Semantic Web Technologies

Lecture 9: SWT for the Life Sciences 2: Successes and challenges for ontologies

Maria Keet
email: keet -AT- inf.unibz.it
home: http://www.meteck.org
blog:
http://keet.wordpress.com/category/computer-science/72010-semwebtech/

KRDB Research Centre
Free University of Bozen-Bolzano, Italy

21 December 2009

Outline

When can SWT considered to be successful?

- Only if Berners-Lee's vision of the SemWeb (as in the SciAm 2001 paper) has been realised?
- Absolute measures? e.g.,
  - User's (browsing and buying) usage of Amazon's recommender system with and without SWT
  - Information retrieval: compare precision and recall between a statistics-based and a SWT-based implementation of a document system
  - Feasibility and performance of a set of user queries posed to a RDBMS and its RDF-ised version
- Relative measures
  - According to whom is it a success?
    - philosopher, logician, engineer, domain expert, CEO...
  - What was taken as baseline material? e.g.,
    - from string search in a digital library to ontology-annotated sorting of query answer
    - no or clustering-based instance classification to a SWT one with OWL-based knowledge bases

Examples in different application areas, using different features

- Data integration
- Instance classification (example today)
- Matchmaking and services
- Querying, information retrieval
  - Ontology-Based Data Access (example today)
  - Aid in browsing large ontologies
  - Ontologies to improve NLP (more tomorrow)

SWT challenges or failures?

- Challenge: solution to problem y not possible yet (or very difficult to achieve) with current SWT, but in theory is (expected to be) feasible
- Failure: technology x claims to solve problem y but it does not and will not do so, or technology x is developed for a non-existing problem but does not solve real problems
  - Is y one that, at least in theory, can be solved with SWT?
  - Was y described too broadly, so that it solves only a subset of the cases?
  - Were there perhaps additional requirements put on a solution?
- Are disconnected technologies with ad-hoc patches a challenge to solve or a failure in devising a generic suite?
- A failure according to one may be considered a challenge by another
- Offer and demand, perceptions, perspectives, expectations

Instance classification with protein phosphatases (Wolstencroft et al, 2007)

- The setting:
  - Lots of sequence data in data silos that needs to be enriched with biological knowledge
  - Need to organise and classify genes and proteins into functional groups to compare typical properties across species
- The problems:
  - There is no proper, real life, use case that demonstrates the benefits of DL reasoning services such as taxonomic and instance classification
  - Limitations of traditional similarity methods, and automated protein motif and domain matching
  - Automation of p-domain analysis, but not for its interpretation (i..e, detects presence but not consequences for sub-family membership)

Idea

- Maybe OWL reasoning can help with the interpretation of the analysis results:
  - That it does the classification of the (family of) proteins as good as a human expert for organisms x (in casu, human)
  - That the approach is 'transportable' to classification of the (family of) proteins in another organism of which much less is known (in casu, *Aspergillus fumigatus*), hence make predictions for those instances by means of classifying them
- Use taxonomic classification and instance classification reasoning services

How it can be done

- Develop ontology for the subject domain, in OWL
  - Extract knowledge from peer-reviewed literature
  - Protein phosphatases; e.g.

```
Class R5Phosphatase Complete
  (Protein and
   (hasDomain two TyrosinePhosphataseCatalyticDomain) and
   (hasDomain some TransmembraneDomain) and
   (hasDomain some FibronectinDomain) and
    (hasDomain some CarbonicAnhydraseDomain) and
    hasDomain only (TyrosinePhosphataseCatalyticDomain and
     TransmembraneDomain and
     CarbonicAnhydraseDomain))
```

- Obtain instance data
  - Process protein sequences by InterProScan
  - Transform into OWL
- Put it together in some system with a reasoner
  - InstanceStore
  - Racer reasoner

Results

- Human phosphatases:
  - The reasoner as good as human expert classification
  - Identification of additional p-domains, refined the classification into further subtypes
- *A. fumigatus* phosphatates:
  - Some phosphatases did not fit in any class, representing differences between the human and *A. fumigatus* protein families
  - Identification of a novel type of calcineurin phosphatase (has extra domain, like in other pathogenic fungi)
- Overall: demonstration that ontology-based approach with automated reasoning has some advantages over (in addition to the) existing technologies & human labour, and resulted in discovery of novel biological information

Idea

- Ontology-Based Data Access, to achieve data access at the 'what-layer', i.e., adding a semantic layer to the database
- Web-based, like most other bioinformatics resources
- Graphical querying to make it usable by the domain expert
- Usage of, mainly, reasoning services for querying the ontology and the data

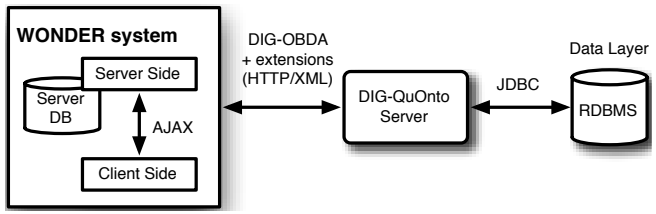
Web-based, graphical, ontology-based querying of lots of data (Calvanese et al, 2010)

- The setting:
  - Large amounts of data available on the Web, which can be accessed by canned or precomputed queries presented via web forms, or SQL
  - Domain expert wants more flexibility in data analysis and hypothesis testing, and independence from the sysadmin to do the queries for them
- The problems:
  - There is no proper, real life, use case that demonstrates the benefits of scalable, user-usable, Ontology-Based Data Access
  - That one has to know *how* the data is stored, instead of concerning oneself with *what* kind of information is in the database
  - Domain expert-unfriendly query mechanisms (SQL, SPARQL)

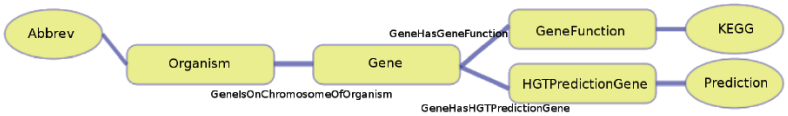
How it can be done

- Develop ontology of the subject domain, in OWL
  - Reverse engineering existing database HGT-DB (`http://genomes.urv.cat/HGT-DB/`), further manual improvements to create a proper conceptual data model
  - Simplify this conceptual data model into the appropriate OWL language (*DL-Lite*$_{\mathcal{A}}$, which is roughly OWL 2 QL)
- Create mappings between the terms in the ontology to SQL queries over the database
  - Using the OBDA Plugin for Protégé
  - Oracle database (can also be PostgreSQL, DB2, ...), 4GB genomics database (HGT-DB), tables with 16-46 columns
- Connect this to an OBDA-enabled reasoner
  - In this case: QuOnto (but can be others)

Architecture



Example: Diagram – *DL-lite*$_\mathcal{A}$ correspondence



$\delta(Abbrev) \sqsubseteq Organism$	An abbreviation is for an organism
$\rho(Abbrev) \sqsubseteq$ **xsd:string**	An abbreviation is of type string
$Organism \sqsubseteq \delta(Abbrev)$	Each organism has an abbreviation
(funct $Abbrev$)	Each individual has a single abbreviation
$\exists GeneIsOnChromosomeOfOrganism \sqsubseteq Gene$	Domain of object property
$\exists GeneIsOnChromosomeOfOrganism^- \sqsubseteq Organism$	Range of object property
$Gene \sqsubseteq \exists GeneIsOnChromosomeOfOrganism$	Each gene belongs to some organism
$Organism \sqsubseteq \exists GeneIsOnChromosomeOfOrganism^-$	Each organism has some gene
(funct $GeneIsOnChrOfOrganism$)	Each gene belongs to at most one organism
...	

Figure 2: Section of the HGT application ontology.

Formalisation of the graphical elements

Element name	Graphical Representation	Semantics of ontology elements	Semantics of building blocks of query graphs
Class node	C	C	$C(x)$
	C, D		$C(x), D(x)$
Is-a link	C — D	$C \sqsubseteq D$	
Attribute node and link	C — A	$\delta(A) \sqsubseteq C$ $\rho(A) \sqsubseteq \top_d$	$C(x),\ A(x,y)$
Role link	C — P — D	$\exists P \sqsubseteq C$ $\exists P^- \sqsubseteq D$	$C(x),\ R(x,y),\ D(y)$

Example: mapping concepts & relations of the Ontology to SQL query over the relational database

`SELECT id, abbrev`
`FROM organism`
`JOIN genes`
`ON abbrev = idorganism`
	$\rightsquigarrow$	$OrganismHasGene($
		gene(id),
		organism$(abbrev))$

`SELECT id, kegg` `FROM genes`	$\rightsquigarrow$	$GeneHasGeneFunction($ **gene**(id), **function**$(id))$ $KEGG($**function**$(id), kegg)$

Figure 3: Extract of the mapping from the HGT-DB database to the *DL-Lite*$_\mathcal{A}$ application ontology.

Queries

- SPARQL queries for conjunctions and equalities
- Epistemic queries in *EQL-Lite* for constraints involving inequalities and string matching
  - Imposes constraints on top of the certain answers retrieved by a *DL-Lite*$_A$ conjunctive query
  - Result obtained by:
    i. computing the certain answers for the CQ $q(\vec{y}) \leftarrow conj(\vec{z})$ (with $conj(\vec{z})$ the conjunction of atoms, and $\vec{y}$ a vector comprising the variables in $\vec{x}$ and in $\vec{w}$),
    ii. filtering the resulting tuples according to the constraint expression $cons(\vec{w})$, and
    iii. projecting onto $\vec{x}$ (a vector comprising the variables corresponding to the highlighted nodes in the query pane)

Results

- Demo of the WONDER system (Web-ONtology baseD Extraction of Relational data)
  - Builds upon the theory, technology, and implementation developed for Ontology-Based Data Access
  - Graphical ontology browsing, query formulation, and query execution in a Web browser
  - Rigorous formal characterisation and uses a coupling with an OWL file
  - (U)CQs (in SPARQL syntax) and EQL-Lite queries managed by the DIG-QUONTO reasoner
- Performance good, GUI insignificant influence on performance
- Usability testing: usable, and domain experts came up with a range of new queries to analyse the data

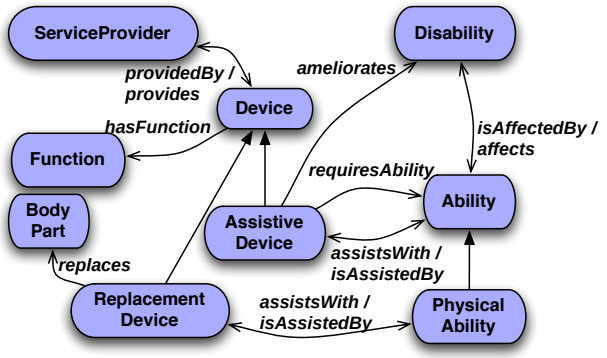
Additional features

- WONDER currently focuses on querying one database
- OBDA architecture allows for querying incomplete data (data integration scenario[1])
- Querying of the application ontology itself, as well as a combination of querying the ontology and the data[2]
  - in certain settings, possible to include queries that use the knowledge in the ontology for which there is no data in the database, and still retrieve the right results

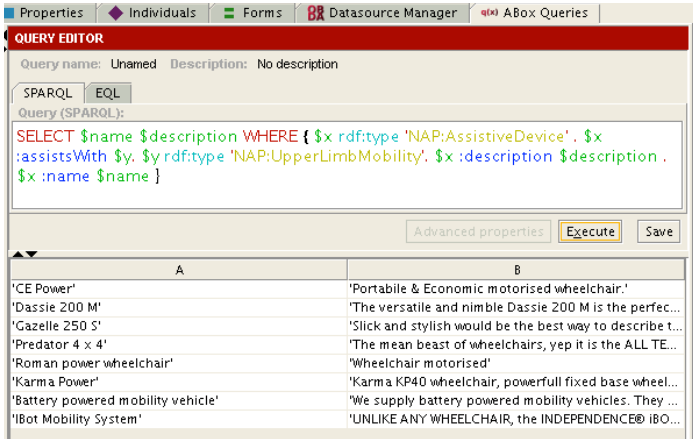
[1] A. Amoroso, G. Esposito, D. Lembo, P. Urbano, and R. Vertucci. Ontology-based data integration with MASTRO-I for configuration and data management at SELEX Sistemi Integrati. In Proc. of SEBD 2008, pages 8192, 2008.

[2] C. M. Keet, R. Alberts, A. Gerber, and G. Chimamiwa. Enhancing web portals with Ontology-Based Data Access: the case study of South Africa's Accessibility Portal for people with disabilities. In Proc. of OWLED 2008, 2008. CEUR-WS Vol 432

Informal overview of kind of knowledge in ADOLENA



Sample query in OBDA Plugin




```
q(x) :- Device(x), assistsWith(x, y), UpperLimbMobility(y)
```


A few general issues

- RDF triple stores vs. RDBMSs vs OWL ABoxes in memory; more generally:
  - Making 'legacy' (operational) systems 'Semantic Web compliant'
  - Add a 'wrapper' over the legacy system so that from the outside it looks like it uses SWT
- How to integrate rules other than at instance level
- Modularization
- Semantics-based language transformations
- Coordination among tools with different functionalities

Language limitations considerations

- Known trade-offs between expressiveness and computational complexity
- Different ontology developers and their scopes (and purposes of the ontologies):
  - to some, there is more in OWL/OWL2 than needed and used (recollect slide 32 of lecture 8)
  - to some, there is not enough (some of the limitations and extensions discussed in lecture 2, 6 and 7)
- From a logician's perspective, language limitations are not failures per sé, only *challenges* to find the more interesting and useful combinations of features
- From a modeller's perspective, the trade-offs can be such that it is deemed a *failure* with respect to the expectations and application needs

Limitations as identified by users/modellers (Schulz et al, 2009)

- n-ary relations, where $n > 2$
- "Hepatitis hasSymptom Fever in most but not all cases"
  - What about doing it with probabilistic default knowledge (lecture 7)?
  - $(\psi \mid \phi)[l, u]$ as "generally, if an object belongs to ϕ, then it belongs to ψ with a probability in $[l, u]$"
  - e.g., $(\exists \texttt{hasSymptom.Fever} \mid \texttt{Hepatitis})[1, 1]$
- "In 2000, worldwide prevalence of diabetes mellitus was 2.8%"
  - Probabilistic, or arithmetic, or what have we?
  - First, it assumes some class `Human` and a class `HumanDiabetesMellitus`, where some of the instances of the former have (are `bearerOf`) an instance of the latter
  - Second, we have some notion of `prevalence`, but what is it associated to (a property of)? of the human *population* in the world, not a property of an individual human

Limitations as identified by users/modellers (Schulz et al, 2009)

- ... Diabetes example continued
  - Authors' proposal to put it in the ABox with arithmetic operators, e.g. "$\frac{|DiabeticHuman|}{|Human|} = 0.028$"
  - Another option: put in TBox with a data property, e.g., $\texttt{HumanDiabetesMellitus} \sqsubseteq \exists \texttt{hasPrevalence.real}$
  - Yet another: represent the *probability* of a human having diabetes mellitus
  - What are the pros and cons of each option w.r.t. subject domain semantics, Ontology, and the ontology languages?
- Problems with Drug Abuse Prevention (in SNOMED CT)
  - $\texttt{DrugAbusePrevention} \sqsubseteq \texttt{Procedure} \sqcap \exists \texttt{hasFocus.DrugAbuse}$
  - $\texttt{DrugAbusePrevention} \equiv \texttt{Procedure} \sqcap \exists \texttt{hasParticipant.Person} \sqcap \exists \texttt{causes.(State} \sqcap \texttt{hasParticipant.(Person} \sqcap \exists \texttt{participatesIn.} \neg \texttt{DrugAbuse))}$

Limitations as identified by users/modellers (Schulz et al, 2009)

- "Concussion of the brain without loss of consciousness", and the temporal aspects (recollect lecture 6)
- "aspirin prevents myocardial infarction"
  - Let us assume that is total prevention (though we could add a probability to it)
  - This only holds for humans actually ingesting aspirin, not for the substance itself
  - It then intends to say that the human taking aspirin will not have a myocardial infarction *at all times in the future*, which can be represented in a suitable temporal logic with the $\Box^+$
  - e.g., $\texttt{AspirinIntake} \sqsubseteq \Box^+ \texttt{prevents.MyocardialInfarction}$, or $\texttt{MyocardialInfarction} \sqsubseteq \Box^+ \texttt{preventedBy.AspirinIntake}$, or $\texttt{AspirinIntake} \sqsubseteq \Box^+ \texttt{hasPhysiologicalEffect.} \neg \texttt{MyocardialInfarction}$?

- The standard reasoning services (recollect lecture 5) are obviously sorted out
- Performance issues for the 'debugging' and explanation reasoning, and how to provide the 'best' explanation
- Querying OWL 2 DL, and any ABox data
- Additional reasoning scenarios

Scenarios

1. Supporting the ontology development process
2. Classification
3. Model checking (violation)
4. Finding gaps in an ontology & discovering new relations
   - Deriving types and relations from instance-level data
   - Computing derived relations at the type level
5. Comparison of two ontologies ([logical] theories)
6. Reasoning with part-whole relations
7. Using (including finding inconsistencies in) a hierarchy of relations
8. Reasoning across linked ontologies
9. Complex queries

explanation and examples in: Keet, C.M., Roos, M. and Marshall, M.S. A survey of requirements for automated reasoning services for bio-ontologies in OWL. Third international Workshop OWL: Experiences and Directions (OWLED 2007), 6-7 June 2007, Innsbruck, Austria. CEUR-WS Vol-258.

Checking against instances

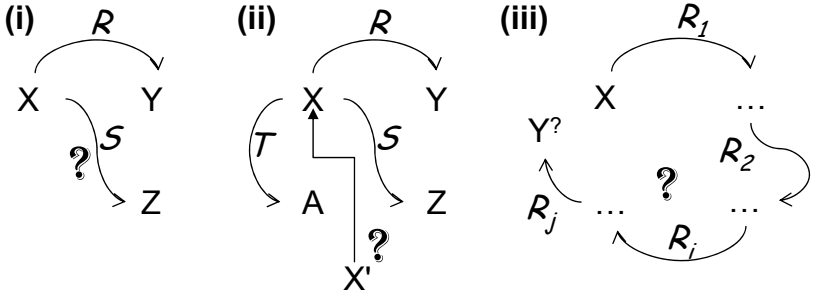
- Usual model checking
- Model checking against *real* instances in the ABox/Database
  - For each DL-concept in the OWL-formalised ontology (representing a universal), there has to be at least one ABox instance (as representation of the entity in reality)
  - To spot "redundant" DL-concepts w.r.t. the data-needs
- Model violation
  - Reducing the amount of instances to only those that do not violate the TBox (or: the more inconsistencies, the better)
  - For instance, to find a few candidate molecules that satisfy a given set of properties, out of a large pool of possibly suitable molecules; e.g., for drug discovery in pharmainformatics, tyre production

Discovering information

- The idea is that the combination of bio-ontologies, instances, and automated reasoning services somehow can find either the missing relations, or the types, or both
- How can one find what is, or may, not be in the ontology but ought to be there?
- At the TBox-level
  - computing derived relations (object properties)
  - find out where relations that are known by the developer have not yet been added to the ontology (finding 'known gaps')
  - add 'ontological' notions with top type 'whole' in a partonomy; e.g., 17 types of macrophage in the FMA each must be part of something
  - flag classes that have no relation (no or no is_a) to anything else in the ontology

Discovering information

- For the TBox through querying the data (ABox, RDBMS)
  - i. "for each x:X, y:Y, r:R, XRY, does there exist a z:Z, s:S, such that there exist ≥ 1 x and xsz?"
  - ii. "for each x:X, y:Y, r:R, XRY, does there exist an xsz and an xta where z:Z, s:S, a:A, t:T hold?"
  - iii. Find-me-anything-you-have: "for each x:X, return any $r_1, \ldots r_n$, their type of role and the concepts $Y_1, \ldots Y_n$ they are related to"



Summary