Razonamiento en Ontologías: Por qué y Cómo.

Basado en material de Ian Horrocks

Dra. Pilar Bulacio
bulacio@cifasis-conicet.gov.ar

Bibliografía

- Comparison of Reasoners for large Ontologies in the OWL 2 EL Profile. Semantic Web – Interoperability, Usability, Applicability an IOS Press Journa. http://www.semantic-web-journal.net/content/comparison-reasoners-large-ontologies-owl-2-el-profile
- A Survey on Ontology Reasoners and Comparison. Int. Journal of Computer App. http://www.ijcaonline.org/archives/volume57/number17/9208-3748
- Benchmarking OWL Reasoners. Research Center for Information Technologies, Karlsruhe, Germany. http://ceur-ws.org/Vol-350/paper1.pdf
- DL Handbook, Cambridge University Press
  - http://books.cambridge.org/0521781760.htm
- FaCT system (open source)
  - http://www.cs.man.ac.uk/FaCT/
- Protégé
  - http://protege.stanford.edu/plugins/owl/
- W3C Web-Ontology (WebOnt) working group (OWL)
  - http://www.w3.org/2001/sw/WebOnt/

Repaso

Orígenes: Lógicas descriptivas (DL)

- Son una familia de formalismos basados en lógica de primer órden para la **Representación de Conocimiento**.
  - Describen al **dominio** en función de **conceptos** (classes), **roles** (relationships) e **individuos**.
  - Describen a la **semántica** que establece equivalencias entre *fórmulas lógicas de descripción y expresiones lógicas de predicados.*

•*http://es.wikipedia.org/wiki/Lógica_de_descripción*

DL Basics

- Concepts (**formulae**)
  - E.g., Person, Doctor, HappyParent, (Doctor ⊔ Lawyer)
- Roles (**modalities**)
  - E.g., hasChild, loves
- Individuals (**nominals**)
  - E.g., John, Mary, Italy
- Operators (para formar conceptos y roles):
  - Computables y si es posible, de baja complejidad

DL: Ej. de **Constructores** de conceptos y roles

- Restricciones numéricas de **cardinalidad** sobre roles, e.g., ≥3 hasChild, ≤1 hasMother
- **Nominales** (conceptos *singleton*), e.g., {Italy}
- Dominios concretos (tipos de datos), e.g., hasAge.(≥ 21)
- Roles **Inversos**, e.g., hasChild- (hasParent)
- Roles **Transitivos**, e.g., hasChild* (descendant)
- Composición de roles, e.g., hasParent o hasBrother (uncle)

The DL Family (1)

- Smallest propositionally closed DL is $\mathcal{ALC}$ (Attributive Concept Language with Complements)
  - Concepts constructed using booleans

    $\sqcap, \sqcup, \neg$

    plus restricted quantifiers

    $\exists, \forall$
  - Only atomic roles

E.g., Person all of whose children are either Doctors **or** have a child who is a Doctor:

$$\text{Person} \sqcap \forall \text{hasChild}.(\text{Doctor} \sqcup \exists \text{hasChild}.\text{Doctor})$$

$$\text{Person} \land [\text{hasChild}](\text{Doctor} \lor \cup \text{hasChild} \uplus \text{Doctor})$$

Base de Conocimiento (KB) en LD

F. Baader, W. Nutt

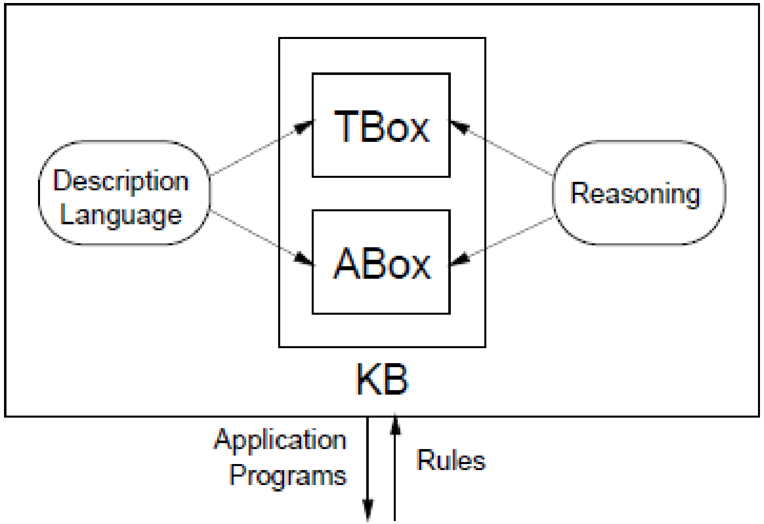


Fig. 2.1. Architecture of a knowledge representation system based on Description Logics.

•TBox (caja terminológica) contiene sentencias describiendo conceptos generales (i.e., relaciones entre conceptos).

• ABox (caja de aserciones) contiene sentencias "ground" indicando a donde pertenecen los individuos en la jerarquía (i.e., relaciones entre individuos y conceptos). Por ejemplo, las frases:

•(1) Cada empleado es una persona (pertenece a la TBox),

•(2) Bob es un empleado (pertenece a la ABox)

Razonamiento:
Por qué lo necesitamos?

Philosophical Reasons

- Applications such as the Semantic Web aim at "machine understanding"
- Understanding is closely related to reasoning
  - Recognising semantic similarity in spite of syntactic differences
  - Recognising implicit consequences given explicitly stated facts

Practical Reasons

- Given key role of ontologies in many applications, it is essential to provide tools and services to help users:
  - Design and maintain high quality ontologies, e.g.:
    - Meaningful — all named classes can have instances
    - Correct — captured intuitions of domain experts
    - Minimally redundant — no unintended synonyms
    - Richly axiomatised — (sufficiently) detailed descriptions
  - Answer queries over ontology classes and instances, e.g.:
    - Find more general/specific classes
    - Retrieve individuals/tuples matching a given query
  - Integrate and align multiple ontologies

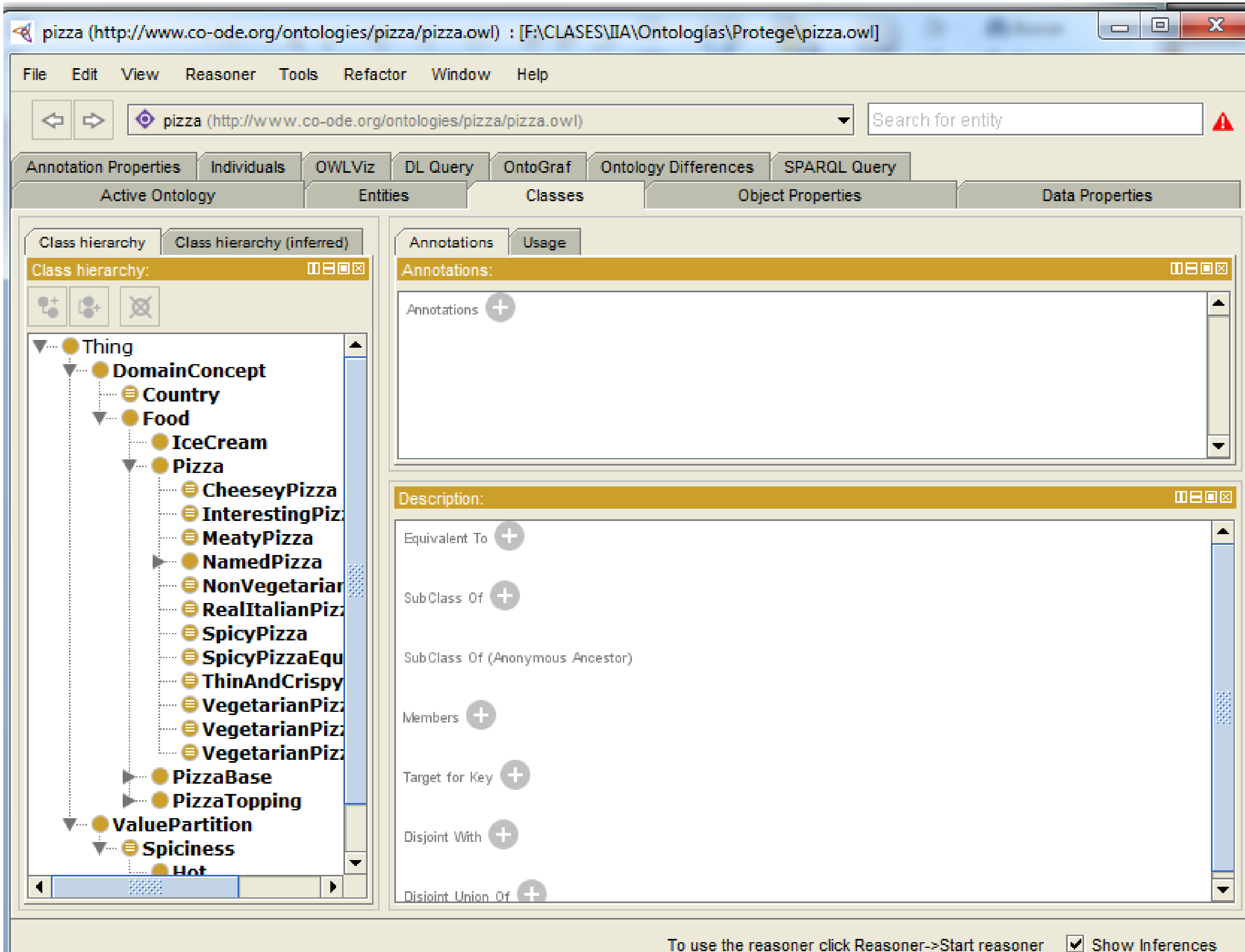
Why Decidable Reasoning?

- OWL constructors/axioms have been restricted so reasoning is decidable
- Consistent with Semantic Web's layered architecture
  - XML provides syntax transport layer
  - RDF(S) provides basic relational language and simple ontological primitives
  - OWL provides powerful but still decidable ontology language
  - …
- W3C requirement for "implementation experience"
  - "Practical" algorithms for sound and complete reasoning

Why Sound & Complete Reasoning?

- Important for ontology design
  - Ontologists need to have complete confidence in reasoner
  - Otherwise they will cease to trust results
  - Doubting unexpected results makes reasoner useless
- Important for ontology deployment
  - Many realistic web applications will be agent $\leftrightarrow$ agent
  - No human intervention to spot glitches in reasoning
- …

Protégé: Pizza



Ontology Reasoning: How do we do it?

Los razonadores DL brindan los siguientes servicios de inferencia

- **Validación de la consistencia** de una ontología: el razonador puede comprobar si hay hechos contradictorios
- **Validación del cumplimiento de los conceptos** de la ontología: el razonador determina si es posible que una clase tenga instancias. *En el caso de que un concepto no sea satisfecho la ontología será inconsistente.*
- **Clasificación de la ontología**: el razonador computa a partir de los axiomas declarados en el TBox, las relaciones de subclase entre todos los conceptos declarados explícitamente a fin de construir la jerarquía de clases.
- **Resolución de consultas**: a partir de la jerarquía de clases se pueden formular consultas como *conocer todas las subclases de un concepto, inferir nuevas subclases de un concepto, las superclases directas, etc*.
- **Precisiones sobre los conceptos de la jerarquía**: el razonador puede inferir cuáles son las clases a las que directamente pertenece y mediante la jerarquía inferida obtener todas las clases a las cuales indirectamente pertenece una clase o individuo dentro de la ontología.

Usando DL: Estándar DIG, DL Implementation Group

- Smallest propositionally DL (conceptos booleanos + restricciones de existencia) es $\mathcal{ALC}$ Attributive Concept Language with Complements
  - $\mathcal{ALCR}$+ (con roles transitivos) $\mathcal{ALCH}$ (con roles de inclusión) $\mathcal{ALCHR}$+ es $\mathcal{SH}$ conocido como $\mathcal{SHOIQ}$
- OWL DL based on $\mathcal{SHIQ}$ Description Logic
  - OWL extends $\mathcal{SHIQ}$ with datatypes and nominals ($\mathcal{SHOIN}(\mathbf{D}_n)$)
  - In fact it is equivalent to $\mathcal{SHOIN}(\mathbf{D}_n)$ DL
- OWL DL Benefits from many years of DL research
  - Well defined semantics
  - Formal properties well understood (complexity, decidability)
  - Known reasoning algorithms
  - Implemented systems (highly optimised)
- In fact there are three "species" of OWL (!)
  - OWL full is union of OWL syntax and RDF
  - OWL DL restricted to FOL fragment (≈ DAML+OIL)
  - OWL Lite is "simpler" subset of OWL DL

Recordemos: OWL Class Constructors

Constructor	DL Syntax	Example	FOL Syntax
intersectionOf	$C_1 \sqcap \ldots \sqcap C_n$	Human $\sqcap$ Male	$C_1(x) \wedge \ldots \wedge C_n(x)$
unionOf	$C_1 \sqcup \ldots \sqcup C_n$	Doctor $\sqcup$ Lawyer	$C_1(x) \vee \ldots \vee C_n(x)$
complementOf	$\neg C$	$\neg$Male	$\neg C(x)$
oneOf	$\{x_1\} \sqcup \ldots \sqcup \{x_n\}$	{john} $\sqcup$ {mary}	$x = x_1 \vee \ldots \vee x = x_n$
allValuesFrom	$\forall P.C$	$\forall$hasChild.Doctor	$\forall y.P(x,y) \rightarrow C(y)$
someValuesFrom	$\exists P.C$	$\exists$hasChild.Lawyer	$\exists y.P(x,y) \wedge C(y)$
maxCardinality	$\leqslant nP$	$\leqslant$1hasChild	$\exists^{\leqslant n} y.P(x,y)$
minCardinality	$\geqslant nP$	$\geqslant$2hasChild	$\exists^{\geqslant n} y.P(x,y)$

- XMLS datatypes as well as classes in ∀P.C and ∃P.C
  - Restricted form of DL concrete domains

Recordemos: RDFS Syntax

E.g., Person $\sqcap$ $\forall$hasChild.(Doctor $\sqcup$ $\exists$hasChild.Doctor):

```
<owl:Class>
  <owl:intersectionOf rdf:parseType=" collection">
    <owl:Class rdf:about="#Person"/>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasChild"/>
      <owl:toClass>
        <owl:unionOf rdf:parseType=" collection">
          <owl:Class rdf:about="#Doctor"/>
          <owl:Restriction>
            <owl:onProperty rdf:resource="#hasChild"/>
            <owl:hasClass rdf:resource="#Doctor"/>
          </owl:Restriction>
        </owl:unionOf>
      </owl:toClass>
    </owl:Restriction>
  </owl:intersectionOf>
</owl:Class>
```

Recordemos: OWL Axioms

Axiom	DL Syntax	Example
subClassOf	$C_1 \sqsubseteq C_2$	Human $\sqsubseteq$ Animal $\sqcap$ Biped
equivalentClass	$C_1 \equiv C_2$	Man $\equiv$ Human $\sqcap$ Male
disjointWith	$C_1 \sqsubseteq \neg C_2$	Male $\sqsubseteq \neg$Female
sameIndividualAs	$\{x_1\} \equiv \{x_2\}$	{President_Bush} $\equiv$ {G_W_Bush}
differentFrom	$\{x_1\} \sqsubseteq \neg\{x_2\}$	{john} $\sqsubseteq \neg${peter}
subPropertyOf	$P_1 \sqsubseteq P_2$	hasDaughter $\sqsubseteq$ hasChild
equivalentProperty	$P_1 \equiv P_2$	cost $\equiv$ price
inverseOf	$P_1 \equiv P_2^-$	hasChild $\equiv$ hasParent$^-$
transitiveProperty	$P^+ \sqsubseteq P$	ancestor$^+ \sqsubseteq$ ancestor
functionalProperty	$\top \sqsubseteq \leqslant 1P$	$\top \sqsubseteq \leqslant$1hasMother
inverseFunctionalProperty	$\top \sqsubseteq \leqslant 1P^-$	$\top \sqsubseteq \leqslant$1hasSSN$^-$

- Axioms (mostly) reducible to inclusion ($\sqsubseteq$)
  - $\mathbb{C} \equiv \mathbb{D}$ iff both $\mathbb{C} \sqsubseteq \mathbb{D}$ and $\mathbb{D} \sqsubseteq \mathbb{C}$
- Obvious FOL (Lógica de Primer Orden) equivalences
  - E.g., $\mathbb{C} \equiv \mathbb{D} \quad \Leftrightarrow \quad \forall x.\mathbb{C}(x) \leftrightarrow \mathbb{D}(x)$, $\quad \mathbb{C} \sqsubseteq \mathbb{D} \quad \Leftrightarrow \quad \forall x.\mathbb{C}(x) \rightarrow \mathbb{D}(x)$

Basic Inference Tasks

- Kw is correct (captures intuitions)
  - Does C subsume D w.r.t. ontology $\mathcal{O}$? ($C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ in *every* model $\mathcal{I}$ of $\mathcal{O}$)
- Kw is minimally redundant (no unintended synonyms)
  - Is C equivallent to D w.r.t. $\mathcal{O}$? ($C^{\mathcal{I}} = D^{\mathcal{I}}$ in *every* model $\mathcal{I}$ of $\mathcal{O}$)
- Kw is meaningful (classes can have instances)
  - Is C is satisfiable w.r.t. $\mathcal{O}$? ($C^{\mathcal{I}} \neq \emptyset$ in *some* model $\mathcal{I}$ of $\mathcal{O}$)
- Querying Kw
  - Is x an instance of C w.r.t. $\mathcal{O}$? ($x^{\mathcal{I}} \in C^{\mathcal{I}}$ in *every* model $\mathcal{I}$ of $\mathcal{O}$)
  - Is $\langle x,y \rangle$ an instance of R w.r.t. $\mathcal{O}$? ($(x^{\mathcal{I}},y^{\mathcal{I}}) \in R^{\mathcal{I}}$ in *every* model $\mathcal{I}$ of $\mathcal{O}$)

- Above problems can be solved using highly optimised DL reasoners

DL Reasoning: Basics

- Tableau algorithms used to test satisfiability (consistency)
- Decompose C syntactically
  - Infer constraints on elements of model
- Tableau rules correspond to constructors in logic ($\sqcap$, $\sqcup$ etc)
- Stop when no more rules applicable or clash occurs
  - Clash is an obvious contradiction, e.g., $\mathrm{A}(\mathrm{x})$, $\neg\mathrm{A}(\mathrm{x})$
- Cycle check (blocking) may be needed for termination
- C satisfiable iff rules can be applied such that a fully expanded clash free tree is constructed

- Un razonador *tableaux* posee sólo la funcionalidad de verificar la satisfactibilidad de un ABox con respecto a un TBox. Todas las otras tareas de razonamiento pueden ser reducidas a pruebas de consistencia con la KB mediante la transformación apropiada.

DL Reasoning: Advanced Techniques

- Satisfiability w.r.t. an Ontology $\mathcal{O}$
  - For each axiom $C \sqsubseteq D \in \mathcal{O}$, add $\neg C \sqcup D$ to every node label
- More expressive DLs
  - Basic technique can be extended to deal with
    - Role inclusion axioms (role hierarchy)
    - Number restrictions
    - Inverse roles
    - Concrete domains/datatypes
    - Aboxes
    - etc.

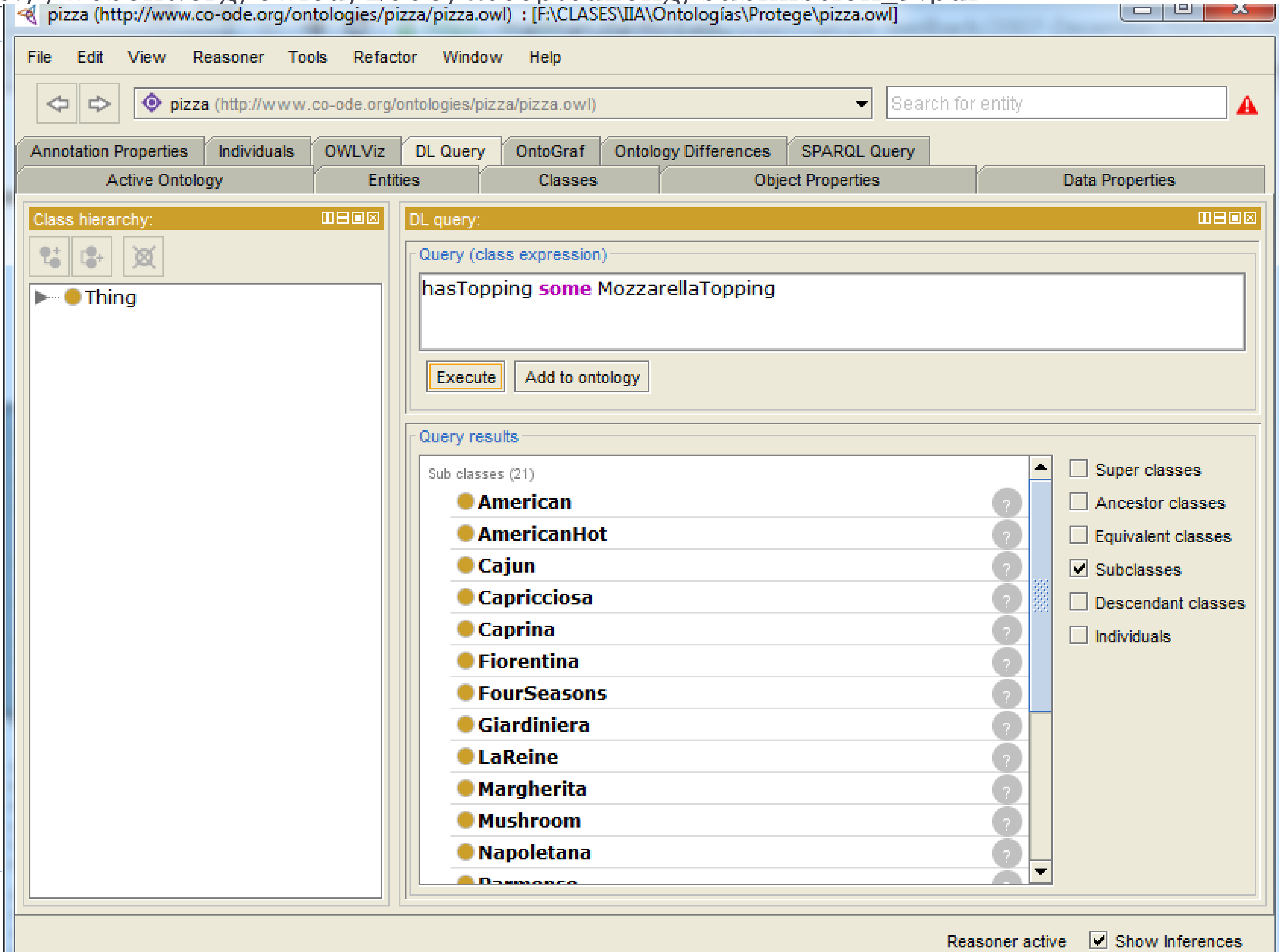
Comparison of Reasoners for large Ontologies in the OWL 2 EL Profile

Reasoning Characteristics

	CB	CEL	FaCT++	HermiT	Pellet	RP	SR	TrOWL (REL)
Methodology	consequence-based	completion rules	tableau-based	hypertableau	tableau-based	tableau-based	completion rules	approximation (completion rules)
Soundness	+	+	+	+	+	+	+	+ (+)
Completeness	+	+	+	+	+	+	+	- (+)
Expressivity	Horn $\mathcal{SHIF}$	$\mathcal{EL}^{+}$	$\mathcal{SROIQ}(\mathcal{D})$	$\mathcal{SROIQ}(\mathcal{D})$	$\mathcal{SROIQ}(\mathcal{D})$	$\mathcal{SHIQ}(\mathcal{D}-)$	$\mathcal{EL}^{+}$	third-party reasoner (approximating SROIQ; subset of $\mathcal{EL}^{++}$)
Incremental Classification (addition/removal)	-/-	+/-	-/-	-/-	+/+	-/-	+/-	-/-
Rule Support	-	-	-	+ (SWRL)	+ (SWRL)	+ (SWRL, nRQL)	-	-
Justifications	-	+	-	-	+	+	-	-
ABox Reasoning	-	+	+	+	+ (SPARQL)	+ (SPARQL, nRQL)	-	+ (SPARQL)

DL Query:

http://webont.org/owled/2006/acceptedLong/submission_9.pdf



Ejemplos: DL Queries

- `hasTopping only (MozzarellaTopping`
  `or PeperoniSausageTopping`
  `or TomatoTopping)`
- `hasTopping some MozzarellaTopping`
- `hasTopping some PeperoniSausageTopping`
- `hasTopping some TomatoTopping`

SPARQL

skos.um.es/TR/rdf-sparql-query/

Aplicaciones Google Disney Junior Poisson Rouge . Red... Protocolos TCP/IP » Otros marcadore

W3C Recommendation

SPARQL Lenguaje de consulta para RDF

Recomendación del W3C de 15 de enero de 2008

Versión original (en Inglés):
http://www.w3.org/TR/2008/REC-rdf-sparql-query-20080115/

Última versión:
http://www.w3.org/TR/rdf-sparql-query/

Versión anterior:
http://www.w3.org/TR/2007/PR-rdf-sparql-query-20071112/

Editores:
Eric Prud'hommeaux, W3C <eric@w3.org>
Andy Seaborne, Hewlett-Packard Laboratories, Bristol <andy.seaborne@hp.com>

SPARQL

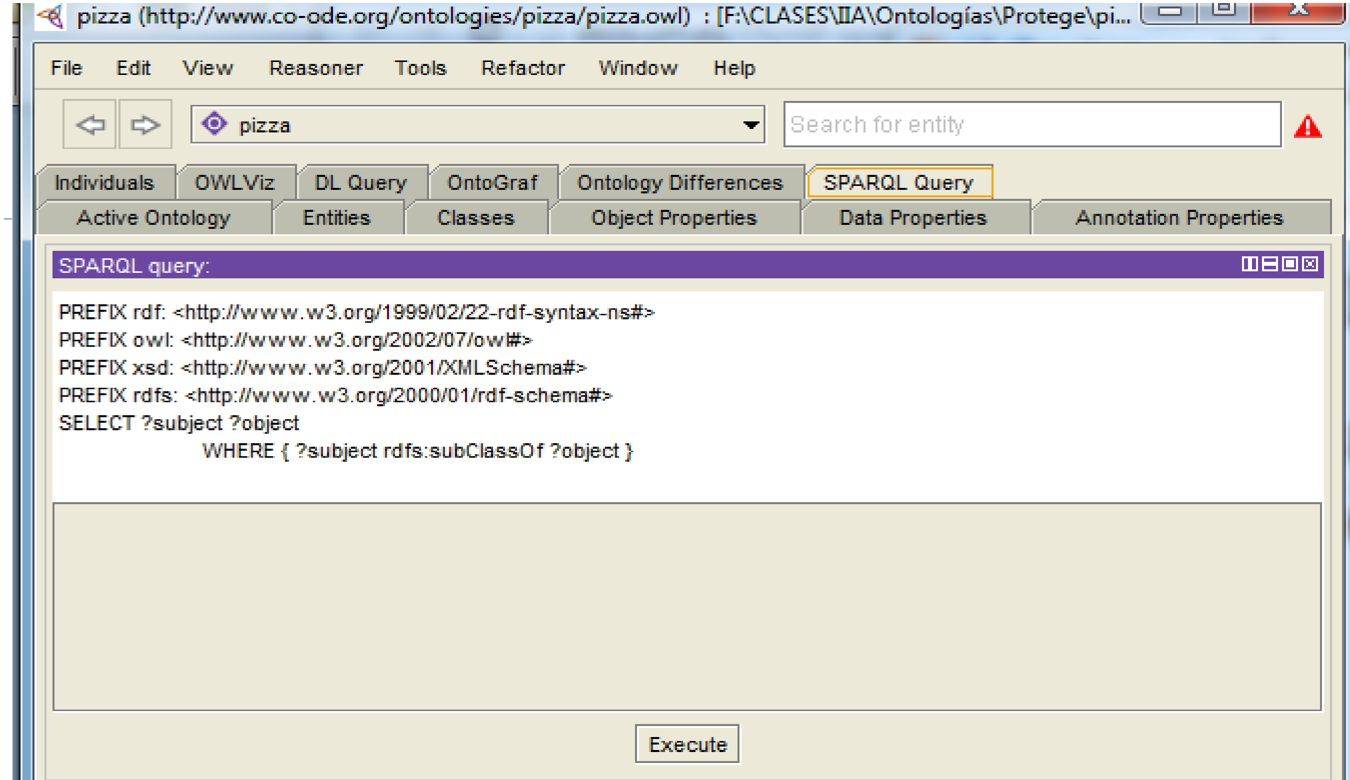
Form of the query atom	**Condition on interpretation**
Type(a, C)	$\sigma(a) \in C^{\mathcal{I}}$
PropertyValue(a, p, v)	$\langle\sigma(a), \sigma(v)\rangle \in p^{\mathcal{I}}$
SameAs(a, b)	$\sigma(a) = \sigma(b)$
DifferentFrom(a, b)	$\sigma(a) \neq \sigma(b)$
SubClassOf(C_1, C_2)	$C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$
EquivalentClass(C_1, C_2)	$C_1^{\mathcal{I}} = C_2^{\mathcal{I}}$
DisjointWith(C_1, C_2)	$C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}} = \emptyset$
ComplementOf(C_1, C_2)	$C_1^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C_2^{\mathcal{I}}$
SubPropertyOf(p, q)	$p^{\mathcal{I}} \subseteq q^{\mathcal{I}}$
EquivalentProperty(p, q)	$p^{\mathcal{I}} = q^{\mathcal{I}}$
Functional(p)	$\langle x, y\rangle \in p^{\mathcal{I}}$ and $\langle x, z\rangle \in p^{\mathcal{I}}$ implies $y = z$
InverseFunctional(p)	$\langle y, x\rangle \in p^{\mathcal{I}}$ and $\langle z, x\rangle \in p^{\mathcal{I}}$ implies $y = z$
Transitive(p)	$\langle x, y\rangle \in p^{\mathcal{I}}$ and $\langle y, z\rangle \in p^{\mathcal{I}}$ implies $\langle x, z\rangle \in p^{\mathcal{I}}$
Symmetric(p)	$\langle x, y\rangle \in p^{\mathcal{I}}$ implies $\langle y, x\rangle \in p^{\mathcal{I}}$
Annotation(s, p_a, o)	$\langle s, o\rangle \in p_a^{\mathcal{I}}$

Table 2. Satisfaction of a SPARQL-DL query atom w.r.t. an interpretation

```
?C rdfs:subClassOf _:x .
_:x rdf:type owl:Restriction .
_:x owl:onProperty ex:q .
_:x ?p ?C .
```

SPARQL:

- Ejemplo:



Trabajando sobre el proyecto en formato OWL, formamos las consultas en *SPARQL*. Las mismas se corresponden a escenarios especifico que presentan los problemas anteriores.

1. *Buscar las Personas que hayan creado alguna Pista en el año 1969.*

```
PREFIX cla: <http://www.owl-ontologies.com/Ontology1383937954.owl#>
PREFIX ind: <http://www.owl-ontologies.com/>
SELECT DISTINCT ?autor
WHERE {
  ?pista rdf:type cla:Pista .
  ?pista cla:Año_creación_obra ?año .
  ?pista cla:Autores ?autor .
}
HAVING (?año = 1969)

RESULT:
  Otis_Spann
```

Research Challenges

- Increased expressive power
  - Existing DL systems implement (at most) $\mathcal{SHIQ}$
  - OWL extends $\mathcal{SHIQ}$ with datatypes and nominals ($\mathcal{SHOIN}(\mathbf{D}_n)$)
- Scalability
  - Very large ontologies
  - Reasoning with (very large numbers of) individuals
- Other reasoning tasks (non-standard inferences)
  - Querying
  - Matching
  - Least common subsumer
  - ...
- Tools and Infrastructure
  - Support for large scale ontological engineering and deployment

Summary

- An Ontology is an engineering artifact consisting of:
  - A vocabulary of terms
  - An explicit specification their intended meaning
- Ontologies are set to play a key role in many applications
  - e-Science, Medicine, Databases, Semantic Web, etc.
- Reasoning is important because
  - Understanding is closely related to reasoning
  - Essential for design, maintenance and deployment of ontologies
- Reasoning support based on DL systems
  - Sound and complete reasoning
  - Highly optimised implementations
- Challenges remain
  - Expressive power; scalability; new reasoning tasks; tools and infrastructure

Select Bibliography

- Ian Horrocks, Peter F. Patel-Schneider, and Frank van Harmelen. From SHIQ and RDF to OWL: The making of a web ontology language. Journal of Web Semantics, 2003.
- Franz Baader, Ian Horrocks, and Ulrike Sattler. Description logics as ontology languages for the semantic web. In *Festschrift in honor of Jörg Siekmann*, LNAI. Springer, 2003.
- I. Horrocks and U. Sattler. Ontology reasoning in the SHOQ(D) description logic. In Proc. of IJCAI 2001.

All available from http://www.cs.man.ac.uk/~horrocks/Publications/