Semantic Data

Chapter 10 : Rules and advanced topics

Jean-Louis Binot

Sources

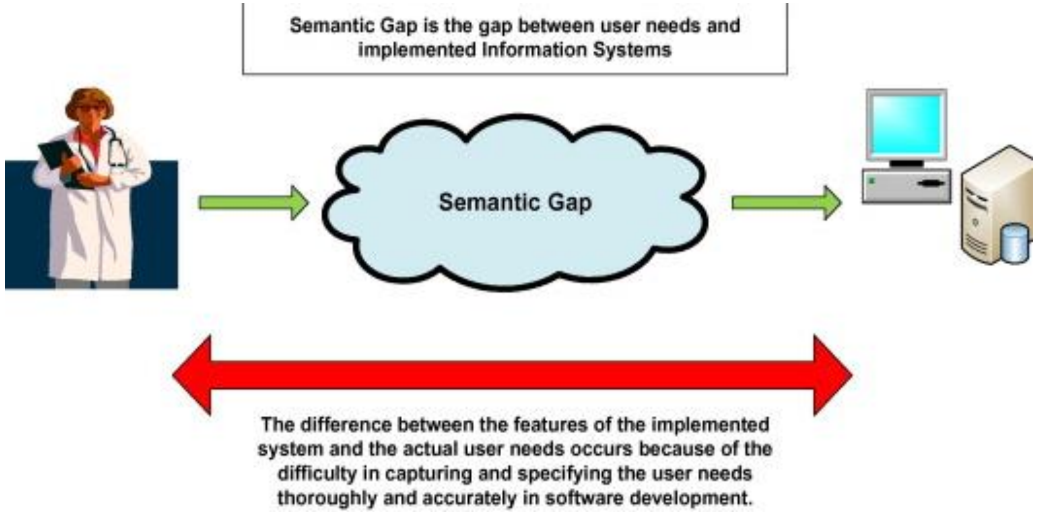
- There are no additional required references for this chapter.
- Sources and useful additional readings :
  - References for SWRL include the SWRL rule language proposal submission to W3C *(Horrocks et al. 2004)* and the Protege wiki SWRL language FAQ.
  - The *Advanced topics* section is inspired from various chapters of the *Handbook of Ontologies (Staab and Studer 2009)*, and other survey papers listed in the references.
- University courses having partially inspired ideas and examples for this chapter :
  - *Introduction to Semantic Web Rule Language, B. Espinasse, Aix-Marseilles University.*
  - *OWL 2 and SWRL Tutorial, M. Kuba, Mazaryk University.*

Agenda

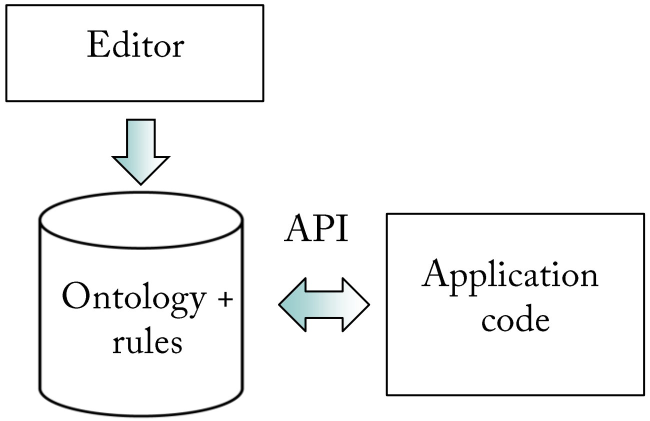
1	Extending an ontology with rules
2	Advanced topics

Capturing business rules

- Business rules are an intrinsic part of any enterprise.
  - *"A platinum disc is an album having sold 300000 copies or a single having sold 600000 copies."*
  - *"A General Manager in our organisation has a signature power of at most 500000 euros."*
- Business rules are often hardcoded into software code.
  - Many difficulties in making sure rules are correctly captured and in maintaining the code.
- A better approach is to use a rule engine :
  - Rules are expressed and maintained declaratively, reducing the semantic gap.
  - Rules can be triggered at execution time by a rule engine being an integral part of the application.



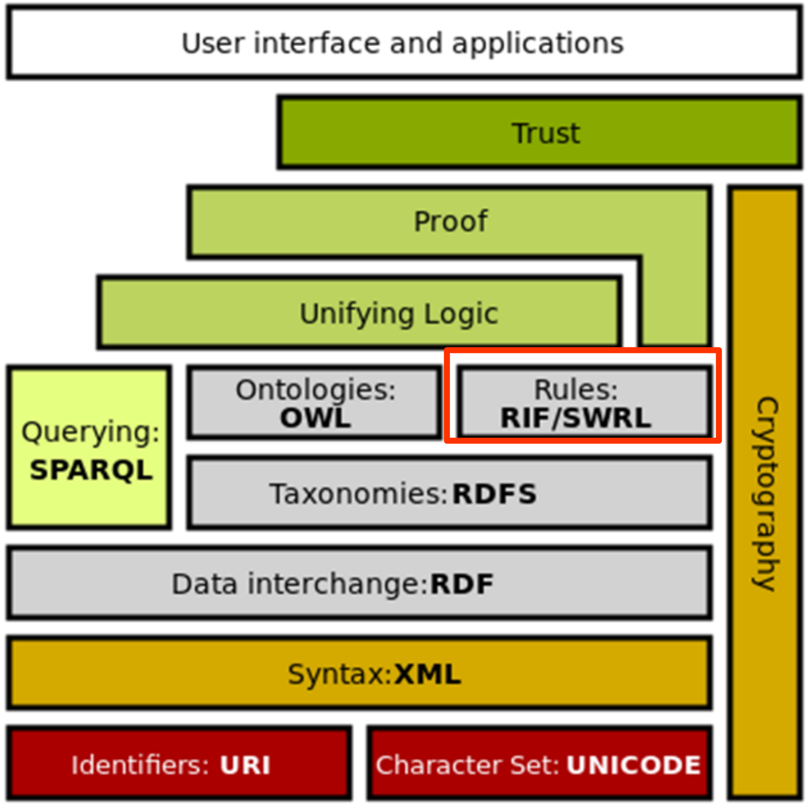
(Gurupur et al. 2014)



SWRL

- SWRL : Semantic Web Rule Language.
  - Intended to be the rule language of the Semantic Web.
  - Has remained at the status of a W3C standard *proposal*; competitors exist.
  - All rules are expressed in terms of OWL concepts (classes, properties, individuals).

- RIF : Rule Interchange Format.
  - Standard for exchanging rules among rule systems and dialects.



The semantic web stack of standards

Why a rule-based language ?

- OLW reasoning is limited to classification and related reasoning services (cf. chapter 4).
  - A single reasoning algorithm is sufficient (cf. chapter 8).
- Description logics and the OWL language cannot express all relations :
  - It cannot express for example the relation *child of married parents*.

  OWL has no variables : there is no way in OWL to express relationships between individuals with which an individual has relations.
- Rule based reasoning uses general rule-based inferences, as in expert systems :
  - Rules are IF … THEN statements expressed in a rule language;

  A very simple rule : IF x is a Man, THEN x is a Person (this one is also expressible in OWL).
  - Reasoning may use forward chaining or backward chaining algorithms (cf. later).

SWRL

- Extending OWL with a rule language is useful for several reasons :
  - Augmenting expressivity.
  - Reusing existing set of rules (e.g., from knowledge bases or expert systems).
  - A rule-based syntax may be more intuitive to read and write knowledge.
- SWRL *(Horrocks et al. 2004)* : W3C standard proposal combining ontologies and rules.
  - Rules are based on a Rule Markup Language, RuleML, including Horn clause-based sublanguages.

    Horn clause : disjunctions of literals with at most one positive (Datalog : function-free Horn language).

    $\neg P(X, Y) \vee \neg Q(Y, Z) \vee R(X, Z)$ is equivalent to $P(X, Y) \wedge Q(Y, Z) \rightarrow R(X, Z)$
  - SWRL is a union of OWL DL and Horn logic rules; this union will have a decidability cost.
- SWRL is supported by the main OWL tools :
  - The Protégé editor, several DL reasoners (Pellet, Hermit), the OWL API.

SWRL syntax

- Basic idea: extend OWL ontology axioms with a new axiom : *axiom ::= rule*.
- Rules can be serialized in various syntaxes, including XML and RDF. We will use the human-readable syntax, also supported by the SWRL Protégé plugin :

    antecedent -> consequent

    - Both antecedent and consequent are conjunctions of atoms : a1 ∧ … ∧ an.
    - Atoms may refer to individuals, data literals, individual variables or data variables.
    - Variables are prefixed by a ? (as in SPARQL).
    - Variables are treated as universally quantified, with their scope limited to a given rule.
- A classical example :

    hasParent(?x1, ?x2) ∧ hasBrother(?x2, ?x3) -> hasUncle(?x1, ?x3)

(not expressible in OWL 1 but well in OWL 2 through role composition : uncleOf ≡ brotherOf o parentOf)

Forward and backward chaining

- Forward
  - Input: rules + axioms (data).
  - Output: extended data.
  - Starts with available facts.
  - Uses rules to derive new facts (which can be stored).
  - Stops when there is nothing else to be derived.
- Backward
  - Input: rules + axioms (data) + goal (statement).
  - Output: goal statement is true, or it is is false.
  - Goes backwards from hypothesis to set of axioms.
  - If it can find the path to the original axioms, then the hypothesis is true (otherwise false)

Axioms

Axioms

hasParent(Fred, John)

hasBrother(John, Luc)

Forward

Backward

Rule :

hasParent(?x1, ?x2) ∧
hasBrother(?x2, ?x3)
-> hasUncle(?x1, ?x3)

New derived facts

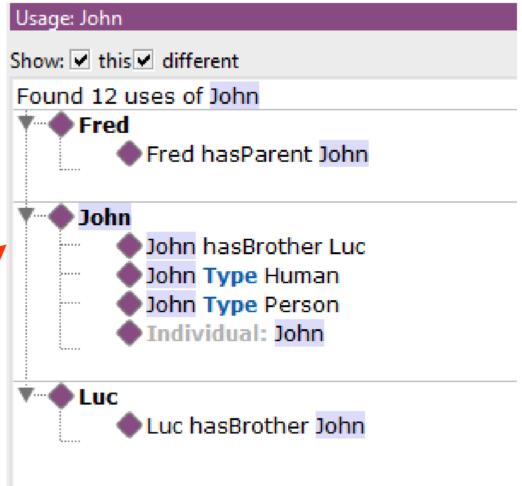
Goals

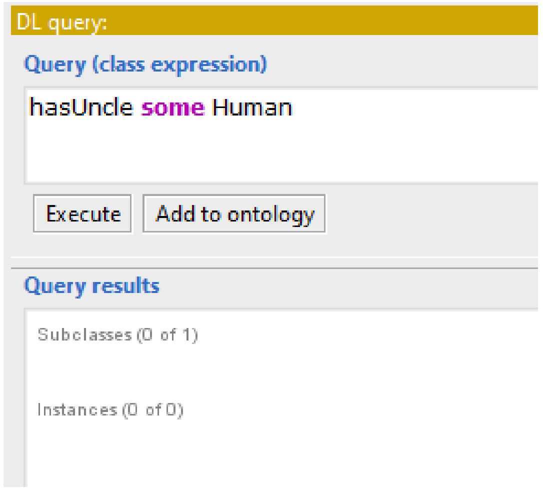
hasUncle(Fred, Luc)

SWRL in Protégé

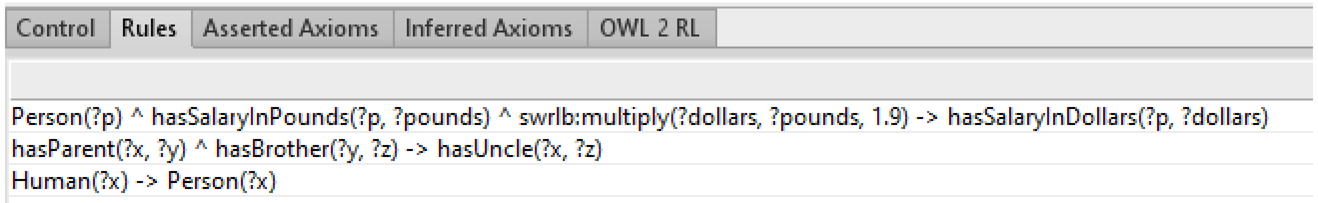
- Illustration of example of previous slide

1. Create instances, and hasBrother and hasParent relationships.
   - No Uncle instance.

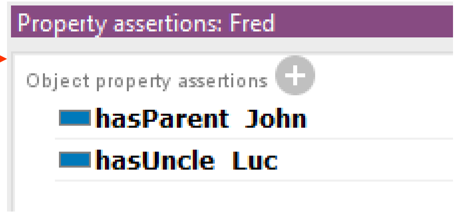


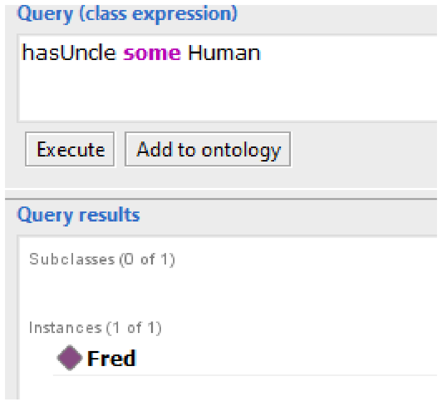


2. Add rules :
   - hasParent(?x, ?y) ^ hasBrother(?y, ?z) -> hasUncle(?x, ?z).



3. Execute rule reasoner (Drools):
   - New facts get added to the ontology.
   - The query for hasUncle now has a solution.





- Another simple rule :
  - Human(?x) -> Person(?x)

  This rule is expressible in OWL (subsumption).

Types of atoms supported in SWRL

- Class expression : named class or class expression, with a single individual argument : Person(?p), Man(Fred), (hasChild >= 1)(?x)
- Individual property : hasBrother(?x, ?y), hasSibling(Fred, ?y)
- Data valued property : hasHeight(Fred, ?h), hasAge(?x, 232), hasName(?x, "Fred")
- Different individuals, same individuals : differentFrom(?x, ?y), sameAs(Fred, Freddy)
- Data range : xsd:int(?x), [3, 4, 5](?x)
- Built-ins :
  - Core built-ins from SWRL submission : swrlb:greaterThan(?age, 17)
  - Custom built-ins : defined using Java code.

Examples

- If a person has a sibling who is a man, he has a brother.
  Person(?p) ^ hasSibling(?p, ?s) ^ Man(?s) -> hasBrother(?p, ?s)

- If a person Fred has two parents such as one is the spouse of the other, he is a child of married parents.
  Person(Fred), hasParent(Fred, ?y), hasParent(Fred, ?z), hasSpouse(?y, ?z) -> ChildOfMarriedParents(Fred)

- If a person has a car he is a driver.
  Person(?p) ^ hasCar(?p, true) -> Driver(?p)

- Person(?p) ^ hasAge(?p, ?age) ^ swrlb:greaterThan(?age, 17) -> Adult(?p)
  If a person has an age greater than 17, he is an adult.

- Person(?p) ^ hasNumber(?p, ?number) ^ swrlb:startsWith(?number, "+")
  -> hasInternationalNumber(?p, true)
  If a person has a number starting with +, he has an international number.

Semantics of SWRL

- The semantics of atoms is captured by a model-theoretic interpretation $\mathcal{I}$ made of :
  - A set of resources R constituting the interpretation domain;
  - An interpretation function $.^{\mathcal{I}}$ mapping language constructs to objects and tuples of the interpretation domain.

- The semantics of rules of the form antecedent -> consequent, where both antecedent and consequent are conjunctions of atoms, is defined as follows :

  - A binding B($\mathcal{I}$) extends an interpretation $\mathcal{I}$ so that every individual variable and data variable is bound to a resource of the interpretation. Then :

  - A binding B($\mathcal{I}$) satisfies an antecedent A if A is empty, or B($\mathcal{I}$) satisfies every atom in A.

    A rule with an empty antecedent is asserting facts.

  - A binding B($\mathcal{I}$) satisfies a consequent C if C is empty, or B($\mathcal{I}$) satisfies every atom in C.

  - A rule is satisfied by an interpretation $\mathcal{I}$ iff for every binding B such that B($\mathcal{I}$) satisfies the antecedent, B($\mathcal{I}$) also satisfies the consequent.

Semantics of SWRL ./.

As OWL, SWRL semantics respect the following characteristics :

- The Open World Assumption.
- The No Unique Name Assumption.
- Monotonicity.

Concept expressions in rules and OWA

- In principle class atoms can be concept expressions :

(hasChild >= 1)(?x) -> Parent(?x)

Those who have >= 1 child are parents.

- Such a rule has two drawbacks :

1. As variables are universally quantified, the rule will classify as parents individuals with no children.

2. As SWRL supports the open world assumption :
   - This rule may also match individuals that have no explicit values for the hasChild property in the ontology but for which the existence of such values can be deduced from the OWL axioms.

   Expressing *has exactly one value for hasChild among the explicit individuals of the ontology* is impossible.

- OWL concept expressions are not supported in current version of Protégé.

No unique name assumption, sameAs, differentFrom

Consider the rule :

Publication(?x) ^ hasAuthor(?x, ?y) ^ hasAuthor(?x, ?z) -> cooperatedWith(?y, ?z)

- Due to rule pattern matching, ?y and ?z can be matched to the same individual.
- Due to the No Unique Name Assumption, we cannot assume that two individuals with different names are distinct.

- Unicity of names (or not) has to be stated explicitly, using sameAs or differentFrom.

Publication(?x) ^ hasAuthor(?x, ?y) ^ hasAuthor(?x, ?z) ^ differentFrom(?y, ?z) -> cooperatedWith(?y, ?z)

:OJerusalem a :publication .
:OJerusalem :has Author :DominiqueLapierre .
:OJerusalem :has Author :LarryCollins .
:DominiqueLapierre owl:differentFrom :LarryCollins .
=>
:DominiqueLapierre cooperatedWith :LarryCollins .

Built-ins and value computation

- Built-ins can assign (bind) values to arguments.

Person(?p) ^ hasSalaryInPounds(?p, ?pounds) ^ swrlb:multiply(?dollars, ?pounds, 1.25)
-> hasSalaryInDollars(?p, ?dollars)

  - The execution of the rule binds the first variable of swrlb:multiply to the resulting value.
  - This binding can then be reused in other atoms. Binding precedence is from left to right.
  - The place of unbound variables in the argument list is fixed by the definition of the built-in.

- Another example :

Rectangle(?r) ^ hasWidthInMetres(?r, ?w) ^ hasHeightInMetres(?r, ?h) ^
swrlb:multiply(?areaInSquareMeters, ?w, ?h) ^ swrlb:greaterThan(?100, ?areaInSquareMeters)
-> hasAreaInSquareMetres(?r, ?areaInSquareMeters) ^ BigRectangle(?r)

Main limits of SWRL

- Disjunction of atoms is not supported :
  - Consider A(?x) or B(?y) -> C(?x) or C(?x) -> A(?x) or B(?x) : how should the rule react?
  - (A or B)(?x) -> C(?x) is supported as A or B is a concept expression, but not in Protégé.
  - A(?x) or B(?x) -> C(?x) can be expressed as two rules : A(?x) -> C(?x) and B(?x) -> C(?x).
- Concept expressions and data range atoms are not supported in Protégé.
- OWL Full is not supported.
- RDF and RDFS syntaxes are not supported.
- Full SWRL with OWL DL is not decidable!
  - E.g., transitive roles with number restrictions are undecidable and adding an SWRL rule can transform a "simple" role into a transitive role.
  - But a safe (decidable) restriction on SWRL exists.

DL-Safe SWRL rules

- To ensure decidability, DL-Safe SWRL rules are restricted to only bind variables to known individuals in an ontology.

  Consider the example :

  Vehicle(?v) ^ Motor(?m) ^ hasMotor(?v, ?m) -> MotorizedVehicle(?v)

  $Car \sqsubseteq Vehicle \sqcap \exists hasMotor.Motor$

  One could expect that an individual of class Car would be classified as a MotorizedVehicle.

  A DL safe implementation will not make this inference as it would require to bind variable ?m to an individual (the motor) which is not known.

- DL-Safe SWRL rules may produce incomplete inferences (not produce all the deductions entailed by the ontology), but all inferences deduced are sound.

- The limitation to support only DL-Safe rules is built-in the reasoner.

Agenda

1	Extending an ontology with rules
2	Advanced topics

1. Inferences in knowledge graphs

Reminder of the basics of knowledge graphs :

- A knowledge graph describes objects of interest and connections between them (*semantic network*).
- The graph can support (intelligent) search, question answering, natural language interaction…
- Google was the initiator. Many large technological compagnies now use them (Google, Microsoft, Facebook, eBay …)
- Many practical implementations impose constraints on the links in knowledge graphs by defining an ontology of reference *(ontological commitment).*
- Technologies used for knowledge graphs involve usually NoSQL databases, RDF triples or property-based graphs, graph-traversal search and graph query languages…
- Given the complexities of large-scale knowledge graphs, machine learning is used to make graph-based predictions.

Large scale knowledge graph challenges

- Large scale knowledge graphs are still very incomplete.
- Example : consider statistics about Freebase :
  - Large collaborative KB developed by Metaweb, acquired by Google in 2010. Google KG was in part powered by Freebase.
  - Freebase was shut down in 2016 (its API replaced by a Google KG API).
- Challenges
  - Entity resolution and type membership resolution
  - Link prediction
  - Extracting structured / unstructured knowledge from multiple sources …
  - Managing changes …
- Manual data management is not feasible at very large scale. Systematic learning inference approaches are needed.

Relation	Percentage unknown	
	All 3M	*Top 100K*
PROFESSION	68%	24%
PLACE OF BIRTH	71%	13%
NATIONALITY	75%	21%
EDUCATION	91%	63%
SPOUSES	92%	68%
PARENTS	94%	77%
CHILDREN	94%	80%
SIBLINGS	96%	83%
ETHNICITY	99%	86%

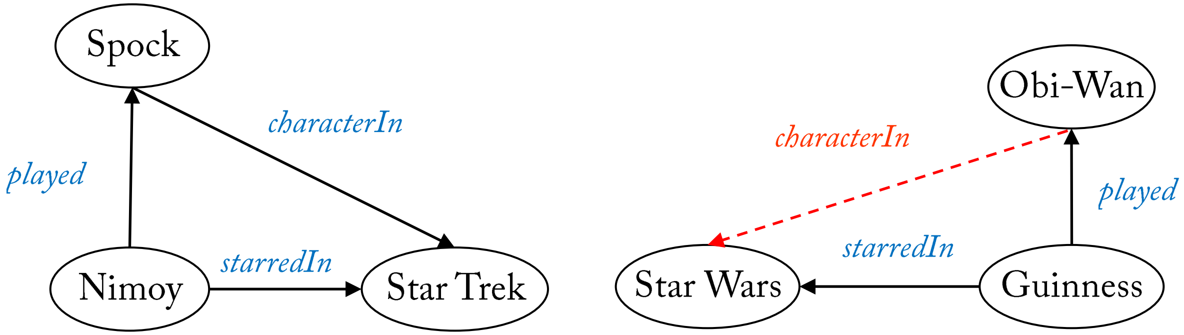
Table 1: Incompleteness of Freebase for some relations that apply to entities of type PERSON. *Left:* all 3M Freebase PERSON entities. *Right:* only the 100K most frequent PERSON entities.

(West et al. 2014)

Knowledge graph inferences and machine learning

- Knowledge graphs use OWA : missing information is unknown.
- This information can be learned :
  - Either by consulting external sources,
  - Or by learning from the graph itself.
- Knowledge graph learning can support data curation tasks :
  - Link or triple prediction (or knowledge graph completion); entity resolution.

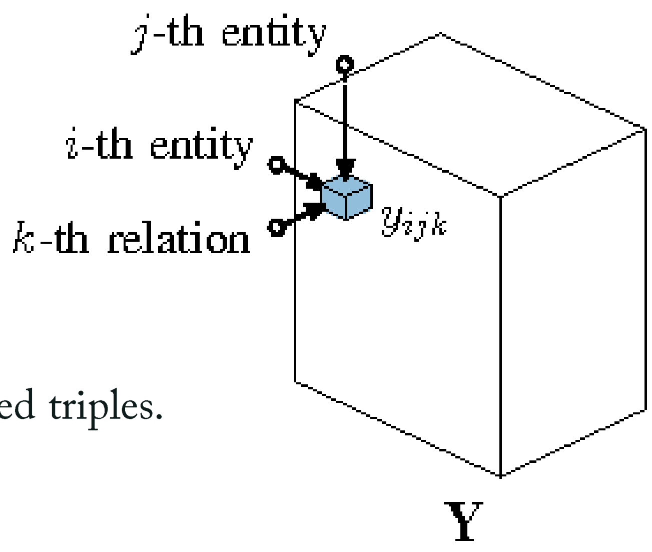
Example : the link from Obi-Wan Kenobi to Star Wars can be predicted based on structural similarities.



(Nickel et al. 2015)

Statistical relational learning for knowledge graphs

- A knowledge graph can be represented by an adjacency tensor Y :
  - Y_{ijk} = 1 if the triple $n_i l_j n_k$ exists: relation l_j exists between node n_i and node n_k.,
  - Y_{ijk} = 0 otherwise.



- We are interested in estimating the joint probability distribution P(Y) :
  - To predict $P(Y_{ijk})$ for unobserved triples, based on a subset $\mathcal{D} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E} \times \{0,1\}$ of observed triples.
- Several approaches :
  - If all Y_{ijk} are assumed conditionally independent given observed features : graph feature model techniques.
  - If all Y_{ijk} are assumed conditionally independent given latent features : knowledge graph embedding techniques.

  A dynamic area of research with many approaches *(cf. e.g., Nickel et al. 2015)*.
- More generally convergences are being built between knowledge representation and machine learning.
  - Knowledge can constraint and guide machine learning, can be used to validate results …

(Nickel et al. 2015)

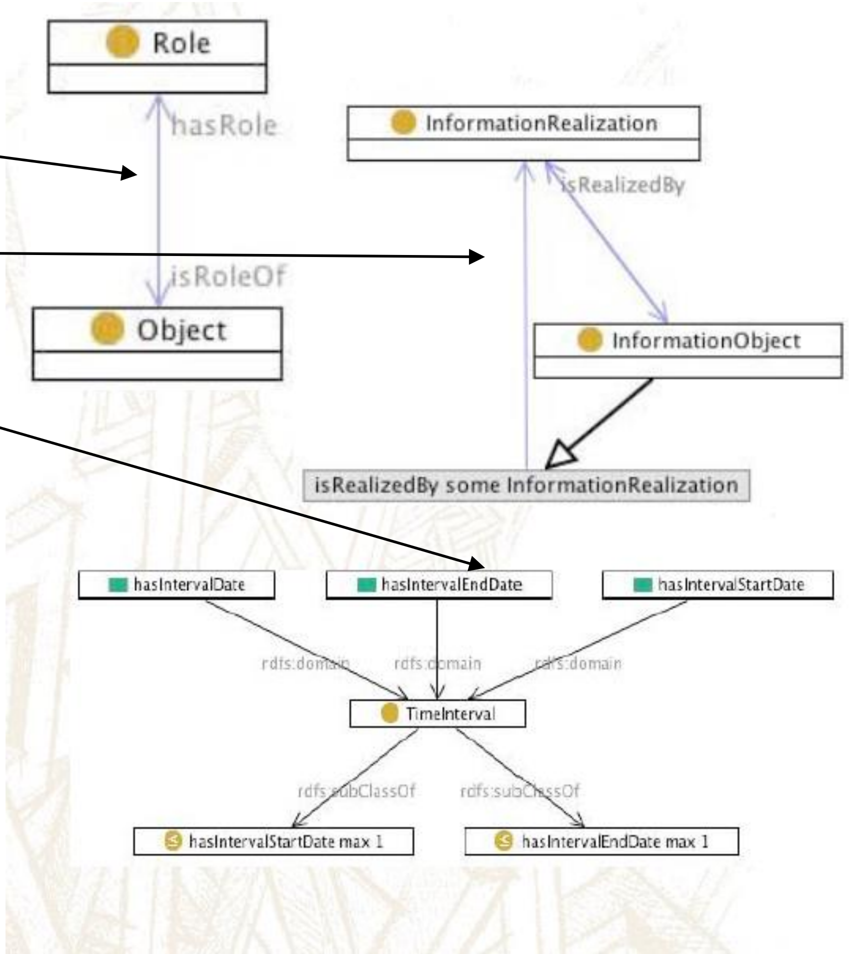
2. Ontology design patterns

- Challenges
  - Not every user (or even knowledge engineer) is at ease with logical structures.
  - Good practices must often be discovered by trial and error or from the literature.
  - Existing ontologies are often large and difficult to understand; their reuse may need costly adaptations.
  - We are missing basic building blocks, that could be selected and fitted together.
- Ontology design patterns : a modelling solution to solve a recurrent ontology design problem.
- Pattern-based design is the activity of searching, selecting, and composing patterns :
  - With a common framework to understand, collect, select and reuse patterns
    (for example, http://www.ontologydesignpatterns.org).

(based on Gangemi and Presutti 2009a and 2009b)

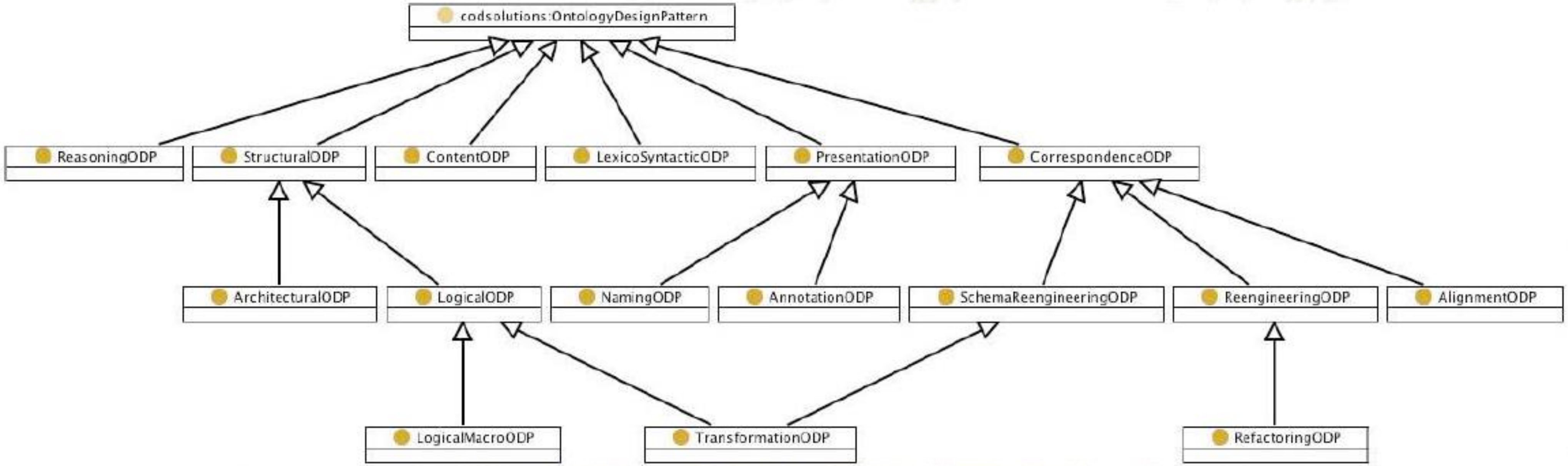
An example of content-based pattern

Arnold Schwarzenegger is Shylock in the play of "Merchant of Venice", that is given at the theater "Roma" during September and October 2009.



- A person plays a character.
- In a play of a drama.
- During a time period.

Types of Ontology Design Patterns



- Still an open area. Challenges :
  - Populating repositories of patterns;
  - Discovering or extracting them from existing ontologies;
  - Semi-automating their use to assist users in their application;
  - Defining a robust semantics and algebra for design patterns…

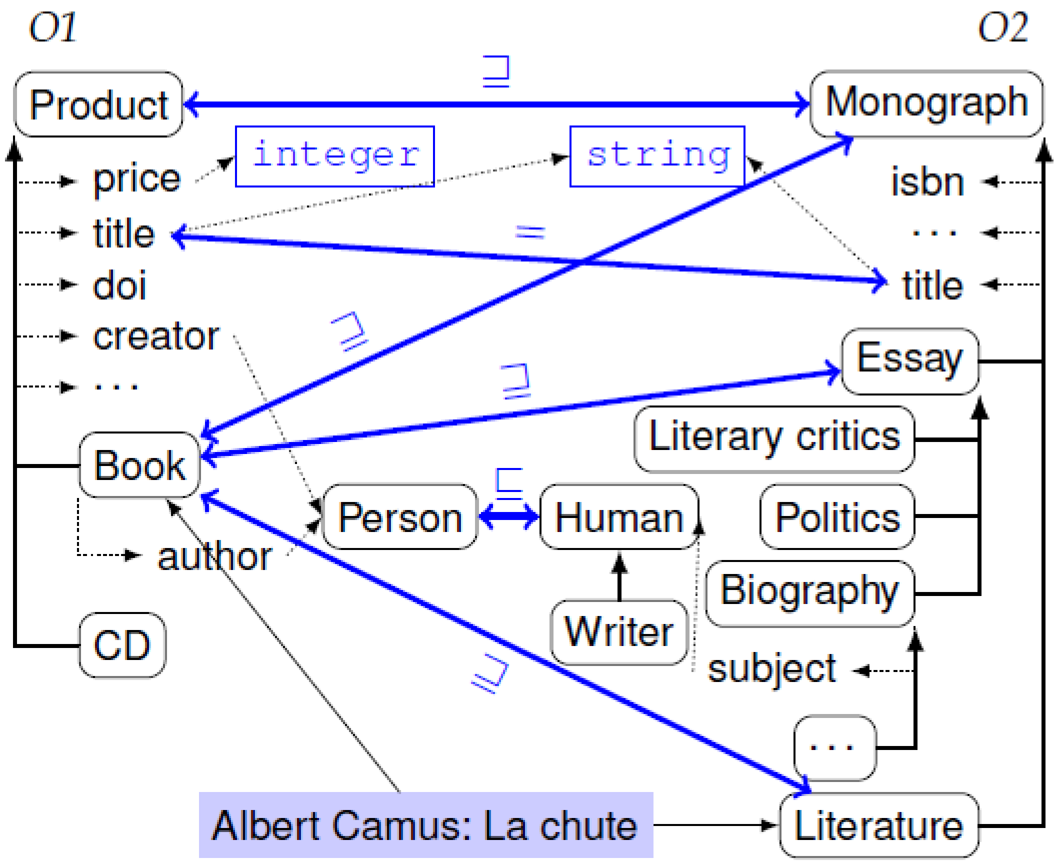
3. Ontology alignment

- Semantic heterogeneity : variations in meaning or ambiguities in interpretation.
  - Ontologies developed separately may not be able to inter-operate.
- Overcoming semantic heterogeneity is typically achieved in two steps :
  1. Matching entities to determine an alignment, i.e., a set of correspondences;
  2. Interpreting an alignment according to application needs (data translation, query answering…).
- Ontology matching is a solution to the semantic heterogeneity problem.
  - It finds correspondences between semantically related entities of ontologies.
- Current systems are able to deal with tens of thousands of entities efficiently.
  - Their primary target is OWL but other languages (RDFS, SKOS) are also covered.
  - Approaches may come both from AI and from databases.

(based on Shvaiko and Euzenat 2013)

The ontology alignment problem

- Example: matching two simple ontologies.
  - Classes are in rounded boxes, properties without boxes. There is one shared instance.
- Matching is expressed by axioms relating the classes and roles of both ontologies :
  - Axioms can be equivalences or inclusions.
- These axioms can be used to manage the two ontologies as a single integrated ontology :
  - Making use of the axioms at use time (e.g., in queries);
  - Or physically integrating the ontologies.



Some techniques (often combined)

- Identification of anchors (pairs of entities which can be mapped) :
  - Linguistic processing techniques (tokenization, morphology analysis …) to compare strings;
  - Word vector distance;
  - Use of external references such as WORDNET (cf. chapter 5) to detect similarities…
- Iterative structural matching : exploiting the hierarchy and reference ontologies.
- Similarity propagation through the ontology graphs.
- Semantic inference rules.
- Using clustering to partition large ontologies into blocks.
  - Mapping concepts within blocks and identifying similarities between blocks.
- Techniques can be semi-manual : user is asked for confirmation
- Many open challenges : selecting and tuning mappers, using background knowledge…

4. Ontology learning

Dynamic area of research touching to ontology mapping. Several directions:

- Ontology Learning from Text : automatic or semi-automatic generation of lightweight taxonomies by means of text mining and information extraction.
  - Based on computational linguistics (acquisition of lexical information from corpora).
- Concept Learning in description logics and OWL : aims at learning schema axioms, such as definitions of classes, from existing ontologies and instance data.
  - Most methods in this area are based on Inductive Logic Programming methods.
- Linked Data Mining : detecting meaningful patterns in RDF graphs.
  - Uses statistical relational learning methods to mine correlations in large data sets. Combines logical and probabilistic modeling to be able to deal with complex relations and handle noise.
- Ontology crowdsourcing : an alternative to automatic approaches as it combines the speed of computers with the accuracy of humans (e.g., Nell).

Summary

- Implementation of business rules, as in expert systems, can rely i.a. on the Semantic Web Rule Language SWRL, which is merging OWL DL with Horn logic rules. A restriction, DL-Safe rules, must be respected to keep the result decidable.
- Ontology engineering is an evolving area. Research topics include knowledge graph inferences, reusable patterns, ontology alignment, ontology learning…
- Convergences between machine learning and ontologies offer new perspectives for AI and knowledge representation.

References

- *[Gangemi and Presutti 2009a] : Gangemi A. and Presutti V.:* Ontology Design Patterns, *in (Staab, S., Studer R. eds.), Handbook on Ontologies 2nd edition, Springer, 2009.*
- *[Gangemi and Presutti 2009b]: Gangemi A. and Presutti V.*, Best practices in ontology Design, *The Seventh Summer School on Ontological Engineering and the Semantic Web (SSSW'09), Cercedilla, Spain 2009.*
- *[Horrocks et al. 2004] : Horrocks, I ., Patel-Schneider, P., Boley, H., Tabet, S., Grosof, B. and Dean, M.*, SWRL: A Semantic Web Rule Language Combining OWL and RuleML, *W3C Member Submission, 21 May 2004.*
- *[Nickel et al. 2015] : Nickel M., Murphy K., Tresp V. and Gabrilovich E.*, A Review of Relational Machine Learning for Knowledge Graphs, *in Proceedings of the IEEE, vol. 104, no. 1, pp. 11-33, Jan. 2016.*
- *[Shvaiko and Euzenat 2013]: Shvaiko P. and Euzenat J.*, Ontology matching: state of the art and future challenges, *IEEE Transactions on knowledge and data engineering, 25(1):158-176 , January 2013.*
- *[Staab and Studer 2009]: Staab, S., Studer R. (Eds)*, Handbook on Ontologies 2nd ed., *Springer, 2009.*
- *[West et al. 2014] : West R., Gabrilovich E., Murphy K., Sun S.,Gupta R. and Lin D.*, Knowledge base completion via search-based question answering, *Proceedings of the 23rd international conference on World wide web, WWW2014, April 2014.*

THANK YOU