Universitat Jaume I

Departamento de Lenguajes y Sistemas Informáticos

Tesis Doctoral

Extracción y análisis de información desde la perspectiva de la Web Semántica

Roxana María Danger Mercaderes

Director: Dr. Rafael Berlanga Llavorí

Febrero 2007

Castellón, España

A mis padres.

Agradecimientos

No es mérito mio en absoluto la presente tesis doctoral. Muchas, muchas personas han contribuido a mi formación profesional y mi desarrollo y estabilidad personal, lo que me ha permitido llegar hasta aquí. En especial deseo agradecer a:

- mi tutor Dr. Rafael Berlanga Llavorí y su esposa María José Aramburu, por todo su apoyo y ayuda, no sólo en el ámbito profesional sino también en el personal, que ha sido aún mucho más importante para mí. No he recibido muestras más sinceras de amistad y solidaridad que las recibidas por ustedes y su familia. No hay palabras que describan mi agradecimiento, pero al menos algo está escrito,
- mis compañeros del grupo de investigación, con los que he compartido dudas, ilusiones y desilusiones. Un gracias muy especial a Juanma y a Jordi con quienes he compartido los tres años más duros de mi vida y de quienes he recibido la amistad, apoyo y cariño que no esperaba encontrar a 8000 km de mi isla,
- mis profesores y amigos de Cuba que no me olvidan,
- mi abuelito y demás familiares en Cuba, que me regalan alegría en cada llamada,
- mis padres y hermanas por ser ejemplo, apoyo, amor... sin importar la lejanía,
- Ivana y Oscar que me han acogido y quieren como una hija,
- Eris por su amor infinito...

Esta tesis ha incluido muchos doctorados:

el profesional, el de la vida, el de la emigración...

Sin ustedes, ninguno hubiera terminado satisfactoriamente.

Mil gracias!!!

Resumen

Actualmente, la búsqueda de información en Internet no se concibe sin el empleo de los llamados *motores de búsqueda de la web*. Pero, a pesar de su popularidad, las experiencias durante las búsquedas se vuelven, en muchos casos, frustrantes. Primero, por la imposibilidad de realizar una petición precisa de la información requerida; segundo, por el inmenso número de respuestas devueltas; y tercero, por la imprecisión y poca relevancia de dichos resultados. La causa fundamental de tal insatisfacción se deriva precisamente de lo que le ha dado fuerza y popularidad a la web: el empleo del lenguaje HTML para la codificación de los documentos. El HTML está orientado a la visualización, y no permite la formalización del contenido semántico de los documentos, que queda así fuera del alcance de los ordenadores.

Una nueva forma de web, llamada *Web Semántica*, ha sido concebida para proveer recursos web cuya información pueda ser compartida y procesada tanto por herramientas automáticas como por usuarios humanos. El sustento de la Web Semántica es un conjunto de estándares que dan uniformidad sintáctica y fundamentación semántica a todos los conocimientos descritos en ella. Así, por ejemplo, la lógica descriptiva ha sido escogida como basamento teórico para la descripción de los objetos de la web, y los lenguajes RDF y OWL para su expresión sintáctica.

Sin embargo quedan muchos obstáculos por vencer para hacer de la Web Semántica una realidad. Los más importantes son: 1) transformar la información actualmente disponible en la web en información semántica mantenida por medio de ontologías; 2) construir mecanismos de análisis que manipulen instancias ontológicas de la Web Semántica; 3) crear estructuras que permitan realizar un análisis semántico para colecciones de documentos para los que no se dispone una formalización conceptual de los temas tratados o son éstos muy dispersos.

Precisamente, cada uno de éstos son los temas abordados en esta tesis. En primer lugar, se introduce una metodología para la extracción de instancias ontológicas complejas desde textos (y documentos web) basada en la información ofrecida por ontologías de dominio. La solución propuesta no solamente permite extraer las entidades

de dichas ontologías descritas en los textos, sino además reconstruir instancias complejas en las que se aprecie las relaciones entre instancias.

En segundo lugar, se formaliza un modelo multidimensional que permite utilizar instancias ontológicas en procesos de análisis como los propuestos en los modelos de almacenes de datos, y se describe un mecanismo para inferir esquemas multidimensionales interesantes. Todo ello está basado en ciertos mecanismos de razonamiento de la lógica descriptiva y en información de análisis provista por el analista de datos.

Por último, se trata el problema de analizar la información contenida en páginas web o en una colección de documentos sobre temas de dominio general o para los cuales se desconoce una ontología específica. En estos casos, es inviable extraer y analizar instancias ontológicas por el método propuesto. En su lugar, se propone emplear técnicas de *minería de textos* y utilizar recursos léxicos externos para reconstruir espacios conceptuales multidimensionales que permitan realizar análisis de este tipo de colecciones.

Abstract

Nowadays, *search engines* are popular tools indispensable for finding any kind of information in the Internet. Nevertheless, web searching can still constitute a frustrating experience. The main reasons for this are the impossibility of specifying exactly the nature of the required information, the huge number of results provided, and the lack of precision and appropriateness of such results. This situation arises from what has given the Web its strength and popularity: the coding of documents in HTML. Visualization is the only concern of HTML coding, and no formalization of the semantic of documents is possible, so that computers can not access the meaning of texts.

The *Semantic Web* is a new form of web conceived for allowing human users and software tools to process and share the same sources of information. The Semantic Web builds on a set of standards which provide syntactic consistency and semantic value to all of its content. For example, description logic is used as the theoretic base for the description of web items, and the languages RDF and OWL for their syntactic representation.

Despite the promising beginnings, many obstacles still need to be overcome to make the Semantic Web a reality. The most important tasks to perform are: 1) translate the information now available on the web into semantic information maintained through ontologies; 2) build analysis tools for the manipulation of ontological instances; 3) create structures for the semantic analysis of sets of documents lacking of a conceptual formalization.

This thesis tackles each one of these issues. Firstly, a methodology for the extraction of complex ontological instances from texts (and web documents) is introduced. Such methodology allows to extract from texts entities belonging to given domain ontologies, and to reconstruct complex instances in which relations between instances are also taken into account.

The second step is the formalization of a multidimensional model in which ontological instances are used in analysis processes such as those in the datawarehouse model.

A method for inferring interesting multidimensional schemas is also described. These tools build on description logic inference mechanisms, and on information provided by a human analyst.

Finally, we deal with the problem of analyzing data contained in web pages or in a set of documents which domain is not specific or for which an ontology is not available. In these cases, the above methods of extraction and analysis of ontological instances can not be applied. The proposal is to make use of *text mining* techniques and external lexical resources in order to reconstruct multidimensional conceptual spaces suitable for the analysis of these kinds of text collections.

Índice General

Índice de Figuras

Índice de Tablas

Índice de algoritmos

Capítulo 1

Introducción

La Internet actual ya poco nos recuerda a la Internet de 1960. Un cambio colosal en cuanto a tipos y número de usuarios y funciones han hecho de la red de redes del mundo un fenómeno social y económico. En enero de 2006, Internet alcanzó los mil cien millones de usuarios y se considera una herramienta imprescindible para cientos de millones de personas por razones económicas, políticas o sociales.

La *web* o WWW (del inglés World Wide Web), el servicio Internet más popular, nace alrededor de 1990 a partir de un proyecto del CERN[1], en el que Tim Berners-Lee encabezó un grupo de físicos que se encargaron de diseñar el lenguaje HTML para la recuperación y navegación de documentos a través de Internet. La web ha ampliado el abanico de funcionalidades de Internet y actualmente la *Internet económica* es tan importante como la *Internet académica* de sus inicios. Baste destacar que Google, Yahoo!, eBay/Skype, Vonage, AOL, News Ltd y Amazon son todas compañías totalmente integradas en Internet que luchan por su supervivencia en el mundo del ciberespacio. También realizar transacciones económicas y búsquedas por viajes, juegos, música y conocimientos son tareas rutinarias para muchos, gracias a la web.

A pesar de la enorme importancia de la web, un detalle de su diseño la hace inviable para manipular, de manera automática, la información que sus páginas atesoran. Tim Berners-Lee concibe en el 2002 el segundo prototipo de web, la *Web Semántica*. En esta oportunidad su objetivo se aleja de las necesidades de visualización de la información y se centra proporcionar el intercambio de conocimientos a través de documentos sintáctica y semánticamente bien definidos, de manera que ordenadores y usuarios puedan comprender e intercambiar la información mantenida en dichos documentos.

Desde su propuesta numerosos trabajos han ido conformando la arquitectura general de esta nueva web, y hasta el momento se ha definido la sintaxis (a través de *RDF* y *OWL*) y el modelo

[1] CERN es el Consejo Europeo para la Investigación Nuclear.

semántico (a través de la *lógica descriptiva*) que proporcionan las mayores ventajas para el desarrollo de la Web Semántica. Su base está en la creación de *ontologías* que permitan describir el mundo de manera formal. Así, todo concepto puede ser descrito formalmente y toda relación entre conceptos puede ser explícita o implícitamente especificada.

Si bien es cierto que la formalización de una ontología es en sí un problema difícil, muchas organizaciones y áreas del conocimiento ya han comenzado a presentar sus datos semánticamente etiquetados como propone W3C[1]. Y ya comienzan a surgir buscadores semánticos. El primero de ellos, construido sobre la Interfaz de Programación de Aplicaciones de Google, *Swoogle*, ya cuenta con más de un millón de documentos semánticos y mantiene unos trecientos millones de tripletas (relaciones entre conceptos u objetos).

Sin embargo, el cuello de botella real para el desarrollo de la Web Semántica está dado en lo que se conoce como *población de la Web Semántica*, o sea, en la descripción de los objetos del mundo a través de las especificaciones formales proporcionadas por las ontologías. Es decir, no basta contar con una descripción de los conceptos y las relaciones entre ellos, es indispensable además ejemplificar estos conceptos con las instancias ontológicas que aparecen en el mundo real y les dan sustento. Sin embargo, una gran parte de esta información está descrita en lenguaje natural. Es por tanto una tarea prioritaria transformarla en información útil para la Web Semántica. Sólo así, toda la sociedad podrá compartir e intercambiar información con un único formato y una semántica conocida.

Por otro lado, cuando la Web Semántica esté poblada con una cantidad suficiente de instancias, la tarea de análisis, recuperación y razonamiento sobre la información disponible será de incalculable valor, tanto en aplicaciones económicas como sociales y científicas en cualquier área de conocimiento.

Por último, el dinamismo y heterogeneidad de la información disponible en la web es tal que sólo su clasificación en las diferentes áreas de conocimiento (o en las ontologías) es una tarea compleja. Por ello, debe contarse con tareas básicas que permitan clasificar, organizar y analizar semánticamente las diferentes colecciones de documentos disponibles en la web. Ello podría, por un lado, servir de base para la asignación de ontologías a partes muy concretas de la clasificación obtenida; y por otro, permitir realizar análisis generales acerca de los conceptos o temas aparentemente disjuntos, para los que sí pueden ser reconocidos nexos que les una.

Estos problemas son precisamente la motivación que ha llevado al desarrollo de la presente tesis, cuyos **objetivos** se resumen a continuación:

[1]W3C, Consorcio World Wide Web, es una asociación internacional cuya misión es "...el desarrollo de protocolos y pautas que aseguren el crecimiento futuro de la web" (http://www.w3c.es/consorcio).

1. Desarrollar una herramienta que sirva para la población de la Web Semántica, que sea escalable, precisa y fiable y que no implique complicados mecanismos de procesamiento de los textos.

2. Ampliar el abanico de operaciones disponibles sobre ontologías y las instancias ontológicas de manera que análisis más complejos puedan ser provistos por la Web Semántica.

3. Formalizar un marco de trabajo para el análisis de la información disponible en la Web Semántica que incluya mecanismos de descubrimiento de nuevos conocimientos.

4. Crear un instrumento para el análisis semántico de colecciones de documentos heterogéneas que sirva tanto para el descubrimiento de nuevas relaciones semánticas como para la asignación o clasificación de documentos a espacios conceptuales.

Dar solución a estos problemas ha implicado tratar con numerosos estándares y soluciones computacionales a problemas similares. Como punto de partida, las lógicas descriptivas y el lenguaje OWL han servido de base para todo el trabajo desarrollado. La extracción de instancias ontológicas, en particular, ha necesitado del estudio y selección de las técnicas apropiadas de procesamiento de lenguaje natural según las hipótesis del trabajo. Por último, las técnicas para el análisis tanto de las instancias ontológicas como para las colecciones heterogéneas de documentos han estado basadas en las relativamente recientes líneas de investigación de *minería de datos* y *modelos multidimensionales*.

Gran parte de los trabajos desarrollados en esta tesis han dado solución a problemas que surgieron durante el desarrollo del proyecto CRISOL[1]. La siguiente sección describe el ámbito de aplicación del mencionado proyecto que sirve además como área de conocimiento modelo en la que pueden ser aplicadas casi todas las contribuciones de la presente tesis. En la última sección se describe el trabajo desarrollado a través de la descripción de los temas tratados en cada uno de los siguientes capítulos.

1.1 Ámbito de aplicación

“Análisis y Explotación de Conocimiento Espacio-Temporal en una Web Semántica. Aplicación a la Investigación Arqueológica” es el nombre del proyecto CRISOL del que forma parte, entre otras, la Universidad Jaume I. A grandes rasgos la finalidad del proyecto es proporcionar un marco de análisis y exploración de conocimientos de dominio específico en la Web Semántica, tratando

[1] Proyecto CICYT TIC2002-04586-C04-03.

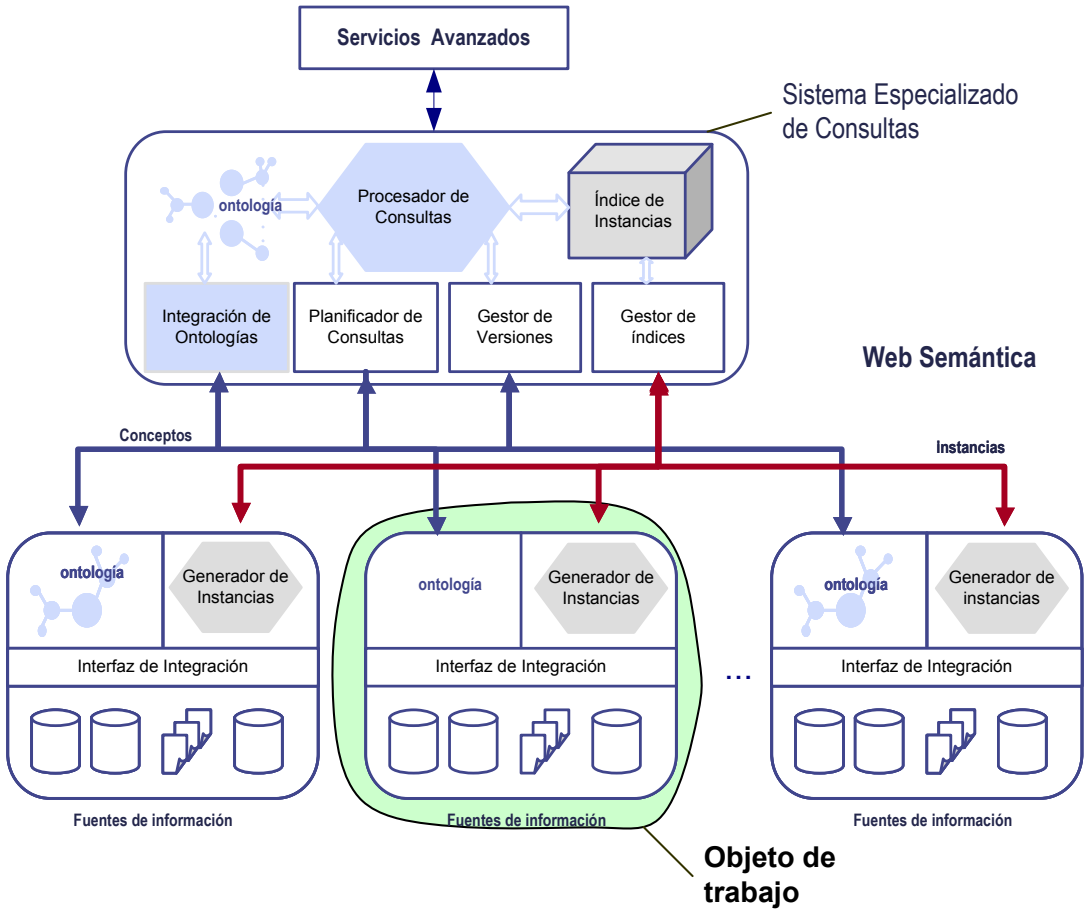


Figura 1.1: Arquitectura del proyecto.

todos los aspectos: la creación de la ontología, la extracción (semi) automática de instancias ontológicas desde diferentes fuentes de información, la consulta distribuida sobre la ontología y el mantenimiento de todos los conocimientos extraídos, considerando para ello el empleo de índices multidimensionales para el acceso rápido a las instancias, así como las tareas de versionado y evolución de la ontología.

En la Figura 1.1 se muestra la arquitectura general del proyecto. Como puede apreciarse en la base de toda la arquitectura está la *ontología* y el *proceso de extracción de instancias ontológicas*. Encima de esta capa, que presumiblemente se integrará a la Web Semántica, se encuentra todo el sistema de consultas especializado y otros servicios avanzados que permitirán el acceso y consulta de los datos almacenados a agentes externos.

El dominio de aplicación concreto seleccionado para desarrollar el sistema fue el de arqueología. Primero, porque es una ciencia con conceptos, clasificaciones y codificaciones bien definidos (aunque poco formalizados matemáticamente) sobre la cual es plausible crear una ontología

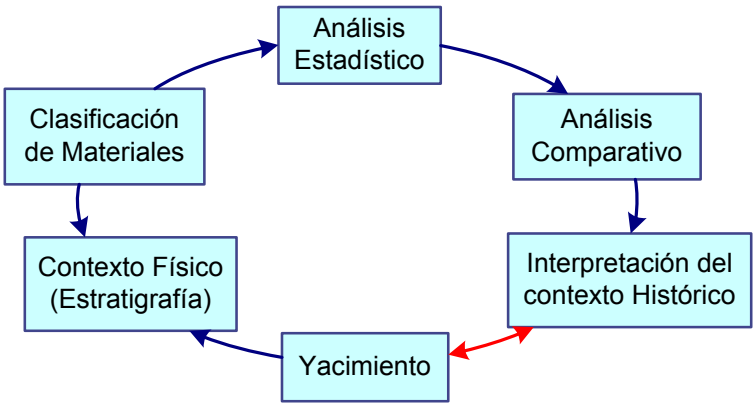


Figura 1.2: Proceso de exploración arqueológica.

de dominio que recoge el consenso de muchos especialistas. Segundo, por la necesidad de extraer información, analizar datos archivados y descubrir nuevos conocimientos desde las numerosas memorias arqueológicas que se disponen.

Escenario de investigación arqueológica

Toda investigación arqueológica sigue el ciclo de vida que se describe en la Figura 1.2. Se divide en tres partes: una primera realizada en campo que permite excavar y extraer los materiales interesantes del yacimiento; otra realizada en laboratorio, que permite clasificar, y determinar medidas objetivas cuantitativas o cualitativas de las características de los materiales encontrados; y por último, una fase de análisis comparativo y de interpretación que permitirá vincular dicho yacimiento con períodos arqueológicos, culturas y descubrir hechos interesantes exclusivos o no del yacimiento y/o vincularlos a hechos o apariciones anteriores. Un descubrimiento durante cualquiera de las fases, podría provocar una reexcavación del yacimiento y por consiguiente, la repetición de todo el ciclo.

Hasta donde conocemos, no existe aún una ontología general que permita describir yacimientos arqueológicos. El proceso de creación de una ontología entraña serias dificultades. Por un lado, debe ser lo suficientemente general como para cubrir todos los temas tratados en las bases de datos y memorias documentales. Por otro lado, debe captar suficientes detalles como para permitir el almacenamiento y consulta de aspectos interesantes de los yacimientos. Encontramos varios tesauros y taxonomías que capturaban algunos conceptos de nuestro dominio, pero en general resultaron demasiado abstractos para capturar los detalles de nuestro escenario. De esta manera, se ha diseñado una ontología asentada en métodos de excavación y esquemas de clasificación de la industria de los artefactos bien establecidos. La ontología de dominio finalmente resultante,

ArchaeoOnto, agrupa aproximadamente unos 500 conceptos y relaciones correspondientes a la arqueología prehistórica. La ontología completa en *OWL* puede verse en Danger (2006) y un fragmento de ella será explicado y definido formalmente en el Capítulo 3, el que servirá de guía para gran parte de los ejemplos expuestos.

1.2 Breve descripción del trabajo

Los capítulos del segundo al sexto de la presente tesis dan solución a los problemas planteados al inicio de este capítulo. Se inicia por un análisis detallado de las tecnologías a aplicar y su actual grado de desarrollo. Luego se detallan las aportaciones de la presente tesis. En el séptimo capítulo se resumen las contribuciones y limitaciones de las propuestas del trabajo.

En el segundo capítulo puede encontrarse una descripción cronológica del desarrollo de la tecnología empleada en la presente tesis. Se comienza abordando las tareas y enfoques del *procesamiento del lenguaje natural* y las formas de representación del conocimiento más utilizadas en *inteligencia artificial*, haciendo especial hincapié en la *lógica descriptiva*. Más tarde, se explica la arquitectura de la Web Semántica y sus retos actuales. Por último, se describen los modelos multidimensionales para el análisis de datos históricos y su relación con las tecnologías de la Web Semántica.

En el tercer capítulo, se introduce toda la formalización teórica que sirve de base a las aportaciones de la presente tesis. Primeramente se describe el lenguaje particular de la lógica descriptiva a utilizar y más tarde se introducen las definiciones, propiedades y algoritmos necesarios para realizar dos tareas de razonamiento de especial interés que en la presente tesis se han identificado. La primera de ellas es el cálculo de caminos de agregación entre conceptos. La segunda, aborda la problemática del razonamiento sobre instancias ontológicas. De esta forma, se establece la base para realizar operaciones de especialización, abstracción, agregación, diferencia y diferencia simétrica entre instancias ontológicas, que hasta el momento no habían sido formalmente descritas.

El cuarto capítulo tesis tiene como objetivo resolver el problema de la extracción de instancias ontológicas desde textos escritos en lenguaje natural. Para ello, en primer lugar se realiza un análisis detallado de las principales aportaciones en este sentido con especial interés en las herramientas externas empleadas y tipos de análisis lingüísticos utilizados. Se introduce una metodología para la población de la Web Semántica y se explica el sistema de extracción de instancias ontológicas desarrollado. Por último, se dan algunos resultados experimentales y aparece una discusión y comparación con los sistemas descritos en la literatura.

El capítulo quinto contiene una formalización para el análisis de instancias ontológicas. Se comienza explicando la necesidad y características de tal herramienta a través de un ejemplo de aplicación real. Posteriormente se describe formalmente cómo podría en el entorno de la Web Semántica desarrollarse tal herramienta y por último, en sus conclusiones se tratan las ventajas y limitaciones de la formalización propuesta.

El sexto capítulo, formaliza y describe un sistema que permite crear espacios conceptuales asociados a colecciones de documentos de los que se desconoce las ontologías a las que se refiere. Para ello, emplea recursos lingüísticos externos que le proporcionan jerarquías conceptuales para una clasificación semántica de los documentos. Se define además cómo crear cubos multidimensionales para el análisis de los contenidos asociados a tales colecciones. Por último se muestran algunos resultados experimentales y una breve discusión de éstos.

El último capítulo resume las principales aportaciones de la presente tesis y la producción científica derivada de su desarrollo.

Capítulo 2

Antecedentes

2.1 Un poco de historia

Los datos, la información y el conocimiento rigen nuestra propia evolución[1]. Ya desde el siglo XIX, algunos economistas aseveraban la importancia del conocimiento como arma fundamental para el desarrollo económico de una organización y un siglo más tarde toda la sociedad estaba comenzando a hablar de la "*sociedad del conocimiento*", la sociedad que se mueve gracias al conocimiento[2].

La aparición de la computación e Internet constituyen un catalizador en el proceso de generación e intercambio de información y conocimientos. El almacenamiento electrónico de información no solamente ha permitido el acceso a gigantescos almacenes de datos, sino también, el desarrollo y empleo de métodos para extraer información selectivamente, transformarla, compartirla y generar nuevos conocimientos. Precisamente, el *descubrimiento de conocimiento* (en inglés, *knowledge discovery*), es una disciplina de la ciencia de la computación que se encarga de la "extracción no trivial de información implícita, previamente desconocida y potencialmente útil desde los datos", Fayyad *et al.* (1996).

Son muy ambiciosas las metas de esta disciplina. Sin embargo, ésta va ganándole terreno a las incertidumbres gracias a la aparición continua de nuevos métodos y herramientas para la manipulación del conocimiento. En el campo de las bases de datos, por ejemplo, los llamados *data warehouse* o *almacenes de datos*, basados en *modelos multidimensionales*, ya son herramienta

[1] Los *datos*, constituyen representaciones de características de entidades que se convierten en *información* cuando se conoce el contexto al que dichos datos se refieren. A su vez, la interpretación de la información y su síntesis con experiencias anteriores, datos e informaciones permite generar conocimiento.

[2] Un interesante análisis económico-filosófico de todo este fenómeno puede verse en http://www.cema.edu.ar/publicaciones/download/documentos/192.pdf.

obligada en las compañías que manipulan grandes volúmenes de información, tal como veremos en la sección 2.5.

Sin embargo, las características propias del lenguaje natural impone grandes dificultades para la manipulación del conocimiento así expresado. Su naturaleza no-estructurada, así como la heterogeneidad de símbolos para describirla y la asociación de una semántica contextualizada (por la fecha y lugar de redacción, las vivencias del autor, etc.) hacen muy difícil el descubrimiento de la información contenida en los textos.

Los primeros pasos en esta dirección datan de los años cuarenta cuando se abordó el desafío de la traducción automática, aunque sólo ahora aparecen los primeros sistemas con resultados satisfactorios. Así surgió el *procesamiento del lenguaje natural*, disciplina encargada de desarrollar técnicas computacionales para el modelado y procesamiento del lenguaje. Estas técnicas van desde aplicaciones simples para el conteo y/o estimación de la distribución de palabras en colecciones de textos, hasta sofisticadas que resuelven problemas de *preguntas y respuestas* (en inglés, *question and aswering* o el acrónimo Q&A) o de *extracción de información*.

El 31 de Agosto 1955, J. McCarthy, M. L. Minsky, N. Rochester y C. E. Shannon lanzaron una propuesta para reunir en el verano de 1956 a un grupo de investigadores que quisieran trabajar en el descubrimiento de las características del aprendizaje y la inteligencia. En el encuentro, ahora conocido como la Conferencia de Dartmouth, DAIC (1956), se acuña con el nombre de *inteligencia artificial* a todo el proceso de modelar la inteligencia humana en sistemas computacionales. Hoy, casi a 50 años de esta conferencia, y a pesar del inmenso abanico de ciencias que lo estudian y las diversas áreas que le constituyen (entre otras: el reconocimiento de patrones, los métodos de búsqueda de hipótesis, aprendizaje, planificación, desarrollo de heurísticas, representación, lógica, inferencia, conocimiento de sentido común y razonamiento, tal como ha explicado John McCarthy en su disertación *"What is artificial intelligence?"*, McCarthy (2004)), los problemas allí planteados mantienen toda su vigencia.

Mientras algunos investigadores estaban interesados en la inteligencia artificial, otros se encargaban de llevar el destino de la computación por un camino muy distinto, sin saber, que al final ambos se unirían. Es el surgimiento de *Internet*. En una primera instancia es plataforma de intercambio de información entre profesionales e instituciones universitarias y que ha devenido en el más amplio almacén de datos y mercado internacional.

Veinte años después del surgimiento de Internet, se lanza la convocatoria de la creación de la *Web Semántica*. Ideado por Tim Berners-Lee, creador de *HTML*, *URI*, *HTTP*, *WWW* y actualmente director del *W3C* (acrónimo del inglés, World Wide Web Consortium), la Web Semántica no es más, a decir de sus propias palabras, que "una extensión de la web actual en la cual la información recibe un significado bien definido, permitiendo a los computadores y las personas trabajar en

cooperación de la mejor forma", Lee *et al.* (2001). El reto ahora se traduce en *dar semántica a los datos*, de manera que no sean sólo comprensibles por los humanos sino también por las computadoras. Para ello, es imprescindible dotar a los datos de cierta base sintáctica-semántica, que les permita homogeneizar sus descripciones y una lógica que les permita deducir la verdad de las hipótesis e inferir nuevos conocimientos.

Las secciones siguientes abordan las temáticas relacionadas con el procesamiento del lenguaje natural, los métodos de representación e inferencia de conocimientos, la Web Semántica y los modelos multidimensionales.

2.2 Procesamiento del lenguaje natural

Las técnicas de *procesamiento del lenguaje natural (PLN)* (en inglés, *natural language processing* o el acrónimo NLP) se remontan a los años 40 cuando partiendo del modelo de computación algorítmica expuesto por Turing en 1936, se comenzó el desarrollo de la *teoría de autómatas*.

Los primeros intentos fueron introducidos en 1948 por Shannon, quien aplicó modelos probabilísticos de *procesos discretos de Markov* en autómatas para el lenguaje. Entre 1951 y 1956, Kleene introdujo los autómatas finitos y las expresiones regulares. A partir de esta fecha e inspirado en los trabajos de Shannon, Chomsky inició sus estudios de lo que ahora se conoce como *teoría de lenguajes formales*. Comenzó sus trabajos con *máquinas de estados finitos* como forma de caracterizar las gramáticas y definió los *lenguajes de estados finitos* como aquellos lenguajes generados por gramáticas de estados finitos. La Teoría de Lenguajes Formales incluye las gramáticas libres de contexto para lenguaje natural, definidas por Chomsky en el 1956, pero descubiertas independientemente por Backus, en el 1959. Hasta mediados de los años 60, hubo un intenso trabajo en el desarrollo de las teorías de lenguajes formales y en el desarrollo de algoritmos de análisis de sintaxis (en lo sucesivo, parsing) de las construcciones sintácticas del lenguaje.

Un gran paso de avance fue el sistema SHRDLU[1], desarrollado por Terry Winograd (Winograd (1972)). Este programa era capaz de captar comandos escritos en lenguaje natural, correspondientes a acciones en un juego de bloques. Fue el primer intento de construir una gramática del inglés extensa y demostró que el problema del parsing ya había sido suficientemente comprendido y que era imprescindible enfocarse en el descubrimiento de la semántica y la intención del discurso. De esta manera comienzan a aparecer sistemas con representación semántica basada en lógicas, algunos de sus paradigmas aún dominan en este campo, tal como veremos en la sección 2.3.

[1]El nombre del sistema SHRDLU proviene de la frase mnemotécnica "ETAOIN SHRDLU" que agrupa las doce letras más frecuentes en los textos ingleses.

A pesar de los avances en la teoría de los lenguajes formales, ya en los años 80 se comenzó a dudar de la idoneidad de las gramáticas libres de contexto para todas las aplicaciones del procesamiento del lenguaje natural. En particular, las gramáticas no resultan adecuadas para representar todos los posibles textos, y los sistemas desarrollados, demasiado lentos al analizar frases largas (comunes en los textos), aún cuando se emplean heurísticas y procesamientos estadísticos que permitan decidir con antelación la sintaxis más probable de cierta frase. Así, en 1980, Church propuso el retorno a los autómatas de estados finitos para modelar la sintaxis.

Hasta mediados de los 90 muchos trabajos fueron desarrollados sobre la base de combinar las gramáticas libres de contexto con autómatas finitos, Jurafsky & Martin (2000), o mediante el uso de técnicas de análisis parcial tal como se propuso en Abney (1991).

Por otro lado, se identificaron los diferentes niveles del lenguaje natural, lo que permite desarrollar sistemas en cascada que aíslan los problemas y características que aparecen en cada nivel. Se han identificado los siguientes cuatro niveles:

- Morfológico, que permite a partir de una secuencia de caracteres devolver una secuencia de unidades significativas, empleando por ejemplo, reglas morfológicas para extraer raíces o diccionarios para encontrar sinónimos, hipónimos, hiperónimos[1], etc.

- Sintáctico, que analiza la estructura gramatical de las unidades léxicas y devuelve una estructura con los diferentes sintagmas de cada oración.

- Semántico, que a partir de los sintagmas genera una estructura o una lógica (veremos las lógicas y su uso en la representación de conocimiento en la sección 2.3) que representa el significado de la oración, independientemente del contexto.

- Pragmático o contextual, que lleva el análisis un poco más allá de los límites de la frase o la oración, por ejemplo, para determinar los antecedentes referenciales de los pronombres o dar una interpretación final de la frase dependiendo de las circunstancias que la rodean.

La mayor problemática relacionada con el procesamiento de lenguaje natural viene dada por la ambigüedad, referida a la necesidad de discernir la función y extensión de cada elemento en cada uno de los niveles. La resolución de la ambigüedad es una tarea difícil y en muchos casos no es soluble en el mismo nivel donde se presenta. Por ejemplo, a nivel léxico, una misma palabra puede tener varias raíces léxicas, y la selección de la apropiada se debe deducir a partir de cierto conocimiento básico (ya sean diccionarios, gramáticas, bases de conocimientos o correlaciones

[1] A una palabra se le llama hiperónimo de otra si su significado abarca al de la segunda, la cual a su vez es un hipónimo de la primera. Por ejemplo: *día* es hiperónimo de *lunes*, *martes*, etc. Ello implica que a su vez, *lunes*, *martes*, etc. son hipónimos de *día*.

estadísticas) o del contexto en el cual aparece dicha palabra. Sin embargo, a nivel contextual, una oración, a menudo, no significa lo que realmente se está diciendo pues elementos como la ironía tienen un papel importante en la interpretación del mensaje. A pesar de los numerosos estudios para resolver estos problemas, en la mayoría de los casos los resultados satisfactorios no superan el 60%, un umbral bastante por debajo de lo deseable en los sistemas de procesamiento de lenguaje natural.

A partir de la segunda mitad de la década de los noventa, se hizo evidente un cambio en los métodos del PLN, Manning & Schütze (1999). Abordar estos métodos es el objetivo de la siguiente subsección.

Últimas propuestas del PLN

En la década de los 90 las aplicaciones relacionadas con la extracción de información cobraron una gran importancia. Así lo demuestran las conferencias celebradas bajo el nombre "Message Understanding Conferences" (MUC) hasta el año 1997. En ellas se trataron tareas relacionadas con la extracción de entidades nombradas, atributos, hechos y eventos, obteniendo una fiabilidad en el mejor de los casos del 94% (para entidades nombradas) y en el peor, del 51% (en eventos). La última conferencia demostró la dificultad de mejorar los resultados obtenidos hasta el momento, Chinchor (2001), y dejó claro que mientras para el reconocimiento de entidades los sistemas alcanzaron resultados parecidos a los obtenidos por expertos humanos, la fiabilidad de las anotaciones por humanos de los restantes elementos, incluídos el *llenado de plantillas de objetos*[1], resultaron significativamente mejores que las conseguidas por los sistemas de anotación.

Las conferencias sucesoras, "Conference on Language Resources and Evaluation" (LREC), celebradas cada dos años desde el 2000, han demostrado un cambio en los recursos empleados en la extracción de información, por cuanto los sistemas expuestos comienzan a emplear recursos externos que apoyen la tarea. Es evidente que cuando leemos o hablamos, estamos usando conocimiento implícito acerca del significado y relación entre las palabras. Suplir dichos conocimientos es el objetivo de estos recursos externos. Algunos de éstos son:

- Diccionarios electrónicos, es decir, diccionarios tradicionales pero en formato electrónico.
- Bases de datos léxicas, diccionarios de términos dentro de un contexto específico.
- Tesauros, agrupaciones de términos en clases de acuerdo a su significado y dominio.
- Redes semánticas, representación de términos en una estructura de red para representar las relaciones entre ellos.

[1] Esta tarea intenta completar plantillas predefinidas que permiten describir los objetos de los cuales se hablan en los textos.

- Matrices léxicas, representación matricial en la que se enlazan términos y significados, que permiten determinar relaciones de polisemia y sinonimia.

- Ontologías, especificación de relaciones entre conceptos.

En las secciones 2.3 y 2.4 abordaremos con más detalle algunos de ellos.

A modo de resumen, podemos decir que a pesar de los 60 años de estudios en las tareas del procesamiento del lenguaje natural, aún el problema de "comprender" los textos está lejos de ser alcanzado. Los sistemas desarrollados incluían al inicio autómatas de estados finitos y gramáticas. Actualmente las técnicas se han desplazado hacia el uso de modelos probabilísticos o guiados por datos combinándolos con el uso de recursos lingüísticos que permitan una mejor desambiguación.

Sin embargo, aún resultan insuficientes los resultados obtenidos. El problema fundamental está en el intento de desarrollar sistemas capaces de analizar todos los tipos de ambigüedades que pueden aparecer en los textos de cualquier dominio, siendo precisamente el dominio o el contexto de un documento un factor importante en la reducción de la ambigüedad y en muchos casos, un dato conocido antes del proceso de "interpretación" del texto. En el Capítulo 4 se expone la aproximación propuesta en esta tesis para extraer información de textos de dominio específico, basado en el uso de ontologías de dominio.

2.3 Representación del conocimiento

La *representación del conocimiento* es una de las ramas de la IA que se encarga del diseño e implementación de lenguajes y sistemas que representen el conocimiento acerca del mundo. Abarca tanto el conocimiento del mundo (o dominios específicos de éste) como el conocimiento que permite realizar operaciones de razonamiento.

Los enfoques para la representación del conocimiento pueden dividirse, más o menos, en dos clases dependiendo si están o no basados en formalismos lógicos. Los enfoques no basados en lógica, surgieron sin una formalización que les sustente. Posteriormente, con el desarrollo de los lenguajes de representación del conocimiento basados en lógicas, algunos han sido traducidos a lógicas, con más o menos pérdida de semántica.

En la Tabla 2.1 se resumen algunas formas de representación del conocimiento no lógicas que han sido utilizadas desde los años 40. Varias cuestiones saltan a la vista. La primera está relacionada al consenso de la descripción del mundo mediante la descripción de los objetos y las secuencias de sucesos (o estados) que ocurren en él. La segunda es el reconocimiento y empleo de la herencia de propiedades como ente inherente de la descripción del mundo[1].

[1] Más adelante veremos que no solamente en las representaciones no lógicas está presente este concepto.

Por último, estas formas de representación emplean estructuras de datos *ad-hoc* y, en consecuencia, el razonamiento es desarrollado por procedimientos, también *ad-hoc*, que manipulan dichas estructuras, y no disponen de mecanismos formales que permitan la *inferencia* y verificación de *consistencia* basados en una semántica bien formalizada e independiente del contexto. Ello, a pesar de los más de 20 siglos de rigor científico de lo que hoy denominamos *lógica de primer orden*, y que había iniciado Aristóteles en sus famosos escritos de lógica.

Un importante descubrimiento permitió el desarrollo de los paradigmas de representación del conocimiento basados en la lógica: el reconocimiento de que los *frames* (al menos, sus características principales) podían ser dotados de semántica gracias a la lógica de primer orden, Hayes (1980). En ella, los elementos básicos de representación son caracterizados por predicados unarios, que permiten denotar conjuntos de individuos, y predicados binarios, que denotan relaciones entre los individuos.

Pero, por un lado esta caracterización no capturaba todas las restricciones que a través de las redes semánticas y los frames podrían describirse; por otro, ya estaba demostrada la viabilidad de algunos algoritmos de inferencia sobre estas estructuras *ad-hoc*, de manera que no requerían toda la maquinaria de la lógica de primer orden (aunque sí podían ser consideradas como parte de ésta), Brachman & Levesque (1985). Por último, diferentes características de los lenguajes de representación podían conducir a diferentes fragmentos útiles de la lógica de primer orden, donde los mecanismos de razonamiento conducen a problemas de diferente complejidad, algunos de los cuáles ya estaban solucionados considerando representaciones basadas en estructuras, que no necesariamente requerían los teoremas probados de la lógica de primer orden, Nardi & Brachman (2002).

Así surge la *lógica descriptiva*, un formalismo lógico que representa el conocimiento de un dominio de aplicación definiendo sobre la base de cierto *lenguaje de descripción*, primero, los conceptos relevantes del dominio (su terminología) y luego, usando estos conceptos para especificar las propiedades de los objetos e individuos que ocurren en el dominio (la descripción del mundo). En la sección 2.3.1 veremos las características fundamentales de esta forma de representación del conocimiento y en el 2.4.3 el papel que juega este formalismo lógico en todo el entramado de conceptos alrededor de la Web Semántica.

2.3.1 Lógica descriptiva

Baader, un importante investigador de la lógica descriptiva (DL, *description logics*), resume las ideas más importantes subyacentes a este formalismo de la siguiente manera:

Forma de representación	**Estructura de Representación**	**Herencia**	**Otros conceptos**	**Centro del conocimiento**
Redes Semánticas, Quillian, 1968	Grafo, siendo sus nodos objetos, conceptos, eventos o atributos y sus arcos etiquetados representan las relaciones entre dichos nodos	Sí	Operadores de cuantificación en redes semánticas particionadas (Gary Hendrix)	Objetos
Frames (marcos), Marvin Minsky, 1974	Los marcos representan clases de objetos (o un objeto en particular) a través de las llamadas ranuras que permiten describir los atributos de dicha clase (o los valores de cada atributo del objeto que se describe)	Sí	Inconsistencia de la base de conocimiento, valores por defecto, inferencias por analogías	Objetos
Scripts, Schank y Abelson, 1977	Marcos que describen eventos mediante la especificación del lugar, sujetos y objetos que intervienen, condiciones para que la acción tenga lugar, subeventos, resultados esperados, etc.	Sí, de eventos	Invocación [a]	Eventos
Reglas de producción, Post, 1943	Conjunto de pares ordenados $(antecedente, consecuente)$ que permite ejecutar o reconocer el cumplimiento de un *consecuente*, dada la presencia de las condiciones descritas por el *antecedente*.	No	Inferencia de reglas[b]	Estados

Tabla 2.1: Formas de representación del conocimiento no basados en lógica.

[a] La invocación permite la ejecución de un script, para analizar acontecimientos que pasan desapercibidos, o predecir acontecimientos que no se han observado explícitamente.

[b] La inferencia de reglas permite a partir de una base de conocimiento inicial, aplicar las reglas satisfacibles de manera que se alcance un objetivo. Así, a la salida del proceso se conocen todas las acciones a ejecutar y las condiciones intermedias hasta lograr el objetivo.

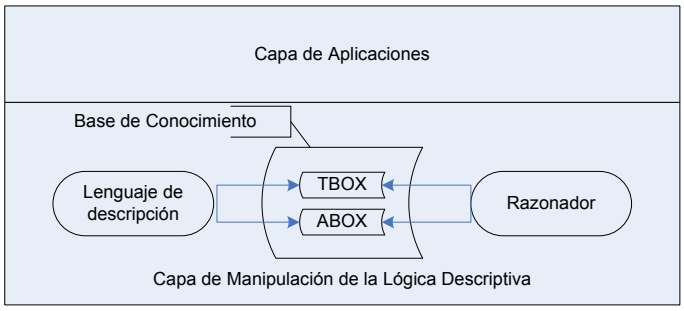


Figura 2.1: Arquitectura de un sistema de representación de conocimiento basado en la lógica descriptiva.

"una característica de estos lenguajes[1] es que, a diferencia de sus predecesores, están equipados con una semántica formal basada en la lógica. Otra característica distintiva es el énfasis en el razonamiento como servicio central: el razonamiento permite inferir conocimiento representado implícitamente del conocimiento que está explícitamente contenido en la base de conocimiento. La lógica descriptiva soporta patrones de inferencia que ocurren en muchas aplicaciones de sistemas de procesamiento de información inteligentes, y que son también usados por los humanos para estructurar y entender el mundo: clasificación de conceptos e individuos", Baader & Nutt (2003).

En la Figura 2.1 se muestra la arquitectura básica de un sistema de representación del conocimiento basado en la lógica descriptiva, normalmente formada por dos capas. La capa inferior realiza toda la lógica del sistema: se encarga tanto del mantenimiento de la base de conocimiento como de las funciones relacionadas con el razonamiento (todo ello basado en un *lenguaje de descripción* que define la semántica de cada elemento en la base de conocimiento). La capa superior es la cara visual de las aplicaciones, encargada de interactuar con la base de conocimiento y el razonador para devolver respuestas a los usuarios.

La base de conocimiento está formada por dos componentes: el TBOX y el ABOX. El TBOX contempla toda la *terminología*, o sea, el vocabulario de un dominio de aplicación en función de: 1) los *conceptos*, que denotan clases o conjuntos de individuos; 2) las *relaciones*, que denotan relaciones binarias entre los individuos; 3) un conjunto de descripciones complejas sobre este vocabulario (restringidos, por supuesto, por el lenguaje de descripción). El ABOX que contiene *aserciones* acerca de individuos nombrados en términos del vocabulario.

El lenguaje de descripción tiene un *modelo teórico-semántico* y es éste el que caracteriza al sistema de representación de conocimiento basado en la lógica. Gracias a él, todas las sentencias

[1] Se refiere a los lenguajes de descripción de la lógica descriptiva.

Constructor	**Sintaxis**	**Lenguaje**
Concepto atómico[a]	A	FL_0, FL^-, AL, S
Relación	R	FL_0, FL^-, AL, S
Intersección de conceptos[b]	$C \sqcap D$	FL_0, FL^-, AL, S
Restricción de valores	$\forall R.C$	FL_0, FL^-, AL, S
Cuantificador existencial limitado	$\exists R$	FL^-, AL, S
Universo	$\top$	FL^-, AL, S
Negación de átomo	$\neg A$	AL, S
Negación	$\neg C$	C, S
Unión	$C \sqcup D$	U, S
Restricción existencial	$\exists R.C$	E, S
Restricciones de cardinalidad[c]	$\leq nR$ ($\geq nR$)	N
Nominales	$a_1, ..., a_n$	O
Jerarquía de relaciones	$R \subseteq S$	H
Relaciones Inversas	R^-	I
Restricciones de cardinalidad cualificada[c]	$\leq nR.C$ ($\geq nR.C$)	Q
Restricción existencial sobre tipos de datos	$\exists u.\Phi$	(D)
Restricciones de valores sobre tipos de datos (t.d.)	$\forall u.\Phi$	(D)
Restricciones de cardinalidad cualificada sobre t.d.	$\geq nu.\Phi$ ($\geq nu.\Phi$)	(D)

Tabla 2.2: Principales constructores de la lógica descriptiva y las familias que caracterizan.

[a] Un concepto atómico es aquel que es básico para la construcción de otros conceptos.
[b] C y D representan conceptos, ya sean atómicos o no.
[c] También puede usarse la restricción de cardinalidad de igualdad, sustituyendo $\leq$ ($\geq$) por $=$.

en el TBOX y en el ABOX pueden ser formuladas en lógica de primer orden o una pequeña extensión de ésta.

A continuación veremos con más detalle cada una de las partes de esta capa inferior.

Lenguaje de descripción

El lenguaje de descripción de un sistema de representación del conocimiento basado en la lógica descriptiva, no es más que la especificación sintáctica y semántica del conjunto de constructores que pueden ser empleados en las definiciones de los conceptos.

La Tabla 2.2 (una actualización de la descrita en Gomez-Perez *et al.* (2003)) resume los principales constructores descritos en la literatura relacionándolos con la nomenclatura utilizada para designar a los lenguajes descriptivos. Es de especial interés la familia de los lenguajes *AL* (del inglés attribute languages, lenguajes atributivos), introducidos en Schmidt-Schauß & Smolka (1991) e identificados como el mínimo conjunto de constructores con interés práctico (los primeros ocho constructores de la tabla). Una descripción detallada de cada uno de los lenguajes y su significado puede encontrarse en Baader & Nutt (2003).

Nótese que este conjunto de constructores permite la definición de conceptos con un gran poder expresivo. Pueden, por ejemplo, ser definidos desde conceptos atómicos y relaciones, hasta conceptos complejos por medio de la unión o intersección de otros conceptos y especificación de la cardinalidad y rango de valores que puede tomar una relación (incluyendo el uso de los *tipos de datos*[1]).

Definición de la base de conocimiento

Como dijimos, la base de conocimiento se describe por medio de la terminología y las aserciones acerca de los individuos del mundo. En el caso más general, los axiomas terminológicos tienen la forma:

$$C \sqsubseteq D \; (R \sqsubseteq S) \text{ ó } C \equiv D \; (R \equiv S)$$

A modo de ejemplo, podemos definir la siguiente terminología:
$Mujer \equiv Persona \sqcap GeneroFemenino$
$Madre \equiv Mujer \sqcap \exists TieneHijo.Persona$
$MadreConMuchasHijas \equiv Madre \sqcap \geq 3 TieneHijo \sqcap \forall TieneHijo.Mujer$
$MadreJovenConMuchasHijas \equiv MadreConMuchasHijas \sqcap = 1 TieneEdad. \leq 20$

En este ejemplo, hay dos conceptos atómicos: *GeneroFemenino* y *Persona*, cuatro conceptos complejos: $Mujer$, $Madre$, $MadreConMuchasHijas$ y $MadreJovenConMuchasHijas$, y dos relaciones: $TieneHijo$ y $TieneEdad$. En la definición de $MadreConMuchasHijas$ se han empleado los constructores de restricción de cardinalidad y de valores para especificar que una *madre con muchas hijas* debe tener al menos 3 hijos y que todos tienen que ser mujeres.

Por último, el concepto de $MadreJovenConMuchasHijas$ añade a las características de una $MadreConMuchasHijas$ la de tener una edad menor de 20 años. Nótese que en este caso, ≤ 20 es una definición de un tipo de datos. Básicamente un *tipo de datos* no es más que un conjunto de elementos de igual tipo (como los números o cadenas de caracteres) y un conjunto de predicados y funciones que pueden realizarse sobre éstos. Así, en el ejemplo de $MadreJovenConMuchasHijas$ el predicado ≤ 20 nos permitió describir el tipo de datos de enteros menores o iguales a 20, en este caso asociado a la *edad*. Se ha empleado, además, el constructor $= nR (= nR.C)$ para especificar que exactamente hay que tener un valor referido al atributo edad.

La descripción del mundo se define por medio de las aserciones acerca de individuos. Para ello, se emplean relaciones unarias que asocian literales (nombres de individuos) a conceptos, $C(a)$, y

[1] Los tipos de datos (en inglés, datatypes) inicialmente no fueron incluidos en las especificaciones de los lenguajes descriptivos, pero su importancia en describir, por ejemplo, medidas de conceptos (“edad”, “peso”, “humedad”) ha obligado a la definición de extensiones que suplan esta carencia.

relaciones binarias que asocian parejas de literales entre si, $R(b,c)$; donde a, b y c representan dichos literales y R representa una relación.

Dada la terminología anterior, podemos tener la siguiente descripción del mundo:

$$\begin{array}{ll} Mujer(Ana) & Mujer(Maria) \\ TieneHijo(Maria,Rosa) & TieneHijo(Maria,Rafaela) \\ TieneHijo(Maria,Adela) & \end{array}$$

de la que puede deducirse que *Maria* es una mujer con muchas hijas (*Maria* que pertenece a los individuos del tipo *MadreConMuchasHijas*, aún cuando no está expresado explícitamente en la base de conocimiento) y de *Ana* no se sabe más que es una mujer.

La *interpretación*, $I = (\Delta^I, .^I)$, de la sintaxis de un lenguaje descriptivo permite definir una semántica formal para éste. Ella consiste en un conjunto no vacío Δ^I considerado el dominio de interpretación, y una función de interpretación, $.^I$, que asigna: 1) a cada concepto atómico, un subconjunto de Δ^I; 2) a cada relación R, una relación binaria en $\Delta^I \times \Delta^I$ y 3) a cada concepto descrito, el conjunto que resulta de definiciones inductivas que reflejan la semántica de las relaciones expresadas por los constructores. Por ejemplo: $\bot^I = \emptyset$, significa que no hay individuo alguno que pertenezca a $\bot$; y $\forall R.C = \{a \in \Delta^I | \forall b, (a,b) \in R^I, b \in C^I\}$ significa que el concepto $\forall R.C$ está compuesto por todos los individuos en Δ^I que se relacionan a través de R sólo con individuos que cumplan las características del concepto C.

La interpretación de toda la base de conocimiento, *Tarski-style interpretation*, $I = (\Delta^I, .^I)$ es la extensión de la interpretación anterior con la interpretación del ABOX. Así, un nombre de individuo, a, es asociado a un elemento $a^I \in \Delta^I$ y una aserción de relación $R(b,c)$ es asociada como $(b^I, c^I) \in R^I$.

Razonamiento

A continuación describiremos las tareas de razonamiento más importantes, las cuales están asociadas con la comprobación de propiedades en la base de conocimiento. Las cuatro primeras están relacionadas con el TBOX y las siguientes tienen que ver con al ABOX:

1. *Satisfacibilidad*: Un concepto C es satisfacible con respecto a una terminología T si existe una interpretación, I, en la cual C^I no es vacío. Desde el punto de vista lógico ello significa que un concepto que no puede ser ejemplificado con individuos carece de interés.

2. *Subsunción*: Un concepto C está subsumido por otro, D, con respecto a una terminología T si para cualquier interpretación I, $C^I \subseteq D^I$. Ello se describe como $C \sqsubseteq_T D$ o simplemente $C \sqsubseteq D$.

3. *Equivalencia*: Dos conceptos C y D son equivalentes con respecto a T, si para toda interpretación, $C^I \equiv D^I$. En este caso se escribe: $C \equiv_T D$ o simplemente $C \equiv D$.

4. *Disyunción*: Dos conceptos C y D son disjuntos con respecto a T, si para toda interpretación, $C^I \cap D^I = \emptyset$.

5. *Comprobación de aserción*: Una aserción está vinculada a un *ABOX*, si toda interpretación que satisface al *ABOX*, también satisface la aserción.

6. *Consistencia del ABOX con respecto a un TBOX*: Un ABOX, A, es consistente con respecto a un TBOX, T, si existe una interpretación que satisfaga tanto a T como a A.

La solución de cada uno de estos problemas, se deriva en su mayoría de teoremas de la lógica y su implementación consiste en la creación de algoritmos que verifiquen las hipótesis de dichos teoremas. Un estudio detallado de estos tipos de inferencias, así como de los algoritmos y teoremas sobre los que se sustentan son ampliamente expuestos en Baader & Nutt (2003).

Estos razonamientos permiten la comprobación de propiedades de la base de conocimiento durante su fase de desarrollo. También sirven de apoyo a los sistemas de desarrollo de bases de conocimiento, en tanto les posibilita crear por ejemplo, estructuras arbóreas para las jerarquías o señalar propiedades apenas éstas aparecen en la base de conocimiento.

Sin embargo, inferencias más complejas son requeridas en los sistemas de representación del conocimiento. Ellas están relacionadas con la formulación de demandas, tanto sobre la terminología como los individuos descritos. En el Capítulo 3 describimos algunas otras inferencias que pueden hacerse tanto sobre el TBOX como sobre el ABOX.

2.3.2 Otras extensiones de los lenguajes descriptivos

Las lógicas descriptivas han sido extendidas considerando diferentes aplicaciones. En Baader *et al.* (2003) aparece una magnífica disertación de las principales extensiones de los lenguajes descriptivos, a decir: restricciones de dominio; operadores modales, epistémicos y temporales; lógica de probabilidades y borrosa.

Tampoco puede dejarse de mencionar dos recientes trabajos: Horrocks & Sattler (2004) y Horrocks *et al.* (2006), que han sido enfocados a extender el lenguaje subyacente al OWL, SHOIN(D), con un conjunto de constructores que incrementan su expresividad. El nuevo lenguaje, SROIQ(D), considera la composición y propagación de relaciones, la definición de relaciones reflexivas, simétricas, transitivas, irreflexivas, disjuntas, una relación universal y el constructor $\exists R.self$ para permitir crear instancias que se relacionan consigo mismas. Además, permite la inclusión de relaciones negadas en los ABOX y las restricciones cualificadas.

2.4 Web Semántica

Sin lugar a dudas, la *World Wide Web*, (también llamada *web* o WWW), es el servicio más popular de Internet gracias a la capacidad provista por los *navegadores* para encontrar los recursos de información (*páginas web*) desde los *servidores web* y mostrarlos en las pantallas de los usuarios. Este servicio se sustenta en tres estándares: el *localizador uniforme de recursos* (URL, *uniform resource locator*), que especifica cómo asociar a cada recurso una "dirección" única donde encontrarlo; el *protocolo de transferencia de hipertexto* (HTTP, *hypertext transfer protocol*), que especifica cómo el navegador y el servidor intercambian datos de las peticiones y respuestas, y el *lenguaje de marcación de hipertexto* (HTML, *hypertext markup language*), un lenguaje de marcado que permite la presentación de la información textual y gráfica de los documentos de acuerdo a ciertas anotaciones.

Actualmente, la búsqueda de información no se concibe sin el empleo de los llamados *motores de búsqueda* de la web. A pesar de su popularidad (en gran parte porque son los únicos tipos de recursos disponible para este tipo de necesidades) las experiencias durante las búsquedas se vuelven, en muchos casos, frustrantes. Primero, por la imposibilidad de realizar una petición precisa de la información requerida; segundo, por el inmenso número de respuestas devueltas; y tercero, por la imprecisión de dichos resultados: gran parte de éstos son irrelevantes mientras las respuestas realmente relevantes o interesantes no son encontradas. La causa fundamental de tal insatisfacción se deriva precisamente de lo que le ha dado fuerza y popularidad a la web: el empleo de HTML para la codificación de los documentos. HTML permite la anotación de documentos, pero desafortunadamente, esta codificación carece de todo sentido para las computadoras, en tanto no está asociada a la semántica de los elementos anotados, sino a la forma de representación visual de éstos.

Una nueva forma de web, llamada *Web Semántica*, ha sido concebida para proveer recursos web cuya información pueda ser compartida y procesada tanto por herramientas automáticas como por usuarios humanos. De acuerdo a lo expuesto en Berners-Lee (2000b) la Web Semántica debe:

1. proveer una sintaxis común para descripciones comprensibles por la máquina,
2. establecer vocabularios comunes,
3. congeniarlos en un lenguaje lógico y
4. utilizar dicho lenguaje para el intercambio de pruebas.

Berners-Lee ideó, además, la arquitectura para su desarrollo. Ésta aparece representada en la Figura 2.2 y nos servirá de guía en el resto de esta sección. Nótese que en su base aparecen

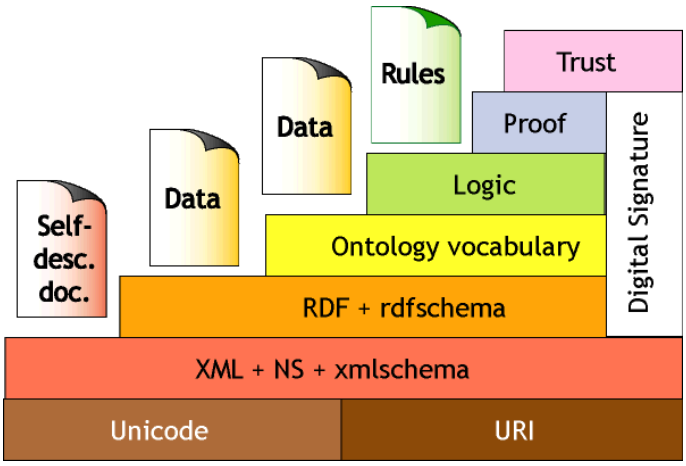


Figura 2.2: Arquitectura de la Web Semántica.

tanto los estándares disponibles para referirse a entidades: el *Unicode* (Unicode (1991-2006)) y los *identificadores de recursos únicos*, (URI, en inglés *uniform resource identifier*, Berners-Lee (1998)), una superclase de las *URL*, como las tecnologías web ya existentes como XML (en ingles, *extensible markup language*, Bray *et al.* (2000)) para definir la sintaxis de los recursos de la Web Semántica. Precisamente por esta capa, comenzaremos el recorrido.

2.4.1 Capa de sintaxis

XML provee un estándar para codificar los documentos y está rodeado por toda una familia de especificaciones. Para el contexto de la Web Semántica, dos especificaciones son las más importantes: XML Schema[1], Thompson *et al.* (2001), y XML NameSpace, Bray *et al.* (1999). La primera de estas especificaciones, permite definir *documentos XML válidos* proveyendo un conjunto de tipos de datos predefinidos y reglas para definir y derivar nuevos tipos de datos; la segunda, permite distinguir los nombres de diferentes aplicaciones, de manera que se puedan utilizar múltiples vocabularios heterogéneos dentro un mismo documento.

Un documento XML válido crea un árbol balanceado de elementos nombrados anidados que se corresponden con la definición provista por el XML Schema, y considerando el espacio de nombre (namespace) que se declara mediante el elemento: *xmlns*.

Es importante tener presente que XML Schema solamente especifica convenios sintácticos para el intercambio de documentos. Por tanto, no están diseñados para proveer descripción semántica acerca del área de conocimiento por ellos tratada. Actualmente, muchas aplicaciones utilizan XML como plataforma para el intercambio de datos, sin embargo, el polimorfismo de las posibles

[1] XML Schema es un superconjunto de los llamados Document Type Definition, DTD.

codificaciones sintácticas para representar los elementos[1] limita las posibilidades de las aplicaciones cuando éstas no conocen apriori el esquema de los datos descritos.

2.4.2 Capa de datos

La capa de datos intenta mejorar la interoperabilidad en la Web, especializando el modelo de datos XML por medio del RDF (*resource description framework*), Klyne & Carrol (2003). En esencia, RDF permite codificar sintácticamente grafos dirigidos y etiquetados, siendo por tanto dicho grafo el modelo de datos. La idea es construir sentencias acerca de los recursos en la forma: $(sujeto, predicado, objeto)$, lo cual se conoce como una *tripleta RDF*. El $sujeto$ es el recurso que se describe, el $predicado$ es un rasgo del recurso cuyo valor se expresa en el $objeto$. El poder de esta solución radica en que existe una única forma de expresar dicha sentencia, por tanto, un único árbol RDF puede ser descrito (lo cual difiere de las múltiples maneras en que podría expresarse en el árbol de un documento XML[2]).

Utilizando la sintaxis de RDF, se definen los RDF Schema, RDFS, Brickley & Guha (2003). Su intención, como la de los XML Schema respecto a los XML, es la de proveer las primitivas requeridas para describir el vocabulario usado en los modelos RDF particulares. La sintaxis de RDFS contiene primitivas para la definición de clases, relaciones de especialización en clases y propiedades, restricciones en los rangos y dominios de las propiedades. Además, permite ciclos en las jerarquías de especialización y la herencia de múltiple en la definición de clases. El lenguaje RDFS es un metamodelo cíclico, o sea, los elementos del lenguaje mismo son parte del vocabulario. Por ejemplo, una *rdfs:Class* es una instancia de ella misma y por tanto siempre está incluida en la extensión de una clase.

Desafortunadamente, "la especificación inicial de RDF no proveía ninguna semántica formal, lo cual condujo a diversas interpretaciones en sus datos, en algunos casos con diferencias significativas entre ellas", Volz (2004). Una especificación formal oficial apareció tres años más tarde, a partir de las propuestas en Hayes (2003) y que posteriormente se convirtieron en recomendaciones de la W3C, W3C (2002). Este modelo teórico de RDF es simplemente la asignación de semántica a los grafos RDF.

Sin embargo, la semántica de dicho modelo es incompleta, pues, por ejemplo, ignora todos los aspectos del significado de los URI (al tratarlos como nombres simples), y asigna menos significado formal a varias partes de los RDF y los RDFS que los esperados según las explicaciones provistas por las especificaciones de la sintaxis.

[1] Por ejemplo, los datos pueden ser codificados como atributos o como elementos de contenido.

[2] Una amena explicación de toda esta problemática puede encontrarse en: http://www.w3.org/DesignIssues/RDF-XML.

Por un lado, el metamodelo implícito en los RDF dificulta los procesos de inferencia, al no distinguir entre los elementos del lenguaje y otros objetos en el dominio del discurso (pues tanto las clases como las propiedades son también objetos del dominio) y al permitir que propiedades como *rdfs:SubClassOf* tengan un papel dual en la semántica: definen el lenguaje mismo, pero a su vez son usadas en la definición de los objetos. Por otro lado, diferentes autores han apuntado que la poca especificación semántica de algunos elementos de los RDF, conllevan a problemas más serios si se intentase extender el modelo del lenguaje con constructores más expresivos, Volz (2004).

Una alternativa a este modelo teórico, podría ser la traducción de RDF a una lógica formal bien definida. Aunque diversos autores han propuesto varias versiones alternativas para la definición de tal lógica, como en Conen & Klapsing (2000); Decker *et al.* (1998); Fikes & McGuinness (2001); Guha & Hayes (2003); Roo (2002), hasta el momento no se cuenta con un modelo teórico basado en la lógica aceptado por la comunidad.

2.4.3 Capa de ontología

En el año 2003 W3C declaró que aunque RDF Schema provee un lenguaje simple para describir el vocabulario de los datos RDF, se necesitaba relaciones más complejas entre las entidades, que permitieran, entre otras:

1. limitar las propiedades de las clases con respecto a su número y tipo,

2. definir un modelo de herencia de propiedades (y otras extensiones semánticas similares) bien definido,

3. inferir la clase de algún elemento para el cual se conocieran varias de sus propiedades.

Tras varios proyectos de investigación, la W3C escogió, en ese mismo año, la Lógica Descriptiva como la base lógica para un lenguaje expresivo para ontologías, y presentó *OWL* (Web Ontology Language), W3C (2004), como una recomendación para la especificación de ontologías, basada en el lenguaje DAML+OIL, Horrocks & Sattler (2001), y construida sobre la base sintáctica de RDF(S).

OWL se divide en tres partes: OWL Full, OWL DL y OWL Lite. OWL Full, contiene todos los constructores del lenguaje OWL y permite la combinación arbitraria de ellos. OWL DL es un subconjunto de OWL Full que contempla las siguientes restricciones: el vocabulario de OWL no puede ser empleado como identificadores para clases, propiedades e individuos; no son admitidas las construcciones cíclicas en las clases; los conjuntos de identificadores para clases, propiedades

e individuos deben ser disjuntos y por último, las restricciones de cardinalidad no pueden ser colocadas en propiedades transitivas.

OWL Lite, por su parte, es el subconjunto de OWL DL formado a partir de la aplicación de las siguientes restricciones: no pueden ser empleados los constructores de intersección, unión, complemento, restricción existencial, ni *hasValue*[1]; los valores de cardinalidad son limitados a 0 y 1 y por último, no pueden ser definidos objetos anónimos[2].

OWL Full es un lenguaje indecidible, por tanto por razones de complejidad sólo resultan interesantes OWL DL y OWL Lite. La complejidad de reconocer la satisfacibilidad de una base de conocimiento OWL DL es NExpTime, Horrocks & Patel-Schneider (2003). Desafortunadamente, a pesar de las restricciones impuestas a OWL Lite, la complejidad de su razonador no mejora sustancialmente y también está en el orden de ExpTime, Volz (2004). Ello ocurre, porque a pesar de las diferencias sintácticas, semánticamente OWL Lite y OWL DL pueden expresar prácticamente lo mismo, pues muchos constructores no permitidos en el primero pueden obtenerse a través de la combinación de otros.

En el anexo A se muestra una tabla donde aparecen las primitivas de OWL Lite, OWL DL y RDFS, así como su traducción al lenguaje de definición que se explica en la sección 2.3.1.

"Claramente, OWL no es la palabra final en el tema de los lenguajes ontológicos para la Web Semántica", Horrocks *et al.* (2003b), pues muchas características útiles para un lenguaje de ontología de la Web fueron identificadas entre los requerimientos de OWL, Hefflin *et al.* (2002), pero nunca han sido incorporadas al lenguaje final.

Algunas de las limitaciones de OWL son:

1. **Las ontologías definidas son monolíticas.** Ello significa que las ontologías OWL deben ser utilizadas como un todo, por tanto, no podría formarse una nueva por importación de subconjuntos de otras ontologías.

2. **Indefinición en las tareas de transformación de ontologías**: Cuando múltiples ontologías comiencen a desarrollarse, y se requieran realizar demandas complejas sobre los datos almacenados en la Web Semántica diferentes problemas (como la integración o alineación de ontologías) podrían surgir. Los estándares actuales, sin embargo, no consideran constructores para tales necesidades, aunque existen algunas propuestas para realizar este tipo de tareas como C-OWL, Bouquet *et al.* (2003).

[1] *hasValue* define el valor para todos los objetos que cumplen una determinada propiedad.

[2] Los objetos anónimos son definidos dentro de la descripción de otro objeto, o sea, no tienen identificador ni se asocian con otro objeto que no sea el que lo contiene.

3. **Tratabilidad**: Aunque varias optimizaciones para los problemas de razonamiento clásico en los TBOX son conocidas (aunque no llegan a resolver todos los casos de los problemas del TBOX), hasta el momento se desconocen algoritmos eficientes para el razonamiento en los ABOX Volz (2004).

4. **Poca disponibilidad de sistemas**: Hasta el 2004 sólo se conocían tres sistemas, FaCT (Horrocks *et al.* (1999)), Racer (Haarslev & Moller (2001)) y Cerebra (NetworkInferenceInc. (2003)), que proveían un limitado soporte pare el razonamiento con OWL. Todos estos sistemas son incompletos y no soportan todas las características del lenguaje. En el 2004, fue presentado el sistema Pellet[1], el primer razonador que soporta todas las características de OWL DL. Es un sistema de código abierto en Java y que permite el desarrollo de aplicaciones complejas que requieran modelización e inferencia basados en la lógica descriptiva.

5. **Entorno dinámico**: Actualmente la mayoría de las páginas en la Web son dinámicas, pero los datos que ellas exponen son mantenidos de manera estática en bases de datos. Para extender la web actual a la semántica, se podría exponer los datos en formatos comprensibles por los razonadores, como hace FaCT o Racer, ó emplear sistemas que accedan directamente a bases de datos relacionales, como hacen las bases de datos lógicas como XSB (Sagonas *et al.* (1994)), Ontobroker (Decker *et al.* (1999)) o CORAL (Ramakrishnan *et al.* (1994)) ó se podrían crear variantes de lenguajes de Programación Lógicos que sean implementados directamente sobre *SQL*.

A pesar de todas estas limitaciones, actualmente la Web Semántica se encuentra en un estado de crecimiento continuo, gracias a Swoogle se conoce existen al menos unas diez mil ontologías, más de un millón de documentos semánticos y unos trecientos millones de tripletas que mantienen relaciones entre conceptos y/o instancias.

2.4.4 Capas de lógica, reglas y *trust*

Tal como hemos visto en la sección anterior, actualmente, las ontologías consideran la parte lógica de las bases de conocimiento. Por tanto, que la llamada “capa lógica” debería verse como una capa vertical y ortogonal a las capas funcionales de la arquitectura. Ello permitiría considerarla como requerimiento de la Web Semántica, pero implementada a través de las capas funcionales.

Lenguajes como OWL permiten definir además del vocabulario ciertos axiomas para deducir nuevas informaciones desde la información explícita, pero no admiten la definición de reglas

[1] http://www.mindswap.org/2003/pellet/

generales sobre las propiedades. Tal capacidad debería ser provista por la capa de reglas; sin embargo, aún no hay consenso en cómo ella debe ser. A continuación enunciaremos brevemente las alternativas propuestas, a pesar de que ninguna ha recibido especial atención por la falta de una formalización semántica completa:

1. **Enfoque basado en XML**. Este enfoque define un vocabulario XML para crear varios tipos de programas lógicos y una base de conocimiento subyacente. El representante más importante de este enfoque es RuleML, Boley *et al.* (2001), que define una sintaxis para crear una jerarquía de lenguajes de reglas que permiten, a su vez, construir lenguajes cada vez más sofisticados. Hasta ahora, incluye 12 sublenguajes. Su principal limitación es su incompletitud parcial, en la parte sintáctica y total en la semántica.

2. **Enfoque basado en RDFS con soporte para OWL**. Este enfoque se basa en permitir formar reglas sobre la base de definiciones de los grafos RDF. Algunos de los sistemas, como Euler (Roo (2002)) y CWM (Berners-Lee (2000a)), utilizan la *Notación3* por su simplicidad para la escritura y comprensión de las descripciones. Otros, como el Racer, operan directamente con los conocidos grafos de las tripletas de RDF traduciendo éstos a cláusulas en la *lógica de Horn* y compilados como programas Prolog, que deben llamar a programas externos que permitan manejar los constructores de OWL que no puedan ser manipulados como cláusulas de Horn. Sin embargo, al igual que en el caso anterior, este enfoque no ha recibido especial atención por la falta de formalización semántica. A ello se une la posible indecibilidad de sus sistemas por la falta de pruebas de sus propiedades computacionales.

3. **Enfoque basado en OWL**. Las propuestas basadas en OWL, son más recientes y tratan de combinar OWL y reglas, como en Grosof *et al.* (2003) o extender el lenguaje OWL para adicionar a OWL reglas de cláusulas de Horn, definiendo el lenguaje SWRL, *Semantic Web Rule Language*, Horrocks *et al.* (2003a). Ambas soluciones corren el riesgo de trabajar sobre conjuntos de datos y propiedades indecidibles, lo que les impide asegurar la terminación de sus algoritmos.

La capa de reglas aún se encuentra en un estado muy incipiente, en gran parte porque capas anteriores han completado sus estándares muy recientemente. Lo mismo ocurre con la capa trust o de la verdad, de la que se desconocen trabajos de formalización y/o descripción detallada. El objetivo de esta capa es mantener y promover hacia almacenes de datos todo conocimiento que se reconozca como verdadero. Ella debe considerar los problemas relacionados con la autenticidad y autoría de los conocimientos.

2.4.5 Resumen

La comunidad de investigadores de la Web Semántica ha creado en pocos años una poderosa herramienta para el manejo de información, que permite a personas y computadoras, comprender y compartir conocimientos de manera uniforme y sin ambigüedades (a través de definiciones formales en el lenguaje OWL), así como realizar ciertas inferencias acerca de los conocimientos acumulados.

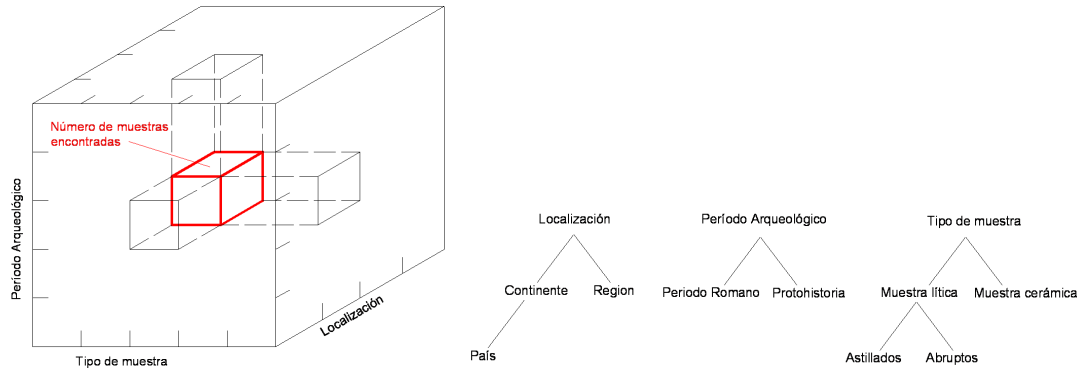
En estos momentos, el esfuerzo de la comunidad de la Web Semántica debe enfocarse entonces en dos puntos fundamentales: 1) en la población de la Web Semántica, o sea, en la creación de ontologías con suficiente información semántica que contemple descripciones de objetos del mundo y 2) en la definición, formalización y creación de las capas de reglas y de la verdad, las que finalmente harán posible el mantenimiento y descubrimiento continuo de nuevos conocimientos.

2.5 Modelos multidimensionales

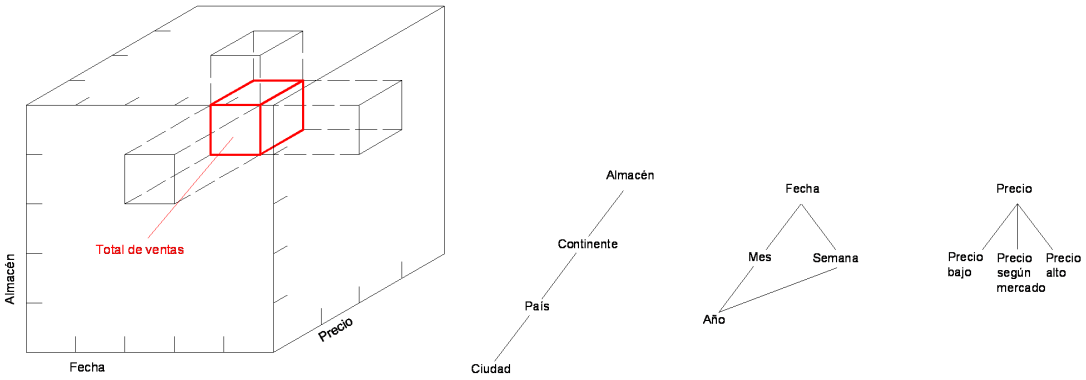
Las bases de datos surgieron a raíz de la necesidad de almacenar y analizar datos, fundamentalmente en el sector empresarial. Pero, a partir de 1960, cuando Codd propuso el modelo entidad-relación para el almacenamiento de datos, el boom de su empleo se extendió a todos los sectores de la sociedad. Más tarde, Internet permitió generar bases de datos actualizables y encuestables desde cualquier punto del planeta. Así, con el empleo cada vez más intensivo de las bases de datos (no sólo tradicionales bases de datos relacionales, sino también las orientadas a objeto), muchas organizaciones han logrado acumular grandes cantidades de información histórica. El sector empresarial enseguida notó la valía de la información acumulada: el análisis de las tendencias de ventas, cambios de gustos de las personas, ciclos de precios y fases económicas, comparación con productos competidores, etc., eran preguntas plausibles a responder.

La idea básica de los almacenes de datos es proveer la funcionalidad de generalización de datos, o sea, permitir abstraer enormes conjuntos de datos desde niveles conceptuales bajos a otros con mayor significación semántica. La Figura 2.3 representa gráficamente almacenes de datos con soporte para la generalización de datos, a través de los llamados *cubos multidimensionales*. Cada dimensión representa una entidad[1] a la que se le han asociado categorías lingüísticas que describen vistas diferentes de la información. El cubo multidimensional colecciona hechos que se asocian a las dimensiones. Cada hecho relaciona los valores de ciertas medidas de análisis a los valores de las dimensiones. Un primer ejemplo, de interés en aplicaciones arqueólogicas, permite analizar la relación entre la riqueza de muestras encontradas pertenecientes a diferentes períodos

[1] Atributo, desde el punto de vista de las bases de datos.



(a) Análisis de muestras de yacimientos arqueológicos



(b) Ventas de una cadena de almacenes

Figura 2.3: Ejemplos de modelos multidimensionales.

y la localización donde éstas fueron halladas. El segundo ejemplo, muestra una aplicación para un analista económico que revisa las ventas obtenidas considerando diferentes tipos de productos, sus precios y fechas de venta. Los ejemplos de la figura se han limitado a tres dimensiones para permitir la visualización. Por supuesto, los sistemas de manipulación de cubos multidimensionales, no imponen tal restricción.

Los pasos para permitir la generalización de datos en almacenes con grandes volúmenes de información fueron introducidos en 1991, Cai *et al.* (1991), empleando el enfoque de *inducción orientada al atributo*. La tarea se realiza examinando los diferentes valores de cada dimensión, generalizando dichos valores hasta el nivel semántico deseado y realizando operaciones de agregación de ciertas medidas de interés para aquellas descripciones que resultan idénticas después de la generalización.

Unos años más tarde, en 1993, Codd introdujo el concepto de Online Analitical Processing (OLAP), Codd *et al.* (1993), una tecnología para el análisis y navegación de datos multidimensio-

nales, que se basa en realizar cálculos *off-line* antes del procesamiento analítico para incrementar la velocidad de cálculo. Aunque realmente no hay barreras distintivas entre la inducción orientada al atributo y las técnicas OLAP (pues conducen a los mismos resultados y tanto en uno u otro enfoque pueden emplearse cálculos *on* y *off-line* para acelerar la velocidad de respuesta), a partir de 1993, el primero de estos enfoques se vio totalmente desplazado por el segundo, de manera que los procesos de generalización de datos modelados desde entonces, tienden a considerar mayoritariamente los preceptos dados por Codd.

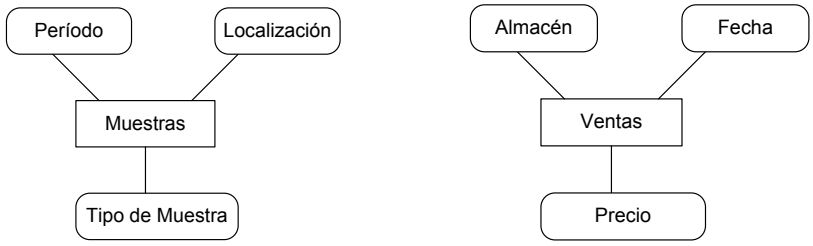
Numerosos modelos OLAP han sido descritos en la literatura: los primeros estaban enfocados a las bases de datos relacionales, poco después surgieron modelos para bases de datos orientadas a objetos y por último, aprovechando las bondades de XML como repositorio de datos, numerosos trabajos han sido desarrollados para permitir el intercambio y la recuperación de la información almacenada en formato XML. En las siguientes subsecciones se describe brevemente cada uno de éstos enfoques. Una información más completa, así como modelos que permiten generalizar los diferentes modelos y terminologías, puede verse en Franconi & Kamble (2004); Franconi & Sattler (1999) y Abelló *et al.* (2001)[1].

2.5.1 Modelos multidimensionales basados en BD relacionales

En Li & Wang (1996) se introduce, por primera vez una formalización de un modelo multidimensional basado en elementos relacionales, donde las dimensiones son modeladas anotando los atributos del modelo relacional con nombres de dimensiones; los cubos se definen como productos cartesianos de dimensiones y las medidas son asociadas a cada celda del cubo. Así mismo se definen las bases de datos multidimensionales como colecciones de cubos multidimensionales y se presenta un álgebra para formular demandas y obtener vistas de dichas bases de datos.

Por su parte, en Gyssens & Lakshmanan (1997) se presenta otra formalización donde las bases de datos multidimensionales son consideradas como composiciones de tablas formando un esquema de estrellas (como se muestra en la Figura 2.4), y las jerarquías de los atributos (dimensiones) son modeladas a través de dependencias funcionales en los atributos. La popularidad de este modelo se debe a la estrecha vinculación entre esta formalización y los conceptos del modelo entidad-relación (E/R), lo que le ha merecido además la implementación en importantes sistemas de gestión de bases de datos relacionales como Microsoft SQL Server y Oracle.

[1] Estos trabajos son particularmente interesantes pues han intentado homogeneizar las aportaciones más importantes y crear un marco de trabajo común para la definición de modelos multidimensionales, aunque ninguno de ellos han tenido suficiente apoyo en la comunidad que les permita estandarizar las definiciones y operaciones subyacentes a los modelos multidimensionales.



(a) Análisis de muestras de yacimientos arqueológicos

(b) Ventas de una cadena de almacenes

Figura 2.4: Ejemplos de esquemas en estrella para modelos multidimensionales.

En Cabibbo & Torlone (1997, 1998a,b) un modelo de datos multidimensional es propuesto basado en la noción de dimensiones y F-tablas. Las dimensiones son categorías parcialmente ordenadas que representan los niveles semánticos de interés. Las F-tablas son repositorios para hechos funcionalmente dependientes de las dimensiones y los datos son caracterizados por un conjunto de funciones que asocian instancias de un nivel a otro de una dimensión. Los autores cualifican el modelo como lógico, dado que describe la percepción del usuario de los cubos multidimensionales, o sea, la lógica subyacente a los datos tratados.

Otras aproximaciones pueden verse en Tryfona *et al.* (1999) y Moody & Kortink (2000). Particularmente importantes son los trabajos de Agrawal *et al.* (1997); Pedersen (2000); Pedersen & Jensen (1998, 1999), que describen álgebras muy completas para el desarrollo de modelos multidimensionales basados en el modelo relacional.

2.5.2 Modelos multidimensionales basados en BD orientadas a objeto

En Buzydlowski *et al.* (1998) es introducido el modelo O3LAP bajo la óptica del modelado orientado a objetos y una implementación física de éste modelo es presentada, en contraposición a las existentes: ROLAP (para bases de datos relacionales) y MOLAP (para bases de datos multidimensionales). La idea básica es la definición de clases de dimensiones y clases de hechos, con el mismo basamento teórico de la modelación orientada a objetos y su implementación en bases de datos de naturaleza orientada a objetos.

Las ideas del modelo multidimensional tradicional son extiendas al formalismo orientado a objetos de UML (Unified Modeling Language, un lenguaje que permite la modelación de datos basado en el paradigma orientado a objetos) en Binh & Tjoa (2001); Nguyen *et al.* (2000); Trujillo & Palomar (1998); Trujillo *et al.* (2000, 2001) y Abelló (2002) aparecen formalizaciones matemáticas que permite definir elegantemente los operadores del álgebra del modelado multidimensional considerando las características del modelado orientado a objetos, las que son: necesi-

dad de clasificación, instanciación, especialización, generalización, agregación, descomposición, intercambio de mensajes, empleo de relaciones derivables y dinamismo de los datos.

2.5.3 XML y los modelos multidimensionales

La sección 2.4 dio una pincelada de qué es XML y su influencia en las estandarizaciones para la Web Semántica. En esta subsección se discute la revolución que ha suscitado XML en el mundo de las bases de datos.

XML fue diseñado para que cumpliera objetivos pragmáticos muy claros, entre otros servir de sostén para el intercambio de datos en Internet y entre aplicaciones y permitir la creación y procesamiento de documentos XML de manera rápida. Además, los documentos XML debían contar con una sintaxis formal, concisa, legible y razonablemente clara. Estos preceptos permitieron la creación de un lenguaje particularmente cómodo para describir información de manera rápida y flexible. La popularidad de XML hizo que en poco tiempo la web comenzara a contar también con muchos documentos XML y que el sector de las bases de datos comenzara a emplear este formato para representar información semi-estructurada.

Dos trabajos particularmente interesantes en este sentido son Xymele (Xyleme (2001)) y Whoweda (Whoweda (1997)), los que comparten el objetivo de mantener repositorios de datos de la Web. Xymele fue un proyecto ambicioso que permitía encuestar documentos XML de la web utilizando sus DTD (predecesor de XML Schema) como descriptores de la información contenida en ellos. La metodología puede resumirse de la siguiente manera: por medio de un conjunto de *crawlers* se recuperan de la web documentos XML, y su estructura es almacenada por piezas en sistemas de gestión de bases de datos tradicionales y/o cadenas de bytes en dispositivos de almacenamiento secundarios. Los documentos son agrupados por temas y para cada agrupamiento se infiere un DTD abstracto y un emparejamiento con el DTD original. Por último, los usuarios del sistema pueden realizar consultas empleando el DTD abstracto y obtener los documentos que satisfacen con el patrón dado.

Whoweda es otro proyecto cuya finalidad es recuperar documentos de la web, y emplear los metadatos de estos documentos (contenido, estructura, hiperenlaces) en las operaciones de recuperación por demanda. Para ello, los documentos en la web son mantenidos en tablas-web, donde las tuplas representan las relaciones de hiperenlaces entre los documentos. Los usuarios del sistema pueden preguntar por ciertas condiciones en los metadatos y el sistema devuelve una porción de la web que satisface dichas restricciones. El sistema desarrollado está basado en una definición formal del modelo de datos, un álgebra para la representación y manejo de documentos web, la definición de la estructura de almacenamiento físico y la manipulación del versionado de los datos.

En otro sentido, trabajos como Golfarelli *et al.* (2001); Pokorn (2001) y Jensen *et al.* (2001), han sido desarrollados para permitir el diseño de bases de datos multidimensionales basados en colecciones de documentos XML. La estrategia común de estos trabajos es emplear los DTD o los XML Schema de los documentos para inferir las dimensiones y los cubos multidimensionales. Las estructuras inferidas son implementadas en algún sistema multidimensional tradicional, el que finalmente permite realizar análisis acerca de los documentos.

Una filosofía basada exclusivamente en datos XML, que permite procesar grandes volúmenes de información a la manera de OLAP pero sin necesidad de transformar las fuentes de datos en modelos multidimensionales, ha sido expuesta en los trabajos Pedersen *et al.* (2002, 2004). Los argumentos para el empleo de tal enfoque están fundamentados en el dinamismo de los datos, la imposibilidad de reducir los tiempos de integración física de datos provenientes desde diferentes fuentes y la posibilidad de emplear XML como un repositorio de datos. En Pedersen *et al.* (2002) y Pedersen *et al.* (2004) se describe un sistema que ejecuta operadores OLAP sobre documentos XML.

No menos interesantes son otras aplicaciones de XML para las bases de datos multidimensionales. En particular, pueden mencionarse: el empleo de XML como lenguaje de intercambio de datos multidimensionales, como en Hümmer *et al.* (2003); Trujillo *et al.* (2004); los repositorios de datos distribuidos, como en Mangisengi *et al.* (2001); Nguyen *et al.* (2001); Tseng & Chen (2005) y Foster & Kesselman (1998); y las recientes aplicaciones asociadas al análisis y recuperación de documentos que emplean modelos multidimensionales, para mantener y recuperar eficientemente los documentos en la web (como alternativa a los tradicionales índices documentales de los sistemas de recuperación de información), como se expone en Lee *et al.* (2002, 2003); McCabe *et al.* (2000).

2.6 Web Semántica y modelos multidimensionales

La comunidad investigadora de la Web Semántica, desde hace varios años ha comenzado a dar sus primeros pasos con el objetivo de unificar la lógica subyacente en la Web Semántica con el análisis provisto en los modelos multidimensionales.

En Hacid & Sattler (1997, 1998) se describen los resultados más significativos del proyecto DWQ (Data Warehouse Quality), en el que sus autores exponen toda una formalización para el modelado multidimensional basado en DL, incluso para definir los cubos multidimensionales. Para ello, sus autores proponen extender los constructores tradicionales de DL para que los nuevos objetos puedan incluir en sus descripciones operaciones de agregabilidad, y en segundo lugar, defi-

nen las operaciones tradicionales de modelo multidimensional como operaciones de razonamiento en la lógica descriptiva.

El lenguaje definido es una extensión del ALC(D), con un constructor que captura las dependencias funcionales (introducido en Borgida & Weddell (1997)) y un conjunto de operadores sobre los cubos especificados a nivel extensional. Las potencialidades, a nivel semántico, del modelo propuesto constituye en sí mismo un resultado muy importante en aras de la unificación de los dos campos de conocimiento. Sin embargo estos trabajos, dejan abiertos importantes problemas de razonamiento como la satisfacibilidad y la subsunción.

Una idea similar también es propuesto en van Zyl *et al.* (1998), pero en este trabajo sólo se explica teóricamente qué ofrecer a usuarios de almacenes de datos orientados al DL y no han sido publicados resultados posteriores.

Un trabajo de Baader, Baader & Sattler (2003), es el primero en examinar la dificultad de realizar tareas de razonamiento sobre este tipo de lenguajes descriptivos con operadores de agregabilidad. En él se demuestra la indecidibilidad del lenguaje $FL_0(\Sigma)$ con los operadores *min*, *max* y *suma*, así como la necesidad de relajar las potencialidades del lenguaje descriptivo hasta $ALC(D^{min}_{max})$ y $EL(\Sigma)$ para obtener lenguajes decidibles. Por último, se expone un algoritmo tableau que decide la satisfiabilidad de conceptos de este último lenguaje y su extensión a conceptos $ALC(\Sigma^{-})$.

Una alternativa diferente ha sido considerada en Ismailova *et al.* (2000), en el que se describe un sistema que permite generar vistas de bases de datos basadas en DL. Para ello se describe una interfaz visual que permite definir objetos en DL. Entonces, el sistema es capaz de convertir estos objetos DL en objetos de algún lenguaje OO, asociándoles eventos que permiten realizar inferencias, como la deducción de la subsunción, a partir de las definiciones de atributos y métodos que genera la interfaz (aunque no se explica qué mecanismos utilizar para el razonamiento). La propuesta es muy ambiciosa y hasta el momento los autores no han publicado resultados posteriores de esta aproximación.

A diferencia de los trabajos anteriormente mencionados, en el Capítulo 5 se expone un modelo para el análisis de instancias ontológicas que permite analizar la información de una o varias ontologías, sin restringir las actuales prestaciones del lenguaje provisto por DL. Para ello, en lugar de definir un nuevo lenguaje de descripción con operadores de agregabilidad, se crean meta-ontologías que enriquecen con información de análisis a las ontologías en DL. Esta información de análisis permitirá realizar las agregaciones[1] como normalmente se realiza en el modelo multidimensional.

[1] En el Capítulo 3 se describen como *funciones de combinación*.

2.7 Conclusiones

Cuanto más se estudia la problemática de la construcción de Web Semántica, más se comprende cuánto trabajo hay aún por hacer. La Web Semántica impone grandes retos: primero porque debe alimentarse fundamentalmente de la dinámica y poco organizada web actual, y luego, porque la formalización de los conocimientos y el razonamiento sobre éstos son tareas poco plausibles a desarrollar en un plazo medianamente corto.

La web actual contiene billones de páginas con información en lenguaje natural. La conversión de esta información en entidades de razonamiento (instancias ontológicas) es una tarea sumamente difícil. Tal como fue explicado en la sección 2.2 a pesar del empleo de recursos externos que apoyen la tarea, los resultados obtenidos quedan muy lejos de los deseables. Otros recursos externos y nuevas alternativas a las propuestas de los investigadores del PLN deben ser estudiados si se desea poblar más ágilmente la Web Semántica. En este sentido, la presente tesis emplea las propias formalizaciones del conocimiento en aras de recuperar entidades de razonamiento disponibles en la web, tal como podrá verse en el Capítulo 4.

A pesar de estos obstáculos, muchas decisiones cruciales ya han sido tomadas, y se cuenta con todo un conjunto de estándares que darán uniformidad sintáctica y fundamentación semántica a todos los conocimientos descritos en la Web Semántica. Así, por ejemplo, la lógica descriptiva ha sido escogida como basamento teórico para la descripción de los objetos de la web, y RDF y OWL para su expresión sintáctica.

La acertada selección de la lógica descriptiva como base de toda la arquitectura de la Web Semántica, impulsa su propia evolución, pues las descripciones conceptuales en DL (aún con basamento en la lógica matemática) son muy cercanas a modelos de datos ampliamente aceptados como UML y E/R. Ello no solamente permite la reutilización de muchos modelos existentes, sino también engranar la Web Semántica a los sistemas y procedimientos tradicionales (como es el caso del trabajo en Ismailova *et al.* (2000) donde se intenta traducir objetos DL en objetos de algún lenguaje de programación que contenga dicho paradigma; ó el empleo de DL durante el diseño, evolución y optimización de demandas en bases de datos como es propuesto en Calvanese *et al.* (1998)) o al menos, condicionar a que tales sistemas y procedimientos comiencen a surgir.

En este sentido, en la sección 2.6 se han descrito las primeras aproximaciones que acercan el mundo de los almacenes de datos al de la Web Semántica y en el Capítulo 5 se propone otro método que permite analizar los datos de la Web Semántica. Estos son los primeros pasos para comenzar a pensar qué podría contener la capa de reglas de la Web Semántica.

Capítulo 3

Fundamentación teórica

Este capítulo describe los fundamentos teóricos en los que se basan las propuestas del presente trabajo. Antes de dar paso a las formalizaciones matemáticas, permítase que se justifiquen las dos tareas de razonamiento más importantes tratadas en este capítulo. La primera es el cálculo de caminos de agregación entre conceptos. La segunda, aborda la problemática del razonamiento sobre instancias ontológicas.

Intuitivamente, un camino de agregación de un concepto C a un concepto C' es una composición de relaciones que permite arribar a C', partiendo desde el concepto C. Su cálculo abre las puertas para muchas aplicaciones, no solamente en el campo del razonamiento lógico, sino además en campos tan dispares como las Bases de Datos, Minería de Datos, Procesamiento del Lenguaje Natural, etc. Algunas posibles aplicaciones son:

- *Población de la Web Semántica*: A pesar de los muchos años que el Procesamiento del Lenguaje Natural ocupa a los investigadores, aún no existe un método robusto que permita a la computadora "entender" un texto. El empleo de ontologías, junto la deducción de las relaciones entre conceptos puede permitir poblar la Web Semántica de manera semi-automática. Por ejemplo, si tenemos una ontología sobre arqueología y un texto en el que se habla de un yacimiento y luego de una pieza cerámica, pudieran deducirse que: (1) el yacimiento es de tipo arqueológico, (2) la pieza pertenece al yacimiento, (3) el lugar puede estar relacionado con un asentamiento poblacional. Toda esta información podría ser añadida a la ontología poblándola con instancias que anteriormente no existían. Una metodología que permite tal proceso será expuesta en el Capítulo 4.

- *Cálculo de distancia entre conceptos*: Las distancias que separan a los conceptos, calculadas lo mismo por su distancia jerárquica que por la distancia en los caminos de agregación,

pueden ser importantes parámetros en las funciones de similitud entre objetos (en particular, entre conceptos). Ellas han probado su utilidad en los procesos de agrupamiento de documentos (como en Wang & Hodges (2006)) y en la clasificación de conceptos y objetos (como se propone en Srinivasan *et al*. (2005)), así como en la desambiguación de palabras, tal como será mostrado en los siguientes capítulos.

- *Creación y reconstrucción de vistas y ontologías personalizadas*: Cuando la Web Semántica sea una realidad, un proceso importante será permitir a los usuarios seleccionar y "mezclar" ontologías (o partes de ellas) de forma más o menos arbitraria para crear una visión personalizada de la realidad descrita en la web. Ello sería imposible si no se pudiese reconocer "todo" lo relacionado con una clase. Gracias a los caminos de agregación el constructor de una vista no sólo puede recuperar las definiciones asociadas a las clases de interés, sino también podar ciertas relaciones asociadas a las clases que no le resultan útiles en la nueva perspectiva de dominio que se está creando.

- *Creación de bases de datos y almacenes de datos consistentes asociados a ontologías*: A pesar de la buena acogida de la Web Semántica, este paradigma difícilmente sustituirá a herramientas tradicionales de análisis y almacenamiento de datos; pero sí será muy importante la reutilización de toda información disponible en cualquiera de los entornos. Ya es conocido que una ontología puede ser "traducida" en una base de datos, Vysniauskas & Nemuraite (2006). De manera similar que en la aplicación anteriormente descrita, la reconstrucción del esquema conceptual de la base de datos requiere el reconocimiento, provisto por los caminos de agregación, de la información asociada a cada clase. En los almacenes de datos el uso de los caminos de agregación es determinante, en tanto permiten definir cómo realizar las operaciones de generalización que son expresadas a través de la agregación. Por ejemplo, las relaciones entre *continente*, *país*, *provincia* y *localidad* son expresadas por lo general, como agregaciones del tipo "parte de". Un modelo multidimensional basado en la lógica descriptiva debe proveer operaciones que permitan moverse entre los diferentes niveles de conceptualización, tanto los definidos por las jerarquías conceptuales (a través del constructor $\sqsubseteq$) como los definidos por las agregaciones entre objetos (que se identifican por medio de los caminos de agregación). En el Capítulo 5 se describe una metodología que permite crear almacenes de datos a partir de ontologías concretas.

Si bien la recuperación de todos los caminos de agregación, como es nuestro objetivo, no ha sido tratada en la literatura, los algoritmos de razonamiento acerca del TBOX disponibles en la literatura, de una u otra manera permiten recuperar una parte de dicha información.

El razonamiento acerca del TBOX de las bases de conocimiento de la lógica descriptiva ha recibido mucha atención por parte de los investigadores. Desafortunadamente, no puede decirse lo mismo del ABOX de las lógicas descriptivas. Las tareas de razonamiento relacionadas al ABOX se circunscriben a operadores que permiten recuperar toda la información asociada a una instancia y reconocer las clases a las que pertenece, y definir conceptos y recuperar las instancias que se le asocian. Sin embargo, en un futuro muy cercano esas operaciones serán insuficientes para resolver problemas de análisis en los datos de la Web Semántica. Hasta el momento no se ha descrito un conjunto de operadores que permitan:

1. *describir una instancia según una clase particular*, o sea, consolidar toda la información disponible de una instancia para generar la vista de la instancia lo más completa posible según una clase de interés. Nótese que tanto las generalizaciones (abstracciones) como especializaciones pueden ser igualmente interesantes de tratar. Las operaciones de unificación, especialización y abstracción definidas en la sección 3.4 dan solución a este tema.

2. *comparar instancias*; puede ser interesante ver las semejanzas y diferencias entre dos instancias no necesariamente relacionadas por alguna jerarquía conceptual. Tal como se muestra en la sección 3.4, las operaciones de diferencia y diferencia simétrica son útiles para proveer esta información.

Como ya fue anunciado en el primer párrafo de este capítulo, las siguientes secciones detallan formalmente la base teórica y algunas de las contribuciones del presente trabajo. En la sección 3.1 se define el espacio de la lógica matemática con el que se trabaja. La sección 3.2 explica el algoritmo tableau, descrito en Horrocks & Sattler (2005a,b) y cuya estructura de datos es empleada en la definición de nuevas operaciones de razonamiento en las secciones 3.3 y 3.4. La primera de estas secciones muestra algunos resultados relacionados con las propiedades de las clases mientras la segunda analiza interesantes operaciones entre instancias ontológicas.

3.1 Preliminares

La semántica de la primera propuesta de OWL, hecha en febrero del 2004, abarca el lenguaje de descripción $SHOIN(D)$, Sirin *et al.* (2006). Pero su evolución continúa con un desarrollo vertiginoso, tanto es así que en Octubre del 2006 ha aparecido una extensión de esta recomendación: OWL 1.1, cuyo lenguaje descriptivo se corresponde a SROIQ(D), Grau (2006). Los sistemas Pellet, Protege4Alpha, y FACT++ ya ha incorporado las nuevas características, y están ahora en fase de prueba, W3C (2006).

Nuestra propuesta supone la definición ontológica del conocimiento de un dominio específico empleando una base de conocimiento en un lenguaje de la lógica descriptiva SHIOQ(D), punto medio entre los lenguajes SHOIN(D) y SROIQ(D) y que ya está incorporado en Pellet de manera estable.

Definición 3.1. Una ontología, $\mathcal{O}$, basada en la lógica descriptiva SHIOQ(D) es aquella cuyos elementos pueden ser generados a partir de la siguiente gramática:

BaseConoc → TBOX ABOX
TBOX → DefsConcepts AxiomasR
DefsConcepts → AxiomaC | AxiomaC DefsConcepts

AxiomaC → C $\sqsubseteq$ C | C $\equiv$ C

C → A | o | $\top$ | $\bot$ | $\neg$ C | C $\sqcup$ C | C $\sqcap$ C | RestricsR
RestricsR → RestR RestricsR | RestR
RestR → $\exists$ R.C | $\exists$ u.Φ | $\forall$ R.C | $\forall$ u.Φ | $\leq$ n R.C | $\leq$ n u.Φ | $\geq$ n R.C |
$\geq$ n **u**.Φ

AxiomasR → AxiomaR AxiomasR | ε
AxiomaR → R = R^- | R $\equiv$ R | $\leq 1R$ | $\leq 1R^-$ | R $\sqsubseteq$ R | R_+

ABOX → C(a) ABOX | R(a,b) ABOX | ε

R^- denota la relación inversa de R, y $\equiv$ la equivalencia semántica entre los elementos relacionados; $\leq 1R$ ($\leq 1R^-$) permite expresar la relación funcional R (la inversa funcional R^-), en tanto $R \sqsubseteq R$ y R_+ expresan la jerarquía entre relaciones y las relaciones transitivas, respectivamente. $R \sqsubseteq^* R$ representa la clausura transitiva en la jerarquía entre relaciones. El conjunto de conceptos SHOIQ(D) (o simplemente conceptos) de la ontología es denotado por N_C y está formado por los elementos especificados a partir de los tokens A (conceptos atómicos), o (conceptos nominales) y C. Destacaremos el conjunto de los conceptos nominales N_o y el de los atómicos N_A; $N_o, N_A \subseteq N_C$. Los conceptos atómicos permiten dar nombre a clases de individuos que cumplen un conjunto de propiedades. Un concepto nominal, o, define la clase de individuos formado sólo por el individuo o[1]. El conjunto N_R, formado por los tokens de R y u se llama conjunto de relaciones, entre los que se destacan las relaciones transitivas y funcionales que denotaremos por N_{R_+} y N_{R_1}, respectivamente, $N_{R_+}, N_{R_1} \subseteq N_R$. El rango de las relaciones R es alguna clase en N_C, en tanto el rango para u es algún tipo de datos.

[1]Normalmente los conceptos nominales se emplean para individuos que alcanzan connotación de conceptos, por ejemplo organizaciones, países, etc.

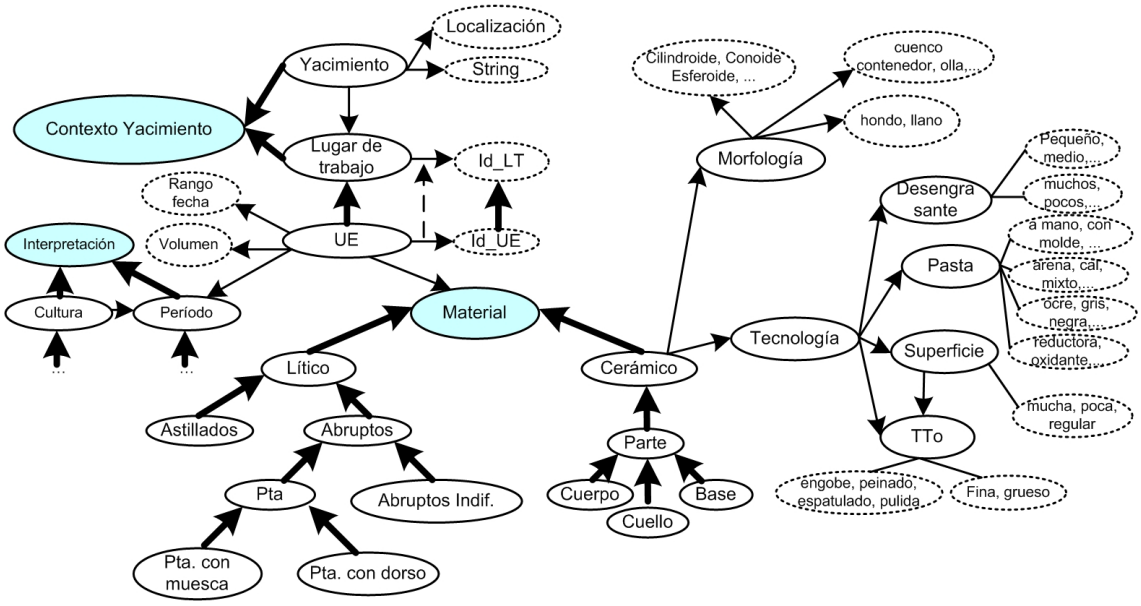


Figura 3.1: Fragmento de la ontología de arqueología. Los conceptos son representados en elipses, las elipses sombreadas se corresponden a clases raíces de diferentes jerarquías de la ontología y las de lineas discontinuas representan tipos de datos. En líneas gruesas aparecen las relaciones jerárquicas entre los conceptos, las finas se corresponden a relaciones de agregación y las discontinuas, representan jerarquías entre relaciones.

Los axiomas de relaciones $\mathcal{R}$, el TBox, $\mathcal{T}$, y el Abox, $\mathcal{A}$ de $\mathcal{O}$, son especificados por medio de las reglas de producción *AxiomasR*, *TBOX* y *ABOX*, respectivamente. Si $\mathcal{A}=\emptyset$ diremos que $\mathcal{O}$ es abstracta, en caso contrario $\mathcal{O}$ es concreta.

Como ejemplo, una representación gráfica de un fragmento de la ontología de arqueología aparece en la Figura 3.1.

Definición 3.2. Una interpretación de una ontología abstracta definida por $I = (\Delta^I, .^I)$ consiste en un conjunto no vacío Δ^I y una función $.^I$ que asocia:

- cada relación R a un subconjunto de $\Delta^I \times \Delta^I$ ($R^I \subseteq \Delta^I \times \Delta^I$) donde se satisface que:
  - si $R \in N_R$, $\langle x,y\rangle \in R^I$ ssi $\langle y,x\rangle \in R^{-I}$
  - si $R \in N_{R_+}$, $\langle x,y\rangle \in R^I \wedge \langle y,z\rangle \in R^I$ entonces $\langle x,z\rangle \in R^I$.
- cada relación u a un subconjunto de $\Delta^I \times \Phi^D$ ($u^I \in \Delta^I \times \Phi^D$) siendo $I^D = (\Phi^D, .^D)$ a su vez la interpretación de los tipos de datos, cuya semántica viene dada por el conjunto no vacío Φ^D, $\Phi^D \cap \Delta^I = \emptyset$ y la función $.^D$ que asocia cada tipo de datos Φ a un subconjunto fijo (estrictamente definido) de Φ^D. Llamaremos *dato* a cualquier elemento del universo $\mathcal{U}$ $= \Delta^I \cup \Phi^D$.

- cada concepto C a un subconjunto de Δ^I ($C^I \subseteq \Delta^I$) donde se cumple que:
  - $(C \sqcap D)^I = C^I \cap D^I$
  - $(C \sqcup D)^I = C^I \cup D^I$
  - $\neg C^I = \Delta^I \setminus C^I$
  - $(\exists R.C)^I = \{x \in \Delta^I | \exists y, \langle x,y \rangle \in R^I\}$
  - $(\exists u.\Phi)^I = \{x \in \Delta^I | \exists \phi \in \Phi, .^D(\Phi) \subseteq \Phi^D, \langle x,\phi \rangle \in u^I\}$
  - $(\forall R.C)^I = \{x \in \Delta^I | \forall y, \langle x,y \rangle \in R^I, y \in C^I\}$
  - $(\forall u.\Phi)^I = \{x \in \Delta^I | \forall \phi, \langle x,\phi \rangle \in u^I, \phi \in \Phi, .^D(\Phi) \subseteq \Phi^D\}$
  - $(\leq nR.C)^I = (\exists R.C)^I$ si $|(\exists R.C)^I| \leq n$
  - $(\leq nu.\Phi)^I = (\exists u.\Phi)^I$ si $|(\exists u.\Phi)^I| \leq n$
  - $(\geq nR.C)^I = (\exists R.C)^I$ si $|(\exists R.C)^I| \geq n$
  - $(\leq nu.\Phi)^I = (\exists u.\Phi)^I$ si $|(\exists u.\Phi)^I| \geq n$

I satisface la jerarquía de relaciones de $\mathcal{R}$ si $\forall R \sqsubseteq S \in \mathcal{R}$, $R^I \subseteq S^I$.

Emplearemos la nomenclatura $I \Vdash x$ para describir que x se deduce de (se modela mediante) I, ó que I modela a x.

De acuerdo a las definiciones anteriores el *Abox* de una ontología concreta $\mathcal{O}$ define una *interpretación* para la ontología abstracta que puede obtenerse de $\mathcal{O}$.

Tal como fue descrito en el capítulo anterior, existen algunas tareas básicas de razonamiento. Entre ellas la verificación de la satisfacibilidad de un concepto y la verificación de la consistencia de la base de conocimiento, formalmente definidas como sigue.

Definición 3.3. Un concepto C de una ontología $\mathcal{O}$ es satisfacible si existe una interpretación I en la cual $C^I \neq \emptyset$.

Definición 3.4. Un ABox, $\mathcal{A}$, es consistente con respecto a un TBox, $\mathcal{T}$, si existe una interpretación modelo de A que satisfaga tanto a $\mathcal{T}$ como a $\mathcal{A}$.

Por el momento, baste pensar que una interpretación modelo no es más que una interpretación donde cada tipo de dato está representado por el conjunto de elementos de dicho tipo, en lugar por que un valor particular de éste. En la sección 3.2, volveremos a retomar esta discusión.

Un trabajo reciente, Horrocks & Sattler (2005a,b), define la solución a dichos problemas para lenguajes SHIOQ(D), mediante la definición de un algoritmo de tableau de complejidad del orden NExpTime-complete (Tobies (2000)), y que ya ha sido implementado en Pellet. Podría desilusionar el elevado orden de complejidad del algoritmo, pero sus propios autores expresan que: "... está

claro que, en el peor de los casos, el procedimiento se comportará muy mal, o sea, en práctica, no terminará. Sin embargo, el algoritmo diseñado se comportará bien en muchos casos típicamente encontrados... ", Horrocks & Sattler (2005b). Además, la inexistencia de otros métodos que traten el problema del razonamiento para lenguajes SHIOQ(D), han hecho que el método de solución descrito y el sistema Pellet que lo implementa sean los recursos más empleados en las tareas de razonamiento de la Web Semántica.

Tal como se ha explicado, en el presente trabajo se ha identificado la necesidad de reconocer lo que hemos denominado caminos de agregabilidad entre conceptos y algunas operaciones que permiten razonar sobre las instancias. Afortunadamente las salidas de los algoritmos de tableau, permiten recuperar los caminos de agregabilidad entre conceptos. En la siguiente sección se describe el algoritmo de tableau para gramáticas SHIOQ(D), especificado en Horrocks & Sattler (2005a,b), lo cual permitirá en la sección 3.3 emplear la estructura de datos resultante en aras de obtener los caminos de agregabilidad entre conceptos. La sección 3.4 se dedica a explicar las operaciones de especialización, agregación, diferencia y diferencia simétrica entre instancias de un ABox.

3.2 Algoritmo tableau

En teoría de pruebas, un tableau semántico o un algoritmo tableau es un procedimiento de decisión para sentencias lógicas, pruebas de formulas de primer orden y un método para probar la satisfacibilidad de formulas en varias lógicas (como la descriptiva). Éste consiste en formar un grafo que descompone y explica una formula lógica (un concepto, como se ha utilizado en este trabajo).

En el caso particular de las lógicas descriptivas el algoritmo tableau parte de las descripciones de un concepto particular (axiomas del tipo *AxiomaC*) y genera un grafo en el que cada nodo representa un concepto y los arcos, las relaciones que asocian dichos conceptos. La generación del grafo es guiado por un conjunto de reglas que permiten interpretar un concepto semánticamente. Mientras se genera el grafo se revisa que no exista ninguna contradicción en sus nodos y si ello ocurre, se descarta el grafo actual y se continua revisando otras interpretaciones (generando otros grafos). De esta forma, si al menos aparece una interpretación sin contradicciones (se completa un grafo), puede entonces asegurarse que el concepto es satisfacible.

Como es habitual con los algoritmos de tableau, se asumirá que todos los conceptos están en forma normal negada (NNF, normal negation form). Para ello cada concepto $C \in N_C$ puede ser transformado empleando las leyes de De Morgan y las dualidades entre las restricciones de *existencia* y *universalidad* y entre las restricciones de *al menos* y *a lo sumo*. Estas reglas se resumen de la siguiente manera:

$$\neg\neg C = C \qquad \neg(C \sqcap D) = \neg C \sqcup \neg D$$
$$\neg(C \sqcup D) = \neg C \sqcap \neg D \qquad \neg\exists R.C = \forall R.\neg C$$
$$\neg\exists u.\Phi = \forall u.\neg\Phi \qquad \neg\forall R.C = \exists R.\neg C$$
$$\neg\forall u.\Phi = \exists R.\neg\Phi \qquad \neg \leq nR.C = \geq (n+1)R.C$$
$$\neg \leq nu.\Phi = \geq (n+1)u.\Phi \qquad \neg \geq (n+1)R.C = \leq nR.C$$
$$\neg \geq (n+1)u.\Phi = \leq nu.\Phi \qquad \neg(\geq 0R.C) = \top$$

Emplearemos la nomenclatura $\neg C$ para denotar la forma normal negada de $\neg C$.

También se definen $sub(C)$ como el conjunto de todos los subconceptos de C y $cl(C,\mathcal{R})$, conjunto de sub-conceptos relevantes de C relacionados con los axiomas de relaciones $\mathcal{R}$, como:

$$cl(C,\mathcal{R}) = sub(C) \cup \{\neg D | D \in sub(C)\} \cup \\ \cup\{\forall S.C' | \{\forall R.C', \neg\forall R.C'\} \cap sub(C) \neq \emptyset \text{ y } S \text{ ocurre en } \mathcal{R} \text{ o en } C\}$$

Emplearemos $cl(C)$ en lugar de $cl(C,\mathcal{R})$ en los casos en que se sobrentienda cuál es el conjunto de axiomas de relaciones $\mathcal{R}$ al que se refiere.

Definición 3.5. Sea $\mathcal{R}$ un conjunto de axiomas de relaciones, C un concepto SHOIQ(D) en *NNF*. Un árbol de completamiento para C con respecto a $\mathcal{R}$ es un grafo dirigido $G = (V,E,L,\dot{\neq})$ donde cada nodo $x \in V$ está etiquetado con un conjunto $L(x) \subseteq cl(C) \cup N_o \cup \{\leq mR.C | \leq nR.C \in cl(C) \wedge m \leq n\}$ y cada arco $\langle x,y\rangle \in E$ está etiquetado con un conjunto de relaciones en $L(\langle x,y\rangle)$, posiblemente también inversos, ocurriendo en C ó en $\mathcal{R}$. Adicionalmente se mantiene la traza de desigualdades entre los nodos del grafo mediante la relación simétrica $\dot{\neq}$.

Si el arco $\langle x,y\rangle \in E$, entonces y es llamado *sucesor* de x y x es llamado *predecesor* de y; *ancestro* y *descendiente* son las cláusulas transitivas de predecesor y sucesor respectivamente. Un nodo y es llamado *R-sucesor* de un nodo x si para alguna R' con $R' \sqsubseteq^* R$, $R' \in L(\langle x,y\rangle)$. Al nodo x se le llama *R-predecesor* de y si y es un R-sucesor de x. Un nodo y es *vecino* (*R-vecino*) de un nodo x si y es un sucesor (R-sucesor) de x o si x es un sucesor (R^--sucesor) de y. Se define además el conjunto de los R-vecinos con concepto C de un nodo x, como $R^G(x,C) = \{y | y \text{ es un R-vecino de } x \text{ y } C \in L(y)\}$.

En G hay dos tipos de nodos: los nodos nominales y los bloqueables. Un nodo x es nominal si $L(x)$ contiene un nominal. Un nominal $o \in N_o$ se dice nuevo en G si no hay nodo en G que contenga a o en su etiqueta. Un nodo que no es nominal es bloqueable[1]. Un nodo x es bloqueado directamente por y si tiene ancestros x', y e y' tales que:

[1] Intuitivamente un nodo es bloqueable si tiene un ancestro con su misma definición conceptual que ha sido formado a partir del mismo conjunto de relaciones que le forman él. Utilizando la nomenclatura anterior esto significa que desde y se llega un nodo x con igual descripción que él y utilizando para ello un arco con igual etiqueta que la del arco a través del cual se arriba a y. De ahí que, x e y representan un mismo concepto y por tanto, no es necesario extender a

- x es un sucesor de x' e y es un sucesor de y',
- y, x y todos los nodos del camino desde y hasta x son bloqueables,
- $L(x) = L(y)$, $L(x') = L(y')$ y
- $L(\langle x, x' \rangle) = L(\langle y', y \rangle)$.

Un nodo es bloqueado si es bloqueado directamente por otro ó es bloqueable y su predecesor está bloqueado (en cuyo caso se dice que está indirectamente bloqueado).

Un R-vecino y de un nodo x es seguro si:

- x es bloqueable ó
- x es nominal y y no está bloqueado.

Sean $o_1, ..., o_l$ los nominales de C. El nivel de un nodo nominal x se define por:

$$nivel(x) = \begin{cases} 0, & \text{si } \exists o_i, o_i \in L(x),\ 1 \leq i \leq l \\ min\{nivel(y) | \text{y es vecino de x}\} + 1, & \text{en otro caso} \end{cases}$$

El algoritmo de tableau para un concepto C y un conjunto de axiomas de relaciones $\mathcal{R}$, comienza inicializando el bosque de grafos de completamiento de C, $\mathcal{F}$, con el conjunto unitario formado por el grafo de completamiento inicial de C, $G_{inic}(C)$, $G_{inic}(C) = (V_{inic}, \emptyset, L_{inic}, \emptyset)$, siendo $V_{inic} = \{r_0, r_1, ..., r_l\}$, $o_1, ..., o_l$ los nominales de C, $L_{inic}(r_0) = \{C\}$, $L_{inic}(r_i) = \{o_i\}$ para $1 \leq i \leq l$. $\mathcal{F}$ es expandido aplicando repetidamente las reglas de expansión en la Figura 3.3 (según el orden en la Figura 3.2) a las definiciones asociadas a C mediante la reglas de producción *AxiomaC* (ver definición 3.1) y deteniendo la extensión de algún grafo de completamiento si ocurre un clash (ver definición 3.6) en él o si no puede ser aplicada ninguna otra regla (en este caso se dice que el grafo de completamiento está completo). El algoritmo tableau devuelve que un concepto C es satisfacible con respecto a $\mathcal{R}$ si en $\mathcal{F}$ se encuentra un grafo de completamiento que no contiene un clash. El anexo B describe la función *Merge* que se emplea en las reglas de expansión $R \leq$ y *Ro*. En las siguientes secciones emplearemos la nomenclatura r_{0_G} para denotar al nodo raíz no nominal de un grafo de completamiento G, o sea, $r_{0_G} \in V_{inic}$ y $C \in L_{inic}(r_{0_G})$.

Definición 3.6. Un grafo G contiene un clash si:

1. para algún concepto $A \in N_A$ y un nodo x de G, $\{A, \neg A\} \subseteq L(x)$
2. para un nodo x de G, $\leq nR.C \in L(x)$ y existen $n+1$ R-vecinos $y_0, ..., y_n$ de x con $C \in L(y_i)$, $y_i \dot{\neq} y_j, 0 \leq i < j \leq n$, ó

x y debe ser bloqueado. Un nodo nominal no se bloquea, porque idealmente, podría existir otro objeto (nominal) que reemplaza al nominal del nodo, dando lugar a otro nodo nominal que representa una instancia diferente a la del nodo nominal original.

1. *Ro* es aplicada con la más alta prioridad
2. *Ro*? y $R \leq$ son aplicadas primero a los nodos nominales de bajos niveles. En caso de que ambas reglas puedan ser aplicadas, *Ro*? debe ser aplicada primero.
3. El resto de las reglas son aplicadas con la más baja prioridad.

Figura 3.2: Orden de aplicación de las reglas de expansión de tableau.

3. para algún $o \in N_o$, $x \dot{\neq} y$, $o \in L(x) \cap L(y)$.

El bosque de modelos de C, denotado por $\mathcal{F}_m(C)$, es el subconjunto de grafos de completamiento de C que no contienen clash, $\mathcal{F}_m(C) \subseteq \mathcal{F}$.

Antes de continuar es importante comprender también intuitivamente el significado de los grafos de completamiento y del bosque de modelos de una clase C. Los primeros son una representación en forma de grafo de una instancia de clase C, o sea, una instancia particular de clase C podría describirse, si a cada nodo cuya etiqueta representa un tipo de datos se le asocia un elemento particular de ese tipo. Los grafos de completamiento con clash no representan individuos de clase C, son precisamente combinaciones de relaciones que conllevan a conflictos. Por ello, un bosque de modelos de una clase C detalla posibles patrones de descripción de individuos de clase C, y como consecuencia todo *Abox* consistente de una ontología cuenta con un conjunto de bosques de modelos (de las clases de su $Tbox$) que le describen[1].

Cálculo de jerarquía de conceptos

El cálculo de la jerarquía de conceptos es tratado empleando las definiciones de la interpretación de una inclusión entre conceptos, o sea, $C \sqsubseteq D$ si $C^I \subseteq D^I$.

Tal como vimos, el algoritmo de tableau anteriormente descrito permite obtener el "conjunto de modelos genéricos" del TBox. De manera que determinando la inclusión entre cada par de modelos de conceptos (nótese que es suficiente con los elementos en $L(r_{0_G})$) puede determinarse la jerarquía o equivalencia entre éstos.

3.3 Caminos de agregación entre conceptos

La presente sección describe cómo calcular los caminos de agregación entre conceptos a partir de los grafos de completamiento de una ontología. Para nuestros objetivos no solamente es interesante el conjunto (más bien, la lista) de relaciones que conforman un camino de agregación, sino además los conceptos a través de los cuales se forma dicho nexo.

[1] Es posible que ahora se comprenda mejor la definición 3.4.

Sea $G = (V, E, L, \dot{\neq}) \in \mathcal{F}$, entonces:

$R\sqcap$: Si $C_1 \sqcap C_2 \in L(x)$, x no está bloqueada indirectamente y $\{C_1, C_2\} \nsubseteq L(x)$ entonces: $G = (V, E, L', \dot{\neq})$ donde $L' = L \setminus L(x) \cup \{\langle x, L(x) \cup \{C_1, C_2\}\rangle\}$)

$R\sqcup$: Si $C_1 \sqcup C_2 \in L(x)$, x no está bloqueada y $\{C_1, C_2\} \cap L(x) = \emptyset$ entonces: $\mathcal{F} = \mathcal{F} \setminus G \cup_{i \in \{1,2\}} G_i$, $G_i = (V, E, L \cup \{\langle x, \{C_i\}\rangle\}, \dot{\neq})$, $i \in 1, 2$.

$R\exists$: Si $\exists R.C \in L(x)$, x no está bloqueada y x no tiene un R-vecino seguro de y con $C \in L(y)$ entonces: $G = (V \cup \{y\}, E \cup \{\langle x, y\rangle\}, L \cup \{\langle \langle x, y\rangle, \{R\}\rangle, \langle y, \{C\}\rangle\}, \dot{\neq})$.

$R\forall$: Si $\forall R.C \in L(x)$, x no está bloqueada indirectamente y x tiene un R-vecino de y con $C \notin L(y)$ entonces: $G = (V, E, L \cup \{\langle y, \{C\}\rangle\}, \dot{\neq})$.

$R\forall_+$: Si $\forall R.C \in L(x)$, x no está bloqueada indirectamente, existe algún R' con R'_+ y $R \sqsubseteq^* R$ y existe R-vecino y de x con $\forall R.C \notin L(y)$ entonces: $G = (V, E, L', \dot{\neq})$, $L' = L \setminus L(y) \cup \{\langle y, L(y) \cup \{\forall R.C\}\rangle\}$.

$R \leq ?$: Si $\leq R.C \in L(x)$, x no está bloqueada indirectamente, existe R-vecino y de x con $\{C, \neg C\} \cap L(y) = \emptyset$ entonces: $\mathcal{F} = \mathcal{F} \setminus G \cup_{D \in \{C, \neg C\}} G_D$, $G_D = (V, E, L'_D, \dot{\neq})$, $L'_D = L \setminus L(y) \cup \{\langle y, L(y) \cup D\rangle\}$, $D \in \{C, \neg C\}$.

$R \geq$: Si $\geq R.C \in L(x)$, x no está bloqueada, y no hay n R-vecinos seguros de x, $y_1, ..., y_n$, con $C \in L(y_i)$ y $y_i \dot{\neq} y_j$ para $1 \leq i < j \leq n$, entonces: $G = (V \cup \{y_1, ..., y_n\}, E, L \cup \{\langle y_i, \{C\}\rangle, \langle \langle x, y_i\rangle, \{R\}\rangle\}, \dot{\neq} \cup \{\langle y_i, y_j\rangle\})$ para $1 \leq i < j \leq n$.

$R \leq$: Si $\leq R.C \in L(z)$, z no está bloqueada indirectamente, y $\left|R^G(z, C)\right| > n$ y existen dos R-vecinos x, y de z con $C \in L(x) \cap L(y)$ y no hay n R-vecinos $y_1, ..., y_n$ de x con $C \in L(y_i)$ y $y_i \dot{\neq} y_j$ para $1 \leq i < j \leq n$, $x \dot{\neq} y$, entonces:

Si x es un nodo nominal *entonces* $Mezclar(y, x)$

Sino Si y es un nodo nominal ó un ancestro de x *entonces* $Mezclar(x, y)$

Sino $Mezclar(y, x)$

Ro: Si para algún $o \in N_o$ existen dos nodos x, y con $o \in L(x) \cap L(y)$ y no se tiene que $x \dot{\neq} y$ entonces: $Mezclar(x, y)$.

Ro?: Si 1. $(\leq nR.C) \in L(x)$, x es un nodo nominal, y existe un R-vecino bloqueable y de x tal que $C \in L(y)$ y x es un ancestro de y y

2. $\neg\exists m, 1 \leq m < n, \leq mR.C \in L(x)$ y hay m-nominales R-vecinos de x, $z_1, ..., z_m$, con $C \in L(z_i)$ y $z_i \dot{\neq} z_j$, $\forall 1 \leq i < j \leq m$, entonces:

a) Seleccionar m o_i nuevos en G, $m \leq n$,

b) $G = (V \cup \{y_1, ..., y_m\}, E \cup \{\langle x, y_i\rangle\}, L', \dot{\neq}')$, donde:

$L' = L \cup \{\langle x, L(x) \cup \{\leq mR.C\}\rangle\} \cup \{\langle \langle x, y_i\rangle, \{R\}\rangle, \langle y_i, \{C, o_i\}\rangle \,|\, 1 \leq i \leq m\}$ y

$\dot{\neq}' = \dot{\neq} \cup \{\langle y_i, y_j\rangle \,|\, 1 \leq i < j \leq m\}$.

Figura 3.3: Reglas de expansión del algoritmo tableau.

Formalmente un camino de agregación desde un concepto C a un concepto $C^{'}$ puede definirse de la siguiente manera.

Definición 3.7. Se dice que $\oplus_{1\leq i\leq n}\langle RS_i, C_i\rangle$ es un camino de agregación del concepto C al concepto $C^{'}$ de la ontología $\mathcal{O}$ si $C_1 = C$, $C_n = C^{'}$, y existe una interpretación $I = (\Delta^I, .^I)$ de $\mathcal{O}$ tal que $\exists x_i \in \Delta^I, 0 \leq i \leq n$, tales que $x_0 \in C^I$ y $x_i \in C_i^I, \langle x_{i-1}, x_i\rangle \in R_{i_j}^I$, para $1 \leq i \leq n$, $R_{i_j} \in RS_i$. Se dirá también que C_j pertenece al rango del seudo-camino $\oplus_{1\leq i\leq j-1}\langle RS_i, C_i\rangle \oplus RS_j$, representado por: $range(\oplus_{1\leq i\leq j-1}\langle RS_j, C_i\rangle \oplus RS_j), 1 \leq j \leq n$.

No es difícil notar que empleando los grafos de completamiento generados por el algoritmo de tableau para un concepto C, pudieran recuperarse los caminos de agregación desde C, dado que los nodos de dichos grafos representan datos del universo $\mathcal{U}$ y las etiquetas de sus arcos las relaciones entre dichos datos. Pero es necesario considerar los siguientes hechos.

Sea $G = (V, E, L, \dot{\neq}) \in \mathcal{F}_m(C)$, entonces:

1. En la etiqueta de cada nodo $x \in V$, aparece un subconjunto de $cl(C)$, sin embargo, la aplicación de las reglas no mantiene de manera explícita, en $L(x)$, el concepto más específico al que pertenece el dato representado por x.

2. No todas las relaciones R asociadas a un nodo x, generan un R-sucesor de x.

3. Se desconoce la satisfacibilidad C cuando se especializa la clase de algún *sucesor* de r_{0_G}[1].

Cada uno de estos puntos son analizados a continuación con más atención.

Deducción del concepto asociado a cada nodo

La etiqueta de cada nuevo nodo es inicializada con un único concepto, excepto en la regla *Ro*?, cuyo objetivo es garantizar la generación de vecinos nominales diferentes para aquellos nodos nominales que tienen como sucesores tanto un nodo bloqueable como nodos nominales que comparten todos un mismo concepto a través de la misma relación. En este caso los nuevos nominales creados por aplicación de *Ro*?, conjunto que denotaremos por $N_{o'}$, no pertenecen a N_C[2] y no forman parte del concepto ontológico asociado a cada uno de los nodos formados, al que denotaremos por $C_d(x)$.

Definición 3.8. Sea x un nodo de un grafo de completamiento $G = (V, E, L, \dot{\neq})$ formado a partir de la aplicación de las reglas de expansión de tableau. Llamaremos concepto deducido del nodo x al concepto más específico asociado a dicho nodo, y será denotado por $C_d(x)$.

[1] Recuérdese que había sido asignado el nombre de r_0 al nodo raíz no nominal de G, tal que $C \in L(r_0)$.

[2] Estos nominales en $N_{o'}$ pueden ser vistos como denominaciones alternativas a la instancia representada en los nodos.

Teorema 3.1. Sea x un nodo de un grafo de completamiento $G = (V, E, L, \dot{\neq})$ formado a partir de laaplicación de las reglas de expansión de tableau, entonces $C_d(x) = \bigsqcap_{\substack{C \in L(x), \\ C \notin N_{o'}}} C$.

Sea I la interpretación en G. Las reglas que invalidan la posibilidad de tener un único concepto general asociado a un nodo son $R\forall$, $R \leq ?$ y los procesos de mezcla. Analicemos cada caso:

- $R\forall$: Si es aplicable esta regla, entonces $\exists y \in C_d(y)^I$, pero por definición de $(\forall R.C)^I$, $y \in C^I$ por lo tanto $y \in (C_d(y) \sqcap C)^I$. De lo que puede concluirse que después de aplicar $R\forall, C_d(y) = C_d(y) \sqcap C$.

- $R \leq ?$: Si es aplicable esta regla al nodo x, entonces $\exists y \in C_d(y)^I$, pero por definición de $(\leq nR.C)^I$ el número de R-vecinos de x de tipo C es limitado a n. Por tanto, se revisa si cada caso $y \in C^I$ ó $y \notin C^I$ (en grafos de completamiento diferentes) genera un clash. Análogamente al caso anterior, se deduce $C_d(y) = C_d(y) \sqcap D$, en el grafo de completamiento G_D, que considera $D \in \{C, \neg C\}$.

- Mezcla: El algoritmo de mezcla se produce en los casos donde los nodos a mezclar representan la misma instancia. Por tanto, si se realiza $Mezcla(y,x)$ se tiene que $x \in (C_d(x))^I \wedge x \in (C_d(y))^I$ entonces el nodo actualizado $x \in (C_d(x) \sqcap C_d(y))^I$. De ello se deduce que después de aplicar la mezcla, $C_d(x) = C_d(x) \sqcap C_d(y)$. ■

Aplicando el teorema anterior durante la expansión de las reglas, podemos obtener un grafo de completamiento en el que se puede asociar a cada nodo x su concepto $C_d(x)$. Exactamente, ello significa adicionar a G la relación C_d y realizar la actualización de C_d cada vez que se actualice de $L(x)$ de acuerdo al teorema (excepto en el caso que analizaremos en el punto siguiente). En lo sucesivo consideraremos $G = (V, E, L, \dot{\neq}, C_d)$.

No expansión de nodo por inexistencia de reglas $R\exists$ y $R \geq$

Las reglas $R\exists$ y $R \geq$ permiten expandir un nodo hacia conceptos en $\mathcal{O}$ que hasta el momento no se relacionan a él, pero ello no ocurre con las reglas $R\forall$ y $R \leq$. Por tanto, si existe un nodo x tal que $\{\forall R.C, \leq R.C\} \cap L(x) \neq \emptyset$ y $\{\exists R.\top, \geq nR.\top\} \cap L(x) = \emptyset$[1] entonces $\neg\exists \langle x,y \rangle \in E$ tal que $R \in L(\langle x,y \rangle)$. Sin embargo, tal expansión podría provocarse con el empleo del siguiente lema.

Lema 3.1. Todas las relaciones R asociadas a un nodo x generan un R-sucesor de x, sin alterar el resultado de la satisfacibilidad de C, si se actualiza el valor de $L(x)$, pero no el de $C_d(x)$, de la siguiente manera[1]:

$$L(x) = L(x) \cup \{\exists R.C | \{\forall R.C, \leq nR.C\} \cap L(x) \neq \emptyset \wedge \{\exists R.\top, \geq nR.\top\} \cap L(x) = \emptyset\}$$

[1] En este caso $\top$ ha sido usado para designar cualquier concepto.

Sea $G = (V, E, L, \dot{\neq}, C_d) \in F$, entonces:

R∃*forzado*: Si $\{\forall R.C, \leq R.C\} \cap L(x) \neq \emptyset$, $\{\exists R.\top, \geq nR.\top\} \cap L(x) = \emptyset$ y x no está bloqueado, entonces:
$G = (V, E, L', \dot{\neq}, C_d)$, $L' = L \setminus L(x) \cup \{L(x) \cup \{\exists R.C\}\}$.

Rclash?: Si G contiene un clash en una rama forzada, eliminar dicha rama de G.

Figura 3.4: Reglas de creación y eliminación de ramas forzadas.

y se ignoran los clash generados por nodos en las ramas forzadas a partir de *x*. Las ramas forzadas a partir de *x* son aquellas generadas dada la aplicación de *R*∃ para los conceptos $\exists R.C$ agregados a $L(x)$ por el presente lema.

Nótese que este lema obliga a la aparición de los R-sucesores "ocultos" de *x*, dada aplicación de *R*∃ para los conceptos $\exists R.C$ agregados a $L(x)$ por el lema anterior, pero sin alterar al valor de $C_d(x)$ ni el de la satisfacibilidad de *C*. Ambas son consecuencia del propio lema que dice de manera explícita "se actualiza $L(x)$ pero no $C_d(x)$" y "se ignoran los clash generados por nodos en las ramas forzadas a partir de *x*".

Definición 3.9. Llamaremos grafos de completamiento con ramas forzadas aquellos grafos de completamiento $G = (V, E, L, \dot{\neq}, C_d)$ en los que cada nodo *x* contiene al menos un R-sucesor para cada relación *R* en L(x). Análogamente llamaremos bosque de modelos con ramas forzadas de *C* al subconjunto de grafos de completamiento con ramas forzadas de *C* que no contienen clash.

Los grafos de completamiento con ramas forzadas se forman agregando las reglas que se introducen en la Figura 3.4 a las de expansión de tableau. Nótese que si una rama forzada contiene un clash, no genera clash para el grafo pero tampoco puede considerarse ningún camino de agregación que le contemple, por tanto dicha rama debe ser eliminada del grafo de completamiento, de ahí la inclusión de la regla *Rclash*?.

Por último, el empleo de las reglas *R*∃*forzado* y *Rclash*? permiten observar las siguientes propiedades en los grafos de completamiento.

Propriedad 3.1. Sea $\mathcal{F}_m$ el bosque de modelos con ramas forzadas de un concepto *C* y sea $G = (V, E, L, \dot{\neq}, C_d) \in \mathcal{F}_m$, $x \in V$, un nodo de G. El conjunto de relaciones directamente asociadas al nodo *x*, denotado por $Rs(x)$, es:

$$\bigcup_{\langle x,y \rangle \in E} L(\langle x, y \rangle)$$

Propriedad 3.2. Sea $\mathcal{F}_m$ el bosque de modelos con ramas forzadas de un concepto *C*. El conjunto de relaciones directamente asociadas a dicha clase *C*, denotado por $Rs(C)$, es definido por:

$$\bigcup_{G \in \mathcal{F}_m} Rs(r_{0_G})$$

La primera propiedad permite recuperar las relaciones asociadas directamente a algún nodo del grafo de completamiento. La segunda, permite recuperar las relaciones asociadas directamente a una clase. Nótese que esta segunda propiedad es particularmente importante, pues posibilita la conversión de la descripción conceptual en DL a la descripción conceptual en frames, ya que se reconoce para cada concepto las relaciones que directamente se le asocian, y para cada uno de ellos se puede recuperar el conjunto de conceptos de llegada empleando el valor de las etiquetas de los nodos en L.

Además, el hecho de contar con los modelos de cada clase, permite revisar las semejanzas y diferencias entre ellos. Ello es particularmente interesante cuando se trata de pares de clases que guardan una relación jerárquica, ya que tal comparación permite descubrir si es posible obtener vistas diferentes de una misma instancia, obviando las relaciones no comunes. O sea, si es posible "pensar" que una instancia de clase más general puede ser también instancia de una clase más específica[1]. El objetivo de la siguiente definición es proveer un modelo para la especialización una instancia.

Definición 3.10. Sean $G = (V, E, L, \dot{\neq}, C_d) \in \mathcal{F}_m$ y $G' = (V', E', L', \dot{\neq}', C'_d) \in \mathcal{F}_m$, grafos de completamiento con ramas forzadas de los conceptos C y C', respectivamente, $C' \sqsubseteq C$. Existe un modelo de especialización de C a C' a través de los grafos G y G', si es posible definir la función de especialización del nodo r_{0_G} a $r_{0_{G'}}$. La función de especialización de un nodo x a un nodo x', $f_{e_{n_{x\to x'}}}$: $N_R \times N_C \to N_R \times N_C \times f_{e_{n_{y\to y'}}}$ se define por el conjunto de quintetos $\left\langle R, C_d(x), R', C_d(y'), f_{e_{n_{y\to y'}}} \right\rangle$ $\forall R \in Rs(x)$ siendo R, R', y, y' y $f_{e_{n_{y\to y'}}}$ tal como se describen a continuación:

- $\exists! y$, tal que $\langle x, y\rangle \in E$, $R \in L(\langle x, y\rangle)$
- $\exists! R' \in Rs(x')$, $R' \sqsubseteq^* R$ tal que $\exists! y'$ con $\langle x', y'\rangle \in E'$, $R' \in L(\langle x', y'\rangle)$ y además se cumple que:
  1. $C_d(y) = \Phi$, $C_d(y') = \Phi'$, $\Phi'^{I^D} \subseteq \Phi^{I^D}$ ó
  2. puede definirse una función de especialización de y a y'.

El modelo de especialización entre el par de grafos G y G' es denotado por la función $f_{e_{g_{G\to G'}}} = f_{e_{n_{r_{0_G}\to r_{0_{G'}}}}}$.

Definición 3.11. Sean C, C' dos conceptos tales que $C' \sqsubseteq C$. El modelo de especialización de C a C', $f_{e_{e_{C\to C'}}}$, es definido por:

$$\bigcup_{\substack{\forall G, G', G \in \mathcal{F}_m(C),\\ G' \in \mathcal{F}_m(C')}} f_{e_{g_{G\to G'}}}$$

Así mismo, se define la función de especialización entre conceptos, f_e, como aquella que le asocia a cada par de conceptos $\langle C, C'\rangle$, $C \sqsubseteq C'$, el modelo de especialización entre dichas clases.

[1] Recuérdese que por la propia definición de ontología, toda instancia de una clase específica es también instancia de una clase más general.

La idea detrás de estas definiciones es la de encontrar emparejamientos entre pares de nodos de grafos de completamiento de conceptos que mantienen una cierta relación jerárquica. Estos emparejamientos, por demás únicos, permiten describir cómo especializar las relaciones y conceptos asociados a una instancia para convertirla de una clase más general a una más específica.

Esta definición de especialización entre conceptos puede parecer muy atrevida, pues podría decidir una "especialización" para una instancia en los casos en los que para un concepto sólo puede encontrarse un modelo de especialización para uno y sólo un concepto más específico. Pero ello no ocurre porque en ese caso los restantes conceptos específicos no podrían ser subsumidos por el concepto padre (posiblemente porque se demuestra que son insatisfacibles). Por otro lado, gran parte de las taxonomías definen espacios de conceptos que cubren el universo del concepto base. De ahí que, la aplicación de esta definición tiene un carácter práctico importante, pues permite intuir la clase específica de una instancia sin ambigüedad en aquellas partes de la taxonomía que formen un cubrimiento de un concepto padre.

Revisión de conceptos especializados

Recorriendo los grafos de completamiento con ramas forzadas de un concepto C se obtienen todos los caminos que pueden ser usados en alguna interpretación de dicho concepto, lo que denotaremos $Cams(C)$ (tal como se describe en la definición 3.13). Sin embargo, pongamos por caso que en algún nodo x del grafo de completamiento se tiene que $\exists R.C' \in L(x)$, y ha sido deducido que $I \Vdash C'' \sqsubset C'$, entonces reconocer todos los conceptos a los que se puede arribar desde C, tiene que considerar también caminos análogos a los anteriores en los que se sustituya adecuadamente C' por C''.

Si se emplean los resultados de la clasificación de los conceptos nombrados (proceso primario en el razonamiento), se pueden calcular todos los caminos que parten desde C, empleando para ello el siguiente resultado.

Lema 3.2. Sea $G = (V, E, L, \dot{\neq}, C_d)$ un grafo de completamiento de C, $x \in V$, $x \neq r_{0_G}$, tal que $C_d(x) = C'$ y sea $C'' \sqsubset C'$ con $G'' = (V'', E'', L'', \dot{\neq}'', C''_d)$ un grafo de completamiento de C''. Es posible formar nuevas interpretaciones de C empleando los grafos de completamiento G y G''. El grafo de completamiento que representa dicha interpretación se calcula mezclando el nodo $r_{0''_G}$, $r_{0''_G} \in V''$, con el nodo $x \in V$ y completando el grafo mediante la aplicación de las reglas de expansión de la manera tradicional.

El lema anterior explica que nuevas interpretaciones de C asociadas a la especialización de alguna de las clases de llegada, pueden obtenerse a partir de los grafos de completamiento de las clases de la base de conocimiento. Para ello, considerando un grafo de completamiento particular (una interpretación particular) de C, debe mezclarse un nodo no raíz de dicho grafo, dígase x, con

el nodo raíz de algún grafo de completamiento de alguna clase más específica que $C_d(x)$ y completar el grafo así creado. Es importante considerar varios temas: (1) la mezcla en sí no acarreará conflictos pues una de las hipótesis es que se conozca que las clases de los nodos mezclados mantienen una relación en jerarquía conceptual; (2) a pesar de ello, el grafo de completamiento inicial formado a partir de la mezcla puede no estar completo e incluso contener o generar un clash y por ello es necesario la aplicación de las restantes reglas de expansión; (3) el algoritmo de completamiento sobre el grafo de completamiento así construido también termina, basta mantener el orden de aplicación de las reglas de manera tradicional, lo cual es consecuencia del lema 6 en Horrocks & Sattler (2005b); (4) se tendrán todos los caminos de agregación hasta clases especializadas desde un concepto C sólo cuando se aplique este lema para todas las posibles sustituciones de cada concepto de llegada de algún grafo de completamiento de C con los conceptos más específicos que dicho concepto de llegada; (5) por último, el lema anterior no excluye su aplicación a los grafos de completamiento con ramas forzadas.

Definición 3.12. Llamaremos grafo de completamiento (con ramas forzadas) con especialización de clases a cualquier grafo de completamiento (con ramas forzadas) calculado por aplicación del lema 3.2.

En lo sucesivo se emplea $tipo \in \{\ell, f, fe, e\}$ para denotar el tipo de grafo de completamiento (bosque de grafos de completamiento) con el que se trabaja:

- ℓ denota grafos o bosques de grafos tal como en la definición 3.6.
- f denota grafos o bosques de grafos con ramas forzadas.
- fe denota grafos o bosques de grafos con ramas forzadas y especialización de clases.
- e denota grafos o bosques de grafos con especialización de clases.

La siguiente definición explica tan sólo cómo recorrer y reconstruir los caminos de agregación a partir de los grafos de completamiento. Nótese que $tipo$ también afecta a la notación de las funciones de recuperación de caminos de agregación.

Definición 3.13. Sea $\mathcal{O}$ una ontología y $\mathcal{F}^{\mathcal{O}}_{tipo}$ el bosque de grafos de completamiento de tipo $tipo$ de todas las clases nombradas en $\mathcal{O}$. Denotaremos por $\mathcal{F}_{m_{tipo}}$ una función que recupera el bosque de grafos de completamiento de tipo $tipo$ asociado a cada concepto. Sea $Path$ una función que recupera todos los caminos en los nodos de G, $G \in \mathcal{F}^{\mathcal{O}}_{tipo}$, o sea, $Path(G) = \{\oplus_{1\leq i\leq n} x_i | x_i \in V, 1 \leq i \leq n, x_1 = r_{0_G}, \{\langle x_i, x_{i+1}\rangle \,|\forall 1 \leq i \leq n-1\} \subseteq E\}$. Denotaremos por $CamsC_{tipo}$ y $CamsCC_{tipo}$ dos funciones que permiten recuperar el conjunto de caminos de agregación que existen desde algún concepto y desde un concepto a otro en $\mathcal{O}$, respectivamente. O sea, $\forall C, C' \in N_A$ se tiene que:

- $CamsC_{tipo}(C) = \bigcup_{\substack{\oplus_{1\leq i\leq n} x_i \in Paths(G),\\ G\in\mathcal{F}_{m_{tipo}}(C)}} \oplus_{1\leq j\leq n-1} \left\langle L(\langle x_j, x_{j+1}\rangle), C_d(x_{j+1})\right\rangle$
- $CamsCC_{tipo}(C,C') = \{cam | cam = \oplus_{1\leq i\leq n} \langle RS_i, C_i\rangle, C_n = C',$
  $cam \in CamsC_{tipo}(C)\}.$

Por último la siguiente definición es útil para reconocer si dos conceptos están relacionados a través de un conjunto único de relaciones.

Definición 3.14. Sea *CCams* un conjunto de caminos desde un concepto C al concepto C', $CCams \subseteq Cams_{tipo}(C,C')$. Se dice que de C a C' existe un camino no ambiguo respecto a *CCams* si $|CCams| = 1 \wedge CCams = \{\oplus_{1\leq i\leq n} \langle RS_i, C_i\rangle, |RS_i| = 1, 1 \leq i \leq n\}$.

Nótese que se ha empleado *CCams* en lugar que la función $Cams_{tipo}(C,C')$, para flexibilizar el concepto de camino no ambiguo, de acuerdo a intereses particulares, tal como podrá apreciarse en la siguiente sección.

A pesar de la importancia teórica de estos resultados, su aplicación directa es inviable, en tanto ellos implican la generación de todas las combinaciones de interpretaciones de conceptos, usando todas las especializaciones de cada clase. Lo cual conlleva a obtener una cantidad exponencial (en la cantidad de clases especializadas) de nuevas interpretaciones a partir de cada interpretación obtenida por ramas forzadas de cada clase.

Desde el punto de vista práctico, por tanto, los caminos de agregación entre conceptos deben ser obtenidos de manera aproximada, obviando la verificación de la consistencia de las instancias que estarían contenidas en los grafos de completamiento con especialización de clases. Para ello, se definen los caminos de agregación entre conceptos con incertidumbre. De manera intuitiva pueden verse como aquellos caminos que se forman por la extensión de los obtenidos a partir de los grafos con ramas forzadas desde un concepto C hasta un concepto C', $CamsCC_f(C,C')$, con los caminos obtenidos de los grafos con ramas forzadas desde los conceptos especializados del concepto de llegada C', $CamsC_f(C,C')$. Formalmente pueden definirse como:

Definición 3.15. Sea $\mathcal{O}$ una ontología. Sean $CamsC_f$ y $CamsCC_f$ las funciones de caminos de agregación de conceptos obtenida desde los grafos de completamiento con ramas forzadas para $\mathcal{O}$ como en la definición 3.13. La función $CamsC_i$ permite recuperar los caminos de agregación entre conceptos con incertidumbre para $\mathcal{O}$ de la siguiente manera $\forall C \in N_A$,

$$CamsC_i(C) = \{\oplus_{1\leq i\leq n-1} \langle RS_i, C_i\rangle \oplus \langle RS_n, C^*\rangle \oplus cam' | C^* \sqsubset C',$$
$$\oplus_{1\leq i\leq n} \langle RS_i, C_i\rangle \in CamsCC(C,C'),$$
$$cam' = \oplus_{1\leq j\leq m} \left\langle RS'_j, C'_j\right\rangle \in \{\ell\} \cup CamsC(C^*) \cup CamsC_i(C^*)\} \wedge$$
$$\neg\exists C_i, C'_j \text{ tal que } C_i \sqsubseteq C'_j\},$$

con ℓ representando un camino vacío. Análogamente a lo descrito en la definición 3.13 se define $CamsCC_i$.

Finalmente, en el Algoritmo 3.1 puede verse el proceso completo del cálculo de los caminos de agregación entre conceptos. La salida de dicho algoritmo es un diccionario que relaciona pares de conceptos con los caminos de agregación entre ellos y un valor booleano que describe si ha sido constatada la consistencia de instancias que contengan dicha composición de relaciones. Para ello, en la primera parte del algoritmo se clasifican todos los conceptos nombrados empleando las consideraciones para obtener los grafos de completamiento con ramas forzadas y se recupera para cada concepto nombrado todos caminos que aparecen en sus grafos. En la segunda parte, cada camino es extendido según la definición de los caminos con incertidumbre.

Como puede verse, el diccionario *Cams* representa la función $CamsCC_i$ de la definición anterior; la función $CamsC_i$ puede obtenerse por la unión de todos los caminos en $CamsCC_i$ para cada concepto de salida. Por otro lado, este algoritmo encuentra todos los posibles caminos entre conceptos. Si se deseara obtener sólo los caminos no ambiguos, habría que adicionar condiciones durante la actualización del diccionario *Cams* que garanticen mantener el único camino entre los conceptos en cuestión que contenga sólo conjuntos unitarios de relaciones, si es que éste existe y es único.

3.4 Operaciones sobre instancias ontológicas

En esta sección describimos algunas operaciones que consideramos interesantes en el procesamiento de instancias ontológicas. Comenzaremos por la definición de descripción de una instancia ontológica y posteriormente se describen las definiciones de especialización, abstracción, unificación, agregación, diferencia y diferencia simétrica entre instancias.

Definición 3.16. Sea a una instancia en el Abox, $\mathcal{A}$, de una ontología concreta $\mathcal{O}$, con interpretación I. Diremos que la descripción de la instancia a es el conjunto $d(a) = \{R(a,b) | R(a,b) \in \mathcal{A}\}$ y que dicha instancia es de clase C, lo que se denota por $a \in_* C$, si C es el concepto más específico que puede deducirse de $\mathcal{A}$ para a, o sea, $\forall C^*$ tal que $I \Vdash C^*(a)$, $C^* \sqsubset C$. Por analogía $\subseteq_*$ y $\subset_*$ denotan las operaciones de subconjuntos y subconjuntos propios entre instancias y una clase.

Definición 3.17. Definimos la especialización de la instancia a de clase C hacia la clase C', $C' \sqsubset C$, denotado por $d(a \uparrow_{C'} a') \neq indefinido$ a la instancia a' de tipo C', cuya descripción es calculada mediante $d_m(a \uparrow_{C'} a', f_{e_{c_{C \to C'}}})$, siendo $f_{e_{c_{C \to C'}}}$ el modelo de especialización entre conceptos C y C' (definición 3.11) y d_m la descripción calculada por modelo de especialización a través de:

$$d_m(a \uparrow_{C'} a', f_e) = \begin{cases} \{R'(a',b') | R(a,b) \in d(a), b \in_* C^*, \\ \qquad f_e(R,C^*) = \{\left\langle R', C'', f_e^{'} \right\rangle\}, & \text{si } |d(a \uparrow_{C'} a', f_e)| = |d(a)| \\ \qquad d(b \uparrow_{C''} b', f_e^{'}) \neq indefinido\} \\ \\ indefinido & \text{en otro caso} \end{cases}$$

Algoritmo 3.1 Cálculo de los caminos de agregación con incertidumbre de una ontología $\mathcal{O}$

Entradas: $\mathcal{F}_f^{\mathcal{O}}$

{ $\mathcal{F}_f^{\mathcal{O}}$, bosque de grafos de completamiento con ramas forzadas de todas las clases nombradas de $\mathcal{O}$, donde $\mathcal{F}_f$ es una función que devuelve el subconjunto de grafos en $\mathcal{F}_f^{\mathcal{O}}$ asociados a cada concepto. }

Salidas: *Cams*

{*Cams*, diccionario que mantiene todos los caminos de agregación asociados a cada par de clases y valor de certeza de consistencia del camino. Será usada la nomenclatura $LLaves(cams)$ para recuperar las llaves del diccionario y cada entrada del diccionario será un conjunto de elementos de la forma <camino, certeza de consistencia del camino>.}

Primera Parte: Clasificación de conceptos nombrados y colección de caminos seguros.

Clasificar todos los conceptos nombrados empleando las reglas de expansión del algoritmo tableau SHOIQ(D) en las Figuras 3.3 y 3.4, así como en el teorema 3.1.

for all $C \in N_A$ **do**

  for all $G \in \mathcal{F}_f(C)$ **do**

    $AddPaths(Cams, C, G)$

Segunda Parte: Extensión de los caminos seguros.

$lC_{camsExt} = \emptyset$ { $lC_{camsExt}$ es el conjunto de las clases para las que sus caminos ya han sido extendidos con los caminos con incertidumbre}

for all $C \in N_A, C \notin lC_{camsExt}$ **do**

  $AddPathExts(Cams, C, lC_{camsExt})$

function $AddPaths(Cams, C, G)$:

- Realizar recorrido en preorden en G. Sea $cams_{G,C'}$ el conjunto de caminos coleccionados de esta manera.
- Actualizar diccionario *Cams*, usando como llave (C, C') y como valor los caminos en $cams_{G,C'}$ y valor de certeza "$true$".

function $AddPathExts(Cams, C, lC_{camsExt})$:

for all $(C, C') \in Llaves(Cams)$ **do**

  for all $C^* \sqsubset C'$ **do**

    if $C^* \notin lC_{camsExt}$ **then**

      AddPathExts($Cams, C^*, lC_{camsExt}$)

    for all $(C^*, C'') \in Llaves(Cams)$ **do**

      Actualizar diccionario *Cams*, considerando la Definición 3.15, usando como llave (C, C'') y como valor los caminos en $\{\oplus_{1 \leq i \leq n-1} \langle RS_i, C_i \rangle \oplus RS_n, C^* \oplus cam'' \mid \langle cam', certeza' \rangle \in Cams(C, C'), cam' = \oplus_{1 \leq i \leq n} \langle RS_i, C_i \rangle, \langle cam'', certeza'' \rangle \in Cams(C^*, C'')\}$ con valor de certeza "false".

$lC_{camsExt} = lC_{camsExt} \cup \{C\}$

La actualización del Abox con tal especialización viene dado por: $\mathcal{A}\backslash\{C(a), C'(a'), d(a)\} \cup d(a \uparrow_{C'} a')$.De manera análoga se define la operación de abstracción de una instancia y se denota por $d(a \downarrow_{C'} a')$.

La especialización (abstracción) de una instancia se realiza considerando un modelo de especialización que permite "transformar" cada instancia y relación asociada a la que se especializa (abstrae) según el modelo. Nótese la recursividad en el proceso de especialización que es inducida por el propio modelo.

Las siguientes dos definiciones tienen como objetivo generar una nueva instancia a partir de la combinación de las descripciones de dos instancias cuyas clases mantienen una relación jerárquica. La primera es llamada combinación segura pues sólo combina aquellos datos que forman una única instancia. La combinación total combina todos los datos que se asocian a las instancias "originales"[1] a través del mismo conjunto de relaciones.

Definición 3.18. Sea c_{Φ^D} una función (llamémosla de combinación de datos simples[2]) que permite asociar Φ a otra función rep_Φ, la cual asocia a su vez subconjuntos de Φ en una *representación compacta del subconjunto de entrada*[3]. Sea d, d' dos datos en $\mathcal{U}$. Llamaremos combinación segura de los datos d y d' al dato calculado por medio de:

$$d(d \cup d') = \begin{cases} c_{\Phi^D}(\Phi)(\{d, d'\}), & \text{si } \{d, d'\} \subseteq \Phi, \Phi \in \Phi^D \\ d' \cup d, & \text{si } d \in_* C, d' \in_* C', C \sqsubseteq C' \\ d(d \uparrow_{C'} d') \cup d(d') \backslash & \\ \quad \backslash \{R(d', b_1) \in d(d \uparrow_{C'} d'), R(d', b_2) \in d(d') | & \text{si } d \in_* C, d' \in_* C', C' \sqsubseteq C \text{ y} \\ \qquad R \in N_{R_1}\} \cup & \text{durante el proceso} \\ \quad \cup \{R(d', b_1 \cup b_2) | R(d', b_1) \in d(d \uparrow_{C'} d'), & \text{no se ha obtenido ninguna indefinición} \\ \qquad R(d', b_2) \in d(d') | R \in N_{R_1}\}, & \\ indefinido, & \text{en otro caso} \end{cases}$$

La combinación segura de datos permite para datos simples combinarlos según las funciones en c_{Φ^D}, y para instancias generar una nueva si las instancias que se combinan mantienen alguna

[1] No exactamente las originales, pues es posible que una de ellas se especialice para poder realizar la combinación.

[2] Léase también como anotación compacta de datos simples.

[3] Por ejemplo, sea $\Phi^D = \{\mathbb{Z}\}$, $c_{\Phi^D} = \{\langle \mathbb{Z}, rangeOfIntegerSets \rangle\}$ donde la función *rangeOfIntegerSets* tiene como dominio $\mathbb{Z}$ y como imagen las representaciones más compactas de conjuntos de enteros, $2^{\mathbb{Z}}$, empleando conjuntos de rangos de enteros. Así, $rangeOfIntegerSets(\{1,2,3,4,5,7\}) = [1,5] \cup [7,7]$.

relación de orden jerárquico. En este caso, la clase de la instancia combinada es el concepto más específico entre los asociados a las instancias que se combinan (por lo que la instancia más general se especializa hacia la clase más específica) y la combinación se realiza de manera recursiva mezclando los datos de las relaciones funcionales.

Definición 3.19. Sea c_{Φ^D} una función de combinación de datos simples. Sea d, d' dos datos en $\mathcal{U}$. Llamaremos combinación total de los datos d y d' al dato calculado por medio de:

$$d(d \cup_t d') = \begin{cases} c_{\Phi^D}(\Phi)(\{d,d'\}), & \text{si } \{d,d'\} \subseteq \Phi, \Phi \in \Phi^D \\ d' \cup_t d, & \text{si } d \in_* C, d' \in_* C', C \sqsubseteq C' \\ d(d \uparrow_{C'} d') \cup d(d') \setminus & \text{si } d \in_* C, d' \in_* C', C' \sqsubseteq C \text{ y} \\ \quad \setminus \{R(d',b_1),...,R(d',b_n) | \{b_1,...,b_n\} \subseteq_* C''\} \cup & \text{durante el proceso} \\ \quad \cup \{R(d',b_1 \cup ... \cup b_n) | R(d',b_1),...,R(d',b_n) \in & \text{no se ha obtenido ninguna} \\ \quad \in R(d',b_1),...,R(d',b_n) | \{b_1,...,b_n\} \subseteq_* C''\} & \text{indefinición} \\ indefinido, & \text{en otro caso} \end{cases}$$

La combinación total se diferencia de la segura en que se combinan todos los datos asociados a las mismas relaciones, independientemente de que éstas sean funcionales. De esta forma la instancia que se genera contiene sólo una entrada por cada relación. Ello es muy útil, por ejemplo, para hacer un resumen acerca de un conjunto de instancias. Una aplicación que salta a la vista es la generalización para modelos multidimensionales de instancias ontológicas, tal y como se describe en el Capítulo 5.

En particular podemos identificar dos tipos de funciones de combinación de datos simples:

- de unificación: que devuelven un valor, o sea, donde cada función rep_Φ asocia datos 2^Φ en Φ. Entre ellas, pueden tenerse también algunas orientadas a la estadística como calculo de medias, desviaciones, etc.. De especial interés es la función de *unificación restrictiva* que se define para cualquier tipo de datos por medio de:

$$f(\{d_1,...,d_n\}) = \begin{cases} d, & \text{si } d_1 = ... = d_n \\ indefinido, & \text{en otro caso} \end{cases}$$

- de generalización: que no devuelven un valor en Φ, o sea, donde cada función rep_Φ asocia datos 2^Φ en una *anotación compacta*. A pesar de la utilidad de los resultados que se obtienen por la generalización de datos, tal como se muestra en los Capítulos 4 y 5, los datos

resultantes por generalización de datos simples no son representables en el mismo lenguaje DL que los datos que les dieron origen, en tanto no pertenecen a la interpretación de la misma ontología.

En lo adelante, se utilizará "combinación de datos" para significar a cualquiera de ellas: sea combinación de datos segura o total. Sin embargo, la definición que aparece a continuación permite diferenciar las combinaciones cuando se emplean sólo funciones de unificación restringida.

Definición 3.20. Sean a, a' dos instancias de clase C y C', respectivamente. Definimos la unificación de las instancias a y a', denotado por $a\widehat{\cup}a'$, como la combinación de los datos a y a' donde todas las funciones de combinación de datos simples son de tipo unificación restrictiva. La generalización de las instancias a y a' se define como la combinación de los datos a y a', denotado por $a\widehat{\cup}_t a'$, en las que no todas las funciones de combinación de datos simples son de tipo unificación restrictiva. En ambos casos la descripción de las combinaciones resultantes se denotan por $d(a \text{ op } a')$, con $op \in \{\widehat{\cup}, \widehat{\cup}_t\}$.

La siguiente definición establece la condición bajo la cual dos instancias contienen información complementaria y pueden ser unificadas en el Abox de las ontologías. Esta definición podría ser utilizada para eliminar redundancias y aumentar la velocidad durante el razonamiento en el Abox, pues todas las instancias que describen a un único objeto (las que pueden reconocerse por las relaciones de equivalencias entre instancias) pueden ser consolidadas en una sola que contiene toda la información de las instancias originales, incluyendo el tipo particular al que se le asocia.

Definición 3.21. Sea a, a' dos instancias de clase C y C' respectivamente, en el Abox $\mathcal{A}$. Sea c_{Φ^D} la función de combinación de datos simples que emplea función de combinación tipo unificación restrictiva para todos los tipos de datos. Las instancias a y a' son complementarias si $d(a\widehat{\cup}a')\}$ es consistente. La actualización del Abox por la unificación de las instancias a y a' viene dada por: $\mathcal{A}\backslash\{a, a', d(a), d(a'), C(a), C'(a')\} \cup \{C'(a\widehat{\cup}a'), d(a\widehat{\cup}a')\}$[1].

Otra operación importante sobre instancias es la agregación. Una instancia puede agregar a otra si existe un camino de agregación de la primera a la segunda. Claro está que puede haber más de un camino para realizar dicha agregación y puede resultar indecidible seleccionar sólo uno de ellos. Sin embargo, si se seleccionan aquellos caminos que "utilizan" las mismas relaciones y los conceptos más generales, el conflicto puede quedar solucionado[2]. A continuación se define la

[1] Aunque en estas definiciones se eliminan del *Abox* todas las descripciones asociadas a a y a', ello no es imprescindible, su objetivo es reducir la redundancia. Si no se desea eliminar la información original, bastaría con adicionar las equivalencias pertinentes. Por ejemplo, en el caso de la definición 3.25 equivalencias entre a y a^* y entre a' y a'^* deben ser añadidas al Abox.

[2] En el Capítulo 4 se considera además la longitud mínima de los caminos para realizar una selección más fina durante las agregaciones. Aunque en el marco de trabajo de ese capítulo esta condición es justificable, ello no ocurre en entornos más generales.

propiedad de camino más general, así como las operaciones de selección de caminos más generales de un conjunto de caminos y de agregación de instancias. Antes que nada se introduce la propiedad que establece si dos caminos son comparables.

Definición 3.22. Sean $cam = \oplus_{1\leq i\leq n} \langle RS_i, C_i\rangle$ y $cam' = \oplus_{1\leq i\leq n'} \langle RS'_i, C'_i\rangle$ dos caminos de agregación. Se dice que *cam* es más general que cam' si:

- $n = n'$,
- $\exists C, C'$ tal que $cam \in CamsC_i(C)$, $cam' \in CamsC_i(C')$ y $C' \sqsubseteq C$,
- $\forall i \in \{1,...,n\}, C'_i \sqsubseteq C_i$

Definición 3.23. Sea *CCams* un conjunto de caminos. El conjunto de caminos más generales de *CCams*, denotado por medio de la operación $CamsGrales(CCams)$, es el conjunto $\{cam | cam \in CCams$ y $\neg\exists cam' \in CCams$ tal que cam' es más general que $cam\}$.

Definición 3.24. Sean a, a' dos instancias de $\mathcal{O}$ de clase C y C', respectivamente. La agregación de a' a a, denotada por $a \leftarrow a'$, es definida por:

$$d(a \leftarrow a') = \begin{cases} d(a \uparrow_{C^*}, a^*) \cup R_1(a^*, a_1) \cup \\ \bigcup_{2\leq i\leq n-1} R_i(a_{i-1}, a_i) \cup \\ \cup R_n(a_n, a'^*) \end{cases} \quad \begin{array}{l} \text{si } CamsGrales(\bigcup_{\substack{\forall C^*, C'^*, \\ C^*\sqsubset C, \\ C'^*\sqsubset C'}} CamsCC_i(C^*, C'^*)) = \\ \quad = \{cam\}, cam = \oplus_{1\leq i\leq n} \langle \{R_i\}, C_i\rangle \wedge \\ \exists! C^* \in CInics = \{C'' | cam \in CamsC_i(C'')\} \\ \quad \text{tal que } \forall C'' \in CInics, C'' \sqsubseteq C^* \wedge \\ d(a \uparrow_{C^*} a^*) \neq indefinido, \\ d(a' \uparrow_{C_n} a'^*) \neq indefinido \end{array}$$

$$indefinido, \qquad \text{en otro caso}$$

Definición 3.25. Sean a, a' dos instancias en el Abox $\mathcal{A}$ de clase C y C', respectivamente. La instancia a' es agregable a la instancia a si $d(a \leftarrow a') \neq indefinido$ y consistente. La actualización del Abox empleando tal agregación se obtiene por: $\mathcal{A}\backslash\{a, a', d(a), d(a'), C(a), C'(a)\} \cup C^*(a^*) \cup C'^*(a') \cup d(a \leftarrow a')$[1].

Nótese que en este caso se hace alusión a la verificación de la consistencia de la nueva instancia formada mediante la agregación. Sin embargo, gracias al algoritmo tableau y que las modificaciones expuestas para crear las ramas forzadas no varían el valor de la satisfacibilidad de un concepto, la nueva instancia puede ser inconsistente sólo si ha sido utilizada para la agregación un camino en $CamsCC_i$, o hablando en términos del Algoritmo 3.1, si ha sido empleado un camino en el diccionario *Cams* con valor de incertidumbre "false".

Finalmente, se definen las operaciones de diferencia y diferencia simétrica entre un par de instancias. Nótese que en el primer caso, la descripción de la seudo-instancia resultante es una

subdescripción de a mientras en la segunda lo que se obtiene no es asociable a ninguna de las clases C ó C' de las instancias a y a', respectivamente. Estas funciones están orientadas al análisis de datos, y no deben ser empleadas en el proceso de actualización de las instancias de un Abox de una ontología.

Definición 3.26. Sean a, a' dos instancias en el Abox $\mathcal{A}$ de clase C y C', respectivamente. La diferencia de las instancias a y a' es la instancia $a\widehat{\setminus}a'$, cuya descripción $d(a\widehat{\setminus}a') \neq indefinido$ es resultante de:

$$d(a\widehat{\setminus}a') = \begin{cases} d(a \downarrow_C a)\widehat{\setminus}d(a') & \text{si } C' \sqsubseteq C \\ d(a \uparrow_C a')\widehat{\setminus}d(a') & \text{si } C \sqsubseteq C' \\ \{R(a,b) | R(a,b) \in d(a) \wedge \neg\exists R(a',b) \in d(a')\} & \text{en otro caso} \end{cases}$$

Definición 3.27. Sean a, a' dos instancias en el Abox $\mathcal{A}$ de clase C y C', respectivamente. La diferencia simétrica de las instancias a y a' es la instancia $a\widehat{\setminus}a' \cup a'\widehat{\setminus}a$.

3.5 Análisis de la complejidad e implementación desarrollada

La solución brindada en este capítulo a las nuevas tareas de razonamiento identificadas (cálculo de los caminos de agregación y operaciones sobre instancias ontológicas) está basada en extender el algoritmo tableau de satisfacibilidad para lenguaje SHOIQ(D) de manera que cada nodo contenga al menos un arco de salida por cada relación en su etiqueta (concepto).

Como es conocido, el comportamiento del algoritmo tableau SHOIQ es NExpTime-completo en $|Abox|$, Horrocks & Sattler (2005a,b); Tobies (2000), pero que puede ser implementado para que funcione como SHIQ, Haarslev & Moller (2001); Horrocks & Patel-Schneider (1998); Sirin (2003), o sea, en tiempo ExpTime-completo en $|Abox|$. La extensión propuesta para el cálculo de los caminos de agregación entre conceptos no adiciona complejidad al peor de los casos del algoritmo tableau, pero hace que se comporta como el caso peor considerado el tableau original, pues se requiere la saturación del bosque de grafos de completamiento de cada clase. Por tanto, la complejidad del nuevo tableau para obtener todos los grafos de completamiento con ramas forzadas es también de ExpTime-completo.

El cálculo del bosque de completamiento de todos los conceptos permite extender cada camino para determinar los caminos de agregación con incertidumbre. Por tanto, como el tamaño de cada grafo de completamiento es a lo sumo ExpSpace en $|Abox|$, se tendrán a lo sumo $2^{p(|Abox|)}$ nodos y por tanto caminos. Cada uno de los cuales debe extenderse con a lo sumo $|Abox| \times 2^{p(|Abox|)}$ otros caminos. De lo que vuelve a resultar una complejidad ExpTime-completo.

Por otro lado, la determinación de los modelos de especialización entre pares de conceptos requiere la comparación de a lo sumo $2^{2\times p(|Abox|)}$ grafos, o sea, $2^{p(|Abox|)}$ grafos, cada uno con $2^{p(|Abox|)}$ nodos. De que resulta nuevamente una complejidad de ExpTime-completo.

De todo lo anterior, puede concluirse que las operaciones desarrolladas en el presente capítulo, pueden ser utilizadas con el mismo optimismo que cuando se emplean los algoritmos de tableau, pues no introducen un cambio alguno en el orden de la complejidad de los algoritmos tableau.

Los tiempos de ejecución de estos algoritmos podrían ocasionar comportamientos muy malos, o sea, tiempos inviables para aplicaciones reales. Sin embargo, para la mayoría de los casos típicos, los tiempos de ejecución aquí expuestos están sobredimensionados. En la práctica, no todas las bases de conocimiento emplean todos los constructores, y ontologías complejas (de más de 20 000 conceptos) han empleado Pellet para reconocer la satisfiabilidad de sus conceptos [1] y sus usuarios se han mostrado satisfechos con su funcionamiento. Por tanto, los elevados tiempos de ejecución de estos algoritmos no deben perjudicar ni empañar la utilidad práctica real de las soluciones propuestas.

Todo el procesamiento expuesto en este capítulo ha sido introducido en el sistema Pellet. Como resultado colateral se ha desarrollado una aplicación (Figura 3.5) que permite calcular los grafos de completamiento con ramas forzadas y mostrar la evolución de éstos durante su proceso de formación junto con las reglas tableau que han sido aplicadas.

3.6 Conclusiones

El fundamento teórico necesario para comprender las soluciones propuestas por esta tesis en el ámbito de la Web Semántica ha sido introducido en el presente capítulo. Toda la descripción ha sido basada en el lenguaje de la lógica descriptiva SHOIQ(D) e implementado en Pellet. Se han identificado y solucionado nuevas tareas de razonamiento: el cálculo de caminos de agregación, y las operaciones de especialización, abstracción, combinación, agregación, diferencia y diferencia simétrica entre instancias.

Todo ello está basado en una extensión del algoritmo de tableau que permite formar grafos de completamiento que no excluyen ninguna relación ni asociación entre conceptos. Esta extensión tiene tiempo de ejecución ExpTime en el tamaño del $|Abox|$. A pesar de este elevado tiempo de ejecución, como mismo exponen Ian Horrocks y Ulrike Sattler[2], el algoritmo de tableau (y por tanto de los resultados de la extensión) "...ha sido diseñado para que se comporte bien en muchos

[1] http://lists.mindswap.org/pipermail/pellet-users/.

[2] Ian Horrocks y Ulrike Sattler son los investigadores que encabezan la teoría de la lógica descriptiva. Sus grupos han desarrollado casi todos los algoritmos de tableau para DL conocidos y los estudios de comportamiento en tiempo y espacio de éstos y sus aplicaciones.

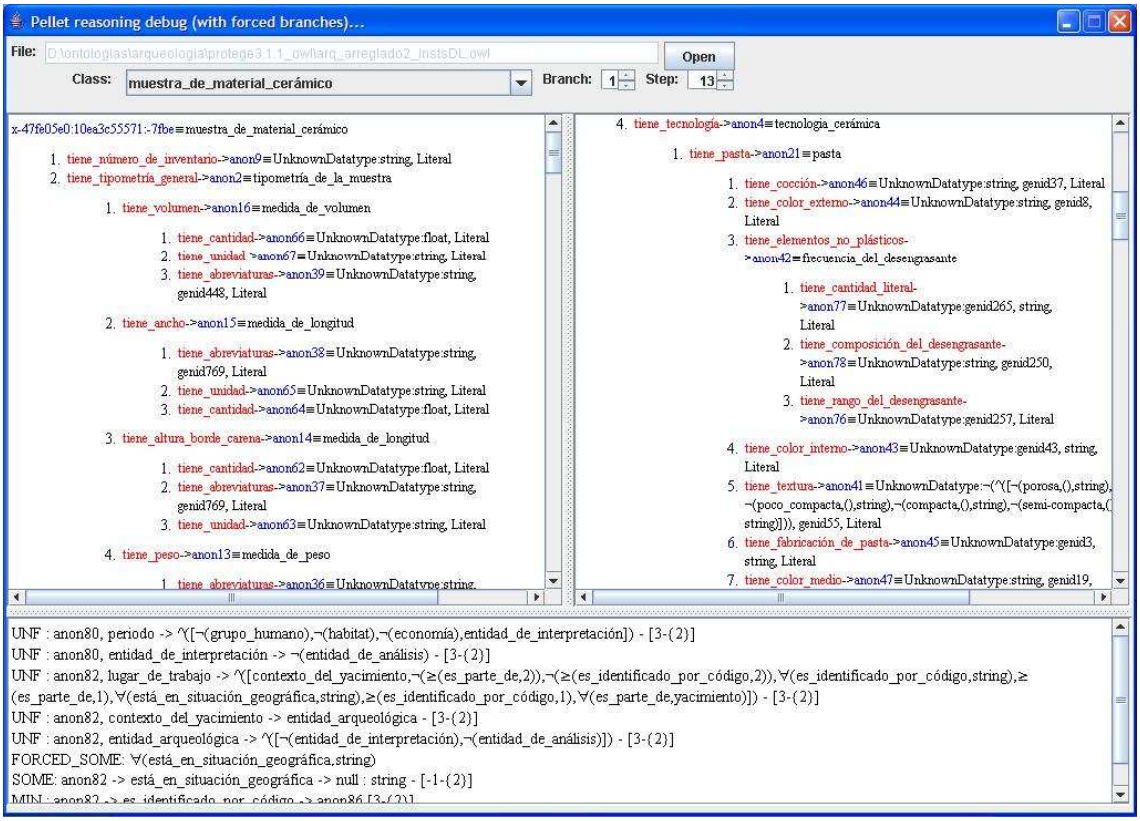


Figura 3.5: Sistema para el cálculo de grafos de completamiento con ramas forzadas. Como entrada se da el fichero OWL con la ontología, especificando entonces la clase de interés se puede observar la evolución de sus grafos de completamiento. En el ejemplo particular se muestran los grafos de completamiento formados en los pasos 12 y 13 (en los paneles izquierda y derecha respectivamente) para el primer grafo de completamiento (branch = 1) contenido en el bosque de modelos de la clase *muestra de material cerámico*, así como las reglas tableau empleadas en la trasformación (en el panel inferior).

casos típicos y exhibe un comportamiento “play as you go”... ”, Horrocks & Sattler (2005a). Por lo tanto, los tiempos de ejecución elevados no perjudican la utilidad práctica de los resultados obtenidos.

Capítulo 4

Extracción de instancias ontológicas

4.1 Introducción

En muchas áreas del conocimiento como la medicina, química, biología, ciencias jurídicas, arqueología, etc., existe entre los expertos en la materia un consenso de cómo describir las informaciones, y todo un conjunto de reglas para su análisis. Ello se traduce en la disponibilidad de bases de conocimiento de dominios específicos con ambigüedad reducida. La formalización de tales bases de conocimiento así como su uso en la extracción de información puede ser la clave para mejorar la interpretación de textos de dominios específicos, más concretamente, la clave para pasar de la web actual a la Web Semántica con un alto grado de fiabilidad.

En el presente capítulo se describe una propuesta para la formalización y extracción de instancias ontológicas para dominios de aplicación concretos, lo cual constituye parte de los resultados obtenidos durante la realización del proyecto CRISOL[1]. Las dos secciones siguientes estarán dedicadas a explicar la motivación de la propuesta a través de un breve análisis del estado del arte de la anotación semántica sobre dominios de aplicación concretos, la descripción de una propuesta de solución para la población de la Web Semántica y la descripción de un dominio de aplicación particularmente interesante y suficientemente formalizado sobre el cuál es plausible desarrollar nuestras ideas.

4.2 Estado actual

El proceso de reconocimiento de las instancias de una ontología que son descritas en un texto es conocido como *anotación semántica*. Muchas soluciones para la anotación semántica están

[1] Proyecto CICYT TIC2002-04586-C04-03.

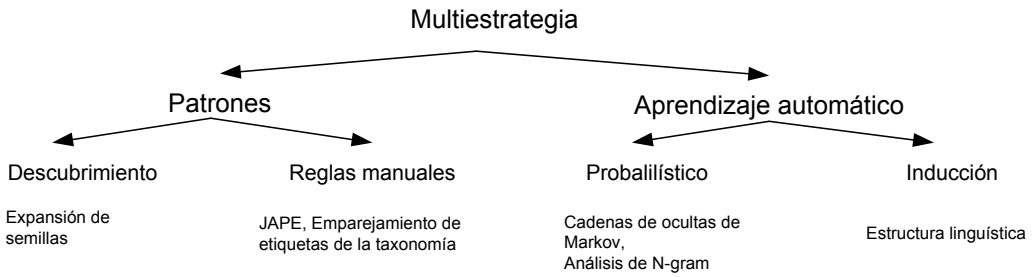


Figura 4.1: Clasificación de plataformas de anotadores semánticos.

basadas en la creación de sistemas que faciliten la anotación manual, usando herramientas de autorización y anotación en textos. Sin embargo, "el uso de anotadores humanos es frecuentemente fuente de errores debido a factores como la experiencia de la persona, nivel entrenamiento en la anotación, motivación personal y otros esquemas complejos de nuestro cerebro", Bayerl *et al.* (2003). Pero además, la anotación manual es un proceso caro, que frecuentemente no considera las múltiples perspectivas que pudiera tener una fuente de datos (múltiples ontologías pueden considerar visiones diferentes de un mismo fragmento de texto). Además, el volumen de documentos a procesar así como el tiempo requerido para ello, superan con creces las disponibilidades de equipos de expertos.

Es por ello que diferentes métodos de anotación semi-automáticos han sido propuestos[1]. La anotación (semi)automática proporciona la escalabilidad necesaria para anotar de manera consistente los documentos existentes en la web, y permite por otro lado, el uso de múltiples ontologías para anotar un único documento, pudiéndose extraer instancias con diferentes perspectivas de cada documento. A continuación, resumiremos brevemente los métodos de anotación semi-automática descritos en la literatura, guiándonos por la clasificación en Reeve & Han (2005) reproducida en la Figura 4.1 y culminaremos con una breve discusión acerca de las soluciones propuestas.

Según Reeve, los métodos de anotación semántica pudieran clasificarse en dos tipos fundamentales (Figura 4.1): los basados en patrones y los basados en aprendizaje automático. Los primeros tratan de descubrir patrones que identifican cada entidad a partir de un conjunto inicial de entidades etiquetadas tal como fue propuesto en Brin (1998) ó en el uso de reglas manualmente extraídas para encontrar entidades en los textos. Los segundos se basan en métodos probabilísticos ó de inducción. Los métodos probabilísticos, usan modelos estadísticos para predecir la ubicación de las entidades en los textos; los de inducción, permiten considerar la estructura lingüística para inducir reglas que permiten extraer entidades.

[1]La anotación totalmente automática es muy difícil debido a la imposibilidad de identificar todas las entidades posibles en los documentos sin la intervención del ser humano en algún momento durante el proceso de anotación.

La evaluación de la calidad de los resultados obtenidos por la anotación, es realizada utilizando las medidas *precisión*, *recuperación* y $F1$ empleada en los Sistemas de Recuperación de Información, Baeza-Yates & Ribeiro-Neto (1999), las que se definen por:

$$precisión = \frac{Total\ de\ entes\ recuperados\ correctos}{Total\ de\ entes\ recuperados}$$
$$recuperación = \frac{Total\ de\ entes\ recuperados\ correctos}{Total\ de\ entes\ a\ recuperar}$$
$$F1 = \frac{2 * precisión * recuperación}{precisión + recuperación}$$

La *precisión* permite evaluar en cuánto los resultados obtenidos son correctos, la *recuperación* permite evaluar cuánto de lo deseado se ha obtenido y $F1$ permite dar una evaluación global de los resultados considerando estas dos medidas de calidad.

4.2.1 Métodos basados en patrones

4.2.1.1 Descubrimiento de patrones

Los sistemas de extracción de información basados en patrones utilizan un conjunto de entrenamiento (consistente en textos en los que aparecen marcados cada elemento de información) para descubrir patrones que se asocian a cada elemento de información. Ejemplo de éstos son los trabajos de Maedche *et al.* (2002); Popov *et al.* (2003) y Buitelaar & Ramaka (2005) que a continuación se describen.

KIM, Popov *et al.* (2003), es un sistema que reconoce en el texto nombres de personas, organizaciones y lugares y los asocia con conceptos en una base de datos (también llamada KIM). Para ello emplean GATE[1], una arquitectura completa para el análisis sintáctico-semántico y la anotación de textos. La base de datos KIM les permite mantener las entidades encontradas. KIM se inicializa con un conjunto de entidades relevantes (lugares, personas, organizaciones muy conocidas) y luego las emplea para crear gramáticas que reconocen nuevas entidades. Las co-referencias son también consideradas con el objetivo de ubicar en los textos todas las posiciones donde se menciona cada concepto.

En Maedche *et al.* (2002) se expone un sistema muy parecido, el que se basa en otros tres: el analizador sintáctico SMES, Neumann *et al.* (1997), OntoEdit para crear e instanciar ontologías y Ontobroker, un sistema deductivo basado en bases de datos que permite formular demandas y recuperar la información anotada en las propias páginas web (por medio de crawlers o wrappers) que se asocian a entidades de las ontologías.

[1] http://gate.ac.uk/.

SmartWeb, Buitelaar & Ramaka (2005), explica un método no supervisado para anotar textos utilizando ontologías. Emplean los nombres de las clases de la ontología para detectar dónde aparecen los conceptos en los textos. Usando un contexto de palabras (ventana) alrededor de cada clase encontrada, entrenan un sistema de vecinos más cercanos que les permite reconocer, dado un texto nuevo, las clases asociadas a cada parte del texto. La recuperación y precisión que devuelven sus métodos es de aproximadamente un 90%.

4.2.1.2 Uso de reglas

Los sistemas de extracción de información basados en reglas utilizan conjuntos de reglas manualmente descritas que asocian ciertas características del texto a entidades. Los textos a anotar son revisados y se devuelven todas las entidades asociadas a cada parte del texto que se reconocen por medio de las reglas mencionadas. Los trabajos en Baumgartner *et al.* (2001); Kogut & Holmes (2001); Maynard *et al.* (2004) y Amardeilh *et al.* (2005) utilizan este enfoque.

AeroDAML, Kogut & Holmes (2001), realiza un emparejamiento entre nombres propios y relaciones a clases y propiedades de ontologías DAML, para lo cual emplean AeroText, una herramienta de *extracción de información* sobre textos que permite el reconocimiento de entidades tipo personas, organizaciones, lugares, fechas, expresiones temporales y co-referencia de entidades.

Lixto, Baumgartner *et al.* (2001), permite marcar un documento HTML y reconstruye los patrones relacionados con el texto marcado. A cada patrón se le puede añadir filtros (especificaciones de patrones que deben ocurrir anteriormente o posteriormente, etc.). de manera que el usuario puede definir iterativamente las condiciones de cada patrón. Por último, el sistema construye un wrapper que permite procesar otras páginas web con características similares.

h-TechSight, Maynard *et al.* (2004), con la ayuda de GATE realiza el análisis morfológico y lematiza las palabras contenidas en un documento HTML. Emplea además listas de palabras asociadas a conceptos de la ontología y reglas escritas manualmente (en lenguaje JAPE) para extraer instancias de dicha ontología. Sus resultados experimentales muestran un 97% de precisión y un 92% de recuperación. Lo más interesante es que luego monitorean el dinamismo de los conceptos e instancias para reconocer el comportamiento histórico de los objetos de la ontología.

En Amardeilh *et al.* (2005) se presenta un sistema que después de realizar el análisis lingüístico completo de un texto, obtienen un árbol conceptual etiquetando cada zona del texto con un concepto genérico [1](emplean el sistema IDE[2] y GATE). Este árbol conceptual es entonces parseado, asignándole a cada nodo un concepto de una ontología particular. La asignación de un concepto

[1]Un concepto genérico es cualquiera de contenidos en un diccionario, o sea, no necesariamente asociado a una ontología.

[2]http://www.temis.com/?id=51&selt=1.

a un nodo, es realizado considerando un conjunto de reglas de adquisición que son manualmente definidas y mantenidas.

4.2.2 Métodos basados en aprendizaje automático

4.2.2.1 Métodos probabilísticos

Los sistemas de extracción de información probabilísticos emplean o generan modelos probabilísticos para predecir el uso de una palabra (o una frase) en relación a una ontología. En Cimiano *et al.* (2004); Dill *et al.* (2003); Valarakos *et al.* (2004) y Yang & Choi (2003) pueden verse algunos de los trabajos representativos.

Pankow, Cimiano *et al.* (2004), encuentra todos los nombres propios en una página Web y entonces emplea un conjunto de patrones sintácticos (calculados manualmente) que conectan conceptos de una ontología a identificadores para generar un conjunto de patrones para cada nombre propio. Por último, encuestan a Google para darle ranking a cada patrón candidato y seleccionar el mejor (más frecuente) patrón para cada instancia (nombre propio). De esta forma, cada nombre propio de la página web es anotado con el concepto que define el patrón seleccionado.

SemTag, Dill *et al.* (2003), describe un método para la anotación semántica de instancias de una ontología. Para ello emplean la taxonomía de una ontología y asocian a cada nodo de la taxonomía un conjunto de etiquetas (sinónimos del concepto) y a cada etiqueta varios contextos (conjuntos de palabras cercanas a la etiqueta). Además, cada nodo de la taxonomía contiene una función que dice si un contexto dado se le ajusta. Por último, a un subconjunto de nodos de la taxonomía se le asignan dos medidas: una que valora el ajuste de un nodo a un tópico considerando el juicio humano y otra, que valora el ajuste de un nodo a un tópico considerando las salidas de las funciones de similitud asociadas a dicho nodo y a las de su subárbol. El algoritmo de anotación toma los diferentes contextos (conjuntos de palabras consecutivas) de las páginas web a anotar y decide para cada contexto el concepto que le es más cercano. Lo más interesante es que ha procesado 264 millones de páginas web obteniendo aproximadamente 434 millones de anotaciones, aunque no reportan los valores de precisión y recuperación. Las funciones de similitud usadas son el coseno y bayes considerando los contextos como vectores del TF-IDF ó la frecuencia de las palabras. TF-IDF con coseno, resultó la de mejor ajuste.

En Valarakos *et al.* (2004) se puebla una ontología siguiendo varios pasos: 1) anotan los documentos usando expresiones regulares (ER), 2) usan cadenas de Markov para entrenar y luego anotar aquellos entes de la ontología que no pueden ser expresados por ER, 3) revisan entre los entes encontrados si alguno de ellos pueden corresponderse a la misma entidad, considerando para

ello la existencia de un conjunto de caracteres comunes (solución a la co-referencia), 4) validación y agregación de las instancias a la ontología por parte de un experto.

En Yang & Choi (2003) se describe un sistema para reconocer elementos ontológicos a partir de un conjunto de documentos etiquetados, cuya salida es un documento XML que representa las diferentes formas que pudiera tener cada elemento de la ontología (ente). El cálculo de estas formas está basado en la descripción morfológica de los entes (tiene letra mayúscula, número, etc.) y el porcentaje de entes etiquetados de igual tipo que coinciden con un patrón candidato. Se seleccionan aquellos patrones (formas morfológicas) con mayor porcentaje de validez.

4.2.2.2 Métodos de inducción

Los métodos de inducción para la extracción de información emplean la estructura lingüística de los textos para inducir patrones que reconocen entidades. En Celjuska & Vargas-Vera (2004); Ciravegna *et al.* (2002, 2004); Etzioni *et al.* (2005); Handschuh & Staab (2002); Handschuh *et al.* (2001, 2003); Vargas-Vera *et al.* (2001, 2002) se describen trabajos que emplean este prototipo.

Amilcare y Melita, Ciravegna *et al.* (2002), permite anotar instancias de una ontología, empleando el sistema de extracción de información Amilcare. Amilcare (que a su vez usa ANNIE, un componente de GATE para realizar el análisis sintáctico de un texto) permite a partir de ciertas anotaciones manuales, inferir reglas para la extracción de los conceptos. Ellos muestran una precisión entre el 75%-97% y una recuperación de un 50%-100%, para al menos 9 conceptos. Proponen además emplear las correcciones de los usuarios para reentrenar el sitema. Este proceso de extracción y retroalimentación debe garantizar molestias mínimas en tiempo y errores.

Armadillo, Ciravegna *et al.* (2004), también usa Amilcare. Basado en las reglas y entidades extraídas por Amilcare, Armadillo induce los patrones de páginas web de sitios con una estructura altamente regular. Los patrones encontrados (atributos de una ontología particular) son entonces verificados analizando las estadísticas resultantes de consultas a servicios web de Google y CiterSeer. Los patrones más representativos pueden ser finalmente empleados para extraer información de páginas web, empleando un crawler de búsqueda.

KnowItAll, Etzioni *et al.* (2005), presenta tres métodos para mejorar los datos extraídos desde la web, pero sin el empleo de ejemplos hechos a mano. El primero se basa en la inducción de reglas para textos de dominios específicos; el segundo, trata de encontrar nuevas subclases de una clase dada; y el tercero, localiza en la web listas de instancias, y para cada clase de instancia aprende sus patrones regulares, los que luego emplea para añadir nuevos elementos a las listas. Este trabajo representa un avance en la teoría de extracción de información, pues no sólo trata el problema de la anotación semántica, sino que además, el del aprendizaje de “seudo-ontologías” y mezcla la solución de ambos problemas en un mismo proceso.

MnM, Vargas-Vera *et al.* (2001, 2002), es una herramienta de anotación semántica que provee soporte tanto para la anotación automática y semi-automática para anotar páginas web. Para ello, integra un navegador de la web con un editor de ontologías y un conjunto de APIs abiertas que enlazan servidores de ontologías y herramientas de extracción de información. Sus autores definen la aplicación como "un ejemplo de la próxima generación de editores ontológicos, basados en web y orientados al marcado semántico, y proveyendo mecanismos para el marcado automático a gran escala", Vargas-Vera *et al.* (2001, 2002).

Onto-O-mat, CREAM y S-CREAM, respectivamente en Handschuh & Staab (2002); Handschuh *et al.* (2001) y Handschuh *et al.* (2003), son una serie de trabajos que engloban un marco de trabajo para la anotación semántica. En sus inicios permitía a los usuarios a través de herramientas de marcado sobre el texto de las páginas web y las técnicas de arrastrar y soltar (en inglés, drag&drop) enlazar instancias y propiedades de una ontología a sus apariciones en el texto (muy parecido al sistema descrito en Kalyanpur *et al.* (2002)). Las últimas versiones, han considerado también las salidas del sistema de anotación semi-automática Amilcare, para ayudar al usuario en el proceso de anotación. El dinamismo de las páginas que se van anotando también es tenido en cuenta pues el sistema permite mantener diferentes versiones de las anotaciones desarrolladas.

En **OntoSophie**, Celjuska & Vargas-Vera (2004), se emplea Marmot, un sistema de procesamiento de lenguaje natural, para determinar las funciones gramaticales de cada palabra en cada oración. Con ello emplean una herramienta de inducción de reglas para deducir reglas que asocie cada entidad de la ontología con las palabras y sus funciones gramaticales. A cada regla se le asigna un cada peso, eliminando las de muy baja relevancia y entonces emplea las reglas calculadas para formar instancias no complejas. El usuario decidirá finalmente si adicionar o no las instancias a la ontología. Emplean 41 clases en la experimentación, los resultados muestran un 15% de mejora respecto a trabajos similares.

En **Artequakt**, Alani *et al.* (2003), también se realiza el análisis sintáctico y semántico de cada oración en los textos a procesar. Ello les permite realizar un emparejamiento entre los verbos y sustantivos encontrados en los textos, con los verbos y sustantivos expuestos por la ontología, empleando además WordNet[1] para hacer un matching más suave considerando sinónimos, hipónimos, merónimos e hiperónimos. La aplicación particular desarrollada permite recuperar desde textos la información relacionada con artistas. Los documentos proveedores de la información son obtenidos directamente desde la web por medio de consultas a servidores como Google y descartando las páginas irrelevantes. Todo el proceso obtienen una precisión del 85% y una recuperación del 42%.

[1] WordNet es un lexicón semántico para el idioma Inglés que comenzó a desarrollarse en 1985, en la Universidad de Princenton. Agrupa frases y palabras en conceptos a los que se le asignan identificadores únicos que luego sirven para enlazar los diferentes conceptos a través de las diferentes relaciones semánticas.

Una vez que los datos se han introducido en la ontología, los usuarios pueden realizar consultas y generar, en lenguaje natural, biografías de artistas personalizadas con los datos de interés de los usuarios.

En Surdeanu *et al.* (2003) se hace una comparación entre usar métodos estadísticos y árboles de decisión para extraer información de los textos. Ambos paradigmas requieren de un *parsing* total de los textos y luego un etiquetado manual de cada estructura de predicados a un concepto a extraer. Sobre estas estructuras se aplican dos tipos de clasificadores: basados en métodos estadísticos y basados en árboles de decisión, que determinan el conjunto de características importantes a considerar para cada concepto. Según los resultados, los árboles de decisión obtienen mejores resultados.

4.2.3 Discusión

En la Tabla 4.1 se comparan cada uno de los algoritmos anteriormente descritos considerando los valores de ajuste obtenidos, los instrumentos para la automatización del sistema, los tipos de análisis lingüísticos requeridos, los conceptos y/o relaciones descubiertos en las experimentaciones y las herramientas lingüísticas que emplean. La tabla está ordenada cronológicamente y puede apreciarse que no hay estabilidad ni en cuanto a los valores de ajustes obtenidos ni en cuanto al instrumento para la automatización del sistema, aunque los algoritmos inductivos son los más recurridos.

Por otro lado, a excepción de los métodos probabilísticos, el resto requiere análisis sintáctico completo y en algunos casos se extiende a análisis semántico, incluyendo la solución de las coreferencias y las anáforas. Casi todos los métodos necesitan de un conjunto de instancias de entrenamiento descritas manualmente y de herramientas lingüísticas externas que apoye la tarea. Las herramientas externas más empleadas son bases de datos de nombres de personas, lugares, etc., diccionarios de sinónimos y para los métodos probabilísticos servicios web de buscadores genéricos para el cálculo de la frecuencia de las diferentes entidades.

En las experimentaciones, las instancias tratadas normalmente son entidades nombradas como nombres de personas, organizaciones, fechas. Una excepción lo constituyen la aplicación a documentos jurídicos en Amardeilh *et al.* (2005), Semtag de Dill *et al.* (2003) y Smartweb de Buitelaar & Ramaka (2005).

En cuanto a los valores de ajuste, se ha de mencionar que para casi todos los sistemas en los que se emplean conjuntos de entrenamiento y análisis sintáctico completo, logran rebasar el 85% de precisión. Algunos de los sistemas ofrecen una precisión superior al 95% y una recuperación

Tabla 4.1: Algoritmos de anotación semántica. **CE**=Conjunto de entrenamiento; Co_r=Co-referencias; **Sin**=Análisis Sintáctico; **IC**=Instancias complejas. Tipo I=Inducción, Pa=Patrones, Pr=Probabilístico, R=Reglas manuales. (1)=Anáfora; (2)=Parser HTML; (3)=Análisis Semántico.

Método	**Tipo**	**Prec**	**Rec.**	F_1	**CE**	Co_r	**Sin**	**Tipo de Instancias tratadas**	**IC**	**Herram. Ling.**
Kogut & Holmes (2001) (AeroDAML)	R					si	si	organizaciones, nombres personas, medidas	No	No desc., posiblemente sí se usan
Baumgartner *et al.* (2001) (Lixto)	R				no	no	(2)	precios, productos	No	No
Vargas-Vera *et al.* (2001, 2002) (MnM)	I	0.95	0.90		si	no	si	Historietas KMi	No	No
Ciravegna *et al.* (2002) (Amilcare & Melita)	I	0.9	0.75		si	no	si	ciudad, países, compañías	No	No desc., posiblemente sí se usan
Handschuh & Staab (2002); Handschuh *et al.* (2001) (Onto-O-Mat)	I				si	no	si	publicaciones		
Maedche *et al.* (2002)	Pa				si	no	si	depende de la ontología	1 nivel	BD lingüísticas
Popov *et al.* (2003) (KIM)	Pa	0.86	0.84	0.85	si	si	si	organizaciones, nombres personas, fechas	No	BD lugares, personas, países
Dill *et al.* (2003) (SemTag)	Pr	0.82	-	0.49	si	no	no	depende de la onto	No	sinónimos, funciones de ajuste
Yang & Choi (2003)	Pr	1.0	1.0		si	no	no	precio, título, ISBN		
Alani *et al.* (2003) (Artequakt)	I	0.85	0.42			si (1)	si	Datos de artistas	1 nivel	
Surdeanu *et al.* (2003)	I	0.91		0.91	si	no	si	mercado, defunciones	No	
Maynard *et al.* (2004) (h-TechSight)	R	0.97	0.92			si	si (3)	nombres personas, fechas, trabajos en el área química	No	BD
Cimiano *et al.* (2004) (Pankow)	Pr	0.49	0.49		no	no	no	ciudad, país, compañía	No	Servicios web
Valarakos *et al.* (2004)	Pr		0.97		si	no	no	depende de ontología, laptop	No	sinónimos
Ciravegna *et al.* (2004) (Armadillo)	I	0.98	0.8		si	si	si	personas, home pages	1 nivel	lexicón varios servidores web
Celjuska & Vargas-Vera (2004) (OntoSophie)	I	0.95	0.53		si	no	si	depende de onto, conferencia, premios, visitantes	No	
Buitelaar & Ramaka (2005) (SmartWeb)	Pa	0.89	0.92		si	no	si	depende de onto	1 nivel	
Amardeilh *et al.* (2005)	R	0.79	0.55	0.85	no	si	si	documentos jurídicos	Si	Lexicón

también alta (por encima del 80%), aunque en este caso por lo general tratan dominios específicos y/o entidades bien definidas, como en Maynard *et al.* (2004).

Sin embargo, los sistemas que no realizan un análisis sintáctico completo, el ajuste es inferior a 82% y si además, no se emplean conjuntos de entrenamiento ocurre una alta disminución de la precisión y recuperación como puede apreciarse comparando los sistemas Pankow y Amilcare.

Los porcentajes de recuperación obtenidos varían sin asociarse directamente ninguno de los parámetros analizados.

Por último, los sistemas analizados asumen el tratamiento de ontologías tipo frames, o sea, no consideran ontologías en lógicas descriptivas. Y sólo en cuatro casos se analiza el problema de descubrimiento de relaciones y la creación de instancias complejas y en todos ellos, sólo se consideran asociaciones entre instancias que involucran caminos con una única relación (1 nivel). O sea, la mayoría de los sistemas actuales sólo permiten anotar las instancias de referencia, pero no las relaciones entre ellas.

A modo de resumen, puede destacarse el predominio en el uso de conjuntos de entrenamiento y análisis sintáctico completo (a pesar de la dificultad implícita de crear estos conjuntos de entrenamiento o la demora y relativa inexactitud de los analizadores sintácticos), la poca diversidad de entidades tratadas, niveles de precisión y recuperación relativamente altos (alrededor de un 85%) y la inexistencia de un sistema que trate instancias ontológicas complejas como las descritas a través de la lógica descriptiva.

Lo hecho hasta el momento es un gran paso para el poblamiento de la Web Semántica, pero aún es imprescindible cierto procesamiento que permita no sólo reconocer las instancias, sino además formar de ellas una (o varias) **instancia compleja** que describa un ente ontológico completo con los detalles que se mencionan en el texto. Además, debe considerarse la posibilidad de realizar anotaciones semánticas que no requieran conjuntos de entrenamiento y análisis sintáctico y semántico complejos y sí que empleen las informaciones propias de las ontologías para descubrir y anotar instancias complejas.

4.3 Propuesta de población de la Web Semántica

Un 40% de las ontologías encontradas a partir de la página http://www.daml.org/ontologies/ contienen sólo la especificación terminológica del dominio de conocimiento que ellas expresan, y el total de instancias no sobrepasa 65% del total de las clases, siendo por tanto la definición axiomática de instancias practicamente nula[1]. Una metodología completa para poblar dichas ontologías es la que se describe en la Figura 4.2.

[1]Este no es un caso aislado, sino el comportamiento generalizado en las librerías de ontologías disponibles.

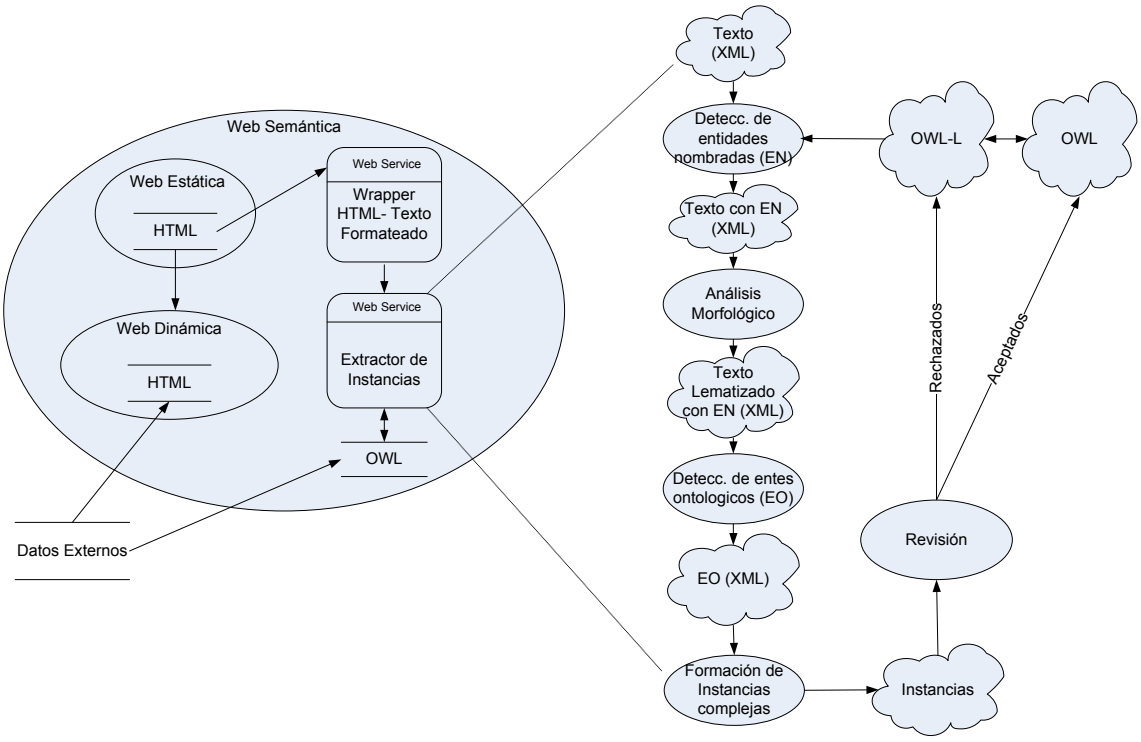


Figura 4.2: Propuesta de proceso de población de la Web Semántica.

El proceso de población ontológica para la Web Semántica discierne entre la presencia de la Web dinámica, la estática y fuentes externas a la web cuya información se desea exponer públicamente a través de la Web Semántica. La información contenida en la web dinámica y en muchas fuentes externas es generalmente mantenida en formatos estructurados (por ejemplo bases de datos) y por tanto, su incorporación a la Web Semántica tiene que pasar necesariamente por la comprensión de las estructuras de datos de almacenamiento y su alineamiento con ontologías que reflejen su contenido semántico. De ahí, que la población de la Web Semántica basada en tales datos, es en muchos casos, más factible desarrollarla desde las mismas entidades corporativas donde es conocida la semántica y el dinamismo de los datos. Sin embargo, la información almacenada en lenguaje natural debe ser tratada de manera diferente.

La metodología propuesta supone la existencia de un *wrapper* o servicios web que permiten obtener a partir de páginas HTML o ficheros textos de cualquier formato, textos XML equivalentes que reflejan la estructura sintáctica (sus párrafos, secciones, capítulos, partes, etc.) de los textos originales. Finalmente el *Extractor de Instancias* permitirá extraer las instancias descritas en los textos. Para ello, las siguientes operaciones son realizadas secuencialmente:

1. *Detección de entidades nombradas*, permite encontrar en el texto aquellas *unidades de in-*

formación[1] relacionadas con la ontología que necesitan de la sintaxis del texto para su recuperación. Ejemplo de éstos elementos son: números, fechas, nombres de personas, instituciones, lugares. La salida dada por la aplicación de este paso, es un diccionario (que es mantenido en el mismo fichero XML) que contiene las apariciones de todas las entidades nombradas encontradas. Por ejemplo, en el ámbito de una ontología sobre arqueología los yacimientos, unidades estratigráficas y números de inventario pueden ser reconocidos en este paso.

2. *Análisis morfológico simple*, convierte cada párrafo del XML en una lista de lemas y entidades nombradas que se asocian con las posiciones de las palabras en texto original. Como paso previo, se excluyen de este proceso aquellas palabras que pertenecen a una lista de parada.

3. *Detección de entes ontológicos*, recupera las restantes unidades de información de la ontología (aquellas que no requieren de la sintaxis del texto) empleando funciones de similitud y listas de palabras que caracterizan a dichas unidades de información. Considerando éstas unidades y las recuperadas en el primer paso, se crea un XML que contiene para cada párrafo sólo aquellas unidades informacionales (y sus referencias en el texto original) para las que se corrobore que la intensión de su uso es la mención de algún concepto, relación o valor de la ontología. Como es de esperarse, este proceso es fundamental, pues dependiendo del buen ajuste de su salida, así será la bondad de los resultados finales.

4. *Formación de instancias complejas*, emplea las anotaciones del paso anterior para crear adecuadamente las cadenas de relaciones entre conceptos de manera que las descripciones formadas se correspondan con instancias válidas en la ontología y estén semánticamente muy relacionadas con las descritas en el texto.

5. *Revisión y retroalimentación*, un último paso para la población de una ontología es precisamente la actualización de la ontología con las instancias formadas. Para ello, un experto del dominio de aplicación, debe dar “el visto bueno” a aquellas instancias que sean semánticamente equivalentes a las descritas en el texto y señalar los errores y omisiones del procesamiento. Las instancias correctas son adicionadas directamente en la ontología y los errores y omisiones empleados para retroalimentar el sistema, que mantendrá un conjunto de reglas de excepción, y moderadamente influirá en el lexicón[2] descrito para la ontología.

[1] Léase como bloque de palabras indisoluble que expresa algún elemento particular en la ontología.

[2] Como se muestra en la siguiente sección el buen funcionamiento de este método depende de un conjunto de definiciones léxicas asociadas a la ontología, lo que se conoce como lexicón del dominio o simplemente lexicón.

Justificación del método propuesto

El método propuesto está basado en dos criterios fundamentales: el análisis gramatical del texto debe ser muy simple, y la ontología puede servir de guía durante todo el proceso de creación de las instancias.

Basados en el hecho de que un análisis sintáctico-semántico completo del texto, no garantiza una mejor interpretación de éste, Grishman (1997), el método propuesto realiza un procesamiento muy simple, que extrae las raíces morfológicas de las palabras (lo que reduce el número de formas gramaticales a revisar) y elimina las palabras de una lista de parada, considerando la baja influencia, en la mayoría de los casos, de dichas palabras para la comprensión de la esencia del texto.

Por otro lado, tenemos que el proceso de población de la Web Semántica requiere formar instancias asociadas a ontologías. La definición formal de dichas ontologías no sólo permite reconocer cómo formar las instancias descritas en un texto, sino además explicita qué relaciones son de interés para la ontología particular. De ahí que, sea prácticamente natural el emplear la definición formal de la ontología para guiar el proceso de extracción de instancias ontológicas. Emplear un enfoque abierto, como los descritos en PLN, requiere un estudio más minucioso del texto y proceso de desambiguación que en la mayoría de los casos requerirán de un lexicón del dominio (que en parte lo constituye la propia ontología).

El proceso de formación de instancias complejas permite relacionar instancias a través de los caminos de agregación entre los conceptos ontológicos, que como se mostrará en la sección 4.4.3 permiten encontrar asociaciones con muy variadas formas, lo que hasta el momento no es capaz de realizar ninguno de los sistemas revisados en la literatura.

Por último, el proceso de revisión y retroalimentación permitirá a los usuarios del sistema aumentar/reducir el lexicón asociado a una ontología, así como variar los valores de relevancia de las instancias extraídas de acuerdo al ajuste de los patrones encontrados y experiencias en las extracciones anteriores. Este último paso ha sido sugerido en varios trabajos, por ejemplo en Ciravegna *et al.* (2002), y la retroalimencaión es un conocido método de ajuste en los sistemas de inteligencia artificial. Sin embargo, hasta donde conocemos, no ha sido descrito qué tipo de información recuperar y cómo utilizarla para corregir los errores cometidos por los sistemas ni se ha reportado ningún resultado en sistemas de extracción de información que emplee las decisiones de los usuarios del sistema para ajustar los resultados del procesamiento.

4.4 Definiciones formales

Las cuatro subsecciones siguientes describen cada una de las tareas relacionadas con la extracción de instancias ontológicas, resumidas brevemente en la sección anterior. Todas las explicaciones de esta sección están referidas a un párrafo del texto. En la sección 4.4.3 se describe cómo extender el procesamiento al texto completo.

4.4.1 Detección de entidades nombradas y análisis morfológico

Tal como vimos en la sección 4.3, el proceso de extracción de instancias ontológicas está guiado por la ontología y presupone el uso de especificaciones léxicas que le ayudan a reconocer los elementos de la ontología.

Sin embargo, la definición ontológica de un dominio de conocimiento, no impone restricciones para la designación de nombres de conceptos y relaciones. Ello significa que la especificación ontológica resultante puede ser poco comprensible, pues, en la gran mayoría de los casos, cada concepto y relación se denomina con uno de sus posibles sinónimos en un idioma particular. Garantizar un uso realmente satisfactorio de una definición ontológica obligaría a "traducir" los nombres de conceptos y relaciones de una definición ontológica en entidades léxicas realmente comprensibles por todos e independientes del idioma.

Por otro lado, las base de conocimiento tipo SHOIQ(D) consideran la posibilidad de emplear tipos de datos cualesquiera en la especificación de conceptos. Sin embargo, sólo algunos tipos de datos son reconocidos en los sistemas de razonamiento y/o lenguajes de la lógica descriptiva. OWL, por ejemplo sólo reconoce tipos de datos como enteros, reales o cadenas de caracteres y subconjuntos de éstos (aunque no da errores al compilar especificaciones ontológicas que describen tipos de datos no reconocibles).

En este trabajo se sugiere suplir tales problemas con el desarrollo paralelo tanto de ontologías como de sus especificaciones léxicas. Una especificación léxica no es más que la descripción de las asociaciones entre un concepto, tipo de dato o relación de una base de conocimiento y las cadenas de caracteres con que pudieran especificarse. Esta asociación debe permitir en principio que: 1) pueda usarse una definición extensional de las cadenas de caracteres asociados a cada elemento de la ontología; y 2) pueda evaluarse la aproximación de los emparejamientos encontrados durante el proceso de extracción de instancias ontológicas. Supondremos la existencia de funciones, llamémosle f_{verif}, que evalúe en cuánto una cadena de caracteres se corresponde con el elemento de la ontología con el que se le asocia. En el ámbito de la Web Semántica podemos definir una ubicación única tanto para los conceptos, tipos de datos y relaciones de una ontología

como para las funciones f_{verif} mediante las URI. La siguiente definición, resume y formaliza una especificación léxica para una ontología.

Definición 4.1. Sea $\mathcal{O}$ una ontología, cuya definición ha sido expuesta en las direcciones URI que aparecen en el conjunto $URIs_{\mathcal{O}}$. La especificación léxica de $\mathcal{O}$ es el conjunto de instancias de clase *EnteLexico* de la siguiente ontología concreta (a la que llamaremos *OntoLexicon*). Cada instancia de *EnteLexico* se refiere a algún ente (concepto, tipo de dato o relación) en $\mathcal{O}$:

$$\begin{aligned} EnteLexico \equiv \quad & = 1Ente.URI_E \sqcap \\ & \sqcap \exists Deflexica.(String \cup URI_{verif}) \sqcap = 1Deflexica.URI_{verif} \end{aligned}$$

donde URI_E es la especificación de una dirección *URI* en la cual ha sido descrito un ente empleando el lenguaje OWL y URI_{verif} es la especificación de una dirección donde reside un servicio web capaz de recuperar desde texto una cadena de caracteres asociada a un ente y/o evaluar en cuánto dicha cadena de caracteres se asocia al ente en cuestión.

A partir del Abox asociado a *OntoLexicon*, $\mathcal{A}_l$, y el conjunto de URIs asociados a $\mathcal{O}$, $URIs_{\mathcal{O}}$, se definen las funciones de entidades léxicas nombradas, f_{l_n}, y entidades léxicas no nombradas, $f_{l_{nn}}$, de la siguiente manera:

- $f_{l_n} : N_C \cup \Phi \cup N_R \rightarrow URI$ y se forma a través del conjunto:

$$\begin{aligned} \{\langle e,b\rangle \,|Ente(e,uri_e) \in \mathcal{A}_l \text{ tal que } e \in_* EnteLexico, uri_e \in URIs_{\mathcal{O}} \wedge \\ \forall b, DefLexica(e,b) \in \mathcal{A}_l, b \in_* URI_{verif}\} \end{aligned}$$

- $f_{l_{nn}} : N_C \cup \Phi \cup N_R \rightarrow 2^{String} \times URI$ y se forma a través del conjunto:

$$\begin{aligned} \{\langle e,S,b\rangle \,|Ente(e,uri_e) \in \mathcal{A}_l \text{ tal que } e \in_* EnteLexico, uri_e \in URIs_{\mathcal{O}}, \\ DefLexica(e,b) \in \mathcal{A}_l, b \in_* URI_{verif} \wedge \\ S = \{s | \exists DefLexica(e,s), s \in_* String\} \wedge S \neq \emptyset\} \end{aligned}$$

Por último llamaremos conjunto de cadenas léxicas de la ontología al conjunto:[1]
$S_l = \bigcup_{\substack{\forall e \in dom(f_{l_{nn}}), \\ f_{l_{nn}}(e)=(b,S)}} S$ y a cada elemento en S_l, cadena léxica.

Esta definición describe un ente léxico como aquel que asocia un ente (concepto, tipo de dato o valor) de la ontología (a través de la URI en la que puede ser encontrada su descripción) con una función de verificación y quizás también con un conjunto de cadenas que permiten reconocer a dicho ente. Esta definición permite formar dos funciones: una que agrupa entes que contienen sólo la función de verificación y otra cuyo dominio contiene a los restantes entes. La primera función,

[1] Como es usual $dom(f)$ e $Img(f)$ representan los conjuntos dominio e imagen de alguna función f.

función de entidades léxicas nombradas, tiene como imagen la URI de las funciones de verificación y deben permitir reconocer en un texto la presencia del ente que se le asocia (dado que no se cuenta con cadenas de caracteres que describan extensionalmente al ente) y evaluar la fiabilidad del emparejamiento. Ejemplo de estas funciones son: expresiones regulares y el reconocimiento de fechas, lugares, organizaciones, como es hecho en GATE. La segunda función, la de entidades léxicas no nombradas, tiene como imagen pares formados por un conjunto de cadenas y la URI de la función de verificación. De esta forma para cualquier ente podría describir una lista de cadenas léxicas que le identifiquen y mediante la función de verificación podría evaluarse la aproximación entre una zona del texto y alguna cadena léxica asociada al ente. Por último, se mantiene a través de S_l la lista de todas las cadenas léxicas de una ontología.

En OWL existen los llamados campos de anotaciones que pueden ser empleados para este propósito. Sin embargo, en el presente trabajo, asumiremos la existencia de especificaciones léxicas como en la definición anterior para facilitar la comprensión de las formalizaciones subsecuentes. Por otro lado, es importante notar que estas especificaciones léxicas subsumen teóricamente las especificaciones de los RDF esquemas referidas a tipos de datos pues a través de la especificación léxica es posible definir cualquier tipo de datos, tanto de manera extensional o por el uso de funciones de verificación.

En lo sucesivo supondremos se cuenta con las funciones f_{l_n} y $f_{l_{nn}}$, dada alguna especificación léxica y una ontología $\mathcal{O}$.

El reconocimiento de las entidades nombradas se describe en el Algoritmo 4.1 y como puede verse, el procesamiento realizado es recorrer el texto de manera que puedan ser recopilados en un diccionario las posiciones de los entes nombrados (de mayor longitud y máxima relevancia entre los que se solapan) que aparecen en los textos.

Por su parte, el análisis morfológico se circunscribe a:

1. eliminar del texto todos los signos,
2. eliminar del texto las palabras de una lista de parada y
3. extraer los lemas de las palabras del texto actual con el objetivo de disminuir las diferentes formas con las que se puede expresarse un mismo ente.

4.4.2 Detección de entes ontológicos

El paso siguiente corresponde a encontrar todas las referencias de conceptos y relaciones de la ontología escritas en el texto, así como desambiguar aquellas que lo requieran. Como paso previo, debemos realizar el análisis morfológico de todas las cadenas léxicas de la ontología, tal como se había descrito en la sección 4.4.1. De esta forma, se puede encontrar las intersecciones entre

Algoritmo 4.1 Extracción de entidades nombradas

Entradas: T, f_{l_n}, R_{min}

{ T, Texto a procesar;
f_{l_n}, función de entidades nombradas;
R_{min}, mínimo valor de relevancia de una cadena de caracteres para un ente.
}

Salidas: EN

{EN, diccionario que relaciona posición de aparición y lista de posibles entidades ontológicas referenciadas. Será usada la nomenclatura $Llaves(EN)$ para recuperar las llaves del diccionario y cada entrada, $EN[k]$, $k \in Llaves(EN)$ del diccionario es subconjunto de $dom(f_{l_n})$.}

for all $e \in dom(f_{l_n})$ **do**

- Recorrer T y detectar las posiciones donde e ocurre, y sus valores de relevancia de acuerdo a la función en $f_{l_n}(e)$
- Seleccionar las posiciones en los que su valor de relevancia sea mayor de R_{min}.
- Actualizar EN de acuerdo a las ocurrencias de e encontradas manteniendo en $Llaves(EN)$ pares disjuntos de posiciones de apariciones de entidades nombrados de la forma: $\langle pos_{inic}, pos_{fin} \rangle$ y en las entradas de EN las listas de entidades nombradas con relevancias máximas entre las que se solapan para la posición en cuestión.

el texto y las cadena léxicas (ambos procesados) para seleccionar, en una primera fase, zonas del texto de longitud y relevancia máximas respecto a la cadena léxica a la que se le asocia; y en una segunda fase, que aquí llamamos de desambiguación, recuperar entes ontológicos para los que se concluya determinada intención semántica de acuerdo a la ontología.

El Algoritmo 4.2 describe el proceso de detección de entes ontológicos. Se apoya en la siguiente definición y en el Algoritmo 4.3 que se encarga de la fase de desambiguación.

Definición 4.2. Sea S_l el conjunto de cadenas léxicas de la ontología. Sea Am una función que permite realizar el análisis morfológico de cada entidad léxica de acuerdo a lo explicado en la sección 4.4.1. Definiremos L como el conjunto de lemas extraídos al aplicar Am a los elementos de S_l,

$$L = \bigcup_{\forall s \in S_l} Am(s)$$

y $DefLex^{-} : L \rightarrow 2^{N_C \cup \Phi \cup N_R \times 2^S}$ como la función que asocia cada lema $l \in L$ con subconjuntos de entes y cadenas léxicas de cada ente para las que puede ser extraído el lema l, o sea, $DefLex^{-}$ se describe por medio del conjunto:

$$\{\langle l, E_l \rangle | E_l = \{\langle e, S' \rangle | f_{l_{nn}}(e) = \langle S, b \rangle, S \supseteq S' = \{s | s \in S, l \in Am(s), l \in L\}\}\}$$

La definición anterior permite mantener en L el conjunto de lemas asociados a las cadenas léxicas de una ontología y en la función $DefLex^{-}$ la asociación entre cada uno de estos lemas y el conjunto de cadenas léxicas y entes donde éstos aparecen. Nótese que para cada lema se cuenta con un conjunto de pares ente-conjunto de cadenas léxicas asociadas al ente donde el lema aparece.

Definición 4.3. Sea I^e la especificación léxica de una ontología. Sea $T = w_1, \cdots, w_n$ un texto lematizado donde cada w_i, $1 \leq i \leq n$, se corresponde a un lema del texto original T' (o la palabra del texto original). Llamaremos lista de entes de la ontología en T, EO, a la lista devuelta al ejecutar el Algoritmo 4.2 sobre T, considerando los elementos de las entidades nombradas EN, R_{min}, el valor de relevancia mínima permitida, gap, una cantidad máxima de lemas que pueden ser ignorados entre cada par de lemas en una cadena léxica candidata y la función $f_{l_{nn}}$.

Desambiguación de entes ontológicos

La segunda parte del Algoritmo 4.2 está referido a la desambiguación de la lista de entes de la ontología, $EO = \oplus_{1 \leq i \leq n} EO_i$, pues cada bloque semántico EO_i puede contener múltiples elementos. Este proceso de desambiguación debe considerar:

- la importancia de cada ente ontológico (o una cadena léxica) en el dominio de conocimiento de la ontología,
- la relevancia semántica con respecto a los restantes entes ontológicos del texto,
- la distancia sintáctica entre los entes ontológicos.

El primero de estos aspectos considera él hecho de que un concepto específico de un dominio de aplicación, tiene prioridad respecto a un concepto que puede ser empleado en múltiples áreas de conocimiento. Los otros dos, capturan el nexo entre la descripción ontológica, la distancia semántica de los conceptos y la probabilidad de encontrarlos cerca en un texto. Es más probable que pares de conceptos con una distancia semántica pequeña (alta relevancia semántica relativa) se encuentren cerca en un texto que aquellos con mayor distancia semántica y sintáctica.

Calcular la importancia de un ente ontológico (o una cadena léxica) para un dominio de aplicación presupone correlacionar *la frecuencia de uso de sus definiciones léxicas para textos del dominio de aplicación y textos de cualquier dominio*. Podría pensarse en el empleo de *corpus especializados* que permitan calcular tal medida. Sin embargo, existen varios inconvenientes al considerar tal solución. Por un lado, el cálculo de un corpus para un dominio específico no es una tarea fácil, requiere tanto de un equipo de expertos del dominio de aplicación como de un equipo de lingüistas que puedan etiquetar correctamente los textos. Y por otro lado, mantenerlo actualizado puede ser muy difícil dependiendo del dinamismo del área de conocimiento.

Algoritmo 4.2 Extracción de entes ontológicos

Entradas: T, EN, sim, S_{min}, gap

{ T, texto ya procesado según el análisis morfológico descrito;
EN, entidades nombradas calculadas por Algoritmo 4.1;
R_{min}, mínimo valor de relevancia de una cadena de caracteres a un ente;
gap, máxima cantidad de palabras entre lemas;
$f_{l_{nn}}$, función de verificación de entidades léxicas no nombrados. }

Salidas: EO

{EO, Lista de entes de la ontología en T.}

Primera Parte: Coleccionar las entidades ontológicas más importantes.

$i = 0$

while $i < |T|$ **do**

if $\exists \langle i, posic_{fin} \rangle \in LLaves(EN)$ **then**

$EO_i = EN[\langle i, posic_{fin} \rangle]$

else if $T_i \in L$ **then**

Primer paso: Coleccionar todas las tripletas <cadena del texto, cadena léxica, ente léxico> *que pueden contener al lema* T_i.

$EO_i = \bigcup_{\forall \langle e,S' \rangle \in DefLex^{-}(T_i), \forall s \in S'} \langle Am(s) \pitchfork T_{i..i+gap*|Am(s)|}, s, e \rangle$ {$\pitchfork$ es la función de intersección entre conjuntos de palabras.}

Segundo paso: Filtrar las tripletas para mantener las de relevancia máxima.

$$EO_i = \{\langle s_t, s, e \rangle \,|\, f_{l_{nn}}(e) = \langle S, f \rangle, f(s_t, s) = max_{\substack{\forall \langle s'_t, s', e' \rangle \in EO_i \\ f_{l_{nn}}(e') = \langle S', f' \rangle}} f'(s'_t, s') \geq R_{min}\}$$

Tercer paso: Filtrar las tripletas de relevancia máxima para mantener sólo aquellas con máxima longitud.

$EO_i = \{\langle s_t, s, e \rangle \,||sf| = max_{\forall \langle s'_t, s', e' \rangle \in EO_i} |s'_t|\}$

$EO = EO \oplus EO_i$

Desplazar i hasta la posición mínima donde terminan las seudo-frases en EO_i.

Segunda Parte: Desambiguación.

Desambiguar cada elemento de la lista en EO empleando el Algoritmo 4.3.

Una solución alternativa puede ser encontrada por medio de la recuperación de documentos en la web, si consideramos:

- los documentos de la web como una gigantesca colección de documentos que hablan de cualquier dominio de aplicación y
- las respuestas a las consultas de búsqueda en la web, como una colección de documentos cuyo tema es definido por dicha consulta.

Específicamente la propuesta es definir consultas cuyas respuestas permitan inferir la importancia de una cadena léxica para un dominio de aplicación, a través de la estimación del número de documentos en la web en las respuestas a las demandas realizadas. A continuación se define una medida que permite estimar la importancia de una cadena léxica para el dominio de aplicación de la ontología al cuál está asociado.

La finalidad de esta medida es considerar, cuán útil es un ente ontológico, para reconocer que el documento que lo contiene, trata sobre el dominio de conocimiento de la ontología al que está asociado y por tanto pudiera ser empleado en la descripción de instancias ontológicas de algún texto.

Definición 4.4. Sea s una cadena léxica en S. Sea dom el dominio de una ontología. La relevancia de s en el dominio de la ontología viene dado por:

$$R(s,dom) = \begin{cases} 1 & \text{si } \frac{n_{s,dom}}{n_s}\frac{n}{n_{dom}} \geq 1+\beta \\ \frac{n_s}{n} & \text{en otro caso} \end{cases}$$

donde:

- $n_{s,dom}$: es el número de páginas web que contienen la cadena s y alguna de las cadenas que representan al dominio dom,
- n_s: es el número de páginas web que contienen la cadena s,
- n: es el total de páginas web,
- n_{dom}: es el número de páginas web que contienen alguna de las cadenas que representan al dominio dom.

Por un lado $\frac{n_{s,dom}}{n_s}\frac{n}{n_{dom}}$ establece la relación entre la frecuencia de aparición de una cadena léxica entre las páginas del dominio de la ontología y todas las páginas de la Web. Por otro lado, revisa en cuánto la distribución de la palabra difiere de la mínima distribución esperada para las palabras el dominio. Si $R(s,dom) < 1$, la cadena léxica por sí sola decididamente no es útil para el

dominio, en tanto si $R(s, dom) < 1 + \beta$[1], la palabra es decisiva para el dominio. Los casos restantes se corresponden a cadenas que son útiles en muchos dominios de aplicación, incluyendo el área de conocimiento de la ontología examinada. Un ejemplo claro lo constituyen los adjetivos, pues permiten describir propiedades de muchos tipos de objetos. En estos casos la importancia de la cadena léxica para el dominio de aplicación depende más bien de la frecuencia en que esta palabra aparece en toda la web, de ahí que su relevancia sea considerada a través de $\frac{n_s}{n}$.

Por último, para considerar posibles casos de indefiniciones en las ecuaciones anteriores, asumiremos, sin perdida de generalidad, que existe en la web una página más que contiene a todas las cadenas léxicas de la ontología. Así, una cadena que sólo aparezca en esta página "ficticia" obtendrá relevancia máxima (1) y se tiene que $Img(R) \in (0, 1]$.

La medida anteriormente descrita no solamente es importante porque podría ser utilizada como medida para realizar agrupamientos temáticos en la web, sino porque además es intuitivamente fácil de entender para personas poco relacionadas a las matemáticas (valor entre (0, 1] que explica si una cadena léxica muy útil y prácticamente sólo utilizada en el dominio). Esto último, podría permitir realizar aproximaciones de los valores de R dados por los usuarios en lugar de emplear los resultados devueltos por los buscadores. La motivación para tal actitud podría deberse a casos sospechosos que aparecen cuando se inspeccionan las estimaciones de números de páginas devueltas durante las consultas a buscadores como Google.

Ahora bien, si durante el proceso recuperación de un ente ontológico, aparece algún ente con un bajo valor de relevancia para el dominio y la significación de dicho ente tiene que ver con la ontología, entonces algún otro ente ontológico vecino (anterior o posterior a él) debe permitir la especificación de lo que se describe. Por ello, el primer paso en el proceso de desambiguación es calcular la importancia de cada ente ontológico encontrado en un texto T, $imp(e, T)$, considerando los entes ontológicos vecinos y filtra aquellos que se consideran redundantes o que no pertenecen al dominio de la aplicación.

La segunda parte del proceso de desambiguación se define como un proceso iterativo que permite seleccionar (si ello es posible) el ente ontológico más apropiado entre los múltiples encontrados en cada cada bloque semántico EO_i. Para ello, se consideran las distancias sintácticas y semánticas entre pares entes ontológicos del texto, donde uno de ellos es "seguro". Un ente ontológico $e \in EO_i$ es seguro si $|EO_i| = 1$.

Las siguientes definiciones describen el proceso de selección de un ente ontológico y el Algoritmo 4.3, se describe todo el proceso de desambiguación.

[1] En resultados experimentales $\beta > 0.2$.

Definición 4.5. Sea $EO = \oplus_{1\leq i\leq n} EO_i$ una lista de entes ontológicos del texto T. Llamaremos relevancia relativa entre $\langle sf_1, s_1, e_1\rangle \in EO_{i_1}$ y $\langle sf_2, s_2, e_2\rangle \in EO_{i_2}$, $i_1, i_2 \in 1,...,n$, denotada por $Rr(e_1,e_2)$, a la medida:

$$Rr(e_1,e_2) = R(e_1,e_2) * (1 - log_\gamma(|i_1 - i_2|))$$

siendo $R(e_1,e_2) = 1 - log_{|N_C|} dist(e_1,e_2)$ y $dist(e_1,e_2)$ es la distancia semántica entre los entes e_1 y e_2 de acuerdo al Tbox de la ontología y que pueden calcularse por medio de las siguientes ecuaciones:

$$dist(e_1,e_2) = min_{\substack{\forall e_1', e_2' \\ e_i' \in \{e_i\} \cup cls(e_i), \\ i \in \{1,2\}}} d(e_1') + 1/2 d_t(e_1, e_1') + 1/2 d_t(e_2, e_2')$$

donde[1]:

$$d(c_1,c_2) = min\{d_j(c_1,c_2), d_{ag}(c_1,c_2)\}$$

$$d_j(c_1,c_2) = \begin{cases} |\{c' | c_1 \sqsubseteq c' \sqsubseteq c_2\}| & \text{si } c_1 \sqsubseteq c_2 \\ |\{c' | c_2 \sqsubseteq c' \sqsubseteq c_1\}| & \text{si } c_2 \sqsubseteq c_1 \\ min_{\substack{c_a \in ancestros(c_1) \cap \\ \cap ancestros(c_2)}} d_j(c_a,c_1) + d_j(c_a,c_2) & \text{en otro caso} \end{cases}$$

$$d_{ag}(c_1,c_2) = min_{\forall cam \in CamsCC_i(c_1,c_2)}\{|cam|\}$$

$$cls(e) = \begin{cases} e & \text{si } e \in N_C \cup \Phi^D \\ \{C_{n-1} | \exists C, C' \in N_C, \text{tales que} & \\ \quad CamsCC_i(C,C') = \oplus_{1\leq i\leq n} \langle RS_i, C_i\rangle & \text{si } e \in N_R \\ \quad e \in RS_n\} & \end{cases}$$

$$d_t(e_1,e_2) = \begin{cases} 0 & \text{si } \{e_1,e_2\} \subseteq N_*, N_* \in \{N_R, N_C\} \cup \Phi^D \\ 1 & \text{si } e_i \in N_C \cup \Phi \wedge e_j \in N_R, i,j \in \{1,2\}, i \neq j \end{cases}$$

R_r permite reconocer la relevancia relativa entre dos entes ontológicos e_1 y e_2. Para ello, considera tanto la relevancia semántica entre los entes establecida por el Tbox como la relevancia inversa asociada a la distancia sintáctica entre los entes, definida mediante $1 - log_\gamma(|i_1 - i_2|)$, pues i_1 e i_2 son los índices donde parecen los entes ontológicos en la lista de entes ontológicos y no debe sobrepasar el valor de γ.

La distancia entre dos entes ontológicos definidos a través del Tbox, se define como la mínima distancia entre los entes, ya sea por la jerarquía de conceptos (d_j) ó por la definida a través de la agregación (d_{ag}). Si alguno de los entes es una relación, la distancia entre conceptos es calculada

[1] Nótese que los argumentos de d están en el conjunto $N_C \cup \Phi$, como resultado del uso de la función cls.

considerando todas las clases que se asocian a otra a través de la relación (ente). Este conjunto de clases es definido por *cls*.

Definición 4.6. Sea $e_{i_j} \in EO_i$. Llamaremos a e_{i_j} el ente seleccionado de EO_i si:

- e_{i_j} es seguro ó
- e_{i_j} cumple que:

$\neg\exists e_{i_k} \in EO_i, e_{i_j} \neq e_{i_k}$ tales que:
$\exists e_{l_s} \in EO_l, e_{h_m} \in EO_h, l, h \neq i$ tales que:
$Rr(e_{i_k}, e_{l_s}) > Rr(e_{i_j}, e_{h_m}) \vee$
$(Rr(e_{i_k}, e_{l_s}) = Rr(e_{i_j}, e_{h_m}) \wedge$
$(|l-i| < |h-i| \vee (|l-i| = |h-i| \wedge l < h)))$

O sea, un ente es seleccionado, si es el único en un bloque EO_i, ó si es entre los de su bloque el que mantiene con algún ente en otro bloque, máxima relevancia y mínima distancia.

Algoritmo 4.3 Desambiguación de entes ontológicos

Entradas: EO
$\{EO$, Lista de entes de la ontología en $T.\}$
Salidas: EO
$\{EO$, Lista de entes de la ontología en T tal que $\forall e \in EO_i$, e es seguro.$\}$

Primera Parte: Cálculo de la importancia de cada ente y filtrado.
for all $i \in 1..|EO|$ **do**
  for all $e_{i_k} \in EO_i$ **do**
    $imp(e_{i_k}) = \min\{1, \max\{R(e_{i_k}) + \max\{\{0\} \cup \{R(e_j) | e_j \in EO_{k-1}\}\},$
    $R(e_{i_k}) + \max\{\{0\} \cup \{R(e_j) | e_j \in EO_{k+1}\}\}\}\}$
  if $\max\{imp(e_{i_k}) | e_{i_k} \in EO_i\} > relevMin$ **then**
    $EO_i = \{e_{i_k} | imp(e_{i_k}) > relevMin\}$

Segunda Parte: Selección.
cambio $\leftarrow 1$
while *cambio* **do**
  cambio $\leftarrow 0$
  for all $i \in 1..|EO|$ **do**
    if $|EO_i| > 1 \wedge \exists e_{i_k} \in EO_i$, e_{i_k} el ente seleccionado de EO_i **then**
      cambio $\leftarrow 1$
      $EO_i = e_{i_k}$, e_{i_k} el ente seleccionado de EO_i

4.4.3 Formación de instancias complejas

A continuación se presenta el algoritmo de formación de instancias complejas. La idea general de éste es recorrer la lista de entes desambiguados formando por cada clase una nueva instancia vacía (o sea, sin relaciones) y por cada relación una instancia (con al menos una relación) si existiese una única clase que hace referencia a dicha relación. En la medida que se forman nuevas instancias éstas intentan completar su descripción realizando unificaciones, y agregaciones, siempre que les sea posible.

Más específicamente el Algoritmo 4.4 utiliza las reglas en la Figura 4.3 para crear las instancias complejas. Las reglas *CC* y *CR* son aplicadas con la máxima prioridad con el objetivo de formar las instancias iniciales. Cada instancia creada llama al procedimiento *Completa* para completar su descripción. Para ello, recorre los entes posteriores a él y para cada uno de ellos primero, crea la instancia apropiada de la clase (nuevamente mediante la regla *CC* y ahora considerando además la regla CR_1) y luego se revisan los siguientes casos:

1. si las clases de la instancia que se completa y de la nueva instancia creada mantienen una relación jerárquica, entonces se deben **unificar**, posiblemente en el texto se hable de una misma instancia, **utilizando "sinónimos"**;

2. si la clase de la instancia que se completa contiene un único camino hacia la clase de la nueva instancia creada, se debe **agregar** la segunda a la primera, posiblemente en el texto se está **describiendo** características de dicha instancia;

3. si la clase de la nueva instancia creada contiene un único camino hacia la clase de la instancia que se completa, se debe **agregar la instancia que se completa a la nueva instancia**, posiblemente instancia global que realmente se describe es la nueva, pero se comenzó por una característica particular;

4. si durante este proceso se obtiene una inconsistencia, es porque posiblemente en el texto se está comenzando la descripción de otra instancia, y ya se ha completado la instancia objeto.

Hasta el momento, la formación de instancias complejas ha sido tratado considerando un único párrafo. El algoritmo anterior puede extenderse de forma trivial para tratar con un texto completo. Para ello, sólo es necesario considerar todos los entes ontológicos del texto en una única lista y emplear la estructura sintáctica del texto para limitar hasta dónde extender las instancias. Por ejemplo, si en el título de un capítulo encontramos una instancia, su descripción no debe extenderse más allá del capítulo.

Algoritmo 4.4 Formación de instancias complejas

Entradas: EO

$\{EO$, Lista de entes desambiguados de la ontología en $T.\}$

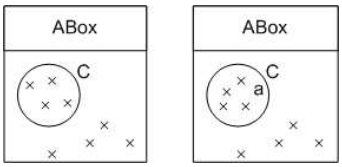
Salidas: EO

$\{EO$, Lista de entes de la ontología en T tal que $\forall e \in EO_i$, e es seguro.$\}$

$indexesExcluir = \emptyset$
$InstanceName = a, ClassName = C$
for all $i \in \{1,...,n\}$ **do**
  if $i \notin indexesExcluir$ **then**
    Sea $e \in EO_i$
    if $e \in N_R$ **then**
      Aplicar regla CR
    else if $e \in N_C$ **then**
      Aplicar regla CC
    if ha sido creada a **then**
      $indexesExcluir = indexesExcluir \cup Completa(a)$

function Completa(a, indexInic):
$InstanceName = a', ClassName = C'$
$indexesExcluir = \emptyset$
for all $j \in \{indexInic,...,n\}$ **do**
  $indexesExcluirAct = \emptyset$
  if $j \notin indexesExcluir$ **then**
    Sea $e \in EO_j$
    if $e \in N_C$ **then**
      Aplicar regla CC
    else if $e \in N_R$ **then**
      Aplicar regla CR_1
    if $C \sqsubseteq C' \vee C' \sqsubseteq C$ **then**
      Aplicar regla $Unif$
    else if $\exists! cam \in CamsCC_i(C, C')$ **then**
      Aplicar regla $Agreg$
      $indexesExcluirAct = Completar(a', j+1)$
    else if $\neg\exists cam \in CamsCC_i(C, C') \wedge \exists! cam \in CamsCC_i(C', C)$ **then**
      Aplicar regla $Agreg_1$
      $indexesExcluirAct = Completar(a', j+1)$
    $indexesExcluir = indexesExcluir \cup indexesExcluirAct$
    if el Abox no cambió debido a una inconsistencia **then**
      Salir

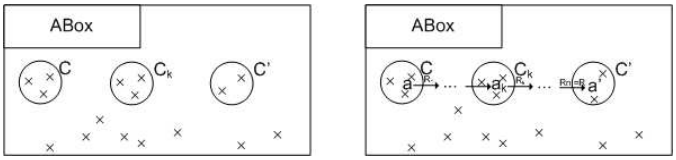
CC : Regla de creación por mención de clase C, $Abox = Abox \cup \{C(a)\}$



CR : Regla de creación por mención de relación $e \in N_R$

if $\exists! cam = \oplus_{1 \leq j \leq m} \langle RS_j, C_j \rangle \in \bigcup_{\forall C,C' N_C \times N_C} CamsCC_i(C,C')$tal que$\exists RS_k = \{e\}, k \in \{1,...,m\}, cam \in CamsCC_i(C,C')$ **then**

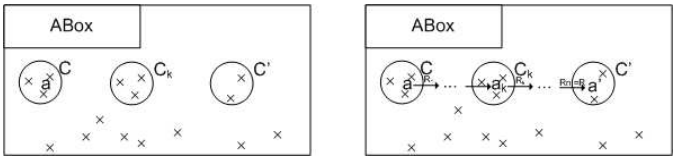
$Abox = Abox \cup \{C(a), C'(a'), d(a \leftarrow a'\}$



CR_1 : Regla de creación por mención de clase C y relación $e \in N_R$

if $\exists! cam = \oplus_{1 \leq j \leq m} \langle RS_j, C_j \rangle \in CamsCC_i(C,C')$tal que$\exists RS_k = \{e\}, k \in \{1,...,m\}$ **then**

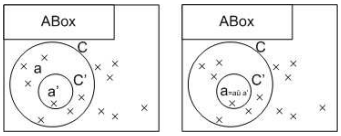
$Abox = Abox \cup \{C'(a'), d(a \leftarrow a'\}$



$Unif$: Regla de unificación de instancias $a \in_* C$ y $a' \in_* C'$

if a y a' son complementarias **then**

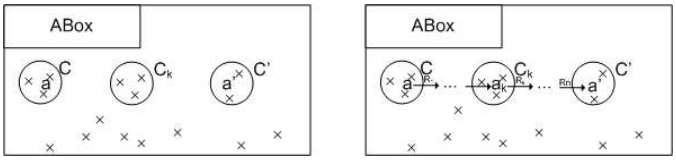
$a'' = a \widehat{\cup} a'$; $Abox = Abox \setminus \{C(a), C'(a), d(a), d(a')\} \cup \{C'(a''), d(a'')\}$; $a = a''$



$Agreg$: Regla de agregación de instancia $a' \in_* C'$ a $a \in_* C$

if a aparece antes que a' y $\exists! cam \in CamsCC_i(C,C')$ **then**

$Abox = Abox \setminus \{C(a), C'(a), d(a), d(a')\} \cup \{d(a \leftarrow a')\}$ si el Abox se mantiene consistente.



$Agreg_1$: Regla de agregación de instancia $a \in_* C$ a $a' \in_* C'$

- Intercambiar a y a' {Como consecuencia: a es la instancia agregadora y a' la agregada}
- Aplicar regla *Agreg*

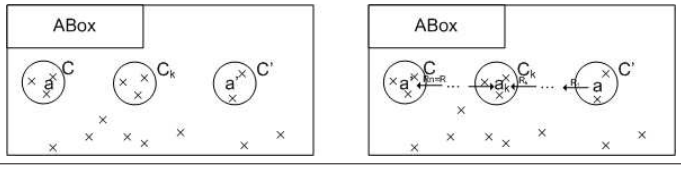


Figura 4.3: Reglas de transformación del Abox.

4.4.4 Revisión y retroalimentación

El último paso para la extracción de instancias ontológicas requiere que usuario señale los errores y/u omisiones que ha tenido el sistema durante el proceso de extracción de instancias. El objetivo se divide en dos vertientes: ajustar el valor de la relevancia de las cadenas léxicas asociadas a un ente para el dominio de la ontología y añadir datos al lexicón y revisar posibles fuentes de errores durante la formación de instancias. De acuerdo a ello se reconocen los siguientes casos:

1. *error en las entidades encontradas*, en este caso se podría tener que:

   - el ente se asocie a una cadena léxica y el ajuste de la relevancia de la cadena léxica podría resolver el problema, así $R(s,dom) = R(s,dom) - \beta \times \frac{t_e}{t}$, siendo t_e el total de veces en que se encontrado el error del total t de veces que la cadena se ha encontrado y $\beta \in [0,1]$, un valor que permite regular en cuánto los errores del sistema influirán en el ajuste de la relevancia de las cadenas léxicas;
   - o por el contrario ha sido determinada por una función de verificación en URI_{verif}. En este caso el proveedor del servicio web debe ser informado del error y realizar los ajustes pertinentes. Alternativamente, el sistema local podría ajustar la relevancia igual que en cuando se está en el caso de una cadena léxica.

2. *omisión en las entidades encontradas*, en este caso el usuario deberá señalar las entidades que no han sido localizadas con el lexicón actual. Si son entes no nombrados las cadenas léxicas asociadas podrán ser añadidas al lexicón actual y trabajar en lo sucesivo considerándolas. En caso contrario, el proveedor del servicio web debe ser informado y no se podrá garantizar el reconocimiento del ente en sucesivas iteraciones si el proveedor no realiza los ajustes necesarios.

3. *error durante el proceso de formación de instancias complejas*, el usuario debe señalar las instancias incorrectas (cuando la clase que se les asocia no es la adecuada) o las relaciones incorrectas. En ambos casos debe reconocerse las instancias iniciales empleadas en el proceso de completamiento de éstas y caracterizar el texto entre ellos para crear y mantener reglas de excepción para la unificación y agregación. La caracterización de los textos, en una primera instancia, puede contar con los siguientes parámetros:

   - Signos de puntuación: número y tipos de signos de puntuación en la zona del texto donde se encontraron las instancias iniciales.
   - Clases detectadas entre las unificadas o agregadas.

La unificación y agregación puede consolidarse si se considera baja la frecuencia de la caracterización del texto entre las instancias que se unifican o agregan, considerando las caracterizaciones de textos con errores encontrados en iteraciones anteriores.

4.5 Sistema implementado

En las Figuras 4.6 hasta 4.9 se muestran varias pantallas del prototipo desarrollado, el cual es una implementación java con posibilidades de integrar como plugin en Protegè. El sistema necesita como entrada:

- una ontología en OWL,
- lexicón,
- relevancia mínima permitida durante el proceso de extracción de entes ontológicos (R_{min} en los Algoritmos 4.1 y 4.2),
- distancia máxima de los entes ontológicos γ y
- texto a procesar.

Aunque todo el trabajo ha sido desarrollado para cualquier ontología, las experimentaciones y ejemplos de la presente y siguiente secciones han sido desarrollados para la ontología de Arquelogía. Ésta ha sido especificada utilizando el editor de ontologías Protegè[1] (ver en Figura 4.4), en la que también se han insertado las especificaciones del lexicón empleando los campos de anotaciones de OWL.

Las cadenas léxicas asociadas a las entidades no nombradas del lexicón se formaron empleando las mismas etiquetas de conceptos y relaciones descritas en la ontología, en un procesamiento off-line que considera la morfología de los identificadores utilizados en la mayoría de las aplicaciones. Ellos son:

- el uso de caracteres como: "-", "_" y "ˆ" y espacios para separar palabras y
- el uso de mayúsculas en las palabras más importantes del identificador.

Las entidades léxicas no nombradas, por su parte, se dividieron en dos grupos:

- entidades descritas por tipos de datos predefinidos (enteros, flotantes, etc.) y

[1] http://protege.stanford.edu/.

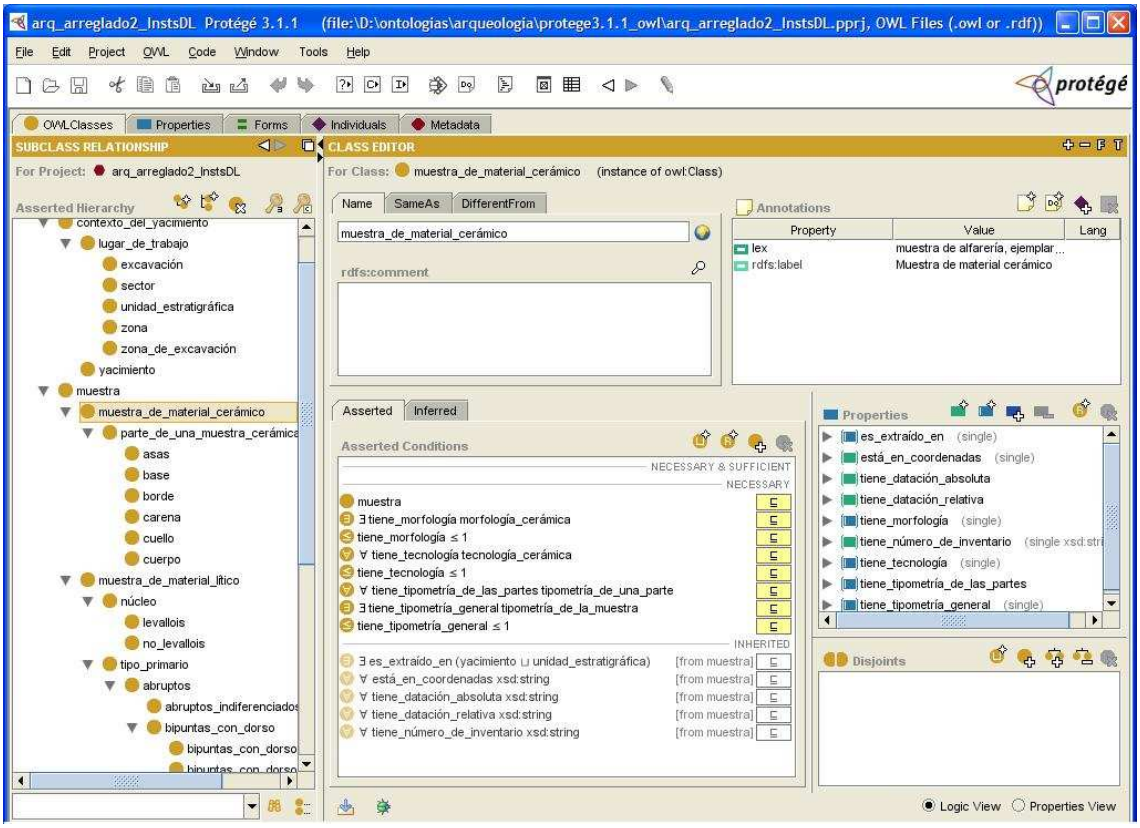


Figura 4.4: Ontología de arqueología descrita en Protegè. Específicamente, en esta pantalla se muestra una parte de la jerarquía de clases de la ontología (en el panel izquierdo) y la especificación DL de la clase *muestra_de_material_cerámico* (en el panel centro inferior). Puede verse además, en el panel derecho superior la propiedad de anotación *lex* que ha sido empleada para especificar el lexicón de la ontología.

- entidades nombradas específicas del dominio, por ejemplo: unidades estratigráficas, número de inventario, yacimiento, fechas, etc.

Afortunadamente, con sólo el uso de expresiones regulares pudieron recuperarse de modo satisfactorio todas estas entidades. Para ello, fue desarrollado un módulo, EEON, que permite componer expresiones regulares y asociarles las funciones de verificación. De esta forma, estas especificaciones pueden ser empleadas para reconocer entidades nombradas sobre las que se habla en el texto. Un pequeño ejemplo las especificaciones se describe en la Figura 4.5.

Nótese que aunque para la ontología de arqueología ha sido suficiente emplear este módulo, basta sustituirlo con otro que se considere oportuno en el caso de otras ontologías (por ejemplo, es conocida la dificultad de encontrar entidades de ontologías médicas, sólo la búsqueda de estas entidades es objetivo de numerosas y complejas investigaciones) y se mantiene la funcionalidad del sistema.

Los valores de relevancia mínima 0.6 y $\gamma = 50$ junto al lexicón descrito anteriormente, garantizaron resultados satisfactorios, tal como puede verse en la siguiente sección. A continuación se presentan algunos de los resultados obtenidos por el sistema, a través de la descripción de casos representativas.

Ejemplo 4.1. Entidades nombradas. Este es uno de los casos más simples con los que se trata. Tal como puede apreciarse en la Figura 4.6, a partir del descubrimiento de un nombre de yacimiento (su denominación) y un municipio se reconstruye una instancia compleja de 1 nivel que agrega un municipio a un yacimiento.

Ejemplo 4.2. La Figura 4.7 muestra un caso típico en el que se describe una muestra arqueológica. La instancia compleja que el sistema descubre engloba todos las especificaciones del párrafo. Nótese la complejidad del elemento tratado: se han agregado a la instancia compleja (muestra de material cerámico) instancias de tipo especializadas a él (como base y borde). En este caso el sistema ha relegado la unificación a la agregación. Por otro lado, ha formado una instancia compleja de 5 niveles y no sólo ha tratado frases bastante específicas del dominio, como “golpe de punzón”, sino que palabras muy frecuentes como “abundantes” no han sido descartadas durante el proceso de selección de entes. Por último, puede apreciarse un error que ocurre con cierta frecuencia. La ontología especifica que el tratamiento asociado a una muestra cerámica es único y se corresponde a valores como *engobe* o *espatulado*. Sin embargo, se ha encontrado una instancia cerámica que responde a varios tratamientos. Ello podría significar que la ontología no está correctamente descrita, pero en algunos casos ésta no es la explicación: puede tratarse de un comportamiento anómalo de un objeto que no ha sido modelado en la ontología, o aunque se conocía de la existencia de esta irregularidad se prefirió por obviarla durante la construcción de la ontología por considerarse una rareza.

```
/home/rox/instancias/fttotexto/FuncER.py

@LetMay==[A-ZÁÉÍÓÚÜÑÈÇ]
@LetMin==[a-záéíóúüñàòèç]

@PalLetInicMay==@LetMay@LetMin+
@PalLetInicMin==@LetMin{3,20}
@PalLetInicMin==@LetMin{1,20}

@De==del?
@Art==(el|las?|los|una?|unos|unas)

@Propio==(@PalLetInicMay )*@PalLetInicMay
@Sintagma==(@Art (@PalLetInicMin )+@De)+
@SLugar==@Sintagma (@Propio)

@Municipio=verifMunic=@PalLetInicMay( @PalLetInicMay)*(( @PalLetInicMin)+
                              ( @PalLetInicMay)+)?
@Yacimiento=verifYacim=@SLugar

@NumEnt=noRevisar=[]+

@Sector=revisarCalificativos['sector','area']=@LetMay@LetMin*(?:[-]?(@NumEnt|
                                                  -@NumRomano))?
@Nivel=noRevisar=[Nn](?:ivel)? ?– ?(@NumRomano|@LetMay+)
                          (?: ?@LetMay@LetMin*@NumEnt)?
@Corte=noRevisar=C ?– ?@NumRomano(?: ?@LetMay@LetMin*@NumEnt)?

@UnidadLong==[mcdk]?m.?
@UnidadArea==@UnidadLong(2|²)
@MedidaLong=noRevisar=(@NumReal|@NumEnt) @UnidadLong( de @PalLetInicMin)?
@MedidaArea=noRevisar=(?:(@MedidaLong|@NumReal|@NumEnt)
                               (x|por) @MedidaLong|(@NumReal|@NumEnt) @UnidadArea)
                               ( de @PalLetInicMin por @PalLetInicMin)?
```

Figura 4.5: Ejemplo de fichero de especificación de expresiones regulares empleado en el módulo de extracción de entes ontológicos nombrados (EEON).

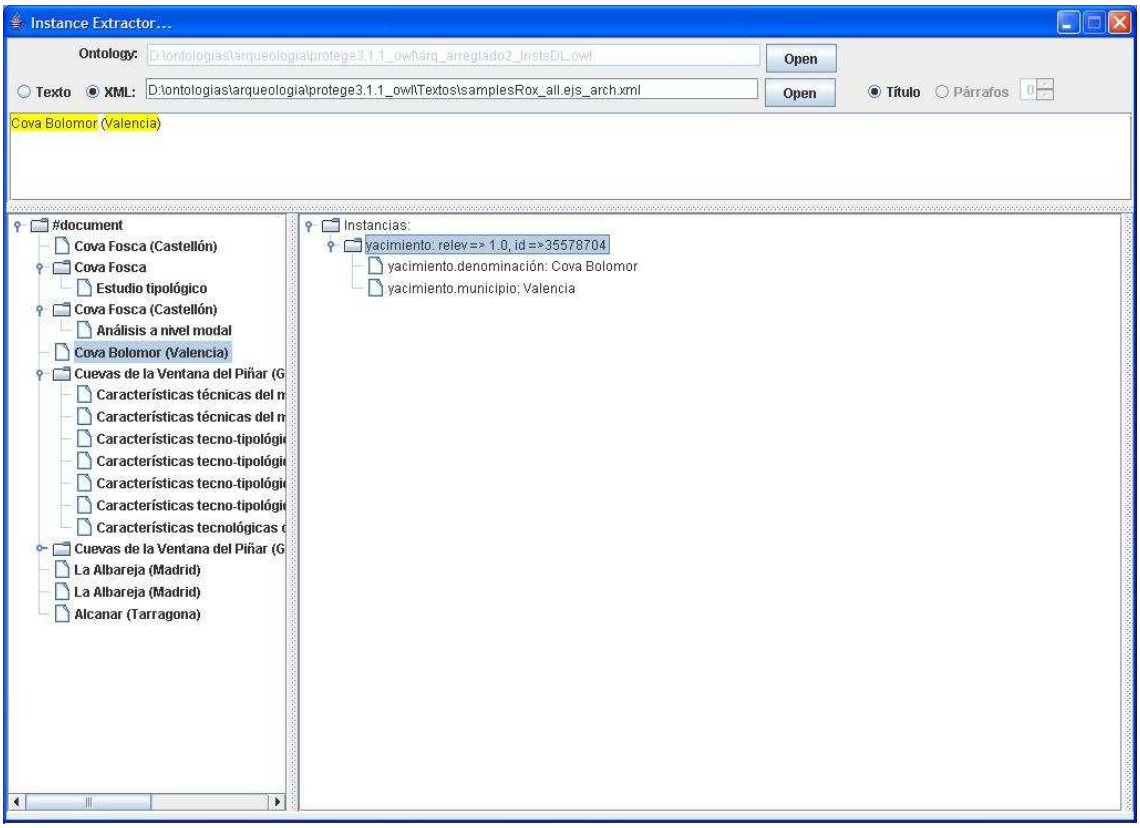


Figura 4.6: Ejemplo de extracción de entidades nombradas. La aplicación desarrollada toma como entrada una ontología y un texto ó un XML (con un formato particular que permite procesar varios documentos a la vez) y muestra en los paneles inferiores las instancias complejas recuperadas por cada párrafo procesado.

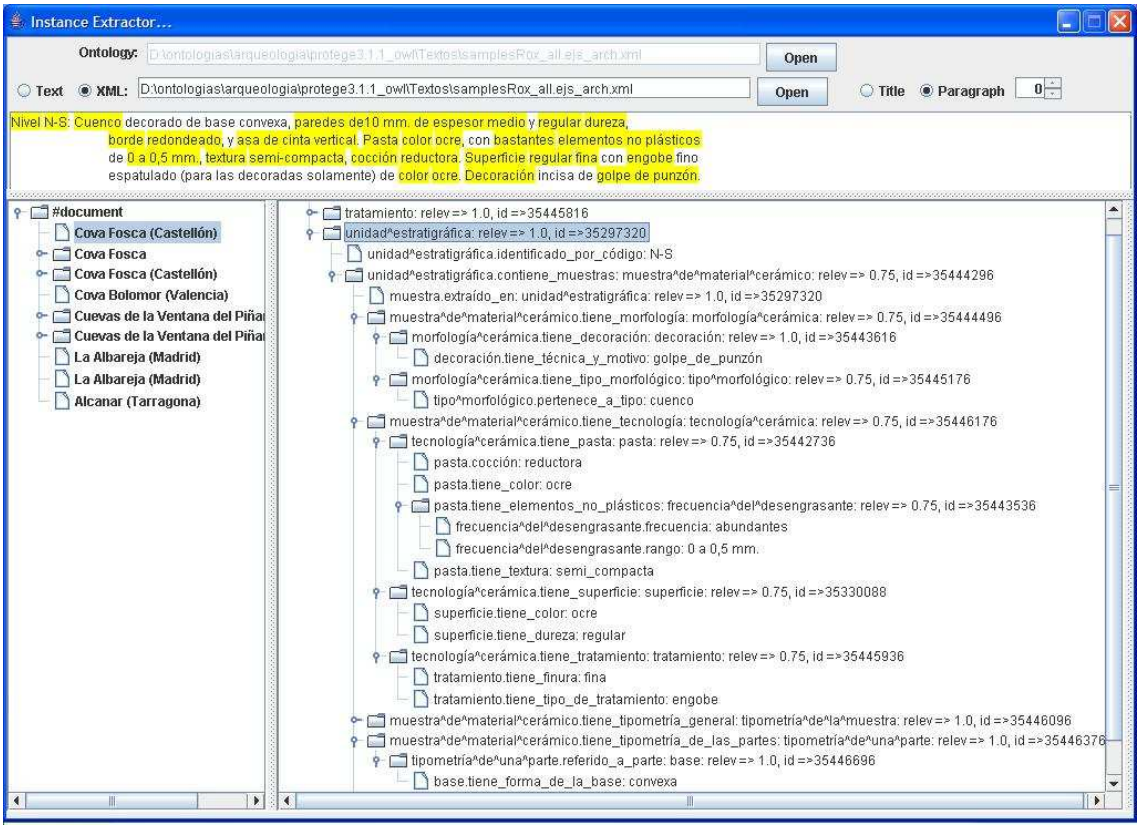


Figura 4.7: Ejemplo de extracción de instancias complejas.

Ejemplo 4.3. Cuando se realizan descripciones de muchos objetos, en lugar de realizar descripciones pormenorizadas, como en el caso anterior, el investigador opta por dar comentarios generales de sus observaciones sobre varios objetos. De ahí que sea frecuente encontrar en los textos párrafos, como el que se describe en el ejemplo de la Figura 4.8, donde se realiza una comparación y generalización de ciertas descripciones ontológicas. En este caso, se trata de las coloraciones de las pastas empleadas para crear objetos cerámicos. Nótese además que a través de la negación se excluyen cierto valores. El sistema desarrollado cuenta con un procesamiento básico para tratar este tipo de casos.

Las generalizaciones son incluidas revisando las palabras que aparecen en plural antes de formar los lemas para el procesamiento final. Con esta información se obvia durante el proceso de agregación las cardinalidades asociadas a las relaciones que puedan asociarse con alguna palabra en plural.

Las negaciones por su parte, son tratadas recuperando expresiones negativas, también antes de la formación de los lemas y de la eliminación de palabras de parada. Si una expresión negativa está "suficientemente" cerca de un ente ontológico, el valor que a éste se le asocia es negado, tal como ocurre en el caso de las coloraciones rojizas de la unidad estratigráfica N-II en el presente ejemplo.

Ejemplo 4.4. Por último, en la Figura 4.9 se expone un ejemplo de dos errores encontrado en algunos de los párrafos analizados. El primero se relaciona con descripciones que aunque se refieren a entes de la ontología no forman instancias como las de las interpretaciones de la ontología, como ocurre con la primera instancia de este ejemplo que fue creada de como un *unifaz ˆcomplejo* en lugar que de tipo "escotadura". El segundo, se relaciona con la especificación de listas de instancias que el sistema es incapaz de reconocer.

Pese a estos errores, el sistema que se propone para la extracción de instancias ontológicas para el desarrollo de la Web Semántica, es de una alta valía pues como se expone en la siguiente sección los resultados experimentales son muy satisfactorios. Las instancias que el usuario desee pueden ser insertadas en la ontología de origen, manteniendo un enlace a las descripciones originales. Así mismo, tal y como se expuso en la sección 4.4.4 los errores u omisiones del sistema pueden ser tenidos en cuenta en posteriores ejecuciones.

4.6 Resultados experimentales y discusión

La experimentación desarrollada de la aplicación, ha estado centrada en la revisión y cálculo de la *precisión* y *recuperación* obtenido por el sistema para un conjunto de párrafos extraídos al azar desde textos de arqueología y agrupados (según el texto del que fueron extraídos en 9 textos de prueba.

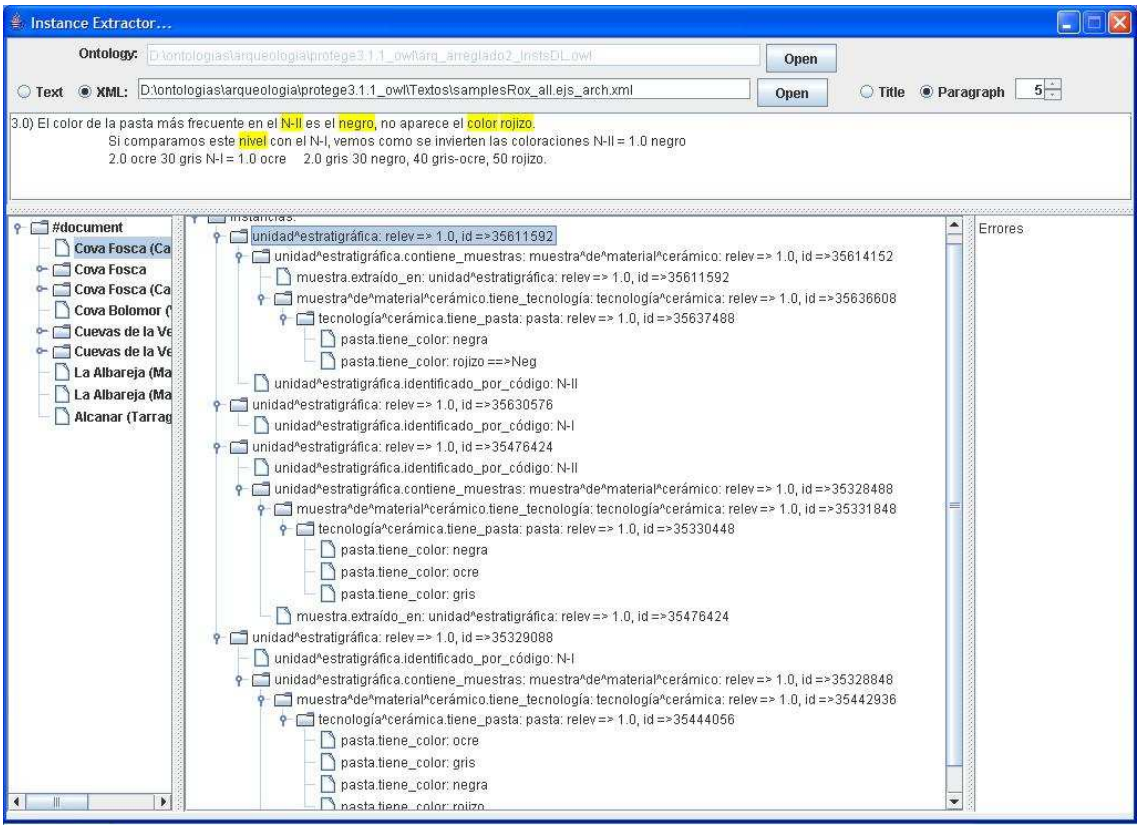


Figura 4.8: Ejemplo de extracción de instancias generalizadas y negadas.

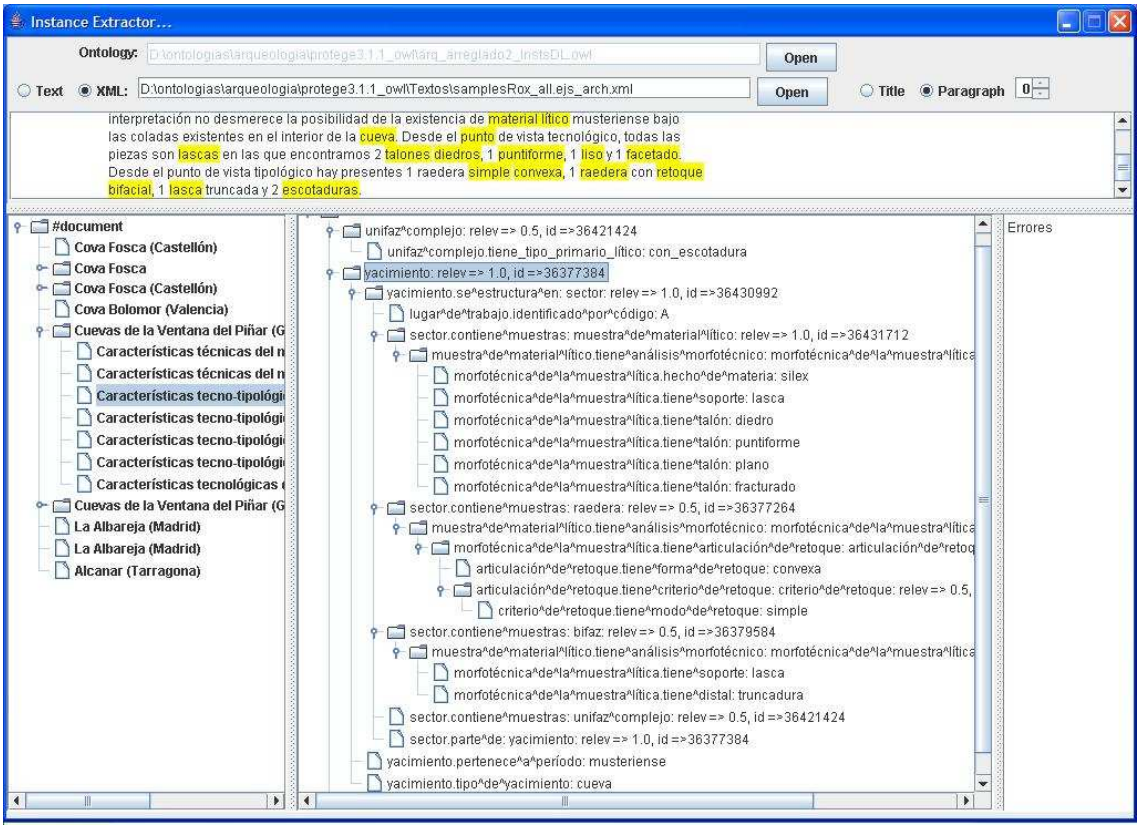


Figura 4.9: Ejemplo de error común.

Ontología		**Lexicón**	
No. Clases	194	Total de entidades nombradas	28
No. Relaciones	131	Funciones de verificación	EEON
Profundidad máxima de las especializaciones	9		
Profundidad máxima de las agregaciones	5		

Tabla 4.2: Descripción de la ontología y el lexicón.

	No. Parrs.	**No. Insts.**	**No. Rels.**	**Insts. Geners.**	**Insts. Negs.**
Texto prueba 1	15	26	222	si	si
Texto prueba 2	5	8	56	si	no
Texto prueba 3	6	9	26	si	no
Texto prueba 4	1	1	2	no	no
Texto prueba 5	55	68	85	si	si
Texto prueba 6	13	28	105	si	si
Texto prueba 7	5	5	3	si	no
Texto prueba 8	2	4	21	si	no
Texto prueba 9	2	4	18	si	no
Total	104	153	538		

Tabla 4.3: Descripción de los textos de prueba analizados. No. Parrs. = Número de párrafos; No. Insts. = Número de instancias; No. Rels. = Número de relaciones; Insts. Geners. = Instancias generalizadas; Insts. Negs. = Instancias negadas

En la Tabla 4.2 se detallan algunos datos referentes a la ontología y el lexicón. Como puede apreciarse, si bien no es una ontología grande, no es comparable a las mayoría de las ontologías tratadas en los sistemas de anotación semántica descritos en la sección 4.2.

A continuación se describe de manera general los resultados obtenidos. Primeramente en la Tabla 4.3 se caracterizan los textos de prueba tratados considerando el total de párrafos contenidos, el número de instancias y relaciones reales en dichos textos y los estilos que en ellos se aprecian (si aparecen entre las instancias algunas generalizadas o negadas).

Por su parte, en la Tabla 4.4 se describen los resultados del número de instancias y relaciones correctas e incorrectas obtenidas por el sistema, así como la precisión y recuperación para cada uno de los textos de prueba y para el sistema en general en cada uno de las tres categorías de interés: instancia, relación e instancia y relación. Los datos asociados a los textos de prueba han sido representados gráficamente en la Figura 4.10.

Todas las precisiones obtenidas para las instancias complejas son mayores o iguales al 90%. Ello significa que casi todas las instancias complejas que han sido recuperadas son correctas. El

	NP	NI	NR	NIC	NII	NRC	NRI	PI	RI	PR	RR	PI+R	RI+R
Tp. 1	15	26	222	25	2	199	6	0,93	0,96	0,97	0,90	0,97	0,90
Tp. 2	5	8	56	6	0	19	0	1	0,75	1	0,34	1	0,39
Tp. 3	6	9	26	8	0	25	0	1	0,89	1	0,96	1	0,94
Tp. 4	1	1	2	1	0	2	0	1	1	1	1	1	1
Tp. 5	55	68	85	54	6	75	19	0,9	0,79	0,79	0,88	0,84	0,84
Tp. 6	13	28	105	22	0	82	4	1	0,78	0,95	0,78	0,96	0,78
Tp. 7	5	5	3	5	0	2	1	1	1	0,67	0,67	0,875	0,875
Tp. 8	2	4	21	3	0	20	3	1	0,75	0,87	0,95	0,88	0,92
Tp. 9	2	4	18	3	0	18	1	1	0,75	0,95	1	0,95	0,95
Total	**104**	**153**	**538**	**127**	**8**	**442**	**34**	**0,94**	**0,83**	**0,93**	**0,82**	**0,93**	**0,82**

Tabla 4.4: Resultados por texto de prueba. Tp. = Texto de prueba; NIC=No. de instancias correctas; NII = No. de instancias incorrectas; NRC= No. de relaciones correctas; NRI= No. de relaciones incorrectas.

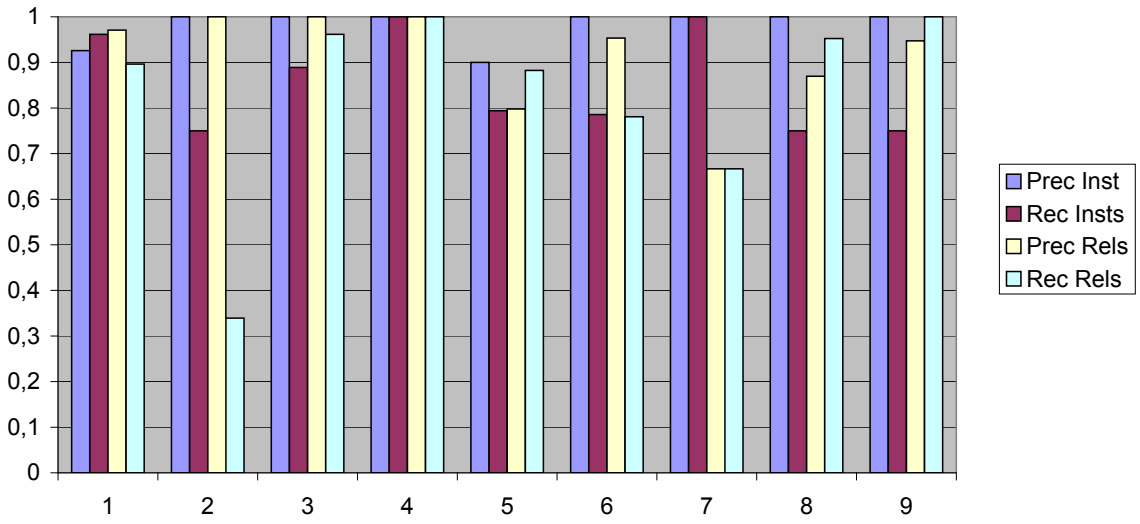


Figura 4.10: Resultados de precisión y recuperación por texto de prueba.

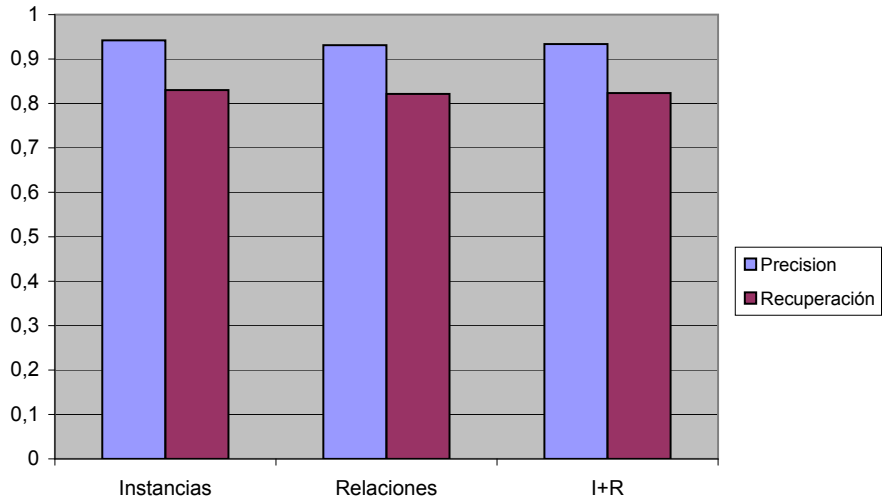


Figura 4.11: Resultados globales de precisión y recuperación.

hecho de que sean instancias complejas no debe pasar desapercibido, ello no sólo significa que casi todas las cadenas léxicas recuperadas se identifican con la clase correcta, sino que además todas las unificaciones necesarias para su formación han sido realizadas entre las instancias iniciales oportunas.

En todos los textos se observa además que se recupera al menos el 75% de todas las instancias complejas. Este valor relativamente bajo de recuperación de las instancias complejas se debe fundamentalmente al uso de listas y explicaciones complejas para describir conjuntos de instancias que se confunden en el sistema con una única instancia.

Nótese que cuando una instancia compleja es recuperada, en lugar de una lista de instancias, una dualidad en los errores son provocados, pues por un lado, se recupera sólo una instancia donde debían recuperarse varias pero las restantes se agregan, por tanto provoca la disminución de la recuperación tanto para instancias como para las relaciones, como se aprecia claramente en el los textos de prueba 2, y en menor medida en los textos 1, y 6.

También influye en la precisión de las relaciones, como ocurre en los textos 5 y 7. Puede verse que excepto estos dos casos la precisión de las relaciones es alta y existe una alta correspondencia entre las precisiones de las instancias y relaciones.

Cabe destacar que se observó un 99% de precisión en la recuperación de los entes ontológicos nombrados. Por otro lado, la entrada al proceso de desambiguación contiene un 96% de todos los entes posibles. De ahí que este proceso podría refinarse, con ciertas heurísticas que añadan peso a ciertos entes en relación también con la morfología de las oraciones tratadas.

Por último, tal como se observa en la Tabla 4.4 la **precisión total del sistema es del 93% y una recuperación del 82%**. Como se puede notar en la Figura 4.11 los resultados son igualmente

buenos por instancias y relaciones. En comparación con los sistemas descritos en la literatura los resultados obtenidos pueden categorizarse como muy satisfactorios. No hay un sistema exactamente comparable al descrito en el presente capítulo dado que el sistema propuesto:

- no necesita de un análisis sintáctico completo, le basta un análisis morfológico,
- no necesita ningún método de aprendizaje,
- permite extraer instancias complejas de cualquier nivel,
- puede ser configurado para emplearse en cualquier ontología de dominio específico,

Sin embargo, a continuación enumeramos algunas interesantes observaciones:

1. En los trabajos Cimiano *et al.* (2004); Dill *et al.* (2003); Valarakos *et al.* (2004) y Yang & Choi (2003) no se realiza ningún análisis sintáctico. En el primer caso la precisión es parecida a la obtenida por la propuesta de este capítulo y en el segundo tanto la precisión como la recuperación son inferiores. El caso de Yang & Choi (2003) obtiene resultados máximos, pero los tipos de instancias extraídas no son comparables.

2. Los trabajos en Ciravegna *et al.* (2002, 2004); Maynard *et al.* (2004); Surdeanu *et al.* (2003); Vargas-Vera *et al.* (2001, 2002) y Celjuska & Vargas-Vera (2004) obtienen resultados superiores, pero requieren de análisis sintáctico completo y algún algoritmo de entrenamiento y excepto en Ciravegna *et al.* (2004) que construye instancias de un nivel, en ninguno se tratan el problema de la formación de instancias complejas.

3. Aquellos sistemas, como Alani *et al.* (2003); Buitelaar & Ramaka (2005); Ciravegna *et al.* (2004) y Amardeilh *et al.* (2005), que permiten permiten extraer instancias complejas (los tres primeros sólo crea instancias complejas de 1 nivel) requieren de análisis sintáctico completo. En algunos casos necesitan también de algún procesamiento semántico y casi todos, algún entrenamiento. El tercero de éstos trabajos obtiene resultados comparables, en tanto el primero obtiene una precisión parecida pero una recuperación casi de un 40% inferior al encontrado por el sistema propuesto. El segundo obtiene resultados visiblemente superiores, pero no son semejantes las complejidades de las instancias tratadas y además realiza análisis sintáctico y semántico completo y emplea algoritmo de aprendizaje. El último de estos trabajos, que trata un dominio complejo, obtiene resultados inferiores tanto en precisión como en recuperación, a pesar del empleo de un análisis sintáctico y semántico completo.

4. En ninguno de los trabajos revisados se ha constatado mención ni solución a los problemas de descripciones generalizadas o negadas.

5. Ninguno de los métodos procesa ontologías en OWL, por tanto no consideran posibles inferencias que pueden ayudar durante el proceso de extracción de información.

4.7 Conclusiones

En este capítulo se ha descrito un método para la extracción de información desde la perspectiva de la Web Semántica, que permite poblar una ontología con las instancias descritas en el texto. La precisión y recuperación del sistema es del 93% y 82% respectivamente, los cuales constituyen resultados muy satisfactorios.

El método propuesto es novedoso en tanto se guía por las especificaciones de dominio contenidas en una ontología para realizar y conformar instancias complejas. Gracias a estas inferencias se evita la necesidad de realizar complejos análisis sintácticos y semánticos en los textos tratados. Además se han incluido heurísticas que permiten reconstruir no solamente instancias simples sino descripciones de instancias generalizadas e instancias con relaciones negadas. Se ha descrito también la metodología para retroalimentar al sistema con los errores u omisiones por él cometidos.

Con el objetivo de recuperar entes ontológicos relevantes para el dominio, se ha propuesto una medida, $R(e, dom)$, que permite evaluar el interés de ciertas cadenas léxicas para un dominio particular, lo cual no solamente es interesante en la aplicación concreta sino en otras como la clasificación y agrupamiento de páginas en la web. Ello podría servir como paso previo de selección de páginas para realizar el proceso de extracción de instancias ontológicas.

Por último, el trabajo ha sido completamente implementado y es factible a integrar como plugin en Protegè. Se ha realizado una experimentación para la ontología de arqueología y se han explicado los principales errores detectados. La solución a estos errores, la creación de un módulo que permita de manera flexible asociar funciones de verificación a entes ontológicos, así como una experimentación extensa que considere múltiples ontologías, idiomas y complejidades de descripciones diversas será el objetivo central de futuros trabajos.

Capítulo 5

Análisis de instancias ontológicas

5.1 Hacia el descubrimiento de patrones en la Web Semántica

Contar con una gran base de datos y toda la formalización de los conceptos allí modelados obliga dar el siguiente paso: el análisis de datos. Las herramientas de OWL permiten crear, a partir de una o varias ontologías, nuevos conceptos asociados a éstas, y determinar las instancias que se corresponden a las especificaciones de los nuevos conceptos. Ésta precisamente es la base fundamental para el análisis de datos en la Web Semántica. Sin embargo, un estudio un poco más profundo de dichos datos requiere alguna herramienta que permita realizar de manera flexible análisis globales y parciales en los datos, de manera qué cada objeto pueda verse con lupas de calibres diferentes y focalizando sólo las partes de interés. Ello es, una herramienta que nos permita navegar por un conjunto de instancias y sus propiedades, pero que además:

- abstraiga de detalles superfluos creando instancias generalizadas para caracterizar conjuntos de éstas,
- muestre y calcule estadísticas que permitan fundamentar los patrones descubiertos.

Formalizar dicha herramienta es el objetivo del presente capítulo. La idea general es: partir de una especificación ontológica, enriquecerla con información de análisis (por ejemplo, especificando las funciones de combinación de datos simples) y con ellas formar dos estructuras bases: las dimensiones conceptuales y los espacios conceptuales multidimensionales. Las dimensiones conceptuales no son más que especificaciones de orden parcial entre objetos que permitirán navegar a través del entramado de relaciones que existe entre ellos. Los espacios conceptuales, por su parte, pueden verse como contenedores “inteligentes” de objetos. Ellos emplean un subconjunto de dimensiones conceptuales, una especificación de niveles de abstracción de interés y un conjunto de funciones de comparación de datos simples para (re)construir instancias generalizadas que

se asocian a dichas especificaciones. Diferentes medidas estadísticas (por ejemplo frecuencias) pueden asociarse a las dimensiones conceptuales y con ellas recuperar patrones que caracterizan a los espacios conceptuales.

La siguiente sección ilustra las características deseables para una herramienta de este estilo y un ejemplo sobre el dominio de arqueología ilustra la importancia de tal instrumento. Analogías con aplicaciones médicas, económicas y sociales pueden ser encontradas con mucha facilidad.

5.2 Características de los sistemas de análisis de instancias ontológicas

Antes de comenzar a exponer el ejemplo concreto, a continuación se describen las características más importantes que deben contener las herramientas de análisis de instancias ontológicas. Ellas son:

- permitir formar nuevas clases de instancias, incluso por definición extensional,
- realizar generalizaciones a partir de un conjunto de funciones de combinación de datos que a los usuarios parezca conveniente,
- utilizar la información de la ontología para construir las dimensiones y los espacios multidimensionales de análisis,
- permitir realizar selecciones a cualquier nivel de abstracción provisto por la ontología,
- recuperar conjuntos de características que garanticen el descubrimiento de patrones interesantes,
- proveer un conjunto de herramientas de minado para el descubrimiento de patrones.

Cada una de estas características son ilustradas a través del siguiente ejemplo.

En Arquelogía una vez terminada la fase de excavación, un arduo trabajo de clasificación, análisis y comparación es llevado a cabo. Su objetivo es caracterizar los materiales encontrados y "recuperar" la memoria de lo que existió en esos sitios hace miles o millones de años atrás.

En la futura Web Semántica un arqueólogo dispondrá de miles de yacimientos descritos en ontologías OWL. Es muy posible que cada grupo de trabajo, cuente con su propia ontología. También grupos de trabajos diversos definen intereses de análisis diferentes. Por ello es indispensable que, en primer lugar, el analista pueda alinear e integrar las diversas ontologías con las que desea trabajar. No sólo las ontologías específicas de arqueología pueden resultar interesantes, otras como las de historia y geografía son de sumo valor. Y en segundo lugar, el sistema debe ser suficientemente flexible como para permitir que el usuario defina, de acuerdo a sus criterios de análisis,

cómo desea realizar las generalizaciones o mezcla los datos. La siguiente sección brinda un esquema general para describir esta información, a la que se ha denominado como *información de análisis*.

Para ilustrar los tipos de análisis de interés para los arqueólogos, en la Tabla 5.1 se presentan 22 de los millones de registros que contienen las bases de datos de tres yacimientos arqueológicos importantes: la Necrópolis Setefilla de Sevilla, la trinchera Dolina de Atapuerca en Madrid y Cova Fosca en Castellón.

La primera tarea que podría resultar muy importante es la de generalización. Es indispensable dar una visión general sobre dónde están ubicados y qué hay en estos yacimientos. Para ello, las características sobre el nombre del yacimiento y los tipos de materiales genéricos encontrados debe ser suplida y la combinación de datos se realiza por medio de la unión de los conjuntos de valores de los datos originales, excepto en el caso de los *pesos de los materiales líticos* para los que es más oportuno emplear como función de combinación el máximo de valores. De esta forma la base de conocimientos (también puede considerarse como almacén de datos en OWL) anterior queda transformada (más bien, puede ser vista) como en la Tabla 5.2. Nótese en ella aparece entre paréntesis las cantidades de materiales asociadas a la instancia generalizada en cuestión (la misma notación es empleada en lo sucesivo).

Como puede verse para cada yacimiento se han resumido cada uno de los tipos de materiales más generales. Nótese que lo que realmente se está haciendo es seleccionar características de interés que pueden ser especificadas con los caminos de agregación desde una clase de referencia (en este caso *yacimiento*) y luego se describen las instancias de la ontología en función de dichas características (formando sub-descripciones de instancias), manteniendo sólo una de las sub-descripciones iguales.

Es fácil notar, por ejemplo, que los yacimientos de Cova Fosca y Setefilla contienen abundantes materiales cerámicos, habiendo más variedad en cuanto a decoración y color en el primero de éstos. Atapuerca, por su parte contiene gran variedad de elementos líticos y un importante legado de un esqueleto de homínido perteneciente al paleolítico de unos 780 000 años.

Así un arqueólogo interesado en la evolución del genero humano en la época más primitiva, posiblemente revise con mayor detalle los datos de la trinchera Dolina, por ejemplo para incluir la datación y las dimensiones de los restos de material líticos allí encontrados. En cambio, un arqueólogo interesado en las cerámicas del sur de España, puede comenzar a revisar y analizar la industria cerámica considerando las características de la forma, decoración y coloración empleada en su creación, como se muestra en las Tablas 5.3 y 5.4.

Materiales cerámicos

m.p.den	m.p.lugar	m.p.per	m.tym	m.cl	m.gr	m.or	m.cocc	m.ce
Cova Fosca	Castellón	neol.	incis.	abierto	hondo	ovoide	oxid.	negro
Cova Fosca	Castellón	neol.	bruñ.	abierto	hondo	conoide	oxid.	ocre
Cova Fosca	Castellón	neol.	bruñ.	cerrado	plano	ovoide	red-ox	gris
Cova Fosca	Castellón	neol.	estamps.	abierto	plano	conoide	red.	gris
Cova Fosca	Castellón	neol.	estamp.	abierto	hondo	conoide	oxid.	gris
Cova Fosca	Castellón	neol.	bruñ.	cerrado	hondo	ovoide	red-ox	gris
Setefilla	Sevilla	br. fin.	incis.	abierto	hondo	esférica	red-ox	rojizo
Setefilla	Sevilla	br. fin.	incis.	abierto	hondo	esférica	red.	rojizo
Setefilla	Sevilla	br. fin.	sin	cerrado	plano	esférica	oxid.	amarill.
Setefilla	Sevilla	br. fin.	sin	cerrado	plano	esférica	red.	amarill.

Materiales líticos

m.p.den	m.p.lug	m.p.per	m.tp	material	peso(gr)	diám(mm)
Atapuerca	Madrid	paleol.	lasca	cr. roca	25	10
Atapuerca	Madrid	paleol.	bipunta	silex	16	14
Atapuerca	Madrid	paleol.	bipunta	cr. roca	12	8
Atapuerca	Madrid	paleol.	pico	silex	9	6
Atapuerca	Madrid	paleol.	lasca	silex	8	8
Cova Fosca	Castellón	neol.	dentic.	silex	10	6
Cova Fosca	Castellón	neol.	raedera	silex	14	6
Cova Fosca	Castellón	neol.	raspador	silex	12	12
Cova Fosca	Castellón	neol.	dentic.	silex	6	6
Cova Fosca	Castellón	neol.	raedera	silex	13	8
Cova Fosca	Castellón	neol.	raedera	silex	7	12

Materiales óseos

m.p.den	m.p.lug	m.p.per	fragm.	especie
Atapuerca	Madrid	paleol.	esqueleto	homínido

Tabla 5.1: Almacén de datos OWL ejemplo. Las columnas representan caminos de agregación desde la clase muestra (materiales cerámicos, lítico u óseos) a los conceptos de interés:
m.p.den = muestra.pertenece_a.denominación;
m.p.lugar = muestra_material_cerámico.pertenece_a.lugar;
m.p.per = muestra.pertenece_a.Período;
m.tym = muestra_material_cerámico.decoración_técnica_y_motivo;
m.cl = muestra_material_cerámico.tiene_descripción_morfológica.clase;
m.gr = muestra_material_cerámico.tiene_descripción_morfológica.grupo;
m.or = muestra_material_cerámico.tiene_descripción_morfológica.orden;
m.cocc = muestra_material_cerámico.cocción;
m.co = muestra_material_cerámico.color;
m.tp = muestra_material_lítico.tipo_primario.

Yacimiento	**Muestras**
Cova Fosca	{mmc: decoración: {incisiones, bruñidos y estampados} morfología: clase: {abierta, cerrada} grupo: {hondo, plano} orden: {ovoide, conoide} pasta: cocción{oxidante, reductor-oxidante, reductor} color: {ocre, rojizo, amarillento} (6) mml: material:{sílex} peso: 23 (6)}
Setefilla	{mmc: decoración: {incisiones, sin decoración} morfología: clase: {abierta, cerrada} grupo: {hondo, plano} orden: {esférica} pasta: cocción:{reductor-oxidante, oxidante, reductor} color: {amarillento}} (4)
Atapuerca	{mml: material:{sílex, cristal roca} peso: 36 (5) mmo: especie: {homínido indiferenciado} fragmento: {esqueleto completo} }(1)

Tabla 5.2: Generalización y selección al primer nivel de abstracción. De la generalización se obtienen los siguientes conceptos específicos:
mmc = muestra_material_cerámico;
mml = muestra_material_lítico;
mmo = muestra_material_óseo.

Decoración	Clase	Orden	Color	Período Arqueológico
incisiones	abierto	ovoide	negro	Neolítico (1)
incisiones	abierto	esférico	rojizo	Bronce final (2)
bruñidos	abierto	conoide	ocre	Neolítico (1)
bruñidos	cerrado	ovoide	gris	Neolítico(2)
estampados	abierto	conoide	gris	Neolítico(2)
sin.	cerrado	esferoide	amarillento	Neolítico(2)

Tabla 5.3: Análisis de muestras cerámicas.

Decoración	Morfología	Color	Período Arqueológico
incisiones	grupo:{abierto} orden:{ovoide, esferoide}	{negro, rojizo}	{Neolítico, Bronce final} (3)
bruñidos	grupo:{abierto, cerrado} orden:{conoide, ovoide}	{ocre, gris}	{Neolítico} (3)
estampados	grupo:{abierto} orden:{conoide}	{gris}	{Neolítico} (2)
sin	grupo:{cerrado} orden:{esferoide}	{amarillento}	{Neolítico}(2)

Tabla 5.4: Análisis y generalización en muestras cerámicas.

Ambas manipulan la misma información, pero en el segundo caso se representa la morfología completa de las instancias en lugar de descomponerla en sus relaciones *clase* y *orden*, como en la primera de las tablas.

Partiendo de la descripción de estas tablas, varias representaciones visuales podrían ser provistas al usuario. En la Figura 5.1 se muestra un primer gráfico que muestra las distribuciones de las diferentes prototipos de muestras cerámicas; en el segundo, se muestra la relación entre la decoración y el color de las muestras.

Revisando los datos de los materiales encontrados durante las excavaciones, varias conclusiones acerca del comportamiento de los datos pueden ser extraídas:

- El 66,67% de los materiales cerámicos de Cova Fosca son de color gris.
- Todos los materiales cerámicos grises son de Cova Fosca, en cambio si tienen forma esférica pertenecen a la Necrópolis de Setefilla.
- El 20% del total de las muestras pertenecen a la Necróplis de Setefilla y tienen coloración rojiza y de las pertenecientes a Setefilla, la mitad son rojizas.

La primera conclusión da una *caracterización* de las muestras de un yacimiento particular. La segunda, permite reconocer características que sólo aparecen en una de las clases, o sea, descubre

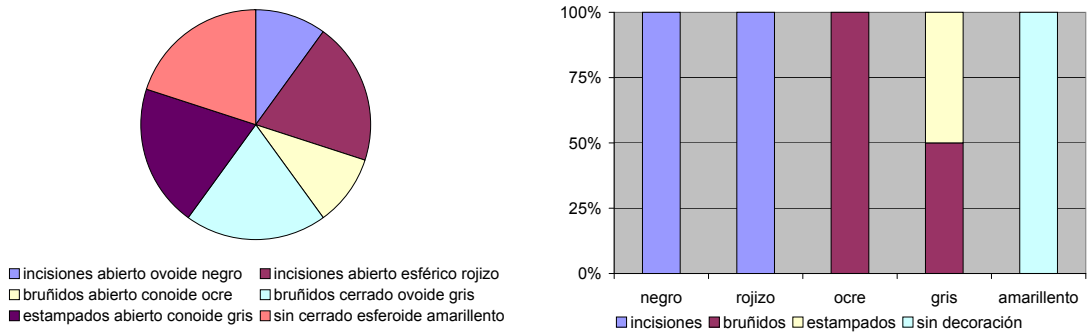


Figura 5.1: Ejemplos de análisis hechos por los arqueólogos.

características *discriminatorias* para dos yacimientos y la tercera, permite reconocer conjuntos de valores que *co-ocurren* entre las descripciones de las muestras, así como explica una dependencia entre estos valores concluyentes. Patrones como los de caracterización, discriminación y co-ocurrencias son recomendados en el área de investigación *minería de datos* para el descubrimiento de nuevos conocimientos. En la sección 5.7 se describen brevemente las técnicas de extracción de patrones más utilizadas y cómo pueden ser éstas adaptadas al nuevo entorno propuesto.

Las siguientes cinco secciones formalizan los procedimientos necesarios para la construcción de espacios conceptuales multidimensionales y el descubrimiento de patrones sobre ellos. En la última sección del capítulo se discuten las limitaciones y aportaciones del formalismo descrito.

Los algoritmos de las secciones 5.5 y 5.6 están inspirados en las ideas de la *inducción orientada a atributos* (en inglés, *attribute-oriented induction*). La inducción orientada a atributos fue propuesta inicialmente en Cai *et al.* (1991) y más tarde extendida en diversos trabajos como Carter & Hamilton (1998); Han & Fu (1996); Han *et al.* (1993, 1998). La idea básica es realizar generalizaciones de los datos almacenados en bases de datos para su exploración y análisis. Para ello, los valores de cada atributo (o dimensión) se generalizan hasta el nivel semántico deseado, y se realizan operaciones de agregación de ciertas medidas de interés para aquellas descripciones que resultan idénticas después de la generalización. La sencillez de estas ideas así como la no imposición de restricciones en los datos han hecho que se considere la inducción orientada a atributos como la primera técnica candidata a explorar para el análisis de instancias ontológicas. La adaptación de las ideas de la inducción orientada a atributos al entorno de la Web Semántica es el objetivo de las formalizaciones de los siguientes epígrafes.

5.3 Información de análisis

Antes de describir el proceso de enriquecimiento de la ontología con información de análisis intentemos responder dos preguntas: ¿qué implica exactamente este proceso? y ¿porqué es importante?

El enriquecimiento de una ontología con información de análisis no es más que la descripción de las especificaciones para el análisis que pueden ser empleadas sobre los objetos. Estas descripciones pueden ser de dos modalidades:

- *descripción de nuevos conceptos*, que servirá para introducir niveles de abstracción adicionales a los especificados en las jerarquías de conceptos expresadas en las ontologías y/o para enlazar conceptos de ontologías diferentes. Nótese que los nuevos conceptos podrían representar especializaciones de conceptos de la ontología original en los que se incluyen caminos hacia otros con los que anteriormente no se relacionaban o incluso representar semánticamente agrupamientos jerárquicos de objetos/valores obtenidos mediante algoritmos de clustering. Con ello se logra aumentar la granularidad de los análisis realizados.

- *descripción de las funciones de combinación de datos simples de los conceptos*, que servirá para especificar cómo pueden ser generalizados conjuntos de valores de un mismo tipo durante el proceso de generalización de instancias. Ello significa que el analista de datos puede definir si es semánticamente compatible una función de combinación particular con un tipo de datos. Por ejemplo, la función de combinación que calcula el promedio de un conjunto de valores se puede asociar semánticamente a una secuencia temporal de temperaturas de una ciudad, pero no a un conjunto de temperaturas de diferentes ciudades.

Aunque es totalmente plausible definir este tipo de descripciones cada vez que se cree un nuevo espacio conceptual multidimensional, es preferible mantener esta información semántica siempre disponible y adaptable a cada caso. Con tal finalidad, el analista debe construir una meta-ontología que contenga la información anterior, también empleando la lógica descriptiva. Por supuesto, los nuevos conceptos serán formulados empleando la notación usual. Esta solución no sólo facilita el trabajo del analista, sino además permite la ejecución de estudios posteriores que emplean conclusiones extraídas de análisis con dichas especificaciones. Estos estudios pueden ir desde análisis sobre el mismo dominio de conocimiento hasta análisis sobre los procedimientos empleados por un grupo de analistas.

La descripción de las funciones de combinación de los conceptos podrían especificarse por medio de instancias asociadas al concepto *ConceptoCombinable* de la ontología descrita por medio de:

$$ConceptoCombinable \equiv \exists tieneConcepto.URI \sqcap \forall tieneRelacion.URI \sqcap \exists tieneFuncCombinacion.FuncCombinacion$$
$$FuncCombinacion \sqsubseteq \exists tieneNombre.String \sqcap \exists tieneImplementacion.URI$$

Como puede verse, los conceptos combinables son aquellos para los cuales ha sido definida una función de combinación. Nótese que lo mismo puede ser un tipo de datos, que un concepto simple, definido en la ontología o en su extensión, a través del concepto que lo agrega y la relación que establece dicha agregación.

5.4 Dimensiones y sus operaciones

Una dimensión, tal como explicamos en la introducción del capítulo, no es más que una especificación de un conjunto de conceptos y el orden en el que podemos navegar entre ellos. Esta navegación se garantiza gracias a los operadores de abstracción y generalización entre instancias ontológicas ya definidos en el Capítulo 3 y por el operador de selección que se introduce en esta sección.

Como punto de partida, se define el concepto de orden parcial dimensional, que relaciona los conceptos utilizando jerarquías conceptuales descritas en la ontología y las agregaciones entre conceptos que se describen por medio de las relaciones. Por supuesto, aquí se refiere a la ontología extendida que fue explicada en la sección 5.3.

Definición 5.1. Sea $\mathcal{O}$ una ontología. Llamaremos orden parcial dimensional, denotado por $\overset{\sqsubset}{\rightarrow}$, al orden parcial entre conceptos definido $\forall C, C' \in N_C$ empleando las siguientes restricciones:

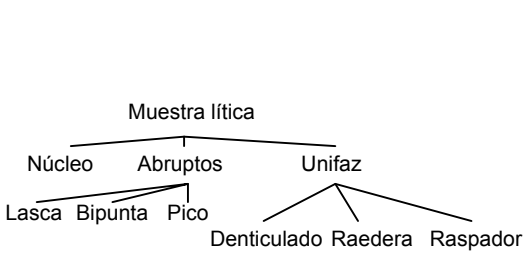
- $C \overset{\sqsubset}{\rightarrow} C'$ si $C' \sqsubset C$, ó
- $C \overset{\sqsubset}{\rightarrow} C'$ si existe un camino de C a C', o sea, si $CamCC_i(C, C') \neq \emptyset$

Usaremos $\overset{\sqsubset}{\rightarrow} *$ como la clausura reflexiva y transitiva para la relación $\overset{\sqsubset}{\rightarrow}$.

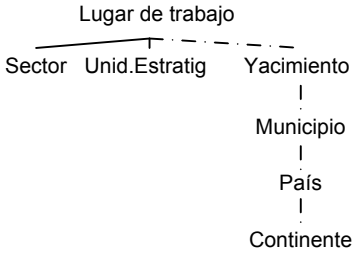
Definición 5.2. Sea $\mathcal{O}$ una ontología. Llamaremos dimensión conceptual al par $D = (C_d, \overset{\sqsubset}{\rightarrow})$ siendo C_d un subconjunto de conceptos satisfacibles en $\mathcal{O}$, $\top \in C_d$ y $\overset{\sqsubset}{\rightarrow}$ la relación de orden parcial dimensional para los elementos en C_d.

Nótese que aunque en la definición 5.1 no aparecen restricciones para la selección de los caminos de agregación a emplear para formar un orden parcial dimensional, al formar una dimensión se especifica el conjunto de conceptos que se desean mantener en la dimensión. De ahí que caminos no interesantes a tratar en las dimensiones pueden ser podados.

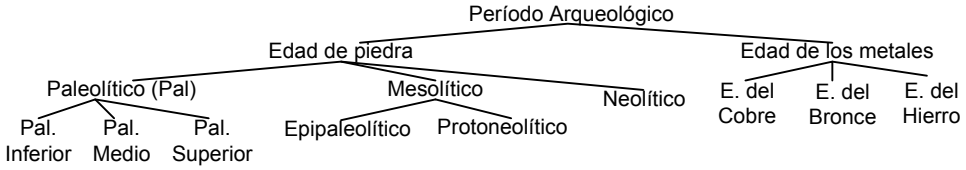
Ejemplo 5.1. En la Figura 5.2 se muestran las dimensiones asociadas a los conceptos *Muestra lítica*, *Lugar de trabajo* y *Período Arqueológico*. Aparecen en líneas discontinuas las relaciones en el orden parcial dimensional formadas por las agregaciones entre conceptos.



(a) Dimensión muestra lítica



(b) Dimensión Lugar de trabajo



(c) Dimensión Período Arqueológico

Figura 5.2: Ejemplos de dimensiones para un esquema conceptual multidimensional.

Definición 5.3. Sea $\mathcal{O}$ una ontología concreta con Abox $\mathcal{A}$. Sea ℓ un dato que representa uno cualquiera en $\mathcal{U}$. Diremos que una instancia a de tipo C es seleccionadora si:

$$\forall b, \exists \{R_1,...,R_n\} \subseteq N_R, R_1(a,a_1),...,R_{n-1}(a_{n-1},a_n),R_n(a_n,b) \in \mathcal{A}, b = \ell \vee b \in_* \bigcup_{\substack{\forall C' \text{ tal que} \\ CamsCC_i(C,C') \neq \emptyset}} C'$$

Definición 5.4. Sea $\mathcal{O}$ una ontología concreta con Abox $\mathcal{A}$, $a \in \mathcal{U}$. Sea ℓ un dato que representa uno cualquiera en $\mathcal{U}$. Diremos que $a' \in \mathcal{U}$ es seleccionada considerando a a si:

- $a' \in \Phi$, $a' \in \{a, \ell\}$
- $a \in_* C, C \in N_C$ y a'' calculada por $d(a' \uparrow_C a'')$ cumple que:

$$\forall b \in \Phi \cup \{\ell\} \text{ tal que } \exists \{R_1,...,R_n\}, R_1(a,a_1),...,R_{n-1}(a_{n-1},a_n),R_n(a_n,b) \in \mathcal{A},$$
$$\exists R_1(a'',a''_1),...,R_{n-1}(a''_{n-1},a''_n),R_n(a''_n,b'') \in \mathcal{A}, b'' \in \Phi \wedge (b'' = b \vee b = \ell)$$

Ejemplo 5.2. La instancia a descrita por $d(a) = \{identificado\_por\_codigo(a, \text{“N-S”})\}$ permite seleccionar en el contexto arqueológico tratado a todas las unidades estratigráficas con código “N-S”. Si a se describiese por $d(a) = \{identificado\_por\_codigo(a, \ell)\}$ se seleccionarían todas las unidades estratigráficas sin excepción.

5.5 Espacio conceptual multidimensional

A continuación se describen los conceptos de esquema conceptual multidimensional y de espacio conceptual multidimensional. El primero de ellos puede verse como la estructura que permite describir qué y cómo se analiza (o sea, especifica las dimensiones a emplear para describir la

información de las instancias ontológicas). El segundo es el contenedor donde se describen, en función de las especificaciones del esquema conceptual multidimensional, las instancias que se analizan.

Definición 5.5. Sea $\mathcal{O}$ una ontología, $C \in N_C$ un concepto y $CC = cam_1,...,cam_n$ un conjunto de caminos de C, con conceptos de llegada $C_1^*,...,C_n^*$ (o sea, $cam_i = \oplus_{1 \leq j \leq n_{i-1}} \left\langle RS_{i_j}, C_{i_j}^* \right\rangle \oplus \left\langle R_{i_{n_i}}, C_n^* \right\rangle$). Llamaremos *esquema conceptual n-dimensional* (o simplemente *multidimensional*) de $\mathcal{O}$ asociado a C usando los caminos CC a la tupla $E = (D_1,...,D_n, c_{\phi_1},...,c_{\phi_n})$ en la que se cumple que $\forall D_i = (Cd_i, \overset{\sqsubset}{\rightarrow})$, $\forall C_{i_j} \in Cd_i$, $C_i^* \overset{\sqsubset}{\rightarrow} *C_{i_j}$ y además c_{ϕ_i} es una función de combinación de datos simples que asigna una función de combinación a cada tipo de datos simple al que puede llegarse desde algún concepto de Cd_i (ver definición 3.18). Si el conjunto de caminos CC es disjunto, se dice que E es un esquema multidimensional de conceptos disjuntos.

Definición 5.6. Sea $\mathcal{O}$ una ontología concreta con Abox $\mathcal{A}$. Sea E un esquema conceptual m-dimensional de $\mathcal{O}$ asociado a C usando los caminos en CC. Llamaremos espacio conceptual m-dimensional de $\mathcal{O}$ respecto a E al conjunto de tuplas $\{t_1,...,t_n\}$, $t_i = (d_{i_1},...,d_{i_m})$, $t_i \neq t_j, \forall i,j \in \{1,...,n\}$, donde cada dato cumple que:

1. $d_{i_k} \in_* C_{i_s}, C_{i_s} \in Cd_i$, ó
2. $\exists d \in_* C_{i_s}$ tal que d_{i_k} es una instancia seleccionada respecto a una instancia de selección de clase C_{i_s}, ó
3. d_{i_k} es un dato generalizado de un conjunto de datos seleccionados respecto a una instancia de selección de clase $C_{i_s}, C_{i_s} \in Cd_i$.

Además, cada t_i representa una instancia generalizada de un conjunto de instancias de $\mathcal{A}$ seleccionadas respecto a una instancia de selección de clase C que contiene todos los caminos en CC.

Ejemplo 5.3. El esquema multidimensional expresado en la Tabla 5.4, ha sido construido dada la siguiente definición de esquema de espacio conceptual multidimensional asociado al concepto *MuestraCerámica*, $E = (D_1, D_2, D_3, D_4, c_{\phi_{union}}, c_{\phi_{union}}, c_{\phi_{union}}, c_{\phi_{union}})$ siendo:

$$
\begin{aligned}
&D_1 = (\{Decoración\}, \emptyset),\\
&D_2 = (\{Morfología\}, \emptyset),\\
&D_3 = (\{Color\}, \emptyset),\\
&D_4 = (\{Período\ Arqueológico, Edad\ de\ piedra, Edad\ de\ los\ metales, Paleolítico,\\
&\qquad Neolítico, Bronce\ final\}, \{(Período\ Arqueológico, Edad\ de\ piedra),\\
&\qquad (Período\ Arqueológico, Edad\ de\ los\ metales),\\
&\qquad (Edad\ de\ piedra, Paleolítico), (Edad\ de\ piedra, Neolítico),\\
&\qquad (Edad\ de\ los\ metales, Bronce\ Final)\}),\\
&c_{\phi_{union}} = \{\langle \phi^D, \hat{\cup} \rangle \mid \phi^D \in \Phi\}
\end{aligned}
$$

y $\hat{\cup}$ definida por: $\hat{\cup}(d_1,...,d_n) = \{d_1\} \cup ... \cup \{d_n\}$. Nótese que la cuarta dimensión es una selección de la representada en la Figura 5.2(c) y la combinación establecida por la función de combinación de datos simples es la unión de conjuntos.

La construcción de un espacio conceptual m-dimensional a partir de una ontología concreta $\mathcal{O}$ y de un esquema conceptual E de $\mathcal{O}$ asociado a la clase C se describe en el Algoritmo 5.1. La idea básica es, en un primer paso, construir una conexión (en ingles, *mapping*) entre cada dato asociado a cada dimensión y su valor generalizado, de acuerdo al esquema conceptual multidimensional para las instancias de tipo C. En un segundo paso, se forman las instancias generalizadas sustituyendo cada m-tupla por una m-tupla generalizada que constituya la generalización de las instancias de igual tipo por cada dimensión.

Algoritmo 5.1 Construcción de espacios conceptuales multidimensionales

Entradas: $\mathcal{O}$, E, C, β_m, β_M

{ $\mathcal{O}$, ontología concreta con Abox $\mathcal{A}$;
C, clase de referencia para crear los espacios multidimensionales;
E, esquema multidimensional;
β_m, β_M, porcentajes mínimo y máximo de valores diferentes en cada dimensión. }

Salidas: E'

{E', espacio multidimensional asociado al esquema E y la ontología $\mathcal{O}$ }

Primera parte: Eliminación de dimensiones irrelevantes y creación de mappings de generalización entre datos.

Sea $\mathcal{A}_C = \{a | a \in_* C\}$.

Recorrer $\mathcal{A}_C$ y agrupar los datos diferentes asociados a cada dimensión de E.

if $\beta_m \geq |\frac{D_i}{\mathcal{A}_C}| \geq \beta_M$ (la dimensión D_i debe ser removida) **then**

$E = (D_1,...,D_{i-1},D_{i+1},...,D_n,c_{\phi_{c_1}},...,c_{\phi_{c_{i-1}}},c_{\phi_{c_{i+1}}},...,c_{\phi_{c_n}})$

else

Crear los mappings (d,d') para cada valor d coleccionado en D_i, $d(d \uparrow_{C_i^{sup}} d')$ con C_i^{sup} una de las clases ancestras directas de C_i, $d \in C_i^I$.

Segunda Parte: Formación de los espacios conceptuales multidimensionales.

$E' = \{\}$

for all $a \in I_C$ **do**

Sea $t = (d_1,...,d_n)$ la tupla de datos asociados a a, considerando E.

$t' = (d_1^{'},...,d_n^{'})$, calculado a partir del mapping creado en el paso anterior, siendo $C_i^{'}$ el tipo del dato d_i.

if $\exists t'' = (d_1^{''},...,d_n^{''}) \in E'$, siendo $C_i^{''}$ el tipo del dato $d_i^{''}$, $\forall i \in \{1,..,n\}$ **then**

$t'' = (d_1^{''} \cup d_1^{'},...,d_n^{''} \cup d_n^{'})$

else

$E' = E' \cup \{t'\}$

5.6 Espacios conceptuales interesantes

El espacio multidimensional explicado en la sección anterior permite a sus usuarios moverse libremente por el conglomerado de objetos y revisar los aspectos que les parecen más interesantes. Pero, ¿qué sucede si el analista de datos no tiene idea acerca de las características de la distribución de los objetos en el dominio o si el número de características es sumamente elevado? En cualquier caso, toda capacidad de análisis quedaría prácticamente anulada. Sin embargo, la herramienta de análisis podría sugerir al usuario ciertas dimensiones de análisis interesantes, basándose en un proceso de *selección de características*.

La selección de características[1] tiene sus orígenes en la tarea de reducción de dimensionalidad descrita en la Estadística Matemática y ha sido ampliamente estudiada en varios campos relacionados con el Aprendizaje Automático.

Ninguna otra clasificación de los métodos de extracción de características ha sido tan aceptada como la definida por Pat Langley en 1994. En ella, los métodos se dividen según la estrategia usada para evaluar subconjuntos de características. Un primer tipo, de filtrado, emplean medidas para seleccionar/excluir ciertos conjuntos de atributos, sin tener en cuenta ningún algoritmo de inducción. Los métodos del segundo tipo, de wrapper, evalúan la relevancia de cada conjunto de características candidatas según la aproximación de la descripción resultante empleando algún algoritmo inductivo, Langley (1994).

En cualquier caso, es imprescindible contar con una medida de evaluación de la importancia del conjunto de atributos en cuestión. Las medidas más empleadas en la literatura incluyen la ganancia informacional (Kullback & Leibler (1951)), el índice Gini (Breiman *et al.* (1984)), la incertidumbre (Johnson & Wichern (1988)) y los coeficientes de correlación. Los métodos tradicionalmente empleados forman agrupamientos de atributos o árboles de decisión, Buntine & Niblett (1992). Múltiples trabajos han sido desarrollados en relación a este tema, los más recientes pueden encontrarse en FSDM (2005), FSDM (2006) y Guyon *et al.* (2004). Algunos otros trabajos, como Molina *et al.* (2002) y Barber & Hamilton (1997), muestran comparaciones de diferentes métodos de selección de características. A pesar de ello, no existe un consenso sobre los métodos más adecuados, sobre todo porque el método a emplear y la medida de evaluación para la selección de las características dependen de la naturaleza de los atributos en cuestión.

Sin embargo, la gran cantidad de trabajos que argumentan a favor de árboles de decisión y ganancia informacional (como ID3 y C4.5) hace que, como una primera aproximación, seleccionemos dicha combinación como método de selección de características. Más específicamente, en

[1] En inglés, feature selection.

este trabajo se propone el cálculo de esquemas conceptuales multidimensionales interesantes asociados a un concepto C por medio del Algoritmo 5.2, una variante del explicado en Han & Kamber (2001). La idea general de este algoritmo consiste en emplear las distribuciones de un conjunto de objetos en relación a un conjunto de clases para tomar en cada momento las composiciones de relaciones (caminos) que mayor ganancia informacional obtienen respecto a la distribución. El bloque principal es el procedimiento *calcSeudoEsquemas* que selecciona, en un primer paso, los caminos con alta ganancia informacional. Luego, por cada camino, se subdivide la distribución inicial según los posibles valores que pueden tomar los datos asociados a los objetos a través del camino analizado, y vuelve a realizarse el proceso de subdivisión mientras se obtengan ganancias informacionales mayores que el umbral dado (γ). Al mismo tiempo que se realiza este proceso, se calculan pesos que indican el interés del esquema conceptual que pudiera formarse con cada conjunto de caminos.

La ganancia informacional de un atributo dice en cuánto dicho atributo permite discriminar las clases en las que se divide un conjunto de datos. Sea $S = \{S_1, ..., S_n\}$ las clases de objetos que se desean analizar. La ganancia informacional de utilizar un atributo A se define por:

$G(A) = I(\{S_1, ..., S_n\}) - E(A)$,

o sea, la diferencia de la información esperada empleando S y la entropía de la dimensión o atributo A, siendo:

$I(S) = -\sum_{i=1}^{m} \frac{|S_i|}{N} log_2 \frac{|S_i|}{N}$, con $N = \sum_{i=1}^{n} |S_i|$

y la entropía del atributo A considera los valores $\{a_1, ..., a_m\}$ que puede tomar dicho atributo y que produce una partición de S' en los conjuntos S_{i_j} que contienen los objetos en S_i con valor a_j en el atributo A. Su valor se calcula por medio de:

$E(A) = \sum_{j=1}^{m} \frac{|S_{1_j}| + ... + |S_{n_j}|}{N} I(\{S_{1_j}, ..., S_{n_j}\})$

A continuación se ilustra brevemente ejemplo del cálculo de la ganancia informacional, considerando los datos de los materiales líticos de la Tabla 5.1. En este caso, consideraremos la dimensión de Período Arqueológico como la representada en 5.2(c), para la dimensión *peso* la función de combinación es definida de tal modo que los datos se subdividan en los intervalos $[0..10)$, $[10..15)$ y $[15, \infty)$ y la función de combinación para el *diámetro* subdivide sus datos en $[0..10)$ y $[10..\infty)$. La clase de referencia del esquema conceptual multidimensional que se desea construir es la de material cerámico. La cuarta columna de esta tabla describe la clase específica de cada material, cuya dimensión viene dada en la Figura 5.2(a).

Teniendo en cuenta las clases derivadas de *Muestra lítica*, la Tabla 5.1 se dividen en dos partes: una que se corresponde a las muestras de tipo abrupto que aparece en las 5 primeras filas de la tabla, y las correspondientes a las muestras unifaces en las últimas 6 filas. Identificaremos por $o_1, ..., o_{11}$ a estos objetos según el orden de la tabla.

Sea Cp un concepto padre en una dimensión y sean los conceptos $\{C_1,...C_n\}$ los conceptos directamente especializados de Cp y sea $objs$ una función que asocia cada concepto con el conjunto de objetos de dicho tipo. Sea β el porcentaje de máxima correlación entre número de objetos en un concepto hijo y en el de su padre.
Si $\exists C_i$ tal que $|objs(C_i)| \geq \beta |objs(C_p)|$
Promover a los conceptos $C_1,...,C_n$ al nivel del concepto C_p
Borrar a C_p de la dimensión

Figura 5.3: Reglas de filtrado de conceptos poco interesantes.

De esta forma, se tiene que la partición inicial de las muestras líticas es:
$S = \{\{o_1,...,o_5\},\{o_6,...,o_{11}\}\}$.
$I(S) = -\frac{5}{11} log_2 \frac{5}{11} - \frac{6}{11} log_2 \frac{6}{11} = 0,99$
Considerando la dimensión *peso*, se tiene que la nueva división, S', queda dada por:
$S_{1_{[0..10)}} = \{o_4,o_5\}$, $S_{2_{[0..10)}} = \{o_9,o_{11}\}$,
$S_{1_{[10..15)}} = \{o_3\}$, $S_{2_{[10..15)}} = \{o_6,o_7,o_8,o_{10}\}$,
$S_{1_{[15,\infty)}} = \{o_1,o_2\}$, $S_{2_{[15,\infty)}} = \emptyset$,
De manera que:
$E(peso) = \frac{4}{11} I(\{S_{1_{[0..10)}}, S_{2_{[0..10)}}\}) + \frac{5}{11} I(\{S_{1_{[10..15)}}, S_{2_{[10..15)}}\}) + \frac{2}{11} I(\{S_{1_{[15,\infty)}}\}) = 0,69$
Resultando una ganancia: $G(peso) = 0,99 - 0,69 = 0,30$

De igual manera puede comprobarse que si se subdivide S' empleando ahora la dimensión *diámetro* se obtiene una ganancia muy baja de $0,004$, en tanto adicionando la dimensión *material* la ganancia informacional aumenta a $0,35$. Ello sucede porque mientras diámetro no permite discriminar mucho las clases más que lo que hace el *peso*, el *material* sí que ayuda a mejorar dicha discriminación.

Un paso de filtrado adicional puede hacerse para cada dimensión teniendo en cuenta la relación entre la cantidad de objetos asociados a un concepto y a su concepto padre. De manera que un concepto padre poco significativo podría eliminarse de la dimensión. Para ello se aplica la regla descrita en la Figura 5.3.

5.7 Usando un espacio conceptual multidimensional

Por supuesto, tal y como explicamos con anterioridad en este capítulo, el uso más trivial de un espacio multidimensional es permitirle a los usuarios ver sus datos desde diferentes perspectivas. Los modelos tabulares, en 3-D, las gráficas de curvas, barras o de sectores y los grafos que enlazan conceptos son las primeras herramientas de análisis que pudieran ofrecerse. Ello, conjuntamente con la posibilidad de realizar generalizaciones y selecciones a cualquier nivel. La gran utilidad de

Algoritmo 5.2 Selección de esquemas conceptuales interesantes

Entradas: $\mathcal{O}$, C, γ, α

{ $\mathcal{O}$, ontología concreta con Abox $\mathcal{A}$,
C, clase de referencia para crear los esquemas de espacios multidimensionales,
γ, mínima ganancia informacional permitida
α, mínima cantidad de objetos para una descripción }

Salidas: *SP*

{*SP*, conjuntos de caminos asociados a C con cuyos subconjuntos pueden formarse esquemas multidimensionales interesantes}

Sea $Cams_C$ el diccionario de los caminos que parten de C con llave en el concepto de llegada.
Sea $S = \{S_i | S_i = \{a \in_* C_i\}, \forall C_i, i \in \{1,...,n\}, C \sqsubseteq C_i, C_i \neq C_j, j \in \{1,...,n\}, i \neq j\}$
$SP = calcSeudoEsquemas(Cams_C, S, \gamma)$

function $calcSeudoEsquemas(Cams_C, S, \gamma)$:
{devuelve una lista de pares que contienen en el primer elemento un camino y en el segundo un valor real que indica la importancia del camino para la formación de los esquemas interesantes}
Primer Paso: Calcular la importancia de la actual subdivisión de los objetos.
$ve = \sum_{S_i \in S} imp(S_i)$, donde

$$imp(S_i) = \begin{cases} 1, & \text{si } |S_i| > \alpha \\ 1 - |S_i|/\alpha, & \text{en caso contrario} \end{cases}$$

Segundo Paso: Revisar y devolver subconjuntos de caminos que mantengan una alta ganancia informacional en la subdivisión de los objetos por ellos inducidos.

if $|S| = 1$ **then**
  Devolver $(\emptyset, ve)$
else
  $SP = \emptyset$
  Calcular ganancia informacional, G, para los conceptos de llegada en $CamsC_I(C)$ considerando la clasificación en S
  Sea $\{cam_1, ..., cam_m\}$ el conjunto de caminos que permiten una alta discriminación entre los objetos ordenados según el valor de la ganancia: $G(cam_1) \geq ... \geq G(cam_n) > \gamma$; sea C_k el concepto de llegada asociado al camino cam_k, $k \in \{1,...,m\}$.

  for all $k \in \{1,...,m\}$ **do**
    Sea $CD \subseteq \{C' | C' \sqsubseteq C_k\}$
    for all $C' \in CD$ **do**
      Sea $S = \{S_{Cl_{C'}} = \{a \in_* C_i | C_i \in \{1,...,n\}, C \sqsubseteq C_i$, a está relacionado a algún dato d según $cam_i \wedge d \in_* C'\}\}$
    $SP = SP \cup_{(sp,v) \in calcSeudoEsquemas(Cams_C - \{cam_i\}, S, \gamma)} \{\langle C_k \cup sp, v + ve \rangle\}$
  Devolver *SP*

estos métodos de análisis de patrones es que permiten caracterizar clases de objetos en relación a otras, permitiendo la comparación y/o el descubrimiento de particularidades entre clases.

Aunque estos son los métodos que tradicionalmente se han usado, un analista puede estar interesado en obtener algunos patrones más complejos acerca del comportamiento de sus datos. Múltiples instrumentos de extracción de patrones han sido descritos en la literatura, sobre todo a partir del surgimiento de la Minería de Datos. Es por tanto perfectamente plausible crear algoritmos que permitan la extracción de estos tipos de patrones en el entorno conceptual multidimensional, tal como se ha demostrado en Han & Kamber (2001); Han *et al.* (1993) y Danger *et al.* (2004c). A continuación se describen tres de los patrones más interesantes a extraer de los datos. Estos se corresponden con las reglas descritas en la sección 5.2.

- *Patrones de caracterización*: representan reglas para caracterizar una clase a partir de los valores de un subconjunto de sus dimensiones. Pueden ser éstas expresadas por: $clase\ X \Rightarrow Condición[p_c]$, siendo $p_c = \frac{100 \times count(Condición)}{n}$, la que significa que en la clase X se evidencia la presencia de *Condición* un porcentaje p_c de los casos, siendo *Count* una función que cuenta los casos en los que aparecen ciertas condiciones y n el total de objetos tratados. p_c es conocido como coeficiente de caracterización.

- *Patrones de discriminación*: representan reglas que permiten reconocer el porcentaje de objetos de una clase dada que cumplen una condición dada respecto al total de objetos. Pueden ser éstas expresadas por: $clase\ X \Leftarrow Condición[p_d]$, siendo $p_d = \frac{100 \times count(Condición \wedge claseX)}{count(Condición)}$. p_d es conocido como coeficiente de discriminación.

- *Reglas de asociación a diferentes niveles*: representan reglas que caracterizan una clase de objetos en la que se relacionan co-ocurrencias de valores en los atributos del espacio multidimensional. Pueden ser expresadas por: $Condición_1 \Rightarrow Condición_2[s,c]$, siendo $s = \frac{100 \times count(Condición_1 \wedge Condición_2)}{n}$, $c = \frac{100 \times count(Condición_1)}{count(Condición_1 \wedge Condición_2)}$, la que significa que en un s% de los objetos se observa tanto la condición $Condición_1$ como $Condición_2$ y en un c% de los que cumplen la primera condición cumple también con la segunda. s es conocido como soporte de la regla y c como su confianza.

Una explicación detallada de cómo pueden ser extraídas cada una estas reglas en un entorno de bases de datos relacionales usando la inducción orientada a atributos puede verse en Han *et al.* (1998). Unas pocas transformaciones son necesarias para emplear estos resultados en el modelo desarrollado en el presente capítulo.

5.8 Conclusiones

En este capítulo ha sido formalizado un modelo para el análisis de instancias ontológicas que no impone restricciones en los datos. Está basado en las formalizaciones dadas en el Capítulo 3 y dan continuidad al proceso de desarrollo para la Web Semántica. La formalización dada permite realizar generalizaciones y selecciones a cualquier nivel. Se ha descrito un algoritmo para construir los espacios conceptuales multidimensionales dado un esquema conceptual deseado y ha sido provisto otro que permite calcular esquemas apropiados para realizar caracterizaciones deseadas. Por último, se han mostrado patrones loables a extraer en los espacios conceptuales y que pueden enriquecer los análisis que tradicionalmente se usan en muchas áreas de conocimiento.

A pesar de las ventajas del modelo propuesto, debe considerarse que la escalabilidad del sistema puede verse afectada en grandes bases de conocimientos, dado que cualquier generalización requiere de cálculos más o menos complejos, (por el uso de las funciones de combinación dadas en el Capítulo 3). La manera de vencer estos obstáculos requiere de un estudio profundo, pero en una primera instancia puede recurrirse a las teorías descritas en OLAP, aunque con esta solución la ventaja de no imposición de restricciones sobre los datos, podría no ser mantenida.

Capítulo 6

Espacios conceptuales para colecciones de documentos

Hasta el momento hemos tratado los temas de extracción de información y su análisis en entornos de dominios de aplicación cerrados, donde se conoce una ontología que describe los objetos que pudieran encontrarse. Pero, ¿qué sucede si necesitáramos analizar la información contenida en páginas web o en una colección de documentos tratando temas de dominio general? En estos casos, es inviable presuponer la existencia de un formalismo como el de la lógica descriptiva que contenga todos los conceptos y relaciones (entre conceptos) de manera que las descripciones en los textos puedan ser analizadas de manera similar a como se ha propuesto en el capítulo anterior.

El método más utilizado actualmente para trabajar con grandes colecciones de documentos consiste en aplicar algún Sistema de Recuperación de la Información (SIR), Salton (1989). En estos sistemas, los usuarios pueden definir consultas combinando una serie de operadores (por ejemplo booleanos) sobre un conjunto de palabras clave o términos. El sistema devuelve como respuesta una lista de documentos ordenados según su *relevancia* con respecto a la consulta formulada. Esta forma de trabajar tiene varios inconvenientes a la hora de abordar una tarea de análisis. En primer lugar, los usuarios generalmente desconocen los contenidos de los documentos requeridos, lo que dificulta la formulación adecuada de consultas. En segundo lugar, los SIR no proporcionan mecanismos de navegación que permitan explorar los contenidos de la colección con distintos niveles de abstracción, como por ejemplo los proporcionados por los tesauros.

Una forma ampliamente utilizada para estructurar una colección de documentos es la utilización de sistemas de clasificación jerárquica, Koller & Sahami (1997). Estos parten de una taxonomía de conceptos previamente definida (por ejemplo el Directorio de Google, Open Directory Project [1]) y asignan cada documento entrante al concepto que mejor describe sus contenidos. De

[1] http://www.dmoz.org/.

esta forma, la clasificación resultante puede verse como un modelo conceptual mono-dimensional para representar la colección de documentos. Generalmente, estos clasificadores son supervisados, lo que hace muy costoso el proceso inicial de entrenamiento, que puede requerir un conjunto grande de documentos previamente etiquetados. Además, en un escenario tan dinámico como la Web, la taxonomía puede cambiar con frecuencia, lo que provoca una revisión continua del conjunto de entrenamiento y/o el no ajuste de la taxonomía manual a los contenidos de la colección, o que los tópicos de dicha taxonomía sean muy amplios. Todo ello provoca que este enfoque sea muy pobre para los procesos de análisis.

Alternativamente a la clasificación supervisada de documentos, existen numerosas aproximaciones de clasificación no supervisada o *clustering* de documentos. Una familia de métodos de clustering son los algoritmos jerárquicos, Pons (2004), que construyen una jerarquía de grupos en función de una medida de similitud entre documentos previamente definida. A diferencia de los clasificadores de documentos, estos algoritmos no requieren de ningún conjunto de entrenamiento y las jerarquías obtenidas se ajustan al contenido de la colección. Sin embargo, son muy sensibles tanto a la medida de similitud seleccionada como a la estrategia de clustering escogida. Ademas, el principal inconveniente de estas aproximaciones es que no proveen ninguna información al usuario acerca del contenido de cada grupo, y tampoco aseguran que los niveles jerárquicos formados por el algoritmo se corresponden con niveles de abstracción interesantes, Pons (2004).

Otra forma posible de estructurar una colección de textos consiste en calcular reglas de asociación o co-ocurrencias de palabras interesantes, y a partir de ellas construir árboles de decisión o alguna otra estructura informacional. Sin embargo, la naturaleza de los textos hacen poco útiles estas técnicas. En primer lugar, los términos se distribuyen siguiendo la ley de Zif (Power Law), de tal modo que hay pocos términos que son muy frecuentes y que además suelen tener poco contenido semántico. Los términos útiles suelen ser relativamente poco frecuentes, y se mezclan con una alta probabilidad con los términos muy frecuentes. De este modo, un soporte alto provoca un efecto no deseado, ya que elimina términos útiles. Por otro lado, un soporte bajo hace que se consideren demasiados términos que se combinan entre sí, produciendo una explosión combinatoria. Así pues, resulta necesario poder obtener co-ocurrencias interesantes de términos a partir de otro tipo de medidas de interés.

La intención de este capítulo es proveer una metodología para la construcción automática de espacios conceptuales multidimensionales con diferentes niveles de abstracción a partir de una colección de documentos. Con este modelo se podrá estructurar y analizar un colección de textos desconocida, infiriendo las diferentes vistas semánticas subyacentes o dimensiones, así como los distintos niveles de abstracción que pueden identificarse en ellas. Para ello, combinaremos técnicas

de minería de textos, taxonomías de dominio (WordNet, Fellbaum (1998)), y los resultados del capítulo anterior.

6.1 Importancia de un esquema multidimensional para colecciones de documentos

La representación más utilizada en los Sistemas de Recuperación de la Información es la conocida como modelo vectorial, Salton (1989). En este modelo, cada documento se representa como un punto de un espacio multidimensional, donde cada dimensión es un término de la colección. Claramente, este modelo presenta una dimensionalidad altísima (del orden de decenas de miles) y el espacio resultante es muy disperso. Aún con estas deficiencias, el modelo vectorial ha sido aplicado con éxito en la recuperación de documentos, el cálculo de similitud entre documentos y el agrupamiento automático de documentos.

A diferencia de este modelo, aquí se propone usar los espacios conceptuales multidimensionales para organizar y analizar una colección de documentos.

Por ejemplo, supongamos que tenemos las siguientes dimensiones en una colección de textos: Persona, Grupo, Actividad y Estado, todas ellas referentes al ámbito de la política. La dimensión Persona podría presentar las siguientes categorías: Persona.[líder], Persona.[opositor] y Persona.[diplomático] [1]. Estas categorías pueden descomponerse en otras subcategorías, por ejemplo Persona.[líder].[jefe de estado], Persona.[líder].[senador], etc. Finalmente, cada categoría tiene asociado un conjunto de miembros, por ejemplo, los miembros de Persona.[líder].[jefe de estado] podrían ser (presidente, canciller, primer ministro, rey). Y en el caso de una categoría con subcategorías, los miembros serían los nombres de las últimas, así los miembros de Persona.[líder] serían (jefe de estado, senador, líder político).

Los documentos que sean suficientemente *relevantes* con respecto a cada una de estas dimensiones se adscribirán a la correspondiente categoría de esa dimensión. Por ejemplo, un documento *d* que contenga la co-ocurrencia de términos (presidente, visita, Perú) podría adscribirse a las dimensiones (Persona.[líder].[jefe de estado].presidente, Actividad.visita, Estado.[América].Perú), respectivamente.

Con un modelo de este estilo, podemos abordar de forma natural varias tareas de análisis:

- **Navegación de contenidos.** Empleando operadores de generalización y selección podrían seleccionarse los temas y categorías de nuestro interés, sabiendo de antemano cuántos documentos nos vamos a encontrar en cada nivel.

[1] En estos ejemplos utilizamos una notación similar al lenguaje de consulta MDX: http://msdn.microsoft.com/library/en-us/dnsql90/html/imdxsmss05.asp.

- **Identificación de sucesos.** Mediante la especificación de un esquema, el usuario puede expresar los tipos de sucesos en los qué está interesado, indicando además el nivel de abstracción deseado. Por ejemplo, para analizar las actividades de las diferentes personalidades políticas a lo largo del tiempo, podríamos utilizar el siguiente cubo: (Persona,Actividad,Fecha).

- **Reconocimiento de entidades nombradas.** Los nombres propios identificados durante la fase de extracción de términos podrían ser adscritos a las dimensiones donde se adscriben los documentos que los contienen. De este modo, si una entidad nombrada co-ocurre fuertemente con algún valor de alguna dimensión, la entidad se adscribiría a dicha categoría. Así por ejemplo, si la entidad "Pancho García" co-ocurre con el valor presidente, este se adscribiría a Persona.[líder].[jefes de estado].presidente.

Pero, *¿Cómo abordar la inferencia de esquemas multidimensionales para una colección de documentos?*

La metodología propuesta se muestra en la Figura 6.1 y consiste en: 1) seleccionar desde los textos los términos[1] interesantes que deben aparecer en alguna dimensión a través de un proceso de desambiguación, 2) encontrar el sentido de cada palabra según el contexto en el que aparece, 3) extraer de WordNet subárboles de hiperónimos[2] asociados a los conceptos interesantes extraídos en el primer paso para formar las dimensiones conceptuales, 4) emplear los algoritmos de construcción de esquemas y espacios multidimensionales y permitir el análisis de la información de igual manera a como fue explicado en el capítulo anterior. Nótese que la selección de términos interesantes en los textos no es directa: sólo aquellos que aparezcan en una co-ocurrencia interesante son seleccionados. En la sección 6.2 se explica y justifica el proceso de selección de términos interesantes a partir de co-ocurrencias interesantes. En la sección 6.3 se explica el proceso de cálculo de las dimensiones y las condiciones que permiten el empleo de los ideas definidas en el capítulo anterior para el análisis de una colección de documentos.

[1] Un término se corresponde a lemas o frases (por ejemplo, entidades nombradas) extraídos directamente desde los textos.

[2] Una palabra es hiperónimo de otra si su significado engloba el de otras más específicas. A su vez las palabras con significado más específico son hipónimos de la primera. Por ejemplo, "mueble" es un hiperónimo de "silla" y "silla" es un hipónimo de "mueble".

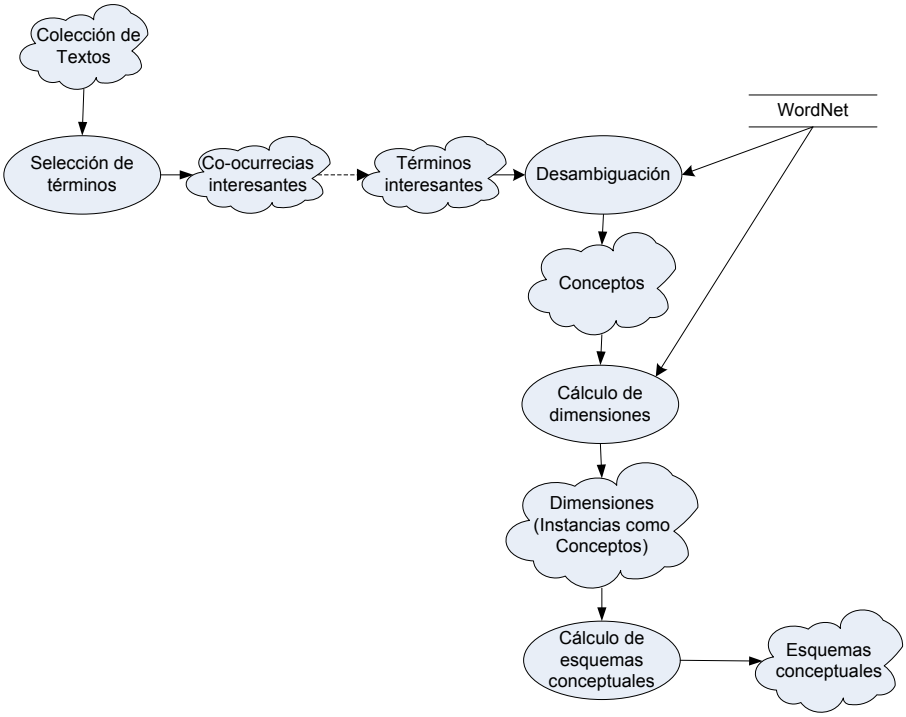


Figura 6.1: Propuesta para creación de espacios conceptuales multidimensionales.

6.2 Extracción de términos interesantes a partir de apariciones co-ocurrentes

Los términos que pueden resultar interesantes durante el proceso de análisis de la colección de documentos son aquellos que de alguna manera están relacionados con otros. En esta sección se describe un método para su recuperación considerando una cierta función que evalúa el interés de términos co-ocurrentes.

Sea $\mathcal{C} = \{d_1, \cdots, d_N\}$ una colección de documentos, donde cada documento d_i consiste en una secuencia de términos $w_{i_1} w_{i_2} \cdots w_{i_{m_i}}$. Sea además, $\mathcal{T}_\mathcal{C}$ el conjunto de todos los términos de la colección $\mathcal{C}$.

Definición 6.1. Una co-occurrencia de $\mathcal{C}$ es un conjunto no vacío de términos $cw \subseteq \mathcal{T}_\mathcal{C}$ que aparecen juntos en un conjunto de documentos $hits(cw) \subseteq \mathcal{C}$, $|hits(cw)| > 1$. Una *medida de interés de una co-ocurrencia* es una función que devuelve su relevancia para un propósito de análisis específico.

La medida de interés de una co-ocurrencia puede estar basada en la relación semántica de los términos de la co-ocurrencia y/o en el conjunto de documentos donde aparece dicha co-ocurrencia. La medida de interés más popular en la minería de datos es la del *soporte*, que asigna mayor peso

a aquellas co-ocurrencias que aparecen en la mayor cantidad de documentos. Otras funciones muy empleadas son la *proporción de probabilidad* (en inglés, Likelihood Ratio, Feller (1971)), la información mutua (Cover & Thomas (1991)), y *z-score* (Sheskin (1997)).

Un primer proceso de eliminación de palabras comunes y/o de poco interés para el indexado o análisis[1] garantiza que los conjuntos de términos co-ocurrentes extraídos no contengan palabras superfluas, pero el problema de la alta cardinalidad del conjunto de términos co-ocurrentes se mantiene (hasta 2^n para un conjunto de n palabras).

Con el objetivo de manejar de manera eficiente un conjunto relativamente pequeño de co-ocurrencias útiles, en el presente trabajo se propone aproximar el cálculo de los conjuntos de co-ocurrencias de la siguiente manera. Primeramente, se calcula el conjunto de pares de términos muy interesantes, según alguna medida de interés y una función de filtrado, luego se extiende cada conjunto de par de términos con palabras que co-ocurran con éstos para formar co-ocurrencias interesantes de mayor cardinalidad.

Un punto importante en este proceso es el de determinar qué granularidad utilizar para definir una co-ocurrencia. O sea, cada documento en una colección está compuesto de manera lógica por secciones, subsecciones, párrafos y oraciones. Así, una co-ocurrencia puede estar referida a alguna de estas partes si se observa su presencia en ella. Por supuesto, en el caso de secciones y subsecciones un análisis de sus contenidos (o tan sólo del contenido de sus títulos) o del objetivo de la extracción de co-ocurrencias debe preceder el procedimiento de recuperación de co-ocurrencias interesantes a estos niveles; pues por ejemplo, conjuntos de palabras que co-ocurren entre secciones que tratan temas diferentes ofrecerán, en la mayoría de los casos, respuestas con un bajo valor semántico. En Danger *et al*. (2003) fue descrito un algoritmo para el cálculo de co-ocurrencias considerando los diferentes niveles de estructuración sintáctica. En él se demostraron las diferencias entre las co-ocurrencias encontradas en uno u otro nivel y que el nivel de párrafo parecía fuese el más adecuado para realizar minados de colecciones de documentos para los que se desconoce sus contenidos.

A continuación se formaliza el proceso de cálculo de co-ocurrencias interesantes anteriormente descrito.

Definición 6.2. Sea $\phi_{interest}$ una función que evalúa el interés de una co-ocurrencia de términos. Sea además ϕ_{filter} una función que a partir de un conjunto de co-ocurrencias filtra aquellas irrelevantes. El conjunto de pares interesantes de una colección de documentos $\mathcal{C}$, $IP(\mathcal{C})$, se define por:

$$IP(\mathcal{C}) = \phi_{filter}(\{\{w_i, w_j\} | w_i, w_j \in \mathcal{T}_{\mathcal{C}}, w_j \in \arg\max_{w \in \mathcal{T}_{\mathcal{C}}} \phi_{interest}(\{w_i, w\})\})$$

[1] Las llamadas stopwords.

Nótese que la función $\phi_{interest}$ debe modelar el problema de la granularidad a emplear para formar las co-ocurrencias de los textos.

Definición 6.3. Sea $\phi_{extension}$ una función que permite extender una co-ocurrencia dada considerando los términos de los textos donde ella aparece. Sea $terms(d_i)$ el conjunto de términos contenidos en el documento d_i. Sea el conjunto de pares interesantes de una colección de documentos $\mathcal{C}$, el conjunto de co-ocurrencias interesantes de $\mathcal{C}$, $IC(\mathcal{C})$, viene dado por:

$$IC(\mathcal{C}) = \{\phi_{extension}(\{terms(d_i), d_i \in hits(p)\}) | p \in IP(\mathcal{C})\}$$

En la sección 6.4 se expondrán funciones que pueden ser utilizadas por las aquí llamadas $\phi_{interest}$, ϕ_{filter} y $\phi_{extension}$. Sin embargo, es importante tener presente que las funciones seleccionadas deben obtener conjuntos de co-ocurrencias con alto valor semántico. Es muy difícil definir una medida de *valor semántico de una co-ocurrencia*. Las encontradas en la literatura, requieren de recursos externos e igualmente su selección depende del uso que se le vaya a dar a dichas co-ocurrencias. Por ello, en lugar de intentar encontrar una medida que se ajuste a esta necesidad, pueden modelarse las funciones anteriores de manera que se garantice que el conjunto de co-ocurrencias obtenido es realmente interesante.

6.3 Construcción de las dimensiones conceptuales para colecciones de documentos. Vínculo con los espacios conceptuales.

Los términos de las co-ocurrencias interesantes forman la base para la creación de las dimensiones de los espacios multidimensionales. Cada uno de estos términos puede asociarse a algún concepto de WordNet y empleando su taxonomía de hiperónimos podemos reconstruir toda una dimensión. Veámoslo por pasos:

- *Asociación de términos a conceptos.* Éste no es más que el problema típico en PLN llamado *desambiguación de palabras*. Visto de la manera más general, se trata de asociar un concepto a cada término. Múltiples trabajos han sido desarrollados en esta línea de investigación, tal como se resume en Pons (2004) y Anaya-Sánchez *et al.* (2006). Pero, a pesar de los muchos esfuerzos, no existen diferencias significativas entre los resultados de los trabajos más prometedores, que ofrecen alrededor de un 70% de fiabilidad aplicando técnicas de aprendizaje automático. Sin embargo cuando no se dispone de conjuntos de entrenamiento los algoritmos no supervisados existentes apenas alcanzan el 50% de fiabilidad para todas las formas gramaticales.

Los métodos de desambiguación inician su proceso asociando cada término con el conjunto de sentidos que éste pudiera tener, y van descartando poco a poco algunos de estos sentidos hasta quedarse finalmente con el sentido de mayor probabilidad. Sin embargo, dos problemas surgen al intentar utilizar los métodos de desambiguación disponibles utilizando sólo las palabras de una co-ocurrencia:

- *no se cuenta con una sintaxis* que permita realizar ningún análisis gramatical y en correspondencia realizar la desambiguación. Los métodos de desambiguación a emplear, por tanto, tienen que ser capaces de intuir los sentidos sólo mediante el contexto ofrecido por el conjunto de palabras de cada co-ocurrencia. Claro está, siempre se puede recurrir nuevamente a los textos y realizar el análisis gramatical que se desee. Sin embargo, considerando que las co-ocurrencias permiten definir un contexto semántico, sería deseable algún algoritmo que no necesite de tal análisis. Ejemplos de algoritmos de este estilo son Montoyo & Palomar (2001); Sussna (1993); Voorhees (1993) y Anaya-Sánchez *et al.* (2006).
- difícilmente se logra recuperar un *sentido asociado a una frase* si las palabras a desambiguar no están en orden o no son términos multipalabras, tal como aquí ocurre. Con el objetivo de solventar este problema, la propuesta en este trabajo es enriquecer cada co-ocurrencia con frases que pudieran formarse con las palabras de la co-ocurrencia. De esta manera el algoritmo de desambiguación dará peso no sólo a las palabras independientes sino también a las frases que se han encontrado. El proceso de búsqueda de frases es muy simple y consiste tan sólo en reconstruir frases de WordNet con las palabras de la co-ocurrencia. Una implementación eficiente realiza la búsqueda en un *trie de frases* de WordNet eliminando las palabras de la lista de parada (*stopwords*).

Este proceso de desambiguación permite que se pueda poblar la siguiente ontología que recopila los conceptos mencionados en los documentos:

$Documento \equiv \exists seMenciona.\top \sqcap = 1 apareceEn.URI_{doc}$, siendo

URI_{doc} es la dirección URI donde aparece algún documento de la colección. Nótese que $\top$ permite que se asocie cualquier concepto (como los de WordNet) con los documentos donde éstos aparecen.

- *Reconstrucción de dimensiones*. Cada palabra desambiguada (a partir de ahora le llamaremos concepto de una co-ocurrencia) aparece en uno o más árboles de hiperónimos de WordNet.

La base de datos de hiperónimos WordNet puede verse como una ontología que define una taxonomía entre conceptos a través del constructor $\sqsubseteq$ que especifica las relaciones de subsunción entre conceptos. De esta forma, que concepto *A* sea hiperónimo del concepto *B* significa que $A \sqsubseteq B$ aparece en el Tbox de la ontología especificada por WordNet.

Por tanto, según las definiciones 5.1 y 5.2 pueden ser creadas dimensiones asociadas a la colección de documentos empleando el orden parcial dimensional determinado por las relaciones de subsunción de WordNet y el conjunto de conceptos que fueron reconocidos a partir del proceso de desambiguación de los términos interesantes. Hablando en términos de los árboles de hiperónimos de WordNet, este proceso no es más que la poda, en las taxonomías de conceptos de WordNet, de aquellos conceptos a los que no se le hace referencia alguna en las co-ocurrencias[1].

Todas los conceptos pudieran formar una única dimensión, sin embargo, una subdivisión que permite un análisis flexible y una mejor organización de la información, puede ser obtenida considerando los siguientes aspectos:

1. los niveles más altos en WordNet se corresponden a palabras muy genéricas que proporcionan poca información al usuario (por ejemplo *cualquier_cosa, entidad, entidad física*), de ahí que pueden ser eliminados. En nuestra experimentación trabajamos a partir del tercer nivel de abstracción de Wordnet, que se corresponde con palabras como *objeto físico* y *agente causal*.

2. en WordNet también tenemos la información referida a los dominios de conocimiento (por ejemplo: *ley*, *administración*, *economía* y *comercio*) asociados a cada concepto. De ahí, las dimensiones pueden ser enriquecidas si se define como concepto más general para todas las palabras que comparten un dominio particular dicho dominio. Para hacerlo, 1) hace falta "alterar" las definiciones originales de WordNet, incluyendo como conceptos de segundo nivel los asociados a cada dominio[2] y a partir de estos conceptos-dominio reproducir las taxonomías conceptuales que determina WordNet para las palabras que se refieren a dicho dominio; 2) recuperar los dominios más frecuentes para cada co-ocurrencia. De esta forma, todas las palabras de una co-ocurrencia serán adscritas a dimensiones que se relacionan con los dominios de dicha co-ocurrencia.

[1] Que no se le haga referencia a un concepto *A* significa que ningún hiperónimo de los conceptos de las co-ocurrencias contiene a *A*.

[2] En el primer nivel de abstracción aparece sólo el concepto *cualquier_cosa* (en inglés, thing).

La ontología que representa la colección de documentos contiene entonces la definición de *Documento* y los axiomas de subsunción de WordNet para todos los conceptos referidos en al menos en un documento de la colección y en las instancias de *Documento*.

De esta forma, con las dimensiones ya formadas, pueden ser creados los espacios conceptuales multidimensionales asociados a los documentos utilizando los diferentes caminos que pueden ser obtenidos desde *Documento* a cualquier concepto de WordNet. Por ejemplo:

Puede formarse el espacio $E = \langle D_1, D_2, D_3, D_4, c_{\phi_{vacio}}, c_{\phi_{vacio}}, c_{\phi_{vacio}}, c_{\phi_{union}} \rangle$, siendo:

D_1, la dimensión construida para los conceptos subsumidos por el dominio de *administración*, D_2, la dimensión construida para los conceptos subsumidos por el dominio de *ley*, D_3, la dimensión construida para los conceptos subsumidos por el dominio de *economía*, D_4, la dimensión construida para el concepto *URI*,

$c_{\phi_{vacio}} = \emptyset$, $c_{\phi_{union}} = \{\langle \phi^D, \hat{\cup} \rangle | \phi^D \in \Phi\}$, con $\hat{\cup}(d_1, ..., d_n) = \{d_1\} \cup ... \cup \{d_n\}$.

Es importante notar que cada una de las dimensiones D_1, D_2, D_3 aparecen gracias a los caminos de agregación: $\langle \{seMenciona\}, concepto_i \rangle$, siendo $concepto_i$ el concepto asociado a la dimensión i, eso es: *administración*, *ley* y *economía*. Así mismo puede verse que para dichas dimensiones la función de combinación de datos simples es $c_{\phi_{vacio}}$ pues las instancias en estas dimensiones son todas representadas por conceptos nominales, por tanto no contienen relaciones, y en consecuencia no contienen tipos de datos que necesiten ser combinados.

La función de combinación $c_{\phi_{union}}$ permite que las generalizaciones en las instancias *Documento* al mismo tiempo que "abstraen" los conceptos mencionados en los documentos, mantiene la lista de documentos asociados a los conceptos de la instancia generalizada en la cuarta dimensión. La dimensión D_4 se obtiene por el camino de agregación $\langle \{apareceEn\}, URI_{doc} \rangle$.

Además, considerando todo lo anterior, podría emplearse el algoritmo de generación de esquemas multidimensionales interesantes, Algoritmo 5.2 y pueden recrearse los espacios conceptuales multidimensionales para la colección de documentos inicial usando el Algoritmo 5.1, según los intereses del usuario. Un analista económico, por ejemplo, podría interesarse en el espacio anteriormente descrito.

Finalmente, un análisis más profundo puede ser realizado de igual manera a como fue explicado en la sección 5.7.

Los espacios conceptuales multidimensionales descritos en esta sección son también importantes porque permiten "clasificar" de cierta forma la colección de documentos, pues el analista de la colección podría reconocer los conceptos asociados a cada documento. Si además cuenta con una biblioteca de ontologías de dominio, por ejemplo de economía, derecho, etc., podría intentar aplicar el algoritmo de extracción de instancias ontológicas del Capítulo 4 sobre los documentos asociados a dichas ontologías.

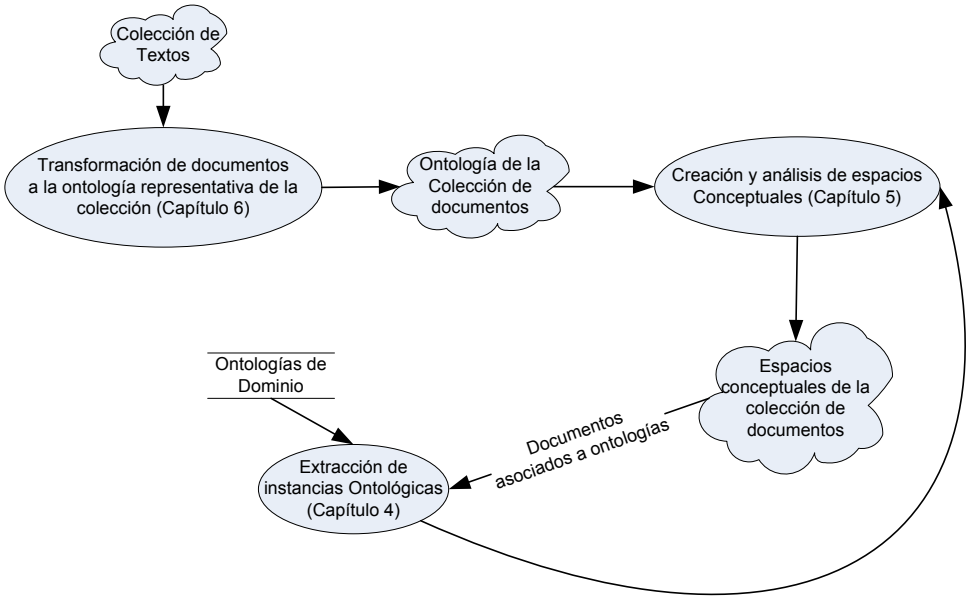


Figura 6.2: Ciclo de extracción y análisis de la información desde la perspectiva de la Web Semántica.

De esta manera se cierra el ciclo de extracción y análisis de información desde la perspectiva de la Web Semántica propuesto en esta tesis y representado gráficamente en la Figura 6.2.

6.4 Resultados experimentales

No existen colecciones de documentos con textos del dominio de arqueología con las que realizar una experimentación que considere todo el ciclo de extracción y análisis anteriormente descrito. Por ello, para la evaluación de la propuesta hemos utilizado la colección estándar TDT2[1] (versión 4.0 de LDC, Linguistic Data Consortium), que es comúnmente empleada en el área de Recuperación de Información. Esta colección contiene 9824 noticias clasificadas manualmente en 193 tópicos, las que han sido procesadas utilizando el analizador morfosintáctico Tree-Tagger[2] para obtener los lemas de las palabras. El fichero invertido (estructura de datos usual en los SRI) final contiene 55.112 términos.

Para evaluar la bondad de las co-ocurrencias seleccionadas con la medida de fuerza, se ha realizado una serie de experimentos que miden diferentes aspectos de calidad de las co-ocurrencias para discriminar documentos de tópicos diferentes (pues una co-ocurrencia es útil en la medida que "dice algo acerca de un tópico"). Así, se han seleccionado dos medidas de calidad: la clásica F-medida para comparar los documentos de las co-ocurrencias con los documentos de los tópicos,

[1] http://www.ldc.upenn.edu/Catalog/index.jsp

[2] http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/DecisionTreeTagger.html.

α	**|CoOccs|**	**F1**	**Coseno**
0,7	5865	0,620	0,420
0,6	7416	0,689	0,446
0,5	8976	0,727	0,505
0,3	12325	0,743	0,512

Tabla 6.1: Calidad de las co-ocurrencias detectadas.

y la medida del coseno para comparar las co-ocurrencias con las descripciones de los tópicos. Para las medidas globales se toma siempre la macro-media obtenida con los pares tópico/co-ocurrencia que maximizan la correspondiente medida.

En la Tabla 6.1 se muestran los resultados obtenidos utilizando los siguientes funciones como parámetros para el proceso de selección de palabras interesantes:

- $f_{interest}(cw) = |hits(cw)|$, la medida de interés de una co-ocurrencia, que considera una co-ocurrencia más interesante en la medida en que ella aparece en mayor cantidad de documentos de la colección.

- $f_{filter} = \frac{|hits(cw)|}{min_{w_i \in cw}|hits(w_i)|} > \alpha$, la función de filtro de pares de palabras co-ocurrentes, que premia a aquellas co-ocurrencias cuyo conjunto de documentos difiere poco del conjunto de documentos de mínima cardinalidad entre los asociados a los términos de la co-ocurrencia; en la medida que aumenta α, menor diferencia puede existir entre los conjuntos de documentos que se comparan, por tanto mayor cohesión debe haber entre los términos seleccionados y menor debe ser la cantidad de éstos,

- $f_{extension} = \{\bigcap_{d_i \in hits(p)} terms(d_i) | p \in IP(\mathcal{C})\}$ como función de extensión de pares frecuentes, que extiende un par de términos interesante con todas las palabras que co-ocurren con ellas en los documentos donde el par aparece.

Los pares interesantes son obtenidos de una manera muy eficiente con el fichero invertido de la colección de documentos utilizando el orden impuesto por los tamaños de las listas de documentos de las palabras, para minimizar las intersecciones requeridas para su cálculo. La expansión de los pares interesantes es también realizada de manera eficiente mientras se calculan los pares de máximo interés.

Como puede observarse en la Tabla 6.1 el número de co-ocurrencias es inversamente proporcional a α, y las medidas de calidad también crecen inversamente respecto a la reducción de este mismo parámetro. De esta manera, el cubrimiento de los tópicos decrece proporcionalmente con

Soporte mínimo	Co-ocurrencias generadas	% Tópicos cubiertos por co-ocs.
60	3528717	14% (25)
100	483719	9% (16)
300	9071	3.4% (6)

Tabla 6.2: Resultados del FP-growth.

Categoría gramatical	Sentidos recuperados	Recuperación	Precisión	F1
Todos	3382	0.7351	**0.8338**	0.7813
Sustantivos	2876	0.8519	**0.8519**	0.8519
Verbos	118	0.2428	0.5536	0.3376
Adjetivos	388	0.4407	0.7848	0.5644

Tabla 6.3: Resultados de la desambiguación.

respecto al número de co-ocurrencias filtradas. Por tanto, contrariamente a lo que ocurre en los algoritmos clásicos de la minería de datos, el método propuesto (al menos con esta combinación de funciones) no es afectado por la tendencia de la ley de Zipf sobre la distribución de los tópicos. Además, estos resultados son comparables a los obtenidos por los sistemas específicos de detección de sucesos (con F1 entre 0,75 y 0,8). Por tanto, la calidad de las co-ocurrencias obtenidas es aceptable para nuestro propósito de análisis.

Se ha evaluado además los conjuntos de co-ocurrencias de palabras tradicionalmente empleado en minería de textos. Se ejecutó para ello el algoritmo FP-growth[1] a los vectores de términos de los documentos para obtener co-ocurrencias frecuentes de términos. La Tabla 6.2 muestra los resultados alcanzados por este algoritmo para diferentes valores de soporte. Como puede observarse el algoritmo genera un número aceptable de co-ocurrencias para valores muy altos de soporte, lo que implica dejar sin cubrir muchos tópicos.

Un asunto crítico para el proceso de inferencia de las dimensiones es la precisión del método de desambiguación aplicado. El método empleado para la desambiguación, Anaya-Sánchez *et al.* (2006), ha sido evaluado con la colección SEMCOR[2], que contiene 186 documentos y un total de 22680 términos etiquetados semánticamente. Para ello, fueron extraídos los términos interesantes utilizando las funciones anteriormente explicadas, y se corroboró el porcentaje de aciertos de la desambiguación obtenida para los términos interesantes (considerando los términos etiquetados del SEMCOR). En la Tabla 6.3 aparece un resumen de los resultados de desambiguación

[1] Una implementación eficiente de este algoritmo aparece en: http://fimi.cs.helsinki.fi/src/fimi19.tgz.

[2] http://multisemcor.itc.it/semcor.php

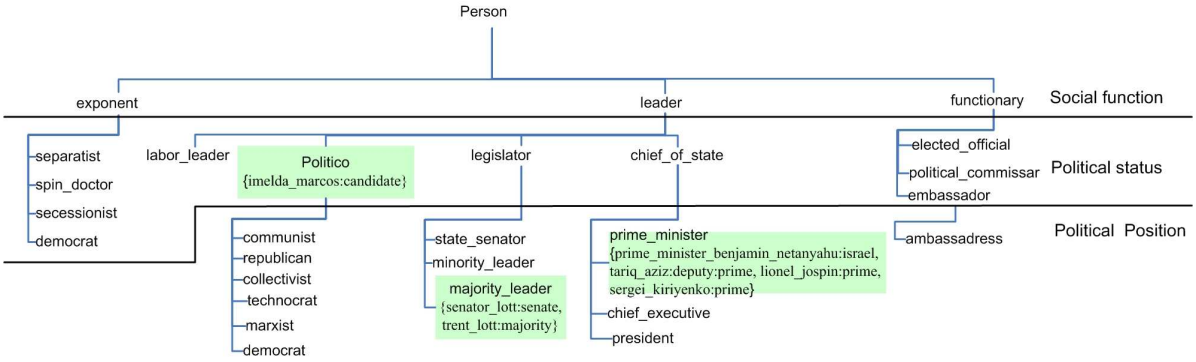


Figura 6.3: Ejemplo de una dimensión obtenida por el sistema. En este caso se asocia a personalidades políticas. Nótese que se ha señalado algunas de las entidades (nombres propios) que se han logrado asociar a conceptos particulares.

Dimensiones del esquema multidimensional	**Dominios**	**Soporte**
(Planet, Earth)	astronomy, earth	58
(Territory, Region, Organization, Leader)	administ.#earth, economy, politics	58
(Territory, Substance, Creature)	administ.#earth, aliment., biology	54
(Territory, Health_problem, Artifact)	administ.#earth, medicine	47
(Territory, Due_process_of_law, request)	administ.#earth, law	42
(Amount, Artifact)	economy, medicine	24

Tabla 6.4: Esquemas conceptuales multidimensionales inferidos pata la colección TDT2.

alcanzados. Se ha de destacar una aproximación total del 83% y un 85% para los sustantivos.

Un pequeño esbozo de una de las dimensiones obtenidas por el sistema es mostrada en la Figura 6.3. El conjunto completo de las dimensiones puede ser encontrado en Danger (2006). Se obtuvieron un total de 107 dimensiones, que con un total de 3551 conceptos, una profundidad promedio de 2.4 niveles (pero considerando solo aquellas co-ocurrencias con alto valor de la medida F1 con respecto a los tópicos, el número de conceptos se reduce a 1270 y una profundidad promedio de 3.2 niveles) y una cobertura del 90,6% de los documentos iniciales.

Con el objetivo de revisar algunas combinaciones de dimensiones interesantes, se empleó el algoritmo FP-growth para descubrir conjuntos de conceptos de nivel 3 de WordNet co-ocurrentes entre los documentos. En la Tabla 6.4 se muestran las combinaciones de mayor soporte, relacionados éstos con los dominios de conocimientos que se les asocian.

Finalmente los espacios conceptuales multidimensionales obtenidos pueden ser explorados

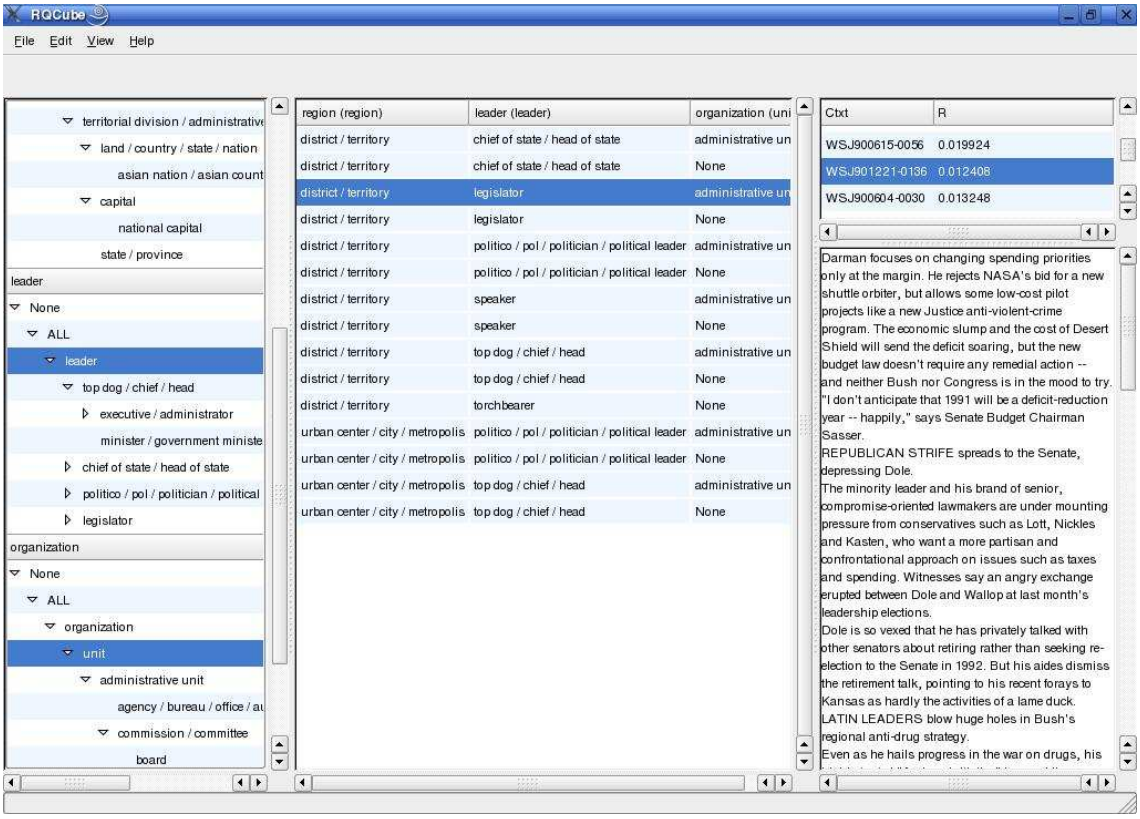


Figura 6.4: Dimensiones y espacio conceptual multidimensional asociado a los conceptos *región*, *organización* y *líder*. A la izquierda pueden verse cada una de las dimensiones; en el centro el espacio conceptual determinado por dichas dimensiones y a la derecha los documentos asociados a los hechos del espacio conceptual.

utilizando herramientas de tipo OLAP (puesto que pueden obtenerse fácilmente dimensiones que cumplen con las restricciones OLAP), que permitan además asociarte un nivel de relevancia a cada hecho (conjunto de palabras mencionadas en la colección) del espacio multidimensional. Estas propiedades han sido tratadas en otra tesis, mediante la introducción de los denominados R-Cubos, Pérez *et al.* (2005). En la Figura 6.4 se muestra la interfaz de navegación para el segundo esquema conceptual de la Tabla 6.4. Como puede apreciarse un reportero de noticias del ámbito de la política, con tal esquema conceptual podría revisar los lugares (regiones) donde han ocurrido diferentes acontecimientos, asociados éstos con las instituciones políticas (organizaciones) y los cargos o tendencia política de los líderes involucrados en el evento (como aparece a partir de *leader* en la Figura 6.3). Además, navegando a través de los diferentes conceptos de las dimensiones, otras co-ocurrencias conceptuales interesantes (no exactamente aquellas que se corresponden a acontecimientos históricos, sino otras que, por ejemplo, muestren relaciones poco evidentes que aparecen repetidamente en múltiples eventos) pueden ser identificados.

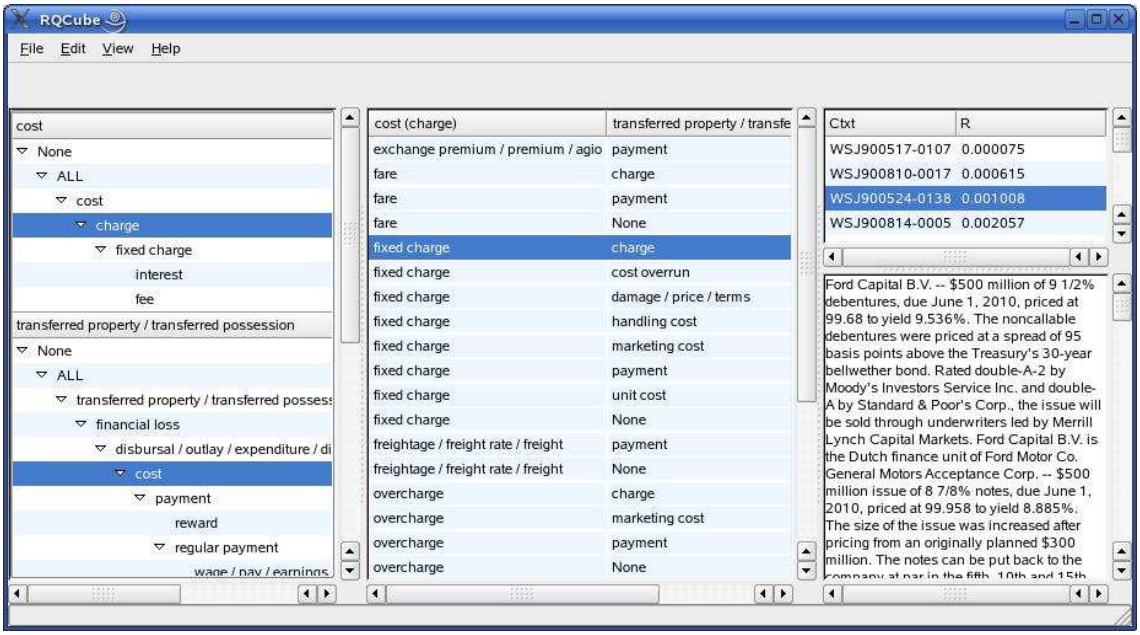


Figura 6.5: Dimensiones y espacio conceptual para el par de conceptos *costo* y *propiedad transferida*. A la izquierda pueden verse cada una de las dimensiones; en el centro el espacio conceptual determinado por dichas dimensiones y a la derecha los documentos asociados a los hechos del espacio conceptual.

En las Figuras 6.5, 6.6 y 6.7 se muestran otro posible entorno de trabajo, relacionado en esta ocasión con el ámbito económico. En estas figuras se han descrito algunas de las operaciones típicas (selección, abstracción, etc.) que pueden ejecutarse sobre estos espacios conceptuales multidimensionales.

6.5 Conclusiones

En este capítulo se ha propuesto un método que identifica un conjunto de términos interesantes para representar una colección de documentos, reconstruye con dichos términos dimensiones conceptuales, que luego se emplean para formar espacios conceptuales como los descritos en el capítulo anterior. Los resultados experimentales muestran que las palabras incluidas en las co-ocurrencias (base para la selección de palabras de los espacios conceptuales) presentan una buena cualidad en tanto permiten describir los tópicos descritos en la colección de documentos. Asimismo las dimensiones inferidas contienen un alto significado para los propósitos de análisis y su número es razonable para permitir una navegación adecuada a los usuarios.

La propuesta de este capítulo es interesante además porque permite reconocer los conceptos tratados en los documentos y con ello brindar la posibilidad de asociarles una ontología específica. Los documentos asociados a una ontología específica pueden ser tratados como en el Capítulo 4

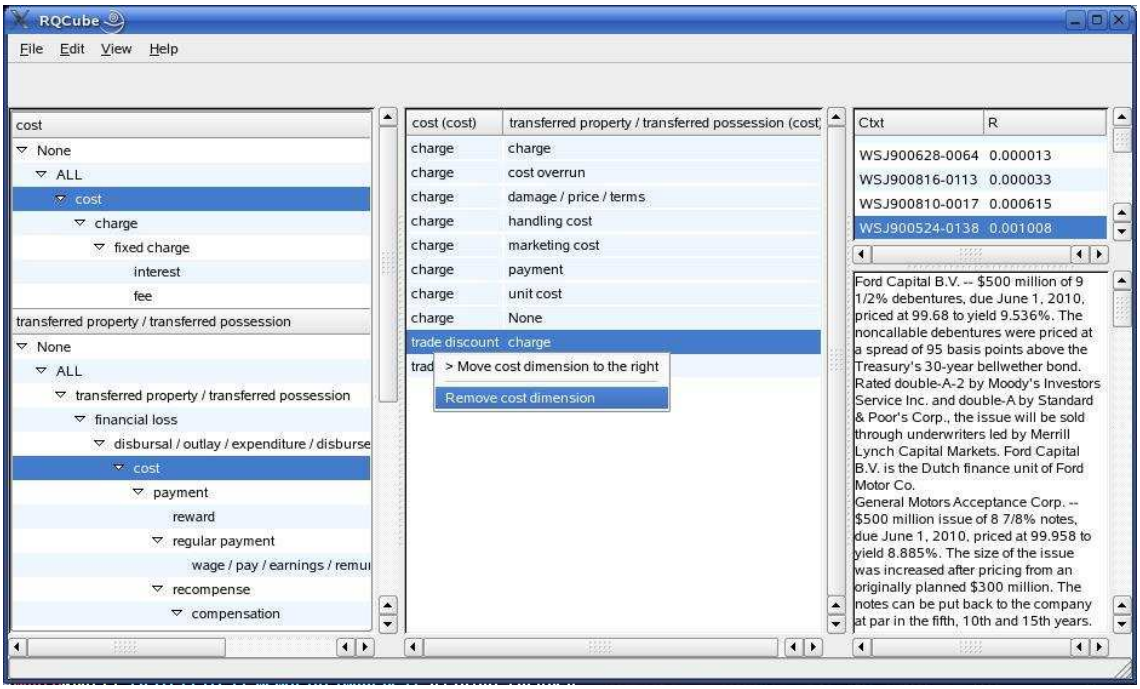


Figura 6.6: Ejemplo de operaciones a realizar con el espacio conceptual. En este caso se realiza una selección para eliminar el concepto *costo* del espacio conceptual.

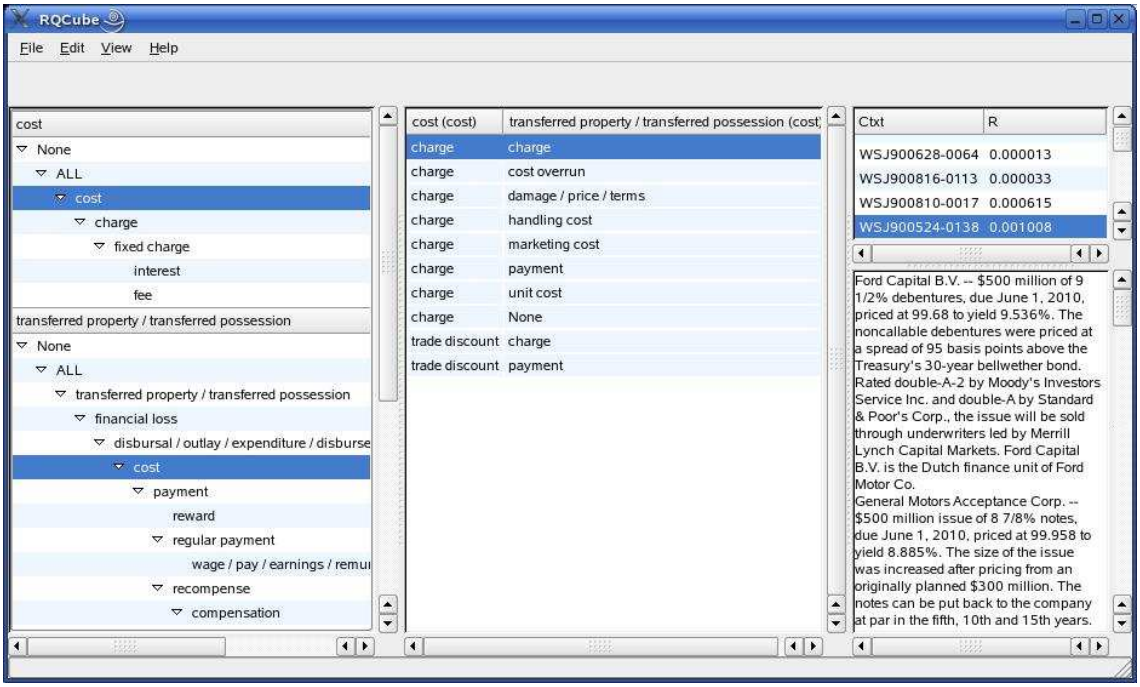


Figura 6.7: Ejemplo de operaciones a realizar con el espacio conceptual. En este caso puede notarse, al comparar con la Figura 6.5, una abstracción en la primera de las dimensiones desde el concepto *precio* (*charge*) al concepto *costo*.

de manera que pueda poblarse la ontología (y por ende la Web Semántica) con las nuevas instancias descubiertas. Por último, estas instancias pueden ser usadas en procesos análisis como fue explicado en el Capítulo 5, cerrándose así un ciclo de extracción y análisis de información desde colecciones de documentos para las que se desconoce los temas o áreas de conocimiento que abarcan.

En el futuro, pueden ser estudiadas otras medidas de interés para la selección de las palabras con el objetivo de proveer diferentes colecciones de dimensiones de acuerdo a los propósitos de análisis. Por último, es importante mantener actualizado el método de desambiguación a aquel que proporcione las mejores prestaciones.

Capítulo 7

Conclusiones

En este trabajo se han abordado los principales problemas para el desarrollo de la Web Semántica. En primer lugar, se propone una solución viable para la extracción de instancias ontológicas desde textos escritos en lenguaje natural. Se expone un marco de trabajo para crear espacios conceptuales que permitan el análisis de dichas instancias ontológicas. Además, se expone una solución para la clasificación y análisis de documentos para aquellas colecciones de textos para los que no se disponga una ontología de dominio particular. Todas las soluciones dadas han sido unificadas empleando los conceptos de la lógica descriptiva y sus algoritmos para el razonamiento, así como las soluciones dadas a dos tipos de problemas de razonamiento identificados en la presente tesis: el cálculo de caminos de agregación entre conceptos y operaciones entre instancias ontológicas.

La sección 7.1 describe con detalle las aportaciones de la presente investigación. La sección 7.2 enumera las publicaciones obtenidas durante el desarrollo de la presente tesis doctoral. Por último, las limitaciones del trabajo actual así como propuestas de trabajo futuro son expuestas en la sección 7.3.

7.1 Aportaciones

Las principales contribuciones del presente trabajo para el desarrollo de la Web Semántica pueden resumirse en:

- Se ha identificado y solucionado el **problema de cálculo de caminos de agregación** entre conceptos de la lógica descriptiva. Los caminos de agregación entre conceptos abre las puertas para muchas aplicaciones, no solamente en el campo del razonamiento lógico, sino además en otros campos como las Bases de Datos, Minería de Datos y Procesamiento del Lenguaje Natural, tal como fue descrito en el Capítulo 3 y reafirmado en los Capítulos 4, 5, y 6. La solución aportada examina y extiende el algoritmo tableau de razonamiento para

el lenguaje SHOIQ(D), y provee un algoritmo que permite recuperar todos los caminos que podrían aparecer en instancias ontológicas de la ontología que se examina.

- Se ha identificado y provisto solución para varias **operaciones sobre instancias ontológicas**. Ellas son: especialización, abstracción, unificación, agregación, diferencia y diferencia simétrica entre instancias ontológicas. Las cuatro primeras permiten actualizar axiomas del *Abox* para incluir nuevas características observadas sobre las instancias que se describen. Las dos restantes han sido propuestas con el objetivo de proveer mecanismos para la comparación de instancias ontológicas. La notación empleada para describir los operaciones sobre las instancias permite que con poco esfuerzo puedan ser trasformadas y utilizadas en otros paradigmas de modelación como *UML*.

- Se ha descrito una **metodología para la población de la Web Semántica**, que discierne entre la presencia de la Web dinámica, la estática y fuentes externas cuya información se desea exponer públicamente a través de la Web Semántica. En el caso de la web dinámica (o fuentes externas) cuya información proviene de almacenes estructurados, se propone el desarrollo de la Web Semántica desde las mismas entidades corporativas donde es conocida la semántica y el dinamismo de los datos. En el caso de datos no estructurados escritos en lenguaje natural, se propone primeramente asociar los documentos a ontologías y posteriormente extraer las instancias ontológicas desde los textos empleando el método descrito en el Capítulo 4.

- Se ha propuesto un **algoritmo novedoso para la extracción de instancias ontológicas** que se guía por las especificaciones de dominio contenidas en una ontología para realizar y conformar instancias complejas. Gracias a inferencias en el *Tbox* de las ontologías se evita la necesidad de realizar complejos análisis sintácticos y semánticos en los textos tratados. Además se han incluido heurísticas que permiten reconstruir no solamente instancias simples sino descripciones de instancias generalizadas e instancias con relaciones negadas. Se ha descrito también la metodología para retroalimentar al sistema de los errores u omisiones por él cometidos. A pesar de la sencillez de los análisis lingüísticos, el método descrito obtiene una precisión y recuperación del 93% y 82% respectivamente, lo cual constituyen resultados muy satisfactorios.

- Se ha propuesto un **marco de trabajo para el análisis de instancias ontológicas** que permite a partir de los caminos de agregación entre conceptos formar dimensiones y esquemas de espacios conceptuales. Se ha descrito un algoritmo para formar los espacios conceptuales

desde instancias de una ontología, y otro que permite generar esquemas de espacios conceptuales interesantes con el objetivo de formar reglas de caracterización y discriminación de clases con elevado interés. Se ha descrito además cómo adaptar y extraer algunos patrones de comportamiento de los datos de manera similar a las descritas en el área de la Minería de Datos.

- Se ha descrito un método para la **creación de espacios conceptuales para colecciones de documentos heterogéneas**, que por un lado permite organizar y analizar una colección de documentos, y por otro posibilita puedan generarse asociaciones entre ontologías existentes y documentos de la colección, y en consecuencia la población con nuevas instancias la Web Semántica.

7.2 Publicaciones relacionadas

Durante la realización de la presente investigación varias contribuciones han sido publicadas en la comunidad científica. A continuación son enumeradas y desglosadas para cada uno de los capítulos.

Relacionadas con el Capítulo 4:

- En Danger *et al.* (2004c) se expuso la primera aproximación para la extracción de instancias desde una colección de documentos desde la perspectiva del minado de textos. El objetivo fundamental de este primer trabajo fue dar una visión general de qué tipos de objetos y atributos aparecían con frecuencia entre los documentos de arqueología. Ello nos permitió tanto enriquecer la ontología original con conceptos y relaciones que anteriormente no aparecían, así como formar un conjunto de instancias representativas de la muestra de documentos.

- En Danger *et al.* (2004a) se exponen las primeras ideas para poblar la Web Semántica bajo el principio de poco tratamiento lingüístico. Este primer trabajo las ontologías fueron descritas en forma de *frames*, pero igualmente trataba muchos casos especiales en las asociaciones entre conceptos.

Relacionadas con el Capítulo 5:

- Aún sin la perspectiva general que provee la Web Semántica, dos trabajos iniciales: Danger *et al.* (2004b) y Garcia *et al.* (2006), han sido desarrollados con el objetivo de explorar el minado de objetos complejos. El segundo trabajo es una extensión del primero que permite reducir la dimensionalidad de las subdescripciones interesantes obtenidas.

Relacionadas con el Capítulo 6:

- En los trabajos Danger & Berlanga (2005) y Danger & Berlanga (2006) se exponen las ideas y experimentos descritos en el Capítulo 6. El segundo de ellos contiene una formalización más generalizada y un estudio experimental más completo que el primero.

Otras publicaciones:

- El estudio provisto en Danger & Berlanga (2001) ha servido de base para la extensión de la ontología de arqueología, como se expuso en Danger *et al.* (2004c) y en estudios posteriores acerca del minado de objetos complejos como en Danger *et al.* (2004b) y Garcia *et al.* (2006). Este trabajo provee un conjunto de algoritmos descritos y analizados con profundidad para el minado de bases de datos y colecciones de textos.

- En Danger *et al.* (2003) se describe un algoritmo para el minado de colecciones de documentos sintácticamente estructurados, que permitió reconocer las diferencias semánticas a diferentes niveles de estructuración sintáctica. Gracias a este trabajo, por ejemplo, se pudo comprender que el nivel párrafo era suficiente para formar instancias suficientemente complejas y que el anidamiento sintáctico de los documentos podría ayudar a formar instancias complejas que englobasen a todo el documento, tal como fue explicado en la sección 4.4.3.

7.3 Limitaciones y trabajo futuro

Varias limitaciones pueden ser observadas en las soluciones dadas en la presente tesis:

- El poco análisis lingüístico ha permitido obtener resultados muy satisfactorios en el proceso de extracción de instancias ontológicas desde textos. Sin embargo, algunos de los errores cometidos por el sistema se deben precisamente a tal filosofía. Afortunadamente los errores más importantes revisados hasta el momento pueden son resueltos con un análisis sintáctico sencillo.

- La formalización dada para el análisis de instancias ontológicas puede ver afectado su rendimiento conforme aumente el número de instancias a tratar.

- Se ha expuesto la necesidad de que OWL provea tipos y anotaciones flexibles que permitan describir interacciones con servicios web u otras aplicaciones para dar mayor flexibilidad a las herramientas que sobre la Web Semántica se creen. Por ejemplo, en el Capítulo 4 tal interacción fue útil para definir las funciones de verificación para las entidades ontológicas.

En el Capítulo 5 para describir las funciones de combinación a emplear para los datos simples.

De esta forma, los trabajos iniciados en la presente investigación pueden ser completados y/o mejorados considerando las siguientes direcciones:

- Revisar los errores y omisiones del sistema de extracción de instancias ontológicas con el objetivo de descubrir nuevas heurísticas que mejorar las prestaciones del sistema.
- Realizar una experimentación extensa que considere múltiples ontologías, idiomas y diversas complejidades de las descripciones.
- Revisar la viabilidad de emplear las técnicas OLAP para reducir al máximo los costes de ejecución y minimizando las restricciones a imponer en los datos de cara a una futura implementación del formalismo descrito para el análisis de instancias ontológicas.
- Estudiar y proponer a W3C mecanismos de anotación más flexibles así como la inclusión de tipos de datos que puedan ser descritos, manipulados y reconocidos por servicios web o aplicaciones propias de los usuarios.

Una visión más generalizada de los problemas que aún existen en la Web Semántica, permite definir algunos otros trabajos interesantes. Durante desarrollo e interacción con el mundo de la arqueología una importante entidad ontológica resultó muy difícil de modelar e imposible de tratar a través de los razonadores actuales de la lógica descriptiva: el tiempo. La problemática está en que un objeto varía su funcionalidad, sus características tipológicas e incluso su importancia a lo largo del tiempo y por otro lado, la noción de tiempo que tienen los arqueólogos dista mucho de la noción de tiempo que tenemos la mayoría de nosotros.

Un nuevo y ambicioso proyecto de investigación que el grupo TKGB[1] de la Universidad Jaume I se encuentra liderando la parte la modelación del conocimiento, ha encontrado los mismos problemas en relación al tiempo. En este caso, se debe modelar las diferentes fases o estadios de una enfermedad en un paciente. Una vez más se demuestra que los cambios que se producen a través del tiempo requiere de algo más que lo provisto por la lógica descriptiva.

De ahí que un trabajo de especial interés puede ser **ampliar los constructores de la lógica descriptiva de OWL para que considere objetos "cambiantes" en tiempo**, y además permitir **desarrollar esquemas temporales ad-hoc** a las necesidades del conocimiento representado.

En este mismo proyecto, una segunda meta es la de tratar los diferentes niveles de composición de la materia que aparecen los organismos vivos. De manera que puedan realizarse descripciones

[1] TKGB, Temporal Knowledge Databases, es el grupo de investigación en el que participo.

(y por ende estudios) desde el nivel molecular hasta el nivel social. Aunque está claro que las agregaciones son la base para construir tal jerarquía, son imprescindibles **operadores flexibles de composición de relaciones** que permitan describir relaciones a través del empleo de otras. Una iniciativa para tratar tal problemática es provista en la nueva versión de OWL 1.1, sin embargo, aún se está a tiempo de plantear nuevos constructores y semánticas necesarias en aplicaciones reales.

Tanto para el tratamiento del tiempo como para el tratamiento de agregaciones por nivel, son necesarios mecanismos de almacenamiento flexibles que permitan arribar a la entidad exacta de análisis que se requiera de manera rápida y mecanismos de razonamiento que permitan comprender cuándo se está en presencia de una entidad "sobrecargada" (una entidad que es muchas a su vez) y cuándo prescindir de una parte de su descripción de acuerdo a las demandas formuladas.

En otro orden de cosas, W3C ha descrito poco sobre las **capas de reglas y trust**. Un estudio profundo sobre qué es necesario almacenar, con qué nivel de confianza y cómo recuperar las reglas inferidas es muy importante. Herramientas flexibles para **incorporar conocimientos a la Web Semántica para usuarios generales** y **para la ejecución se búsquedas semánticas a gran escala**, son trabajos que en breve serán de una enorme utilidad.

La Web Semántica a pesar de todo no será la última palabra, no tiene toda la expresividad para describir e inferir todo lo que se conoce. Los pasos actuales son importantes para crear una tercera generación de Web, llamémosle la **Web Instruida**, que no sólo sabe, y razona según la lógica descriptiva, sino según lógicas espaciales, temporales, borrosas, modales, predictivas, etc. porque cada una aporta una visión del mundo, y todas las percepciones son importantes.

Apéndice A

Primitivas principales OWL Lite, OWL DL y RDFS y su representación en la notación DL

OWL DL	
owl:hasValue	$\exists R.\{o\}$
owl:oneOf	$\{o\}$
owl:unionOf	$C \sqcup D$
owl:complementOf	$\neg C$
owl:disjointWith	$C \sqcup D \sqsubseteq \bot$
OWL Lite	
owl:Ontology, owl:VersionInfo, owl:imports, owl:backwardCompatibleWith, owl:incompatibleWith	Permiten describir la ontología
owl:DeprecatedClass, owl:DeprecatedProperty	Permiten especificar clases y propiedades obsoletas
owl:Class, owl:Restriction, owl:onProperty	Permiten definir clases
owl:allValuesFrom[a]	$\forall R.C$
owl:someValuesFrom[a]	$\exists R.C$
owl:minCardinality[b]	$\leq nR$
owl:maxCardinality	$\geq nR$
owl:cardinality	$= nR$
owl:intersectionOf[c]	$C \sqcap D$
owl:ObjectProperty	R
owl:DataTypeProperty	u
owl:TransitiveProperty	^{+}R
owl:SymetricProperty[d]	$R = R^{-}$
owl:FunctionalProperty[d]	$\leq 1R$
owl:InverseFunctionalProperty[d]	$\leq 1R^{-}$
owl:AnnotationProperty	Permite describir una propiedad
owl:Thing	$\top$
owl:Nothing	$\bot$
owl:inverseOf	R^{-}
owl:equivalentClass[c]	$C \equiv D$
owl:equivalentProperty	$R \equiv S$
owl:sameAs, owl:sameIndividualAs, owl:differentFrom, owl:AllDiferent, owl:distinctMembers	Permiten describir el Abox
RDF(S)	
rdf:Property	R
rdfs:subPropertyOf	$R \sqsubseteq S$
rdfs:domain[a]	
rdfs:range[a]	
rdfs:comment, rdfs:label, rdfs:seeAlso, rdfs:isDefinedBy	Permiten describir elementos de la ontología
rdfs:subClassOf[c]	$C \sqsubseteq D$

[a] En RDF(S) y OWL-Lite sólo puede ser usada con identificadores de clases y tipos de datos nombrados. En OWL-DL, se extiende a cualquier expresión que describa clases.

[b] En OWL-Lite n se restringe al intervalo [0, 1]. En OWL, n no está restringido, $n \in \mathbb{Z}^{*+}$.

[c] En RDF(S) y OWL-Lite sólo puede ser usada con identificadores de clases y restricciones de propiedades. En OWL-DL, se extiende a cualquier expresión que describa clases.

[d] Los constructores DL asociados han sido introducidos en esta tesis, ver capítulo 3.

Tabla A.1: Primitivas principales de los lenguajes ontológicos OWL Lite, OWL DL y RDFS y su representación en la notación DL (téngase en cuenta las relaciones de inclusión entre estos lenguajes).

Apéndice B

Algoritmo de mezcla

Algoritmo B.1 Mezcla de nodos de un grafo de completamiento

Entradas: $G = (V, E, L, \dot{\neq})$, y, x
{ G, grafo de completamiento;
$x, y \in V$. }
Salidas: G
{G, transformado al realizar la mezcla del nodo y en x.}

Primera Parte: Revisión de los arcos de entrada a y.
for all z, $\langle z,y\rangle \in E$ **do**
  if $\{\langle x,z\rangle, \langle z,x\rangle\} \cap E = \emptyset$ **then**
    $E = E \cup \{\langle z,x\rangle\}$, $L(\langle z,x\rangle) = L(\langle z,y\rangle)$
  else if $\langle z,x\rangle \in E$ **then**
    $L(\langle z,x\rangle) = L(\langle z,x\rangle) \cup L(\langle z,y\rangle)$
  else if $\langle x,z\rangle \in E$ **then**
    $L(\langle x,z\rangle) = L(\langle x,z\rangle) \cup \{R^- | R \in L(\langle z,y\rangle)\}$
  $E = E \setminus \{\langle z,y\rangle\}$

Segunda Parte: Revisión de los arcos de salida de y.
for all z, $\langle y,z\rangle \in E$ **do**
  if $\{\langle x,z\rangle, \langle z,x\rangle\} \cap E = \emptyset$ **then**
    $E = E \cup \{\langle x,z\rangle\}, L(\langle x,z\rangle) = L(\langle y,z\rangle)$
  else if $\langle x,z\rangle \in E$ **then**
    $L(\langle x,z\rangle) = L(\langle x,z\rangle) \cup L(\langle y,z\rangle)$
  else if $\langle x,z\rangle \in E$ **then**
    $L(\langle z,x\rangle) = L(\langle z,x\rangle) \cup \{R^- | R \in L(\langle y,z\rangle)\}$
  $E = E \setminus \{\langle y,z\rangle\}$

Tercera Parte: Eliminación de y.
$L(x) = L(x) \cup L(y)$
Hacer $x \dot{\neq} z$ para todos z tal que $y \dot{\neq} z$
Prune(y)

function *Prune*(y): {y es un nodo}
for all z, sucesores de y **do**
  $E = E \setminus \{\langle y,z\rangle\}$
  if z es bloqueable **then**
    Prune(z)
$V = V \setminus \{y\}$

Bibliografía

ABELLÓ, A. (2002). *YAM^2: A Multidimensional Conceptual Model*. Ph.D. thesis, Universitat Politècnica de Catalunya. 32

ABELLÓ, A., SAMOS, J. & SALTOR, F. (2001). A data warehouse multidimensional data models classification. 31

ABNEY, S. (1991). Parsing by chunks. In R. Berwick, S. Abney & C. Tenny, eds., *Principle-Based Parsing: Computation and Psycholinguistics*, 257–278, Kluwer. 12

AGRAWAL, R., GUPTA, A. & SARAWAGI, S. (1997). Modeling multidimensional databases. In A. Gray & P.Å. Larson, eds., *Proc. 13th Int. Conf. Data Engineering, ICDE*, 232–243, IEEE Computer Society. 32

ALANI, H., KIM, S., MILLARD, D.E., WEAL, M.J., HALL, W., LEWIS, P.H. & SHADBOLT, N. (2003). Automatic ontology-based knowledge extraction from web documents. *IEEE Intelligent Systems*, **18**, 14–21. 71, 73, 104

AMARDEILH, F., LAUBLET, P. & MINEL, J.L. (2005). Document annotation and ontology population from linguistic extractions. In *K-CAP '05: Proceedings of the 3rd international conference on Knowledge capture*, 161–168, ACM Press. 68, 72, 73, 104

ANAYA-SÁNCHEZ, H., PONS-PORRATA, A. & BERLANGA, R. (2006). Word sense disambiguation based on word sense clustering. In *IBERAMIA-SBIA*, 472–481. 131, 132, 137

BAADER, F. & NUTT, W. (2003). *Basic description logics*, 43–95. Cambridge University Press, New York, NY, USA. 17, 18, 21

BAADER, F. & SATTLER, U. (2003). Description logics with aggregates and concrete domains. *Inf. Syst.*, **28**, 979–1004. 35

BAADER, F., KUSTERS, R. & WOLTER, F. (2003). *The description logic handbook: theory, implementation, and applications*, chap. Extensions to description logics, 219–261. Cambridge University Press. 21

BAEZA-YATES, R.A. & RIBEIRO-NETO, B.A. (1999). *Modern Information Retrieval*. ACM Press / Addison-Wesley. 67

BARBER, B. & HAMILTON, H.J. (1997). A comparison of attribute selection strategies for attribute-oriented generalization. In *ISMIS '97: Proceedings of the 10th International Symposium on Foundations of Intelligent Systems*, 106–116, Springer-Verlag, London, UK. 119

BAUMGARTNER, R., FLESCA, S. & GOTTLOB, G. (2001). Visual web information extraction with Lixto. In *The VLDB Journal*, 119–128. 68, 73

BAYERL, P.S., LUNGEN, H., GUT, U. & PAUL, K.I. (2003). Methodology for reliable schema development and evaluation of manual annotations. In *Workshop on Knowledge Markup and Semantic Annotation at the Second International Conference on Knowledge Capture*. 66

BERNERS-LEE, T. (1998). Semantic road map. http://www.w3.org/DesignIssues/Semantic.html. 23

BERNERS-LEE, T. (2000a). CWM - closed world machine. http://www.w3.org/2000/10/swap/ /doc/cwm.html. 28

BERNERS-LEE, T. (2000b). Semantic Web talk. http://www.w3.org/2000/Talks/1206-xml2k-tbl/slide0-0.html. 22

BINH, N.T. & TJOA, A.M. (2001). Conceptual multidimensional data model based on object-oriented metacube. In *SAC '01: Proceedings of the 2001 ACM symposium on Applied computing*, 295–300. 32

BOLEY, H., TABET, S. & WAGNER, G. (2001). Design rationale for RuleML: A markup language for Semantic Web rules. In *SWWS*, 381–401. 28

BORGIDA, A. & WEDDELL, G.E. (1997). Adding functional dependencies to description logics. In *Proc. of the 5th Int. Conf. on Deductive and Object-Oriented Databases*. 35

BOUQUET, P., GIUNCHIGLIA, F., HARMELEN, F.V., SERAFINI, L. & STUCKENSCHMIDT, H. (2003). C-OWL: Contextualizing ontologies. In *Lecture Notes in Computer Science 2870*, 164–179. 26

BRACHMAN, R.J. & LEVESQUE, H.J. (1985). *Readings in Knowledge Representation*. Morgan Kaufmann. 15

BRAY, T., HOLLANDER, D. & LAYMAN, A. (1999). Namespace in XML. W3C recommendation. http://www.w3.org/TR/REC-xml-names/. 23

BRAY, T., PAOLI, J. & SPERBERG-MCQUEEN, C. (2000). Extensible markup language XML 1.0. W3C recommendation. http://www.w3.org/TR/REC-xml. 23

BREIMAN, L., FRIEDMAN, J., OLSHEN, R. & STONE., C. (1984). *Classification and Regression Trees*. Wadsworth, Belmont, CA. 119

BRICKLEY, D. & GUHA, R. (2003). RDF vocabulary description language 1.0: RDF schema. W3C working draft. http://www.w3.org/TR/2003/WD-rdf-schems-20031010. 24

BRIN, S. (1998). Extracting patterns and relations from the world wide web. In *WebDB Workshop at 6th International Conference on Extending Database Technology, (1998)*. 66

BUITELAAR, P. & RAMAKA, S. (2005). Unsupervised ontology-based semantic tagging for knowledge markup. In W. Buntine, A. Hotho & S. Bloehdorn, eds., *Proc. Of the Workshop on Learning in Web Search at the International Conference on Machine Learning*. 67, 68, 72, 73, 104

BUNTINE, W. & NIBLETT, T. (1992). A further comparison of splitting rules for decision-tree induction. *Mach. Learn.*, **8**. 119

BUZYDLOWSKI, J.W., SONG, I.Y. & HASSELL, L. (1998). A framework for object-oriented online analytic processing. In *DOLAP '98: Proceedings of the 1st ACM international workshop on Data warehousing and OLAP*, 10–15. 32

CABIBBO, L. & TORLONE, R. (1997). Querying multidimensional databases. In S. Cluet & R. Hull, eds., *Database Programming Languages, 6th International Workshop, DBPL-6*, Lecture Notes in Computer Science 1369, 319–335, Springer. 32

CABIBBO, L. & TORLONE, R. (1998a). From a procedural to a visual query language for OLAP. In M. Rafanelli & M. Jarke, eds., *10th International Conference on Scientific and Statistical Database Management*, 74–83, IEEE Computer Society. 32

CABIBBO, L. & TORLONE, R. (1998b). A logical approach to multidimensional databases. *Lecture Notes in Computer Science 1377*, 183–197. 32

CAI, Y., CERCONE, N. & HAN, J. (1991). Attribute-oriented induction in relational database. *In Knowledge Discovery in Databases. AAAI/MIT Press, Cambridge, MA, 1991*, 213–228. 30, 113

CALVANESE, D., LENZERINI, M. & NARDI, D. (1998). Description logics for conceptual data modeling. In *Logics for Databases and Information Systems*, 229–263. 36

CARTER, C.L. & HAMILTON, H.J. (1998). Efficient attribute-oriented generalization for knowledge discovery from large databases. *IEEE Transactions on Knowledge and Data Engineering*, **10**, 193–208. 113

CELJUSKA, D. & VARGAS-VERA, M. (2004). Semi-automatic population of ontologies from text. In J. Paralic & A. Rauber, eds., *Workshop on Data Analysis WDA-2004*. 70, 71, 73, 104

CHINCHOR, N.A. (2001). Overview of muc-7/met-2. http://www.itl.nist.gov/iaui/894.02/related_projects/muc/proceedings/muc_7_toc.html. 13

CIMIANO, P., HANDSCHUH, S. & STAAB, S. (2004). Towards the self-annotating web. In *Proceedings of the 13th World Wide Web Conference*. 69, 73, 104

CIRAVEGNA, F., DINGLI, A., PETRELLI, D. & WILKS, Y. (2002). User-system cooperation in document annotation based on information extraction. In *EKAW '02: Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web*, 122–137, Springer-Verlag, London, UK. 70, 73, 77, 104

CIRAVEGNA, F., CHAPMAN, S., DINGLI, A. & WILKS, Y. (2004). Learning to harvest information for the Semantic Web. In *ESWS*, 312–326. 70, 73, 104

CODD, E.F., CODD, S.B. & SALLEY, C.T. (1993). Providing OLAP (On-Line Analytical Processing) to user-analysts: An IT mandate. Tech. rep., E.F. Codd Associates. 30

CONEN, W. & KLAPSING, R. (2000). A logical interpretation of RDF. *Electronic Transactions on Artificial Intelligence*, **5**. 25

COVER, T.M. & THOMAS, J.A. (1991). *Elements of Information Theory*. New York: Wiley. 130

DAIC (1956). Dartmouth Artificial Intelligence Conference. http://www.livinginternet.com//i/ii_ai.htm. 10

DANGER, R. (2006). http://krono.act.uji.es. 6, 138

DANGER, R. & BERLANGA, R. (2001). Búsqueda de reglas de asociación en bases de datos y colecciones de textos. Tech. rep., Departamento de Lenguajes y Sistemas Informáticos, Universitat Jaume I. 146

DANGER, R. & BERLANGA, R. (2005). Inferencia de cubos multidimensionales para grandes colecciones de documentos. In *Proceedings of CIARP*. 146

DANGER, R. & BERLANGA, R. (2006). Inferring multidimensional cubes for representing conceptual document spaces. In *Lecture Notes in Computer Science 4177*, 280–290. 146

DANGER, R., RUIZ-SHULCLOPER, J. & BERLANGA, R. (2003). Text mining using the hierarchical syntactical structure of documents. In *CAEPIA*, 556–565. 130, 146

DANGER, R., BERLANGA, R. & RUÍZ-SHULCLOPER, J. (2004a). CRISOL: An approach for automatically populating a Semantic Web from unstructured text collections. *Database and Expert Systems, Lecture Notes in Computer Science 3180*, 243–253. 145

DANGER, R., RUIZ-SHULCLOPER, J. & BERLANGA, R. (2004b). Objectminer: A new approach for mining complex objects. In *ICEIS (2)*, 42–47. 145, 146

DANGER, R., SANZ, I., BERLANGA, R. & RUÍZ-SHULCLOPER, J. (2004c). A proposal for the automatic generation of instances from unstructured text. In E. Springer-Verlag, ed., *CIARP 2004, Lecture Notes in Computer Science 3287*, 462–469. 123, 145, 146

DECKER, S., BRICKER, D., SAARELA, J. & ANGELE, L. (1998). A query and inference service for rdf. In *QL98 - Query Lnguages Workshop*. 25

DECKER, S., ERDMANN, M., FENSEL, D. & STUDER, R. (1999). Ontobroker: Ontology based access to distributed and semi-structured unformation. In *DS-8:Semantic Issues in Multimedia Systems*, 351–369, Kluwer Academic. 27

DILL, S., EIRON, N., GIBSON, D., GRUHL, D., GUHA, R., JHINGRAN, A., KANUNGO, T., MCCURLEY, K.S., RAJAGOPALAN, S., TOMKINS, A., TOMLIN, J.A. & ZIEN, J.Y. (2003). A case for automated large-scale semantic annotation. *J. Web Sem.*, **1**, 115–132. 69, 72, 73, 104

ETZIONI, O., CAFARELLA, M., DOWNEY, D., POPESCU, A.M., SHAKED, T., SODERLAND, S., WELD, D.S. & YATES, A. (2005). Unsupervised named-entity extraction from the web: an experimental study. *Artif. Intell.*, **165**, 91–134. 70

FAYYAD, U., PIATETSKY-SHAPIRO, G. & SMYTH, P. (1996). Knowledge discovery and data mining: Towards a unifying framework. In *Knowledge Discovery and Data Mining*, 82–88. 9

FELLBAUM, C. (1998). *WordNet: An Electronic Lexical Database*. The MIT Press. 127

FELLER, W. (1971). *An Introduction to Probability Theory and Its Applications*, vol. 2. New York: Wiley. 130

FIKES, R. & MCGUINNESS, D.L. (2001). An axiomatic semantics for RDF, RDF Schema, and DAML+OIL. Ksl technical report, Stanford University. 25

FOSTER, I. & KESSELMAN, C. (1998). *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann. 34

FRANCONI, E. & KAMBLE, A. (2004). The GMD data model and algebra for multidimensional information. In *In Proc. of the 16th International Conference on Advanced Information Systems Engineering (CAiSE-04)*. 31

FRANCONI, E. & SATTLER, U. (1999). A data warehouse conceptual data model for multidimensional aggregation. In *DMDW*, 13. 31

FSDM (2005). First proceedings of workshop on feature selection for data mining. http://enpub.eas.asu.edu/workshop/FSDM05-Proceedings.pdf. 119

FSDM (2006). Second proceedings of workshop on feature selection for data mining. http://enpub.eas.asu.edu/workshop/2006/FSDM06-Proceedings.pdf. 119

GARCIA, A., BERLANGA, R. & DANGER, R. (2006). Mining of complex objects via description clustering. In *ICSOFT*, 187–194. 145, 146

GOLFARELLI, M., RIZZI, S. & VRDOLJAK, B. (2001). Data warehouse design from XML sources. In *Proceedings of the 4th ACM international conference on Data warehousing and OLAP*, 40–47, ACM Press. 34

GOMEZ-PEREZ, A., FERNÁNDEZ-LÓPEZ, M. & CORCHO, O. (2003). *Ontological Engeneering, whith examples from areas of Knowledge Management, e-commerce and Semantic Web*. Springer-Verlag. 18

GRAU, B.C. (2006). OWL 1.1 Web Ontology Language model-theoretic semantics. http://owl1-1.cs.manchester.ac.uk/old/Semantics.html. 39

GRISHMAN, R. (1997). Information extraction: Techniques and challenges. In *SCIE '97: International Summer School on Information Extraction*, 10–27. 77

GROSOF, B., HORROCKS, I., VOLZ, R. & DECKER, S. (2003). Description logic programs: Combining logic programs with description logics. In *Proc. of WWW-2003*, Budapest, Hungary. 28

GUHA, R. & HAYES, P. (2003). Lbase: Semantics for languages of the Semantic Web. W3C note. http://www.w3.org/TR/2003/Note-lbase-20031010. 25

GUYON, I., GUNN, S.R., BEN-HUR, A. & DROR, G. (2004). Result analysis of the NIPS 2003 feature selection challenge. In *NIPS*. 119

GYSSENS, M. & LAKSHMANAN, L.V.S. (1997). A foundation for multi-dimensional databases. In *The VLDB Journal*, 106–115. 31

HAARSLEV, V. & MOLLER, R. (2001). Description of the RACER system and its applications. In *Description Logics*, Workshop in Description Logics 2001 (DL2001), Stanford, USA. 27, 61

HACID, M.S. & SATTLER, U. (1997). An object-centered multi-dimensional data model with hierarchically structured dimensions. In *Proceedings of the IEEE Knowledge and Data Engineering Workshop*, 65–72, IEEE Computer Society. 34

HACID, M.S. & SATTLER, U. (1998). Modeling multidimensional databases: A formal object-centered approach. In *Proceedings of the Sixth European Conference on Information Systems*. 34

HAN, J. & FU, Y. (1996). Exploration of the power of attribute-oriented induction in data mining. In U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth & R. Uthurusamy, eds., *Advances in Knowledge Discovery and Data Mining*, AIII Press/MIT Press. 113

HAN, J. & KAMBER, M. (2001). *Data Mining: Concepts and Techniques*. Morgan Kaufmann. 120, 123

HAN, J., CAI, Y. & CERCONE, N. (1993). Data-driven discovery of quantitative rules in relational databases. *IEEE Trans. Knowl. Data Eng.*, **5**, 29–40. 113, 123

HAN, J., NISHIO, S., KAWANO, H. & WANG, W. (1998). Generalization-based data mining in object-oriented databases using an object cube model. *Data Knowledge Engineering*, **25**, 55–97. 113, 123

HANDSCHUH, S. & STAAB, S. (2002). Authoring and annotation of web pages in CREAM. In *The Eleventh International World Wide Web Conference (WWW2002)*, 462–473. 70, 71, 73

HANDSCHUH, S., STAAB, S. & MAEDCHE, A. (2001). CREAM - creating relational metadata with a component-based, ontology-driven annotation framework. In *First International Conference on Knowledge Capture (K-CAP')*. 70, 71, 73

HANDSCHUH, S., STAAB, S. & STUDER, R. (2003). Leveraging metadata creation for the Semantic Web with CREAM. In *Proc. of the Annual German Conference on AI*, vol. 2821, 19–33, Springer. 70, 71

HAYES, P. (2003). RDF semantics. W3C working draft. http://www.w3.org/TR/2003/rdf-mt/. 24

HAYES, P.J. (1980). The logic of frames. In D. Metzing, ed., *Frame Conceptions and Text Understanding*, 46–61, de Gruyter. 15

HEFFLIN, J., VOLZ, R. & DALE, J. (2002). Requirements for a web ontology language. W3C technical report. 26

HORROCKS, I. & PATEL-SCHNEIDER, P. (2003). Reducing OWL entailment to description logic satisfiability. In *Proc. of the 2nd International Semantic Web Conference (ISWC)*. 26

HORROCKS, I. & PATEL-SCHNEIDER, P.F. (1998). FaCT and DLP: Automated reasoning with analytic tableaux and related methods. In *Proc. of TABLEAUX'98*. 61

HORROCKS, I. & SATTLER, U. (2001). Ontology reasoning in the SHOQ(D) description logic. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*. 25

HORROCKS, I. & SATTLER, U. (2004). Decidability of SHIQ with complex role inclusion axioms. *Artif. Intell*, **160**, 79–104. 21

HORROCKS, I. & SATTLER, U. (2005a). A tableaux decision procedure for SHOIQ. In *Proceedings of Nineteenth Inter- national Joint Conference on Artificial Intelligence (IJCAI 2005)*. 39, 42, 43, 61, 63

HORROCKS, I. & SATTLER, U. (2005b). A tableaux decision procedure for SHOIQ. Tech. rep., University of Manchester. 39, 42, 43, 53, 61

HORROCKS, I., SATTLER, U. & TOBIES, S. (1999). Practical reasoning for expressive description logics. In H. Ganzinger, D. McAllester & A. Voronkov, eds., *Proceedings of the 6th International Conference on Logic for Programming and Automated Reasoning (LPAR'99)*, 1705, 161–180, Springer-Verlag. 27

HORROCKS, I., PATEL-SCHIENEIDER, P., BOLEY, H. & TABET, S. (2003a). A proposal for an OWL rules languages. draft version. http://www.daml.org/rules/proposal/OWL-rules-20031027/. 28

HORROCKS, I., PATEL-SCHNEIDER, P. & HARMELEN, F.V. (2003b). From SHIQ and RDF to OWL: The making of a web ontology language. *Journal of Web Semantics*, **1**, 7–26. 26

HORROCKS, I., KUTZ, O. & SATTLER, U. (2006). The even more irresistible SROIQ. In *In Proceedings of the 10th International Conference of Knowledge Representation and Reasoning.*. 21

HÜMMER, W., BAUER, A. & HARDE, G. (2003). XCube - XML for data warehouses. In *Proceedings of the 6th ACM intl. workshop on Data warehousing and OLAP*, 33–40, ACM Press. 34

ISMAILOVA, L., KOSIKOV, S., ZINCHENKO, K., MIKHAYLOV, A., BOURMISTROVA, L. & BEREZOVSKAYA, A. (2000). Building views with description logics in ADE: Application development environment. In *The 2-nd Intl. Workshop on Computer Science and Information Technologies*, vol. 1, 153. 35, 36

JENSEN, M.R., MØLLER, T.H. & PEDERSEN, T.B. (2001). Specifying OLAP cubes on XML data. *Journal of Intelligent Information Systems*, **17**, 255 – 280. 34

JOHNSON, R.A. & WICHERN, D.W., eds. (1988). *Applied multivariate statistical analysis*. Prentice-Hall. 119

JURAFSKY, D. & MARTIN, J. (2000). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall. 12

KALYANPUR, A., HENDLER, J., PARSIA, B. & GOLBECK, J. (2002). SMORE - semantic markup, ontology, and RDF editor. Tech. rep., University of Maryland. 71

KLYNE, G. & CARROL, J. (2003). Resourse description framework (RDF): Concepts and abstract data model. W3C working draft. http://www.w3.org/TR/rdf-concepts/. 24

KOGUT, P. & HOLMES, W. (2001). AeroDAML: applying information extraction to generate DAML annotations from web pages. In *Proceedings of the Workshop on Knowledge Markup and Semantic Annotation at 1st International Conference on Knowledge Capture*. 68, 73

KOLLER, D. & SAHAMI, M. (1997). Hierarchically classifying documents using very few words. In *Proc. of the 14th Conf. on Machine Learning ICML'97*, 143–151. 125

KULLBACK, S. & LEIBLER, R.A. (1951). *On information and sufficiency*. Annals of Mathematical Statistics, 22. 119

LANGLEY, P. (1994). Selection of relevant features in machine learning. *AAAI Fall Symposium on Relevance*, 140–144. 119

LEE, J., GROSSMAN, D. & ORLANDIC, R. (2002). MIRE: A multidimensional information retrieval engine for structured data and text. In *Proceedings of the International Conference on Information Technology: Coding and Computing*, 224–229, IEEE Computer Society. 34

LEE, J., GROSSMAN, D. & ORLANDIC, R. (2003). An evaluation of the incorporation of a semantic network into a multidimensional retrieval engine. In *Proceedings of the 12th international conference on Information and knowledge management*, 572–575, ACM Press. 34

LEE, T.B., HENDLER, J. & LASSILA, O. (2001). The Semantic Web. *Scientific American*, **284**, 34–43. 11

LI, C. & WANG, X.S. (1996). A data model for supporting on-line analytical processing. In *CIKM*, 81–88. 31

MAEDCHE, A., NEUMANN, G. & STAAB, S. (2002). Bootstrapping an ontology-based information extraction system. In P. Szczepaniak, J. Segovia, J. Kacprzyk & L. Zadeh, eds., *Intelligent Exploration of the Web*, Springer / Physica Verlag, Heidelberg. 67, 73

MANGISENGI, O., HUBER, J., HAWEL, C. & ESSMAYR, W. (2001). A framework for supporting interoperability of data warehouse islands using XML. In *Proceedings of the 3rd International Conference on Data Warehousing and Knowledge Discovery*, 328–338, Springer, Berlin. 34

MANNING, C.D. & SCHÜTZE, H. (1999). *Foundations of Statistical Natural Language Processing*. The MIT Press. 13

MAYNARD, D., YANKOVA, M., ASWANI, N. & CUNNINGHAM, H. (2004). Automatic creation and monitoring of semantic metadata in a dynamic knowledge portal. *Lecture Notes in Computer Science 3192*, 65 – 74. 68, 73, 74, 104

MCCABE, M.C., LEE, J., CHOWDHURY, A., GROSSMAN, D. & FRIEDER, O. (2000). On the design and evaluation of a multi-dimensional approach to information retrieval. In *Proceedings*

of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval, 363–365, ACM Press. 34

MCCARTHY, J. (2004). What is artificial intelligence? http://www-formal.stanford.edu/jmc/ /whatisai/whatisai.html. 10

MOLINA, L.C., BELANCHE, L. & NEBOT, A. (2002). Feature selection algorithms: A survey and experimental evaluation. In *Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM'02)*, 306. 119

MONTOYO, A. & PALOMAR, M. (2001). *Specification Marks for Word Sense Disambiguation: New Development*, 182–191. Lecture Notes in Computer Science. 132

MOODY, D.L. & KORTINK, M.A.R. (2000). From enterprise models to dimensional models: A methodology for data warehouse and data mart design. In *Proc. of 2nd Int. Workshop on Design and Management of Data Warehouses*. 32

NARDI, D. & BRACHMAN, R.J. (2002). An introduction to description logics. In *The Description Logic Handbook*, 5–44, Cambridge University Press. 15

NETWORKINFERENCEINC. (2003). Cerebra server datasheet. http://cerebra.com/downloads/ /datasheet_cerebra.pdf. 27

NEUMANN, G., BACKOFEN, R., BAUR, J., BECKER, M. & BRAUN, C. (1997). An information extraction core system for real world german text processing. In *ANLP*, 209–216. 67

NGUYEN, T.B., TJOA, A.M. & WAGNER, R. (2000). An object oriented multidimensional data model for OLAP. In *Web-Age Information Management*, 69–82. 32

NGUYEN, T.B., TJOA, A.M. & MANGISENGI, O. (2001). Meta Cube-X: An XML metadata foundation of interoperability search among web data warehouses. In *Proceedings of the Third International Workshop on Design and Management of Data Warehouses*, 8.1–8.8, CEUR-WS.org. 34

PEDERSEN, D., RIIS, K. & PEDERSEN, T.B. (2002). XML-Extended OLAP querying. In *Proceedings of the 14th International Conference on Scientific and Statistical Database Management*, 195–206, IEEE Computer Society. 34

PEDERSEN, D., PEDERSEN, J. & PEDERSEN, T.B. (2004). Integrating XML data in the TARGIT OLAP system. In *Proceedings of the 20th International Conference on Data Engineering*, 778–781, IEEE Computer Society. 34

PEDERSEN, T. (2000). *Aspects of Data Modeling and Query Processing for Complex Multidimensional Data*. Ph.D. thesis. 32

PEDERSEN, T.B. & JENSEN, C.S. (1998). Research issues in clinical data warehousing. In *Statistical and Scientific Database Management*, 43–52. 32

PEDERSEN, T.B. & JENSEN, C.S. (1999). Multidimensional data modeling for complex data. In *ICDE*, 336–345. 32

PÉREZ, J.M., BERLANGA, R., ARAMBURU, M.J. & PEDERSEN, T.B. (2005). A relevance-extended multi-dimensional model for a data warehouse contextualized with documents. In *DOLAP '05: Proceedings of the 8th ACM international workshop on Data warehousing and OLAP*. 139

POKORN, J. (2001). Modelling stars using xml. In *Proceedings of the 4th ACM international conference on Data warehousing and OLAP*, 24–31, ACM Press. 34

PONS, A. (2004). *Desarrollo de algoritmos para la estructuración dinámica de información y su aplicación a la detección de sucesos*. Ph.D. thesis, Depto. Lenguajes y Sistemas Informáticos, Universitat Jaume I. 126, 131

POPOV, B., KIRYAKOV, A., OGNYANOFF, D., MANOV, D., KIRILOV, A. & GORANOV, M. (2003). Towards Semantic Web information extraction. *Human Language Technologies Workshop at the 2nd International Semantic Web Conference (ISWC2003), Florida, USA*. 67, 73

RAMAKRISHNAN, R., SRIVASTAVA, D., SUDARSHAN, S. & SESHADRI, P. (1994). The CORAL deductive system. *VLDB Journal: Very Large Data Bases*, **3**, 161–210. 27

REEVE, L. & HAN, H. (2005). Survey of semantic annotation platforms. In *SAC*, 1634–1638. 66

ROO, J.D. (2002). Euler proof mechanism. http://www.agfa.com/w3c/euler/. 25, 28

SAGONAS, K., SWIFT, T. & WARREN, D. (1994). XSB as an efficient deductive database engine. In *Proc. of the 1994 ACM SIGMOD Int. Conf. on Management of Data (SIGMOD'94)*, 442–453, ACM Press. 27

SALTON, G. (1989). *Automatic Text Processing*. Addison-Wesley. 125, 127

SCHMIDT-SCHAUSS, M. & SMOLKA, G. (1991). Attributive concept descriptions with complements. *Artificial Intelligence*, **48**, 1–26. 18

SHESKIN, G.J. (1997). *Handbook of parametric and nonparametric statistical procedures*. Boca Raton, Florida: CRC Press. 130

SIRIN, E. (2003). Pellet owl reasoner. http://www.mindswap.org/2003/pellet/index.shtml. 61

SIRIN, E., PARSIA, B., CUENCA-GRAU, B., KALYANPUR, A. & KATZ, Y. (2006). Pellet: A practical OWL-DL reasoner. *Journal of Web Semantics*. 39

SRINIVASAN, U., PFEIFFER, S., NEPAL, S., LEE, M., GU, L. & BARRASS, S. (2005). A survey of MPEG-1 audio, video and semantic analysis techniques. *Multimedia Tools Appl.*, **27**, 105–141. 38

SURDEANU, M., HARABAGIU, S., WILLIAMS, J. & AARSETH, P. (2003). Using predicate-argument structures for information extraction. In *Proceedings of the ACL, Sapporo, Japan*. 72, 73, 104

SUSSNA, M. (1993). Word sense disambiguation for free-text indexing using a massive semantic network. *Second International Conference on Information and Knowledge Management*. 132

THOMPSON, H., BEECH, D., MALONEY, M. & MENDELSOHN, N. (2001). XML schema part1: Structures. W3C recommendation. http://www.w3.org/TR/xml-schema-1/. 23

TOBIES, S. (2000). The complexity of reasoning with cardinality restrictions and nominals in expressive description logics. In *Journal of Artificial Intelligence Research*, 12, 199–217. 42, 61

TRUJILLO, J. & PALOMAR, M. (1998). An object-oriented approach to multidimensional database conceptual modeling (OOMD). In *DOLAP '98, ACM First International Workshop on Data Warehousing and OLAP*, 16–21, ACM. 32

TRUJILLO, J., PALOMAR, M. & GÓMEZ, J. (2000). Applying object-oriented conceptual modeling techniques to the design of multidimensional databases and OLAP applications. In *WAIM '00: Proceedings of the First International Conference on Web-Age Information Management*, 83–94. 32

TRUJILLO, J., PALOMAR, M., GÓMEZ, J. & SONG, I.Y. (2001). Designing data warehouses with OO conceptual models. *Computer*, **34**, 66–75. 32

TRUJILLO, J., LUJÁN-MORA, S. & SONG, I. (2004). Applying UML and XML for designing and interchanging information for data warehouses and OLAP applications. *Journal of Database Management*, **14**, 41 – 72. 34

TRYFONA, N., BUSBORG, F. & CHRISTIANSEN, J.G.B. (1999). starER: A conceptual model for data warehouse design. In *International Workshop on Data Warehousing and OLAP*, 3–8. 32

TSENG, F. & CHEN, C. (2005). Integrating heterogeneous data warehouses using XML technologies. *Journal of Information Science*, **31**, 209 – 229. 34

UNICODE (1991-2006). Unicode standard. http://www.unicode.org/standard/standard.html. 23

VALARAKOS, A.G., PALIOURAS, G., KARKALETSIS, V. & VOUROS, G.A. (2004). Enhancing ontological knowledge through ontology population and enrichment. In *Engineering Knowledge in the Age of the Semantic Web, EKAW*, 144–156. 69, 73, 104

VAN ZYL, J., VINCENT, M. & MOHANIA, M. (1998). Representation of metadata in a data warehouse. In *IEEE Region 10 International Conference on Global Connectivity in Energy, Computer, Communication and Control*, vol. 1, 103–106. 35

VARGAS-VERA, M., MOTTA, E. & DOMINGUE, J. (2001). Knowledge extraction by using an ontology-based annotation tool. In *Workshop on Knowledge Markup & Semantic Annotation*. 70, 71, 73, 104

VARGAS-VERA, M., MOTTA, E., DOMINGUE, J., LANZONI, M., STUTT, A. & CIRAVEGNA, F. (2002). MnM: Ontology driven semi-automatic and automatic support for semantic markup. In *EKAW*, 379–391. 70, 71, 73, 104

VOLZ, R. (2004). *Web Ontology Reasoning with Logic Databases*. Ph.D. thesis, AIFB, Karlsruhe. 24, 25, 26, 27

VOORHEES, E.M. (1993). Using WordNet to disambiguate word sense for text retrieval. *Proceeding of ACM SIGIR*, **16**, 171–180. 132

VYSNIAUSKAS, E. & NEMURAITE, L. (2006). Transforming ontology representation from OWL to relational database. *Information Technology And Control*, **Vol. 35A**, 333 – 343. 38

W3C (2002). RDF model theory. http://www.w3.org/TR/2002/WD-rdf-mt-20020429/. 24

W3C (2004). OWL Web Ontology Language overview. http://www.w3.org/TR/owl-features/. 25

W3C (2006). OWL 1.1 Web Ontology Language. http://owl1_1.cs.manchester.ac.uk/. 39

WANG, Y. & HODGES, J. (2006). Document clustering with semantic analysis. In *HICSS '06: Proceedings of the 39th Annual Hawaii International Conference on System Sciences*. 38

WHOWEDA (1997). The web warehousing & mining group. http://www.cais.ntu.edu.sg:8000/~whoweda. 33

WINOGRAD, T. (1972). *Understanding Natural Language*. New York: Academic Press. 11

XYLEME, L. (2001). A dynamic warehouse for XML data of the web. *IEEE Data Engineering Bulletin*, **24**, 40 – 47. 33

YANG, J. & CHOI, J. (2003). Agents for intelligent information extraction by using domain knowledge and token-based morphological patterns. In *PRIMA*, 74–85. 69, 70, 73, 104