COMP718: Ontologies and Knowledge Bases

Lecture 7: Bottom-up Ontology Development

Maria Keet
email: keet@ukzn.ac.za
home: http://www.meteck.org

School of Mathematics, Statistics, and Computer Science
University of KwaZulu-Natal, South Africa

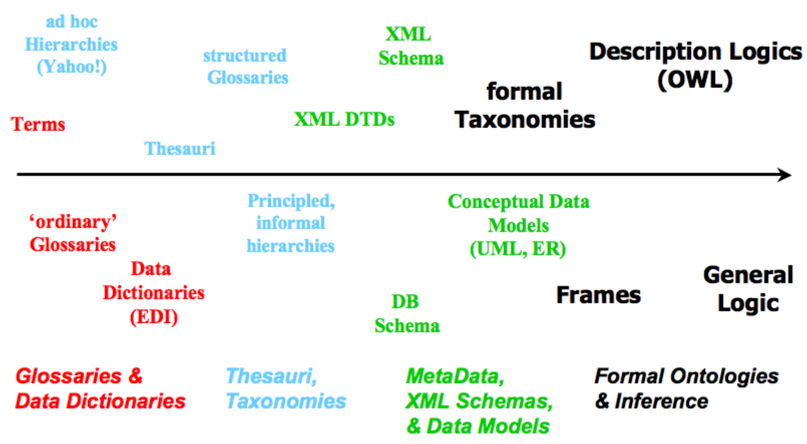
20 March 2012

Outline

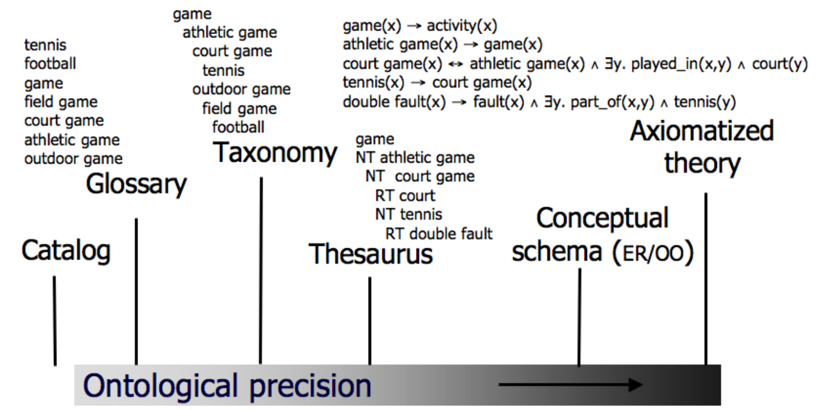
Bottom-up

- From *some* seemingly suitable legacy representation to an OWL ontology
  - Database reverse engineering
  - Conceptual model (ER, UML)
  - Frame-based system
  - OBO format
  - Thesauri
  - Formalizing biological models
  - Excel sheets
  - Text mining, machine learning, clustering
  - etc...

A few languages



Levels of ontological precision



precision: the ability to catch all and only the intended meaning
(for a logical theory, to be satisfied by intended models)

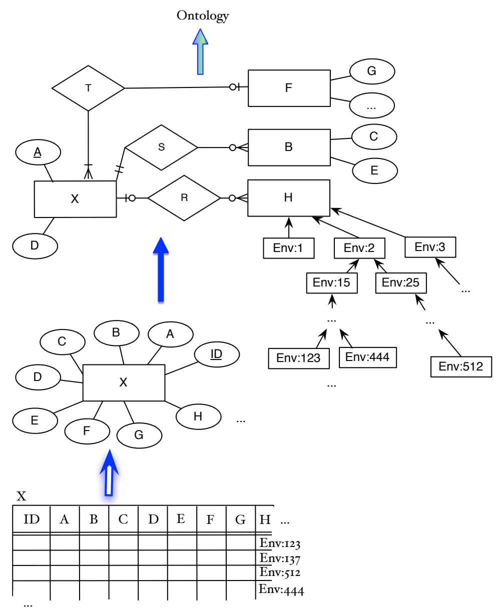
(from Gangemi, 2004)

Examples: OBO

- OBO in OWL 2 DL
  - OBO is a Directed Acyclic Graph (with is_a, part_of, etc. relationships)
  - with some extras (a.o., date, saved by, remark)
  - and 'work-arounds' (not-necessary and inverse-necessary) and non-mappable things (antisymmetry)
  - There are several OBO-in-OWL mappings, some more comprehensive than others
  - Most OBO ontology now also have an OWL version (consult OBO Foundry, BioPortal)

General considerations for RDBMSs

- Set aside of data duplication, violations of integrity constraints, hacks, outdated imports from other databases, outdated conceptual data models
- Some data in the DB—mathematically instances—actually assumed to be concepts/universals/classes
- 'impedance mismatch' DB values and ABox objects
- ⇒ values-but-actually-concepts-that-should-become-OWL-classes and values-that-should-become-OWL-instances



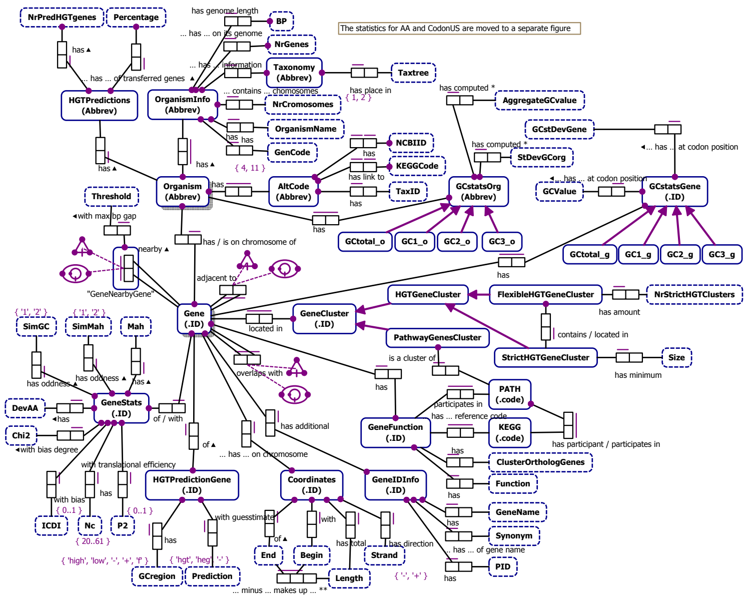
General considerations for RDBMSs

- Reuse/reverse engineer the physical DB schema
- Reuse conceptual data model (in ER, EER, UML, ORM, ...)
- But,
  - Assumes there was a fully normalised conceptual data model,
  - Denormalization steps to flatten the database structure, which, if simply reverse engineered, ends up in the ontology as a class with umpteen attributes
  - Minimal (if at all) automated reasoning with it
- Redo the normalization steps to try to get some structure back into the conceptual view of the data?
- Add a section of another ontology to brighten up the 'ontology' into an ontology?
- Establish some mechanism to keep a 'link' between the terms in the ontology and the source in the database?

Manual Extraction

- Most database are not neat as assumed in the 'Automatic Extraction of Ontologies' (e.g., denormalised)
- Then what?
  - Reverse engineer the database to a conceptual data model
  - Choose an ontology language for your purpose
- Example: the HGT-DB about horizontal gene transfer (the same holds for the database behind ADOLENA)

Section of the HGT conceptual data model (in ORM 2)



Manual mapping to *DL-Lite*$_{\mathcal{A}}$

- Basic statistics:
  - 38 classes
  - 34 object properties of which 17 functional
  - 55 data properties of which 47 functional
  - 102 subclass axioms
- Subsequently used for Ontology-Based Data Access

Automatic Extraction of Ontologies

- Examples
  - Lina Lubyte & Sergio Tessaris's presentation of the DEXA'09 paper
  - Reverse engineering from DB to ORM model with, e.g., VisioModeler v3.1 or NORMA

- See slides `SKOS.pdf`

Overview

- Thesauri galore in medicine, education, agriculture, ...
- Core notions of **BT** broader term, **NT** narrower term, and **RT** related term (and auxiliary ones UF/USE)
- E.g. the Educational Resources Information Center thesaurus:

```
reading ability
  BT ability
  RT reading
  RT perception
```

- E.g. AGROVOC of the FAO:

```
milk
  NT cow milk
  NT milk fat
```

- *How to go from this to an ontology?*

Problems

- Lexicalisation of a conceptualisation
- Low ontological precision
- BT/NT is not the same as *is_a*, RT can be any type of relation: overloaded with (ambiguous) subject domain semantics
- Those relationships are used inconsistently
- Lacks basic categories alike those in DOLCE and BFO (ED, PD, SDC, etc.)

Simple Knowledge Organisation System(s): SKOS

- W3C standard intended for converting Thesauri, Classification Schemes, Taxonomies, Subject Headings etc into one interoperable syntax
  - Concept-based search instead of text-based search
  - Reuse each others concept definitions
  - Search across (institution) boundaries
  - Standard software
- Limitations:
  - 'unusual' concept schemes do not fit into SKOS (original structure too complex)
  - `skos:Concept` without clear properties (like in OWL) and still much subject domain semantics in the natural language text
  - 'semantic relations' have little semantics (`skos:narrower` does not guarantee it is *is_a* or *part_of*)

A rules-as-you-go approach

- A possible re-engineering procedure:
  - Define the ontology structure (top-level hierarchy/backbone)
  - Fill in values from one or more legacy Knowledge Organisation System to the extent possible (such as: which object properties?)
  - Edit manually using an ontology editor:
    - make existing information more precise
    - add new information
    - automation of discovered patterns (rules-as-you-go)

see (Soergel et al, 2004)

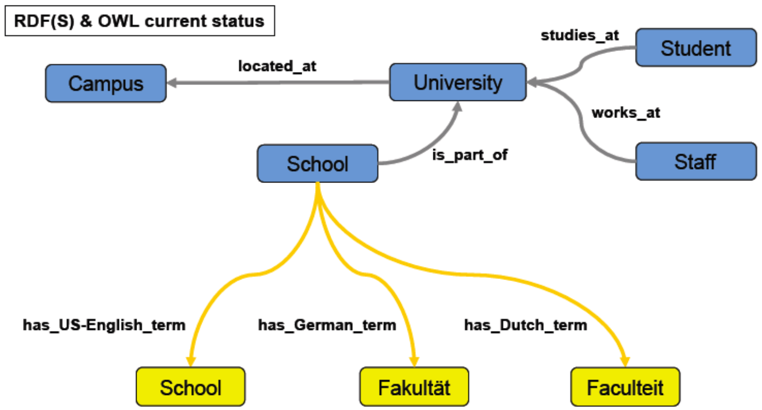
A rules-as-you-go approach

- A possible re-engineering procedure:
  - Define the ontology structure (top-level hierarchy/backbone)
  - Fill in values from one or more legacy Knowledge Organisation System to the extent possible (such as: which object properties?)
  - Edit manually using an ontology editor:
    - make existing information more precise
    - add new information
    - automation of discovered patterns (rules-as-you-go); e.g.
      - observation: *cow* NT *cow milk* should become *cow* <*hasComponent*> *cow milk*
      – pattern: *animal* <*hasComponent*> *milk* (or, more generally *animal* <*hasComponent*> *body part*)
      — derive automatically: *goat* NT *goat milk* should become *goat* <*hasComponent*> *goat milk*
      other pattern examples, e.g., *plant* <*growsIn*> *soil type* and *geographical entity* <*spatiallyIncludedIn*> *geographical entity*

Natural language and ontologies

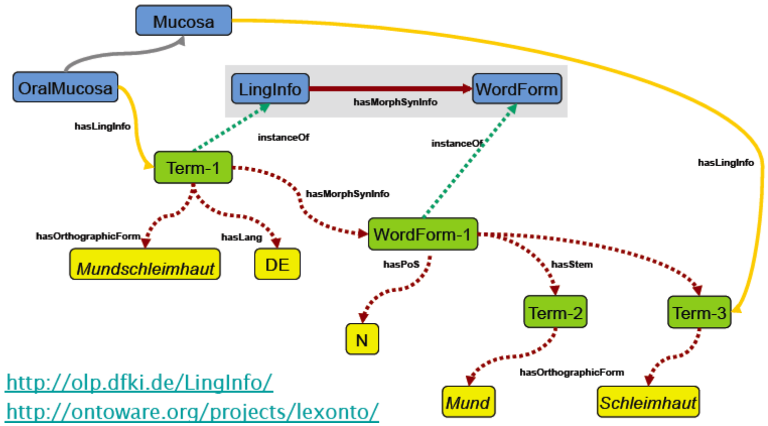
- Using ontologies to improve NLP
  - To enhance precision and recall of queries
  - To enhance dialogue systems
  - To sort literature results
  - To navigate literature (linked data)
- Using NLP to develop ontologies (TBox)
  - Searching for candidate terms and relations: Ontology learning (today; ref Alexopoulou et al, 2008)
- Using NLP to populate ontologies (ABox)
  - Document retrieval enhanced by lexicalised ontologies
  - Biomedical text mining (today; ref Witte et al, 2007)
- Natural language generation from a formal language

Ontologies in practice: Semantic Tagging—Classes, Terms



http://www.deri.ie/fileadmin/documents/teaching/tutorials/DERI-Tutorial-NLP.final.pdf

Ontologies in practice: Semantic Tagging—Lexicalized Ontologies



http://olp.dfki.de/LingInfo/
http://ontoware.org/projects/lexonto/

http://www.deri.ie/fileadmin/documents/teaching/tutorials/DERI-Tutorial-NLP.final.pdf

Examples (out of many)

- Generic tools: see `http://www.deri.ie/fileadmin/documents/teaching/tutorials/DERI-Tutorial-NLP.final.pdf` for a long list
- GoPubMed (Dietze et al, 2009)
  - Layer over PubMed, which indexes ± 19mln articles in the bio(medical) domain; pre-processing of the abstracts (advanced semantic tagging)
  - Results of the PubMed query are sorted according to terms in the ontology
- Question answer system AliQAn for agriculture (Vila and Ferrández, 2009)
  - Question assignment task too difficult for specialised domains
  - Add ontology to an open domain QA system, using AGROVOC and WordNet
- Attempto Controlled English (ACE), rabbit, etc.; grammar engine, template-based approach

Background

- Ontology development is time consuming
- Bottom-up ontology development strategies, of which one is to use NLP
- Where, if anywhere, can NLP make life easier for ontology development, and how?
- Current results are mostly discouraging, and depend on the approach, technique, and ontological commitment
  - We take a closer look at ontology learning limited to finding terms for a domain ontology

Bottom-up ontology development with NLP

- Usual parameters, such as purpose (in casu, document retrieval), formal language (an OWL species)
- A standard kind of ontology (not a comprehensive lexicalised ontology)
- Additional considerations for "text-mining ontologies"
  - Level of granularity of the terms to include (hypo/hypernyms)
  - How to deal with synonyms ('LDL I' and 'large LDL')
  - Handle term variations (e.g., 'LDL-I' and 'LDL I', 'Tangiers' disease' and 'Tangier's Disease')
  - Disambiguation; e.g. w.r.t. abbreviations

Method to test automated term recognition

- Compare the terms of a manually constructed ontology with the terms obtained from text mining a suitable corpus
- Build an ontology manually
  - Lipoprotein metabolism (LMO), 223 classes with 623 synonyms
- Create a corpus
  - 3066 review article abstract from PubMed, obtained with a 'lipoprotein metabolism' search
- Automatic Term Recognition (ATR) tools
  - Text2Onto: relative term frequency, TFIDF, entropy, hypernym structure of WordNet, Hearst patterns
  - Termine: statistics of candidate term, such as total frequency of occurrence, frequency of the term as part of other longer candidate terms, length of term
  - OntoLearn: linguistic processor and syntactic parser, Domain relevance and domain consensus
  - RelFreq: relative frequency of a term in a corpus
  - TFIDF: RelFreq + doc. frequency derived from all phrases in PubMed

Results

- OntoLearn excluded form analysis because it regenerated few terms
- Text2Onto only included in analysis for up to 300 abstracts (could not process all 3066)
- Precision for LMO 17-35% for top 50 terms, and 4-8% for top 1000 terms
- Precision for LMO + expert analysis of the automatically generated terms: up to 75% for top 50 terms, and up to 29% for top 1000 terms
- Termine good for the longer terms, RelFreq and TFIDF for the shorter terms

Results (cont'd)

Table 3: Coverage of LMO terminology in selected document sets. The table sets the upper limit of terms that can be found with text-mining: Even a large text base with 50,000 documents contains only 71% of LMO terms. TFIDF can predict up to 38% of LMO terms.

	LMO terminology predicted by TFIDF		*LMO terminology literally contained*
	1000	**all**	
300 review abstracts for "lipoprotein metabolism"	8.75%	15.35%	20.98%
3,066 abstracts for "lipoprotein metabolism"	14.99%	38.25%	53.00%
50,000 abstracts containing "lipoprotein"			71.22%

from Alexopoulou et al, 2008

What went wrong with some of the terms?

- LMO terms that were not in the 50k abstracts grouped into:
  - Rarely occurring terms: occur rarely even in the whole of PubMed
  - Rarely occurring variants of terms: e.g., 'free chol' (0, instead of 2622 for 'free cholesterol')
  - Very long terms; e.g, 'predominance of large low-density lipoprotein particles', which can be decomposed into smaller terms
  - Combinations of terms/variants; e.g., 'increased total chol' (0, instead of 116 for 'increased total cholesterol'),
  - Terms that should normally be easily found; e.g., 'diabetes type I' (126) and 'acetyl-coa c-acyltransferase', probably due to limited corpus
- Predicted terms, not in LMO: wrongly predicted ($\pm$25% of the TFIDF top50) or can be added to LMO ($\pm$40% of the TFIDF top50)

Typical NLP tasks

- Named Entity recognition/semantic tagging; e.g., "... the organisms were incubated at 37°C")
- Entity normalization; e.g., different strings refer to the same thing (full and abbreviated name, or single letter amino acid, three-letter aminoacid and full name: W, Trp, Tryptophan)
- Coreference resolution; in addition to synonyms (lactase and β-galactosidase), there as pronominal references (it, this)
- Grounding; the text string w.r.t. external source, like UniProt, that has the representation of the entity in reality
- Relation detection; *most of the important information in contained within the relations between entities*, NLP can be enhanced by considering semantically possible relations

Requirements for NLP ontologies

- Domain ontology (at least a taxonomy)
- Text model, concerns with classes such as *sentence*, *text position* and locations like *abstract*, *intorduction*
- Biological entities, i.e., contents for the ABox, often already available in biological databases on the Internet
- Lexical information for recognizing named entities; full names of entities, their synonyms, common variants and misspellings, and knowledge about naming, like *endo-* and *-ase*
- Database links to connect the lexical term to the entity represent in a particular database (the grounding step)
- Entity relations; represented in the domain ontology

MutationMiner use case

- See Witte et al. book chapter for details
- Ontology in OWL, in Protégé; with class name, textual definition and example instances
- Species info from the NCBI taxonomy; note the management of central *scientific name* and its synonyms, common variants and misspellings
- Uniprot and use of its back-links to the NCBI taxonomy

Discussion

- Significant upfront investments due to novelty and complexity of SWT
- Benefits:
  - Standardizes data exchange, consolidate disparate resources
  - Detecting inconsistencies (caused by, e.g. a pronoun with an incompatible relation to another textual entity)
- To do: Ontological NLP, enhancing standard NLP tools to take more of SWT into account

Summary

1. RDBMSs and other 'legacy KR'
   - Example: manual and automated extractions
2. Thesauri
   - SKOS
   - Thesauri
3. Natural language
   - Introduction
   - Ontology learning
   - Ontology population