International Journal of Innovative Research in Computer Science & Technology (IJIRCST)
ISSN: 2347-5552, Volume-2, Issue-6, November-2014

Optimized Data Management Across Large Datasets

Victoria hebseba, B. Vijay Bhaskar Reddy

***Abstract*- The current trend for data management system is to use various algorithms or search engines to extract and retrieve the data collected and stored in the preexisting Data Management System. As the volume of the available information stored in the DMS is increasing regularly data extraction and retrieval is significantly challenging and taking longer and causing delays to the end users. This paper describes a new method which extends the existing definitions of modules and by introducing novel properties of robustness to optimize the data management across large datasets. Through investigations are carried in the setting of description logics which underlie modern ontology based system. This method meets the performance requirements with the highest possible system reliability and the most reasonable systems cost. Reference data is extracted as per the application needs and extra constraints are applied to manage the data using the resulting schema. The global query answering method is used to identify relevant data within the distributed data set consisting of the data set of the module based data. The flexibility associated with this system makes the data maintenance effective and efficient. Users are given authentication so as to allow only the registered users.**

***Index Terms*- Resource description framework, Semantic Web, Data models, Knowledge management**

I. INTRODUCTION

Robust Design is a proven development philosophy focused on achieving target reliability. Approaching this aggressive goal requires that Robust Design principles be an early and integral part of the development cycle. The objective is to make the end-product immune to factors that could affect performance. Robust Design requires that the following four factors be Considered in the design process: signal, response, noise, and control. Noise factors are disturbances that cause the systems response to shift from specification. These factors are likely beyond the designer's control, such as manufacturing tolerances, aging, usage patterns, environmental conditions, etc.

Noise factors must be identified and quantified so that accurate choices can be made about which effects require compensation. Control factors are used by the designer to compensate for noise factors that could significantly influence the system away from nominal performance.

Manuscript received November 02,2014

Victoria hebseba, Department of CSE, Shri Shirdi Sai Institute of Science and Engineering, Anantapur, A.P , India
B. Vijay Bhaskar Reddy, Assistant Professor, Department of CSE, Shri Shirdi Sai Institute of Science and Engineering,Anantapur, A.P, India

Once the critical noise factors are identified and the control factors selected, a Robust Design flow is used to implement and analyze the design to ensure system reliability.

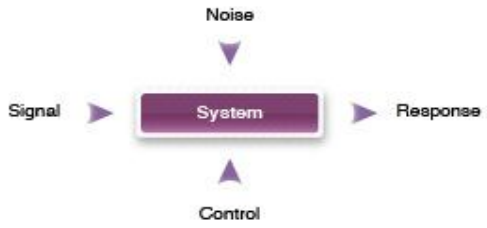

Fig 1: Robust Design

Adopting Robust Design principles to improve reliability means making system performance immune to variations in design technologies, component parameters, manufacturing processes, and operational conditions. In a Robust Design flow, these variations become the noise factors affecting system performance. The method of control for each variation may be as simple as selecting high precision components or as involved as implementing new control algorithms. The matrix of possible variations and control combinations becomes so complex that the traditional design-prototype-test flow is not practical. Designers must move their design activities to the virtual world, where powerful simulation tools support complete system design and verification using Robust Design techniques.

Implementing an effective and efficient Robust Design process requires simulation tools with specialized capabilities. The key tool requirements are simulation support, model library support, modeling language support, and advanced data analysis. A simulator must have special, built-in capabilities for each of the steps in the Robust Design process.

A. Simulator

- Handle "stiff systems">10 orders of magnitude of time constants
- IO buffer of IC, power electronics, magnetics, hydraulic, thermal
- DSPs, ECUs, D/A & A/D

B. Models, Modeling Tools, Languages

- Components, Generic, Industry (VDA), Custom
- State Diagrams, Characterization, and Multi-dimensional TLU
- MAST, VHDL-AMS, Spice

C. Analyses & Results Management

- Sensitivity, Statistical, Stress, Fault, Worst Case
- Grid computing
- Measurements, Calculations, Data analyses

D. Saber Advantages

- Efficient implementation of all Robust Design analyses
- Fast virtual system design supported by 30,000+ models
- Compute intensive statistical analyses performed with grid computing
- Accurate model creation with model characterization tools
- Increased model portability with model language standards VHDL-AMS & MAST
- Intellectual property protected with model encryption.

Logics [BCM+03], while DB was focusing on data management according to simple mathematical structures for the sake of efficiency, e.g., using the relational model [AHV95] or the extensible Markup Language [AMR+12]. In the beginning of the 21st century, these ideological stances have changed with the new era of ontology based data management.

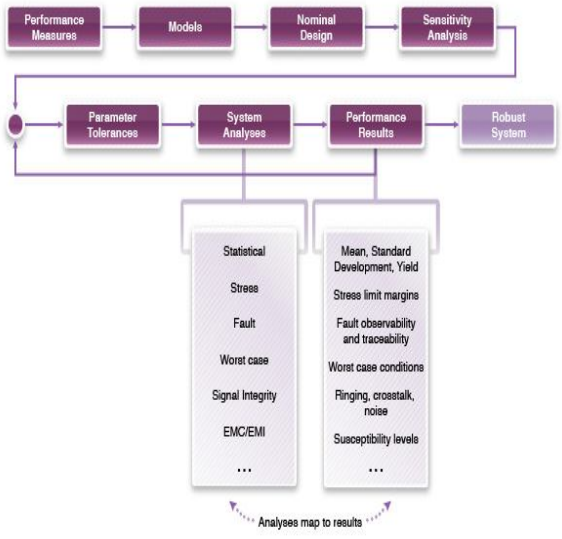

Fig 2: Robust Design Tool Requirements

II. RELATED WORK

Data management is a longstanding research topic in Knowledge Representation (KR), a prominent discipline of Artificial Intelligence (AI), and – of course – in Databases (DB).Till the end of the 20th century, there have been few interactions between these two research fields concerning data management, essentially because they were addressing it from different perspectives. KR was investigating data management according to human cognitive schemes for the sake of intelligibility, e.g., using Conceptual Graphs [CM08] or Description.

Ontology-based data management brings data management. One step closer to end-users, especially to those that are not computer scientists or engineers. It basically revisits the traditional architecture of database management systems by decoupling the models with which data is exposed to end-users from the models with which data is stored. Notably, ontology-based data management advocates the use of conceptual models from KR as human intelligible front-ends called ontologies [Gru09], relegating DB models to back-end storage. The World Wide Web Consortium (W3C) has greatly contributed to ontology-based data management by providing standards for handling data through ontologies, the two Semantic Web data models. The first standard, the Resource Description Framework (RDF) [W3Ca], was introduced in 1998. It's a graph data model coming with a very simple ontology language, RDF Schema, strongly related to description logics. The second standard, the Web Ontology Language (OWL) [W3Cd], was introduced in 2004. It's actually a family of well-established description logics with varying expressivity/complexity tradeoffs. The advent of RDF and OWL has rapidly focused the attention of academia and industry on practical ontology based

Data management. The research community has undertaken this challenge at the highest level, leading to pioneering and compelling contributions in top venues on Artificial Intelligence (e.g., AAAI, ECAI, IJCAI, and KR), on Databases (e.g., ICDT/EDBT, ICDE, SIGMOD/PODS, and VLDB), and on the Web (e.g., ESWC, ISWC, and WWW). Also, open-source and commercial software providers are releasing an ever-growing number of tools allowing effective RDF and OWL data management (e.g., Jena, ORACLE 10/11g, OWLIM, Prot´eg´e, RDF-3X, and Sesame). Last but not least, large societies have promptly adhered to RDF and OWL data management (e.g., library and information science, life science, and medicine), sustaining and begetting further efforts towards always more convenient, efficient, and scalable ontology-based data management techniques.

Introduce the basics of RDF and of the DL-lite family of description logics which underlies the OWL dialect dedicated to the management of large datasets: OWL2 QL. I describe for each the data model, query language, and prevalent techniques for the traditional data management tasks of consistency checking and query answering.

A. Design

The main contribution I present is robust module-based data management in DL-lite. Module-based data management amounts to handling data using an ontology – a module – that derives from that of a preexisting ontology-based data management application. I summarize the results from [GR10, GR12], which introduce the novel notion of robust module and show how to use it to enhance data integrity and to complement the answers to queries in ontology-based data management applications. The ongoing work I present next is the design of RDF query answering techniques that are robust to graph1 updates. I summarize the results from [GMR12b, GMR12a], which build on the prevalent saturation-based query answering technique and on the alternative reformulation-based query answering technique. A saturation maintenance technique is designed for the former to limit the necessary re-computation efforts upon graph updates, while the latter – de facto robust to updates – is extended to a larger fragment of RDF than those investigated in the literature.

B. Optimization

The main contribution I present is view selection for efficient RDF query answering. It amounts to tuning an RDF data management system to users' or applications' needs modeled as a query workload. The idea is to pre-compute and store the results for some automatically selected queries – the views – in order to minimize a combination of query processing, view storage, and view maintenance upon update costs. Summarize the results from [GKLM10b, GKLM10a, GKLM11a], which build on a state-of-the-art view selection technique for the relational data model. In particular, the view selection technique devised for RDF supports both saturation- and reformulation-based query answering, depending on how views are materialized. The ongoing work I present next is a first step towards efficient query answering against XML documents with RDF annotations. I summarize the results from [GKK+11a, GKK+12], which combine the XML and RDF data models and query languages into a uniform XML-RDF hybrid setting for managing annotated documents. In particular, we want to study to which extent query answering can be optimized by using at the same time the structural XML constraints (expected tree shape of the documents) and the semantic RDF constraints (expected ontological descriptions) expressed in queries.

III. DMS (Data Management Systems)

In this paper, we revisit the reuse of a reference ontology based DMS in order to build a new DMS with specific needs. We go one step further by not only considering the design of a module-based DMS (i.e., how to extract a module from a ontological schema): we also study how a module based DMS can benefit from the reference DMS to enhance its own data management skills. We carry out our investigations in the setting of DL-lite, which is the foundation of the QL profile of OWL2 recommended by the W3C for efficiently managing large RDF data sets. RDF is the W3C's Semantic Web data model, which is rapidly spreading in more and more applications, and can be seen as a simple relational model restricted to unary and binary predicates. In addition, DL-lite comes with efficient inference algorithms [11] for querying RDF data through (DL-lite) ontologies and for checking data consistency w.r.t. integrity constraints (ICs) expressed in DL-lite.

Our contribution is to introduce and study novel properties of robustness for modules that provide means for checking easily that a robust module-based DMS evolves safely w.r.t. both the schema and the data of the reference DMS. From a module robust to consistency checking, for any data update in a corresponding module-based DMS, we show how to query the reference DMS for checking whether the local update does not bring any inconsistency with the data and the constraints of the reference DMS. From a module robust to query answering, for any query asked to a module-based DMS, we show how to query the reference DMS for obtaining additional answers by also exploiting the data stored in the reference DMS. It is worth noticing that our investigations are sustained by real use cases. For instance, the My CF DMS has been built by hand from the FMA DMS. This step has focused on particular parts of the human body (e.g., hand, foot, and knee), while the personalization step has enriched the descriptions of these parts with both 3D geometrical and bio-mechanical information. Notably, careful attention was paid so that My CF still conforms to FMA at the end of the manual process. The paper is organized as follows: We start that highlights the issues and solutions on which we elaborate in the rest of the paper. In Section 3, we present the DL-lite description logic, which provides the formal basis of Section 4, in which we study robust modules and safe personalization. In Section 5, we provide algorithms and complexity results for extracting robust modules from schemas and for checking the safe personalization of modules.

IV. ILLUSTRATIVE EXAMPLE

Consider a reference DMS for scientific publications (like DBLP) defined by the ontological schema O and the data set D The schema O is built upon the unary relations. It consists of inclusion constraints and of integrity constraints (disjointness and functional constraints). These constraints using DL-lite.

RDF [W3Ca] is a graph data models that has been recommended by W3C since 1998. It allows defining graphs that can be queried with the SPARQL Protocol and RDF Query Language [W3Cc]). This language, SPARQL in short, has been recommended by W3C since 2008. The prevalent technique for answering SPARQL

queries against graphs is saturation-based query answering a graph is a set of triples of the form s p o: (the final dot preceded by a white space belongs to the normative triple syntax). A triple states that its subject s has the corresponding property p, and the value of that property is the object o. Given a set U of Uniform Resource Identifiers1 (URIs), a set L of literals (constants), and a set B of blank nodes (unknown URIs or literals), such that U, B and L are pairwise disjoint, a triple is well-formed whenever its subject belongs to U [B, its property belongs to U, and its object belongs to U [B [L. In the following, I only consider well-formed triples. Blank nodes are essential features of RDF allowing the support of incomplete information. For instance, one can use a blank node: b1 to state that the country of: b1 is France while the city of the same: b1 is Brest. Many such blank nodes can co-exist within a graph, e.g., one may also state that the country of: b2 is Romania while the city of: b2 is Timisoara; at the same time, the population of Timis¸oara can be said to be an unspecified constant Notations I use s, p, o and: b in triples (possibly with subscripts) as placeholders. That is, s stands for values in U [B, p stands for values in U, o represents values from U [B [L, and: b denotes values in B. Strings between quotes as in \string" denote literals. Finally, the set of values (URIs, blank nodes, literals) of a graph G is denoted val(G)

Table 1 shows how to use triples to describe resources; from now on, I use the name rdf for the normative RDF namespace2 when writing the URIs of classes and properties comprised in the RDF standard

Table 1: RDF statements.

Constructor	Triple
Class assertion	Rdf: type 0
Property assertions	S p 0:

A more intuitive representation of a graph can be drawn from its triples. Every (distinct) subject or object value is represented by a node labeled with this value. For each triple, there is a directed edge from the subject node to the object node, which is labeled with the property value. Example 1 (Running example)

The two representations are equivalent as they model the same graph G. The namespaces for user-defined classes and properties were omitted for the sake of readability. This graph describes the Digital Object Identifier3 doi1 that belongs to an unknown class, whose title (hasTitle) is “Complexity of Answering Queries Using Materialized Views”, whose author (hasAuthor) is “Serge Abiteboul” and which has an unknown contact author (has Contact A).

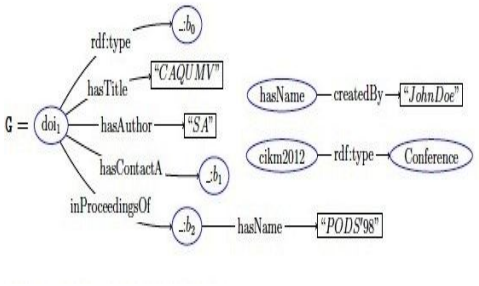

Figure 3: Graph representations.

A valuable feature of RDF is RDF Schema (RDFS) that allows enhancing the descriptions in RDF graphs. An RDF Schema declares semantic constraints between the classes and the properties used in graphs. Figure 2.3 shows the allowed constraints and how to express them; from now on, I use the name rdfs for the normative RDFS Name space when writing the URIs of classes and properties comprised in the RDFS standard.

Table2: RDFS Statements

Constructor	Triple
Subclass constraint s Subproperty constraint Domain typing constraint RangeTyping Constraints	Rdfs :SubClassOf o: S rdfs : sub Property Of o. s rdfs:domain o : s rdfs:range o :

RDFS statements expressing semantic constraints between classes and properties. Example 2 (Continued) Consider next to the above graph G, a schema stating that poster papers (posterCP) together with the unknown class :b0 are subclasses of conference papers (confP), which are scientific papers (paper). Moreover, titles (hasTitle), authors (hasAuthor), contact authors (hasContactA) – who are authors – are used to describe papers. Papers are also described by the conferences (conference) in whose proceedings (inProceedingsOf) they appear. Finally, names (hasName) describe conferences, and creators (createdBy) describe resources. The extended graph G0 of G corresponding to this schema is depicted in Figure 2.4. Entailment The W3C names RDF entailment the mechanism through which, based on the set of explicit triples and some entailment rules (to be described shortly), implicit triples are derived. I denote by `I RDF immediate entailment, i.e., the process of deriving new triples through a single application of an entailment rule. More generally, a triple s p o : is entailed by a graph G, denoted G `RDF s p o : if and only if there is a sequence of applications of immediate entailment rules that leads from G to s p o :, where at each step of the entailment sequence, the triples previously entailed are also taken into account.

The DL-lite family of Description Logics

The DL-lite family [CGL+07] of Descriptions Logics [BCM+03] allows defining knowledge bases that can be queried with the well-known conjunctive queries select-project-join queries, from the relational database theory [AHV95]. The prevalent techniques for consistency checking and query answering is first order logic (FOL) reducibility, which reduces these tasks to the evaluation of FOL queries .Knowledge bases Generally speaking, a DL knowledge base (KB) consists of a schema called a Tbox and its associated dataset called an Abox. A Tbox T is defined upon a signature denoted sig(T), which is the disjointunion of a set of unary relations called atomic concepts and a set of binary relations called atomic roles. A T box is a set of constraints called terminological axioms, typically inclusion constraints between complex concepts or roles, i.e., unary or binary DL formulae built upon atomic relations using the constructors allowed in DL under consideration. An Abox defined upon sig(T) is a set of facts called assertional axioms, relating DL formulae to their instances. The legal KBs vary according to the DL used to express terminological and assertional axioms, and to the restrictions imposed on those axioms. In DL-lite, the concepts and roles that can be built from atomic concepts and atomic roles are of the following form:

B ! A j 9R; C ! B j :B; R ! P j P□; E ! R j :R

where A denotes an atomic concept, P an atomic role, and P□ the inverse of P; B denotes a basic concept (i.e., an atomic concept A or an unqualified existential quantification on a basic role 9R) and R a basic role (i.e., an atomic role P or its inverse P□); C denotes a general concept (i.e., a basic concept or its negation) and E a general role (i.e., a basic role or its negation).
The (set) semantics of concepts and roles is given in terms of interpretations. An interpretation I Consists of a nonempty interpretation domain _I and an interpretation function: I that assigns a subset of _I to each atomic concept, and a binary relation over _I to each atomic role. The semantics of non-atomic concepts and non-atomic roles is defined as follows:

_ (P□)I = f(o2; o1) j (o1; o2) 2 PIg,

The axioms allowed in a Tbox of DL-lite are concept inclusion constraints of the form B v C, role inclusion constraints of the form R v E, and functionality constraints on roles of the form (funct R). Observe that negated concepts or roles are only allowed on the right hand side of inclusion constraints, whereas only positive concepts or roles occur on the left hand side of such constraints. Moreover, only basic roles occur in functionality constraints. Inclusions of the form B1 v B2 or R1 v R2 are called positive inclusions (PIs), while inclusions of the form B1 v :B2 or of the form R1 v :R2 are called negative inclusions (NIs). PIs allow expressing inclusion dependencies, while NIs and functionalities allow expressing integrity constraints (ICs). An interpretation I = (_I ; :I) is a model of an inclusion B v C (resp. R v E) if BI _ CI (resp. RI _ EI).
It is a model of a functionality constraint (funct R) if the binary relation RI is a function, i.e., (o; o1) 2 RI and (o; o2) 2 RI implies o1 = o2. I is a model of a Tbox if it is a model of all of its constraints. A Tbox is satisfiable if it has a model. A Tbox T logically entails (a.k.a. implies) a constraint _, written T j= _, if every model of T is a model of _. Finally, a Tbox T logically entails (a.k.a. implies) a Tbox T0, written T j= T0, if every model of T is a model of T0; and two Tboxes T and T0 are logically equivalent, written T _ T0, iff T j= T0 and T0 j= T.

T representing scientific publications.

Example 5 (Running example) Consider the Tbox T in Figure 2.5, representing domain knowledge about scientific publications. Its signature sig(T) consists of the atomic concepts Publication, ConfPaper, ShortPaper, FullPaper, JournPaper, Survey, and of the atomic roles hasTitle, hasDate, hasVenue, and hasAuthor. The constraints in T state that any publication has a single title (1), a single date of publication (2), a single venue (3), and at least one author (4). In addition, only publications have a title (5), papers in conference proceedings or in journals (which are disjoint) are publications (6), short papers or full papers (which are disjoint) are papers in conference proceedings, and surveys are journal papers (7). The Tbox implies the constraint , which means that a journal paper has at least one author, as it contains JournPaper v Publication and Publication v 9hasAuthor. It also implies the constraint FullPaper v :Survey, which means that surveys and full papers are disjoint,
as it contains FullPaper v ConfPaper, ConfPaper v :JournPaper, and Survey v JournPaper.
An Abox consists of a finite set of membership assertions of the form A(a) and P(a; b), i.e., on atomic
concepts and on atomic roles, stating respectively that a is an instance of A and that the pair of constants (a; b)
is an instance of P. The interpretation function of an interpretation I = (_I ; :I) is extended to constants by
assigning to each constant a a distinct object aI 2 _I , i.e., the so called unique name assumption holds. An
Interpretation I is a model of the membership assertion A (a) (resp. P (a; b)) if aI 2 AI (resp., (aI ; bI) 2 PI). It is a model of an Abox if it satisfies all of its assertions.
Example 6 (Continued) Consider the Abox in Figure 2.6, representing factual knowledge about scientific publications. It is expressed as relational tables and states in particular that:
_ doi1 is the Digital Object Identifier4 (DOI) of the full paper entitled ”Complexity of Answering Queries Using Materialized Views” and published in PODS’98 by Serge Abiteboul ("SA") and Oliver M. Duschka ("OD"),
_ doi2 is the DOI of the survey entitled” Answering queries using views: A survey” and published in VLDB Journal in 2001 by Alon Y. Halevy ("AH"), and _ doi3 is the DOI of the journal paper entitled MiniCon: A scalable algorithm for answering queries and published in VLDB Journal in 2001 by Rachel Pottinger ("RP") and Alon Y. Halevy ("AH").
A KB K is a pair made of a Tbox T and an Abox A, denoted K = hT; Ai. An interpretation I is a model of

a KB K = hT; Ai if it is a model of both T and A. A KB K is satisfiable, a.k.a. consistent, if it has at least one model. Observe that Tboxes and Aboxes are always consistent. That is, a KB is inconsistent whenever there is a contradiction between its Abox and Tbox. A KB K logically entails, a.k.a. implies, a constraint or assertion Written K j= _, if every model of K is a model of _. Example 7 (Continued) Consider the consistent KB K = hT; Ai associating the above Tbox and Abox for scientific publications. KB can be written equivalently as a FOL KB and a relational database following the openworld assumption (OWA) [AHV95]. The correspondences for Tbox constraints are summarized in Figure 2.7 for PIs, in Figure 2.8 for NIs, and in Figure 2.9 for functionalities. As for Abox assertions, they are simply FOL facts (i.e., ground atoms) and instances for atomic concepts and roles.

Table 3: DL-lite PI

DL notation	FOL notation	Relational notation (OWA)
$A \sqsubseteq A'$	$\forall x[A(x) \Rightarrow A'(x)]$	$A \subseteq A'$
$A \sqsubseteq \exists P$	$\forall x[A(x) \Rightarrow \exists y P(x,y)]$	$A \subseteq \Pi_1(P)$
$A \sqsubseteq \exists P^-$	$\forall x[A(x) \Rightarrow \exists y P(y,x)]$	$A \subseteq \Pi_2(P)$
$\exists P \sqsubseteq A$	$\forall x[\exists y P(x,y) \Rightarrow A(x)]$	$\Pi_1(P) \subseteq A$
$\exists P^- \sqsubseteq A$	$\forall x[\exists y P(y,x) \Rightarrow A(x)]$	$\Pi_2(P) \subseteq A$
$\exists Q \sqsubseteq \exists P$	$\forall x[\exists y Q(x,y) \Rightarrow \exists z P(x,z)]$	$\Pi_1(Q) \subseteq \Pi_1(P)$
$\exists Q \sqsubseteq \exists P^-$	$\forall x[\exists y Q(x,y) \Rightarrow \exists z P(z,x)]$	$\Pi_1(Q) \subseteq \Pi_2(P)$
$\exists Q^- \sqsubseteq \exists P$	$\forall x[\exists y Q(y,x) \Rightarrow \exists z P(x,z)]$	$\Pi_2(Q) \subseteq \Pi_1(P)$
$\exists Q^- \sqsubseteq \exists P^-$	$\forall x[\exists y Q(y,x) \Rightarrow \exists z P(z,x)]$	$\Pi_2(Q) \subseteq \Pi_2(P)$
$P \sqsubseteq Q^-$ or $P^- \sqsubseteq Q$	$\forall x,y[P(x,y) \Rightarrow Q(y,x)]$	$P \subseteq \Pi_{2,1}(Q)$ or $\Pi_{2,1}(P) \subseteq Q$
$P \sqsubseteq Q$ or $P^- \sqsubseteq Q^-$	$\forall x,y[P(x,y) \Rightarrow Q(x,y)]$	$P \subseteq Q$ or $\Pi_{2,1}(P) \subseteq \Pi_{2,1}(Q)$

PI axioms in FOL and relational notations. For the relational notation, which corresponds to unary and binary inclusion dependencies, we assume that the first and second attributes of any atomic role are named 1 and 2 respectively.

DL notation	FOL notation	Relational notation (OWA)
$A \sqsubseteq \neg A'$	$\forall x[A(x) \Rightarrow \neg A'(x)]$	$A \cap A' \subseteq \bot$
$A \sqsubseteq \neg\exists P$	$\forall x[A(x) \Rightarrow \neg\exists y P(x,y)]$	$A \cap \Pi_1(P) \subseteq \bot$
$A \sqsubseteq \neg\exists P^-$	$\forall x[A(x) \Rightarrow \neg\exists y P(y,x)]$	$A \cap \Pi_2(P) \subseteq \bot$
$\exists P \sqsubseteq \neg A$	$\forall x[\exists y P(x,y) \Rightarrow \neg A(x)]$	$A \cap \Pi_1(P) \subseteq \bot$
$\exists P^- \sqsubseteq \neg A$	$\forall x[\exists y P(y,x) \Rightarrow \neg A(x)]$	$A \cap \Pi_2(P) \subseteq \bot$
$\exists Q \sqsubseteq \neg\exists P$	$\forall x[\exists y Q(x,y) \Rightarrow \neg\exists z P(x,z)]$	$\Pi_1(Q) \cap \Pi_1(P) \subseteq \bot$
$\exists Q \sqsubseteq \neg\exists P^-$	$\forall x[\exists y Q(x,y) \Rightarrow \neg\exists z P(z,x)]$	$\Pi_1(Q) \cap \Pi_2(P) \subseteq \bot$
$\exists Q^- \sqsubseteq \neg\exists P$	$\forall x[\exists y Q(y,x) \Rightarrow \neg\exists z P(x,z)]$	$\Pi_2(Q) \cap \Pi_1(P) \subseteq \bot$
$\exists Q^- \sqsubseteq \neg\exists P^-$	$\forall x[Q(y,x) \Rightarrow \neg\exists z P(z,x)]$	$\Pi_2(Q) \cap \Pi_2(P) \subseteq \bot$
$P \sqsubseteq \neg Q^-$ or $P^- \sqsubseteq \neg Q$	$\forall x,y[P(x,y) \Rightarrow \neg Q(y,x)]$	$P \cap \Pi_{2,1}(Q) \subseteq \bot$ or $\Pi_{2,1}(P) \cap Q \subseteq \bot$
$P \sqsubseteq \neg Q$ or $P^- \sqsubseteq \neg Q^-$	$\forall x,y[P(x,y) \Rightarrow \neg Q(x,y)]$	$P \cap Q \subseteq \bot$ or $\Pi_{2,1}(P) \cap \Pi_{2,1}(Q) \subseteq \bot$

Figure 4: DL-lite NI

NI axioms in FOL and relational notations. For the relational notation, which corresponds to exclusion/disjointness dependencies, we assume that the first and second attributes of any atomic roles.

Table 4: DL-lite FOL notations

DL notation	FOL notation	Relational notation (OWA)
$(funct\ P)$	$\forall x,y,z[P(x,y) \wedge P(x,z) \Rightarrow y = z]$	$P: 1 \rightarrow 2$
$(funct\ P^-)$	$\forall x,y,z[P(y,x) \wedge P(z,x) \Rightarrow y = z]$	$P: 2 \rightarrow 1$

Functionality axioms in FOL and relational notations. For the relational notation, which corresponds to functional dependencies, we assume that the first and second attributes of any atomic role are named 1 and 2 respectively.

Queries

A FOL query q is of the form q (_x):- _ (_x) where _ (_x) is a FOL formula, the free variables of which are only the variables _x, and the predicates of which are either atomic concepts or roles. The arty of a query is the number of its free variables, e.g., 0 for a Boolean query. When _ (_x) is of the form 9_y conj (_x; _y) with conj (_x; _y) a Conjunction of atoms, q is called a conjunctive query. Conjunctive queries, a.k.a. select project-join queries, are the core relational database queries. Given an interpretation

I = (_I ; :I), the semantics qI of a boolean query q is defined as true if [_(;)]I = true, and false otherwise, while the semantics qI of a query q of arity n _ 1 is the relation of arity n defined on _I as follows: qI = f_e 2 (_I)n j [_(_e)]I = true g. An interpretation that evaluates a Boolean query to true. A non-Boolean query to a non-empty set, is a model of that query.

NI	Corresponding $unsat$ query
$A \sqsubseteq \neg A'$ or $A' \sqsubseteq \neg A$	$\exists x[A(x) \wedge A'(x)]$
$A \sqsubseteq \neg\exists P$ or $\exists P \sqsubseteq \neg A$	$\exists x,y[A(x) \wedge P(x,y)]$
$A \sqsubseteq \neg\exists P^-$ or $\exists P^- \sqsubseteq \neg A$	$\exists x,y[A(x) \wedge P(y,x)]$
$\exists Q \sqsubseteq \neg\exists P$ or $\exists P \sqsubseteq \neg\exists Q$	$\exists x,y,z[Q(x,y) \wedge P(x,z)]$
$\exists Q \sqsubseteq \neg\exists P^-$ or $\exists P^- \sqsubseteq \neg\exists Q$	$\exists x,y,z[Q(x,y) \wedge P(z,x)]$
$\exists Q^- \sqsubseteq \neg\exists P$ or $\exists P \sqsubseteq \neg\exists Q^-$	$\exists x,y,z[Q(y,x) \wedge P(x,z)]$
$\exists Q^- \sqsubseteq \neg\exists P^-$	$\exists x,y,z[Q(y,x) \wedge P(z,x)]$
$P \sqsubseteq \neg Q^-$ or $Q^- \sqsubseteq \neg P$ or $P^- \sqsubseteq \neg Q$ or $Q \sqsubseteq P^-$	$\exists x,y[P(x,y) \wedge Q(y,x)]$
$P \sqsubseteq \neg Q$ or $Q \sqsubseteq \neg P$ or $P^- \sqsubseteq \neg Q^-$ or $Q^- \sqsubseteq \neg P^-$	$\exists x,y[P(x,y) \wedge Q(x,y)]$

Figure 5 NI axioms to queries

Against a KB K = hT; Ai. If q is non-boolean, the answer set of q against K is defined as: ans (q;K) = f_t 2 Cn j K j= q(_t)g where C is the set of constants appearing in the KB, q(_t) is the closed formula obtained by replacing in the query definition the free variables in _x by the constants in _t, and K j= q(_t) means as usual that every model of K is a model of q(_t). If q is boolean, the answer set of q against K is by convention either ftrueg or ffalseg: ans(q;K) = ftrueg if and only if K j= q(), i.e., every model of K is a model of q(). This corresponds to the so-called certain answers semantics requiring that an answer to a query, given a set of constraints (expressed here as a Tbox), to be an answer in all the models satisfying the constraints.

V. CONCLUSION

The Robust module based data management In addition, in contrast with existing work, we have considered the problem of safe personalization of modules built from an existing reference DMS. This raises new issues to check easily that a module-based DMS evolves independently but coherently w.r.t. the reference DMS from which it has been built. We have introduced two notions of module robustness that make possible to build locally the relevant queries to ask to the reference database in order to check global consistency (possibly upon each update), and to obtain global answers for local queries. We have provided internal time algorithms that extract minimal and robust modules from a reference ontological schema. Global query answering, applies under the severe constraints that the data set of the reference DMS has to be modified (write access is required). While keeping data consistency and query answering reducible to standard database queries. Efficiently recommended for managing large data sets as the search and retrieval of the data is optimized.

REFERENCES

[1] Telser, S., Staudacher, M., Ploner, Y., Amann, A., and Hinterhuber, H, and Ritsch-Marte, M. Can One Detect Sleep Stage Transitions for On-Line Sleep Scoring by Monitoring the Heart Rate Variability? Somnologie, Vol. 8,2004,

[2] Vlachos, M., Hadjieleftheriou, M., Gunopulos, D. and Keogh, E. Indexing Multi-Dimensional Time-Series with Support for Multiple Distance Measures, Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM Press, Washington. DC, 2003.

[3] Lin, J., Keogh, E., Lonardi, S., Lankford, J. P., and Nystrom, D. M. Visually Mining and Monitoring Massive Time Series, Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM Press, Seattle.

[4] R. Kontchakov, L. Pulina, U. Sattler, T. Schneider, P. Selmer, F. Wolter, and M. Zakharyaschev, "Minimal Module Extraction from DL-Lite Ontologies Using QBF Solvers," Proc. 21st Int'l Joint Conf. Artificial Intelligence (IJCAI), 2009.

[5] Pyle, D. Data Preparation for Data Mining, Morgan Kaufmann, San Francisco, 1999

[6] B. Konev, D. Walther, and F. Wolter, "Forgetting and Uniform Interpolation in Extensions of the Description Logic EL," Proc. 22nd Int'l Workshop Description Logics, 2009.

[7] B. Konev, C. Lutz, D. Walther, and F. Wolter, "Semantic Modularity and Module Extraction in Description Logics," Proc. 18th European Conf. Artificial Intelligence (ECAI), 2008.

[8] Fukami, T., Emori, R., Shimada, T., Akatsuka, T., and Saito, Y. The Detection of EEG Characteristic Waves by Using Locally Stationary Autoregressive Models, Trans. IEE Japan, Vol. 121-C, No.3, in Japanese 2001

[9] B. Cuenca Grau, I. Horrocks, Y. Kazakov, and U. Sattler, "Just the Right Amount: Extracting Modules from Ontologies," Proc. 16th Int'l Conf. World Wide Web (WWW), 2007.

[10] K. Wang, Z. Wang, R.W. Topor, J.Z. Pan, and G. Antoniou, "Concept and Role Forgetting in ALC Ontologies," Proc. Eighth Int'l Semantic Web Conf. (ISWC), 2009.

[11] http://www.data-miner.com/. Michael J.

[12] A. Berry, Gordon S. Linoff – Mastering Data Mining – Wiley Computer Publishing – 2001