Query rewriting using views in presence of value constraints

Hélène Jaudoin, Jean-Marc Petit, Christophe Rey, Michel Schneider, Farouk Toumani

Query rewriting using views in presence of value constraints

H. Jaudoin[†‡], J-M. Petit[‡], C. Rey[‡], M. Schneider[‡], F. Toumani[‡]

[†] Cemagref, France (helene.jaudoin@clermont.cemagref.fr)

[‡] LIMOS, Blaise Pascal University, France ({rey,ftoumani,jmpetit,michel.schneider}@isima.fr)

1 Introduction-Motivation

The problem of query rewriting using views can be stated as follows: given a set of views $\mathcal{V}$ and a query Q, both the views and the query are defined over the same global/mediated schema, the purpose is to rewrite Q into a query expression that uses only the views in $\mathcal{V}$ and is *maximally contained* in Q [7].

We are interested in query rewriting as a technique for answering queries in data integration systems that follow a LAV (Local As View)[1] mediation approach [7]. In such systems, queries formulated in terms of a mediated schema are reformulated (rewritten) into queries expressed in terms of views. In this paper, we investigate the problem of query rewriting using views in presence of value constraints. Our approach uses Description Logics (DL) to describe views, queries, and mediated schema. We use the `One-Of` constructor to express *value constraints*. This kind of constraints have been shown very useful in many practical applications such as value integrity constraint checking or for expressing some forms of incomplete information [5]. Moreover they can also be very useful to express queries. For example, a typical query Q in our application can ask for cultural parcels of commune number 23200 or 23400 that have received pesticides during years 2001 or 2002. Value constraints provide valuable information to identify when a view is not useful for answering a query. Consequently, capturing and exploiting value constraints improves query processing cost in two points: *(i)* it reduces the number of candidates to be considered during query rewriting, and *(ii)* it prunes the set of sources accessed to answer a query, thereby reducing the communication cost.

This work is part of an ongoing research project[2], which aims at providing a scalable and flexible data sharing infrastructure to enable heterogeneous and autonomous agricultural data sources to cooperate in order to collect and to analyze agricultural practices and to verify compliance with respect to national and european government regulations.

We investigate the problem of query rewriting using views in the setting of the description logics $\mathcal{ALN}$ augmented with the `One-Of` constructor on *values*, noted $\mathcal{O}_v$. We call this language $\mathcal{ALN}(\mathcal{O}_v)$. In this setting, we formalize the

[1] In the LAV approach, sources are defined as views over the mediated schema [4, 7].

[2] This is a collaborative project with the Cemagref (http://www.cemagref.fr/English/) .

query rewriting problem and give its computational complexity. To cope with a large number of information sources, we point out how our query rewriting problem can be mapped into a theoretical framework of Knowledge Discovery in Databases (KDD) [11]. The KDD setting not only supplies scalable solutions that help solving our rewriting problem but also enables to reuse/adapt existing algorithms to implement one of the most costly steps of our approach. Proofs and more details are given in our technical report[8].

2 $\mathcal{ALN}(\mathcal{O}_v)$ description logic

$\mathcal{ALN}(\mathcal{O}_v)$ can be viewed as a sub-language of CLASSIC [9] obtained by augmenting the language $\mathcal{ALN}$ with the `One-Of` constructor on CLASSIC *host individuals*. More precisely, in addition to usual $\mathcal{ALN}$ descriptions, $\mathcal{ALN}(\mathcal{O}_v)$ allows descriptions of the form $\forall R.\{o_1,\ldots,o_n\}$, where $\{o_1,\ldots,o_n\}$ denotes the `One-Of` constructor and the o_i's denote CLASSIC host individuals. In the sequel, host individuals are called *values*. In the spirit of [12], we assume two disjoint sets of roles: $\mathcal{R}_c$, which denotes roles whose range is the set of usual individuals, and $\mathcal{R}_v$, which denotes roles whose range is the set of host individuals. Hence, $\mathcal{ALN}(\mathcal{O}_v)$ concept descriptions allow for atomic negations ($\neg A$), concept conjunctions ($C \sqcap D$), number restrictions (($\leq nR$), ($\geq nR$), with $R \in \mathcal{R}_c \cup \mathcal{R}_v$), and values restrictions (($\forall R_c.C$), with $R_c \in \mathcal{R}_c$, or ($\forall R_v.\{o_1,\ldots,o_n\}$), with $R_v \in \mathcal{R}_v$)[3].

The semantics of a concept description is given by an interpretation $\mathcal{I}=(\Delta^{\mathcal{I}},.^{\mathcal{I}})$. In the same spirit of CLASSIC [9], we assume $\Delta^{\mathcal{I}}$ divided into two disjointed domains: δ_c, the set of individuals in the domain, and δ_v, the set of values. A concept is interpreted as a subset of δ_c while a *value* is interpreted as an element of δ_v following the *unique name assumption* (i.e., different values are assigned to different elements of δ_v). The `One-Of` constructor is interpreted as a set of values, i.e., a subset of δ_v. A role $R_c \in \mathcal{R}_c$ is interpreted as a subset of $\delta_c \times \delta_c$ while a role $R_v \in \mathcal{R}_v$ is interpreted as a subset of $\delta_c \times \delta_v$ and, consequently, values cannot have role successors. Semantics of arbitrary descriptions, subsumption ($\sqsubseteq$) and equivalence ($\equiv$) are defined as usual.

We present now a normal form description for $\mathcal{ALN}(\mathcal{O}_v)$ concepts as well as a characterization of subsumption w.r.t. this normal form that are appropriate to deal with our query rewriting problem. In the sequel, we use the letters $A, B, A_1, \ldots$ to denote atomic concepts and $E, E', E_1, \ldots$ to denote finite sets of values. E can be the empty set. P denotes either an atomic concept (A) or its negation ($\neg A$) or a set of values (E) or a number restriction (($\leq nR$) or ($\geq nR$)). C, C' denote arbitrary descriptions. For sake of simplicity, we assume that descriptions of the forms ($\geq 0R$), $\forall R_c.\top_c$ are translated into $\top_c$, while descriptions of the form $C \sqcap \top_c$ are translated into C, where $\top_c$ stands

[3]Note that $\{o_1,\ldots,o_n\}$ is not an $\mathcal{ALN}(\mathcal{O}_v)$ description while $\forall R_v.\{o_1,\ldots,o_n\}$ is.

for top concept of $\mathcal{ALN}(\mathcal{O}_v)$ descriptions. Roughly speaking, the normal form of a concept description C is obtained by first recursively applying on C the rules (1) to (10) of table 1 until no rule can be more applied. Then, the rule (11) is recursively applied on the obtained description. With regard to this normal form, any concept description in $\mathcal{ALN}(\mathcal{O}_v)$ can be transformed into an equivalent description that is a (nonempty) conjunction of descriptions of the form $\forall R_1 \ldots \forall R_m.P$ for $m \geq 0$, where $R_1, \ldots, R_m$ are (not necessarily distinct) roles. The description $\forall R_1 \ldots \forall R_m.P$ is abbreviated by $\forall R_1 \ldots R_m.P$, where $R_1 \ldots R_m$ is considered as a word w over $\mathcal{R}_c \cup \mathcal{R}_v$[4]. When $m = 0$ then it is the empty word ϵ, and $\forall \epsilon.P$ denotes the description P.

(1) $\forall w.C \sqcap \forall w.C' \rightarrow \forall w.(C \sqcap C')$	(2) $\leq 0R \sqcap \forall R.C \rightarrow \leq 0R$
(3) $A \sqcap \neg A \rightarrow \bot$	(4) $(\geq nR) \sqcap (\leq mR) \rightarrow \bot$ if $n > m$
(5) $(\geq nR) \sqcap (\geq mR) \rightarrow (\geq max(n,m)R)$	(6) $(\leq nR) \sqcap (\leq mR) \rightarrow (\leq min(n,m)R)$
(7) $C \sqcap \bot \rightarrow \bot$	(8) $E_1 \sqcap E_2 \rightarrow E$ such that $E = E_1 \cap E_2$
(9) $\forall R_v.E \rightarrow \forall R_v.E \sqcap (\leq kR_v)$, where $\|E\| = k$	(10) $\forall R.\bot \rightarrow \leq 0R$
(11) $\forall w.(C \sqcap C') \rightarrow \forall w.C \sqcap \forall w.C'$	

Table 1: Normalization rules.

In the sequel we use the expression $\forall w.P \in C$ to denote that the normal form of a concept C contains a conjunct of the form $\forall w.P$ in its description. To enable a characterization of subsumption using the normal form introduced above, we reuse the so-called value-restriction sets introduced in [2]. The value-restriction sets of an $\mathcal{ALN}(\mathcal{O}_v)$ concept C are given in table 2.

$V_C(A) = \{w \mid \forall w.A \in C\}$	$V_C((\geq nR)) = \{w \mid \forall w.(\geq mR) \in C \wedge m \geq n\}$
$V_C(\neg A) = \{w \mid \forall w.\neg A \in C\}$	$V_C((\leq nR)) = \{w \mid \forall w.(\leq mR) \in C \wedge m \leq n\}$
$V_C(E) = \{w \mid \forall w.E' \in C$ and $E' \subseteq E\}$	$E(C) = \{w \mid$ there exists a prefix $w' = vr$ of w s.t. $\forall v.(\leq 0r) \in C\}$

Table 2: Value-restriction sets.

Theorem 1 (Subsumption) *Let C, D be two concept descriptions with C and D given in their normal form. $C \sqsubseteq D$ iff one of the following conditions holds: (1) $C \equiv \bot$ or $D \equiv \top_c$, or (2) $w \in V_C(P) \cup E(C)$ for each $\forall w.P$ in D.*

The proof of this theorem (given in [8]) is easily derived from the structural characterization of subsumption in Classic [9].

A primitive terminology (or TBox) $\mathcal{T}$ is a (finite) set of statements of the form $B \sqsubseteq D$, where B is a concept name and D is a concept. B is called a primitive concept. The semantics of TBoxes is defined as usual. In this paper, we assume that terminologies are acyclic. In this case, reasoning with primitive TBoxes can be reduced to reasoning with concept descriptions by first translating primitive concepts into definitions (i.e., statements of the form $B \equiv D \sqcap \overline{B}$, where $\overline{B}$ is a

[4]More precisely, $R_1 \ldots R_{m-1}$ is a word over $\mathcal{R}_c$ and $R_m \in \mathcal{R}_c \cup \mathcal{R}_v$.

new atomic concept) and then unfolding the TBox as described in [3], chapter 2. Consequently, a normal form of a primitive concept can be obtained by applying the normalization process described above on its *unfolded* description.

3 Query Rewriting

Let us first define the query rewriting problem we are considering. As often in data integration contexts, we are interested by maximally-contained rewritings [4, 7]. The definition of such a rewriting depends on a particular query language, which is fixed in our context to be the language $\{\sqcap, \sqcup\}$.

Definition 1 (Maximally-contained Rewriting) *Let $\mathcal{V}$ be a primitive terminology and Q be a concept. Primitive concepts in $\mathcal{V}$ are called views and Q is called a query. A concept Q' is a maximally-contained rewriting of Q using $\mathcal{V}$ w.r.t. the language $\{\sqcap, \sqcup\}$ if: (i) Q' is a concept in $\{\sqcap, \sqcup\}$ that refers only to the views in $\mathcal{V}$ and $Q' \sqsubseteq Q$, and (ii) there does not exist a concept Q_1 in $\{\sqcap, \sqcup\}$ that refers only to the views in $\mathcal{V}$ such that $Q' \sqsubseteq Q_1 \sqsubseteq Q$ and $Q' \not\equiv Q_1$.*

Please note that, assuming views to be primitive concepts in definition 1 allows to capture the *open world assumption* (i.e., sources are assumed to be incomplete [10, 7]). In such a setting, semantics of query answers is usually formalized using the notion of *certain answers* [1].

A maximally-contained rewriting $Q' \equiv C_1 \sqcup ... \sqcup C_n$, where the C_i's are conjunctions of views, is said to be *redundant* if it contains two distinct disjuncts C_i and C_j such that $C_i \sqsubseteq C_j$. It is clear that redundant rewritings are not worth considering for computing query answers. This is why, in the rest of this paper, we focus our attention on the problem of computing non redundant maximally-contained rewritings of a given query Q using a terminology $\mathcal{V}$. For simplicity, we use the term maximally-contained rewritings to denote non redundant maximally-contained rewritings. The following lemma shows how a (non redundant) maximally-contained rewriting of a query Q using $\mathcal{V}$ can be obtained from its maximally-contained *conjunctive* rewritings (i.e., those rewritings of Q obtained by using as query language $\{\sqcap\}$ instead of $\{\sqcap, \sqcup\}$ in definition 1).

Lemma 1 *Let $\mathcal{V}$ be a primitive terminology and Q be a query. Let $\{Q_1, \ldots, Q_n\}$ be the set of all the maximally-contained conjunctive rewritings of Q using $\mathcal{V}$. Then, Q' is a maximally-contained rewriting of Q using $\mathcal{V}$ iff $Q' \equiv \sqcup_{i=1}^{n} Q_i$.*

Note that from this lemma, if a maximally-contained rewriting of Q exists, then it is unique. We focus now on the problem of computing all the maximally-contained conjunctive rewritings of Q using views $\mathcal{V}$. We note this problem $rewrite(\mathcal{V}, Q)$, where Q and the views are assumed to be unfolded and given in their normal forms. A naïve approach to solve a $rewrite(\mathcal{V}, Q)$ problem would then consist in computing every conjunction of views of size 1 to $|\mathcal{V}|$, and testing

whether these conjunctions are maximally-contained conjunctive rewritings of Q. The following lemma points out an interesting property of maximally-contained conjunctive rewritings that will be useful to reduce the search space.

Lemma 2 *$Q' \equiv \sqcap_{i=1}^{n} V_i$ is a maximally-contained conjunctive rewriting of Q using views $\mathcal{V}$ iff for any concept Q'' obtained by removing from Q' a conjunct V_j, with $j \in [1, n]$, we have $Q'' \not\sqsubseteq Q$.*

This lemma says that a maximally-contained conjunctive rewriting is necessarily made of a minimal, w.r.t. set inclusion, subset of $\mathcal{V}$ such that the conjunction of its elements is subsumed by Q. The search space of $rewrite(\mathcal{V}, Q)$ solutions can be further reduced by providing an upper bound on the size of maximally-contained conjunctive rewritings (i.e., the number of views that appear in the rewriting). An upper bound that applies in our setting is given below, where n denotes the number of conjuncts in Q, l the cardinality of the largest set of values that appears in the views or in the query Q, and p the maximal depth[5] of the conjuncts inside the views or the query.

Theorem 2 *Maximally-contained conjunctive rewritings of Q using views $\mathcal{V}$ are made of a conjunction of at most $n * (l + p + 2)$ views from $\mathcal{V}$.*

The justification of theorem 2 comes from the observation that if Q' is a maximally-contained conjunctive rewriting of Q then $Q' \sqsubseteq \forall w.P$, for every $\forall w.P \in Q$. Hence, an upper bound on the size of Q' can be computed from the maximal number of views necessary to rewrite one conjunct of Q. From Theorem 1 and Lemma 2, to compute the rewritings of a conjunct $\forall w.P \in Q$ it is enough to consider concepts Q' made of minimal subsets of $\mathcal{V}$ such that $w \in V_{Q'}(P) \cup \mathbf{E}(Q')$. The following lemma characterizes the different forms of such rewritings, giving an upper bound of their size.

Lemma 3 *Consider a conjunct $\forall w.P \in Q$ and let l and p defined as previously. Let $Q' \equiv V_{i_1} \sqcap \ldots \sqcap V_{i_k}$. If $\{V_{i_1}, \ldots, V_{i_k}\}$ is a minimal subset of views in $\mathcal{V}$ such that $w \in V_{Q'}(P) \cup \mathbf{E}(Q')$ then one of the following conditions holds:*

- *a) $k = 1$, i) $P \in \{A, \neg A\}$ and $\forall w.P \in V_{i_1}$ or, ii) $P = (\geq nR)$ and $\forall w.(\geq pR) \in V_{i_1}$ with $p \geq n$ or, iii) $P = (\leq nR)$ and $\forall w.(\leq pR) \in V_{i_1}$ with $p \leq n$.*
- *b) $1 \leq k \leq l + 1$, $P = E$ and $\{V_{i_1} ... V_{i_k}\}$ is s.t.: for $j \in [1, k]$, $\forall w.E_{i_j} \in V_{i_j}$, and $\cap_{j=i_1}^{i_k} E_j \subseteq E$.*
- *c) $1 < k \leq l + 1$, $P = (\leq\ n\ R_v)$, with $R_v \in \mathcal{R}_v$, $\{V_{i_1} ... V_{i_k}\}$ is s.t.: for $j \in [1, k]$, $\forall w.E_{i_j} \in V_{i_j}$, and $|\cap_{j=i_1}^{i_k} E_j| \leq n$.*
- *d) $1 \leq k \leq l + p + 2$ and there exists a prefix $w'v$ of w s.t. $\forall w'.(\leq 0v) \in \sqcap_{j=i_1}^{i_k} V_j$.*

[5]The depth of a conjunct $\forall w.P$ is equal to the length of the word w.

Case (d) of this lemma gives an upper bound on the size of maximally-contained conjunctive rewritings. This upper bound is the one used by theorem 2. Moreover, as a consequence of that theorem, time complexity of $rewrite(\mathcal{V}, Q)$ is in $O(m^{n*(l+p+2)})$, where m denotes the number of views in $\mathcal{V}$ and where n, l and p are defined as previously. This result comes from the fact that to tackle a $rewrite(\mathcal{V}, Q)$ problem, and thanks to theorem 2, we only need to consider subsets of $\mathcal{V}$ that are made of at most $n * (l + p + 2)$ views. The search space being still exponential, naïve approaches are intractable and clearly do not scale w.r.t. the number of views. However, the previous analysis suggests that better approaches to narrow the search space in the spirit of the "bucket algorithm" [7] can be used. Given a rewriting problem $rewrite(\mathcal{V}, Q)$ with $Q \equiv \forall w_1.P_1 \sqcap \ldots \sqcap \forall w_n.P_n$, the main idea is to consider first each conjunct $\forall w_i.P_i$ of Q in isolation and to create an associated *bucket*, noted $B(w_i, P_i)$, that contains all the maximally-contained conjunctive rewritings of this conjunct. Then, in a second step, candidate rewritings of Q are computed by combining elements from the buckets by conjunction.

Let us now consider the problem of creating a bucket $B(w, P)$ associated with a conjunct $\forall w.P$ of a given query Q. Lemma 3 gives the main steps to solve this problem. In the remainder of this paper, we focus our attention on the algorithmic difficulties induced by the presence of set of values. A careful look at lemma 3 shows that set of values are dealt with in two kinds of computations that are required, respectively, in case (b) and cases (c) and (d). To formulate precisely these two problems, we use the following notation: $\mathcal{V}_w = \{V_{i_1}, ..., V_{i_p}\}$ denotes the subset of $\mathcal{V}$ such that $\exists E_{i_j} \,|\, (\forall w.E_{i_j}) \in V_{i_j}, \forall i_j \in [i_1, i_p]$. In this case, we also note $\mathtt{F}_w = \{E_{i_1}, ..., E_{i_p}\}$ the set of the E_{i_j}'s associated to the view of $\mathcal{V}_w$.

Therefore, given a conjunct $\forall w.P \in Q$, the first problem is introduced by case (b) of lemma 3 (i.e., when $P = E$) and is related to the computation of every minimal subset MS of $\mathtt{F}_w$ (associated with $\mathcal{V}_w$) such that intersection of the elements of MS is contained in E. We will denote by $S_1(w, E)$ the set of such subsets $\mathtt{F}_w$. The second problem is introduced by case (c) of lemma 3 (i.e., when $P = (\leq nR_v)$) and given $\mathcal{V}_{wR_v}$, amounts to the computation of every minimal subset of MS of $\mathtt{F}_{wR_v}$ such that the cardinality of the intersection of the elements of MS is less than the integer n. We denote by $S_2(wR_v, n)$ the set of such subsets of $\mathtt{F}_{wR_v}$. Note that, the problem of computing $S_2(wR_v, n)$ occurs also in case (d) of lemma 3.

More generally for a word w, sets $S_1(w, E)$ and $S_2(w, n)$ are formally defined below.

Definition 2 *Let E be a set. $\mathcal{P}(E)$ is the power set of E.*

$S_1(w, E) = \{X \in \mathcal{P}(\boldsymbol{F}_w) | \cap \{x \in X\} \subseteq E \text{ and } \forall Y \subset X, \cap\{y \in Y\} \not\subseteq E\}$

$S_2(w, n) = \{X \in \mathcal{P}(\boldsymbol{F}_w) || \cap \{x \in X\}| \leq n \text{ and } \forall Y \subset X, |\cap \{x \in X\}| > n\}$

In next section, we show how the problem of computing $S_1(w, E)$ and $S_2(w, n)$ can be tied up with the theoretical framework of Knowledge Discov-

ery in Databases (KDD) [11] in order to take benefits from existing scalable solutions developed in that context.

4 A KDD-based approach for query rewriting

A theoretical framework for KDD has been proposed in [11], in which scalability issues w.r.t. both the size of the database and the size of the search space can be dealt with. This framework has been successfully applied in many application from association rules to functional or inclusion dependencies.

This framework can be stated as follows: Let r be a database, $\mathcal{L}$ a finite language for expressing sentences or patterns, and P a predicate qualifying *interesting sentences* of $\mathcal{L}$ in r. X is said to be an interesting sentence of $\mathcal{L}$ in r, iff $P(X, r)$ is true. Given $r, \mathcal{L}$ and P, the purpose is to find all interesting sentences, the so-called *theory* of $r, \mathcal{L}$ and P, denoted by $\mathcal{T}h(r, \mathcal{L}, P)$, or simply $\mathcal{T}h$ when $r, \mathcal{L}$ and P are clear from context, and defined as $\mathcal{T}h(r, \mathcal{L}, P) = \{\varphi \in \mathcal{L} \mid P(r, \varphi)$ is true$\}$.

Computing the theory $\mathcal{T}h(r, \mathcal{L}, P)$ can be reduced to the computation of its borders [11]. To do this, a partial order relation on $\mathcal{L}$ has to exist and more importantly, the predicate P must be anti-monotone with respect to this partial order. More formally, let $\preceq$ be a partial order on $\mathcal{L}$. P is *anti-monotone* with respect to $\preceq$ iff for all $X, Y \in \mathcal{L}$ such that $X \preceq Y$, if $P(Y, r)$ is true then $P(X, r)$ is true. Given a theory $\mathcal{T}h$, two borders can be defined, the so-called positive an negative borders. They are defined as follows:

$\mathcal{B}d^+(\mathcal{T}h) = \{\varphi \in \mathcal{T}h \mid$ for all $\theta \in \mathcal{L}$ with $\varphi \prec \theta, \theta \notin \mathcal{T}h\}$

$\mathcal{B}d^-(\mathcal{T}h) = \{\varphi \in \mathcal{L} \setminus \mathcal{T}h \mid$ for all $\gamma \in \mathcal{L}$ with $\gamma \prec \varphi, \gamma \in \mathcal{T}h\}$

We shall see in the sequel how the problem of computing the sets $S_1(w, E)$ and $S_2(w, n)$ can be formulated into this framework.

The Identification of $S_1(w, E)$ and of $S_2(w, n)$ Let $\mathtt{F}_w = \{E_{i_1}, ..., E_{i_p}\}$ be a set of p elements. In our context, the language $\mathcal{L}_w$ turns out to be the powerset of $\mathtt{F}_w$ i.e. $\mathcal{P}(\mathtt{F}_w)$, the partial order being the subset relation $\subseteq$. The database being empty in our formulation, the scalability of this framework applies only on the search space.

Two predicates have to be defined, denoted by P_1 and P_2, to cope with our two problems at hand. They are defined as follows: Let $X \subseteq \mathcal{L}_w$

- $P_1(E, X)$ is true iff $\cap\{E_{i_j} \in X\} \not\subseteq E$.
- $P_2(n, X)$ is true iff $|\cap\{E_{i_j} \in X\}| > n$.

Predicates P_1 and P_2 being anti-monotone with respect to $\subseteq$, $S_1(w, E)$ and $S_2(w, n)$ can be characterized as borders as follows:

Theorem 3 $S_1(w, E) = \mathcal{B}d^-(Th(\emptyset, \mathcal{L}_w, P_1))$

$S_2(w, n) = \mathcal{B}d^-(Th(\emptyset, \mathcal{L}_w, P_2))$

Recently, several algorithms have been proposed to compute efficiently the positive and negative borders of a given theory [11, 6]. Such algorithms can be reused in our context to compute $S_1(w, E)$ and $S_2(w, n)$.

5 Conclusion

Our work complements existing query rewriting approaches [7] by considering a new kind of constraints that can be very useful in practical situations. It should be noted that value constraints (or, in general, the one-of construct) can be simulated by languages that contain negation and disjunction (e.g., $\mathcal{ALC}$). However, it is well known that such an approach may lead to terminologies that are very large and hence, reasoning with such terminologies may be very problematic[6].

In this paper, we investigated the query rewriting problem in the setting of $\mathcal{ALN}(\mathcal{O}_v)$, a language that is suitable for our application context. However, the main results presented here can be extended to $\mathcal{ALN}(\mathcal{O})$ adopting, in this case, a non-standard semantics for individuals in order to avoid intractactability of subsumption [9]. Finally, up to our knowledge, it is the first time an hybrid framework that combines knowledge representation and reasoning with KDD techniques is used to solve such problems. A prototype implementing the presented approach is currently under development.

References

[1] S. Abiteboul and O. M. Duschka. Complexity of answering queries using materialized views. In PODS'1998, pages 254–263.

[2] F. Baader and Ralf Küsters. Least common subsumer computation w.r.t. cyclic aln-terminologies. In *Description Logics*, 1998.

[3] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P.F. Patel-Schneider, editors. *The Description Logic Handbook*. Cambridge University Press, 2003.

[4] C. Beeri, A. Halevy, and M.C. Rousset. Rewriting Queries Using Views in Description Logics. In PODS'97, pages 99–108.

[5] A. T. Borgida and P. F. Patel-Schneider. A semantics and complete algorithm for subsumption in the classic description logic. *JAIR*, 1:277–308, 1994.

[6] D. Gunopulos, R. Khardon, H. Mannila, S. Saluja, H. Toivonen, and R.S. Sharma. Discovering all most specific sentences. *TODS*, 28(2), 2003.

[7] Alon Y. Halevy. Answering queries using views: A survey. *VLDB*, 10(4):270–294, 2001.

[8] H. Jaudoin, F. Toumani, C. Rey, J-M. Petit, and M. Schneider. Query rewriting in presence of value constraints (extended version). TR-ALNOv-05, 2005. http://www.isima.fr/jaudoin/

[9] R. Küsters and A. Borgida. What's in an Attribute? Consequences for the Least Common Subsumer. *JAIR*, 14:167–203, 2001.

[10] M. Lenzerini. Data integration : A theoretical perspective. In *PODS*, 2002.

[11] H. Mannila and H. Toivonen. Levelwise search and borders of theories in knowledge discovery. *DMKD*, 1(3):241–258, 1997.

[12] I. Horrocks and U. Sattler. Ontology Reasoning in the $\mathcal{SHOQ}$(D) Description Logic. *IJCAI 2001*, 1(3):199–204, 2001.

[6] For example, using such an approach in our application context requires adding more than 900 millions of new axioms in our terminology.