OWL2 based Data Cleansing Using Conditional Exclusion Dependencies

Olivier CURE, Chan LE DUC and Myriam LAMOLLE

Université Marne La Valée, Université Paris8 - IUT de Montreuil
Our goal is to use the Ontology Based Data Access (OBDA) approach to improve data quality of underlying databases.

This is performed with dependencies that capture real world inconsistencies: conditional dependencies.

Two forms have been studied: Conditional Functional Dependencies and Conditional Inclusion Dependencies (CINDs).

We propose a novel one: Conditional Exclusion Dependencies (CEDs).
A CED is an extension of Exclusion Dependencies that forbids the appearance of tuples in S when tuples satisfying some patterns appear in R.
A CED is an extension of Exclusion Dependencies that forbids the appearance of tuples in S when tuples satisfying some patterns appear in R.

**Syntax:**

- A CED $\phi$, defined over a pair of relations $R$ and $S$, is a pair $(R(X; X_p) \subseteq \neg S(Y; Y_p), T_p)$ where $X, X_p$ and $Y, Y_p$ are attribute sets of respectively $R$ and $S$.
- $R(X) \subseteq \neg S(Y)$ is a standard exclusion dependency and $T_p$ is a tableau pattern of $\phi$ with attribute sets $X_p$ and $Y_p$ such that for each pattern $t_p$ and each attribute $B$ in $X_p$ and $Y_p$, $t_p[B]$ is either a constant in the domain of $B$ or a wild card, denoted ‘.’.
Semantics:

An instance \((l_1, l_2)\) of \((R, S)\) satisfies a CED \(\phi\), denoted \((l_1, l_2) \vDash \phi\), iff for each tuple \(t_p\) in \(T_p\) and for each \(t_1\) in \(l_1\), if \(t_1[X_p] = t_p[X_p]\) then there does not exist a tuple \(t_2\) in \(S\) such that \(t_1[X] = t_2[Y]\) and \(t_2[Y_p] = t_p[Y_p]\).
Discovery of (C)ED is a hard problem since RDBMS do not store negative facts.

OWL2 corresponds to the SROIQ description logic and allows for new role constructors.

Some of them can be used to discover CEDs:

- RBox axioms of the form $R \sqsubseteq \neg S$ with $R$ and $S$ DL roles.
- General Concept Inclusion (GCI) of the form $\exists R.C \sqsubseteq \neg S.D$ with $C$ a nominal and $D$ a nominal or $\top$.
- Negative property assertions.
CEDs are represented using the formalism of SPARQL.

They aim to detect violations of CEDs and are generated by considering a CED has a graph over elements of the domain ontology.

In this graph, the negated property is asserted to be true. Thus a translation of this graph into an SPARQL query enables to detect objects being violated.

We compact these queries whenever some tableaux are compatible and/or using elements of concept hierarchies.
Detecting violation

- Violation detections of a CED violation are activated whenever a tuple of the data sources is updated, via an SQL query.
- This is handled by the definition of SQL triggers.
- We automatically generate an BEFORE/ROW LEVEL SQL trigger for each relation mapped to a property involved in a CED.
Future Work

- Propose algorithms to generate and tune CED using instances of the database
- Study interactions between conditional functional, inclusion and exclusion dependencies
- Test efficiency of the approach in larger application domains
Questions?

Thank you

ocure@univ-paris-est.fr