCL). From Concept Learning it has inherited the inferential mechanisms for induction, the most prominent of which is *generalization*. Concept Learning is concerned with the problem of automatically inducing the general definition of some concept (called *target*), given *examples* labeled as instances or noninstances of the concept. In ILP the target is the predicate whose definition is returned by the inductive learning process as a *hypothesis*. The definition may consist of one or more clauses. A distinguishing feature of ILP with respect to other forms of Concept Learning is the use of prior knowledge of the domain of interest, called *background knowledge* (BK). Therefore, induction with ILP generalizes from individual instances/observations in the presence of BK, finding valid hypotheses. *Validity* depends on the underlying *setting*.

2.3.1 Settings

At present, there exist several formalizations of induction in ILP that can be classified according to the following two orthogonal dimensions: the *scope of induction* (discrimination versus characterization) and the *representation of observations* (ground definite clauses versus ground unit clauses) (De Raedt and Dehaspe 1997). *Discriminant induction* aims at inducing hypotheses with discriminant power as required in tasks such as classification where observations encompass both positive and negative examples. *Characteristic induction* is more suitable for finding regularities in a data set. This corresponds to learning from positive examples only. For a thorough discussion of differences between discriminant and characteristic induction see (Michalski 1983). The second dimension affects the notion of *coverage*, i.e., the condition under which a hypothesis explains/confirms an observation. In *learning from entailment* (also called normal or explanatory ILP setting), hypotheses are clausal theories, observations are ground definite clauses, and a hypothesis covers an observation if the hypothesis logically entails the observation (Frazier and Pitt 1993). In *learning from interpretations* (also called nonmonotonic or confirmatory ILP setting), hypotheses are clausal theories, observations are Herbrand interpretations (ground unit clauses) and a hypothesis covers an observation if the observation is a model for the hypothesis (De Raedt and Džeroski 1994). Summing up, when learning from entailment with the aim of discrimination, a hypothesis is valid (or correct) if it logically entails all positive examples and none of the negative examples. The former condition of validity is called *completeness*, whereas the latter is referred to as *consistency*. If the scope of induction is characterization, the condition of consistency is dropped out from the notion of validity due to the absence of negative examples. The two settings for the case of learning from interpretations can be defined similarly.

2.3.2 Techniques

In Concept Learning, thus in ILP, generalization is traditionally viewed as search through a partially ordered space of inductive hypotheses (Mitchell 1982). According to this vision, an inductive hypothesis is a clausal theory and the induction of a single clause requires (i) structuring, (ii) searching, and (iii) bounding the space of clauses (Nienhuys-Cheng and de Wolf 1997).

First, we focus on (i) by clarifying how the algebraic notion of ordering can be applied to clauses. A *generality relation* allows for determining which one, between two clauses, is more general than the other. It defines a preorder (or quasi-order) on the set of clauses, i.e., a partially ordered set of equivalence classes. One such ordering is *θ-subsumption* (Plotkin 1970): Given two clauses C and D, we say that C θ-subsumes D if there exists a substitution θ, such that $C\theta \subseteq D$[4]. Given the usefulness of BK, orders have been proposed that reckon with it. Among them is *relative subsumption* (Plotkin 1971): Given two clauses C and D and a clausal theory $\mathcal{K}$, we say that C subsumes D relative to $\mathcal{K}$ if there exists a substitution θ such that $\mathcal{K} \models \forall(C\theta \implies D)$. Also, *generalized subsumption* (Buntine 1988) is of interest to this paper: Given two definite clauses C and D standardized apart[5] and a definite program $\mathcal{K}$, we say that C subsumes D w.r.t. $\mathcal{K}$ iff there exists a ground substitution θ for C such that (i) $head(C)\theta = head(D)\sigma$ and (ii) $\mathcal{K} \cup body(D)\sigma \models body(C)\theta$ where

[4] This definition relies on the set notation for clauses.
[5] Two clauses C and D are said to be *standardized apart* if they have no variables in common.

σ is a Skolem substitution[6] for D with respect to $\{C\} \cup \mathcal{K}$. In the general case, generalized subsumption is undecidable and does not introduce a lattice[7] on a set of clauses. Because of these problems, θ-subsumption is more frequently used in ILP systems. Yet for DATALOG generalized subsumption is decidable and admits a least general generalization.

Once structured according to a generality order, the space of hypotheses can be searched (ii) by means of refinement operators. A *refinement operator* is a function which computes a set of specializations or generalizations of a clause according to whether a top-down or a bottom-up search is performed. The two kinds of refinement operator have been therefore called *downward* and *upward*, respectively. A good refinement operator should satisfy certain desirable properties (van der Laag 1995). We shall illustrate these properties for the case of downward refinement operators but analogous conditions are actually required to hold for the upward ones as well. Ideally, a downward refinement operator should compute only a finite set of specializations of each clause—otherwise it will be of limited practical use. When it accomplishes this condition, it is called *locally finite*. Furthermore, it should be *complete*: every specialization should be reachable by a finite number of applications of the operator. Finally, it is better only to compute *proper* specializations of a clause, for otherwise repeated application of the operator might get stuck in a sequence of equivalent clauses, without ever achieving any real specialization. Operators that satisfy all these conditions simultaneously are called *ideal*. It has been shown that ideal refinement operators do not exist for both full and Horn clausal languages ordered by either subsumption or the stronger orders (e.g., implication).

In order to define a refinement operator for full clausal languages, it is necessary to drop one of the three properties of idealness. Since local finiteness and completeness are usually considered the most important among these properties, this means that locally finite and complete, but improper refinement operators can be defined for full clausal languages. On the other hand, in order to retain all the three properties of idealness, it seems that the only possibility is to restrict the search space. Hence, the definition of refinement operators is usually coupled with the specification of a declarative bias for bounding the space of clauses (iii). *Bias* concerns anything which constrains the search for theories, e.g., a *language bias* specifies syntactic constraints on the clauses in the search space. One such constraint is *connectedness*: A clause C is connected if each variable occurring in $head(C)$ also occurs in $body(C)$. The constraint of *linkedness* is also widely used: A definite clause C is linked if each literal $l_i \in C$ is linked. A literal $l_i \in C$ is linked if at least one of its terms is linked. A term t in some literal $l_i \in C$ is linked with linking chain of length 0, if t occurs in $head(C)$, and with linking chain of length $d + 1$, if some other term in l_i is linked

[6] Let $\mathcal{B}$ be a clausal theory and C be a clause. Let $X_1, \ldots, X_n$ be all the variables appearing in C, and $a_1, \ldots, a_n$ be distinct constants (individuals) not appearing in $\mathcal{B}$ or C. Then the substitution $\{X_1/a_1, \ldots, X_n/a_n\}$ is called a *Skolem substitution* for C w.r.t. $\mathcal{B}$.

[7] A *lattice* is a partially ordered set (also called a poset) in which any two elements have a unique supremum (the elements' least upper bound) and an infimum (greatest lower bound).

with linking chain of length d. The link-depth of a term t in l_i is the length of the shortest linking chain of t.

3 Integrating ontologies and databases with $\mathcal{DL}$+LOG$^{\neg\vee}$

The KR framework of $\mathcal{DL}$+LOG$^{\neg\vee}$ (Rosati 2006) allows for the tight integration of DLs (Baader *et al.* 2007) and DATALOG$^{\neg\vee}$ (Eiter *et al.* 1997). More precisely, it allows a $\mathcal{DL}$ KB to be extended with DATALOG$^{\neg\vee}$ rules according to the so-called *weak safeness* condition as shown in the following.

3.1 Syntax

Formulas in $\mathcal{DL}$+LOG$^{\neg\vee}$ are built upon three mutually disjoint predicate alphabets: an alphabet $P_{\mathcal{C}}$ of concept names, an alphabet $P_{\mathcal{R}}$ of role names, and an alphabet P_{D} of DATALOG predicates. We call a predicate p a *DL-predicate* if either $p \in P_{\mathcal{C}}$ or $p \in P_{\mathcal{R}}$. Then, we denote by $\mathcal{N}$ a countably infinite alphabet of constant names. An *atom* is an expression of the form $p(\vec{X})$, where p is a predicate of arity n and $\vec{X}$ is a n-tuple of variables and constants. If no variable symbol occurs in $\vec{X}$, then $p(\vec{X})$ is called a *ground atom* (or *fact*). If $p \in P_{\mathcal{C}} \cup P_{\mathcal{R}}$, the atom is called a *DL-atom*, while if $p \in P_{\mathrm{D}}$, it is called a DATALOG *atom.*

Definition 1
Given a description logic $\mathcal{DL}$, a $\mathcal{DL}$+LOG$^{\neg\vee}$ KB $\mathcal{B}$ is a pair (Σ, Π), where Σ is a $\mathcal{DL}$ KB and Π is a set of DATALOG$^{\neg\vee}$ rules, where each rule R has the form

$$p_1(\vec{X}_1) \vee \ldots \vee p_n(\vec{X}_n) \leftarrow \\ r_1(\vec{Y}_1), \ldots, r_m(\vec{Y}_m), s_1(\vec{Z}_1), \ldots, s_k(\vec{Z}_k), not\ u_1(\vec{W}_1), \ldots, not\ u_h(\vec{W}_h) \quad (1)$$

with $n, m, k, h \geqslant 0$, each $p_i(\vec{X}_i)$, $r_j(\vec{Y}_j)$, $s_l(\vec{Z}_l)$, $u_k(\vec{W}_k)$ is an atom and:

- each p_i is either a DL-predicate or a DATALOG predicate;
- each r_j, u_k is a DATALOG predicate;
- each s_l is a DL-predicate;
- (DATALOG-safeness) every variable occurring in R must appear in at least one of the atoms $r_1(\vec{Y}_1), \ldots, r_m(\vec{Y}_m), s_1(\vec{Z}_1), \ldots, s_k(\vec{Z}_k)$;
- (weak $\mathcal{DL}$-safeness) every head variable of R must appear in at least one of the atoms $r_1(\vec{Y}_1), \ldots, r_m(\vec{Y}_m)$.

We remark that the condition of weak $\mathcal{DL}$-safeness allows for the presence of variables that only occur in DL-atoms in the body of R. This condition allows to overcome the main representational limits of the safe approaches by keeping the integration scheme still decidable. Indeed, the notion of $\mathcal{DL}$-safeness proposed in Motik *et al.* (2005) can be expressed as follows: every variable of R must appear in at least one of the atoms $r_1(\vec{Y}_1), \ldots, r_m(\vec{Y}_m)$. Therefore, $\mathcal{DL}$-safeness forces every variable of R to occur also in the DATALOG atoms in the body of R. This disables the possibility of expressing CQs and UCQs. By weakening the $\mathcal{DL}$-safeness condition, this possibility can be enabled. For these reasons, $\mathcal{DL}$+LOG$^{\neg\vee}$ is located between $\mathcal{AL}$-log and CARIN along the expressivity line.

Without loss of generality, we can assume that in a $\mathcal{DL}$+LOG$^{\neg\vee}$ KB (Σ, Π) all constants occurring in Σ also occur in Π.

Example 1
Let us consider a $\mathcal{DL}$+LOG$^{\neg\vee}$ KB $\mathcal{B}$ (adapted from Rosati 2006) integrating the following DL-KB Σ (ontology about persons)

[A1] `PERSON` $\sqsubseteq \exists$ `FATHER`$^{-}$`.MALE`
[A2] `MALE` $\sqsubseteq$ `PERSON`
[A3] `FEMALE` $\sqsubseteq$ `PERSON`
[A4] `FEMALE` $\sqsubseteq \neg$`MALE`
`MALE(Bob)`
`PERSON(Mary)`
`PERSON(Paul)`
`FATHER(John,Paul)`

and the following DATALOG$^{\neg\vee}$ program Π (database about students):

[R1] `boy(X)` $\leftarrow$ `enrolled(X,c1,ft)`, `PERSON(X)`, `not girl(X)`
[R2] `girl(X)` $\leftarrow$ `enrolled(X,c2,ft)`, `PERSON(X)`
[R3] `boy(X)`$\vee$ `girl(X)` $\leftarrow$ `enrolled(X,c3,ft)`, `PERSON(X)`
[R4] `FEMALE(X)` $\leftarrow$ `girl(X)`
[R5] `MALE(X)` $\leftarrow$ `boy(X)`
[R6] `man(X)` $\leftarrow$ `enrolled(X,c3,pt)`, `FATHER(X,Y)`
`enrolled(Paul,c1,ft)`
`enrolled(Mary,c1,ft)`
`enrolled(Mary,c2,ft)`
`enrolled(Bob,c3,ft)`
`enrolled(John,c3,pt)`

encompassing rules that mix DL-literals and DATALOG-literals. The rule [R3], e.g., says that: If `X` is a `PERSON` enrolled in the course `c3` as a full-time student (`ft`), then `X` is either a `boy` or a `girl`. The rule [R6] says that: If `X` is a `FATHER` (of some `Y`) enrolled in the course `c3` as a part-time student (`pt`), then `X` is a `man`. Notice that the variable `Y` in R6 is weakly safe but not DL-safe, since `Y` does not occur in any DATALOG literal of R6.

3.2 Semantics

For $\mathcal{DL}$+LOG$^{\neg\vee}$ two semantics have been defined: a FOL semantics and a NM semantics. The FOL semantics does not distinguish between head atoms and negated body atoms. Thus, the rule (1) is equivalent to

$$p_1(\vec{X}_1) \vee \ldots \vee p_n(\vec{X}_n) \vee u_1(\vec{W}_1) \vee \ldots \vee u_h(\vec{W}_h) \leftarrow \\ r_1(\vec{Y}_1), \ldots, r_m(\vec{Y}_m), s_1(\vec{Z}_1), \ldots, s_k(\vec{Z}_k) \text{ (2).}$$

The NM semantics is based on the stable model semantics of DATALOG$^{\neg\vee}$. According to it, DL-predicates are still interpreted under OWA, while DATALOG predicates are interpreted under CWA. Notice that, under both semantics, entailment can be reduced to satisfiability, since it is possible to express constraints in the DATALOG program. In particular, it is immediate to verify the following theorem on ground query answering (Rosati 2006).

Theorem 1
Given a $\mathcal{DL}$+LOG$^{\neg\vee}$ KB (Σ, Π) and a ground atom α, $(\Sigma, \Pi) \models \alpha$ iff $(\Sigma, \Pi \cup \{\leftarrow \alpha\})$ is unsatisfiable.

Analogously, CQ answering can be reduced to satisfiability in DATALOG$^{\neg\vee}$, more precisely it can be performed by means of multiple satisfiability tests. Consequently, Rosati (2006) concentrates on the satisfiability problem in $\mathcal{DL}$+LOG$^{\neg\vee}$ KBs. It has been shown that, when the rules are made out of DATALOG$^{\vee}$ (i.e., without negated atoms), the above two semantics are equivalent with respect to the satisfiability problem. In particular, FOL-satisfiability can always be reduced (in linear time) to NM-satisfiability by rewriting rules from the form (1) to the form (2). Hence, only the satisfiability problem under the NM semantics is deeply treated in Rosati (2006).

Example 2
With reference to Example 1, it can be easily verified that all NM-models for $\mathcal{B}$ satisfy the following ground atoms:

(1) `boy(Paul)` (since rule $[R1]$ is always applicable for {`X/Paul`} and $[R1]$ acts like a default rule, which can be read as follows: if `X` is a person enrolled in course `c1`, then `X` is a boy, unless we know for sure that `X` is a girl);
(2) `girl(Mary)` (since rule $[R2]$ is always applicable for {`X/Mary`});
(3) `boy(Bob)` (since rule $[R3]$ is always applicable for {`X/Bob`}, and, by rule $[R4]$, the conclusion `girl(Bob)` is inconsistent with Σ);
(4) `MALE(Paul)` (due to rule $[R5]$ and conclusion 1);
(5) `FEMALE(Mary)` (due to rule $[R4]$ and conclusion 2).

Notice that $\mathcal{B} \models_{NM}$`FEMALE(Mary)`, while $\Sigma \not\models_{FOL}$ `FEMALE(Mary)`. In other words, adding rules has indeed an effect on the conclusions one can draw about DL-predicates. Moreover, such an effect also holds under the FOL semantics of $\mathcal{DL}$+LOG-KBs, since it can be verified that $\mathcal{B} \models_{FOL}$`FEMALE(Mary)` in this case.

3.3 Reasoning

The problem statement of NM-satisfiability for finite $\mathcal{DL}$+LOG$^{\neg\vee}$ KBs relies on the aforementioned Boolean CQ/UCQ containment problem for the $\mathcal{DL}$ part and on the so-called DL-grounding of the DATALOG$^{\neg\vee}$ component. In particular, DL-grounding is an adaptation of the grounding operation used in stable model semantics to the $\mathcal{DL}$+LOG$^{\neg\vee}$ case.

Given a $\mathcal{DL}$+LOG$^{\neg\vee}$ KB $\mathcal{B} = (\Sigma, \Pi)$, we denote by $\mathcal{C}_{\Pi}$ the set of constants occurring in Π. The *DL-grounding* of Π, denoted as $gr_p(\Pi)$, is a set of Boolean CQs obtained by grounding all and only the DL-parts of rule bodies and the DL-atoms appearing in rule heads in Π with respect to the constants in $\mathcal{C}_{\Pi}$. Note that grounding in $gr_p(\Pi)$ is partial, since the variables that only occur in DL-atoms in the body of

```
NMSAT-$\mathcal{DL}$+log($\mathcal{B}$)
1. satisfiable=false
2. if there exists a partition $(G_P, G_N)$ of $gr_p(\Pi)$ such that
3.         (a) $\Pi(G_P, G_N)$ has a stable model and
4.         (b) $\mathcal{T} \models CQ(\mathcal{A} \cup G_P) \subset UCQ(G_N)$
5.         then satisfiable=true
6. endif
return satisfiable
```

Fig. 1. The algorithm NMSAT-$\mathcal{DL}$+LOG (Rosati 2006).

rules are not replaced by constants in $gr_p(\Pi)$. Similarly to $gr_p(\Pi)$, we define the partial grounding of Π on $\mathcal{C}_\Pi$, denoted as $pgr(\Pi, \mathcal{C}_\Pi)$, as the program obtained from Π by grounding with the constants in $\mathcal{C}_\Pi$ all variables except for the existential variables of rules that only occur in DL-atoms. Finally, given a partition (G_P, G_N) of $gr_p(\Pi)$, we denote by $\Pi(G_P, G_N)$ the ground DATALOG$^{\neg\vee}$ program obtained from $pgr(\Pi, \mathcal{C}_\Pi)$ by taking into account the two sets G_P and G_N so that no DL-predicate occurs in such a program.

Let G be a set of Boolean CQs. Then, we denote by $CQ(G)$ (resp. $UCQ(G)$) the Boolean CQ (resp. UCQ) corresponding to the conjunction (resp. disjunction) of all the Boolean CQs in G. The algorithm NMSAT-$\mathcal{DL}$+log for deciding NM-satisfiability of $\mathcal{DL}$+LOG$^{\neg\vee}$ KBs has a very simple structure (see Fig. 1). It guesses a partition (G_P, G_N) of $gr_p(\Pi)$ that is consistent with the $\mathcal{DL}$-KB $\Sigma = (\mathcal{T}, \mathcal{A})$ (Boolean CQ/UCQ containment problem) and such that $\Pi(G_P, G_N)$ has a stable model. More details can be found in Rosati (2006).

The decidability of reasoning, thus of ground query answering, in $\mathcal{DL}$+LOG$^{\neg\vee}$ depends on the decidability of the Boolean CQ/UCQ containment problem in $\mathcal{DL}$.

Theorem 2
For any $\mathcal{DL}$, satisfiability of $\mathcal{DL}$+LOG$^{\neg\vee}$ KBs (under both FOL and NM semantics) is decidable iff Boolean CQ/UCQ containment is decidable in $\mathcal{DL}$ (Rosati 2006).

From Theorem 2 and from previous results on query answering and query containment in DLs, it follows the decidability of reasoning in several instantiations of $\mathcal{DL}$+LOG$^{\neg\vee}$. In all these decidable cases, ground queries can be answered by applying NMSAT-$\mathcal{DL}$+LOG.

The complexity of reasoning in $\mathcal{DL}$+LOG$^{\neg\vee}$ depends on the specific $\mathcal{DL}$ chosen for instantiating the framework. We remind the reader to Rosati (2006) for the analysis of some cases.

4 Inducing database views in $\mathcal{DL}$+LOG$^{\neg}$ with ILP

In this section we consider the problem of defining a new view in a database whose schema is partly represented by an ontology. We suppose that there are tuples known to belong to the view as well as tuples known not to belong to the view. Cast in the $\mathcal{DL}$+LOG$^{\neg}$ framework, this problem boils down to the problem of building $\mathcal{DL}$+LOG$^{\neg}$ rules defining a DATALOG predicate p which stands for the view name. Tuples are ground DATALOG facts that are true for p if they belong to the view, false otherwise. The database problem of interest can be reformulated as the following problem of discriminant induction.

Definition 2
Given:

- a DATALOG database Π and a $\mathcal{DL}$ ontology Σ integrated into a $\mathcal{DL}$+LOG$^{\neg}$ KB $\mathcal{B}$ (background theory);
- a DATALOG predicate p (target predicate);
- a set $\mathcal{O}$ of ground DATALOG facts that are either true or false for p (examples); and
- a set $\mathcal{L}$ of constraints on the form of $\mathcal{DL}$+LOG$^{\neg}$ definitions for p (language of hypotheses)

the problem of defining the view of name p is to induce a set $\mathcal{H} \subset \mathcal{L}$ (hypothesis) of $\mathcal{DL}$+LOG$^{\neg}$ rules from $\mathcal{O}$ and $\mathcal{B}$ such that $\mathcal{H}$ explains $\mathcal{O}$ by taking $\mathcal{B}$ into account.

We assume that the *background theory* $\mathcal{B}$ in Definition 2 is a $\mathcal{DL}$+LOG$^{\neg}$ KB which consists of an intensional part $\mathcal{K}$ (i.e., the TBox $\mathcal{T}$ plus the set Π_R of rules) and an extensional part $\mathcal{F}$ (i.e., the ABox $\mathcal{A}$ plus the set Π_F of facts). Also we denote by $P_{\mathcal{C}}(\mathcal{B})$, $P_{\mathcal{R}}(\mathcal{B})$, and $P_{\mathrm{D}}(\mathcal{B})$ the sets of concept, role and DATALOG predicate names occurring in $\mathcal{B}$, respectively. Note that $p \notin P_{\mathrm{D}}(\mathcal{B})$.

Example 3
Throughout this section we shall consider a database Π in the form of the following DATALOG$^{\neg}$ program:

```
famous(Mary)
famous(Paul)
famous(Joe)
scientist(Joe)
```

containing also the rule

[$R1$] `RICH(X)` $\leftarrow$ `famous(X), not scientist(X)`

linking the database to the ontology Σ expressed as the following $\mathcal{DL}$ KB:

[$A1$] `RICH`$\sqcap$`UNMARRIED` $\sqsubseteq \exists$ `WANTS-TO-MARRY`$^{-}$.$\top$
[$A2$] `WANTS-TO-MARRY`$\sqsubseteq$`LOVES`
`UNMARRIED(Mary)`
`UNMARRIED(Joe)`

Note that Π and Σ can be integrated into a $\mathcal{DL}$+LOG$^{\neg}$ KB $\mathcal{B}$ (adapted from Rosati 2006) that concerns the individuals `Mary`, `Joe`, and `Paul` and builds upon the alphabets $P_{\mathcal{C}}(\mathcal{B}) = \{\texttt{RICH/1}, \texttt{UNMARRIED/1}\}$, $P_{\mathcal{R}}(\mathcal{B}) = \{\texttt{WANTS-TO-MARRY/2}, \texttt{LOVES/2}\}$, and $P_{\mathrm{D}}(\mathcal{B}) = \{\texttt{famous/1}, \texttt{scientist/1}\}$.

The *language* $\mathcal{L}$ *of hypotheses* in Definition 2 must allow for the generation of $\mathcal{DL}$+LOG$^{\neg}$ rules starting from three disjoint alphabets $P_{\mathcal{C}}(\mathcal{L}) \subseteq P_{\mathcal{C}}(\mathcal{B})$, $P_{\mathcal{R}}(\mathcal{L}) \subseteq P_{\mathcal{R}}(\mathcal{B})$, and $P_{\mathrm{D}}(\mathcal{L}) \subseteq P_{\mathrm{D}}(\mathcal{B})$. Also we distinguish between $P^{+}_{\mathrm{D}}(\mathcal{L})$ and $P^{-}_{\mathrm{D}}(\mathcal{L})$ in order to specify which DATALOG predicates can occur in positive and negative literals, respectively. More precisely, we consider $\mathcal{DL}+\text{LOG}^{\neg}$ rules of the form

$$p(\vec{X}) \leftarrow r_1(\vec{Y}_1), \ldots, r_m(\vec{Y}_m), s_1(\vec{Z}_1), \ldots, s_k(\vec{Z}_k), not\ u_1(\vec{W}_1), \ldots, not\ u_h(\vec{W}_h)$$

where the unique literal $p(\vec{X})$ in the head is formed out of a DATALOG-predicate p which represents the target predicate. Note that the conditions of linkedness and connectedness usually assumed in ILP are guaranteed by the conditions of DATALOG safeness and weak $\mathcal{DL}$ safeness valid in $\mathcal{DL}+\text{LOG}^{\neg\vee}$.

Example 4
Suppose that the DATALOG-predicate happy is the target and the set $P_{\text{D}}^{+}(\mathcal{L}^{\mathtt{happy}}) \cup P_{\mathcal{C}}(\mathcal{L}^{\mathtt{happy}}) \cup P_{\mathcal{R}}(\mathcal{L}^{\mathtt{happy}}) = \{\mathtt{famous/1}, \mathtt{RICH/1}, \mathtt{LOVES/2}, \mathtt{WANTS\text{-}TO\text{-}MARRY/2}\}$ provides the building blocks for the language $\mathcal{L}^{\mathtt{happy}}$. The following $\mathcal{DL}+\text{LOG}^{\neg}$ rules

$R_1^{\mathtt{happy}}$	`happy(X) ← famous(X)`
$R_2^{\mathtt{happy}}$	`happy(X) ← famous(X), RICH(X)`
$R_3^{\mathtt{happy}}$	`happy(X) ← famous(X), LOVES(Y,X)`
$R_4^{\mathtt{happy}}$	`happy(X) ← famous(X), WANTS-TO-MARRY(Y,X)`

belonging to $\mathcal{L}^{\mathtt{happy}}$ can be considered definitions for the target predicate happy.

The set $\mathcal{O}$ of observations in Definition 2 contains facts of the kind $p(\vec{a}_i)$ where p is the target predicate and $\vec{a}_i$ is a tuple of individuals occurring in the ABox $\mathcal{A}$. We assume $\mathcal{B} \cap \mathcal{O} = \emptyset$. Furthermore, the description of each observation $o_i \in \mathcal{O}$ is in the background theory and may be incomplete due to the inherent nature of $\mathcal{DL}+\text{LOG}^{\neg}$. Therefore, the normal ILP setting is the most appropriate to the learning problem in hand and can be extended to $\mathcal{DL}+\text{LOG}^{\neg}$ as follows.

Definition 3
Let $R \in \mathcal{L}$ be a $\mathcal{DL}+\text{LOG}^{\neg}$ rule, $\mathcal{B}$ a $\mathcal{DL}+\text{LOG}^{\neg}$ KB, p the target predicate, and $o_i = p(\vec{a}_i) \in \mathcal{O}$ a ground DATALOG fact. We say that R *covers* o_i *under entailment w.r.t.* $\mathcal{B}$ iff $\mathcal{B} \cup R \models p(\vec{a}_i)$.

Note that the coverage test can be reduced to query answering in $\mathcal{DL}+\text{LOG}^{\neg}$ KBs which in turn can be reformulated as a satisfiability problem of the KB.

Example 5
The rule $R_4^{\mathtt{happy}}$ mentioned in Example 4 covers the observation $o_{\mathtt{Mary}} = \mathtt{happy(Mary)}$ because $\mathcal{B} \cup R_4^{\mathtt{happy}} \models \mathtt{happy(Mary)}$. Indeed, all NM-models for $\mathcal{B}' = \mathcal{B} \cup R_4^{\mathtt{happy}}$ satisfy:

- `famous(Mary)` is in $\mathcal{B}$;
- $\exists\,\mathtt{WANTS\text{-}TO\text{-}MARRY}^{-}.\top\mathtt{(Mary)}$, due to the axiom [$A1$] and to the fact that both `RICH(Mary)` and `UNMARRIED(Mary)` hold in every model of $\mathcal{B}'$. In particular, `RICH(Mary)` holds because of [$R1$];
- `happy(Mary)`, due to the above conclusions and to the rule $R_4^{\mathtt{happy}}$. Indeed, since $\exists\mathtt{WANTS\text{-}TO\text{-}MARRY}^{-}.\top\mathtt{(Mary)}$ holds in every model of $\mathcal{B}'$, it follows that in every model there exists a constant `x` such that `WANTS-TO-MARRY(x,Mary)` holds in the model, consequently from $R_4^{\mathtt{happy}}$ it follows that `happy(Mary)` also holds in the model.

Note that $R_4^{\texttt{happy}}$ does not cover the observations $o_{\texttt{Joe}} = \texttt{happy(Joe)}$ and $o_{\texttt{Paul}} = \texttt{happy(Paul)}$. More precisely, $\mathcal{B}' \not\models \texttt{happy(Joe)}$ because $\texttt{scientist(Joe)}$ holds in every model of $\mathcal{B}'$, thus making the rule $[R1]$ not applicable for $\{\texttt{X/Joe}\}$, therefore $\texttt{RICH(Joe)}$ not derivable. Finally, $\mathcal{B}' \not\models \texttt{happy(Paul)}$ because $\texttt{UNMARRIED(Paul)}$ is not forced to hold in every model of $\mathcal{B}'$, therefore $\exists\texttt{WANTS-TO-MARRY}^-.\top\texttt{(Paul)}$ is not forced by $[A1]$ to hold in every such model.

It can be proved that also $R_3^{\texttt{happy}}$ covers only $o_{\texttt{Mary}}$, while $R_1^{\texttt{happy}}$ covers all the three observations and $R_2^{\texttt{happy}}$ covers $o_{\texttt{Mary}}$ and $o_{\texttt{Paul}}$ only.

In order to support the induction of $\mathcal{DL}$+LOG$^{\neg}$ rules with ILP techniques, the language $\mathcal{L}$ of hypotheses needs to be equipped with a generality order $\succeq$ so that $(\mathcal{L}, \succeq)$ is a search space. Therefore, the next two subsections, Section 4.1 and Section 4.2, are devoted to suggested techniques for structuring and searching the hypothesis space, respectively. Conversely, Section 4.3 sketches an ILP algorithm employing these techniques to solve the original problem of inducing database views.

4.1 The hypothesis space

The definition of a generality order for hypotheses in $\mathcal{L}$ must consider the peculiarities of $\mathcal{DL}$+LOG$^{\neg}$. One issue arises from the presence of NAF literals (i.e., negated DATALOG literals) both in the background theory and in the language of hypotheses. As pointed out in Sakama (2001), rules in normal logic programs are syntactically regarded as Horn clauses by viewing the NAF-literal $\neg p(X)$ as an atom $not\_p(X)$ with the new predicate not_p. Then any result obtained in ILP on Horn logic programs is directly carried over to normal logic programs. Assuming one such treatment of NAF literals, we propose to adapt generalized subsumption (Buntine 1988) to the case of $\mathcal{DL}$+LOG$^{\neg}$ rules and provide a characterization of the resulting generality order, denoted by $\succeq_{\mathcal{K}}^{\neg}$, that relies on the reasoning tasks known for $\mathcal{DL}$+LOG$^{\neg\vee}$ and from which a test procedure can be derived.

Definition 4
Let $R_1, R_2 \in \mathcal{L}$ be two $\mathcal{DL}$+LOG$^{\neg}$ rules standardized apart, $\mathcal{K}$ a $\mathcal{DL}$+LOG$^{\neg}$ KB, and σ a Skolem substitution for R_2 with respect to $\{R_1\} \cup \mathcal{K}$. We say that R_1 is *more general than* R_2 w.r.t. $\mathcal{K}$, denoted by $R_1 \succeq_{\mathcal{K}}^{\neg} R_2$, iff there exists a ground substitution θ for R_1 such that (i) $head(R_1)\theta = head(R_2)\sigma$ and (ii) $\mathcal{K} \cup body(R_2)\sigma \models body(R_1)\theta$. We say that R_1 is *strictly more general than* R_2 w.r.t. $\mathcal{K}$, denoted by $R_1 \succ_{\mathcal{K}}^{\neg} R_2$, iff $R_1 \succeq_{\mathcal{K}}^{\neg} R_2$ and $R_2 \not\succeq_{\mathcal{K}}^{\neg} R_1$. We say that R_1 is *equivalent to* R_2 w.r.t. $\mathcal{K}$, denoted by $R_1 \equiv_{\mathcal{K}}^{\neg} R_2$, iff $R_1 \succeq_{\mathcal{K}}^{\neg} R_2$ and $R_2 \succeq_{\mathcal{K}}^{\neg} R_1$.

Note that condition (ii) is a variant of the Boolean CQ/UCQ containment problem because $body(R_2)\sigma$ and $body(R_1)\theta$ are both Boolean CQs. The difference between (ii) and the original formulation of the problem is that $\mathcal{K}$ encompasses not only a TBox but also a set of rules. Nonetheless this variant can be reduced to the satisfiability problem for finite $\mathcal{DL}$+LOG$^{\neg}$ KBs. Indeed the skolemization of $body(R_2)$ allows to reduce the Boolean CQ/UCQ containment problem to a CQ answering problem. Due to the aforementioned link between CQ answering and satisfiability, checking (ii) can be reformulated as proving that the KB $(\mathcal{T}, \Pi_R \cup body(R_2)\sigma \cup \{\leftarrow body(R_1)\theta\})$ is unsatisfiable. Once reformulated this way, (ii) can be solved by applying the algorithm NMSAT-$\mathcal{DL}$+LOG.

Example 6

Let us consider the hypotheses

R_1^{happy}	`happy(A)` $\leftarrow$ `famous(A)`
R_2^{happy}	`happy(X)` $\leftarrow$ `famous(X), RICH(X)`

reported in Example 4 up to variable renaming. We want to check whether

$$R_1^{\text{happy}} \succeq_{\mathcal{K}}^{\neg} R_2^{\text{happy}}$$

holds. Let $\sigma = \{\mathtt{X/a}\}$ be a Skolem substitution for R_2^{happy} with respect to $\mathcal{K} \cup R_1^{\text{happy}}$ and $\theta = \{\mathtt{A/a}\}$ a ground substitution for R_1^{happy}. Both conditions of Definition 4 are immediately verified. Thus, $R_1^{\text{happy}} \succeq_{\mathcal{K}}^{\neg} R_2^{\text{happy}}$. Since the viceversa does not hold, we can say that $R_1^{\text{happy}} \succ_{\mathcal{K}}^{\neg} R_2^{\text{happy}}$. Analogously, it can be proved that $R_1^{\text{happy}} \succ_{\mathcal{K}}^{\neg} R_3^{\text{happy}}$ and $R_1^{\text{happy}} \succ_{\mathcal{K}}^{\neg} R_4^{\text{happy}}$. Also, it turns out that R_2^{happy} is incomparable under $\succeq_{\mathcal{K}}^{\neg}$ with R_3^{happy} and R_4^{happy}. Finally, it can be proved that $R_3^{\text{happy}} \succ_{\mathcal{K}}^{\neg} R_4^{\text{happy}}$. In particular, the condition (ii) $\mathcal{K} \cup \{\mathtt{famous(a)}, \mathtt{WANTS\text{-}TO\text{-}MARRY(b,a)}\} \models \{\mathtt{famous(a)}, \mathtt{LOVES(b,a)}\}$ is nothing else that a ground query answering problem in $\mathcal{DL}$+LOG$^{\neg}$. The entailment is guaranteed by the axiom [$A2$].

It can be proved that $\succ_{\mathcal{K}}^{\neg}$ is a decidable quasi-order (i.e., it is a reflexive and transitive relation) for $\mathcal{DL}$+LOG$^{\neg}$ rules. In particular, the decidability of $\succ_{\mathcal{K}}^{\neg}$ follows from the decidability of $\mathcal{DL}$+LOG$^{\neg}$.

4.2 A refinement operator

As pointed out in Section 4.1, the space $(\mathcal{L}, \succ_{\mathcal{K}}^{\neg})$ is a quasi-ordered set, therefore it can be searched by refinement operators. In the following, we define a downward refinement operator for a $\mathcal{DL}$+LOG$^{\neg}$ language.

Definition 5

Let $\mathcal{L}$ be a $\mathcal{DL}$+LOG$^{\neg}$ language of hypotheses built out of the three finite and disjoint alphabets $P_{\mathcal{C}}(\mathcal{L})$, $P_{\mathcal{R}}(\mathcal{L})$, and $P_{\text{D}}^{+}(\mathcal{L}) \cup P_{\text{D}}^{-}(\mathcal{L})$, and

$$p(\vec{X}) \leftarrow r_1(\vec{Y}_1), \ldots, r_m(\vec{Y}_m), s_1(\vec{Z}_1), \ldots, s_k(\vec{Z}_k), not\ u_1(\vec{W}_1), \ldots, not\ u_h(\vec{W}_h)$$

be a rule R belonging to $\mathcal{L}$. We define a *downward refinement operator* $\rho^{\neg}$ for $(\mathcal{L}, \succeq_{\mathcal{K}})$ such that the set $\rho^{\neg}(R)$ contains all $R' \in \mathcal{L}$ that can be obtained from R by applying one of the following refinement rules:

$\langle AddDataLit\_B^{+}\rangle$ $body(R') = body(R) \cup \{r_{m+1}(\vec{Y}_{m+1})\}$ if

(1) $r_{m+1} \in P_{\text{D}}^{+}(\mathcal{L})$
(2) $r_{m+1}(\vec{Y}_{m+1}) \notin body(R)$

$\langle AddDataLit\_B^{-}\rangle$ $body(R') = body(R) \cup \{not\ u_{m+1}(\vec{W}_{h+1})\}$ if

(1) $u_{h+1} \in P_{\text{D}}^{-}(\mathcal{L})$

(2) $u_{h+1}(\vec{W_{h+1}}) \notin body(R)$

$\langle AddOntoLit\_B \rangle$ $body(R') = body(R) \cup \{s_{k+1}(\vec{Z_{k+1}})\}$ if

(2) $s_{k+1} \in P_{\mathcal{C}}(\mathcal{L}) \cup P_{\mathcal{R}}(\mathcal{L})$

(2) it does not exist any $s_l \in body(H)$ such that $s_{k+1} \sqsubseteq s_l$

$\langle SpecOntoLit\_B \rangle$ $body(R') = (body(R) \setminus \{s_l(\vec{Z}_l)\}) \cup s'_l(\vec{Z}_l)$ if

(1) $s'_l \in P_{\mathcal{C}}(\mathcal{L}) \cup P_{\mathcal{R}}(\mathcal{L})$

(2) $s'_l \sqsubseteq s_l$

All the rules of $\rho^{\neg}$ are correct, i.e., the R's obtained by applying any of the rules of $\rho^{\neg}$ to $R \in \mathcal{L}$ are such that $R \succ^{\neg}_{\mathcal{K}} R'$. This can be proved intuitively by observing that they act only on $body(R)$. Thus condition (i) of Definition 4 is satisfied. Furthermore, it is straightforward to notice that the application of any of the rules of $\rho^{\neg}$ to R reduces the number of models of R. In particular, as for $\langle SpecOntoLit \rangle$, this intuition follows from the semantics of DLs. So condition (ii) also is fulfilled.

Example 7

With reference to Example 4, applying the refinement rule $\langle AddDataLit\_B^{+} \rangle$ to

R_0^{happy} `happy(X)` $\leftarrow$

produces R_1^{happy} which can be further specialized into R_2^{happy}, R_3^{happy}, and R_4^{happy} by means of $\langle AddOntoLit\_B \rangle$. Note that no other refinement rule can be applied to R_1^{happy} and that R_4^{happy} can be also obtained as refinement via $\langle SpecOntoLit\_B \rangle$ from R_3^{happy}.

Ideal refinement operators have been proven not to exist for clausal languages ordered by θ-subsumption or stronger orders but can be approximated by dropping the requirement of properness or by bounding the language (Nienhuys-Cheng and de Wolf 1997). We choose the latter option because it guarantees that, if $(\mathcal{L}, \succeq)$ is a quasi-ordered set, $\mathcal{L}$ is finite and $\succeq$ is decidable, then there always exists an ideal refinement operator for $(\mathcal{L}, \succeq)$. In our case, since $\succ^{\neg}_{\mathcal{K}}$ is a decidable quasi-order for any $\mathcal{DL}$ with decidable Boolean CQ/UCQ containment problem, we only need to bound $\mathcal{L}$ in a suitable manner. From Definition 5 we know that the alphabets $P_{\mathcal{C}}(\mathcal{L})$, $P_{\mathcal{R}}(\mathcal{L})$, and $P^{+}_{\mathrm{D}}(\mathcal{L}) \cup P^{-}_{\mathrm{D}}(\mathcal{L})$ are finite. Having DATALOG as basis for the CL part of $\mathcal{DL}$+LOG$^{\neg}$ avoids the generation of infinite terms. Yet, the expressive power of $\mathcal{DL}$+LOG$^{\neg}$ requires several other bounds to be imposed on $\mathcal{L}$ in order to guarantee its finiteness. It is necessary to introduce a complexity measure for $\mathcal{DL}$+LOG$^{\neg}$ rules, as a pair of two different coordinates. Considering that the complexity of a $\mathcal{DL}$+LOG$^{\neg}$ rule resides in its body, the former coordinate is the size (i.e., the difference between the number of symbol occurrences and the number of distinct variables) of the biggest literal in $body(R)$, while the latter is the number of literals in $body(R)$. To keep $\mathcal{L}$ finite, we need first to set a maximum value for these two coordinates. Second, it is necessary to set the maximum number of specialization/generalization steps of the DL literals so that the search in the ontology is also depth-bounded.

```
NMLEARN-$\mathcal{DL}$+LOG$^\neg$($\mathcal{L}$, $\mathcal{B}$, $\mathcal{O}$, $p$)
1. $\mathcal{H} \leftarrow \emptyset$
2. $E^+ \leftarrow \{o_i \in \mathcal{O} | o_i \textit{ is true for } p\}$;
3. $E^- \leftarrow \{o_i \in \mathcal{O} | o_i \textit{ is false for } p\}$;
4. while $E^+ \neq \emptyset$ do
5.     $R \leftarrow \{p(\vec{X}) \leftarrow\}$;
6.     $E^-_R \leftarrow E^-$
7.     while $E^-_R \neq \emptyset$ do
8.         $\mathcal{Q} \leftarrow \{R' \in \mathcal{L} | R' \in \rho^\neg(R)\}$;
9.         $R \leftarrow best\_of(\mathcal{Q})$;
10.        $E^-_R \leftarrow E^-_R \setminus \{e \in E^-_R | \mathcal{B} \cup R \models e\}$;
11.    endwhile
12.    $\mathcal{H} \leftarrow \mathcal{H} \cup \{R\}$;
13.    $E^+ \leftarrow E^+ \setminus \{e \in E^+ | \mathcal{B} \cup R \models e\}$;
14. endwhile
return $\mathcal{H}$
```

Fig. 2. Main procedure of NMLEARN-$\mathcal{DL}$+LOG$^\neg$.

4.3 An algorithm

The algorithm in Figure 2 defines the main procedure of NMLEARN-$\mathcal{DL}$+LOG$^\neg$. Notice that the outer loop (4–14) corresponds to a variant of the sequential covering algorithm, i.e., it learns new rules one at a time, removing the positive examples covered by the latest rule before attempting to learn the next rule (13). The hypothesis space search performed by NMLEARN-$\mathcal{DL}$+LOG$^\neg$ is best understood by viewing it hierarchically. Each iteration through the outer loop (4–14) adds a new rule to its disjunctive hypothesis $\mathcal{H}$. The effect of each new rule is to generate the current disjunctive hypothesis (i.e., to increase the number of instances it classifies as positive), by adding a new disjunct. Viewed at this level, the search is a bottom-up search through the space of hypotheses, beginning with the most specific empty disjunction (1) and terminating when the hypothesis is sufficiently general to cover all positive training examples (14). The inner loop (7–11) performs a finer-grained search to determine the exact definition of each new rule. This loop searches a second hypothesis space, consisting of conjunctions of literals, to find a conjunction that will form the preconditions for the new rule. Within this space, it conducts a top-down, hill-climbing search, beginning with the most general preconditions possible (5), then adding literals one at a time to specialize the rule (7) until it avoids all negative examples. To select the most promising specialization from the candidates generated at each step (9), NMLEARN-$\mathcal{DL}$+LOG$^\neg$ considers the performance of the rule over the training examples, i.e., it maximizes the number of positive examples covered while keeping the number of negative examples covered as low as possible.

Example 8

With reference to Example 7 and Example 5, we suppose that

$$E^+ = \{o_{\texttt{Mary}}, o_{\texttt{Joe}}\}$$
$$E^- = \{o_{\texttt{Paul}}\}$$

The outer loop of the algorithm NMLEARN-$\mathcal{DL}+\text{LOG}^{\neg}$ starts from

R_0^{happy} `happy(X)` $\leftarrow$

which is further refined through the iterations of the inner loop, more precisely it is first specialized into

R_1^{happy} `happy(X)` $\leftarrow$ `famous(X)`

which in turn, since it covers negative examples, is then specialized into

R_2^{happy} `happy(X)` $\leftarrow$ `famous(X), RICH(X)`
R_3^{happy} `happy(X)` $\leftarrow$ `famous(X), LOVES(Y,X)`
R_4^{happy} `happy(X)` $\leftarrow$ `famous(X), WANTS-TO-MARRY(Y,X)`

out of which the rule R_3^{happy} is selected as the best and added to the hypothesis because it does not cover negative examples. Note that R_3^{happy} is preferred to R_4^{happy} because it is more general.

5 Inducing database constraints in $\mathcal{DL}+\text{LOG}^{\neg\vee}$ with ILP

In this section we face the problem of inducing an integrity theory $\mathcal{H}$ for a database Π whose instance Π_F is given and whose schema $\mathcal{K}$ encompasses an ontology Σ and a set Π_R of rules linking the database to the ontology. We assume that Π and Σ shares a common set of constants so that they can constitute a $\mathcal{DL}+\text{LOG}^{\neg\vee}$ KB $\mathcal{B}$.

Definition 6
Given:

- an intensional DATALOG database Π_R and a $\mathcal{DL}$ ontology Σ integrated into a $\mathcal{DL}+\text{LOG}^{\neg\vee}$ KB $\mathcal{K}$ (background theory);
- a set $\mathcal{O} = \Pi_F$ of ground DATALOG facts (observation); and
- a set $\mathcal{L}$ of constraints on the form of $\mathcal{DL}+\text{LOG}^{\neg\vee}$ rules to be induced (language of hypotheses)

the problem of defining an integrity theory for Π_F is to induce a set $\mathcal{H} \subset \mathcal{L}$ (hypothesis) of $\mathcal{DL}+\text{LOG}^{\neg\vee}$ rules from $\mathcal{O}$ and $\mathcal{K}$ such that $\mathcal{H}$ confirms $\mathcal{O}$ by taking $\mathcal{K}$ into account.

Note that, as opposite to the learning problem formally stated in Definition 2, the *background theory* in Definition 6 is a $\mathcal{DL}+\text{LOG}^{\neg\vee}$ KB $\mathcal{K}$ which does not include the extensional part Π_F of the database. Indeed Π_F plays the role of the unique *observation* from which the learning process should induce a theory $\mathcal{H}$. Conversely, similarly to Section 4, we denote by $P_{\mathcal{C}}(\mathcal{B})$, $P_{\mathcal{R}}(\mathcal{B})$, and $P_{\mathrm{D}}(\mathcal{B})$ the sets of concept, role and DATALOG predicate names occurring in $\mathcal{B}$, respectively, assuming that $\mathcal{B} = \Sigma \cup \Pi$.

Example 9
Throughout this section we shall refer to a database about students in the form of a DATALOG$^{\neg\vee}$ program Π which consists of an extensional part Π_F with the following facts:

```
boy(Paul)
girl(Mary)
enrolled(Paul,c1)
enrolled(Mary,c1)
enrolled(Mary,c2)
enrolled(Bob,c3)
```

and an intensional part Π_R with the following rules:

[R1] $\texttt{FEMALE(X)} \leftarrow \texttt{girl(X)}$
[R2] $\texttt{MALE(X)} \leftarrow \texttt{boy(X)}$

linking the database to an ontology about persons expressed as the following $\mathcal{DL}$ KB Σ:

[A1] $\texttt{PERSON} \sqsubseteq \exists\, \texttt{FATHER}^{-}.\texttt{MALE}$
[A2] $\texttt{MALE} \sqsubseteq \texttt{PERSON}$
[A3] $\texttt{FEMALE} \sqsubseteq \texttt{PERSON}$
[A4] $\texttt{FEMALE} \sqsubseteq \neg\texttt{MALE}$

```
MALE(Bob)
PERSON(Mary)
PERSON(Paul)
```

Note that Π and Σ can be integrated into a $\mathcal{DL}$+LOG$^{\neg\vee}$ KB $\mathcal{B}$ (adapted from Rosati 2006) that concerns the individuals Bob, Mary, and Paul and builds upon the alphabets $P_{\mathcal{C}}(\mathcal{B}) = \{\texttt{FEMALE/1}, \texttt{MALE/1}, \texttt{PERSON/1}\}$, $P_{\mathcal{R}}(\mathcal{B}) = \{\texttt{FATHER/2}\}$, and $P_{\mathrm{D}}(\mathcal{B}) = \{\texttt{boy/1}, \texttt{girl/1}, \texttt{enrolled/2}\}$.

The *language $\mathcal{L}$ of hypotheses* in Definition 6 must allow for the generation of $\mathcal{DL}$+LOG$^{\neg\vee}$ rules starting from three disjoint alphabets $P_{\mathcal{C}}(\mathcal{L}) \subseteq P_{\mathcal{C}}(\mathcal{B})$, $P_{\mathcal{R}}(\mathcal{L}) \subseteq P_{\mathcal{R}}(\mathcal{B})$, and $P_{\mathrm{D}}(\mathcal{L}) \subseteq P_{\mathrm{D}}(\mathcal{B})$. Analogously to Section 4, we distinguish between $P_{\mathrm{D}}^{+}(\mathcal{L})$ and $P_{\mathrm{D}}^{-}(\mathcal{L})$ in order to specify which DATALOG predicates can occur in positive and negative literals, respectively.

Example 10
The following $\mathcal{DL}$+LOG$^{\neg\vee}$ rules:

$\texttt{PERSON(X)} \leftarrow \texttt{enrolled(X,c1)}$
$\texttt{boy(X)} \vee \texttt{girl(X)} \leftarrow \texttt{enrolled(X,c1)}$
$\leftarrow \texttt{enrolled(X,c2)}, \texttt{MALE(X)}$
$\leftarrow \texttt{enrolled(X,c2)}, \texttt{not girl(X)}$
$\texttt{MALE(X)} \leftarrow \texttt{enrolled(X,c3)}$

belong to the language $\mathcal{L}$ built upon the alphabets $P_{\mathcal{C}}(\mathcal{L}) = P_{\mathcal{C}}(\mathcal{B})$, $P_{\mathcal{R}}(\mathcal{L}) = \emptyset$, $P_{\mathrm{D}}^{+}(\mathcal{L}) = \{\texttt{boy/1}, \texttt{girl/1}, \texttt{enrolled(\_,c1)}, \texttt{enrolled(\_,c2)}, \texttt{enrolled(\_,c3)}\}$, and $P_{\mathrm{D}}^{-}(\mathcal{L}) = \{\texttt{boy/1}, \texttt{girl/1}\}$.

The scope of induction in the learning problem of interest is characterization because we are looking for a theory which confirms the observation. Also, since a $\mathcal{DL}$+LOG$^{\neg\vee}$ KB may be incomplete due to the inherent nature of this KR framework, the most appropriate setting for induction is the one for learning from entailment. The coverage test proposed in the following generalizes the case illustrated in Definition 3 to observations which are not singletons of facts.

Definition 7
Let $R \in \mathcal{L}$ be a $\mathcal{DL}$+LOG$^{\neg\vee}$ rule, $\mathcal{K}$ a $\mathcal{DL}$+LOG$^{\neg\vee}$ KB, and $\mathcal{O} = \{p_i(\vec{a}_i)\}$ a set of ground DATALOG facts. We say that R *covers* $\mathcal{O}$ *under entailment w.r.t.* $\mathcal{K}$ iff $\mathcal{K} \cup R \models \bigwedge p_i(\vec{a}_i)$.

It is immediate to notice that the coverage test of Definition 7 can be reduced to Boolean CQ answering in $\mathcal{DL}$+LOG$^{\neg\vee}$ KBs and therefore to a NM-satisfiability problem.

In the following we sketch the ingredients for an ILP system able to discover such integrity theories on the basis of NMSAT-$\mathcal{DL}$+LOG.

5.1 The hypothesis space

The order of relative subsumption (Plotkin 1971) is suitable for extension to $\mathcal{DL}$+LOG$^{\neg\vee}$ rules because it can cope with arbitrary clauses and admit an arbitrary finite set of clauses as the background theory.

Definition 8
Let $R_1, R_2 \in \mathcal{L}$ be two $\mathcal{DL}$+LOG$^{\neg\vee}$ rules, and $\mathcal{K}$ a $\mathcal{DL}$+LOG$^{\neg\vee}$ KB. We say that R_1 *is more general than* R_2 w.r.t. $\mathcal{K}$, denoted by $R_1 \succeq_{\mathcal{K}}^{\neg\vee} R_2$, if there exists a substitution θ such that $\mathcal{K} \models \forall(R_1\theta \implies R_2)$. We say that R_1 is *strictly more general than* R_2 w.r.t. $\mathcal{K}$, denoted by $R_1 \succ_{\mathcal{K}}^{\neg\vee} R_2$, iff $R_1 \succeq_{\mathcal{K}}^{\neg\vee} R_2$ and $R_2 \not\succeq_{\mathcal{K}}^{\neg\vee} R_1$. We say that R_1 is *equivalent to* R_2 w.r.t. $\mathcal{K}$, denoted by $R_1 \equiv_{\mathcal{K}}^{\neg\vee} R_2$, iff $R_1 \succeq_{\mathcal{K}}^{\neg\vee} R_2$ and $R_2 \succeq_{\mathcal{K}}^{\neg\vee} R_1$.

Example 11
Let us consider the following $\mathcal{DL}$+LOG$^{\neg\vee}$ rules belonging to the language $\mathcal{L}$ specified in Example 10:

R_1 `boy(X)` $\leftarrow$ `enrolled(X,c1)`
R_2 `boy(A)` $\vee$ `girl(A)` $\leftarrow$ `enrolled(A,c1)`

It can be easily proved that $R_1 \succeq_{\mathcal{K}}^{\neg\vee} R_2$. Let $\theta = \{\texttt{X/A}\}$ be the substitution to be applied to R_1 and let us suppose that, for every `A`, if `A` is enrolled in the course `c1`, then `A` is a `boy` (i.e., the rule $R_1\theta$ is true), thus we can also say that `A` is either a `boy` or a `girl` (i.e., the rule R_2 is true). Note that $R_2 \not\succeq_{\mathcal{K}}^{\neg\vee} R_1$.

Let us now consider the following $\mathcal{DL}$+LOG$^{\neg\vee}$ rules also belonging to $\mathcal{L}$:

R_3 `MALE(X)` $\leftarrow$ `enrolled(X,c1)`
R_4 `PERSON(A)` $\leftarrow$ `enrolled(A,c1)`

In order to prove that $R_3 \succeq_{\mathcal{K}}^{\neg\vee} R_4$, we apply $\theta = \{\texttt{X/A}\}$ to R_3 and suppose that, for every `A`, if `A` is enrolled in the course `c1`, then `A` is a `MALE` (i.e., the rule $R_1\theta$ is true). Due to axiom [$A2$] occurring in the ontology Σ reported in Example 9, `A` is a `PERSON` (i.e., the rule R_4 is true). It is immediate to verify that $R_3 \succ_{\mathcal{K}}^{\neg\vee} R_4$.

The generality relation defined by $\succeq_{\mathcal{K}}^{\neg\vee}$ is a quasi-order on $\mathcal{DL}$+LOG$^{\neg\vee}$ rules, therefore the resulting space $(\mathcal{L}, \succeq_{\mathcal{K}}^{\neg\vee})$ can be searched by means of refinement operators.

5.2 The refinement operator

A refinement operator for $(\mathcal{L}, \succeq_{\mathcal{K}}^{\neg\vee})$ should generate $\mathcal{DL}$+LOG$^{\neg\vee}$ rules good at expressing integrity constraints. Since we assume the database Π and the ontology Σ to be correct, a rule R must be modified to make it satisfiable by $\Pi \cup \Sigma$ by either (i) strengthening $body(R)$ or (ii) weakening $head(R)$.

Definition 9
Let $\mathcal{L}$ be a $\mathcal{DL}$+LOG$^{\neg\vee}$ language of hypotheses built out of the three finite and disjoint alphabets $P_{\mathcal{C}}(\mathcal{L})$, $P_{\mathcal{R}}(\mathcal{L})$, and $P_{\mathrm{D}}^{+}(\mathcal{L}) \cup P_{\mathrm{D}}^{-}(\mathcal{L})$, and

$$p_1(\vec{X}_1) \vee \ldots \vee p_n(\vec{X}_n) \leftarrow$$
$$r_1(\vec{Y}_1), \ldots, r_m(\vec{Y}_m), s_1(\vec{Z}_1), \ldots, s_k(\vec{Z}_k), not\ u_1(\vec{W}_1), \ldots, not\ u_h(\vec{W}_h)$$

be a rule R belonging to $\mathcal{L}$. We define a *downward refinement operator* $\rho^{\neg\vee}$ for $(\mathcal{L}, \succeq_{\mathcal{K}})$ such that the set $\rho^{\neg\vee}(R)$ contains all $R' \in \mathcal{L}$ that can be obtained from R by applying one of the following refinement rules:

$\langle AddDataLit\_B^{+} \rangle$ $body(R') = body(R) \cup \{r_{m+1}(\vec{Y}_{m+1})\}$ if

(1) $r_{m+1} \in P_{\mathrm{D}}^{+}(\mathcal{L})$
(2) $r_{m+1}(\vec{Y}_{m+1}) \notin body(R)$

$\langle AddDataLit\_B^{-} \rangle$ $body(R') = body(R) \cup \{not\ u_{m+1}(\vec{W}_{h+1})\}$ if

(1) $u_{h+1} \in P_{\mathrm{D}}^{-}(\mathcal{L})$
(2) $u_{h+1}(\vec{W}_{h+1}) \notin body(R)$

$\langle AddOntoLit\_B \rangle$ $body(R') = body(R) \cup \{s_{k+1}(\vec{Z}_{k+1})\}$ if

(1) $s_{k+1} \in P_{\mathcal{C}}(\mathcal{L}) \cup P_{\mathcal{R}}(\mathcal{L})$
(2) it does not exist any $s_l \in body(H)$ such that $s_{k+1} \sqsubseteq s_l$

$\langle SpecOntoLit\_B \rangle$ $body(R') = (body(R) \setminus \{s_l(\vec{Z}_l)\}) \cup s'_l(\vec{Z}_l)$ if

(1) $s'_l \in P_{\mathcal{C}}(\mathcal{L}) \cup P_{\mathcal{R}}(\mathcal{L})$
(2) $s'_l \sqsubseteq s_l$

$\langle AddDataLit\_H \rangle$ $head(R') = head(R) \cup \{p_{n+1}(\vec{X}_{n+1})\}$ if

(1) $p_{n+1} \in P_{\mathrm{D}}^{+}(\mathcal{L})$
(2) $p_{n+1}(\vec{X}_{n+1}) \notin head(R)$

$\langle AddOntoLit\_H \rangle$ $head(R') = head(R) \cup \{p_{n+1}(\vec{X}_{n+1})\}$ if

(1) $p_{n+1} \in P_{\mathcal{C}}(\mathcal{L}) \cup P_{\mathcal{R}}(\mathcal{L})$
(2) it does not exist any $p_i \in head(R)$ such that $p_{n+1} \sqsubseteq p_i$

$\langle GenOntoLit\_H\rangle$ $head(R') = (head(R) \setminus \{p_i(\vec{X}_i)\}) \cup p'_i(\vec{X}_i)$ if

(1) $p'_i \in P_{\mathcal{C}}(\mathcal{L}) \cup P_{\mathcal{R}}(\mathcal{L})$
(2) $p_i \sqsubseteq p'_i$

Note that, since we are working under NM-semantics, two distinct rules, namely $\langle AddDataLit\_B^-\rangle$ and $\langle AddDataLit\_H\rangle$, are devised for adding negated DATALOG atoms to the body and for adding DATALOG atoms to the head, respectively. It can be proved that all the rules of $\rho^{\neg\vee}$ are correct, i.e. the R''s obtained by applying any of the rules of $\rho^{\neg\vee}$ to $R \in \mathcal{L}$ are such that $R \succ^{\neg\vee}_{\mathcal{K}} R'$. Intuitively, it is sufficient to observe that the application of any of the rules of $\rho^{\neg\vee}$ conceived to strengthen $body(R)$ reduces the number of models of R whereas the rules aiming at weakening $head(R)$, when applied, do not augment the number of models of R.

Example 12
From the rule belonging to the language $\mathcal{L}$ specified in Example 10:

```
← enrolled(X,c1)
```

we obtain the following rules by applying $\langle AddDataLit\_B^+\rangle$:

```
← enrolled(X,c1), boy(X)
← enrolled(X,c1), girl(X)
← enrolled(X,c1), enrolled(X,c2)
← enrolled(X,c1), enrolled(X,c3)
```

the following ones by applying $\langle AddDataLit\_B^-\rangle$:

```
← enrolled(X,c1), not boy(X)
← enrolled(X,c1), not girl(X)
```

the following ones by applying $\langle AddOntoLit\_B\rangle$:

```
← enrolled(X,c1), PERSON(X)
← enrolled(X,c1), FEMALE(X)
← enrolled(X,c1), MALE(X)
```

the following ones by applying $\langle AddDataLit\_H\rangle$:

```
boy(X) ← enrolled(X,c1)
girl(X) ← enrolled(X,c1)
enrolled(X,c2) ← enrolled(X,c1)
enrolled(X,c3) ← enrolled(X,c1)
```

and the following ones:

```
PERSON(X) ← enrolled(X,c1)
FEMALE(X) ← enrolled(X,c1)
MALE(X) ← enrolled(X,c1)
```

by applying $\langle AddOntoLit\_H\rangle$.

```
NMDISC-$\mathcal{DL}$+LOG$^{\neg\vee}$($\mathcal{L}$, $\mathcal{K}$, $\Pi_F$)
1. $\mathcal{H} \leftarrow \emptyset$
2. $\mathcal{Q} \leftarrow \{\Box\}$
3. while $\mathcal{Q} \neq \emptyset$ do
4.     $\mathcal{Q} \leftarrow \mathcal{Q} \setminus \{R\}$;
5.     if NMSAT-$\mathcal{DL}$+LOG($\mathcal{K} \cup \Pi_F \cup \mathcal{H} \cup \{R\}$)
6.         then $\mathcal{H} \leftarrow \mathcal{H} \cup \{R\}$
7.         else $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{R' \in \mathcal{L} | R' \in \rho^{\neg\vee}(R)\}$
8.     endif
9. endwhile
return $\mathcal{H}$
```

Fig. 3. Main procedure of NMDISC-$\mathcal{DL}$+LOG$^{\neg\vee}$.

5.3 The algorithm

The integrity theory $\mathcal{H}$ we would like to discover is a set of $\mathcal{DL}$+LOG$^{\neg\vee}$ rules. It must be induced by taking the background theory $\mathcal{K} = \Sigma \cup \Pi_R$ into account so that $\mathcal{B} = (\Sigma, \Pi \cup \mathcal{H})$ is a NM-satisfiable $\mathcal{DL}$+LOG$^{\neg\vee}$ KB. The algorithm in Figure 3 defines the main procedure of NMDISC-$\mathcal{DL}$+LOG$^{\neg\vee}$: it starts from an empty theory $\mathcal{H}$ (1), and a queue $\mathcal{Q}$ containing only the empty clause (2). It then applies a search process (3) where each element R is deleted from the queue $\mathcal{Q}$ (4), and tested for satisfaction w.r.t. the data Π_F by taking into account the background theory $\mathcal{K}$ and the current integrity theory $\mathcal{H}$ (5). Note that the NM-satisfiability test includes also the current induced theory in order to deal with the nonmonotonicity of induction in the normal ILP setting. If the rule R is satisfied by the database (6), it is added to the theory (7). If the rule is violated by the database, its refinements according to $\mathcal{L}$ are considered (8). The search process terminates when $\mathcal{Q}$ becomes empty (9). Note that the algorithm does not specify the search strategy. In order to get a minimal theory (i.e., without redundant clauses), a pruning step and a post-processing phase can be added to NMDISC-$\mathcal{DL}$+LOG$^{\neg\vee}$ by further calling NMSAT-$\mathcal{DL}$+LOG[8].

Example 13
With reference to Example 12, the following $\mathcal{DL}$+LOG$^{\neg\vee}$ rule:

    PERSON(X) ← enrolled(X,c1)

is the only one passing the NM-satisfiability test at step (5) of the algorithm NMDISC-$\mathcal{DL}$+LOG$^{\neg\vee}$. It is added to the integrity theory. All the other rules are further refined. When the learning process ends at step (9) because the queue of rules has become empty, the integrity theory will encompass the rules reported in Example 10 because they are satisfied by the database.

6 Related work

Very few ILP frameworks have been proposed so far that adopt a hybrid DL-CL representation for both hypotheses and background knowledge (Rouveirol and

[8] Based on the following consequence of the Deduction Theorem in FOL: Given a KB $\mathcal{B}$ and a rule R in $\mathcal{DL}$+LOG$^{\neg\vee}$, we have that $\mathcal{B} \models R$ iff $\mathcal{B} \wedge \neg R$ is unsatisfiable.

Ventos 2000; Kietz 2003; Lisi 2008; Lisi and Esposito 2008). They are less or differently expressive than the one presented in this paper.

The framework proposed in Rouveirol and Ventos (2000) focuses on discriminant induction and adopts the ILP setting of learning from interpretations. Hypotheses are represented as CARIN-$\mathcal{ALN}$ nonrecursive rules with a Horn literal in the head that plays the role of target concept. The coverage relation of hypotheses against examples adapts the usual one in learning from interpretations to the case of hybrid CARIN-$\mathcal{ALN}$ BK. The generality relation for hypotheses is defined as an extension of generalized subsumption. Procedures for testing both the coverage relation and the generality relation are based on the existential entailment algorithm of CARIN. Following Rouveirol and Ventos (2000), Kietz studies the learnability of CARIN-$\mathcal{ALN}$, thus providing a preprocessing method which enables ILP systems to learn CARIN-$\mathcal{ALN}$ rules (2003).

In Lisi (2008), the representation and reasoning means come from $\mathcal{AL}$-log. Hypotheses are represented as constrained DATALOG clauses. Note that this framework is general, meaning that it is valid whatever the scope of induction is. The generality relation for one such hypothesis language is an adaptation of generalized subsumption to the $\mathcal{AL}$-log KR framework. It gives raise to a quasi-order and can be checked with a decidable procedure based on constrained SLD-resolution. Coverage relations for both ILP settings of learning from interpretations and learning from entailment have been defined on the basis of query answering in $\mathcal{AL}$-log. As opposite to Rouveirol and Ventos (2000), the framework has been partially implemented in an ILP system (Lisi and Malerba 2004) that supports a variant of frequent pattern discovery where rich prior conceptual knowledge is taken into account in order to find patterns at multiple levels of description granularity.

The framework presented in Lisi and Esposito (2008) is the closest to the present work. Indeed it faces the problem of learning in $\mathcal{DL}$+LOG, i.e., by disregarding the NM features of $\mathcal{DL}$+LOG$^{\neg\vee}$. Yet the framework is more general than the one illustrated here because two cases of rule learning are considered, one aimed at inducing rules with one DATALOG literal in the head and the other rules with one $\mathcal{DL}$ literal in the head. The former kind of rule will enrich the DATALOG part of the KB, whereas the latter will extend the $\mathcal{DL}$ part.

The main procedure of NMLEARN-$\mathcal{DL}$+LOG$^{\neg}$ follows the principles of FOIL (Quinlan 1990) but shows some peculiarities due to the nature of the underlying KR framework, e.g., the setting of learning from entailment (which is more powerful than the use of extensional background theory and coverage testing), and the ordering of generalized subsumption (instead of θ-subsumption).

The main procedure of NMDISC-$\mathcal{DL}$+LOG$^{\neg\vee}$ is inspired by CLAUDIEN (De Raedt and Bruynooghe 1993) as for the scope of induction and the algorithm scheme but differs from it in several points, notably the adoption of (i) relative subsumption instead of θ-subsumption, (ii) stable model semantics instead of completion semantics, and (iii) learning from entailment instead of learning from interpretations, to deal properly with the chosen representation formalism for both the background theory and the language of hypotheses.

ILP has been also applied to data engineering tasks such as the interactive restructuring of databases giving rise to the so-called Inductive Data Engineering

(IDE) (Flach 1993; Flach 1998; Savnik and Flach 2000). The main idea is to use induction to determine integrity constraints, such as functional and multivalued dependencies, that are valid (or almost valid) in a database and then use the constraints to decompose (restructure) the database.

7 Conclusions and future work

In this paper, we have investigated two ILP solutions for learning in the KR framework of $\mathcal{DL}$+LOG$^{\neg\vee}$, both valid for any $\mathcal{DL}$ for which the instantiation of the framework is decidable, but one restricted to DATALOG$^{\neg}$ and the other for the full framework. Indeed, well-known ILP techniques for induction such as the orderings of generalized subsumption and relative subsumption have been reformulated in terms of the deductive reasoning mechanisms of $\mathcal{DL}$+LOG$^{\neg\vee}$, namely by relying on the algorithm NMSAT-$\mathcal{DL}$+log devised to prove NM-satisfiability of $\mathcal{DL}$+LOG$^{\neg\vee}$ KBs. Notably, we have defined generality orders, refinement operators and coverage tests on the basis of NMSAT-$\mathcal{DL}$+log. Though the work presented in this paper is not yet supported by empirical evidence, it shows that it is feasible for ILP to go beyond DATALOG towards $\mathcal{DL}$+LOG$^{\neg\vee}$. The potential of this extended ILP has been illustrated in two traditional database problems, i.e., the definition of views and the definition of integrity theories, for which we have sketched ad-hoc ILP algorithms, NMLEARN-$\mathcal{DL}$+LOG$^{\neg}$ and NMDISC-$\mathcal{DL}$+LOG$^{\neg\vee}$, respectively. The NM features as well as the DL component of $\mathcal{DL}$+LOG$^{\neg\vee}$ enable these algorithms to build hypotheses with expressiveness far greater than the one reachable with the predecessors FOIL and CLAUDIEN. Notably, ontologies accommodate elegantly in the solution to the database problems being considered. From the ILP viewpoint the expressive power of $\mathcal{DL}$+LOG$^{\neg\vee}$ has, of course, raised some technical difficulties. In particular, the critical point has been the DL component that has required an appropriate treatment when defining both the generality orders and the refinement operators. Also the setting of learning from entailment turned out to be the most appropriate for the induction within the $\mathcal{DL}$+LOG$^{\neg\vee}$ KR framework.

As next step towards any practice, we plan to first analyze the complexity and then produce an efficient and scalable implementation of these ILP algorithms. Adopting less expressive but tractable instantiations of $\mathcal{DL}$+LOG$^{\neg\vee}$ may turn out crucial from this point of view. For example, DL-Lite (Calvanese *et al.* 2007) has been proved to be good at making $\mathcal{DL}$+LOG$^{\neg\vee}$ practically useful (Rosati 2006). Another point is the definition of so-called optimal refinement operators to be actually employed in NMLEARN-$\mathcal{DL}$+LOG$^{\neg}$ and NMDISC-$\mathcal{DL}$+LOG$^{\neg\vee}$. Indeed, ideal refinement operators are mainly of theoretical interest, because in practice they are often very inefficient. More constructive—though possibly improper—refinement operators are usually to be preferred over ideal ones. Optimal refinement operators can be easily derived from those proposed in this paper.

Learning in $\mathcal{DL}$+LOG$^{\neg\vee}$ is also promising for Semantic Web applications for the following reasons. First, it can deal with ontologies almost as expressive as the ones that OWL allow. Indeed, as already mentioned, $\mathcal{SHIQ}$ has been the starting point for the definition of OWL and gives rise to one of the currently most expressive decidable instantiations of $\mathcal{DL}$+LOG$^{\neg\vee}$. Second, it can deal with incomplete knowledge thanks to the NM features of $\mathcal{DL}$+LOG$^{\neg\vee}$. Third, it can deal with ontologies and rules tightly integrated as devised by the W3C Rule Interchange Format (RIF) working group.[9] Indeed the activity of the RIF group concerns (i) the definition of a core language with extensions some of which (the nonmonotonic ones) will most likely be inspired by hybrid DL–CL languages like $\mathcal{DL}$+LOG$^{\neg\vee}$ and (ii) the identification of use cases many of which are suitable to our algorithms for application.

As a final remark, we would like to point out that the shift from DATALOG to $\mathcal{DL}$+LOG$^{\neg\vee}$ in ILP paves the way to an extension of Relational Learning (and Data Mining), named Onto-Relational Learning, which accounts for ontologies in a clear, well-founded, and systematic way. Following the work reported in this paper, we can build new-generation ILP systems able to learn from relational databases integrated with ontologies according to the principles of Onto-Relational Learning.

Acknowledgements

We are grateful to Riccardo Rosati for his precious advice on $\mathcal{DL}$+LOG$^{\neg\vee}$ and Diego Calvanese for his valid support on the Boolean CQ/UCQ containment problem. Also we thank the anonymous reviewers whose commments very much helped us improving this paper.

References

BAADER, F., CALVANESE, D., MCGUINNESS, D., NARDI, D. AND PATEL-SCHNEIDER, P., Eds. 2007. *The Description Logic Handbook: Theory, Implementation and Applications*, 2nd ed., Cambridge University Press.

BERNERS-LEE, T., HENDLER, J. AND LASSILA, O. 2001. The semantic web. *Scientific American May.*

BLOCKEEL, H., DE RAEDT, L., JACOBS, N. AND DEMOEN, B. 1999. Scaling up inductive logic programming by learning from interpretations. *Data Mining and Knowledge Discovery 3,* 59–93.

BORGIDA, A. 1996. On the relative expressiveness of description logics and predicate logics. *Artificial Intelligence 82,* 1–2, 353–367.

BUNTINE, W. 1988. Generalized subsumption and its application to induction and redundancy. *Artificial Intelligence 36,* 2, 149–176.

CALVANESE, D., DE GIACOMO, G., LEMBO, D., LENZERINI, M. AND ROSATI, R. 2007. Tractable reasoning and efficient query answering in description logics: The *dl-lite* family. *Journal of Automated Reasoning 39,* 3, 385–429.

CALVANESE, D., DE GIACOMO, G. AND LENZERINI, M. 2008. Conjunctive query containment and answering under description logics constraints. *ACM Transactions on Computational Logic 9,* 3.

CERI, S., GOTTLOB, G. AND TANCA, L. 1989. What you always wanted to know about datalog (and never dared to ask). *IEEE Transactions on Knowledge and Data Engineering 1,* 1, 146–166.

[9] `http://www.w3.org/2005/rules/wiki/RIF_Working_Group`

Ceri, S., Gottlob, G. and Tanca, L. 1990. *Logic Programming and Databases*. Springer.

De Raedt, L. and Bruynooghe, M. 1993. A theory of clausal discovery. In *IJCAI*. 1058–1063.

De Raedt, L. and Dehaspe, L. 1997. Clausal discovery. *Machine Learning 26,* 2–3, 99–146.

De Raedt, L. and Džeroski, S. 1994. First order jk-clausal theories are PAC-learnable. *Artificial Intelligence 70*, 375–392.

Donini, F., Lenzerini, M., Nardi, D. and Schaerf, A. 1998. $\mathcal{AL}$-log: Integrating datalog and description logics. *Journal of Intelligent Information Systems 10,* 3, 227–252.

Džeroski, S. and Lavrač, N., Eds. 2001. *Relational Data Mining*. Springer.

Eiter, T., Gottlob, G. and Mannila, H. 1997. Disjunctive Datalog. *ACM Transactions on Database Systems 22,* 3, 364–418.

Flach, P. 1993. Predicate invention in inductive data engineering. In *Machine Learning: ECML-93*, P. Brazdil, Ed. Lecture Notes in Computer Science, vol. 667. Springer, 83–94.

Flach, P. 1998. From extensional to intensional knowledge: Inductive logic programming techniques and their application to deductive databases. In *Transactions and Change in Logic Databases*, B. Freitag, H. Decker, M. Kifer and A. Voronkov, Eds. Lecture Notes in Computer Science, vol. 1472. Springer, 356–387.

Frazier, M. and Pitt, L. 1993. Learning from entailment: An application to propositional horn sentences. In *Proc. of the Tenth International Conference on Machine Learning*. 120–127.

Frisch, A. and Cohn, A. 1991. Thoughts and afterthoughts on the 1988 workshop on principles of hybrid reasoning. *AI Magazine 11,* 5, 84–87.

Glimm, B., Horrocks, I., Lutz, C. and Sattler, U. 2008. Conjunctive query answering for the description logic $\mathcal{SHIQ}$. *Journal of Artificial Intelligence Research 31*, 151–198.

Glimm, B., Horrocks, I. and Sattler, U. 2008. Unions of conjunctive queries in $\mathcal{SHOQ}$. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Eleventh International Conference, KR 2008, Sydney, Australia, September 16–19, 2008*, G. Brewka and J. Lang, Eds. AAAI Press, 252–262.

Gómez-Pérez, A., Fernández-López, M. and Corcho, O. 2004. *Ontological Engineering*. Springer.

Gruber, T. 1993. A translation approach to portable ontology specifications. *Knowledge Acquisition 5*, 199–220.

Horrocks, I., Patel-Schneider, P. and van Harmelen, F. 2003. From $\mathcal{SHIQ}$ and RDF to OWL: The making of a web ontology language. *Journal of Web Semantics 1,* 1, 7–26.

Horrocks, I., Sattler, U. and Tobies, S. 2000. Practical reasoning for very expressive description logics. *Logic Journal of the IGPL 8,* 3, 239–263.

Kietz, J. 2003. Learnability of description logic programs. In *Inductive Logic Programming*, S. Matwin and C. Sammut, Eds. Lecture Notes in Artificial Intelligence, vol. 2583. Springer, 117–132.

Levy, A. and Rousset, M.-C. 1998. Combining Horn rules and description logics in CARIN. *Artificial Intelligence 104*, 165–209.

Lisi, F. A. 2008. Building rules on top of ontologies for the semantic web with inductive Logic Programming. *Theory and Practice of Logic Programming 8,* 03, 271–300.

Lisi, F. A. and Esposito, F. 2008. Foundations of onto-relational learning. In *Inductive Logic Programming*, F. Železný and N. Lavrač, Eds. Lecture Notes in Artificial Intelligence, vol. 5194. Springer, 158–175.

Lisi, F. A. and Malerba, D. 2004. Inducing multi-level association rules from multiple relations. *Machine Learning 55*, 175–210.

Michalski, R. 1983. A theory and methodology of inductive learning. In *Machine Learning: an artificial intelligence approach*, R. Michalski, J. Carbonell and T. Mitchell, Eds. Vol. I. Morgan Kaufmann, San Mateo, CA.

Mitchell, T. 1982. Generalization as search. *Artificial Intelligence 18*, 203–226.

Motik, B., Sattler, U. and Studer, R. 2005. Query Answering for OWL-DL with Rules. *Journal on Web Semantics 3,* 1, 41–60.

Muggleton, S. 1990. Inductive logic programming. In *Proceedings of the 1st Conference on Algorithmic Learning Theory*. Ohmsma, Tokyo, Japan.

Nienhuys-Cheng, S. and de Wolf, R. 1997. *Foundations of Inductive Logic Programming*. Lecture Notes in Artificial Intelligence, vol. 1228. Springer.

Plotkin, G. 1970. A note on inductive generalization. *Machine Intelligence 5*, 153–163.

Plotkin, G. 1971. A further note on inductive generalization. *Machine Intelligence 6*, 101–121.

Quinlan, J. 1990. Learning logical definitions from relations. *Machine Learning 5*, 239–266.

Reiter, R. 1980. Equality and domain closure in first order databases. *Journal of ACM 27*, 235–249.

Rosati, R. 2005a. On the decidability and complexity of integrating ontologies and rules. *Journal of Web Semantics 3,* 1, 61–73.

Rosati, R. 2005b. Semantic and computational advantages of the safe integration of ontologies and rules. In *Principles and Practice of Semantic Web Reasoning*, F. Fages and S. Soliman, Eds. Lecture Notes in Computer Science, vol. 3703. Springer, 50–64.

Rosati, R. 2006. $\mathcal{DL}$+log: Tight integration of description logics and disjunctive datalog. In *Proc. of Tenth International Conference on Principles of Knowledge Representation and Reasoning*, P. Doherty, J. Mylopoulos and C. Welty, Eds. AAAI Press, 68–78.

Rouveirol, C. and Ventos, V. 2000. Towards learning in CARIN-$\mathcal{ALN}$. In *Inductive Logic Programming*, J. Cussens and A. Frisch, Eds. Lecture Notes in Artificial Intelligence, vol. 1866. Springer, 191–208.

Sakama, C. 2001. Nonmonotonic inductive logic programming. In *Logic Programming and Nonmonotonic Reasoning*, T. Eiter, W. Faber and M. Truszczynski, Eds. Lecture Notes in Computer Science, vol. 2173. Springer, 62–80.

Savnik, I. and Flach, P. A. 2000. Discovery of multivalued dependencies from relations. *Intelligent Data Analysis 4,* 3-4, 195–211.

Schmidt-Schauss, M. and Smolka, G. 1991. Attributive concept descriptions with complements. *Artificial Intelligence 48,* 1, 1–26.

van der Laag, P. 1995. An analysis of refinement operators in inductive logic programming. PhD thesis, Erasmus University.