Ontologías y Web Semántica

Ontologías y OWL

Pablo R. Fillottrani

Depto. Ciencias e Ingeniería de la Computación
Universidad Nacional del Sur

septiembre 2017

Ontologías y OWL

Ontologías

DL estructurales

DL proposicionales

OWL

Definición

- una ontología es una especificación formal de una conceptualización compartida de un dominio
- por ser formal
  - el significado es no ambiguo
  - se expresa mediante un lenguaje formal
  - permite razonamiento, haciendo explícita información implícita

- por ser una especificación
  - hace explícitas las suposiciones del dominio
  - clarifica el significado y permite razonamiento
  - balance entre información explícita e implícita
- por ser compartida
  - debe establecerse mediante consenso
  - facilita su uso

- por ser una conceptualización de un dominio
  - trata sobre una concreta parte del mundo
  - permite formar una idea del dominio en la mente de las personas (y de las máquinas)
- forman el esqueleto de la Web Semántica, definiendo vocabulario básico y permitiendo el razonamiento

Usos de una ontología

- compartir conocimiento en común entre personas y agentes de software
- permitir el reuso de información de un dominio
- hacer explícitas el conocimiento supuesto del dominio
- separar el conocimiento del dominio del conocimiento de una aplicación
- analizar el conocimiento del dominio

- clases grupos de individuos con características en común
- generalmente con definición implícita
- individuos son objetos particulares del dominio, que pueden ser instancias de clases
- existe diferencia entre individuos y valores de tipos de datos
- relaciones definen las posibles conexiones entre individuos y/o valores
- generalmente son asociadas a las clases, junto con las propiedas sobre su instanciación
- generalmente todo esto se especifica mediante un lenguaje gráfico
- axiomas describen propiedades adicionales sobre el dominio, generalmente especificadas en un lenguaje lógico

Ingeniería de ontologías

- principios
  1. no existe un única forma de modelar un dominio, siempre existen alternativas viables. La mejor solución siempre depende de las aplicaciones, y sus futuras extensiones
  2. necesariamente el desarrollo de una ontología es un proceso iterativo
  3. los conceptos deben ser cercanos a objetos físicos o lógicos, y las relaciones deben ser aquellas interesantes para el dominio

Pasos sugeridos

1. determinar el dominio y el ámbito de la ontología
2. considerar reusar ontologías existentes
3. enumerar términos importantes en la ontología
4. definir clases y su jerarquía
5. definir las propiedades de las clases (atributos y relaciones)
6. definir las facetas de las propiedades
7. crear instancias de los conceptos importantes

Caracterizando clases y su jerarquías

- controlar que la jerarquía de conceptos respeta la relación *es-un*
- analizar la relación entre hermanos en la jerarquía
- analizar el uso de herencia múltiple
- definir cuándo introducir un nuevo concepto o no
  - tienen propiedades adicionales
  - restringen en forma diferente los valores
  - participan en relaciones diferentes

- nueva clase o nuevo atributo?
  - si los conceptos con atributos diferentes se usan en restricciones en otras clases, entonces conviene crear una clase
  - si la distinción es importante para el dominio, entonces conviene crear una clase
  - la clase a la que un individuo pertenece no debería cambiar seguido
- una instancia o una clase?
  - los individuos representan los conceptos más específicos disponibles
  - si los individuos forman naturalmente una jerarquía, entonces conviene representarlos como clases

Caracterizando propiedades de las clases

- controlar propiedades inversas
- definir valores por defecto
- definir valores enumerados
- definir rango de valores
- definir cardinalidad máxima y mínima

Nombres

- es conveniente definir una convención sobre nombres, y adherirse estrictamente a la misma
  - capitalización
  - delimitadores
  - prefijos y sufijos
  - singular o plural
  - tipos de los nombres
  - no usar abreviaturas
  - nombres de subclases

Lógicas para la Descripción - OWL

Ontologías

DL estructurales

DL proposicionales

OWL

- las lógicas para la descripción son un formalismo lógico que unifica distintas herramientas de representación del conocimiento:
  - sistemas basados en marcos
  - redes semánticas
  - representaciones orientadas a objetos
  - modelos de datos semánticos
  - lenguajes de ontologías
- son un fragmento estructurado de la lógica de predicados
- proveen un lenguaje para expresar información estructurada y para razonar sobre la misma

- aplicaciones
  - modelado conceptual
  - optimización de consultas y mantenimiento de vistas (DB)
  - semántica de lenguaje natural
  - integración inteligente de información
  - ontologías y terminología
  - administración de proyectos de software
  - planificación

- la lógicas para la descripción formalizan varias representaciones orientadas a objetos
- por lo tanto, proveen mecanismos para desambiguar representaciones imprecisas
- formalizan
  - marcos u objetos
  - clases
  - slots o atributos
    - valores
    - tipos
    - restricciones

Ambigüedades

- clases e instancias
- clases vs información incompleta en instancias
- relación de subclase o de instancia
- relaciones implícitas o explícitas
- relaciones normales o relaciones especiales
- cuantificación
- razonamiento no monótono

Falsos amigos

- el significado de las representaciones orientadas a objetos e muy ambiguo
- es muy atractivo la naturaleza gráfica de la representación orientada a objetos
- a partir de esta representación, es fácil desarrollar algoritmos que ”simulan” razonar sobre la estructura, pero su comportamiento no tienen justificación

- la familia de las lógicas para la descripción son un fragmento de FOL (lógica de predicados)
- la representación es al nivel de predicados; no se usan variables en estos formalismos
- una teoría en una lógica para la descripción se divide en dos partes
  - la definición de predicados TBox
  - una aserción sobre constantes ABox
- cualquier lógica para la descripción es un subconjunto de L_3 (el fragmento de FOL que usa a lo sumo tres variables)

¿Porqué no FOL?

- si se usa FOL directamente sin restricciones adicionales entonces
  - se destruye la estructura del conocimiento, y no pude ser aprovechada para generar las inferencias
  - el poder expresivo es muy alto: no se pueden definir problemas decidibles y con inferencia eficiente
  - el poder expresivo es muy bajo: existen ciertos conceptos interesantes, decidibles que no pueden ser expresados en FOL

KL-ONE

- el antecedente inmediato de las lógicas para la descripción es el sistema KL-ONE para redes semánticas y marcos (1985)
- permitía la descripción estructurada de objetos, y razonaba sobre la misma
- corresponde a la descripción de una compleja estructura relacional, construida usando un conjunto de constructores epistemológicamente adecuados
- distinguía entre el conocimiento conceptual (terminología) y de las instancias (asercional)
- el algoritmo para determinar subsunción cumplía un rol fundamental en la inferencia automática
- sólo permitía razonamiento estricto, sin defaults

Elementos en el lenguaje de TBox

- conceptos denotan conjunto de entidades o clases; se representan mediante predicados unarios
- ejemplos
  - `Estudiante` con semántica $\{x \mid \texttt{alumno}(x)\}$
  - `Soltero` con semántica $\{x \mid \texttt{soltero}(x)\}$
- roles denotan propidades o relaciones; se representan mediante predicados binarios
- ejemplos
  - `Amigo` con semántica $\{(x,y) \mid \texttt{amigo}(x,y)\}$
  - `Ama` con semántica $\{(x,y) \mid \texttt{ama}(x,y)\}$

Expresiones de conceptos

- las lógicas para la descripción organizan la información en clases (conceptos) que agrupan datos homogéneos de acuerdo a una propiedad en común
- ejemplo: $\texttt{Estudiante} \sqcap \exists \texttt{Amigo}.\texttt{Soltero}$
- con significado $\{x \mid \texttt{alumno}(x) \wedge \exists y.\texttt{amigo}(x,y) \wedge \texttt{soltero}(y)\}$

Clases

- una clase en POO se representa directamente en una lógica para la descripción

$$\{x \mid \texttt{alumno}(x)\} = \{x \mid \texttt{persona}(x) \wedge \exists y.\texttt{nombre}(x,y) \wedge \texttt{cadena}(y) \wedge \exists y.\texttt{direccion}(x,y) \wedge \texttt{cadena}(y) \wedge \exists y.\texttt{cursa}(x,y) \wedge \texttt{carrera}(y)\}$$

$$\texttt{Estudiante} \doteq \texttt{Persona} \sqcap \exists \texttt{Nombre.Cadena} \sqcap \exists \texttt{Direccion.Cadena} \sqcap \exists \texttt{Cursa.Carrera}$$

Objetos

- `s1` identifica a un alumno

$$\texttt{alumno}(\texttt{s1}) \wedge \texttt{nombre}(\texttt{s1},'\texttt{juan}') \wedge \texttt{direccion}(\texttt{s1},'\texttt{patagones}') \wedge \texttt{cursa}(\texttt{s1},\texttt{contador}) \wedge \texttt{carrera}(\texttt{contador})$$

Redes semánticas

- definición de conceptos y relaciones

$$\begin{aligned} \forall x.\texttt{alumno}(x) &\rightarrow \exists y.\texttt{cursa}(x,y) \wedge \texttt{carrera}(y) \\ \forall x.\texttt{docente}(x) &\rightarrow \exists y.\texttt{ensena}(x,y) \wedge \texttt{carrera}(y) \\ \forall x.\texttt{ayudanteB}(x) &\rightarrow \texttt{alumno}(x) \wedge \texttt{docente}(x) \end{aligned}$$

$$\begin{aligned} \texttt{Estudiante} &\sqsubseteq \exists \texttt{Cursa.Carrera} \\ \texttt{Docente} &\sqsubseteq \exists \texttt{Ensena.Carrera} \\ \texttt{AyudanteB} &\sqsubseteq \texttt{Estudiante} \\ \texttt{AyudanteB} &\sqsubseteq \texttt{Docente} \end{aligned}$$

Cuantificadores

- todas las ranas son también verdes

$$\texttt{Rana} \sqsubseteq \exists \texttt{EsColor.Verde}$$

- todas las ranas son sólo verdes

$$\texttt{Rana} \sqsubseteq \forall \texttt{EsColor.Verde}$$

- hay una rana cuyo color es sólo verde

$$\bot \sqsubset \texttt{Rana} \sqcap \forall \texttt{EsColor.Verde}$$

FL menos

- presentaremos ahora la más simple de las lógicas para la descripción estructurales FL^-
- sintaxis

$$C, D \rightarrow A \mid C \sqcap D \mid \forall R.C \mid \exists R$$

siendo A un concepto atómico, R un rol atómico, C, D conceptos generales

- la semántica de FL^-
  - los conceptos representan clases, conjuntos de individuos
  - los roles representan relaciones entre pares de individuos
  - los conceptos atómicos son los nombres de los concepto primitivos (sin definición)
  - los constructores constituyen la forma de definir conceptos más complejos

- el constructor $\sqcap$ es la conjunción. Ejemplo: `Adulto`$\sqcap$`Varon`$\sqcap$`Persona` representa al conjunto de entidades que son al mismo tiempo adultas, de sexo masculino y personas
- en términos POO, esto nos dice que se permite la herencia múltiple
- el constructor $\forall R.C$ permite definir una restricción en los valores de atributos Ejemplo: $\forall$`Hijo.Doctor` representa el conjunto de elementos tales que todos sus hijos son doctores, que incluye a quienes no tienen hijos(!).

- el constructor $\exists R$ garantiza que existirá al menos un valor para el atributo R Ejemplo: `Persona`$\sqcap\exists$`Hijo` es el conjunto de personas que tienen al menos un hijo
- la semántica formal se define en base a interpretaciones de dominios (conjuntos de elementos)

Subsunción

- la subsunción $C \sqsubseteq D$, o sea C es *cubierto* por D, es el principal problema de razonamiento
- en términos de semántica, saber si es válido $\forall x C(x) \rightarrow D(x)$
- en FL^- la subsumción no sólo es decidible, sino que además está en **P**
- en FL^- existe un algoritmo estructural (basado en la sintaxis de las expresiones que definen conceptos) que resulve el problema de subsunción en tiempo $O(n^2)$ donde n es la longitud de las expresiones

$\mathscr{ALC}$

A	$A^I \subseteq \Delta$	concepto primitivo
R	$R^I \subseteq \Delta \times \Delta$	rol primitivo
$\top$	Δ	universo
$\bot$	$\emptyset$	concepto vacío
$\neg C$	$\Delta \setminus C^I$	negación
$C \sqcap D$	$C^I \cap D^I$	conjunción
$C \sqcup D$	$C^I \cup D^I$	disyunción
$\forall R.C$	$\{x \mid \forall y.R^I(x,y) \rightarrow C^I(y)\}$	cuant. universal
$\exists R.C$	$\{x \mid \exists y.R^I(x,y) \wedge C^I(y)\}$	cuant. existencial

- la conjunción se interpreta como la intersección del conjunto de individuos
- la disyunción se interpreta como la unión del conjunto de individuos
- la negación se interpreta como el complemento dle conjunto de individuos
- equivalencias
  - $\exists R.\top \Longleftrightarrow \exists R$
  - $\neg(C \sqcap D) \Longleftrightarrow \neg C \sqcup \neg D$
  - $\neg(C \sqcup D) \Longleftrightarrow \neg C \sqcap \neg D$
  - $\neg\forall R.C \Longleftrightarrow \exists R.\neg C$
  - $\neg\exists R.C \Longleftrightarrow \forall R.\neg C$

Ejemplos

- $\texttt{AyudanteB} \sqsubseteq \texttt{Alumno} \sqcup \neg\texttt{Graduado} \sqcup \texttt{Docente}$ condición necesaria
- $\texttt{AyudanteB} \doteq \texttt{Alumno} \sqcup \neg\texttt{Graduado} \sqcup \texttt{Docente}$ condición necesaria y suficiente
- en general, es posible tener expresiones complejas en los dos lados de la expresión
- es un lenguaje cerrado con sus constructores

Bases de conocimiento

- una base de conocimiento Σ es $\langle \mathtt{TBox}, \mathtt{ABox} \rangle$ donde $\mathtt{TBox}$ es un conjunto de definiciones de conceptos, y $\mathtt{ABox}$ es un conjunto de aserciones sobre instancias
- ejemplo de $\mathtt{TBox}$

$$\begin{aligned} \mathtt{Estudiante} &\doteq \mathtt{Persona} \sqcap \exists \mathtt{Nombre.Cadena} \\ &\quad \sqcap \exists \mathtt{Direccion.Cadena} \\ &\quad \sqcap \exists \mathtt{Estudia.Carrera} \\ \exists \mathtt{Ensena.Carrera} &\sqsubseteq \neg \mathtt{Graduado} \sqcup \mathtt{Docente} \end{aligned}$$

- ejemplo de $\mathtt{ABox}$

$$\mathtt{Estudiante}(\mathtt{juan}), \mathtt{Estudia}(\mathtt{juan}, \mathtt{contador}), \mathtt{Carrera}(\mathtt{contador})$$

Servicios de razonamiento

- satisfacibilidad de conceptos $\Sigma \not\models C \equiv \bot$, $\mathtt{Estudiante} \sqcap \neg \mathtt{Persona}$
- subsunción $\Sigma \models C \sqsubseteq D$, $\mathtt{Estudiante} \sqsubseteq \mathtt{Persona}$
- satisfacibilidad $\Sigma \not\models \bot$, $\mathtt{Estudiante} \doteq \neg \mathtt{Persona}$
- control de instancia $\Sigma \models C(a)$, $\mathtt{Estudiante}(\mathtt{juan})$
- recuperación $\{a \mid \Sigma \models C(a)\}$
- realización $\{C \mid \Sigma \models C(a)\}$

- todos estos servicios se pueden reducir a satisfacibilidad
- existen algoritmos para decidir satisfacibilidad en $\mathscr{ALC}$, y por lo tanto todos los demás tipos de razonamiento

Complejidad de razonamiento

expresividad	$\Sigma \models C \sqsubseteq D$	$\Sigma \models C(a)$
FL^-	P	P
$\neg A$, $\mathscr{AL}$	P	P
$\exists R.C$, $\mathscr{ALE}$	NP	PSPACE
$\neg C$, $\mathscr{ALC}$	PSPACE	PSPACE
$\{a_1, \ldots\}$, $\mathscr{ALCO}$	PSPACE	
$\mathscr{SHIQ}$	EXPTIME	

Algunas extensiones de $\mathscr{ALC}$

constructor	sintaxis	semántica
concepto atómico	A	$A^I \subseteq \Delta$
universo	$\top$	Δ
vacío	$\bot$	$\emptyset$
conjunción	$C \sqcap D$	$C^I \cap D^I$
disyunción ($\mathscr{U}$)	$C \sqcup D$	$C^I \cup D^I$
negación ($\mathscr{C}$)	$\neg C$	$\Delta \setminus C^I$
universal	$\forall R.C$	$\{x \mid \forall y.R(x,y) \rightarrow C(y)\}$
existencial ($\mathscr{E}$)	$\exists R.C$	$\{x \mid \exists y.R(x,y) \wedge C(y)\}$
cardinalidad ($\mathscr{N}$)	$\geq_n R$ $\leq_n R$	$\{x \mid \sharp\{y \mid R(x,y)\} \geq n\}$ $\{x \mid \sharp\{y \mid R(x,y)\} \leq n\}$
card. calificada ($\mathscr{Q}$)	$\geq_n R.C$ $\leq_n R.C$	$\{x \mid \sharp\{y \mid R(x,y) \wedge C(y)\} \geq n\}$ $\{x \mid \sharp\{y \mid R(x,y) \wedge C(y)\} \leq n\}$
enumeración	$\{a_1, \ldots, a_n\}$	$\{a_1^I, \ldots, a_n^I\}$

- RDF(S) tiene severas limitaciones expresivas
  - sólo relaciones binarias
  - no tiene características para propiedades (inversa, transitiva, simétrica)
  - no permite la definición de conceptos complejos
  - no permite restricciones de cardinalidad
  - no permite disyunción

- OWL extiende a RDF Schema para convertirse en un lenguaje para representación de conocimiento completo para la web
- es una recomendación de la W3C desde 2004, modificada en 2009 y 2012, donde se introdujo OWL 2
- OWL 2 está basado en $\mathscr{SROIQ}$, es decir $\mathscr{ALC}$ (conjunción, disyunción, negación) extendido con:
  - roles transitivos e inversos
  - nominales en RBox
  - restricciones de cardinalidad calificadas
- OWL, basado en $\mathscr{SHOIN}$, solo permitía jerarquías simples de roles y restricciones de cardinalidad sin calificar

- objetivos de diseño para OWL
  - compartible
  - modificable en el tiempo
  - interoperable
  - con detección de inconsistencias
  - balance entre expresividad y complejidad
  - facilidad de uso
  - compatible con estándares existentes (XML, RDF)

- requerimientos
  - las ontologías son objetos en la web
  - es decir, deben contener información sobre sus metadatos (versiones, autores, etc)
  - definen clases, propiedades, tipos de datos, rango/dominio, individuos
  - (des)igualdad para clases e individuos
  - restricciones de cardinalidad

- no se tuvieron en cuenta
  - valores por defecto
  - razonamiento mundo cerrado
  - combinación de propiedades
  - aritmética y operaciones sobre cadenas
  - importación parcial
  - definición de vistas
  - extensiones con reglas y procedurales

Variantes de OWL: OWL EL

- OWL 2 EL basado en $\mathscr{EL}++$ está particularmente orientado a
  - aplicciones de ontologías con un gran número de clases y propiedades
  - captura el poder expresivo de muchas ontologías reales
  - tiene consistencia, subsunción y chequeo de instancias en **P**

Variantes de OWL: OWL QL

- OWL 2 QL basado en $\mathrm{DL-Lite}_R$ está orientado a
  - solución completa y sana de consultas en tiempo **LOGSPACE** con respecto al tamaño de los datos
  - posibilidad de expresión de modelos conceptuales en UML y ER
  - el ABox puede almacenarse en un RDBMS y consultado mediante re-escritura de consultas a SQL

Variantes de OWL: OWL RL

- OWL 2 RL está basado en $\mathscr{DLP}$ programación en lógica descriptiva y está orientado a
  - aplicaciones con razonamiento escalable sin mucho sacrificio de poder expresivo
  - basado en sistemas de reglas

Constructores OWL

constructor	DL	ejemplo
`intersectionOf`	$C_1 \sqcap \ldots \sqcap C_n$	`Persona`$\sqcap$`Varon`
`unionOf`	$C_1 \sqcup \ldots \sqcup C_n$	`Alumno`$\sqcup$`Docente`
`complementOf`	$\neg C$	$\neg$`Varon`
`oneOf`	$\{a_1, \ldots, a_n\}$	$\{rojo, verde\}$
`allValuesFrom`	$\forall R.C$	$\forall$`Hijo.Doctor`
`someValuesFrom`	$\exists R.C$	$\exists$`Hijo.Abogado`
`value`	$\exists P.\{o\}$	$\exists$`CiudadanoDe.USA`
`minCardinality`	$\leq_n P.C$	$\leq_n$ `Hijo.Abogado`
`maxCardinality`	$\geq_n P.C$	$\geq_n$ `Hijo.Varon`

+

tipos de datos XML Schema: int, string, real, etc...

Axiomas OWL

- `SubClassOf`
- `EquivalentClasses`
- `SubPropertyOf`
- `EquivalentProperties`
- `SameIndividual`
- `DisjointClasses`
- `DifferentIndividuals`
- `InverseOf`
- `Transitive`
- `Symmetric`

Sintaxis de OWL

- RDF es la sintaxis oficial
  - difícil de leer para humanos
  - difícil de implementar para los parsers
- sintaxis basada en UML
  - posiblemente ambigua
  - gran comunidad de usuarios
- sintaxis XML
  - no es lo mismo que RDF
  - más legible
- sintaxis abstracta, más orientada a humanos