Enabling Data Mining Systems to Semantic Web Applications

Francesca A. Lisi
Dipartimento di Informatica, Università degli Studi di Bari,
Via E. Orabona 4, I-70125 Bari, Italy
Email: lisi@di.uniba.it

***Abstract*— Semantic Web Mining can be considered as Data Mining (DM) for/from the Semantic Web. Current DM systems could serve the purpose of Semantic Web Mining if they were more compliant with, e.g., the standards of representation for ontologies and rules in the Semantic Web and/or interoperable with well-established tools for Ontological Engineering (OE) that support these standards. In this paper we present a middleware, $\mathcal{SW}$ING, that integrates the DM system $\mathcal{AL}$-QUIN and the OE tool Protégé-2000 in order to enable $\mathcal{AL}$-QUIN to Semantic Web applications. This showcase suggests a methodology for building Semantic Web Mining systems.**

I. INTRODUCTION

Data Mining (DM) is an application area arisen in the 1990s at the intersection of several different research fields, notably Statistics, Machine Learning and Databases, as soon as developments in sensing, communications and storage technologies made it possible to collect and store large collections of scientific and commercial data [1]. The abilities to analyze such data sets had not developed as fast. Research in DM can be loosely defined as the study of methods, techniques and algorithms for finding models or patterns that are interesting or valuable in large data sets. The space of patterns if often infinite, and the enumeration of patterns involves some form of search in one such space. Practical computational constraints place severe limits on the subspace that can be explored by a data mining algorithm. The goal of DM is either *prediction* or *description*. Prediction involves using some variables or fields in the database to predict unknown or future values of other variables of interest. Description focuses on finding human-interpretable patterns describing data. Among descriptive tasks, data summarization aims at the extraction of compact patterns that describe subsets of data. There are two classes of methods which represent taking horizontal (cases) and vertical (fields) slices of the data. In the former, one would like to produce summaries of subsets, e.g. producing sufficient statistics or logical conditions that hold for subsets. In the latter case, one would like to describe relations between fields. This class of methods is distinguished from the above in that rather than predicting the value of a specified field (e.g., classification) or grouping cases together (e.g. clustering) the goal is to find relations between fields. One common output of this vertical data summarization is called *frequent (association) patterns*. These patterns state that certain combinations of values occur in a given database with a support greater than a user-defined threshold. The system $\mathcal{AL}$-QUIN [2] supports the DM task of frequent pattern discovery [3]. It implements a framework for learning Semantic Web rules [4] which adopts $\mathcal{AL}$-log [5] as the Knowledge Representation and Reasoning (KR&R) setting and Inductive Logic Programming (ILP) [6] as the methodological apparatus.

Semantic Web Mining [7] is a new application area which aims at combining the two areas of Semantic Web [8] and Web Mining [9] from a twofold perspective. On one hand, the new semantic structures in the Web can be exploited to improve the results of Web Mining. On the other hand, the results of Web Mining can be used for building the Semantic Web. Most work in Semantic Web Mining simply extends previous work to the new application context. E.g., Maedche and Staab [10] apply a well-known algorithm for association rule mining to discover conceptual relations from text. Indeed, we argue that Semantic Web Mining can be considered as DM for/from the Semantic Web. Current DM systems could serve the purpose of Semantic Web Mining if they were more compliant with, e.g., the standards of representation for ontologies and rules in the Semantic Web and/or interoperable with well-established tools for Ontological Engineering (OE) [11], e.g. Protégé-2000 [12], that support these standards.

In this paper we present a middleware, $\mathcal{SW}$ING, that integrates $\mathcal{AL}$-QUIN and Protégé-2000 in order to enable Semantic Web applications of $\mathcal{AL}$-QUIN. This solution suggests a methodology for building Semantic Web Mining systems, i.e. the upgrade of existing DM systems with facilities provided by interoperable OE tools.

The paper is structured as follows. Section II and III briefly introduce $\mathcal{AL}$-QUIN and Protégé-2000 respectively. Section IV presents the middleware $\mathcal{SW}$ING. Section V draws conclusions and outlines directions of future work.

II. THE DM SYSTEM $\mathcal{AL}$-QUIN

The system $\mathcal{AL}$-QUIN [2] (a previous version is described in [13]) supports a variant of the DM task of frequent pattern discovery. In DM a *pattern* is considered as an intensional description (expressed in a given language $\mathcal{L}$) of a subset of $\mathbf{r}$. The *support* of a pattern is the relative frequency of the pattern within $\mathbf{r}$ and is computed with the evaluation function *supp*. The task of *frequent pattern discovery* aims at the extraction of all *frequent* patterns, i.e. all patterns whose support exceeds a user-defined threshold of *minimum support*. The blueprint of most algorithms for frequent pattern discovery

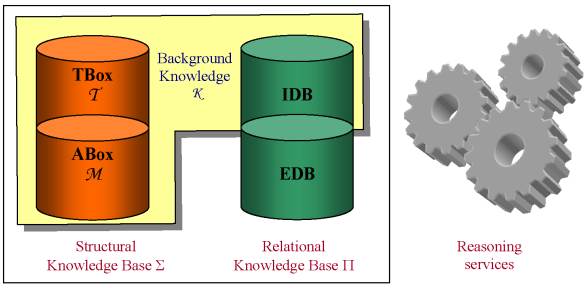

Fig. 1. Organization of the hybrid knowledge bases used in $\mathcal{AL}$-QUIN.

is the *levelwise search* [3]. 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. $supp$, 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 $\succeq$.

The variant of the frequent pattern discovery problem which is solved by $\mathcal{AL}$-QUIN takes concept hierarchies into account during the discovery process [14], thus yielding descriptions of a data set **r** at multiple granularity levels up to a maximum level $maxG$. More formally, given

- a data set **r** including a taxonomy $\mathcal{T}$ where a reference concept C_{ref} and task-relevant concepts are designated,
- a multi-grained language $\{\mathcal{L}^l\}_{1 \leq l \leq maxG}$ of patterns
- a set $\{minsup^l\}_{1 \leq l \leq maxG}$ of minimum 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 **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. Note that a pattern Q is considered to be an ancestor of P if it is a coarser-grained version of P.

In $\mathcal{AL}$-QUIN ($\mathcal{AL}$-log QUery INduction) the data set **r** is represented as an $\mathcal{AL}$-log knowledge base $\mathcal{B}$ and structured as illustrated in Figure 1. The structural subsystem Σ is based on $\mathcal{ALC}$ [15] and allows for the specification of knowledge in terms of classes (*concepts*), binary relations between classes (*roles*), and instances (*individuals*). In particular, the TBox $\mathcal{T}$ contains is-a relations between concepts (*axioms*) whereas the ABox $\mathcal{M}$ contains instance-of relations between individuals (resp. couples of individuals) and concepts (resp. roles) (*assertions*). The relational subsystem Π is based on an extended form of DATALOG [16] that is obtained by using $\mathcal{ALC}$ concept assertions essentially as type constraints on variables. The portion $\mathcal{K}$ of $\mathcal{B}$ which encompasses the whole Σ and the intensional part (IDB) of Π is considered as *background knowledge*. The extensional part of Π is partitioned into portions $\mathcal{A}_i$ each of which refers to an individual a_i of C_{ref}. The link between $\mathcal{A}_i$ and a_i is represented with the DATALOG literal $q(a_i)$. The pair $(q(a_i), \mathcal{A}_i)$ is called *observation*.

The language $\mathcal{L} = \{\mathcal{L}^l\}_{1 \leq l \leq maxG}$ of patterns allows for the generation of $\mathcal{AL}$-log unary conjunctive queries, called $\mathcal{O}$-queries. Given a reference concept C_{ref}, an *$\mathcal{O}$-query* Q to an $\mathcal{AL}$-log knowledge base $\mathcal{B}$ is a (linked and connected)[1] 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*. Note that α_j, $1 \leq j \leq m$, is a DATALOG literal whereas γ_k, $1 \leq k \leq n$, is an assertion that constrains a variable already appearing in any of the α_j's to vary in the range of individuals of a concept defined in $\mathcal{B}$. The $\mathcal{O}$-query

$$Q_t = q(X) \leftarrow \& X : C_{ref}$$

is called *trivial* for $\mathcal{L}$ because it only contains the constraint for the *distinguished variable* X. Furthermore the language $\mathcal{L}$ is *multi-grained*, i.e. it contains expressions at multiple levels of description granularity. Indeed it is implicitly defined by a *declarative bias specification* which consists of a finite alphabet $\mathcal{A}$ of DATALOG predicate names and finite alphabets Γ^l (one for each level l of description granularity) of $\mathcal{ALC}$ concept names. Note that the α_i's are taken from $\mathcal{A}$ and γ_j's are taken from Γ^l. We impose $\mathcal{L}$ to be finite by specifying some bounds, mainly $maxD$ for the maximum depth of search and $maxG$ for the maximum level of granularity.

The *support* of an $\mathcal{O}$-query $Q \in \mathcal{L}^l$ w.r.t an $\mathcal{AL}$-log knowledge base $\mathcal{B}$ is defined as

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

where Q_t is the trivial $\mathcal{O}$-query for $\mathcal{L}$. The computation of support relies on query answering in $\mathcal{AL}$-log. Indeed, an *answer* to an $\mathcal{O}$-query Q is a ground substitution θ for the distinguished variable of Q. 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. $\mathcal{B}_i = \mathcal{K} \cup \mathcal{A}_i$.

The system $\mathcal{AL}$-QUIN implements the aforementioned levelwise search method for frequent pattern discovery. In particular, candidate patterns of a certain level k (called *k-patterns*) are obtained by refinement of the frequent patterns discovered at level $k-1$. In $\mathcal{AL}$-QUIN patterns are ordered according to $\mathcal{B}$-subsumption (which has been proved to fulfill

[1] For the definition of linkedness and connectedness see [6].

the abovementioned condition of monotonicity [13]). The search starts from the most general pattern in $\mathcal{L}$ and iterates through the generation-evaluation cycle for a number of times that is bounded with respect to both the granularity level l ($maxG$) and the depth level k ($maxD$).

Since $\mathcal{AL}$-QUIN is implemented with Prolog, the internal representation language in $\mathcal{AL}$-QUIN is a kind of DATALOG^{OI} [17], i.e. the subset of DATALOG$^{\neq}$ equipped with an equational theory that consists of the axioms of Clark's Equality Theory augmented with one rewriting rule that adds *inequality atoms* $s \neq t$ to any $P \in \mathcal{L}$ for each pair (s,t) of distinct terms occurring in P. Note that concept assertions are rendered as *membership atoms*, e.g. $a : C$ becomes $c\_C(a)$.

III. THE OE TOOL PROTÉGÉ-2000

Protégé-2000[2] [18] is the latest version of the Protégé line of tools, created by the Stanford Medical Informatics (SMI) group at Stanford University, USA. It has a community of thousands of users. Although the development of Protégé has historically been mainly driven by biomedical applications, the system is domain-independent and has been successfully used for many other application areas as well. Protégé-2000 is a Java-based standalone application to be installed and run in a local computer. The core of this application is the ontology editor. Like most other modeling tools, the architecture of Protégé-2000 is cleanly separated into a model part and a view part. Protégé-2000's model is the internal representation mechanism for ontologies and knowledge bases. Protégé-2000's view components provide a Graphical User Interface (GUI) to display and manipulate the underlying model.

Protégé-2000's model is based on a simple yet flexible metamodel [12], which is comparable to object-oriented and frame-based systems. It basically can represent ontologies consisting of classes, properties (*slots*), property characteristics (*facets* and *constraints*), and instances. Protégé-2000 provides an open Java API to query and manipulate models. An important strength of Protégé-2000 is that the Protégé-2000 metamodel itself is a Protégé-2000 ontology, with classes that represent classes, properties, and so on. For example, the default class in the Protege base system is called :STANDARD-CLASS, and has properties such as :NAME and :DIRECT-SUPERCLASSES. This structure of the metamodel enables easy extension and adaption to other representations.

Using the views of Protégé-2000's GUI, ontology designers basically create classes, assign properties to the classes, and then restrict the properties facets at certain classes. Using the resulting ontologies, Protégé-2000 is able to automatically generate user interfaces that support the creation of individuals (instances). For each class in the ontology, the system creates one form with editing components (*widgets*) for each property of the class. For example, for properties that can take single string values, the system would by default provide a text field widget. The generated forms can be further customized with Protégé-2000's form editor, where users can select alternative user interface widgets for their project. The user interface consists of panels (*tabs*) for editing classes, properties, forms and instances.

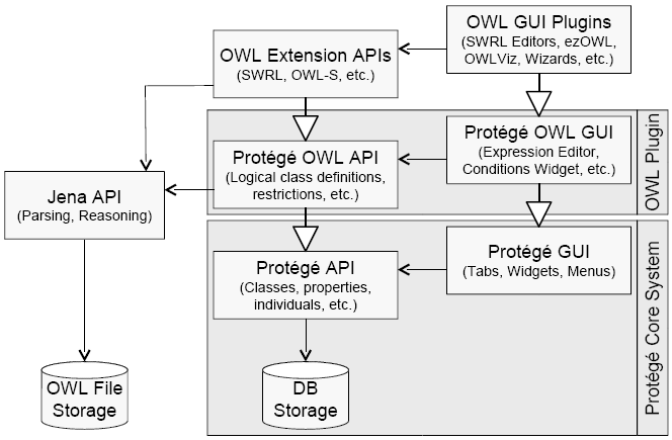

Fig. 2. Architecture of the OWL Plugin for Protégé-2000.

Protégé-2000 has an extensible architecture, i.e. an architecture that allows special-purpose extensions (aka *plug-ins*) to be easily integrated. These extensions usually perform functions not provided by the Protégé-2000 standard distribution (other types of visualization, new import and export formats, etc.), implement applications that use Protégé-2000 ontologies, or allow configuring the ontology editor. Most of these plug-ins are available in the Protégé-2000 Plug-in Library, where contributions from many different research groups can be found. One of the most popular in this library is the OWL Plugin [19].

As illustrated in Figure 2, the **OWL Plugin** extends the Protégé-2000 model and its API with classes to represent the OWL[3] specification. In particular it supports RDF(S), OWL Lite, OWL DL (except for anonymous global class axioms, which need to be given a name by the user) and significant parts of OWL Full (including metaclasses). The OWL API basically encapsulates the internal mapping and thus shields the user from error-prone low-level access. Furthermore the OWL Plugin provides a comprehensive mapping between its extended API and the standard OWL parsing library Jena[4]. The presence of a secondary representation of an OWL ontology in terms of Jena objects means that the user is able to invoke arbitrary Jena-based services such as interfaces to classifiers, query languages, or visualization tools permanently. Based on the above mentioned metamodel and API extensions, the OWL Plugin provides several custom-tailored GUI components for OWL. Also it can directly access DL reasoners such as RACER [20]. Finally it can be further extended, e.g. to support OWL-based languages like SWRL[5].

[2]The distribution of interest to this work is 3.0 (February 2005), freely available at `http://protege.stanford.edu/` under the Mozilla open-source license.

[3]`http://www.w3.org/2004/OWL/`
[4]`http://jena.sourceforge.net`
[5]`http://www.w3.org/Submission/SWRL/`

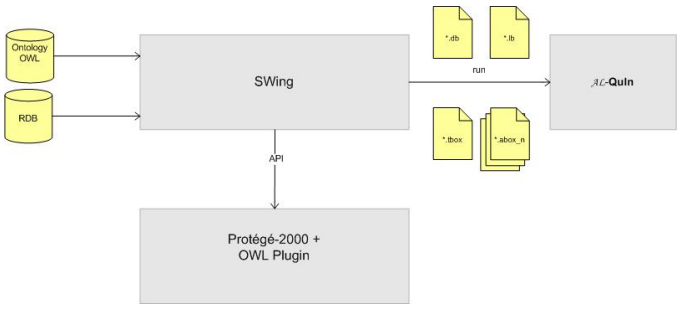

Fig. 3. Architecture and I/O of $\mathcal{SW}$ING.

IV. Enabling $\mathcal{AL}$-QuIn to Semantic Web applications with Protégé-2000

To enable $\mathcal{AL}$-QuIn to Semantic Web applications we have developed a software component, $\mathcal{SW}$ING, that assists users of $\mathcal{AL}$-QuIn in the design of Semantic Web Mining sessions. As illustrated in Figure 3, $\mathcal{SW}$ING is a middleware because it interoperates via API with the OWL Plugin for Protégé-2000 to benefit from its facilities for browsing and reasoning on OWL ontologies.

Example IV.1. *The screenshots reported in Figure 4, 5, 6 and 7 refer to a Semantic Web Mining session with* $\mathcal{SW}$ING *for the task of finding frequent patterns in the on-line CIA World Fact Book*[6] *(data set) that describe Middle East countries (reference concept) w.r.t. the religions believed and the languages spoken (task-relevant concepts) at three levels of granularity (*$maxG = 3$*). To this aim we define* $\mathcal{L}_{CIA}$ *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*

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

of $\mathcal{ALC}$ *concept names for* $1 \leq l \leq 3$*, up to* $maxD = 5$*. Examples of* $\mathcal{O}$*-queries in* $\mathcal{L}_{CIA}$ *are:*

$Q_t =$ `q(X)` $\leftarrow$ & `X:MiddleEastCountry`
$Q_1 =$ `q(X)` $\leftarrow$ `speaks(X,Y)` &
`X:MiddleEastCountry, Y:Language`
$Q_2 =$ `q(X)` $\leftarrow$ `speaks(X,Y)` &
`X:MiddleEastCountry, Y:IndoEuropeanLanguage`
$Q_3 =$ `q(X)` $\leftarrow$ `believes(X,Y)`&
`X:MiddleEastCountry, Y:MuslimReligion`

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

Minimum support thresholds are set to the following values: $minsup^1 = 20\%$*,* $minsup^2 = 13\%$*, and* $minsup^3 = 10\%$*. After* $maxD = 5$ *search stages,* $\mathcal{AL}$-QuIn *returns 53 frequent patterns out of 99 candidate patterns compliant with the parameter settings. One of these findings is the pattern* Q_2 *which turns out to be frequent because it has support* $supp(Q_2, \mathcal{B}_{CIA}) = 13\%$ $(\geq minsup^2)$*. This has to be read as '13 % of Middle East countries speak an Indoeuropean language'.* ◊

[6] http://www.odci.gov/cia/publications/factbook/

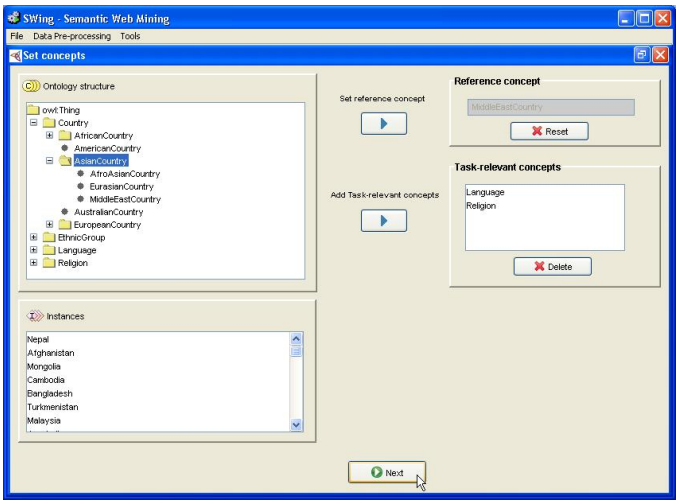

Fig. 4. $\mathcal{SW}$ING: step of concept selection.

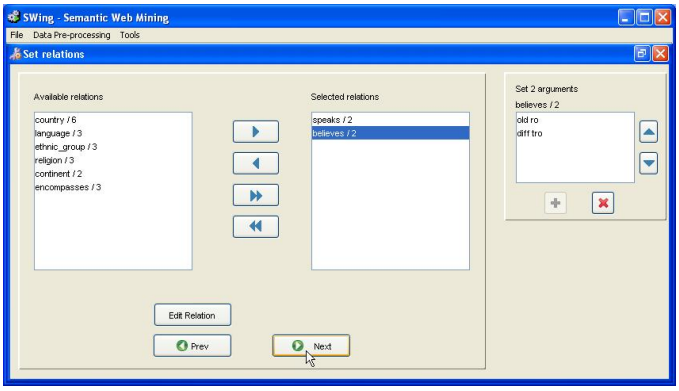

Fig. 5. $\mathcal{SW}$ING: step of relation selection.

A wizard provides guidance for the selection of the (hybrid) data set to be mined, the selection of the reference concept and the task-relevant concepts (see Figure 4), the selection of the relations - among the ones appearing in the relational component of the data set chosen or derived from them - with which the task-relevant concepts can be linked to the reference concept in the patterns to be discovered (see Figure 5 and 6), the setting of minimum support thresholds for each level of description granularity and of several other parameters required by $\mathcal{AL}$-QuIn. These user preferences are collected in a file (see ouput file **.lb* in Figure 3) that is shown in preview to the user at the end of the assisted procedure for confirmation (see Figure 7).

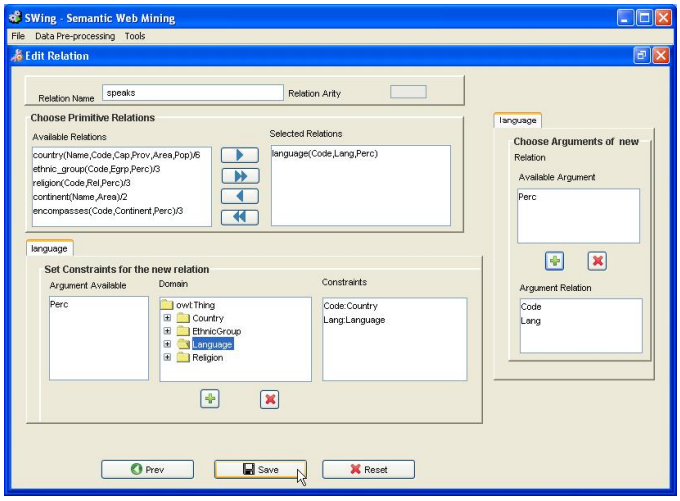

Fig. 6. $\mathcal{SW}$ING: editing of derived relations.

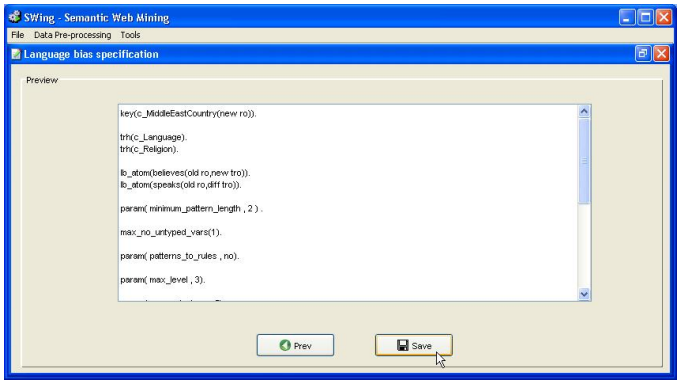

Fig. 7. $\mathcal{SW}$ING: preview of the language bias specification.

A. A closer look to the I/O

The input to $\mathcal{SW}$ING is a hybrid knowledge base that consists of an ontological data source - expressed as a OWL file - and a relational data source - also available on the Web - integrated with each other.

Example IV.2. *The knowledge base $\mathcal{B}_{CIA}$ for the Semantic Web Mining session of Example IV.1 integrates an OWL ontology (file* `cia_exp1.owl`*) with a* DATALOG *database (file* `cia_exp1.edb`*) containing facts*[7] *extracted from the on-line 1996 CIA World Fact Book. The OWL ontology*[8] *contains axioms such as*

$AsianCountry \sqsubset Country.$
$MiddleEastEthnicGroup \sqsubset EthnicGroup.$
$MiddleEastCountry \equiv$
$\quad AsianCountry \sqcap \exists Hosts.MiddleEastEthnicGroup.$

[7] `http://www.dbis.informatik.uni-goettingen.de/Mondial/mondial-rel-facts.flp`

[8] In the following we shall use the corresponding DL notation

$IndoEuropeanLanguage \sqsubset Language.$
$IndoIranianLanguage \sqsubset IndoEuropeanLanguage.$
$MonotheisticReligion \sqsubset Religion.$
$MuslimReligion \sqsubset MonotheisticReligion.$

and membership assertions such as

```
'IR':AsianCountry.
'Arab':MiddleEastEthnicGroup.
<'IR','Arab'>:Hosts.
'Persian':IndoIranianLanguage.
'ShiaMuslim':MuslimReligion.
'SunniMuslim':MuslimReligion.
```

that define taxonomies for the concepts `Country`*,* `EthnicGroup`*,* `Language` *and* `Religion`*. Note that Middle East countries (concept* `MiddleEastCountry`*) have been defined as Asian countries that host at least one Middle Eastern ethnic group. In particular, Iran (*`'IR'`*) is classified as Middle East country.*

Since C_{ref}=`MiddleEastCountry`*, the* DATALOG *database is partitioned according to the individuals of* `MiddleEastCountry`*. In particular, the observation $(q(\texttt{'IR'}), \mathcal{A}_{IR})$ contains* DATALOG *facts such as*

```
language('IR','Persian',58).
religion('IR','ShiaMuslim',89).
religion('IR','SunniMuslim',10).
```

concerning the individual `'IR'`*.* ◊

The output file **.db* contains the input DATALOG database eventually enriched with an intensional part. The editing of derived relations (see Figure 6) is accessible from the step of relation selection (see Figure 5).

Example IV.3. *The output* DATALOG *database* `cia_exp1.db` *for Example IV.1 enriches the input* DATALOG *database* `cia_exp1.edb` *with the following two clauses:*

```
speaks(Code, Lang)← language(Code,Lang,Perc),
                    c_Country(Code), c_Language(Lang).
believes(Code, Rel)←religion(Code,Rel,Perc),
                    c_Country(Code), c_Religion(Rel).
```

that define views on the relations `language` *and* `religion` *respectively. Note that they correspond to the constrained* DATALOG *clauses*

```
speaks(Code, Lang)← language(Code,Lang,Perc) &
                    Code:Country, Lang:Language.
believes(Code, Rel)←religion(Code,Rel,Perc) &
                    Code:Country, Rel:Religion.
```

and represent the intensional part of Π_{CIA}. ◊

The output file **.lb* contains the declarative bias specification for the language of patterns and other directives.

Example IV.4. *With reference to Example IV.1, the content of* `cia_exp1.lb` *(see Figure 7) defines - among the other things - the language $\mathcal{L}_{CIA}$ of patterns. In particular the first*

5 directives define the reference concept, the task-relevant concepts and and the relations between concepts. ◊

The output files **.abox_n* and **.tbox* are the side effect of the step of concept selection as illustrated in the next section. Note that these files together with the intensional part of the **.db* file form the background knowledge $\mathcal{K}$ for $\mathcal{AL}$-QUIN.

B. A look inside the step of concept selection

The step of concept selection deserves further remarks because it actually exploits the services offered by Protégé-2000. Indeed it also triggers some supplementary computation aimed at making a OWL background knowledge Σ usable by $\mathcal{AL}$-QUIN. To achieve this goal, it supplies the following functionalities:

- levelwise retrieval w.r.t. Σ
- translation of both (asserted and derived) concept assertions and subsumption axioms of Σ to DATALOG^{OI} facts

The latter relies on the former, meaning that the results of the levelwise retrieval are exported to DATALOG^{OI} (see output files **.abox_n* and **.tbox* in Figure 3). The *retrieval* problem is known in DLs literature as the problem of retrieving all the individuals of a concept C [21]. Here, the retrieval is called *levelwise* because it follows the layering of $\mathcal{T}$: individuals of concepts belonging to the l-th layer $\mathcal{T}^l$ of $\mathcal{T}$ are retrieved all together.

Example IV.5. *The* DATALOG^{OI} *rewriting of the concept assertions derived for* $\mathcal{T}^2$ *produces facts like:*

```
c_AfroAsiaticLanguage('Arabic').
...
c_IndoEuropeanLanguage('Persian').
...
c_UralAltaicLanguage('Kazak').
...
c_MonotheisticReligion('ShiaMuslim').
c_MonotheisticReligion('SunniMuslim').
...
c_PolytheisticReligion('Druze').
...
```

that are stored in the file `cia_exp1.abox_2`.

The file `cia_exp1.tbox` *contains a* DATALOG^{OI} *rewriting of the taxonomic relations of* $\mathcal{T}$ *such as:*

```
hierarchy(c_Language,1,null,[c_Language]).
hierarchy(c_Religion,1,null,[c_Religion]).
```

for the layer $\mathcal{T}^1$ *and*

```
hierarchy(c_Language,2,c_Language,
   [c_AfroAsiaticLanguage, c_IndoEuropeanLanguage, ...]).
hierarchy(c_Religion,2,c_Religion,
   [c_MonotheisticReligion, c_PolytheisticReligion]).
```

for the layer $\mathcal{T}^2$ *and*

```
hierarchy(c_Language,3,c_AfroAsiaticLanguage,
   [c_AfroAsiaticLanguage]).
...
hierarchy(c_Language,3,c_IndoEuropeanLanguage,
   [c_IndoIranianLanguage, c_SlavicLanguage]).
hierarchy(c_Language,3,c_UralAltaicLanguage,
   [c_TurkicLanguage]).
hierarchy(c_Religion,3,c_MonotheisticReligion,
   [c_ChristianReligion, c_JewishReligion, c_MuslimReligion]).
```

for the layer $\mathcal{T}^3$. ◊

Note that the translation from OWL to DATALOG^{OI} is possible because we assume that *all* the concepts are named. This means that an equivalence axiom is required for each complex concept in the knowledge base. Equivalence axioms help keeping concept names (used within constrained DATALOG clauses) independent from concept definitions.

V. CONCLUSION

The middleware $\mathcal{SW}$ING supplies several facilities to $\mathcal{AL}$-QUIN, primarily facilities for compiling OWL down to DATALOG. Note that DATALOG is the usual KR&R setting for ILP. In this respect, the pre-processing method proposed by Kietz [22] to enable ILP systems to work within the framework of the hybrid KR&R system CARIN [23] is related to ours but it lacks an application. Analogously, the method proposed in [24] for translating OWL to disjunctive DATALOG is far too general with respect to the specific needs of our application. Rather, the proposal of interfacing existing reasoners to combine ontologies and rules [25] is more similar to ours in the spirit. Furthermore, $\mathcal{SW}$ING follows engineering principles because it promotes the reuse of existing systems ($\mathcal{AL}$-QUIN and Protégé-2000) and the adherence to standards (either normative - see OWL for the Semantic Web - or *de facto* - see DATALOG for ILP). Finally the resulting artifact overcomes the capabilities of the two systems when considered stand-alone. In particular, $\mathcal{AL}$-QUIN was originally conceived to deal with $\mathcal{ALC}$ ontologies. Since OWL is equivalent to $\mathcal{SHIQ}$ [26] and $\mathcal{ALC}$ is a fragment of $\mathcal{SHIQ}$ [21], the middleware $\mathcal{SW}$ING allows $\mathcal{AL}$-QUIN to deal with more expressive ontologies and to face Semantic Web applications.

For the future we plan to extend $\mathcal{SW}$ING with facilities for extracting information from semantic portals and for presenting patterns generated by $\mathcal{AL}$-QUIN.

REFERENCES

[1] U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, Eds., *Advances in Knowledge Discovery and Data Mining*. AAAI Press/The MIT Press, 1996.

[2] F. Lisi and F. Esposito, "ILP Meets Knowledge Engineering: A Case Study." in *Inductive Logic Programming*, ser. Lecture Notes in Artificial Intelligence, S. Kramer and B. Pfahringer, Eds. Springer, 2005, vol. 3625, pp. 209–226.

[3] H. Mannila and H. Toivonen, "Levelwise search and borders of theories in knowledge discovery," *Data Mining and Knowledge Discovery*, vol. 1, no. 3, pp. 241–258, 1997.

[4] F. Lisi and F. Esposito, "An ILP Perspective on the Semantic Web," in *Semantic Web Applications and Perspectives 2005*, P. Bouquet and G. Tumarello, Eds. CEUR Workshop Proceedings, 2005, http://ceur-ws.org/Vol-166/.

[5] F. Donini, M. Lenzerini, D. Nardi, and A. Schaerf, "$\mathcal{AL}$-log: Integrating Datalog and Description Logics," *Journal of Intelligent Information Systems*, vol. 10, no. 3, pp. 227–252, 1998.

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

[7] G. Stumme, A. Hotho, and B. Berendt, "Semantic Web Mining: State of the art and future directions," *Journal of Web Semantics*, vol. 4, no. 2, pp. 124–143, 2006.

[8] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web," *Scientific American*, vol. May, 2001.

[9] R. Kosala and H. Blockeel, "Web Mining Research: A Survey," *SIGKDD: SIGKDD Explorations: Newsletter of the Special Interest Group (SIG) on Knowledge Discovery & Data Mining, ACM*, vol. 2, 2000. [Online]. Available: citeseer.ist.psu.edu/kosala00web.html

[10] A. Maedche and S. Staab, "Discovering Conceptual Relations from Text," in *Proceedings of the 14th European Conference on Artificial Intelligence*, W. Horn, Ed. IOS Press, 2000, pp. 321–325.

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

[12] N. F. Noy, R. Fergerson, and M. Musen, "The Knowledge Model of Protégé-2000: Combining Interoperability and Flexibility." in *Knowledge Acquisition, Modeling and Management*, ser. Lecture Notes in Computer Science, R. Dieng and O. Corby, Eds. Springer, 2000, vol. 1937, pp. 17–32.

[13] F. Lisi and D. Malerba, "Inducing Multi-Level Association Rules from Multiple Relations," *Machine Learning*, vol. 55, pp. 175–210, 2004.

[14] J. Han and Y. Fu, "Mining multiple-level association rules in large databases," *IEEE Transactions on Knowledge and Data Engineering*, vol. 11, no. 5, 1999.

[15] M. Schmidt-Schauss and G. Smolka, "Attributive concept descriptions with complements," *Artificial Intelligence*, vol. 48, no. 1, pp. 1–26, 1991.

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

[17] G. Semeraro, F. Esposito, D. Malerba, N. Fanizzi, and S. Ferilli, "A logic framework for the incremental inductive synthesis of Datalog theories," in *Proceedings of 7th International Workshop on Logic Program Synthesis and Transformation*, ser. Lecture Notes in Computer Science, N. Fuchs, Ed. Springer, 1998, vol. 1463, pp. 300–321.

[18] J. Gennari, M. Musen, R. Fergerson, W. Grosso, M. Crubézy, H. Eriksson, N. F. Noy, and S. W. Tu, "The evolution of Protégé: An environment for knowledge-based systems development." *International Journal of Human-Computer Studies*, vol. 58, no. 1, pp. 89–123, 2003.

[19] H. Knublauch, M. Musen, and A. Rector, "Editing Description Logic Ontologies with the Protégé OWL Plugin." in *Proceedings of the 2004 International Workshop on Description Logics (DL2004)*, ser. CEUR Workshop Proceedings, V. Haarslev and R. Möller, Eds., vol. 104, 2004.

[20] V. Haarslev and R. Möller, "Description of the RACER System and its Applications." in *Working Notes of the 2001 International Description Logics Workshop (DL-2001)*, ser. CEUR Workshop Proceedings, C. Goble, D. McGuinness, R. Möller, and P. Patel-Schneider, Eds., vol. 49, 2001.

[21] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, Eds., *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, 2003.

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

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

[24] U. Hustadt, B. Motik, and U. Sattler, "Reducing $\mathcal{SHIQ}$-description logic to disjunctive datalog programs." in *Principles of Knowledge Representation and Reasoning: Proceedings of the Ninth International Conference (KR2004)*, D. Dubois, C. Welty, and M.-A. Williams, Eds. AAAI Press, 2004, pp. 152–162.

[25] U. Assmann, J. Henriksson, and J.Maluszynski, "Combining safe rules and ontologies by interfacing of reasoners." in *Principles and Practice of Semantic Web Reasoning*, ser. Lecture Notes in Computer Science, J. Alferes, J. Bailey, W. May, and U. Schwertel, Eds. Springer, 2006, vol. 4187, pp. 33–47.

[26] I. Horrocks, P. Patel-Schneider, and F. van Harmelen, "From $\mathcal{SHIQ}$ and RDF to OWL: The making of a web ontology language," *Journal of Web Semantics*, vol. 1, no. 1, pp. 7–26, 2003.