On Learning in $\mathcal{AL}$-log

Francesca A. Lisi and Floriana Esposito

Dipartimento di Informatica, Università degli Studi di Bari, Italy
lisi@di.uniba.it

Abstract. In this paper we provide a general framework for learning in $\mathcal{AL}$-log, a hybrid language that integrates the description logic $\mathcal{ALC}$ and the function-free Horn clausal language DATALOG. In this framework inductive hypotheses are represented as constrained DATALOG clauses, organized according to the $\mathcal{B}$-subsumption relation, and evaluated against observations by applying coverage relations that depend on the representation chosen for the observations. The framework is valid whatever the scope of induction (description vs. prediction) is. Yet we concentrate on an instantiation of the framework which corresponds to the logical setting of *characteristic induction from intepretations* and is particularly suitable for descriptive data mining tasks such as frequent pattern discovery (and its variants).

1 Introduction

Hybrid systems are a special class of knowledge representation and reasoning (KR&R) systems which are constituted by two or more subsystems dealing with distinct portions of a knowledge base and specific reasoning procedures [11]. The characterizing feature of hybrid systems is that the whole system is in charge of a single knowledge base, thus combining knowledge and reasoning services of the different subsystems in order to answer user questions. Indeed the motivation for building hybrid systems is to improve on two basic features of knowledge representation formalisms, namely *representational adequacy* and *deductive power*. Hybrid systems such as CARIN [13] and $\mathcal{AL}$-log [8] are particularly interesting because they bridge the gap between two fragments of first-order logic, description logics (DLs) [1] and Horn clausal logic, that are incomparable as for the expressive power [4]. In particular, $\mathcal{AL}$-log integrates $\mathcal{ALC}$ [23] and DATALOG [6] by using $\mathcal{ALC}$ concept assertions essentially as type constraints on variables.

In this paper we provide a *general framework for learning in $\mathcal{AL}$-log*. Inductive hypotheses are represented as constrained DATALOG clauses, organized according to the $\mathcal{B}$-subsumption relation, and evaluated against observations by applying coverage relations that depend on the representation chosen for the observations. In defining this learning framework we resort to the methodological apparatus of Inductive Logic Programming (ILP). ILP has been historically concerned with concept learning from examples and background knowledge within the representation framework of Horn clausal logic and with the aim of prediction [19]. More recently ILP has moved towards either different first-order logic fragments (e.g., DLs) or new learning goals (e.g., description).

The framework proposed is valid whatever the scope of induction (description vs. prediction) is. Yet we concentrate on an instantiation of the framework which corresponds to the logical setting of *characteristic induction from intepretations* and is particularly suitable for descriptive data mining tasks such as frequent pattern discovery (and its variants) [7].

The paper is organized as follows. Section 2 introduces the basic notions of $\mathcal{AL}$-log. Section 3 defines the framework for learning in $\mathcal{AL}$-log. Section 4 illustrates the instantiation of the framework in the case of characteristic induction from intepretations. Section 5 concludes the paper with final remarks.

2 Basics of $\mathcal{AL}$-log

The system $\mathcal{AL}$-log [8] integrates two KR&R systems: Structural and relational.

Table 1. Syntax and semantics of $\mathcal{ALC}$.

bottom (resp. top) concept	$\bot$ (resp. $\top$)	$\emptyset$ (resp. $\Delta^{\mathcal{I}}$)
atomic concept	A	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
role	R	$R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
individual	a	$a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$
concept negation	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
concept conjunction	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
concept disjunction	$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
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	$\exists R.C$	$\{x \in \Delta^{\mathcal{I}} \mid \exists y \ (x, y) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$
equivalence axiom	$C \equiv D$	$C^{\mathcal{I}} = D^{\mathcal{I}}$
subsumption axiom	$C \sqsubseteq D$	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
concept assertion	$a : C$	$a^{\mathcal{I}} \in C^{\mathcal{I}}$
role assertion	$\langle a, b \rangle : R$	$(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$

2.1 The structural subsystem

The structural part Σ is based on $\mathcal{ALC}$ [23] and allows for the specification of knowledge in terms of classes (*concepts*), binary relations between classes (*roles*), and instances (*individuals*). Complex concepts can be defined from atomic concepts and roles by means of constructors (see Table 1). Also 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*). An *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ for Σ consists of a domain $\Delta^{\mathcal{I}}$ and a mapping function $\cdot^{\mathcal{I}}$. In particular, individuals are mapped to elements of $\Delta^{\mathcal{I}}$ such that $a^{\mathcal{I}} \neq b^{\mathcal{I}}$ if $a \neq b$ (*Unique Names Assumption* (UNA) [20]). If $\mathcal{O} \subseteq \Delta^{\mathcal{I}}$ and

$\forall a \in \mathcal{O} : a^{\mathcal{I}} = a$, $\mathcal{I}$ is called *$\mathcal{O}$-interpretation*. Also Σ represents many different interpretations, i.e. all its models (*Open World Assumption* (OWA) [1]).

The main reasoning task for Σ is the *consistency check*. This test is performed with a *tableau calculus* that starts with the tableau branch $S = \Sigma$ and adds assertions to S by means of *propagation rules* such as

- $S \rightarrow_{\sqcup} S \cup \{s : D\}$ if
  1. $s : C_1 \sqcup C_2$ is in S,
  2. $D = C_1$ and $D = C_2$,
  3. neither $s : C_1$ nor $s : C_2$ is in S
- $S \rightarrow_{\forall} S \cup \{t : C\}$ if
  1. $s : \forall R.C$ is in S,
  2. sRt is in S,
  3. $t : C$ is not in S
- $S \rightarrow_{\sqsubseteq} S \cup \{s : C' \sqcup D\}$ if
  1. $C \sqsubseteq D$ is in S,
  2. s appears in S,
  3. C' is the NNF concept equivalent to $\neg C$
  4. $s : \neg C \sqcup D$ is not in S
- $S \rightarrow_{\perp} \{s : \perp\}$ if
  1. $s : A$ and $s : \neg A$ are in S, or
  2. $s : \neg\top$ is in S,
  3. $s : \perp$ is not in S

until either a contradiction is generated or an interpretation satisfying S can be easily obtained from it.

2.2 The relational subsystem

The relational part of $\mathcal{AL}$-log allows one to define DATALOG[1] programs enriched with *constraints* of the form $s : C$ where s is either a constant or a variable, and C is an $\mathcal{ALC}$-concept. Note that the usage of concepts as typing constraints applies only to variables and constants that already appear in the clause. The symbol & separates constraints from DATALOG atoms in a clause.

Definition 1. *A* constrained DATALOG clause *is an implication of the form* $\alpha_0 \leftarrow \alpha_1, \ldots, \alpha_m \& \gamma_1, \ldots, \gamma_n$ *where* $m \geq 0$, $n \geq 0$, α_i *are* DATALOG *atoms and* γ_j *are constraints. A* constrained DATALOG program Π *is a set of constrained* DATALOG *clauses.*

An *$\mathcal{AL}$-log knowledge base* $\mathcal{B}$ is the pair $\langle \Sigma, \Pi \rangle$ where Σ is an $\mathcal{ALC}$ knowledge base and Π is a constrained DATALOG program. For a knowledge base to be acceptable, it must satisfy the following conditions:

- The set of DATALOG predicate symbols appearing in Π is disjoint from the set of concept and role symbols appearing in Σ.

[1] For the sake of brevity we assume the reader to be familiar with DATALOG.

– The alphabet of constants in Π coincides with the alphabet $\mathcal{O}$ of the individuals in Σ. Furthermore, every constant in Π appears also in Σ.
– For each clause in Π, each variable occurring in the constraint part occurs also in the DATALOG part.

These properties state a *safe* interaction between the structural and the relational part of an $\mathcal{AL}$-log knowledge base, thus solving the semantic mismatch between the OWA of $\mathcal{ALC}$ and the CWA of DATALOG [21]. This interaction is also at the basis of a model-theoretic semantics for $\mathcal{AL}$-log. We call Π_D the set of DATALOG clauses obtained from the clauses of Π by deleting their constraints. We define an *interpretation* $\mathcal{J}$ for $\mathcal{B}$ as the union of an $\mathcal{O}$-interpretation $\mathcal{I}_\mathcal{O}$ for Σ (i.e. an interpretation compliant with the unique names assumption) and an Herbrand interpretation $\mathcal{I}_\mathcal{H}$ for Π_D. An interpretation $\mathcal{J}$ is a *model* of $\mathcal{B}$ if $\mathcal{I}_\mathcal{O}$ is a model of Σ, and for each ground instance $\alpha'_0 \leftarrow \alpha'_1, \ldots, \alpha'_m \& \gamma'_1, \ldots, \gamma'_n$ of each clause $\alpha_0 \leftarrow \alpha_1, \ldots, \alpha_m \& \gamma'_1, \ldots, \gamma'_n$ in Π, either there exists one γ'_i, $i \in \{1, \ldots, n\}$, that is not satisfied by $\mathcal{J}$, or $\alpha'_0 \leftarrow \alpha'_1, \ldots, \alpha'_m$ is satisfied by $\mathcal{J}$. The notion of *logical consequence* paves the way to the definition of answer set for queries. *Queries* to $\mathcal{AL}$-log knowledge bases are special cases of Definition 1. An *answer* to the query Q is a ground substitution σ for the variables in Q. The answer σ is *correct* w.r.t. a $\mathcal{AL}$-log knowledge base $\mathcal{B}$ if $Q\sigma$ is a logical consequence of $\mathcal{B}$ ($\mathcal{B} \models Q\sigma$). The *answer set* of Q in $\mathcal{B}$ contains all the correct answers to Q w.r.t. $\mathcal{B}$.

Reasoning for $\mathcal{AL}$-log knowledge bases is based on *constrained SLD-resolution* [8], i.e. an extension of SLD-resolution to deal with constraints. In particular, the constraints of the resolvent of a query Q and a constrained DATALOG clause E are recursively simplified by replacing couples of constraints $t : C$, $t : D$ with the equivalent constraint $t : C \sqcap D$. The one-to-one mapping between constrained SLD-derivations and the SLD-derivations obtained by ignoring the constraints is exploited to extend known results for DATALOG to $\mathcal{AL}$-log. Note that in $\mathcal{AL}$-log a derivation of the empty clause with associated constraints does not represent a refutation. It actually infers that the query is true in those models of $\mathcal{B}$ that satisfy its constraints. Therefore in order to answer a query it is necessary to collect enough derivations ending with a constrained empty clause such that every model of $\mathcal{B}$ satisfies the constraints associated with the final query of at least one derivation.

Definition 2. *Let $Q^{(0)}$ be a query $\leftarrow \beta_1, \ldots, \beta_m \& \gamma_1, \ldots, \gamma_n$ to a $\mathcal{AL}$-log knowledge base $\mathcal{B}$. A* constrained SLD-refutation *for $Q^{(0)}$ in $\mathcal{B}$ is a finite set $\{d_1, \ldots, d_s\}$ of constrained SLD-derivations for $Q^{(0)}$ in $\mathcal{B}$ such that:*

1. *for each derivation d_i, $1 \leq i \leq s$, the last query $Q^{(n_i)}$ of d_i is a constrained empty clause;*
2. *for every model $\mathcal{J}$ of $\mathcal{B}$, there exists at least one derivation d_i, $1 \leq i \leq s$, such that $\mathcal{J} \models Q^{(n_i)}$*

Constrained SLD-refutation is a complete and sound method for answering *ground* queries.

Lemma 1. *[8] Let Q be a ground query to an $\mathcal{AL}$-log knowledge base $\mathcal{B}$. It holds that $\mathcal{B} \vdash Q$ if and only if $\mathcal{B} \models Q$.*

An answer σ to a query Q is a *computed answer* if there exists a constrained SLD-refutation for $Q\sigma$ in $\mathcal{B}$ ($\mathcal{B} \vdash Q\sigma$). The set of computed answers is called the *success set* of Q in $\mathcal{B}$. Furthermore, given *any* query Q, the success set of Q in $\mathcal{B}$ coincides with the answer set of Q in $\mathcal{B}$. This provides an operational means for computing correct answers to queries. Indeed, it is straightforward to see that the usual reasoning methods for DATALOG allow us to collect in a finite number of steps enough constrained SLD-derivations for Q in $\mathcal{B}$ to construct a refutation - if any. Derivations must satisfy both conditions of Definition 2. In particular, the latter requires some reasoning on the structural component of $\mathcal{B}$. This is done by applying the tableau calculus as shown in the following example.

Constrained SLD-resolution is *decidable*. Furthermore, because of the safe interaction between $\mathcal{ALC}$ and DATALOG, it supports a form of *closed world reasoning*, i.e. it allows one to pose queries under the assumption that part of the knowledge base is complete.

3 The general framework for learning in $\mathcal{AL}$-log

In our framework for learning in $\mathcal{AL}$-log we represent inductive hypotheses as constrained DATALOG clauses and data as an $\mathcal{AL}$-log knowledge base $\mathcal{B}$. In particular $\mathcal{B}$ is composed of a *background knowledge* $\mathcal{K}$ and a set O of *observations*. We assume $\mathcal{K} \cap O = \emptyset$.

To define the framework we resort to the methodological apparatus of ILP which requires the following ingredients to be chosen:

- the *language* $\mathcal{L}$ *of hypotheses*
- a *generality order* $\succeq$ for $\mathcal{L}$ to structure the space of hypotheses
- a *relation* to test the *coverage* of hypotheses in $\mathcal{L}$ against observations in O w.r.t. $\mathcal{K}$

The framework is **general**, meaning that it is valid whatever the scope of induction (description/prediction) is. Therefore the DATALOG literal $q(\boldsymbol{X})$[2] in the head of hypotheses represents a concept to be either discriminated from others (*discriminant induction*) or characterized (*characteristic induction*).

3.1 The language of hypotheses

To be suitable as language of hypotheses, constrained DATALOG clauses must satisfy the following restrictions.

First, we impose constrained DATALOG clauses to be linked and connected (or range-restricted) as usual in ILP.

[2] $\boldsymbol{X}$ is a tuple of variables

Definition 3. *Let H be a constrained* Datalog *clause. A term t in some literal $l_i \in H$ is* linked *with linking-chain of length 0, if t occurs in head(H), and is linked with linking-chain of length $d+1$, if some other term in l_i is linked with linking-chain of length d. The link-depth of a term t in some $l_i \in H$ is the length of the shortest linking-chain of t. A literal $l_i \in H$ is linked if at least one of its terms is linked. The clause H itself is linked if each $l_i \in H$ is linked. The clause H is* connected *if each variable occurring in head(H) also occur in body(H).*

Second, we impose constrained Datalog clauses to be compliant with the bias of Object Identity (OI) [24]. This bias can be considered as an extension of the unique names assumption from the semantic level to the syntactic one of $\mathcal{AL}$-log. We would like to remind the reader that this assumption holds in $\mathcal{ALC}$. Also it holds naturally for ground constrained Datalog clauses because the semantics of $\mathcal{AL}$-log adopts Herbrand models for the Datalog part and $\mathcal{O}$-models for the constraint part. Conversely it is not guaranteed in the case of non-ground constrained Datalog clauses, e.g. different variables can be unified. The OI bias can be the starting point for the definition of either an equational theory or a quasi-order for constrained Datalog clauses. The latter option relies on a restricted form of substitution whose bindings avoid the identification of terms.

Definition 4. *A substitution σ is an* OI-substitution *w.r.t. a set of terms T iff $\forall t_1, t_2 \in T\colon\ t_1 \neq t_2$ yields that $t_1\sigma \neq t_2\sigma$.*

From now on, we assume that substitutions are OI-compliant.

3.2 The generality relation

The definition of a generality relation for constrained Datalog clauses can disregard neither the peculiarities of $\mathcal{AL}$-log nor the methodological apparatus of ILP. Therefore we rely on the reasoning mechanisms made available by $\mathcal{AL}$-log knowledge bases and propose to adapt Buntine's generalized subsumption [5] to our framework as follows.

Definition 5. *Let H be a constrained* Datalog *clause, α a ground* Datalog *atom, and $\mathcal{J}$ an interpretation. We say that H* covers *α under $\mathcal{J}$ if there is a ground substitution θ for H ($H\theta$ is ground) such that body$(H)\theta$ is true under $\mathcal{J}$ and head$(H)\theta = \alpha$.*

Definition 6. *Let H_1, H_2 be two constrained* Datalog *clauses and $\mathcal{B}$ an $\mathcal{AL}$-log knowledge base. We say that H_1* $\mathcal{B}$-subsumes *H_2 if for every model $\mathcal{J}$ of $\mathcal{B}$ and every ground atom α such that H_2 covers α under $\mathcal{J}$, we have that H_1 covers α under $\mathcal{J}$.*

We can define a generality relation $\succeq_{\mathcal{B}}$ for constrained Datalog clauses on the basis of $\mathcal{B}$-subsumption. It can be easily proven that $\succeq_{\mathcal{B}}$ is a quasi-order (i.e. it is a reflexive and transitive relation) for constrained Datalog clauses.

Definition 7. *Let H_1, H_2 be two constrained* DATALOG *clauses and $\mathcal{B}$ an $\mathcal{AL}$-log knowledge base. We say that H_1 is* at least as general as *H_2 under $\mathcal{B}$-subsumption, $H_1 \succeq_{\mathcal{B}} H_2$, iff H_1 $\mathcal{B}$-subsumes H_2. Furthermore, H_1 is* more general than *H_2 under $\mathcal{B}$-subsumption, $H_1 \succ_{\mathcal{B}} H_2$, iff $H_1 \succeq_{\mathcal{B}} H_2$ and $H_2 \not\succeq_{\mathcal{B}} H_1$. Finally, H_1 is* equivalent *to H_2 under $\mathcal{B}$-subsumption, $H_1 \sim_{\mathcal{B}} H_2$, iff $H_1 \succeq_{\mathcal{B}} H_2$ and $H_2 \succeq_{\mathcal{B}} H_1$.*

The next lemma shows the definition of $\mathcal{B}$-subsumption to be equivalent to another formulation, which will be more convenient in later proofs than the definition based on covering.

Definition 8. *Let $\mathcal{B}$ be an $\mathcal{AL}$-log knowledge base and H be a constrained* DATALOG *clause. Let $X_1, \ldots, X_n$ be all the variables appearing in H, and $a_1, \ldots, a_n$ be distinct constants (individuals) not appearing in $\mathcal{B}$ or H. Then the substitution $\{X_1/a_1, \ldots, X_n/a_n\}$ is called a* Skolem substitution *for H w.r.t. $\mathcal{B}$.*

Lemma 2. *[15] Let H_1, H_2 be two constrained* DATALOG *clauses, $\mathcal{B}$ an $\mathcal{AL}$-log knowledge base, and σ a Skolem substitution for H_2 with respect to $\{H_1\} \cup \mathcal{B}$. We say that $H_1 \succeq_{\mathcal{B}} H_2$ iff there exists a ground substitution θ for H_1 such that (i) $head(H_1)\theta = head(H_2)\sigma$ and (ii) $\mathcal{B} \cup body(H_2)\sigma \models body(H_1)\theta$.*

The relation between $\mathcal{B}$-subsumption and constrained SLD-resolution is given below. It provides an operational means for checking $\mathcal{B}$-subsumption.

Theorem 1 *Let H_1, H_2 be two constrained* DATALOG *clauses, $\mathcal{B}$ an $\mathcal{AL}$-log knowledge base, and σ a Skolem substitution for H_2 with respect to $\{H_1\} \cup \mathcal{B}$. We say that $H_1 \succeq_{\mathcal{B}} H_2$ iff there exists a substitution θ for H_1 such that (i) $head(H_1)\theta = head(H_2)$ and (ii) $\mathcal{B} \cup body(H_2)\sigma \vdash body(H_1)\theta\sigma$ where $body(H_1)\theta\sigma$ is ground.*

Proof. By Lemma 2, we have $H_1 \succeq_{\mathcal{B}} H_2$ iff there exists a ground substitution θ' for H_1, such that $head(H_1)\theta' = head(H_2)\sigma$ and $\mathcal{B} \cup body(H_2)\sigma \models body(H_1)\theta'$. Since σ is a Skolem substitution, we can define a substitution θ such that $H_1\theta\sigma = H_1\theta'$ and none of the Skolem constants of σ occurs in θ. Then $head(H_1)\theta = head(H_2)$ and $\mathcal{B} \cup body(H_2)\sigma \models body(H_1)\theta\sigma$. Since $body(H_1)\theta\sigma$ is ground, by Lemma 1 we have $\mathcal{B} \cup body(H_2)\sigma \vdash body(H_1)\theta\sigma$, so the thesis follows.

The decidability of $\mathcal{B}$-subsumption follows from the decidability of both generalized subsumption in DATALOG [5] and query answering in $\mathcal{AL}$-log [8].

3.3 Coverage relations

When defining coverage relations we make assumptions as regards the representation of observations because it impacts the definition of coverage.

In the logical setting of *learning from entailment* extended to $\mathcal{AL}$-log, an observation $o_i \in O$ is represented as a ground constrained DATALOG clause having a ground atom $q(\boldsymbol{a}_i)$[3] in the head.

[3] $\boldsymbol{a}_i$ is a tuple of constants

Definition 9. *Let $H \in \mathcal{L}$ be a hypothesis, $\mathcal{K}$ a background knowledge and $o_i \in O$ an observation. We say that H* covers o_i under entailment *w.r.t $\mathcal{K}$ iff $\mathcal{K} \cup H \models o_i$.*

Theorem 2 *[14] Let $H \in \mathcal{L}$ be a hypothesis, $\mathcal{K}$ a background knowledge, and $o_i \in O$ an observation. We say that H* covers o_i under entailment w.r.t. $\mathcal{K}$ *iff $\mathcal{K} \cup body(o_i) \cup H \vdash q(\boldsymbol{a}_i)$.*

In the logical setting of *learning from interpretations* extended to $\mathcal{AL}$-log, an observation $o_i \in O$ is represented as a couple $(q(\boldsymbol{a}_i), \mathcal{A}_i)$ where $\mathcal{A}_i$ is a set containing ground DATALOG facts concerning the individual i.

Definition 10. *Let $H \in \mathcal{L}$ be a hypothesis, $\mathcal{K}$ a background knowledge and $o_i \in O$ an observation. We say that H* covers o_i under interpretations w.r.t. $\mathcal{K}$ *iff $\mathcal{K} \cup \mathcal{A}_i \cup H \models q(\boldsymbol{a}_i)$.*

Theorem 3 *[14] Let $H \in \mathcal{L}$ be a hypothesis, $\mathcal{K}$ a background knowledge, and $o_i \in O$ an observation. We say that H* covers o_i under interpretations w.r.t. $\mathcal{K}$ *iff $\mathcal{K} \cup \mathcal{A}_i \cup H \vdash q(\boldsymbol{a}_i)$.*

Note that the both coverage tests can be reduced to query answering.

4 An instantiation of the framework

As an instantiation of our general framework for learning in $\mathcal{AL}$-log we choose the case of *characteristic induction from interpretations* which is defined as follows.

Definition 11. *Let $\mathcal{L}$ be a hypothesis language, $\mathcal{K}$ a background knowledge, O a set of observations, and $M(\mathcal{B})$ a model constructed from $\mathcal{B} = \mathcal{K} \cup O$. The goal of* characteristic induction from interpretations *is to find a set $\mathcal{H} \subseteq \mathcal{L}$ of hypotheses such that (i) $\mathcal{H}$ is true in $M(\mathcal{B})$, and (ii) for each $H \in \mathcal{L}$, if H is true in $M(\mathcal{B})$ then $\mathcal{H} \models H$.*

The logical setting of characteristic induction has been considered very close to that form of data mining, called *descriptive data mining*, which focuses on finding human-interpretable patterns describing a data set $\mathbf{r}$ [7]. *Scalability* is a crucial issue in descriptive data mining. Recently, the setting of learning from interpretations has been shown to be a promising way of scaling up ILP algorithms in real-world applications [3].

4.1 A task of characteristic induction

Among descriptive data mining tasks, *frequent pattern discovery* aims at the extraction of all patterns whose cardinality exceeds a user-defined threshold. Indeed each pattern is considered as an intensional description (expressed in a given language $\mathcal{L}$) of a subset of $\mathbf{r}$.

The blueprint of most algorithms for frequent pattern discovery is the *level-wise search* [18]. It is based on the following assumption: If a generality order

$\succeq$ for the language $\mathcal{L}$ of patterns can be found such that $\succeq$ is monotonic w.r.t. the evaluation function *supp*, then the resulting space $(\mathcal{L}, \succeq)$ can be searched breadth-first starting from the most general pattern in $\mathcal{L}$ and by alternating *candidate generation* and *candidate evaluation* phases. In particular, candidate generation consists of a refinement step followed by a pruning step. The former derives candidates for the current search level from patterns found frequent in the previous search level. The latter allows some infrequent patterns to be detected and discarded prior to evaluation thanks to the monotonicity of $\succeq$.

We consider a variant of this task which takes concept hierarchies into account during the discovery process, thus yielding descriptions of $\mathbf{r}$ at multiple granularity levels [17]. More formally, given

- a data set $\mathbf{r}$ including a taxonomy $\mathcal{T}$ where a reference concept C_{ref} and task-relevant concepts are designated,
- a multi-grained language $\mathcal{L} = \{\mathcal{L}^l\}_{1 \leq l \leq maxG}$ of patterns
- a set $\{minsup^l\}_{1 \leq l \leq maxG}$ of support thresholds

the problem of *frequent pattern discovery at l levels of description granularity*, $1 \leq l \leq maxG$, is to find the set $\mathcal{F}$ of all the patterns $P \in \mathcal{L}^l$ frequent in $\mathbf{r}$, namely P's with support s such that (i) $s \geq minsup^l$ and (ii) all ancestors of P w.r.t. $\mathcal{T}$ are frequent in $\mathbf{r}$.

4.2 Casting the framework to the task

When casting our general framework for learning in $\mathcal{AL}$-log to the task of frequent pattern discovery at multiple levels of description granularity, the data set $\mathbf{r}$ is represented as an $\mathcal{AL}$-log knowledge base.

Example 1. As a running example, we consider an $\mathcal{AL}$-log knowledge base $\mathcal{B}_{\texttt{CIA}}$ that enriches Datalog facts[4] extracted from the on-line 1996 CIA World Fact Book[5] with $\mathcal{ALC}$ ontologies. The structural subsystem Σ of $\mathcal{B}_{\texttt{CIA}}$ focuses on the concepts `Country`, `EthnicGroup`, `Language`, and `Religion`. Axioms like

`AsianCountry` $\sqsubseteq$ `Country`.
`MiddleEasternEthnicGroup` $\sqsubseteq$ `EthnicGroup`.
`MiddleEastCountry` $\equiv$ `AsianCountry` $\sqcap$ $\exists$`Hosts.MiddleEasternEthnicGroup`.
`IndoEuropeanLanguage` $\sqsubseteq$ `Language`.
`IndoIranianLanguage` $\sqsubseteq$ `IndoEuropeanLanguage`.
`MonotheisticReligion` $\sqsubseteq$ `Religion`.
`ChristianReligion` $\sqsubseteq$ `MonotheisticReligion`.
`MuslimReligion` $\sqsubseteq$ `MonotheisticReligion`.

define four taxonomies, one for each concept above. Note that Middle East countries (concept `MiddleEastCountry`) have been defined as Asian countries that host at least one Middle Eastern ethnic group. Assertions like

[4] `http://www.dbis.informatik.uni-goettingen.de/Mondial/mondial-rel-facts.flp`
[5] `http://www.odci.gov/cia/publications/factbook/`

```
'ARM':AsianCountry.
'IR':AsianCountry.
'Arab':MiddleEasternEthnicGroup.
'Armenian':MiddleEasternEthnicGroup.
<'ARM','Armenian'>:Hosts.
<'IR','Arab'>:Hosts.
'Armenian':IndoEuropeanLanguage.
'Persian':IndoIranianLanguage.
'Armenian Orthodox':ChristianReligion.
'Shia':MuslimReligion.
'Sunni':MuslimReligion.
```

belong to the extensional part of Σ. In particular, Armenia (`'ARM'`) and Iran (`'IR'`) are two of the 14 countries that are classified as Middle Eastern.

The relational subsystem Π of $\mathcal{B}_{\mathtt{CIA}}$ expresses the CIA facts as a constrained DATALOG program. The extensional part of Π consists of DATALOG facts like

```
language('ARM','Armenian',96).
language('IR','Persian',58).
religion('ARM','Armenian Orthodox',94).
religion('IR','Shia',89).
religion('IR','Sunni',10).
```

whereas the intensional part defines two views on `language` and `religion`:

```
speaks(CountryID, LanguageN)←language(CountryID,LanguageN,Perc)
                        & CountryID:Country, LanguageN:Language
believes(CountryID, ReligionN)←religion(CountryID,ReligionN,Perc)
                        & CountryID:Country, ReligionN:Religion
```

that can deduce new DATALOG facts when triggered on $\mathcal{B}_{\mathtt{CIA}}$.

The language $\mathcal{L}$ for a given problem instance is implicitly defined by a declarative bias specification that allows for the generation of expressions, called $\mathcal{O}$-queries, relating individuals of C_{ref} to individuals of the task-relevant concepts.

Definition 12. *Given a $\mathcal{ALC}$ concept C_{ref}, an $\mathcal{O}$-*query *Q to an $\mathcal{AL}$-log knowledge base $\mathcal{B}$ is a (linked, connected, and OI-compliant) constrained* DATALOG *clause of the form*

$$Q = q(X) \leftarrow \alpha_1, \ldots, \alpha_m \& X : C_{ref}, \gamma_1, \ldots, \gamma_n$$

where X is the distinguished variable *and the remaining variables occurring in the body of Q are the* existential variables*.*

The $\mathcal{O}$-query $Q_t = q(X) \leftarrow \& X : C_{ref}$ is called *trivial* for $\mathcal{L}$.

Example 2. We want to describe Middle East countries (individuals of the reference concept) with respect to the religions believed and the languages spoken (individuals of the task-relevant concepts) at three levels of granularity ($maxG = 3$). To this aim we define $\mathcal{L}_{\mathtt{CIA}}$ as the set of $\mathcal{O}$-queries with C_{ref} = `MiddleEastCountry` that can be generated from the alphabet $\mathcal{A}$= {`believes/2`, `speaks/2`} of DATALOG binary predicate names, and the alphabets

$\Gamma^1 = \{\texttt{Language}, \texttt{Religion}\}$
$\Gamma^2 = \{\texttt{IndoEuropeanLanguage}, \ldots, \texttt{MonotheisticReligion}, \ldots\}$
$\Gamma^3 = \{\texttt{IndoIranianLanguage}, \ldots, \texttt{MuslimReligion}, \ldots\}$

of $\mathcal{ALC}$ concept names for $1 \leq l \leq 3$. Examples of $\mathcal{O}$-queries in $\mathcal{L}_{\texttt{CIA}}$ are:

$Q_t = \texttt{q(X)} \leftarrow \& \ \texttt{X:MiddleEastCountry}$
$Q_1 = \texttt{q(X)} \leftarrow \texttt{believes(X,Y)} \ \& \ \texttt{X:MiddleEastCountry}, \texttt{Y:Religion}$
$Q_2 = \texttt{q(X)} \leftarrow \texttt{believes(X,Y)}, \texttt{speaks(X,Z)} \ \& \ \texttt{X:MiddleEastCountry},$
$\qquad \texttt{Y:MonotheisticReligion}, \texttt{Z:IndoEuropeanLanguage}$
$Q_3 = \texttt{q(X)} \leftarrow \texttt{believes(X,Y)}, \texttt{speaks(X,Z)} \ \& \ \texttt{X:MiddleEastCountry},$
$\qquad \texttt{Y:MuslimReligion}, \texttt{Z:IndoIranianLanguage}$

where Q_t is the trivial $\mathcal{O}$-query for $\mathcal{L}_{\texttt{CIA}}$, $Q_1 \in \mathcal{L}^1_{\texttt{CIA}}$, $Q_2 \in \mathcal{L}^2_{\texttt{CIA}}$, and $Q_3 \in \mathcal{L}^3_{\texttt{CIA}}$.

Being a special case of constrained DATALOG clauses, $\mathcal{O}$-queries can be $\succeq_{\mathcal{B}}$-ordered. Also note that the underlying reasoning mechanism of $\mathcal{AL}$-log makes $\mathcal{B}$-subsumption more powerful than generalized subsumption as illustrated in the following example.

Example 3. We want to check whether Q_1 $\mathcal{B}$-subsumes the $\mathcal{O}$-query

$Q_4 = \texttt{q(A)} \leftarrow \texttt{believes(A,B)} \ \& \ \texttt{A:MiddleEastCountry}, \texttt{B:MonotheisticReligion}$

belonging to $\mathcal{L}^2_{\texttt{CIA}}$. Let $\sigma = \{\texttt{A/a}, \texttt{B/b}\}$ a Skolem substitution for Q_4 w.r.t. $\mathcal{B}_{\texttt{CIA}} \cup \{Q_1\}$ and $\theta = \{\texttt{X/A}, \texttt{Y/B}\}$ a substitution for Q_1. The condition (i) of Theorem 1 is immediately verified. It remains to verify that (ii) $\mathcal{B}' =$

$$\mathcal{B}_{\texttt{CIA}} \cup \{\texttt{believes(a,b)}, \texttt{a:MiddleEastCountry}, \texttt{b:MonotheisticReligion}\}$$
$$\models \texttt{believes(a,b)} \ \& \ \texttt{a:MiddleEastCountry}, \texttt{b:Religion}.$$

We try to build a constrained SLD-refutation for

$Q^{(0)} = \leftarrow \texttt{believes(a,b)} \ \& \ \texttt{a:MiddleEastCountry}, \texttt{b:Religion}$

in $\mathcal{B}'$. Let $E^{(1)}$ be $\texttt{believes(a,b)}$. A resolvent for $Q^{(0)}$ and $E^{(1)}$ with the empty substitution $\sigma^{(1)}$ is the constrained empty clause

$Q^{(1)} = \leftarrow \& \ \texttt{a:MiddleEastCountry}, \texttt{b:Religion}$

The consistency of $\Sigma'' = \Sigma' \cup \{\texttt{a:MiddleEastCountry}, \texttt{b:Religion}\}$ needs now to be checked. The first unsatisfiability check operates on the initial tableau $S_1^{(0)} = \Sigma' \cup \{\texttt{a:}\neg\texttt{MiddleEastCountry}\}$. The application of the propagation rule $\rightarrow_\perp$ to $S_1^{(0)}$ produces the final tableau $S_1^{(1)} = \{\texttt{a:}\perp\}$. Therefore $S_1^{(0)}$ is unsatisfiable. The second check starts with $S_2^{(0)} = \Sigma' \cup \{\texttt{b:}\neg\texttt{Religion}\}$. The rule $\rightarrow_\sqsubseteq$ w.r.t. $\texttt{MonotheisticReligion} \sqsubseteq \texttt{Religion}$, the only one applicable to $S_2^{(0)}$, produces $S_2^{(1)} = \Sigma \cup \{\texttt{b:}\neg\texttt{Religion}, \texttt{b:}\neg\texttt{MonotheisticReligion} \sqcup \texttt{Religion}\}$. By applying $\rightarrow_\sqcup$ to $S_2^{(1)}$ w.r.t. $\texttt{Religion}$ we obtain $S_2^{(2)} = \Sigma \cup \{\texttt{b:}\neg\texttt{Religion}, \texttt{b:Religion}\}$ which brings to the final tableau $S_2^{(3)} = \{\texttt{b:}\perp\}$ via $\rightarrow_\perp$.

Having proved the consistency of Σ'', we have proved the existence of a constrained SLD-refutation for $Q^{(0)}$ in $\mathcal{B}'$. Therefore we can say that $Q_1 \succeq_{\mathcal{B}} Q_4$. Conversely, $Q_4 \not\succeq_{\mathcal{B}} Q_1$. Similarly it can be proved that $Q_2 \succeq_{\mathcal{B}} Q_3$ and $Q_3 \not\succeq_{\mathcal{B}} Q_2$.

Example 4. It can be easily verified that Q_1 $\mathcal{B}$-subsumes the following query

Q_5= `q(A)` $\leftarrow$ `believes(A,B)`, `believes(A,C)` & `A:MiddleEastCountry`, `B:Religion`

by choosing σ={`A/a`, `B/b`, `C/c`} as a Skolem substitution for Q_5 w.r.t. $\mathcal{B}_{\mathtt{CIA}} \cup \{Q_1\}$ and θ={`X/A`, `Y/B`} as a substitution for Q_1. Note that $Q_5 \not\succeq_{\mathcal{B}} Q_1$ under the OI bias. Indeed this bias does not admit the substitution {`A/X`, `B/Y`, `C/Y`} for Q_5 which would make possible to verify conditions (i) and (ii) of Theorem 1.

The coverage test reduces to query answering. An *answer* to an $\mathcal{O}$-query Q is a ground substitution θ for the distinguished variable of Q. The conditions of well-formedness reported in Definition 3 guarantee that the evaluation of $\mathcal{O}$-queries is sound according to the following notions of answer/success set.

Definition 13. *An answer θ to an $\mathcal{O}$-query Q is a* correct (resp. computed) answer *w.r.t. an $\mathcal{AL}$-log knowledge base $\mathcal{B}$ if there exists at least one correct (resp. computed) answer to body$(Q)\theta$ w.r.t. $\mathcal{B}$.*

Therefore proving that an $\mathcal{O}$-query Q covers an observation $(q(a_i), \mathcal{A}_i)$ w.r.t. $\mathcal{K}$ equals to proving that $\theta_i = \{X/a_i\}$ is a correct answer to Q w.r.t. $\mathcal{B}_i = \mathcal{K} \cup \mathcal{A}_i$.

Example 5. With reference to Example 1, the background knowledge $\mathcal{K}_{\mathtt{CIA}}$ encompasses the structural part and the intensional relational part of $\mathcal{B}_{\mathtt{CIA}}$. We want to check whether the $\mathcal{O}$-query Q_1 reported in Example 2 covers the observation (`q('IR')`, $\mathcal{A}_{\mathtt{IR}}$) w.r.t. $\mathcal{K}_{\mathtt{CIA}}$. This is equivalent to answering the query

$\leftarrow$ `q('IR')`

w.r.t. $\mathcal{K}_{\mathtt{CIA}} \cup \mathcal{A}_{\mathtt{IR}} \cup Q_1$. Note that $\mathcal{A}_{\mathtt{IR}}$ contains all the Datalog facts concerning the individual `IR`.

The *support* of an $\mathcal{O}$-query $Q \in \mathcal{L}$ w.r.t. $\mathcal{B}$ supplies the percentage of individuals of C_{ref} that satisfy Q and is defined as

$$supp(Q, \mathcal{B}) = \mid answerset(Q, \mathcal{B}) \mid / \mid answerset(Q_t, \mathcal{B}) \mid$$

where $answerset(Q, \mathcal{B})$ is the set of correct answers to Q w.r.t. $\mathcal{B}$. We remind the reader that Q_t is the trivial $\mathcal{O}$-query for $\mathcal{L}$.

Example 6. Since $\mid answerset(Q_1, \mathcal{B}_{\mathtt{CIA}}) \mid = 14$ and $\mid answerset(Q_t, \mathcal{B}_{\mathtt{CIA}}) \mid = \mid$ `MiddleEastCountry` $\mid = 14$, then $supp(Q_1, \mathcal{B}_{\mathtt{CIA}}) = 100\%$. Analogously the support of the other $\mathcal{O}$-queries listed in Example 2 can be computed.

It has been proved that $\succeq_{\mathcal{B}}$ is monotone w.r.t. *supp* [17]. This has allowed us to implement the levelwise search. The resulting ILP system has been called $\mathcal{AL}$-QuIn ($\mathcal{AL}$-log Query Induction) [16,14]. In particular [16] supplies details of candidate generation whereas [14] provides insight into candidate evaluation.

5 Conclusions

Hybrid languages like $\mathcal{AL}$-log supply expressive and deductive power which have no counterpart in *pure* Horn clausal logic. This makes them appealing for challenging applications in application domains that require a uniform treatment of both relational and structural features of data.

Learning in DL-based hybrid languages has very recently attracted attention in the ILP community. In [22] the chosen language is CARIN-$\mathcal{ALN}$, therefore example coverage and subsumption between two hypotheses are based on the existential entailment algorithm of CARIN. Following [22], Kietz studies the learnability of CARIN-$\mathcal{ALN}$, thus providing a pre-processing method which enables ILP systems to learn CARIN-$\mathcal{ALN}$ rules [12]. In [17], Lisi and Malerba propose $\mathcal{AL}$-log as a KR&R framework for the induction of association rules. Closely related to DL-based hybrid systems are the proposals arising from the study of many-sorted logics, where a first-order language is combined with a sort language which can be regarded as an elementary DL [9]. In this respect the study of a sorted downward refinement [10] can be also considered a contribution to learning in hybrid languages.

The main contribution of this paper is the definition of a general framework for learning in $\mathcal{AL}$-log. We would like to emphasize that $\mathcal{AL}$-log has been preferred to CARIN for two desirable properties of constrained SLD-resolution which are particularly appreciated in ILP: *decidability* and *closed world reasoning*. We intend to extend the framework towards more expressive hybrid languages along the direction shown in [21] in order to make it closer to the representation standards for the logical layer of the Semantic Web [2].

Acknowledgements. We are grateful to Francesco M. Donini and Riccardo Rosati for their precious advice on $\mathcal{AL}$-log. Also we acknowledge the financial support of the 2004 project "Astrazione e Logica Descrittiva in Apprendimento Automatico" funded by the *Università degli Studi di Bari*.

References

1. F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P.F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, 2003.
2. T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, May, 2001.
3. H. Blockeel, L. De Raedt, N. Jacobs, and B. Demoen. Scaling Up Inductive Logic Programming by Learning from Interpretations. *Data Mining and Knowledge Discovery*, 3:59–93, 1999.
4. A. Borgida. On the relative expressiveness of description logics and predicate logics. *Artificial Intelligence*, 82(1–2):353–367, 1996.
5. W. Buntine. Generalized subsumption and its application to induction and redundancy. *Artificial Intelligence*, 36(2):149–176, 1988.
6. S. Ceri, G. Gottlob, and L. Tanca. *Logic Programming and Databases*. Springer, 1990.

7. L. De Raedt and L. Dehaspe. Clausal Discovery. *Machine Learning*, 26(2–3):99–146, 1997.
8. F.M. Donini, M. Lenzerini, D. Nardi, and A. Schaerf. $\mathcal{AL}$-log: Integrating Datalog and Description Logics. *Journal of Intelligent Information Systems*, 10(3):227–252, 1998.
9. A.M. Frisch. The substitutional framework for sorted deduction: Fundamental results on hybrid reasoning. *Artificial Intelligence*, 49:161–198, 1991.
10. A.M. Frisch. Sorted downward refinement: Building background knowledge into a refinement operator for inductive logic programming. In S. Džeroski and P. Flach, editors, *Inductive Logic Programming*, volume 1634 of *Lecture Notes in Artificial Intelligence*, pages 104–115. Springer, 1999.
11. A.M. Frisch and A.G. Cohn. Thoughts and afterthoughts on the 1988 workshop on principles of hybrid reasoning. *AI Magazine*, 11(5):84–87, 1991.
12. J.-U. Kietz. Learnability of description logic programs. In S. Matwin and C. Sammut, editors, *Inductive Logic Programming*, volume 2583 of *Lecture Notes in Artificial Intelligence*, pages 117–132. Springer, 2003.
13. A.Y. Levy and M.-C. Rousset. Combining Horn rules and description logics in CARIN. *Artificial Intelligence*, 104:165–209, 1998.
14. F.A. Lisi and F. Esposito. Efficient Evaluation of Candidate Hypotheses in $\mathcal{AL}$-log. In R. Camacho, R. King, and A. Srinivasan, editors, *Inductive Logic Programming*, volume 3194 of *Lecture Notes in Artificial Intelligence*, pages 216–233. Springer, 2004.
15. F.A. Lisi and D. Malerba. Bridging the Gap between Horn Clausal Logic and Description Logics in Inductive Learning. In A. Cappelli and F. Turini, editors, *AI*IA 2003: Advances in Artificial Intelligence*, volume 2829 of *Lecture Notes in Artificial Intelligence*, pages 49–60. Springer, 2003.
16. F.A. Lisi and D. Malerba. Ideal Refinement of Descriptions in $\mathcal{AL}$-log. In T. Horvath and A. Yamamoto, editors, *Inductive Logic Programming*, volume 2835 of *Lecture Notes in Artificial Intelligence*, pages 215–232. Springer, 2003.
17. F.A. Lisi and D. Malerba. Inducing Multi-Level Association Rules from Multiple Relations. *Machine Learning*, 55:175–210, 2004.
18. H. Mannila and H. Toivonen. Levelwise search and borders of theories in knowledge discovery. *Data Mining and Knowledge Discovery*, 1(3):241–258, 1997.
19. S.-H. Nienhuys-Cheng and R. de Wolf. *Foundations of Inductive Logic Programming*, volume 1228 of *Lecture Notes in Artificial Intelligence*. Springer, 1997.
20. R. Reiter. Equality and domain closure in first order databases. *Journal of ACM*, 27:235–249, 1980.
21. R. Rosati. On the decidability and complexity of integrating ontologies and rules. *Journal of Web Semantics*, 2005. to appear.
22. C. Rouveirol and V. Ventos. Towards Learning in CARIN-$\mathcal{ALN}$. In J. Cussens and A. Frisch, editors, *Inductive Logic Programming*, volume 1866 of *Lecture Notes in Artificial Intelligence*, pages 191–208. Springer, 2000.
23. M. Schmidt-Schauss and G. Smolka. Attributive concept descriptions with complements. *Artificial Intelligence*, 48(1):1–26, 1991.
24. G. Semeraro, F. Esposito, D. Malerba, N. Fanizzi, and S. Ferilli. A logic framework for the incremental inductive synthesis of Datalog theories. In N.E. Fuchs, editor, *Proceedings of 7th International Workshop on Logic Program Synthesis and Transformation*, volume 1463 of *Lecture Notes in Computer Science*, pages 300–321. Springer, 1998.