Modelos Computacionales
Actividad 3

José Alberto Benítez Andrades

71454586A

Modelos Computacionales
Máster en Lenguajes y Sistemas Informáticos - Tecnologías del Lenguaje en la Web
UNED
15/02/2011

0. Enunciado

Tema 3.- La web semántica (Web 3.0)

3.1 Estándares

3.2 Ontologías y su población

3.3 Modelos para recuperación de información

o El Tema 3 incluye los conceptos básicos y algunos de los estándares de la web semántica. Además se estudiarán recursos existentes y aspectos candentes como la población de ontologías. Finalmente se estudiarán algunos modelos de recuperación de información basada en ontologías.
o Bibliografía básica: (Verdejo, García-Serrano, 2009) Este libro contiene capítulos independientes sobre temas de actualidad. En concreto para este tema del curso han de leerse los capítulos: 2 y 4.

Ejercicio T3.1: Visitar la dirección http://wordnet.princeton.edu/ y resumir los aspectos mas relevantes del recurso WordNet (WN) (en opinión del estudiante).

Ejercicio T3.2: Describir un recurso léxico semántico y multilingüe de los que aparecen en el capítulo 2: qué tipo de información semántica incorpora, cómo la organiza, ventajas y desventajas de su uso potencial en aplicaciones.

Ejercicio T3.3:

3.3.1 Definición de rol semántico.

3.3.2 Describir brevemente una arquitectura de tres capas potencialmente adecuada para la identificación de roles semánticos (apartado tercero del capítulo 4).

Ejercicio T3.4:

3.4.1 Describir detalladamente lo que es el OWL lite (por ejemplo en http://www.w3.org/TR/owl-features/)

3.4.2 Encontrar en algún repositorio de ontologías, una ontología de dominio. Puede visitar:

http://olp.dfki.de/ontoselect/

http://www.gnoss.com/comunidad/Interoperabilidadsemantica/recurso/Finding-Ontologies-some-help-to-find-existing-ont/a47e9be8-0ab0-4a71-bd61-b22a2c8b3

1. Resolución.

Ejercicio T3.1: Visitar la dirección http://wordnet.princeton.edu/ y resumir los aspectos mas relevantes del recurso WordNet (WN) (en opinión del estudiante).

WordNet es una base de datos léxica estructurada a partir de las principales relaciones conceptuales que vinculan entre sí a los lexemas intra- y trans-categorialmente; algunas relaciones, como la hiponimia, tienen como consecuencia la formación de un sistema jerárquico en el que cada lexema ocupa una posición que le es propia y que además tiene consecuencias directas en la atribución de sentido

En WordNet, el léxico está dividido en cinco categorías gramaticales: sustantivos, verbos, adjetivos, adverbios y palabras funcionales; sin embargo, por el momento, sólo están incluidos los sustantivos, los verbos, los adjetivos y los adverbios.

WordNet trata de dar respuesta práctica a algunas de las preguntas planteadas por la semántica léxica, la cual postula que debe existir una asociación convencional entre un concepto lexicalizado y una expresión que cumple una función sintáctica. Dichos planteamientos se refieren al tipo de expresiones que participan en esas asociaciones, a cuál es la naturaleza y organización de los conceptos lexicalizados que las palabras pueden expresar y al tipo de funciones sintácticas que las palabras tienen.

Una forma de concebir la base de datos de WordNet es a través de una matriz léxica en la que se asocian las formas léxicas con los sentidos.

WordNet, constituye una base de datos léxico relacional que simula, parcialmente, la memoria humana. Es muy tentador referirse a las posibles aplicaciones de este instrumento. Entre las más obvias, tenemos la creación de diccionarios electrónicos multilingües, lo que sin duda sería una herramienta de primera mano en el aprendizaje de lenguas extranjeras. Sin embargo, no hay que dejar de lado que también es posible explotar el conjunto de relaciones dentro de la base datos misma y luego generalizarlas a otros idiomas.

Las aplicaciones en este caso, serían múltiples y sin duda aún no hemos alcanzado a imaginar todo lo que sería posible con WordNet. Entre las primeras posibilidades que se han previsto, se encuentra la desambiguación automática de textos, como una herramienta que posee información semántica pertinente sobre los sintagmas. Otras opciones no deben dejarse de lado, como por ejemplo, el hecho de que WordNet ya ofrece una taxonomía del conocimiento, la cual puede utilizarse como referencia para labores de indexación, por ejemplo. Tampoco hay que olvidar las posibles aplicaciones en la educación y el aprendizaje de lenguas extranjeras.

Ejercicio T3.2: Describir un recurso léxico semántico y multilingüe de los que aparecen en el capítulo 2: qué tipo de información semántica incorpora, cómo la organiza, ventajas y desventajas de su uso potencial en aplicaciones.

En general, la desambiguación del sentido de las palabras es el problema de seleccionar un sentido de un conjunto de posibilidades predefinidas para una palabra dada en un texto o discurso.

En los últimos años se han incrementado las investigaciones para crear métodos de WSD. A continuación se describe la clasificación para métodos de WSD de acuerdo a los recursos que utilizan (Figura 1).

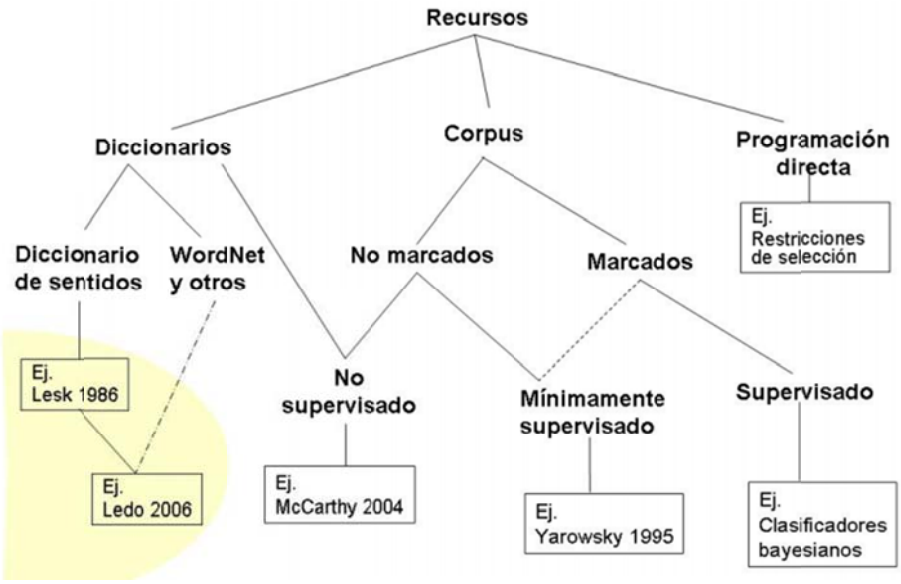

Figura 1. Clasificación de los métodos para WSD de acuerdo a los recursos que utilizan.

Métodos para WSD

Los métodos para desambiguación de sentidos de palabras se clasifican en: los que utilizan diccionarios, los que utilizan corpus, y los que no utilizan ningún recurso léxico.

- Los que utilizan diccionarios:

Los diccionarios pueden ser de sentidos y otros como WordNet.

Los diccionarios proporcionan una lista de glosas (definición de sentido) para las palabras. Los métodos que utilizan sólo diccionarios de sentidos, buscan elegir un sentido (de esta lista) para cada palabra en un texto dado, tomando en cuenta el contexto en el que aparece. Como ejemplo, Lesk (1986) propone utilizar la coherencia global del texto, es decir, el total de sentidos de palabras relacionadas en el texto: mientras más relacionadas estén las palabras entre sí, más coherente será el texto.

Además existen variantes del algoritmo de Lesk que utilizan no sólo diccionarios de sentidos, sino también otro tipo de diccionarios como WordNet.

- Los que utilizan corpus:

Los corpus pueden ser no marcados y marcados.

Los métodos que utilizan corpus no marcados son los no supervisados, estos métodos también utilizan otros recursos como WordNet para poder asignar un sentido a cada palabra que aparece en los textos no marcados. Como ejemplo de éstos tenemos el método de McCarthy et al. (2004), el cual elige de un diccionario (tesauro) las palabras relacionadas con la palabra a desambiguar. Cada palabra relacionada tiene un peso, éstas y la palabra a desambiguar tienen sentidos en un diccionario. Para elegir el sentido correcto, las palabras relacionadas votan por un sentido de la palabra a desambiguar con cierto peso. Se elige el sentido con más peso.

Los métodos que utilizan corpus marcados son los métodos supervisados. Éstos reducen la desambiguación de sentidos de palabras a un problema de clasificación, donde a una palabra dada se le asigna el sentido más apropiado de acuerdo a un conjunto de posibilidades, basadas en el contexto en el que ocurre. Hay muchos algoritmos de aprendizaje supervisado utilizados para WSD, como ejemplo tenemos los clasificadores bayesianos, máquinas de soporte vectorial, árboles y listas de decisión, etc.

Hay métodos que utilizan una gran cantidad de corpus no marcados y muy pocos marcados llamados mínimamente supervisados. Como ejemplo de éstos tenemos el método de Yarowsky (1995), el cual identifica todas las ocurrencias de una palabra a desambiguar en un corpus no marcado. Después identifica un número pequeño de colocaciones semilla representativos de cada sentido de la palabra y etiqueta todos los ejemplos que contienen la colocación semilla con la palabra de dicha colocación (así tenemos los conjuntos etiquetados con cada sentido representativo y el conjunto residuo).

El algoritmo utiliza los conjuntos etiquetados para entrenar una lista de decisión y encontrar nuevas colocaciones, para después etiquetar sobre el conjunto residuo. El algoritmo termina cuando el conjunto residuo se estabiliza

- Los que utilizan **programación directa:**

  Estos métodos se basan en reglas (muchas) que especifican el sentido de una palabra de acuerdo al contexto en el que aparece. Un ejemplo son las restricciones de selección (selectional restrictions), definen reglas de acuerdo a la palabra a desambiguar y su argumento. Ejemplo: el verbo comer puede tener como restricción que su tema argumento sea comida (comer-comida).

Algoritmo de Lesk

El algoritmo de Lesk (1986) es uno de los primeros algoritmos exitosos usados en la desambiguación de sentidos de palabras. Este algoritmo esta determinado por dos principales ideas: un algoritmo de optimización para WSD y una medida de similitud medida para las definiciones de los sentidos.

El primero es acerca de desambiguar palabras, considerando la optimización global del texto, esto es, encontrar la combinación de los sentidos que maximice la relación total entre los sentidos de todas las palabras.

Por ejemplo, para la oración *My father deposits his money in a bank account* y considerando a lo más tres sentidos, para cada palabra, la figura 2 muestra la representación del algoritmo de Lesk original

Tabla 2. Sentidos de las palabras (máximo tres) obtenidas de WordNet para la oración "My father deposits his money in a bank account"

My father deposits his money in a bank account

Father

1: a male parent (also used as a term of address to your father); "his father was born in

Atlanta".

2: `Father' is a term of address for priests in some churches (especially the Roman

Catholic Church or the Orthodox Catholic Church); "`Padre' is frequently used in the ilitary".

3: a person who holds an important or distinguished position in some organization; "the tennis fathers ruled in her favor"; "the city fathers endorsed the proposal".

Deposit

1: fix, force, or implant; "lodge a bullet in the table".

2: put into a bank account; "She deposits her paycheck every month".

3: put (something somewhere) firmly; "She posited her hand on his shoulder"; "deposit the suitcase on the bench"; "fix your eyes on this spot".

Money

1: the official currency issued by a government or national bank; "he changed his money into francs".

bank

1: a financial institution that accepts deposits and channels the money into lending activities; "he cashed a check at the bank"; "that bank holds the mortgage on my home".

2: sloping land (especially the slope beside a body of water); "they pulled the canoe up on the bank"; "he sat on the bank of the river and watched the currents".

3: a supply or stock held in reserve for future use (especially in emergencies)

account

1: a formal contractual relationship established to provide for regular banking or

brokerage or business services; "he asked to see the executive who handled his account".

2: the act of informing by verbal report; "he heard reports that they were causing trouble"; "by all accounts they were a happy couple".

3: a record or narrative description of past events; "a history of France"; "he gave an inaccurate account of the plot to kill the president"; "the story of exposure to lead"

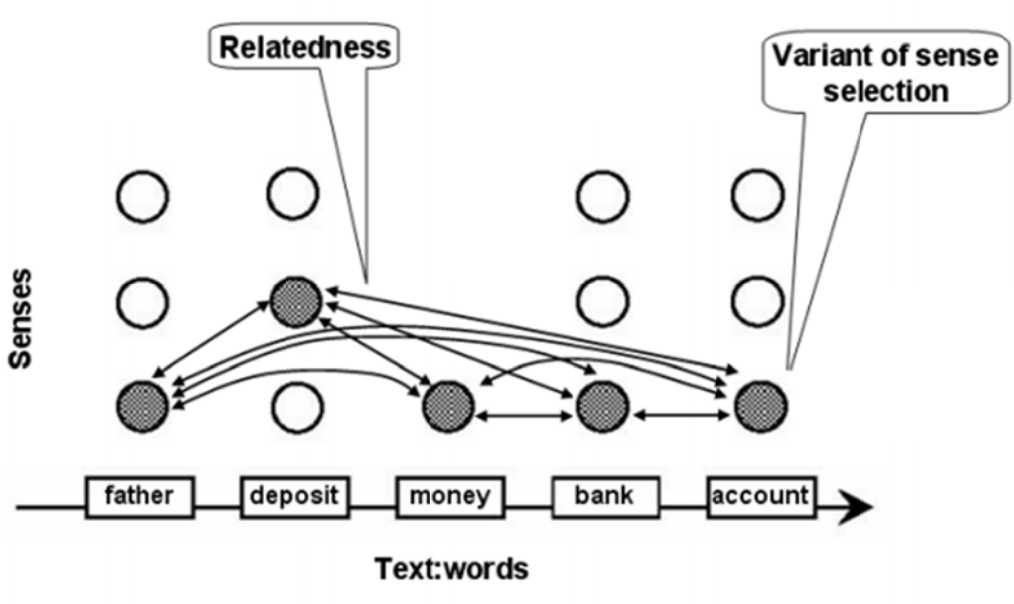

Figura 2. Representación gráfica del algoritmo original de Lesk.

En el segundo punto, relacionado con la medida de similitud, Lesk sugiere usar el traslape entre las definiciones de los sentidos, es decir, contar el número de palabras que tienen en común.

Como ejemplo, para la oración, "My father deposite his mony in the bank account" para medir la relación de las definiciones de los sentidos para la palabra "deposit" y "Bank" como Lesk lo propuso, es necesario contar las palabras en común en todas las definiciones. En este caso, comparando principalmente las tres definiciones de "deposit" contra las tres definiciones de "bank". La relación entre los valores se muestra en la siguiente tabla:

Tabla 3. Valores de relación para las definiciones de sentidos de las palabras "deposti" y "bank".

Número de sentidos para *deposit*	Número de sentidos para *bank*	Valor de relación
1	1	0
1	2	0
1	3	0
2	**1**	**2**
2	2	1
2	3	0
3	1	1
3	2	0
3	3	0

La figura 3 muestra los pasos principales para el método original de Lesk. En el pseudocodigo está resaltado en itálica y negrita donde la función corresponde a la medida de similitud propuesta por Lesk. Todo el pseudocódigo mostrado se refiere al algoritmo de optimización para WSD.

```
for each vector v = (s1, ... sn) where si in vi
    sumv = 0;
    for each i
            for each j
                    sumv += relatedness-overlap(si, sj)
Chose the vector v with the biggest value of sumv
for each word w
    sense(w) = v[w]
```

Figura 3. Pseudocódigo del método de Lesk original

Por un lado, la limitación principal de la medida de similitud propuesta por Lesk, es que las glosas del diccionario, regularmente, son muy cortas y no incluyen el vocabulario suficiente para identificar los sentidos relacionados (Patwardhan et. al., 2003). Esta es la razón de porqué diferentes autores han propuesto distintas medidas de similitud (Resnik, 1995; Jiang and Conrath, 1997; Hirst and St-Onge, 1998; Leacock and Chodorow, 1998; Lin, 1998) (Véase sección 2.8.2).

Como se mencionó con anterioridad, el algoritmo original de Lesk es muy caro computacionalmente hablando, debido a que mientras más palabras tenga el texto, y más sentidos por cada palabra, mayor será el número de combinaciones de sentidos, el cual se incrementa de manera exponencial, haciéndolo prácticamente prohibitivo para una búsqueda exhaustiva que garantice encontrar el óptimo global exacto. Por ejemplo, para una oración de 16 palabras de contenido, donde cada palabra contiene siete sentidos (números cercanos a los observados en el corpus de SemCor).

Para aliviar el problema de la complejidad computacional del algoritmo original de Lesk, dos principales modificaciones al algoritmo han sido propuestas, incluyendo A) una versión simplificada que considere las palabras, una por una, y compare cada sentido de la palabra dada con el contexto, y b) el uso de búsquedas basadas en heurísticas para encontrar una solución óptima cercana a la real.

Ejercicio T3.3:

3.3.1 Definición de rol semántico.

Un rol semántico es la relación entre un constituyente sintáctico (generalmente, aunque no siempre, argumento del verbo) y un predicado (generalmente, aunque no siempre, un verbo). Ejemplos de roles semánticos son agente, paciente, beneficiario, etc. o también adjuntos como causa, manera, lugar, etc.

Por ejemplo, la oración (1), tiene cinco constituyentes cada uno de ellos con un rol semántico diferente. El constituyente sintáctico \Mary" tiene el rol agente, y los constituyentes, “John" y “with a baseball " tienen los roles paciente e instrumento, respectivamente. Además, los constituyentes “in the park " y “yesterday" tienen los roles lugar y tiempo, respectivamente.

3.3.2 Describir brevemente una arquitectura de tres capas potencialmente adecuada para la identificación de roles semánticos (apartado tercero del capítulo 4).

La arquitectura más habitual para el SRL dispone de tres capas. En la primera etapa (filtrado o poda) se selecciona, partiendo de un árbol sintáctico, el conjunto de candidatos a argumento para un predicado dado. Se eliminan aquellos constituyentes que tienen poca probabilidad de ser argumentos, así se consigue esta selección.

En una segunda etapa, se produce una *valoración local* de los candidatos a argumento mediante el uso de una función de puntuación, que toma un candidato como entrada y asigna una probabilidad (también llamada grado de confianza) para cada etiqueta semántica posible más una etiqueta adicional “no-argumento” que indicaría que ese candidato no debe considerarse argumento en la solución final. Los candidatos se tratan de forma independiente e igualmente el *reconocimiento* y la *clasificación* de los argumentos pueden ser tratados separadamente o de forma conjunta en esta etapa. En el primero de los casos, se aplican las dos funciones de forma encadenada, primero puntuando entre *argumento* y *no-argumento*, y después puntuando cada etiqueta semántica para cada argumento reconocido por la primera función. Los sistemas del estado-del-arte se dividen en el 50% entre las dos opciones posibles, aproximadamente.

Para el aprendizaje de las funciones de valoración local, se han utilizado algoritmos discriminantes de aprendizaje automático. Entre los que encontramos: árboles de decisión, modelos log-linear, Support Vector Machines, Snow, AdaBoost, y aprendizaje basado en ejemplos. Todos ellos son muy distintos, pero tienen un mismo propósito general: que la elección del algoritmo concreto no tiene un impacto demasiado importante en los resultados finales de los sistemas. Todos los algoritmos de aprendizaje necesitan que los candidatos argumentos se conviertan en un conjunto de atributos y valores determinados. Esto es muy importante para los resultados en la tarea.

En tercer lugar, es posible aplicar una fase de *puntuación conjunta* (o puntuación global) para combinar las predicciones de los clasificadores locales y producir la mejor solución para la estructura argumental completa del predicado. En esta etapa, se pueden tener en cuenta las dependencias entre varios argumentos del mismo predicado para hacer la

valoración global de la solución. Algunas aproximaciones se producen mediante Programación Lineal Entera (ILP) y otras mediante la reordenación.

De manera eventual, se puede producir una cuarta etapa que se encarga de corregir errores frecuentes o de asegurar la consistencia de la solución final. Consiste en un conjunto de reglas heurísticas realizadas a mano. Poseen una dependencia con respecto a la arquitectura del sistema SRL.

No todos los sistemas SRL ejecutan todas las fases de la arquitectura de tres capas que se ha descrito anteriormente. Los sistemas más simples producen la anotación final a base de una sola etapa de valoración local. Otros prescinden de la etapa local y trabajan directamente en la etapa de puntuación conjunta.

Ejercicio T3.4:

3.4.1 Describir detalladamente lo que es el OWL lite (por ejemplo en http://www.w3.org/TR/owl-features/)

El Lenguaje de Ontologías Web (OWL) está diseñado para ser usado en aplicaciones que necesitan procesar el contenido de la información en lugar de únicamente representar información para los humanos. OWL facilita un mejor mecanismo de interpretabilidad de contenido Web que los mecanismos admitidos por XML, RDF, y esquema RDF (RDF-S) proporcionando vocabulario adicional junto con una semántica formal. OWL tiene tres sublenguajes, con un nivel de expresividad creciente: OWL Lite, OWL DL, y OWL Full.

OWL Lite utiliza únicamente algunas de las características del lenguaje OWL y está más limitado en el uso de características que OWL DL y OWL Full. Por ejemplo, en OWL Lite, las clases sólo pueden ser definidas en términos de superclases definidas (las superclases no pueden ser expresiones arbitrarias), y sólo pueden ser usados ciertos tipos de restricciones de clase. Además, únicamente se permite la equivalencia entre clases y relaciones de subclases entre clases cuando se trata de clases definidas, no en el caso de expresiones de clases arbitrarias. Igualmente, las restricciones en OWL Lite usan sólo clases definidas. OWL Lite tiene además una noción limitada de cardinalidad - las únicas cardinalidades que se pueden definir explícitamente son 0 ó 1.

Características del esquema RDF en OWL Lite

Se incluyen las siguientes características de OWL Lite relacionadas con el esquema RDF.

- Class: Una clase define un grupo de individuos que pertenecen a la misma porque comparten algunas propiedades. Por ejemplo, Deborah y Frank son miembros de la clase Persona. Las clases pueden organizarse en una jerarquía de especialización usando subClassOf. Se puede encontrar una clase general llamada Thing que es una clase de todos los individuos y es una superclase de todas las clases de OWL. También se puede encontrar una clase general llamada Nothing

que es la clase que no tiene instancias y es una subclase de todas las clases de OWL.

- rdfs:subClassOf: Las jerarquías de clase deben crearse haciendo una o más indicaciones de que una clase es subclase de otra. Por ejemplo, la clase Persona podría estar definida como subclase de la clase Mamífero. De esto podemos deducir que si un individuo es una Persona, entonces, también es un Mamífero.

- rdf:Property: las propiedades pueden utilizarse para establecer relaciones entre individuos o de individuos a valores de datos. Ejemplos de propiedades son tieneHijo, tieneFamiliar, tieneHermano, y tieneEdad. Los tres primeros pueden utilizarse para relacionar una instancia de la clase Persona con otra instancia de la clase Persona (siendo casos de ObjectProperty), y el último (tieneEdad) puede ser usado para relacionar una instancia de la clase Persona con una instancia del tipo de datos Entero (siendo un caso de DatatypeProperty). Ambas, owl:ObjectProperty y owl:DatatypeProperty, son subclases de la clase de RDF rdf:Property.

- rdfs:subPropertyOf: Las jerarquías de propiedades pueden crearse haciendo una o más indicaciones de que una propiedad es a su vez subpropiedad de una o más propiedades. Por ejemplo, tieneHermano puede ser una subpropiedad de tieneFamiliar. De esta forma, un razonador puede deducir que si un individuo está relacionado con otro por la propiedad tieneHermano, entonces está también relacionado con ese otro por la propiedad tieneFamiliar.

- rdfs:domain: Un dominio de propiedad reduce los individuos a los que puede aplicarse la propiedad. Si una propiedad relaciona un individuo con otro individuo, y la propiedad tiene una clase como uno de sus dominios, entonces el individuo debe pertenecer a esa clase. Por ejemplo, puede establecerse que la propiedad tieneHijo tenga como dominio la clase Mamifero. De esto, un razonador puede deducir que si "Frank tieneHijo Anna", entonces Frank debe ser un Mamífero. Obsérvese que rdfs:domain se denomina restricción global debido a que la restricción se refiere a la propiedad y no sólo a la propiedad cuando está asociada con una clase en concreto. Consulte al final de este documento la discusión sobre restricciones de las propiedades para obtener más información.

- rdfs:range: El rango de una propiedad reduce los individuos que una propiedad puede tener como su valor. Si una propiedad relaciona a un individuo con otro individuo, y ésta como rango a una clase, entonces el otro individuo debe pertenecer a dicha clase. Por ejemplo, la propiedad tieneHija debe establecerse que tiene como rango la clase Mamífero. A partir de aquí, un razonador puede deducir que si Louise está relacionada con Deborah mediante la propiedad tieneHija, (por ejemplo, Deborah es hija de Louise), entonces Deborah es un Mamifero. El rango es, al igual que el dominio, una restricción global. Para obtener más información, consulte la discusión sobre las restricciones locales (por ejemplo, AllValuesFrom).

- Individual : Los individuos son instancias de clases, y las propiedades pueden ser usadas para relacionar un individuo con otro. Por ejemplo, un individuo llamado Deborah puede ser descrito como una instancia de la clase Persona y la propiedad tieneEmpleador puede ser usada para relacionar el individuo Deborah con el individuo UniversidadDeStanford.

Igualdad y desigualdad de OWL Lite

Las siguientes características de OWL Lite están relacionadas con los valores de igualdad y desigualdad.

- equivalentClass : Es posible definir dos clases como equivalentes. Las clases equivalentes tienen las mismas instancias. El valor de igualdad puede ser utilizado para crear clases sinónimas. Por ejemplo, Coche puede definirse como equivalentClass (clase equivalente) de Automóvil. De esta manera, un razonador puede deducir que cualquier individuo que sea instancia de Coche, es también una instancia de Automóvil y viceversa.

- equivalentProperty: Es posible definir dos propiedades como equivalentes. Las propiedades equivalentes relacionan a un individuo con el conjunto de otros individuos similares. Por ejemplo, tieneLíder debe definirse como equivalentProperty (propiedad equivalente) de tienePresidente. De este modo, un razonador puede deducir que si X está relacionado con Y por la propiedad tieneLíder, X está también relacionado con Y por la propiedad tienePresidente y viceversa. Un razonador también puede deducir que tieneLíder es una subpropiedad de tienePresidente, y tienePresidente es un subpropiedad de tieneLíder.

- sameAs: Es posible definir dos indivíduos como iguales. Estas construcciones pueden utilizarse para crear un número de nombres diferentes que se refieren al mismo individuo. Por ejemplo, el individuo Deborah puede establecerse como un individuo igual a DeborahMcGuiness.

- differentFrom: Es posible definir un individuo como diferente de otros individuos. Por ejemplo, el individuo Frank puede establecerse como distinto de los individuos Deborah y Jim. Por tanto, si los dos individuos, Frank y Deborah, constituyen valores para una propiedad que se ha establecido como funcional (de este modo la propiedad tiene como máximo un único valor), entonces hay una contradicción. Establecer explícitamente que los individuos son diferentes entre sí puede ser importante al utilizar lenguajes como OWL (y RDF) que no asumen que los individuos tienen un único nombre exclusivamente. Por ejemplo, sin información adicional, un razonador no puede deducir que Frank y Deborah hacen referencia a individuos distintos.

- AllDifferent: Se puede definir un número de individuos como mutuamente distintos mediante una indicación AllDifferent. Por ejemplo, se podría establacer que Frank, Deborah y Jim son mutuamente distintos usando la construcción AllDifferent. Al contrario que la indicación differentFrom anterior, ésto además

resaltaría que Jim y Deborah son distintos (no únicamente que Frank es distinto de Deborah y Frank es distinto de Jim). La construcción AllDifferent es particularmente útil cuando hay conjuntos de objetos distintos, y cuando los diseñadores están interesados en reforzar la suposición de nombres únicos dentro de estos conjuntos de objetos. Se usa junto con distinctMembers para establecer que todos los miembros de una lista son distintos, y disjuntos en pares.

Características de las propiedades en OWL Lite

Hay identificadores especiales en OWL Lite que se utilizan para proporcionar la información referente a las propiedades y a sus valores. La distinción entre ObjectProperty y DatatypeProperty se menciona anteriormente en la descripción de propiedad.

- inverseOf: Es posible definir una propiedad como la inversa de otra propiedad. Si se estableciera la propiedad P1 como inversa de la propiedad P2, y relacionáramos X con Y mediante la propiedad P2, entonces Y estaría relacionado con X mediante la propiedad P1. Por ejemplo, si la propiedad tieneHija es la propiedad opuesta de tienePadres y Deborah tienePadres Louise, entonces un razonador puede deducir que Louise tieneHija Deborah.

- TransitiveProperty: Es posible definir propiedades como transitivas. Cuando una propiedad es transitiva, si el par (x, y) es una instancia de la propiedad transitiva P, y el par (y, z) es otra instancia de la propiedad transitiva P, entonces el par (x, z) también es una instancia de P. Por ejemplo, si se indica que la propiedad Antepasado es transitiva, si Sara es un antepasado de Louise (es decir, (Sara, Louise) es una instancia de la propiedad Antepasado) y si Louise es un antepasado de Deborah (es decir, (Louise, Deborah) es una instancia de la propiedad Antepasado), entonces un razonador puede deducir que Sara es un antepasado de Deborah (es decir, (Sara, Deborah) es una instancia de la propiedad Antepasado).OWL Lite y OWL DL imponen la condición paralela de que las propiedades transitivas (y sus súperpropiedades) no pueden tener una restricción de cardinalidad máxima (maxCardinality) con valor 1. Sin esta condición paralela, OWL Lite y OWL DL se convertirían en lenguajes no procesables. Consulte la sección de axiomas de propiedades del documento OWL Semántica y sintaxis abstracta para obtener más información.

- SymmetricProperty: Es posible definir propiedades como simétricas. Si una propiedad es simétrica, y el par (x, y) es una instancia de esa propiedad simétrica P, entonces el par (y, x) es también una ionstancia de la propiedad simétrica P. Por ejemplo, Amigo puede considerarse una propiedad simétrica. De esta forma, si se indica a un razonador que Frank es amigo de Deborah, éste puede deducir que Deborah es amiga de Frank.

- FunctionalProperty: Es posible definir propiedades para que tengan un valor único. Si una propiedad es FunctionalProperty, ésta no tendrá más de un valor para cada individuo (es posible que no tenga un valor para un individuo). Esta

característica se denomina propiedad única. FunctionalProperty es una forma abreviada para indicar que la cardinalidad mínima de la propiedad es 0 y la cardinalidad máxima es 1. Por ejemplo, tieneJefeSuperior puede establecerse como FunctionalProperty. Es por ello que un razonador puede deducir que ningún individuo pueda tener más de un jefe principal. Sin embargo, esto no implica que todas las instancias de Persona deban tener al menos un jefe principal.

- InverseFunctionalProperty: Es posible definir propiedades para que sean funcional inversa. Si una propiedad es funcional inversa, entonces la inversa de la propiedad será funcional. Por tanto, la inversa de la propiedad tiene como máximo de un valor para cada individuo. Esta característica también se denomina propiedad inequívoca. Por ejemplo, tieneNúmerodeSeguridadSocialdeEE.UU (un identificador único para los residentes en los Estados Unidos) puede establecerse como funcional inversa (o inequívoca). La inversa de este propiedad (al que se puede llamar esElNúmeroDeSeguridadSocialPara) tiene como máximo un valor para cada individuo dentro de la clase de los números de Seguridad Social. De esta manera, el número de Seguridad Social de cualquier una persona es el único valor para su propiedad esElNúmeroDeSeguridadSocialPara. Es por esto que un razonador puede deducir que no existen dos individuos diferentes, instancias de Persona, que tengan idéntico el número de Seguridad Social de los EE.UU. Además, un razonador puede deducir que si dos instancias de Persona tienen el mismo número de Seguridad Social de los EE.UU., es porque dichas instancias se refieren al mismo individuo.

Restricciones de propiedad en OWL Lite

OWL Lite permite establecer restricciones sobre la forma en que las propiedades son utilizadas por las instancias de una clase. Estos elementos (y las restricciones de cardinalidad en la siguiente sección) se usan dentro del contexto de una restricción de esta forma owl:Restriction. El elemento owl:onProperty indica la propiedad restringida. Las dos restricciones que siguen a continuación indican los valores que pueden ser utilizados, mientras que las restricciones incluidas en la siguiente sección indican la cantidad de valores que pueden ser utilizados.

- allValuesFrom: La restricción allValuesFrom se establece sobre una propiedad con respecto a una clase. Esto significa que esta propiedad sobre una determinada clase tiene una restricción de rango local asociada a ella. De este modo, si una instancia de la clase está relacionada con un segundo individuo mediante esa propiedad, entonces puede deducirse que el segundo individuo es una instancia de una clase de la restricción de rango local. Por ejemplo, la clase Persona puede tener una propiedad denominado tieneHija restringida a tener allValuesFrom de la clase Mujer. Esto significa que si un individuo Louise esta relacionado mediante la propiedad tieneHija con un indivíduo Deborah, entonces un razonador puede deducir que Deborah es una instancia de la clase Mujer. Esta restricción permite que la propiedad tieneHija sea utilizada por otras clases, como puede ser la clase Gato, y tener una restricción de valor apropiada asociada con el uso de la propiedad sobre esa clase. En este caso, tieneHija podría tener la restricción de

rango local Gato cuando se asocie con la clase Gato y la restricción de rango local Persona cuando se asocie a la clase Persona. Obsérvese que un razonador no puede deducir únicamente de la restricción AllValuesFrom que realmente existe al menos un valor para la propiedad.

- someValuesFrom: La restricción someValuesFrom se establece sobre una propiedad con respecto a una clase. Una clase particular puede tener una restricción sobre una propiedad que haga que al menos un valor para esa propiedad sea de un tipo concreto. Por ejemplo, la clase ArticuloSobreWebSemántica puede tener una restricción someValuesFrom sobre la propiedad tienePalabraClave que indica que algunos valores de la propiedad tienePalabraClave pueden ser instancias de la clase TemaSobreWebSemántica. Esto permite la opción de tener múltiples palabras claves y, siempre que una o más sean instancias de la clase TemaSobreWebSemántica, el papel será consistente con la restricción someValuesFrom. A diferencia de allValuesFrom, someValuesFrom no restringe que todos los valores de una propiedad sean instancias de una misma clase. Si miArticulo es una instancia de la clase ArticuloSobreWebSemántica, entonces miArticulo está relacionada por la propiedad tienePalabraClave con al menos una instancia de la clase TemaSobreWebSemántica. Obsérvese que un razonador no puede deducir, como podría hacer con las restricciones allValuesFrom, que todos los valores de tienePalabraClave son instancias de la clase TemaSobreWebSemántica.

Restricciones de cardinalidad de OWL Lite

OWL Lite incluye una forma limitada de restricciones de cardinalidad. Las restricciones de cardinalidad de OWL (y OWL Lite) se refieren a restricciones locales, puesto que se definen en las propiedades con respecto a una clase particular. Es decir, las restricciones fuerzan la cardinalidad de esa propiedad sobre instancias de esa clase. Las restricciones de cardinalidad de OWL Lite son limitadas porque permiten solamente realizar indicaciones referentes a cardinalidades de valor 0 ó 1 (no está permitido añadir valores arbitrarios para la cardinalidad, como es el caso en OWL DL y OWL Full).

- minCardinality: La cardinalidad se establece sobre una propiedad con respecto a una clase particular. Si se establece minCardinality (cardinalidad mínima) de 1 sobre una propiedad con respecto a una clase, entonces cualquier instancia de esa clase estará relacionada al menos con un individuo mediante esta propiedad. Esta restricción es otra manera de decir que es necesario que la propiedad tenga un valor para todas las instancias de la clase. Por ejemplo, la clase Persona no debería tener restricciones de cardinalidad mínima establecidas sobre la propiedad tieneDescendidente, puesto que no todas las personas tienen descendientes. Sin embargo, la clase Padre debería tener una cardinalidad mínima de 1 sobre la propiedad tieneDescendencia. Si un razonador conoce que Louise es una Persona, entonces nada puede ser deducido a partir de la cardinalidad mínima sobre su propiedad tieneDescendiente. Una vez que se

conoce que Louise es una instancia de Padre, un razonador puede deducir que Louise está relacionada con al menos un individuo mediante la propiedad tieneDescendiente. Con sólo esta información, un razonador no puede deducir el número máximo de descendientes para las instancias individuales de la clase Padre. En OWL Lite las únicas cardinalidades mínimas permitidas son 0 ó 1. Una cardinalidad mínima de 0 en una propiedad indica (en ausencia de más información específica) que la propiedad es opcional con respecto a una clase. Por ejemplo, la propiedad tieneDescendiente puede tener una cardinalidad mínima de 0 para la clase Persona (mientras esté definido tener información más específica con cardinalidad mínima de 1 en la clase Padre).

- maxCardinality: La cardinalidad se establace sobre una propiedad con respecto a una clase particular. Si se establece maxCardinality (cardinalidad máxima) de 1 sobre una propiedad con respecto a una clase, entonces cualquier instancia de esa clase estará relacionada como máximo con un individuo mediante dicha propiedad. Una restricción de maxCardinality 1 es, en ocasiones, denominada una propiedad funcional o propiedad única. Por ejemplo, la propiedad tieneRegistradoEstadoDeVotación sobre la clase CiudadanosdeEstadosUnidos puede tener una cardinalidad máxima de uno (ya que los ciudadanos sólo pueden votar en un Estado). Por tanto, un razonador puede deducir que instancias individuales de la clase CiudadanosdeEstadosUnidos no pueden estar relacionadas con dos o más individuos distintos a través de la propiedad tieneRegistradoEstadoDeVotación. Un razonador no puede deducir que la cardinalidad mínima sea 1 solamente de la cardinalidad máxima de 1. Puede ser útil indicar que determinadas clases no tienen valores para una propiedad particular. Por ejemplo, instancias de la clase PersonaSoltera no deberían estar relacionadas con ningún individuo mediante la propiedad tieneCónyuge. La situación está representada por una cardinalidad máxima de 0 sobre la propiedad tieneCónyuge de la clase PersonaSoltera.

- cardinality: La cardinalidad se presenta como una ventaja cuando es útil establecer que una propiedad tiene sobre una clase minCardinality 0 y maxCardinality 0, o ambos minCardinality 1 y maxCardinality 1. Por ejemplo, la clase Persona tiene exáctamente un valor para la propiedad tieneMadreBiológica. De esta manera, un razonador puede deducir que dos instancias distintas de la clase Madre no pueden ser valores para la propiedad tieneMadreBiológica de la misma persona.

Se discutieron nombres alternativos para denominar estas formas restrictivas de cardinalidad. En la actualidad se recomienda incluir cualquiera de los nombres en un sistema front end. Si desea obtener más información sobre este asunto, puede encontrar el mensaje más relevante de los archivos públicos de lista de correo del grupo WebOnt en http://lists.w3.org/Archives/Public/www-webont-wg/2002Oct/0063.html.

Intersección de clase en OWL Lite

OWL Lite dispone de un contructor de intersección, pero, al utilizarlo, limita el uso del lenguaje.

- intersectionOf: OWL Lite permite establecer intersecciones entre clases identificadas y restricciones. Por ejemplo, la clase PersonaEmpleada se puede describir como la intersectionOf (intersección entre) Persona y ObjetosEmpleados (podrían ser definido como objetos que tienen una cardinalidad mínima de 1 en la propiedad tieneJefe). A partir de esto, un razonador podría deducir que cualquier PersonaEmpleada tiene por lo menos un jefe.

Tipos de datos en OWL

OWL usa los mecanismos de RDF para los valores de datos. Consulte la Guía de OWL sección sobre tipos de datos para obtener una descripción más detallada de los tipos de datos contruidos en OWL, tomados en gran parte de los tipos del esquema XML.

Información de cabecera en OWL Lite

OWL Lite permite incluir ciertas nociones de ontología y relaciones y adjuntar información a las ontologías. Consulte la Referencia de OWL para obtener información más detallada y la Guía de OWL para visualizar algunos ejemplos.

Propiedades de anotación en OWL Lite

OWL Lite permite realizar anotaciones en clases, propiedades, individuos y cabeceras de ontologías. El uso de estas anotaciones está sujeto a ciertas restricciones. Consulte la sección sobre anotaciones en la Referencia de OWL para obtener información más detallada.

3.4.2 Encontrar en algún repositorio de ontologías, una ontología de dominio.

Para encontrar una ontología de dominio, he utilizado el siguiente repositorio:

http://www.daml.org/ontologies/keyword.html

Por ejemplo esta ontología:

http://www.daml.org/ontologies/406

Es una ontología que pertenece al dominio de los sistemas de comunicación (Radio, televisión, teléfono…)