OWL como una Lógica de Descripciones

Paula Severi

University of Leicester

Curso Noviembre-Diciembre 2018. Facultad de Ingeniería.
Universidad de la República.
Montevideo, Uruguay.

WEB SEMÁNTICA	web +lógica para chequear consistencia y realizar consultas
OWL (web ontology language)	una lógica de descripciones donde los constructores de la lógica se escriben como etiquetas de XML

Área de aplicación mas exitosa de la lógica de descripciones.

Aplicaciones de la web semántica

1. Wikis semánticas (usan RDF, SPARQL y OWL)
2. Redes sociales semánticas. Usan OWL ontologia `FOAF` (friend of a friend).
3. Blogs semánticos
4. Ontologias en muchas aplicaciones: biologia, geografia, etc.

OWL: Lenguaje de ontologías para la web

- OWL recomendación de la W3C Recommendation desde 2004.
- OWL 2 es recomendación de la W3C desde 2009.
- fragmentos decidibles de la lógica de primer orden.
- razonamiento sobre ontologías es automático: decidible.

Razonador semántico

Razonador

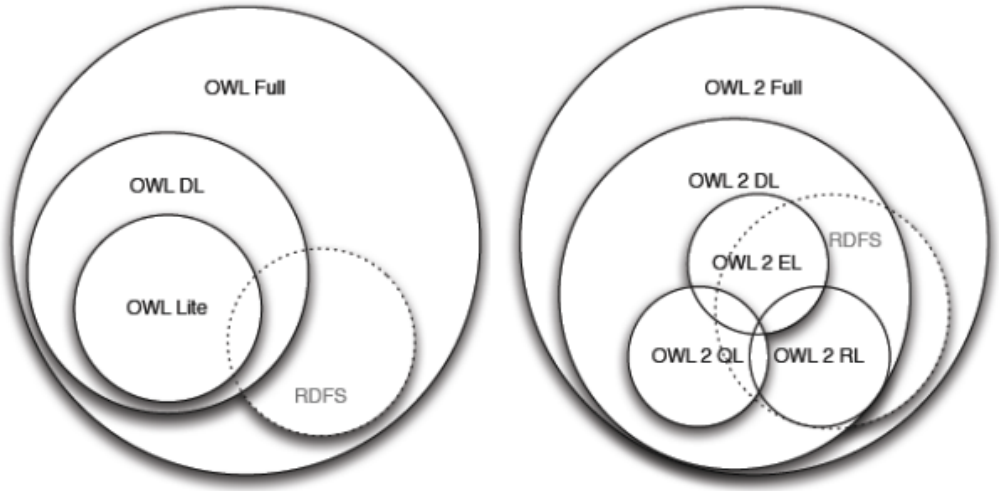
Es software que realiza razonamiento automático, e.g.

- chequea consistencia de una ontologia
- infiere consecuencias lógicas (forma de encuestas o queries)

Ejemplos:

- Pellet
- RacerPro
- FaCT++
- HermiT

OWL: variantes y perfiles

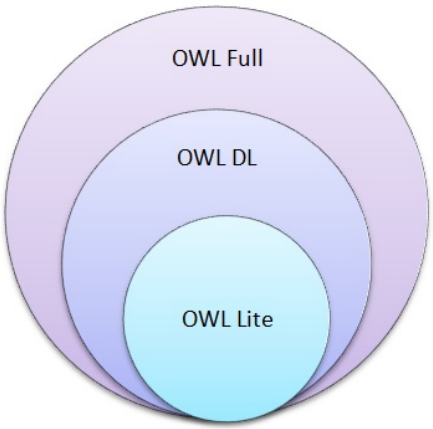


OWL 1 OWL 2 Profiles

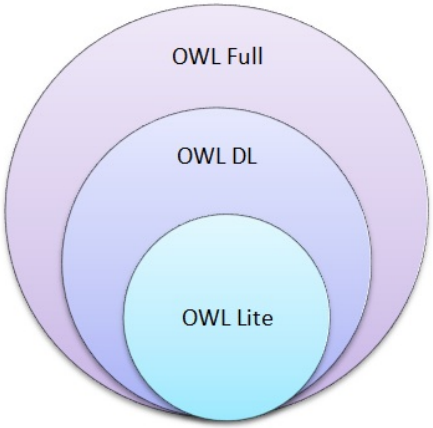
Variantes de OWL

Variantes:

- OWL Lite
- OWL 2 DL (es una extensión de OWL DL)
- OWL Full



Variantes de OWL

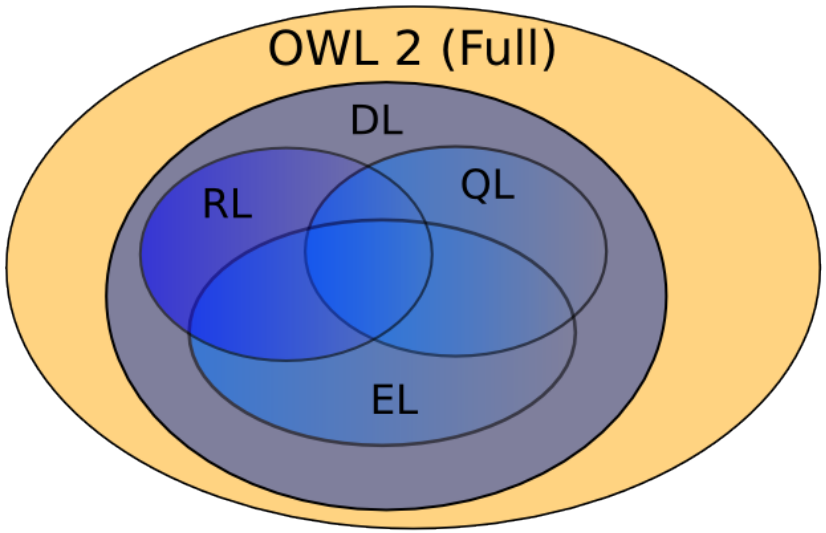


- OWL Full es indecidible.
- OWL 2 DL es decidible corresponde a la lógica de descripciones $\mathcal{SROIQ}^{\mathcal{D}}$. Complejidad N2Exp.
- OWL DL (también decidible) corresponde a una lógica de descripciones $\mathcal{SHOIN}^{\mathcal{D}}$.
- OWL Lite corresponde a $\mathcal{SHIF}^{\mathcal{D}}$

Perfiles de OWL

Son sublenguajes de OWL 2. Tienen menos poder expresivo pero son mas eficientes (complejidad polinomial o logarítmica):

- OWL 2 EL. Eficiente cuando uno tiene muchas clases y muchas propiedades.
- OWL 2 QL. Eficiente cuando uno tiene muchas instancias y quiere hacer encuestas sobre estas instancias.
- OWL 2 RL. Para razonamiento sobre ontologias grandes sin sacrificar tanto poder expresivo.



OWL 2: clases y propiedades

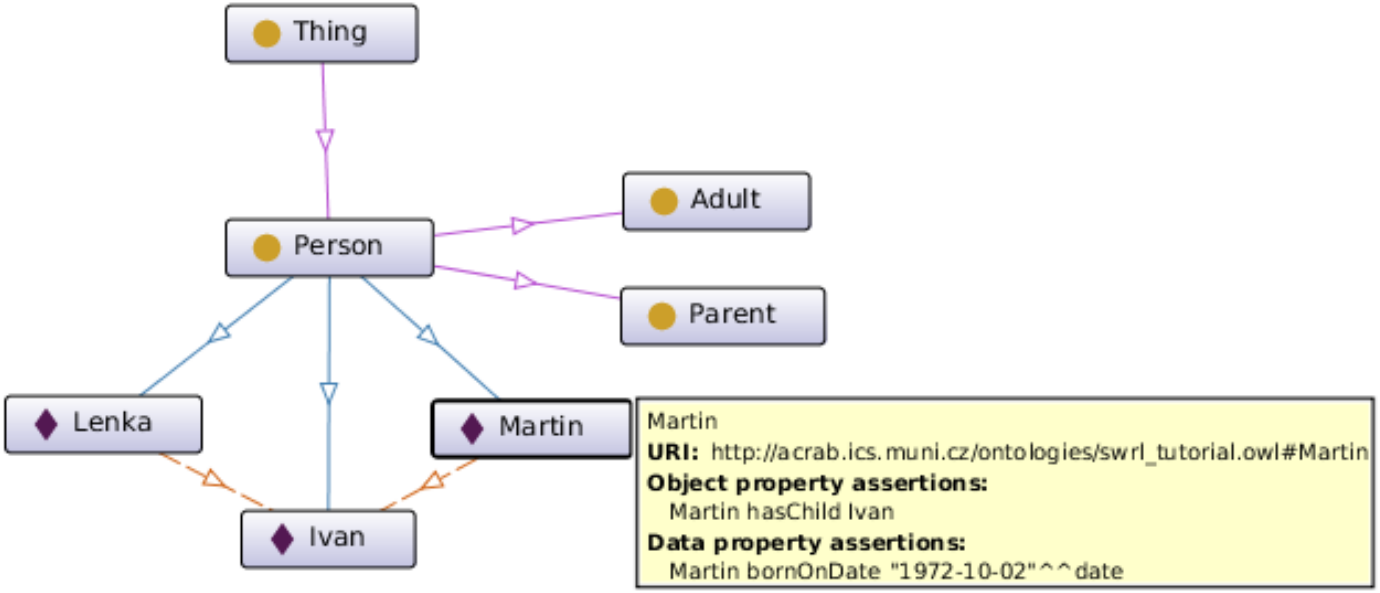
Lógica de Descripciones	OWL	Teoría de Conjuntos
Concepto (descripción de concepto)	Clase	Conjunto
Rol (descripción de rol)	Propiedad	Relación Binaria

Sintaxis de OWL vs DL

Correspondencia entre sintaxis

Constructor OWL	DL	Sintaxis Manchester
SubClassOf	$\sqsubseteq$	SubClassOf
Thing	$\top$	Thing
Nothing	$\bot$	Nothing
intersectionOf	$C \sqcap D$	C and D
unionOf	$C \sqcup D$	C or D
complementOf	$\neg C$	not A
oneOf	$\{a_1, a_2, \ldots a_n\}$	$\{a_1, a_2, \ldots a_n\}$
someValuesFrom	$\exists R.C$	R some C
allValuesFrom	$\forall R.C$	R only C
minCardinality	$\geqslant n\ R$	R min n
maxCardinality	$\leqslant n\ R$	R max n
cardinality	$= n\ R$	R exactly n
hasValue	$\exists R.\{a\}$	R value a

Ejemplo



$$\begin{array}{ll} \text{Adult} \sqsubseteq \text{Person} & \text{Person}(\text{Martin}) \\ \text{Parent} \sqsubseteq \text{Person} & \text{hasChild}(\text{Martin}, \text{Ivan}) \\ \text{Person} \sqsubseteq \text{Thing} & \text{bornOnDate}(\text{Martin}, 1972{-}10{-}02) \end{array}$$

$$\text{Parent} \equiv \text{Person} \sqcap \exists \text{hasChild}.\text{Person}$$

OWL como una Lógica de Descripciones

OWL 2

1. Todos los constructores de $\mathcal{ALC}$,
2. restricciones numéricas (en la cardinalidad),
3. clases cerradas (nominales),
4. igualdad y diferencia entre individuos,
5. inclusion y equivalencia de roles (jerarquías de roles),
6. inversión y composicón de roles,
7. transitividad, simetría, funcionalidad, funcionalidad de la inversión de roles,
8. tipos de datos, restricciones a intervalos.

Sintaxis de $\mathcal{SROIQ}$

- $\mathcal{S}$ significa $\mathcal{ALC}$+transitividad de roles
- $\mathcal{R}$ significa axiomas de inclusión de roles generalizado
- $\mathcal{O}$ significa nominales (one element)
- $\mathcal{I}$ significa roles inversos
- $\mathcal{Q}$ significa restricciones de cardinalidad calificada

Conceptos de $\mathcal{SROIQ}$

$$C, D := \top \mid A \mid \neg C \mid C \sqcap D \mid C \sqcup D \mid \exists R.C \mid \forall R.C$$
$$\{a\} \mid \leq nS.C \mid \geq nS.C \mid \exists R.\mathsf{Self}$$

donde $A \in \mathsf{N_C}$, $R, S \in \mathsf{N_R}$ y S es simple.

Base de conocimiento en OWL: TBOX + RBOX+ ABOX

Base de conocimiento $\mathcal{K} = \langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle$

$\mathcal{T}$ es TBox (conocimiento terminológico)

$C \sqsubseteq D$

$\mathcal{R}$ es RBox (conocimiento sobre roles)

$S_1 \circ \ldots \circ S_n \sqsubseteq R$

$\mathsf{Sym}(R)$	$\mathsf{Tran}(R)$	$\mathsf{Asym}(R)$
$\mathsf{Refl}(R)$	$\mathsf{Irref}(R)$	$\mathsf{Disj}(R, S)$

$\mathcal{A}$ es ABox (conocimiento asercional)

$a : C$	$a \neq b$
$< a, b >: R$	$< a, b >: \neg R$

Restricciones de cardinalidad

Sea $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ una interpretación de $\mathcal{ALC}$.

Semántica de $\geq n\ R$ donde n un natural no negativo

$$(\geq n\ R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \mathsf{card}(\mathsf{imagen}(x, C)) \geq n\}$$

Semántica de $\leq n\ R$ donde n un natural no negativo

$$(\leq n\ R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \mathsf{card}(\mathsf{imagen}(x, C)) \leq n\}$$

Notación:

$\mathsf{card}(X)$ es la cardinalidad de conjunto X

$\mathsf{imagen}(x, C) = \{y \in C^{\mathcal{I}} \mid (x, y) \in R^{\mathcal{I}}\}$

Dualidades

Se cumplen las siguientes equivalencias:

$$\begin{aligned} \neg(\leqslant n\, R.C) &\equiv \geqslant (n+1)\, R.C \\ \neg(\geqslant (n+1)\, R.C) &\equiv \leqslant n\, R.C \\ \neg(\geqslant 0\, R.C) &\equiv \bot \end{aligned}$$

Tenemos también que:

$$\begin{aligned} \exists R.C &\equiv \geqslant 1\, R.C \\ \forall R.C &\equiv \leqslant 0\, R.\neg C \end{aligned}$$

Podriamos sacar $\forall$ y $\exists$ de la sintaxis.

Ejemplo restricción de cardinalidad

$$\texttt{Persona} \sqsubseteq \leq 2\ \texttt{tienePadres.Persona}$$

Ejemplo

$$\texttt{Persona} \sqsubseteq \leq 1\texttt{tieneMadre.Persona}$$

$$\texttt{tieneMadre}(\texttt{Juan}, \texttt{Alicia})$$
$$\texttt{tieneMadre}(\texttt{Juan}, \texttt{Maria})$$

Esta ontologia es consistente. El razonador iguala `Alicia` con `Maria`.

Nominales $\{a_1, \ldots, a_n\}$ = enumerados

Semántica de $\{a_1, \ldots, a_n\}$ donde n un natural no negativo

.

$$\{a_1, \ldots, a_n\}^{\mathcal{I}} = \{a_1^{\mathcal{I}}, \ldots, a_n^{\mathcal{I}}\}$$

Se puede agregar solamente e constructor unario $\{a\}$ al lenguaje y $\{a_1, \ldots, a_n\}$ se puede expresar usando $\sqcup$.

$$\{a_1, \ldots, a_n\} \equiv \{a_1\} \sqcup \{a_2\} \ldots \sqcup \{a_n\}$$

Observación.
$\mathcal{I} \models (a, b) : R$ si y solo si $\mathcal{I} \models a : \exists R.\{b\}$

Ejemplo de nominales

- Tbox contiene:

$$\texttt{MarcoFormas} \equiv \{\texttt{chata}, \texttt{concava}, \texttt{gaussiana}\}$$

- $$\texttt{Paises} \equiv \{\texttt{Uruguay}, \texttt{Argentina}, \texttt{Brasil}\}$$

Dominio y Rango de Roles

Dominio

$\mathsf{dom}(R) = C$ se define como $\exists R.\top \sqsubseteq C$

Rango

$\mathsf{rango}(R) = D$ se define como $\top \sqsubseteq \forall R.D$

Dos tipos de propiedades (roles) en OWL:

1. Object property. Rango es una clase definida por el usuario.
2. Datatype property. Rango es un datatype.

El dominio no puede ser un datatype.

Ejemplo de dominio y rango

En Manchester syntax escribimos:

ObjectProperty: haPintado
Domain: Pintor
Range: Pintura

Esto equivale a poner los siguientes axiomas:

$$\exists \texttt{haPintado}.\top \sqsubseteq \texttt{Pintor}$$
$$\top \sqsubseteq \forall \texttt{haPintado}.\texttt{Pintura}$$

Igualdad y Diferencia entre individuos

SUN (en inglés UNA): Supuesto de Unicidad de Nombres

Nombres distintos de individuos denotan distintos elementos, i.e. $a \neq b$ implica $a^{\mathcal{I}} \neq b^{\mathcal{I}}$.

Tradicionalmente, las lógicas de descripciones asumen SUN. Sin embargo, OWL no asume SUN.
Una Abox puede contener expresiones de la forma $a \neq b$

Igualdad de individuos

$a = b$ se define como $\{a\} = \{b\}$.

Diferencia entre individuos

$a \neq b$ se define como $\{a\} \sqcap \{b\} \sqsubseteq \bot$.

Jerarquía de Roles

- Axioma de inclusión de roles: $R \sqsubseteq S$
- Equivalencia entre roles: $R \equiv S$.

$$\texttt{tienePadre} \sqsubseteq \texttt{tienePariente}$$
$$\texttt{tieneAbuelo} \sqsubseteq \texttt{tienePariente}$$

Operaciones de Roles

- Rol inverso de R se denota como R^{-}.

  Ejemplo,
  `tieneHijo` es el rol inverso de `tienePadre`.

  En Manchester Syntax, `inverse (tienePadre)` denota el rol inverso de `tienePadre`.

- Composición de roles se denota como $R \circ S$. Ejemplo:

$$\texttt{tienePadre} \circ \texttt{tieneHermano} \sqsubseteq \texttt{tieneTio}$$

$$\texttt{tieneAncestro} \circ \texttt{tienePadre} \sqsubseteq \texttt{tieneAncestro}$$

  Inclusiones de roles permitidas. $S_1 \circ \ldots S_n \sqsubseteq R$ tiene que ser regular.

Jerarquía de Roles

El conjunto de nombres de roles (roles atómicos) incluye todos los nombres R, sus inversas R^- y el rol universal U.

Axiomas de inclusión de roles regulares

Sean $S_1, \ldots S_n, R$ nombres de roles. Un *axioma de roles generalizado* es un enunciado de la forma $S_1 \circ \ldots S_n \sqsubseteq R$. Se dice que dicho axioma es regular si existe un orden parcial estricto $\prec$ sobre nombres de roles tal que:

1. $S \prec R$ sii $S^- \prec R$
2. los axiomas de inclusión de roles son de una de estas forma:

$$R \circ R \sqsubseteq R \quad R^- \sqsubseteq R \quad S_1 \circ \ldots \circ S_n \sqsubseteq R$$
$$R \circ S_1 \circ \ldots \circ S_n \sqsubseteq R \quad S_1 \circ \ldots \circ S_n \circ R \sqsubseteq R$$

tal que R no es un nombre de rol inverso y $S_i \prec R$ para todo $1 \leq i \leq n$.

Ejemplo

Pero lo siguiente no esta permitido:

$$\texttt{hasParent} \circ \texttt{hasHusband} \sqsubseteq \texttt{hasFather}$$
$$\texttt{hasFather} \sqsubseteq \texttt{hasParent}$$

Esta jerarquía de roles no es regular ya que:

$$\texttt{hasParent} \prec \texttt{hasFather}$$
$$\texttt{hasFather} \prec \texttt{hasParent}$$

pero $\prec$ tiene que ser estricto.

Características de Roles

- Transitividad $\mathsf{Trans}(R)$
- Simetría $\mathsf{Sym}(R)$ es $R \equiv R^{-}$ y asimetría $\mathsf{Asym}(R)$.
- Funcionalidad $\mathsf{Fun}(R)$ es $\top \sqsubseteq \leq 1\ R$.
- Funcionalidad inversa es $\top \sqsubseteq \leq 1\ R^{-}$.
- Disjunción $\mathsf{Disj}(R, S)$
- Irreflexividad $\mathsf{Irref}(R)$, reflexividad $\mathsf{Refl}(R)$.

Roles Simples

Un rol es simple si pasa lo siguiente:

- si no occurre en el lado derecho de un axioma de inclusión de roles,
- la inversa de un rol simple es simple,
- Si R ocurre en el lado derecho de un axioma de inclusión de roles de la forma $S \sqsubseteq R$ donde S es simple entonces R es un rol simple.

Restricción sobre ciertos roles

Los roles que occurren en axiomas de reflexividad, irreflexividad, asimetría y disjunción tienen que ser simples.

Ejemplo

$$R \sqsubseteq R_1 \qquad R_1 \circ R_2 \sqsubseteq R_3 \qquad R_3 \sqsubseteq R_4$$

Los roles simples son:

R	R^{-1}
R_1	R_1^{-1}
R_2	

Self

Un suicida es alguien que se mata a si mismo.

$$\text{Suicida} \sqsubseteq \exists \text{mata}.\mathit{Self}$$

$$(\exists S.\mathit{Self})^{\mathcal{I}} = \{x \mid (x,x) \in S^{\mathcal{I}}\}$$

OWL como una Lógica de Descripciones

OWL 2 tiene

1. todos los constructores de $\mathcal{SROIQ}$
2. claves
3. punning
4. disjunción de roles
5. dominios concretos (datatypes), restricciones a intervalos

En OWL tenemos mas constructores que los vistos hasta ahora para $\mathcal{SROIQ}$.

Constructor de conceptos para el uso de claves

$$\mathsf{hasKeys}(S_1, \ldots, S_n)$$

donde $S_1, \ldots, S_n$ son roles simples.

Claves (Keys)

Ejemplo 1

```
Person  hasKey CI

Person (Luisa)  CI(Luisa,1990322)
Person (Maria)  CI(Maria,1990322)
```

Como Luisa y Maria tienen *la misma CI* entonces el razonador infiere que Maria = Luisa.

Claves (Keys)

Ejemplo 2

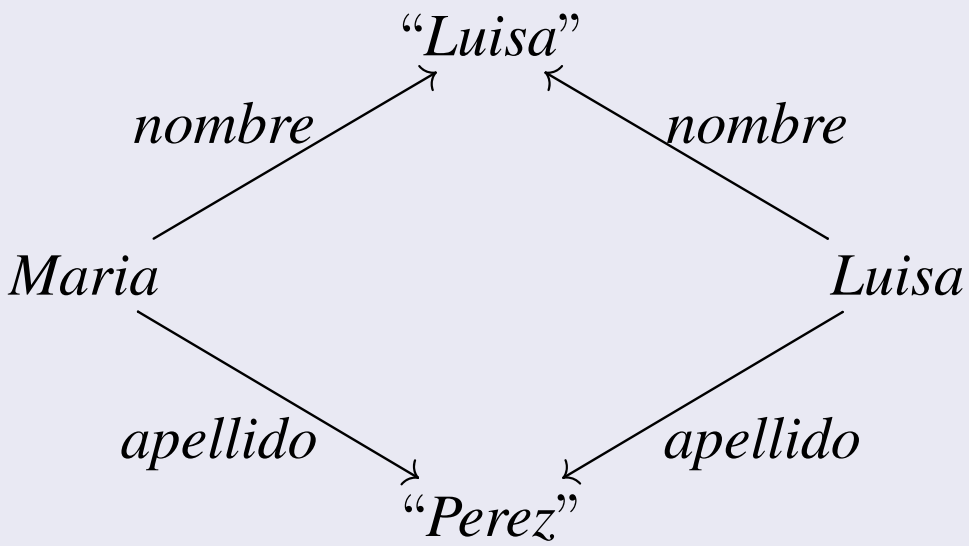
```
Person  subclassof hasKey (nombre, apellido)

Person (Luisa)
nombre (Luisa, "Luisa")
apellido (Luisa, "Perez")

Person(Maria)
nombre (Maria, "Luisa")
apellido (Maria, "Perez")
```

Claves (Keys)

Ejemplo 2 (diagrama)



Como los individuos Maria y Luisa de la clase Persona *coinciden* en los valores de `nombre` y `apellido`, el razonador infiere que $Maria = Luisa$.

Punning

1. Es una forma restringida de meta-modelado.

2. Mismo nombre se puede usar en diferentes “posiciones gramaticales” pero para el razonador son entidades diferentes.

Punning

1. Mismo nombre se puede usar como individuo y como clase.

   animal(oveja) oveja(dolly)

2. Mismo nombre se puede usar para individuo y propiedad.

   localizacion(Montevideo,Uruguay)
   PropiedadObsoleta(localizacion)

3. Mismo nombre para clase y propiedad.

   PersonaEmpresa $\sqsubseteq$ Asociacion
   PersonEmpresa (Gates, Microsoft)

Para el razonador son entidades diferentes.

OWL como una Lógica de Descripciones

OWL 2 tiene

1. todos los constructores de $\mathcal{SROIQ}$
2. claves
3. punning
4. disjunción de roles
5. dominios concretos (datatypes), restricciones a intervalos

Problemas de Inferencia para Bases de Conocimiento

Sea $\mathcal{K}$ una base de conocimiento.

- $\mathcal{K}$ es consistente si existe un modelo $\mathcal{I}$ de $\mathcal{K}$.
- C está subsumido (incluido) en D con respecto a $\mathcal{K}$ si $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ se satisface para todo model $\mathcal{I}$ de $\mathcal{K}$. Notación: $\mathcal{K} \models C \sqsubseteq D$.
- a es una instancia de C con respecto a $\mathcal{K}$. $a^{\mathcal{I}} \in C^{\mathcal{I}}$ para todo modelo $\mathcal{I}$ de $\mathcal{K}$. Notación: $\mathcal{K} \models a : C$.

Reducción a la inconsistencia de una base

El algoritmo de tableaux chequea la consistencia de una base de conocimiento (existencia de un modelo). Los otros problemas se pueden reducir a este:

Reducción a la inconsistencia de una base

1. Inclusión. $\mathcal{K} \models C \sqsubseteq D$ sii $\mathcal{K} \cup \{C \sqcap \neg D(a)\}$ es inconsistente donde a es un individuo nuevo que no aparece en $\mathcal{K}$.
2. Igualdad. $\mathcal{K} \models C = D$ sii $\mathcal{K} \cup \{C \sqsubseteq D\}$ y $\mathcal{K} \cup \{D \sqsubseteq C\}$ son consistentes.
3. Clase consistente. $\mathcal{K} \models C \sqsubseteq \bot$ sii $\mathcal{K} \cup \{C(a)\}$ es inconsistente donde a es un individuo nuevo que no aparece en $\mathcal{K}$.
4. Chequeo de instancia. $\mathcal{K} \models C(a)$ sii $\mathcal{K} \cup \{\neg C(a)\}$ es inconsistente.
5. Recuperación de instancias. Encontrar todos los individuos a tal que $C(a)$, por cada individuo a en la base de concimiento chequear si $\mathcal{K} \models C(a)$.

Bibliografía

- Capítulo 5 (OWL Formal Semantics) de *Foundations of Semantic Web Technologies.* Pascal Hitzler, Markus Krötzsch and Sebastian Rudolph. 2009 Chapman & Hall/CRC.

- Capítulo 2 del *Description Logic Handbook: Theory, Implementation, and Applications.* 2003.

Sitios web recomendados

- *OWL 2 new features and rationale.*

  http://www.w3.org/TR/owl2-new-features/

- *OWL 2 primer.*

  http://www.w3.org/TR/owl2-primer/

- Protégé

  http://protege.stanford.edu/

- Protégé Tutorial

  http://mowl-power.cs.man.ac.uk/protegeowltutorial/resources/ProtegeOWLTutorialP4_v1_3.pdf

- Manchester Syntax.

  http://www.co-ode.org/resources/reference/manchester_syntax/