The use of CARIN language and algorithms for Information Integration: the PICSEL project

Véronique Lattes and Marie-Christine Rousset
L.R.I. Computer Science Laboratory
C.N.R.S & University of Paris-Sud
Building 490, 91405, Orsay Cedex, France
mcr@lri.lri.fr

ABSTRACT

PICSEL is a project[1] aiming at building information servers over existing information sources that are distributed and possibly heteregeneous. The approach which has been chosen in PICSEL is to define an information server as a knowledge-based mediator in which CARIN is used as the core logical formalism to represent both the domain of application and the contents of information sources that are available and relevant to the domain of application.

The goal of this paper is to point out the way we exploit the expressive power of CARIN for describing the contents of information sources in a way that enables avoiding the query reformulation problem, by mixing the *local-as-global* and *global-as-local* approaches. We illustrate it on examples coming from the tourism domain, which the first real case that we have to consider in PICSEL in collaboration with the Web (and Minitel) travel agent Degriftour[2].

[1]granted by CNET (Centre National d'Etudes des Telecommunications) under contract number 97 1B 378
[2]see http://www.degriftour.fr/

1 Introduction

In recent years, the problem of information integration has received a lot of attention. In particular, several information integration systems (e.g, Information Manifold [18], TSIMMIS [6], SIMS [2], Infomaster [7]) have been proposed, based on a *knowledge-based mediator* architecture. They have in common the use of a logical formalism to describe both the *domain model* and the *contents of the information sources*. They differ by (i) the specific logical formalism that they are based on, (ii) the way they express the contents of the information sources, and (iii) the query processing that they perform.

As pointed out in [16] two approaches can be distinguished depending on the mapping between the domain relations (called the *global relations*) and the sources relations (called the *local relations*) which represent the contents and capabilities of the information sources. In the *local-as-global* approach, illustrated by Information Manifold [18] or Infomaster [7], the local relations are defined as *views* over global relations. In the *global-as-local* approach, illustrated by TSIMMIS [6], the mapping is done from the global relations to the local relations: the local relations are considered as as many new base predicates in term of which some global relations can be expressed. The two approaches have dual advantages and drawbacks. The main advantage of the *local-as-global* approach is its flexibility for representing the contents of the sources relatively to an application domain. Its drawback is the complexity of the problem of answering queries. While in the *global-as-local* approach, answering a query can be simply done by expanding it until getting local relations, in the *local-as-global* approach, the problem of answering queries using local relations becomes a problem of reformulation of queries in terms of views. This problem is closely related to the problem of query containment that has been extensively studied in the database literature (e.g., [4, 1, 13, 8, 14, 17, 5, 15, 12, 3]). This problem has been shown as being generally harder than the problem of answering queries over a database. In particular, it is undecidable for Datalog.

In this paper, we describe the framework of the PICSEL information integration system, which makes use of the expressive power of the CARIN language in a way that enables avoiding the query reformulation problem, by mixing the *local-as-global* and *global-as-local* approaches. CARIN[10] is a family of representation languages, that combine the expressive power of Datalog rules and of description logics. Datalog rules are appropriate for defining complex domain relations and global queries in terms of base relations. Description Logics have been designed to represent and reason on classes of objects. In the PICSEL framework, the description logics constructors are specially useful for expressing constraints that are known to hold on the contents of the sources. In contrast with other related systems (like SIMS, Information Manifold or Infomaster), we do not describe the contents of the information sources by views over domain relations, but by CARIN knowledge bases that are added to the CARIN description of the domain application to answer queries. A CARIN knowledge base describing an information source is made of two kinds of assertions : (i) implicative sentences indicating which kinds of data are allowed being extracted from the source, (ii) description logics assertions which enable expressing constraints that are known on the data that can be obtained from the source.

The paper is organized as follows. Section 2 summarizes the CARIN language and associated inference algorithms. Section 3 illustrates by examples the use of CARIN in the PICSEL project. Finally, section 4 points out the current issues that are investigated for implementing and optimizing the algorithmic machinery of CARIN.

2 Knowledge representation and reasoning in CARIN

A CARIN knowledge base (KB) contains three components, the first is a set $\Delta_{\mathcal{T}}$ of description-logic statements, the second $\Delta_{\mathcal{R}}$ is a set of rules and the third is a set $\Delta_{\mathcal{F}}$ of existential sentences (that can contain ground facts). Description logics (DL) statements are definition or inclusion statements about concepts and roles in the domain. Concepts and roles can also appear as (unary or binary) predicates in the antecedents of the rules and in the existential sentences. Predicates that do not appear in the description logics statements are called *ordinary predicates.* Ordinary predicates can be of any arity.

2.1 Description Logics statements in carin

The DL language that we consider in the PICSEL project is referred to as core-Classic. It contains the constructors $\sqcap$(conjunction), $\forall$ (concept restriction), $(\geq n\,R), (\leq n\,R)$ (number restrictions) and $\neg$ (negation, on basic concepts only).

The sentences in the DL component $\Delta_{\mathcal{T}}$ of a carin knowledge base Δ are either *concept definitions* or *concept inclusions.* A concept definition is a statement of the form $A := D$, where A is a concept name and D is a concept description. We assume that a concept name appears on the left hand side of at most one concept definition. An inclusion statement is of the form $C \sqsubseteq D$, where C and D are concept descriptions. Intuitively, a concept definition associates a definition with a name of a concept. An inclusion states that every instance of the concept C must be an instance of D.

The semantics of the DL statements is the standard model-based semantics of first-order logic, in which concepts and roles are interpreted as unary and binary predicates. An interpretation I contains a non-empty domain $\mathcal{O}^I$. It assigns a unary relation C^I to every concept in $\Delta_{\mathcal{T}}$, and a binary relation R^I over $\mathcal{O}^I \times \mathcal{O}^I$ to every role R in $\Delta_{\mathcal{T}}$. The extensions of concept and role descriptions are given by the following equations: ($\sharp\{S\}$ denotes the cardinality of a set S):

$$
\begin{array}{l}
\top^I = \mathcal{O}^I,\ \bot^I = \emptyset, \\
(C \sqcap D)^I = C^I \cap D^I, \\
(\neg C)^I = \mathcal{O}^I \setminus C^I, \\
(\forall R.C)^I = \{d \in \mathcal{O}^I \mid \forall e : (d,e) \in R^I \rightarrow e \in C^I\} \\
(\geq n\,R)^I = \{d \in \mathcal{O}^I \mid \sharp\{e \mid (d,e) \in R^I\} \geq n\} \\
(\leq n\,R)^I = \{d \in \mathcal{O}^I \mid \sharp\{e \mid (d,e) \in R^I\} \leq n\}
\end{array}
$$

An interpretation I is a model of $\Delta_{\mathcal{T}}$ if $C^I \subseteq D^I$ for every inclusion $C \sqsubseteq D$ in $\Delta_{\mathcal{T}}$, $A^I = D^I$ for every concept definition $A := D$.

2.2 Rules and existential sentences in carin

Rule component:

The rule component $\Delta_{\mathcal{R}}$ of a carin knowledge base Δ contains a set of rules that are logical sentences of the form:

$$\forall \bar{X} [p_1(\bar{X}_1) \wedge \ldots \wedge p_n(\bar{X}_n) \Rightarrow q(\bar{Y})]$$

where $\bar{X}_1, \ldots, \bar{X}_n, \bar{Y}$ are tuples of variables (included in $\bar{X}$) or constants. We require that the rules are safe, i.e., a variable that appears in $\bar{Y}$ must also appear in $\bar{X}$. The predicates $p_1, \ldots, p_n$ may be either concept names or expressions, role names, or *ordinary* predicates that do not appear in $\Delta_{\mathcal{T}}$. Recall that ordinary predicates can be of any arity. The predicate q must be an ordinary predicate.

An ordinary predicate p is said to *depend* on an ordinary predicate q if q appears in the antecedent of a Horn rule whose consequent is p. A set of rules are said to be *recursive* if there is a cycle in the dependency relation among ordinary predicates.

Existential sentences component:

The existential sentences component $\Delta_{\mathcal{F}}$ of a carin knowledge base contains a set of sentences of the form

$$\exists \bar{Y} [p_1(\bar{a}_1, \bar{Y}_1) \wedge \ldots \wedge p_n(\bar{a}_n, \bar{Y}_n)]$$

where the p_i's are either concept, role or ordinary predicates. Note that $\bar{Y}$ can be empty. Such existential sentences are then simply a conjunction of ground facts

2.3 SEMANTICS OF CARIN

The semantics of CARIN is derived in a natural way from the semantics of its components languages. An interpretation I contains a non-empty domain $\mathcal{O}^I$. The interpretation assigns to every constant a in Δ an object $\alpha^I(a) \in \mathcal{O}^I$, and a relation of arity n over the domain $\mathcal{O}^I$ to every predicate of arity n in Δ. An interpretation I is a model of a knowledge base Δ if it is a model of each of it components. Models of the DL component were defined in Section 2.1. An interpretation I is a model of a rule r if, whenever α is a mapping from the variables of r to the domain $\mathcal{O}^I$, such that $\alpha(\bar{X}_i) \in p_i^I$ for every atom of the antecedent of r, then $\alpha(\bar{Y}) \in q^I$, where $q(\bar{Y})$ is the consequent of r. Finally, I is a model of an existential sentence f if it exists a mapping α from its variables to the domain $\mathcal{O}^I$ such that $(\bar{a}^I, \alpha(\bar{Y}_i)) \in p_i^I$ for every atom $p_i(\bar{a}_i, \bar{Y}_i)$ in f.

It should be noted that CARIN does not allow concept and role atoms to appear in the consequents of the rules because of the underlying assumption that the DL component *completely* describes the hierarchical structure in the domain, and therefore, the rules should not allow to make new inferences about that structure.

2.4 REASONING IN CARIN

The reasoning problems that have been addressed in CARIN are the problem of answering conjunctive queries from a CARIN knowledge base and the problem of containment of conjunctive queries. In [11, 10], *sound* and *complete* inference algorithms are described for reasoning in CARIN. An important point to note is that combining standard Horn rule inference procedures with intermediate DL reasoning steps does not result in a complete inference procedure for CARIN. The reason is that Horn rule reasoning procedures apply each rule in *isolation,* and they try to *instantiate* the antecedent of the rule in order to derive its consequent. The single step of applying a Horn rule needs to be replaced by a more sophisticated reasoning step referred to as *existential entailment.* Speaking informally, the existential entailment problem is to decide whether a set of existential sentences, together with a set of DL statements logically entail the disjunction of the antecedents of the set of rules relevant to the query. The existential entailment algorithm is precisely described and proved correct and complete in [9]. It consists of first, constructing a set of completions from an initial set of existential facts obtained from the original knowledge base, second evaluating the set of rules relevant to the query on each completion. The construction of completions is based on using propagation rules that account for DL constructors but also on simulating a case-analysis reasoning on concept-facts $C(X)$. It is done by injecting in the completions universal sentences of the form $\forall x[C(x) \vee \neg C(x)]$ for every concept expression C appearing in the rules that are relevant to the query. It results in several branches in the construction of completions and finally in completions in which it is stated explicitly whether $C(s)$ is true or false for every object s.

3 Logical representation of the domain and of the sources in PICSEL

PICSEL is a project aiming at building information servers over existing information sources that are distributed and possibly heteregeneous. An information server is dedicated to a specific domain of application (e.g, tourism) and gives its users the illusion of a centralized and homogeneous information system, allowing them to ask domain-level queries and taking in charge in their place the access to the relevant sources in order to obtain the answers to their queries.

The approach which has been chosen in PICSEL is to define an information server as a knowledge-based mediator in which CARIN is used as the core logical formalism to represent both the domain of application and the contents of information sources that are available and relevant to the domain of application. Query processing is generic (i.e not specific to any particular information server): it is the task of a unique and general query engine which is based on the implementation of the inference algorithm of CARIN. In this way, building a new information server related to a new application domain amounts to design a CARIN knowledge base. The programming work is then reduced to write the wrappers between each new information source and the new mediator. No additional programming is needed for query processing: the general query engine is used for any information server.

The goal of this section is to point out the way we exploit the expressive power of CARIN for describing the contents of information sources in a way that avoids the query reformulation problem of most existing mediator approaches.

We illustrate it on examples coming from the tourism domain, which the first real case that we have to consider in PICSEL in collaboration with the Web (and Minitel) travel agent Degriftour [3]. This travel agent makes available on-line three databases that contain different types of touristic products (flights, tours, stays in different places), each of them having its own specificities. For example, the *BonjourFrance* database offers a large variety of touristic products but all those products are located in France. The so-called *Degriftour* database offers flights, stays and tours for a lot of destinations all over the world. Its specificity however is that all its offers correspond to a departure date which is within the next two weeks. As a counterpart, the corresponding prices are specially interesting. The *Reductour* database provides rather similar products but with a less strong constraint on the departure date: it just has to be within the next eight months. Other differences exist between the contents of those three databases. For instance, we might know that the only housing places that are proposed in the *Reductour* database are hotels, while the others can provide rooms in Bed&Breakfast in addition to hotel rooms.

Modelling the domain of application:

Our approach for modelling the domain is similar in spirit to most of the existing mediator approaches. We define a basic vocabulary in terms of names of base predicates that are meaningful for tourism, and we use our logical formalism (CARIN) for defining new relations that are significant for the tourism domain and that can be defined over the base relations. In the setting of CARIN there are two ways of defining complex relations: by rules, or by concept expressions. It is illustrated by the following example.

Example 3.1: Let us suppose that our basic vocabulary contains basic concepts (i.e unary predicates) such as $HousingPlace$, $Breakfast$, $CollectiveBuilding$, $PrivateBuilding$, which respectively denote the set of housing places and varied sets of services or buildings, and basic roles (i.e binary predicates) such as $AssRoom$ $AssMeal$ and $AssBuilding$, which denote binary relations between housing places and their associated buildings, room and meal services.

We can define the two new concepts $Hotel$ and $Bed\&Breakfast$ by the two following DL definitions:

$$Hotel := HousingPlace \sqcap (\geq\ 1\, AssRoom) \sqcap (\forall AssBuilding.CollectiveBuilding)$$

$$Bed\&Breakfast := HousingPlace \sqcap (\geq\ 1\, AssRoom) \sqcap (\geq\ 1\, AssMeal) \sqcap (\forall AssMeal.Breakfast) \sqcap (\forall AssBuilding.PrivateBuilding)$$

$$CollectiveBuilding \sqcap PrivateBuilding \sqsubseteq \bot$$

Rules can be used to define new n-ary relations. For instance, the following rule defines the notion of flights combined with stays as a 6-ary relation $FlightCombinedStay$. A stay combined with a flight is characterized by a departure city (denoted by the first variable *Dcity* in the consequent of the rule), an arrival city (denoted by the variable *Acity*), a departure date (*Ddate*), a return date (*Rdate*), a number of persons (*Pnumber*) and a price (P). The possible combination of a flight to and back a given destination with a stay at that place obey some constraints that are expressed by the conditions in the antecedent of the rule. The predicates *Stay*, $Flight$, ... appearing in the antecedent of the rule are either base predicates or ordinary non base predicates that are defined in other rules.

[3] see http://www.degriftour.fr/

$$
\begin{array}{l}
Stay(S) \wedge Flight(V) \wedge Assoc(S,H) \wedge Located(H,ACity) \\
\wedge ArrivalCity(V,Acity) \wedge DepartureCity(V,Dcity) \wedge DepartureDate(V,Ddate) \\
\wedge ReturnDate(V,Rdate) \wedge BeginningDate(S,Ddate) \wedge EndDate(S,Rdate) \\
\wedge PersonNumber(S,Pnumber) \wedge PersonNumber(V,Pnumber) \\
\wedge Price(S,P1) \wedge Price(V,P2) \wedge P = P1 + P2 \\
\quad \Rightarrow FlightCombinedStay(Dcity,Acity,Ddate,Rdate,Pnumber,P)
\end{array}
$$

Modelling the contents of the information sources:

Our approach combines the *local-as-global* and *global-as-local* approaches. The mapping between the source relations and the domain relations follows a *global-as-local* approach, but it is completed by a set of CARIN sentences expressing constraints that are known on the content of the sources, and therefore that are known to hold on the source relations.

More precisely, we describe each information source $\mathcal{S}$ by a CARIN knowledge base $Descr(\mathcal{S})$ involving a set of source relations $\mathcal{V}_S$ and containing:

1. a set $\mathcal{I}_S$ of logical implications $v(\bar{X}) \Rightarrow p(\bar{X})$ that indicate which kind of data can be found in the source $\mathcal{S}$. The p's are global relations and there are as many source relations v's in $\mathcal{V}_S$ (and as many implications in $\mathcal{I}_S$) as domain relations whose instances can be found in the source $\mathcal{S}$,
2. a set $\mathcal{C}_S$ of constraints on the instances of the source relations. We allow two types of constraints:
   - core-CLASSIC inclusion statements between concept expressions involving both source relations and concepts or roles involved in the domain vocabulary,
   - incompatibility rules $p_1(\bar{X}_1) \wedge \ldots \wedge p_n(\bar{X}_n) \Rightarrow \perp$ where the p_i's are source relations or domain relations.

As an example, we can have the following (partial, for this illustration) descriptions of the three information sources that we have previously mentioned (i.e $BonjourFrance$, $Degriftour$ and $Reductour$).

Example 3.2: The implications in the description of the three sources say that we can find instances of housing places and flights in all of them (together with instances of associated properties like e.g their location, their departure cities and dates etc ...). The constraints contained in each description enable distinguishing between them: for instance, the housing places and flights that can be found in $BonjourFrance$ are restricted to be located in France; the housing places that can be found in $Reductour$ are necessarily hotels. Some constraints serve to express that instances of some binary relations that can be found in a given source are exclusively related to some unary relations that can be found in the same source: for instance, the constraint $(\geq 1\, v^2_{BF}) \sqsubseteq v^1_{BF}$ in $Descr(BonjourFrance)$ expresses that the locations that can be found in the source $BonjourFrance$ (denoted by the source relation v^2_{BF} and the implication $v^2_{BF}(X,Y) \Rightarrow Located(X,Y)$ in $Descr(BonjourFrance)$) are only locations that are related to the housing places that can be found in the same source (denoted by the source relation v^1_{BF} and the implication $v^1_{BF}(X) \Rightarrow HousingPlace(X)$ in $Descr(BonjourFrance)$)

$Descr(BonjourFrance)$:

$$
\begin{array}{l}
\mathcal{V}_{BonjourFrance} : v^1_{BF}, \ldots, v^n_{BF} \\
\mathcal{I}_{BonjourFrance} : \\
\quad v^1_{BF}(X) \Rightarrow HousingPlace(X) \\
\quad v^2_{BF}(X,Y) \Rightarrow Located(X,Y) \\
\quad v^3_{BF}(X) \Rightarrow Flight(X) \\
\quad v^4_{BF}(X,Y) \Rightarrow DepartureCity(X,Y) \\
\quad \ldots \\
\mathcal{C}_{BonjourFrance} : \\
\quad v^1_{BF} \sqsubseteq (\forall Located.France)
\end{array}
$$

$(\geq 1\, v_{BF}^2) \sqsubseteq v_{BF}^1$
$v_{BF}^3 \sqsubseteq (\forall DepartureCity.France) \sqcap (\forall ArrivalCity.France)$
$(\geq 1\, v_{BF}^4) \sqsubseteq v_{BF}^3$
$\ldots$

$Descr(Degriftour)$:

$\mathcal{V}_{Degriftour} : v_D^1, \ldots, v_D^k$
$\mathcal{I}_{Degriftour}$:
$v_D^1(X) \Rightarrow HousingPlace(X)$
$v_D^2(X,Y) \Rightarrow Located(X,Y)$
$v_D^3(X) \Rightarrow Flight(X)$
$v_D^4(X,Y) \Rightarrow DepartureCity(X,Y)$
$\ldots$
$\mathcal{C}_{Degriftour}$:
$v_D^3 \sqsubseteq (\forall DepartureDate.NextTwoWeeks)$
$(\geq 1\, v_D^4) \sqsubseteq v_D^3$
$\ldots$

$Descr(Reductour)$:

$\mathcal{V}_{Reductour} : v_R^1, \ldots, v_R^p$
$\mathcal{I}_{Reductour}$:
$v_R^1(X) \Rightarrow Hotel(X)$
$v_R^2(X,Y) \Rightarrow Located(X,Y)$
$v_R^3(X) \Rightarrow Flight(X)$
$v_R^4(X,Y) \Rightarrow DepartureCity(X,Y)$
$\ldots$
$\mathcal{C}_{Reductour}$:
$v_R^3 \sqsubseteq (\forall DepartureDate.NextEightMonths)$
$(\geq 1\, v_R^4) \sqsubseteq v_R^3$
$\ldots$

Query processing:

Given a conjunctive query $Q(\bar{X})$, expressed in terms of domain relations, query processing consists of exploiting the domain knowledge (composed of a set of rules $\mathcal{R}$, a terminology $\mathcal{T}$ and possibly a set of facts $\mathcal{F}$) together with the source descriptions (composed of a set $\mathcal{I}_\mathcal{V}$ of logical implications and a set $\mathcal{C}_\mathcal{V}$ of constraints involving a set of source relations $\mathcal{V}$) in order to find a set of conjunctive queries $Q_\mathcal{V}^1(\bar{X}), \ldots Q_\mathcal{V}^n(\bar{X})$ containing only source relations (i.e v's belonging to $\mathcal{V}$), which are (maximally contained) *rewritings* of $Q(\bar{X})$.

A conjonctive query $Q_\mathcal{V}(\bar{X})$ is a rewriting of $Q(X)$ in terms of $\mathcal{V}$ iff

- it contains only atoms whose predicate belongs to $\mathcal{V}$,
- $Q_\mathcal{V}(\bar{X})$, $\mathcal{R}$, $\mathcal{T}$, $\mathcal{F}$, $\mathcal{I}_\mathcal{V}$, $\mathcal{C}_\mathcal{V}$ is satisfiable,
- $Q_\mathcal{V}(\bar{X})$, $\mathcal{R}$, $\mathcal{T}$, $\mathcal{F}$, $\mathcal{I}_\mathcal{V}$, $\mathcal{C}_\mathcal{V} \models Q(\bar{X})$

A rewriting $Q_\mathcal{V}(\bar{X})$ is maximally contained in $Q(X)$ if it does not exist any rewriting $P_\mathcal{V}(\bar{X})$ of $Q(X)$ such that $Q_\mathcal{V}(\bar{X}) \models P_\mathcal{V}(\bar{X})$.

In the *local-as-global* approach, the general way for finding (maximally contained) rewritings of a global query in terms of source relations consists of (i) determining a bound on the size of the conjunctive queries that are needed in the rewriting to be contained in the query, (ii) testing all the possible rewritings whose size is smaller than that bound using query containment algorithms. There are two sources of complexity for that problem: first, the set of possible rewritings can be hudge, second testing query containment can be costly (even undecidable for queries expressed in full Datalog). In the setting of conjunctive queries over description logics, even if we restrict the rewritings to be conjunctive queries, it has been proven [3] that this two-step approach is not applicable because there may be no *bound* on the size of the conjuntive rewritings that need to be considered.

From the point of view of finding the candidate rewritings, our approach is similar to the *global-as-local* approach in which a backward-chaining expansion of the query suffices to determine the possible rewritings. Thus, we avoid one source of complexity of the *local-as-global* approach for finding rewritings. We still have to verify that the possible candidates satisfy the constraints which were not involved in the backward chaining process. In addition, because of the expressive power of the CARIN language and the resulting complexity of the associated inference problems, we also have to verify that the satisfiable rewritings entail the query. It is due to the fact that the backward chaining does not take into account case-analysis reasoning.

More precisely, the query processing in PICSEL proceeds in three steps:

1. **Query expansion step:** this step consists of expanding the query by combining a backward-chaining algorithm for unfolding the rules (of $\mathcal{R}$ and $\mathcal{I}_{\mathcal{V}}$) with a backward-chaining algorithm that determines logical dependency links between core-CLASSIC existential sentences. This backward-chaining algorithm is based on a set of inference rules which characterizes entailment for core-CLASSIC existential sentences. As a result, we obtain the expansion of $Q(\bar{X})$ as a disjunction of existential sentences $CJ(\bar{X})$ of the form $\exists \bar{Y} v_1(\bar{Y}_1, \bar{X}_1) \wedge \ldots \wedge v_i(\bar{Y}_i, \bar{X}_i) \wedge remainder(CJ(\bar{X}, \bar{Y}))$ where the possible $remainder(CJ(\bar{X}, \bar{Y}))$ is a core-CLASSIC existential sentence.
   $\exists \bar{Y} v_1(\bar{Y}_1, \bar{X}_1) \wedge \ldots \wedge v_i(\bar{Y}_i, \bar{X}_i)$ is called the *core* of the existential conjunctive sentence $CJ(\bar{X})$, denoted $core(CJ(\bar{X}))$
   For every $CJ(\bar{X})$ belonging to the expansion of $Q(\bar{X})$: $CJ(\bar{X}) \models Q(\bar{X})$
   Therefore, the possible rewritings of $Q(X)$ in terms of source relations are the cores of the existential sentences resulting from the expansion of the query. We have to check (i) whether they are compatible with the constraints, and (ii) whether, by a case-analysis reasoning on the concept expressions appearing in the remainders, they actually entail the query. It is the goal of the verification step of the query processing.

2. **Verification step:** we use the standard inference algorithms of CARIN which has been recalled in section 2 to check, for each candidate rewriting $Q_{\mathcal{V}}(\bar{X})$ found at the previous step, whether $Q_{\mathcal{V}}(\bar{X}), \mathcal{I}_{\mathcal{V}}, \mathcal{C}_{\mathcal{V}}, \mathcal{R}, \mathcal{T}, \mathcal{F}$ is satisfiable and entail $Q(\bar{X})$.

3. **Simplification step:** this step aims at selecting among the rewritings that have been determined at the previous step those which are maximal. For this step, the CARIN inference algorithm can be used too.

Remark 3.1: The simplification step can be considered as too costly and can be ignored. The result of query processing may then return rewritings that will provide answers to the initial query which may be redundant.

Remark 3.2: The query expansion step suffices for the purpose of finding the sources that are potentially relevant to the initial query. In fact, this first step eliminates all the sources whose content is irrelevant to the query. In some cases, it is informative enough for the users.

Let us illustrate the query processing on a very simple example.

Example 3.3: Let consider a query asking for flights from Paris to Saint-Malo in November 98 and for one night in a Bed&Breakfast in Saint-Malo. It is expressed by the following conjunctive query $q(X, Y, D)$.

$$Flight(X) \wedge DepartureCity(X, paris) \wedge ArrivalCity(X, stmalo)$$
$$\wedge DepartureDate(X, D) \wedge Nov98(D) \wedge Bed\&Breakfast(Y) \wedge Located(Y, stmalo)$$
$$\wedge ArrivalDate(Y, D) \wedge DepartureDate(Y, D') \wedge D' = D + 1$$
$$\Rightarrow q(X, Y, D)$$

At the query expansion step, the *Reductour* source is eliminated because $Bed\&Breakfast(Y)$ cannot be expanded using source relations belonging to the local vocabulary $\mathcal{V}_{Reductour}$, while the two other sources are considered as potentially relevant (we suppose that their descriptions contain as many source relations and implications as domain relations appearing in the body of the query). Different candidate rewritings are obtained involving either source relations of *BonjourFrance* or *Degriftour* alone, or a combination of source relations of *BonjourFrance* and *Degriftour*.

At the verification step, only *BonjourFrance* is proved being able to effectively answer the query by itself. The rewriting involving only source relations of *Degriftour* turns out to be unsatisfiable if we consider the constraints associated with it and the terminology. In particular, the rewriting in $\mathcal{V}_{Degriftour}$ of the conjunct $Flight(X) \wedge DepartureDate(X, D) \wedge Nov98(D)$ in the antecedent of the query together with the constraint $v_D^3 \sqsubseteq (\forall DepartureDate.NextTwoWeeks)$ in $Descr(Degriftour)$ and the terminological statement $Nov98 \sqcap NextTwoWeeks \sqsubseteq \perp$ of the domain knowledge, leads to unsatisfiability.

The rewriting mixing source relations of *BonjourFrance* (relative to Bed&Breakfast) and of *Degriftour* (relative to Flights) is verified as being acceptable too. Note that it is not necessarily redundant with the rewriting obtained from *BonjourFrance* alone.

4 Conclusions

Implementing and Optimization issues:

The query processing of PICSEL is in the process of being implemented in Java. Several optimizations methods are investigated in order to make the algorithmic machinery described in [9] reasonably efficient. In particular, the restriction of the DLlanguage to core- CLASSIC makes possible the characterization of entailment of existential sentences by a set of inference rules that are used to focus the inference algorithm. The complete hierarchy of the concepts involved in the domain and source descriptions can be computed and stored at compile time. Some indexing is done on the rules too at compile time. As a result, at query time, the expansion of the query can be efficiently obtained from the rules and the concept expressions that are relevant to it.

Another important optimization issue that we are investigating is to limit at query time the number of completions that have to be constructed in order to simulate case-analysis reasoning. First, we can restrict case-analysis reasoning on the concepts that appear in the query expansion. Second, we are identifying cases for which non costly tests can show that explicit case-analysis on some of those concepts is useless.

Related work:

PICSEL has similarities with TSIMMIS [6], Information Manifold [18] and Infomaster [7].

Like TSIMMIS, it follows a *global-as-local* approach for mapping the source relations with the domain relations. As a result, finding the candidate rewritings can be done in a focussed way by a backward-chaining expansion of the query. However, in contrast with TSIMMIS, in addition to this simple mapping, it takes into account rather rich constraints about the content of the sources that are expressed as description logics statements and incompatibility rules involving sources and domain relations. This aspect makes the PICSEL approach a combination of the *global-as-local* and *local-as-global* approaches.

The different steps of the query processing in Infomaster are quite similar to the three steps of the query processing in PICSEL. In particular, an abduction step is used in Infomaster to find the candidate rewritings. Some (functional dependency) constraints are expressed on the source relations. However, the language which is used in Infomaster does not contain any description logic component and is less expressive than the one we use in PICSEL for describing both the domain and the content of the sources.

Information manifold is based on the use of Datalog extended with some features of description logics and thus is based on a first and restricted version of CARIN. The main difference with PICSEL is that it follows a full *local-as-global* approach, expressing the source relations as views over domain relations. As a result, query answering is based on rewriting queries using views. In [3], it has been shown that the problem of rewriting queries using views when views and queries are conjonctive queries over core-CLASSIC may be undecidable except if some restrictions are imposed on variables appearing in the views. In PICSEL, we avoid the inherent complexity of this rewriting problem by drastically restricting the views that we allow. In particular, the logical implications that we allow in the description of the sources can be seen as a particular case of views definitions that do not contain existential variables, which corresponds to a restriction identified in [3] as making the rewriting problem decidable. We compensate this restriction by the use of expressive constraints that enable to enrich the description of the contents of the sources while guaranteing the query rewriting to remain decidable.

5 References

[1] Alfred Aho, Yehoshua Sagiv, and Jeffrey D. Ullman. Equivalence of relational expressions. *SIAM Journal of Computing*, (8)2:218–246, 1979.

[2] Yigal Arens, Craig A. Knoblock, and Wei-Min Shen. Query reformulation for dynamic information integration. *Journal of Intelligent Information Systems*, 6, 1996.

[3] Catriel Beeri, Alon Y.Levy, and Marie-Christine Rousset. Rewriting queries using views in description logics. In *Proceedings of the Sixteenth Symposium on Principles of Database Systems (PODS'97)*, 1997.

[4] A.K. Chandra and P.M. Merlin. Optimal implementation of conjunctive queries in relational databases. In *Proceedings of the Ninth Annual ACM Symposium on Theory of Computing*, pages 77–90, 1977.

[5] Surajit Chaudhuri and Moshe Vardi. On the equivalence of recursive and nonrecursive datalog programs. In *The Proceedings of the Eleventh ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, San Diego, CA.*, pages 55–66, 1992.

[6] Sudarshan Chawathe, Hector Garcia-Molina, Joachim Hammer, Kelly Ireland, Yannis Papakonstantinou, Jeffrey Ullman, and Jennifer Widom. The TSIMMIS project: Integration of heterogenous information sources. In proceedings of IPSJ, Tokyo, Japan, October 1994.

[7] Michael R. Genesereth, Arthur M. Keller, and Oliver M. Duschka. Infomaster: An information integration system. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD-97*, 1997.

[8] A. Klug. On conjunctive queries containing inequalities. *Journal of the ACM*, pages 35(1): 146–160, 1988.

[9] Alon Levy and Marie-Christine Rousset. Carin: A knowledge representation language combining horn rules and description logics. *to appear in Artificial Intelligence Journal*, 1998.

[10] Alon Y. Levy and Marie-Christine Rousset. CARIN: a representation language integrating rules and description logics. In *Proceedings of the European Conference on Artificial Intelligence, Budapest, Hungary*, 1996.

[11] Alon Y. Levy and Marie-Christine Rousset. The limits on combining recursive horn rules and description logics. In *Proceedings of the AAAI Thirteenth National Conference on Artificial Intelligence*, 1996.

[12] Alon Y. Levy and Yehoshua Sagiv. Queries independent of updates. In *Proceedings of the 19th VLDB Conference, Dublin, Ireland*, pages 171–181, 1993.

[13] Y. Sagiv and M. Yannakakis. Equivalence among relational expressions with the union and difference operators. *Journal of the ACM*, 27(4):633–655, 1981.

[14] Yehoshua Sagiv. Optimizing datalog programs. In Jack Minker, editor, *Foundations of Deductive Databases and Logic Programming*, pages 659–698. Morgan Kaufmann, Los Altos, CA, 1988.

[15] Oded Shmueli. Equivalence of datalog queries is undecidable. *Journal of Logic Programming*, 15:231–241, 1993.

[16] Jeffrey D. Ullman. Information integration using logical views. In *Proceedings of ICDT'97*, 1997.

[17] Ron van der Meyden. The complexity of querying indefinite data about linearly ordered domains. In *The Proceedings of the Eleventh ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, San Diego, CA.*, pages 331–345, 1992.

[18] Alon Y.Levy, Anand Rajamaran, and Joann Ordille. Query-answering algorithms for information agents. In *Proceedings of the thirtenth AAAI conference on Artificial Intelligence: AAAI'96*, 1996.