Población de un almacén de datos con datos enriquecidos semánticamente.

V. Nebot y R. Berlanga*

Resumen— **Debido al auge de la Web Semántica cada vez existen más repositorios con grandes cantidades de datos semánticos (datos anotados con elementos de ontologías), con una estructura compleja, semi-estructurada y heterogénea. En este artículo se presenta un método para la construcción automática de tablas de hechos a partir de datos enriquecidos semánticamente y almacenados en forma de instancias RDF/OWL. Como punto de partida, el método toma un esquema multidimensional en forma de estrella (el tema de análisis, las dimensiones y medidas) diseñado por el analista a partir de los conceptos y propiedades de la ontología que describe el almacén de instancias. El método explota la semántica encapsulada en los axiomas de la ontología para derivar "combinaciones válidas" de instancias y literales y así construir hechos. Además, tanto la ontología como las instancias se indexan con unos índices específicos que proporcionan escalabilidad a la propuesta. El método ha sido evaluado con un conjunto de instancias OWL generadas de forma sintética.**

***Palabras clave*— Almacenes de datos, Procesos ETL, Web Semántica. (*Data Warehousing, ETL Processes, Semantic Web*).**

I. Introducción

Las herramientas de análisis de datos como OLAP (On-Line Analytical Processing) [1] permiten extraer información valiosa a partir de grandes bases de datos transaccionales. Estas herramientas se apoyan en las estructuras multidimensionales (MD) que componen un almacén de datos como los hechos y las jerarquías de las dimensiones, las cuales permiten al analista explorar y agregar la información con distinto nivel de detalle. Aunque el análisis OLAP ha ganado gran aceptación como herramienta de análisis en aplicaciones de negocio tradicionales, el aumento de repositorios que contienen datos con una estructura más rica y compleja que la típica estructura de datos relacional, sugiere el estudio y adaptación de las técnicas OLAP a este nuevo tipo de fuentes de datos [2], [3].

El continuo desarrollo de la Web Semántica nos proporciona cada vez más tecnologías y herramientas para la generación de datos anotados semánticamente. Hoy en día, muchas aplicaciones (p. ej. aplicaciones médicas) adjuntan metadatos y anotaciones semánticas a los datos que generan, (p. ej. radiografías, análisis de laboratorio, etc). Las anotaciones semánticas son descripciones formales acerca de recursos de información y se suelen basar en ontologías de dominio comúnmente aceptadas. La razón por la que se utilizan ontologías de dominio es para establecer una terminología y una semántica común para los conceptos de un dominio particular. Las anotaciones semánticas son especialmente útiles para describir datos sin estructura, semi-estructurados o texto, ya que este tipo de datos no suele ser gestionado de forma correcta por los sistemas de bases de datos actuales. En la actualidad, existen numerosos repositorios de datos enriquecidos semánticamente (p. ej. DBpedia [4], BioRDF [5], ...) los cuales nos brindan la oportunidad de mejorar los sistemas de soporte a la decisión que existen actualmente.

El objetivo principal de este artículo es el de proponer un método automático para la extracción de hechos derivados a partir de grandes cantidades de datos enriquecidos semánticamente, siguiendo estructura MD adecuada para la realización de análisis OLAP. Esta aproximación incluye tanto el diseño como la población del esquema MD. Nuestro trabajo previo [6] aborda la fase del diseño, para la cual se propone un método formal a través del cual el analista define el modelo MD identificando el tema de análisis, las medidas y las dimensiones de un conjunto de datos. Otras aproximaciones como [7] tratan de automatizar el diseño MD del almacén de datos a partir de una ontología de dominio. En cualquier caso, el método que presentamos en este artículo se centra en automatizar el proceso de combinación y población de la tabla de hechos a partir de instancias OWL[1] cuando el modelo MD ya se ha definido. La población de las jerarquías de las dimensiones se deja como trabajo futuro. Para abordar el problema de forma eficiente se han introducido índices tanto sobre la ontología que describe las instancias como sobre las propias instancias. Nuestro método puede ser considerado un proceso *ETL (Extract, Transform and Load)* cuyo objetivo es poblar un almacén de datos con datos semánticos, puesto que adaptamos la estructura de los datos al modelo MD en lugar de diseñar nuevos modelos de análisis para estos nuevos tipos de datos.

Esta propuesta conlleva nuevos retos con respecto a los procesos actuales de población de un almacén de datos. En primer lugar, no podemos asumir que las fuentes de datos están implementadas sobre una base de datos relacional, sino que hay que centrarse en las ontologías que describen y representan las fuentes de datos. Las ontologías son mucho

* Este trabajo ha sido subvencionado por el Plan Nacional de I+D+I, proyecto TIN2008-01825/TIN de España, y el proyecto europeo integrado Health-e-Child (IST 2004-027749).

V. Nebot y R. Berlanga trabajan en el Departamento de Lenguajes y Sistemas Informáticos de la Universitat Jaume I de Castellón, Campus Riu Sec, 12071 Castellón (correos e.: romerom@lsi.uji.es; berlanga@lsi.uji.es).

[1] OWL Web Ontology Language: http://www.w3.org/TR/owl-features/, 2004.

más expresivas y flexibles que un esquema relacional. De hecho, las ontologías no pueden considerarse un esquema, sino que son un conjunto de axiomas con los cuales las instancias han de ser consistentes. Por tanto, la definición de un hecho y el proceso de extracción de hechos estará guiado por el conocimiento expresado en la ontología. Hasta la fecha, sólo conocemos una propuesta similar [8], la cual también utiliza tecnologías de la Web Semántica para poblar cubos OLAP. En este trabajo, se utilizan *mappings* para convertir las fuentes de datos a RDF[2] y luego consultan los datos en RDF utilizando SparQL[3] para poblar el esquema OLAP. El inconveniente de esta propuesta es que los usuarios tienen que conocer el esquema de los datos para poder realizar las consultas RDF con SparQL, lo cual puede resultar tedioso si los datos tienen una estructura rica y heterogénea. A diferencia de la aproximación anterior, nuestro método extrae y combina de manera automática instancias para crear hechos válidos.

El resto del artículo se organiza de la siguiente forma: en la sección 2 se describe un escenario de aplicación que motiva nuestra propuesta. La sección III contiene los fundamentos. En la sección IV se dan los detalles acerca del método propuesto. La Sección V muestra la evaluación y en la Sección VI se dan algunas conclusiones y algunas líneas de trabajo futuro.

II. ESCENARIO DE APLICACIÓN

El escenario de aplicación elegido es el área de la Biomedicina, en la cual existen diversos sistemas de gestión de datos, como [9], capaces de generar grandes cantidades de datos anotados semánticamente. Para guiar el proceso de anotación, estos sistemas de gestión de datos adoptan ontologías de aplicación específicas que se apoyan en una o varias ontologías de dominio comúnmente aceptadas. Una ontología de dominio en el escenario biomédico suele estar compuesta por un gran corpus de conceptos relacionados que describen el vocabulario y el conocimiento acordado por la comunidad biomédica.

En este escenario, los datos anotados semánticamente pueden ser de naturaleza muy distinta y heterogénea (p. ej. informes sobre análisis de laboratorio, ecografías, radiografías, etc.) Estos datos presentan relaciones complejas que evolucionan de forma rápida debido a los continuos avances y descubrimientos que surgen casi diariamente. Como consecuencia, la tecnología de almacenes de datos actual no puede gestionar toda esta información, ya que es compleja, semi-estructurada, dinámica y altamente heterogénea.

La Fig. 1 muestra un fragmento de una ontología para el dominio de Reumatología usando lógica de descripción (*DL)* [10]. Las lógicas de descripción son formalismos para la representación del conocimiento con una semántica formal y bien definida. OWL está basado en DL pero en la figura se muestra la ontología en DL porque su sintaxis es bastante menos verbosa y más entendible que la sintaxis OWL. Los conceptos en negrita son conceptos externos, es decir, esos conceptos junto con todas sus subclases se toman de ontologías de dominio externas. Tal y como muestra la figura, un paciente tiene asociados una serie de datos personales además de varios informes (*Reports*), los cuales almacenan los resultados de los análisis de sangre, (definidos en la ontología GALEN[4]), los exámenes reumatológicos, el diagnóstico (definido en la ontología NCI[5]) y el tratamiento. El tratamiento consiste en un conjunto de medicamentos (definidos de acuerdo a la ontología UMLS[6]). El paciente también tiene un perfil genético. Los genes involucrados se describen utilizando la ontología GO[7].

$$Patient \sqsubseteq \exists\, hasAge.integer$$
$$Patient \sqsubseteq \exists\, sex.Gender$$
$$Patient \sqsubseteq \exists\, hasProfile.(\exists\, relatedGene.\mathbf{Gene})$$
$$Patient \sqsubseteq \exists\, hasReport.Report$$
$$Report \sqsubseteq \exists\, date.string$$
$$Report \sqsubseteq \exists\, hasDiagnosis.\mathbf{Disease}$$
$$Report \sqsubseteq \exists\, hasSection.Section$$
$$RE \sqsubseteq Section \sqcap \exists\, damageIndex.float$$
$$LBT \sqsubseteq Section \sqcap \exists\, measuresIndicant.\mathbf{BloodIndicant}$$
$$Treat \sqsubseteq Section \sqcap \exists\, hasTherapy.\mathbf{Drug}$$
$$\ldots$$

Fig. 1. Fragmento de una ontología de aplicación sobre Reumatología.

A través de las ontologías de dominio utilizadas en el fragmento de ontología anterior se pueden reusar conceptos y propiedades de dichas ontologías en distintos ámbitos además de servir para controlar el vocabulario y para aportar semántica a los datos anotados [11]. La Fig. 2 muestra la estructura en forma de grafo que subyace a algunas instancias OWL generadas de forma consistente con respecto a la ontología de aplicación de la Fig. 1. Este ejemplo se toma como referencia y se desarrollará durante el artículo. Tal y como puede observarse, las instancias y literales (los nodos) se relacionan entre ellos a través de propiedades (las flechas), formando una estructura compleja en forma de grafo. En este caso, la figura muestra una instancia de tipo *Patient* junto con sus características y relaciones con otras instancias y literales. Este ejemplo, aunque es sencillo, refleja la complejidad y heterogeneidad que las instancias pueden alcanzar, en comparación con las fuentes de datos relacionales de los almacenes de datos tradicionales.

Dado un gran repositorio de instancias semánticas como las de la Fig. 2, nuestro objetivo es construir un método automático que permita extraer y combinar sólo los valores de las instancias y literales relevantes para el modelo MD del analista (en la Fig. 2 estos valores son los nodos sombreados).

[2] RDF, Concepts and Abstract Syntax: http://www.w3.org/TR/rdf-concepts/, 2004.

[3] SparQL for RDF: http://www.w3.org/TR/rdf-sparql-query/, 2008

[4] GALEN: http://www.opengalen.org/

[5] NCI: http://www.nciterms.nci.nih.gov/

[6] UMLS: http://www.nlm.nih.gov/research/umls/

[7] GO: http://www.geneontology.org/

Para ello, se utilizará la semántica proporcionada por los axiomas de la ontología. Como resultado final se obtendrá un conjunto de hechos que poblarán una tabla de hechos de un almacén de datos, la cual podrá ser analizada más adelante utilizando herramientas OLAP típicas.

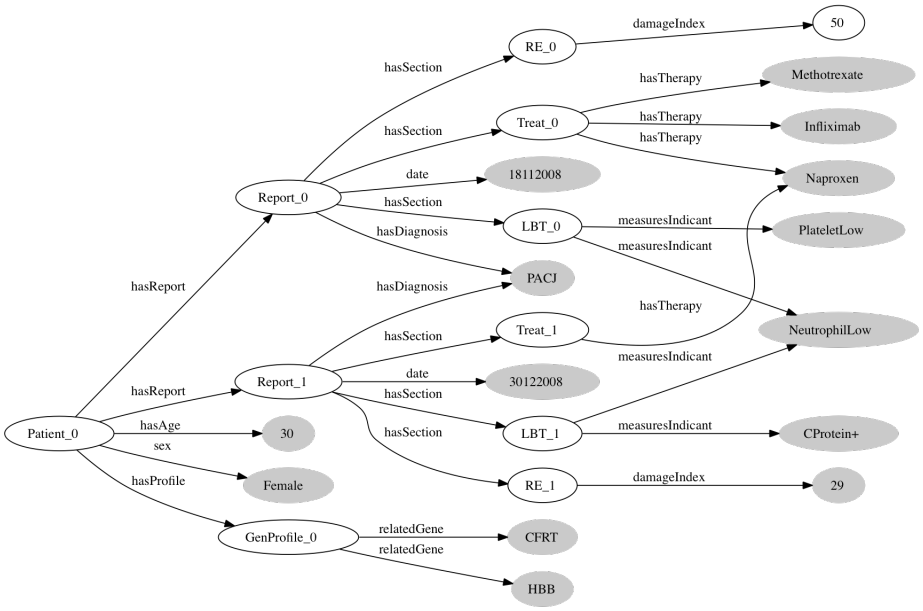

Fig. 2. Ejemplo de instancias OWL que son consistentes con los axiomas de la ontología de aplicación de la Fig. 1. Los nodos sombreados son las instancias y literales que se quiere combinar.

A. *Esquema multidimensional*

Para poder realizar una tarea de análisis, el analista ha de expresar el esquema MD que tiene en mente. En nuestro caso, al analista se le permite construir su modelo MD a partir de un subconjunto de conceptos y propiedades de la ontología que han sido preseleccionados manualmente. Como *tema de análisis* se ha de seleccionar un concepto de la ontología. A continuación se escogen las *dimensiones y medidas*. Una dimensión se define como un conjunto de niveles con una relación de orden parcial entre ellos, es decir, una jerarquía. Puesto que en este trabajo no se aborda el tema de las jerarquías de dimensión , el analista especificará sólo el nivel base de cada dimensión, que podrá ser un concepto o una propiedad. En el caso de escoger una propiedad, el analista debe especificar también el concepto dominio al cual se asocia dicha propiedad mediante la sintaxis *Dominio.propiedad*. Con respecto a las medidas, en este trabajo se tratan igual que las dimensiones, siguiendo la aproximación en [12]. Por tanto, cualquier medida se expresará igual que una dimensión. De ahora en adelante nos referiremos con el término *dimensiones* al conjunto de las dimensiones y medidas. La Fig. 3 muestra un posible esquema MD diseñado por el analista a partir de la Fig. 1. El analista está interesado en analizar la eficacia de los tratamientos prescritos a los pacientes durante el desarrollo de la enfermedad. Por tanto, como tema de análisis selecciona el concepto *Patient*. Como dimensiones ha especificado la enfermedad diagnosticada (concepto *Disease*), la edad y género del paciente (*Patient.hasAge* y *Gender,* respectivamente), la fecha del informe (*Report.date*), los medicamentos prescritos (*Drug*), los indicadores biomédicos (*Gene* y *BloodIndicant*) y el índice de daño articular (*RE.damageIndex*).

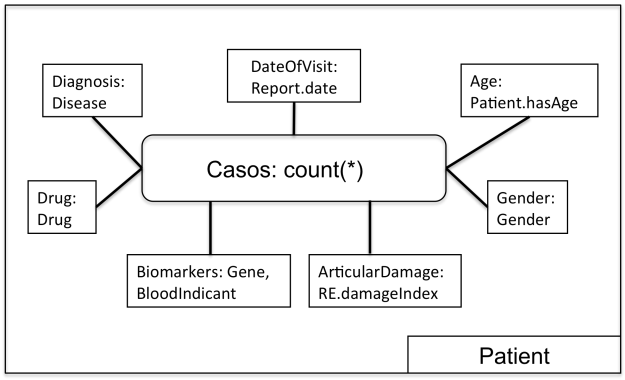

Fig. 3. Ejemplo de un esquema MD para el tema de análisis *Patient a partir de la ontología de la Fig. 1*.

III. FUNDAMENTOS

En esta sección se presentan los conceptos y definiciones que dan soporte al método desarrollado.

Definición 1 (Esquema Multidimensional). Se define un esquema multidimensional como una tupla $S=(F, D)$ donde F es el tema de análisis y $D=\{T_i\}_{i=1,...,m}$ son las dimensiones de análisis. Las dimensiones T_i son conceptos y propiedades escogidos de la ontología de aplicación.

La Fig. 3 muestra el esquema multidimensional diseñado por el analista compuesto por el tema de análisis y sus dimensiones. Al igual que en el diseño tradicional de almacenes de datos, las etiquetas de las fuentes de datos (p. ej. los nombres de atributos) suelen cambiar en la tabla de hechos. En nuestro caso, hemos adjuntado la etiqueta correspondiente junto a los elementos seleccionados de la ontología, por ejemplo, la etiqueta "Diagnosis" al concepto "Disease" de la ontología.

Definición 2 (Instancia Foco). Una instancia foco es una instancia consistente, asertada o inferida, perteneciente al tema de análisis.

En el ejemplo, *Patient* el es tema de análisis y sus instancias son las instancias foco.

La granularidad de la tabla de hechos se corresponde con el nivel más detallado de información que se almacenará en la misma. Ésta determina qué dimensiones se incluirán y a qué nivel a lo largo de la jerarquía de la dimensión se almacenará la información. En nuestro caso, queremos almacenar todas las dimensiones especificadas por el analista y consideramos la información encontrada en la ontología el nivel más detallado (o nivel base) en las jerarquías de las dimensiones. Por tanto, un hecho se define como sigue:

Definición 3 (Hecho). Se define un hecho como una *combinación válida* de valores de dimensión que co-ocurren en el contexto de una instancia foco.

Más adelante definiremos *combinación válida* pero por ahora digamos que no toda combinación posible de valores de dimensión que co-ocurren en una instancia foco es admisible. Por ejemplo, en un negocio de venta al por menor en donde las fuentes de datos son relacionales, las instancias foco típicas serían las *compras* y una compra puede ser descrita por un hecho que contiene los valores de dimensión de *lugar* de la compra, el tipo de *producto* y la *fecha* de la compra [12], los cuales aparecen junto de forma explícita en alguna tabla. Sin embargo, la estructura en forma de grafo de nuestras instancias foco restringe de forma implícita las combinaciones válidas con otras instancias y literales. Para encontrar de forma automática las combinaciones válidas de valores dentro de una instancia foco es necesario introducir algunas definiciones. A partir de ahora, las definiciones se aplican a una sola instancia foco, que además se asume que tiene estructura en forma de grafo acíclico directo.

Definición 4 (Contexto de dos dimensiones y dos valores de dimensión). Dadas dos dimensiones T_i y T_j se define el contexto de T_i y T_j, $context(T_i, T_j)$, como el concepto ancestro común más cercano de T_i y T_j inferido de la ontología a través de relaciones de composición entre propiedades. De forma similar, el contexto de dos valores de dimensión $v_i \in T_i$ y $v_j \in T_j$ es la instancia ancestro común más cercana de v_i y v_j en el repositorio de instancias.

Por ejemplo, el contexto de las dos dimensiones *Disease* y *RE.damageIndex* es *Report*, y el contexto de los valores de dimensión *PACJ* y *29* es *Report_1*.

Definición 5 (Dimensiones independientes / dependientes de contexto). Dadas dos dimensiones T_i y T_j, se consideran independientes si su contexto es el tema de análisis. En cualquier otro caso, las dimensiones son dependientes.

Por ejemplo, *Patient.hasAge* y *Gender* son dimensiones independientes (su contexto es *Patient*), mientras que *Disease* y *RE.damageIndex* son dependientes (su contexto es *Report*).

Definición 6 (Dependientes de contexto). Dados dos valores de dimensión v_i y v_j, $Type(v_i)=T_i$, $Type(v_j)=T_j$, $T_i \neq T_j$, $T_i, T_j \in D$ que co-ocurren en una instancia foco, v_i y v_j son dependientes de contexto si $Type(context(v_i, v_j)) = context(T_i, T_j)$.

Los valores de dimensión anteriores, *PACJ* y *29*, son dependientes de contexto porque su contexto es *Report_1*, cuyo tipo es *Report* y coincide con el contexto de sus respectivas dimensiones, *Disease* y *RE.damageIndex*.

Definición 7 (Combinación válida). Se define una combinación válida de valores de dimensión de una instancia foco como una tupla $(v_1, v_2, ..., v_n)$ donde $n=|D|$ y $\forall (i,j)$, (v_i, v_j) son dependientes de contexto.

Como ejemplo de la definición anterior, los valores de dimensión *30122008* y *CProtein+* son una combinación válida porque son dependientes de contexto. Sin embargo, los valores *30122008* y *Infliximab* no son una combinación válida porque no son dependientes de contexto, es decir, el contexto de sus dimensiones (*Report*) no coincide con el tipo de su contexto real (*Patient*).

IV. Método

La Fig. 4 muestra de forma esquemática las distintas etapas que constituyen el método desarrollado. Tal y como puede observarse, existe una fase *off-line*, en la cual se crean unos índices sobre la ontología para acelerar la fase *on-line*, mediante la cual se extraen hechos de forma automática a partir de un modelo MD y se puebla una tabla de hechos con instancias de una ontología.

Fase on-line: En primer lugar, para poder extraer hechos válidos el analista tiene que diseñar un esquema MD a partir de conceptos y propiedades de la ontología de aplicación, obteniendo así un esquema MD según la Def. 1. A continuación, se ejecuta el algoritmo de computación de hechos, que es el encargado de agrupar las dimensiones y combinar valores de dimensión dentro de un grupo y entre grupos. Finalmente, se filtran sólo las combinaciones válidas de valores de dimensión de acuerdo a la Def. 7.

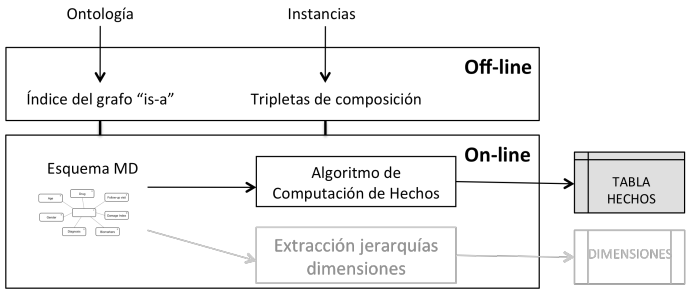

Fig. 4. Arquitectura del método propuesto.

Fase off-line: Para poder realizar los pasos anteriores de forma eficiente, se preprocesan tanto la ontología como las instancias. Con respecto a las instancias, se crean las *tripletas de composición*, las cuales permiten que la extracción de instancias y valores alcanzables desde otras instancias a través de la composición de propiedades sea eficiente. Las tripletas de composición se enriquecen con el *índice del grafo is-a* de forma que tanto las instancias asertadas de un concepto como las inferidas puedan ser extraídas eficientemente. El *índice del grafo is-a* se construye aplicando un esquema de etiquetado basado en intervalos a la jerarquía *"is-a"* de la ontología, en donde las relaciones *"is-a"* implícitas se han explicitado. De esta forma, cada nodo tiene asociado todos sus ancestros y descendientes en forma de intervalos de números enteros. Para más detalles acerca del esquema de etiquetado basado en intervalos consultar [13], [14]. Por ejemplo, si queremos extraer todas las instancias de tipo *Patient* junto con las enfermedades que tienen, con estos índices seremos capaces de extraer instancias cuyo tipo asertado es *Patient* y todas aquellas instancias clasificadas bajo *Patient*, como por ejemplo *Pacientes Reumáticos,* junto con sus correspondientes enfermedades clasificadas bajo el concepto *Disease*. Por tanto, con estos índices somos capaces de realizar razonamiento básico a nivel de instancia sin la necesidad de un razonador.

A. Tripletas de composición

Siguiendo con el ejemplo en desarrollo, para las instancias foco, en este caso las instancias de tipo *Patient*, se quiere encontrar los distintos valores de dimensión que las caracterizan. Para ello, se define un tipo de tripletas especial, las *tripletas de composición*.

Definición 8 (Tripletas de composición). Una tripleta de composición es una tripleta con la estructura (sujeto, predicado, objeto) donde el sujeto es un recurso, el objeto puede ser un recurso o un literal y el predicado es un camino (*path)* de composición del sujeto al objeto. Un *path* de composición es una secuencia alternada de propiedades y recursos que conectan el sujeto con el objeto.

La Tabla I muestra las tripletas de composición de la instancia *Patient_0*. La cuarta columna hace referencia al índice *"is-a"* y lo que realmente se guarda es una clave ajena al concepto al cual la instancia objeto pertenece. La misma columna existe también para el sujeto. Actualmente, esta tabla se genera al parsear el archivo RDF/OWL que describe las instancias atravesando el grafo de cada instancia (grafo como el de la Fig. 2).

TABLA I
TRIPLETAS DE COMPOSICIÓN DE LA INSTANCIA *PATIENT_0* A CADA UNO DE LOS VALORES DE DIMENSIÓN

Subject	**Composition Path**	**Object**	**Dimension**
Patient_0	/sex	Female	Gender
Patient_0	/hasAge	30	hasAge
Patient_0	/hasProfile/GeneProfile_0/relatedGene	HBB	Gene
Patient_0	/hasProfile/GeneProfile_0/relatedGene	CFRT	Gene
Patient_0	/hasReport/Report_0/date	18112008	date
Patient_0	/hasReport/Report_0/hasDiagnosis	PACJ	Disease
Patient_0	/hasReport/Report_0/hasSection/LBT_0/measures...	PlateletLow	BloodIndicant
Patient_0	/hasReport/Report_0/hasSection/LBT_0/measures...	NeutrophilLow	BloodIndicant
Patient_0	/hasReport/Report_0/hasSection/RE_0/damageIndex	50	damageIndex
Patient_0	/hasReport/Report_0/hasSection/Treat_0/hasTherapy	Naproxen	DrugTherapy
Patient_0	/hasReport/Report_0/hasSection/Treat_0/hasTherapy	Methotrexate	DrugTherapy
Patient_0	/hasReport/Report_0/hasSection/Treat_0/hasTherapy	Infliximab	DrugTherapy
Patient_0	/hasReport/Report_1/date	30122008	date
Patient_0	/hasReport/Report_1/hasDiagnosis	PACJ	Disease
...	...	...	...

B. Algoritmo de Computación de Hechos

El algoritmo que obtiene combinaciones válidas de valores de dimensión tiene dos etapas: partición de las dimensiones en grupos independientes y combinación de valores de dimensión dentro de los grupos y entre grupos. Sea *D* el conjunto de dimensiones. *G* es una partición de *D* formada de la siguiente manera:

$\forall g \in G, \forall T_i, T_j \in g, \quad T_i \text{ is dependent of } T_j.$

$\neg \exists g, g' \in G \text{ such that } T_i \in g, T_j \in g' \text{ and } T_i \text{ is dependent of } T_j.$

$\forall g \in G, context(g) = context(T_1,...,T_n) \text{ such that } T_i \in g, n = |g|.$

Cada uno de los grupos *g* de la partición *G* contiene dimensiones dependientes y el contexto de cada grupo es el contexto del conjunto de dimensiones en el grupo. Esta partición puede calcularse de manera eficiente comprobando las dependencias entre dimensiones a través de los *paths* de composición de las instancias. Con respecto al ejemplo que estamos desarrollando, la partición *G* de dimensiones junto con el contexto de cada grupo (siguiendo la sintaxis $Grupo_{contexto}$<*dimensiones*>) es la siguiente:

$G1_{Patient}$<*Gender*>,
$G2_{Patient}$<*Patient.hasAge*>,
$G3_{GeneProfile}$<*Gene*>,
$G4_{Report}$<*Report.date, RE.damageIndex, Disease, BloodIndicant, Drug*>

La segunda etapa del algoritmo consiste en combinar valores de dimensión dentro de cada grupo y entre grupos para cada instancia foco. Esto se consigue aplicando la siguiente expresión del álgebra relacional sobre las tripletas de composición:

$$\bowtie_{g_i \in G} (\sigma_{valid\_comb.}(\bigcup_{I \in Inst(context(g_i))} \times_{T_j \in g_i} \Pi_{(T_j)} I))$$

Es decir, para cada instancia *I* del contexto de un grupo, se realiza el producto cartesiano de la tuplas que resultan de proyectar dicha instancia sobre cada dimensión del grupo. Las tuplas resultantes se filtran a través del operador de selección, el cual sólo selecciona las tuplas que son combinaciones válidas según la Def. 7. Estas operaciones se realizan para cada grupo de la partición *G* y finalmente, las tuplas obtenidas de cada grupo se unen mediante una operación de *join*. La operación de proyección de las instancias *I* sobre las dimensiones T_j se realiza de manera eficiente gracias a las tripletas de composición. La Tabla II muestra el resultado de aplicar la fórmula anterior al ejemplo desarrollado. En ella se muestran los valores de dimensión sin realizar el producto cartesiano dentro de un grupo ni el *join* final entre grupos. Tal y como puede observarse, la generación de valores de dimensión para los grupos G1, G2 y G3 es trivial, ya que cada grupo está compuesto sólo por una dimensión. Para el grupo G1 se proyecta la instancia *Patient_0* sobre la dimensión *Gender*, obteniendo el valor *Female*. Para el grupo G2 se proyecta *Patient_0* sobre *Patient.hasAge*, obteniendo el valor *30*. Para el grupo G3 se proyecta *GeneProfile_0* sobre *Gene*, obteniendo *HBB* y *CFRT*. Para el grupo G4, su contexto es *Report*, y existen dos instancias de *Report*, *Report_0* y *Report_1* en *Patient_0*. Por tanto, primero se proyecta *Report_1* sobre cada dimensión del grupo obteniendo los valores de la primera fila del grupo G4 y luego *Report_0*, obteniendo los valores de la segunda fila de G4.

TABLA II
GRUPOS DE DIMENSIONES Y VALORES DE DIMENSIÓN PARA LA INSTANCIA FOCO *PATIENT_0*

Patient	**G1**	**G2**	**G3**	**G4**				
	Gender	hasAge	Gene	date	damageIndex	Disease	BloodIndicant	DrugTherapy
Patient_0	Female	30	HBB	30122008	29	PACJ	Cprotein+ NeutrophilLow	Naproxen
			CFRT	18122008	50	PACJ	NeutrophilLow PlateletLow	Naproxen Infliximab Etacernept

Después de aplicar el producto cartesiano de los valores de dimensión para cada fila en cada grupo se obtienen combinaciones que puede que no sean válidas. Para filtrar las combinaciones válidas se utilizar el operador de selección sobre las tuplas de cada grupo. Finalmente, las tuplas de cada grupo se unen mediante un *join* utilizando la instancia foco como clave. En el ejemplo, todas las combinaciones que se generan de la Tabla II son válidas, y el resultado de hacer el *join* de las tuplas de cada grupo se muestra en la Tabla III.

V. EVALUACIÓN

Para la evaluación de nuestro método, se ha diseñado un experimento con la ontología de aplicación sobre pacientes de la Fig. 1, la cual es una simplificación de la ontología de aplicación desarrollada en el proyecto Health-e-Child[8]. Esta ontología contiene 245 clases y 15 propiedades. Las instancias se han generado de forma sintética a partir de las características de un pequeño conjunto de pacientes, de forma que se pueden generar conjuntos de instancias de cualquier tamaño y con las características estructurales deseadas. En nuestro caso hemos generado 3000 instancias de paciente con una estructura heterogénea. Cada paciente tiene información sobre uno y tres genes, entre uno y cinco informes, entre uno y tres indicadores de la sangre y entre uno y tres tratamientos. El número total de instancias generadas es de 188.742.

La Fig. 5 muestra la evaluación de la ejecución del método propuesto con las instancias de ontología anterior. Dado un conjunto de dimensiones como el del ejemplo desarrollado, se ha generado una tabla de hechos para cada posible subconjunto de dimensiones. Es decir, se han generado todas las tablas de hechos que pueden generarse con subconjuntos de ocho dimensiones y se han organizado de acuerdo al número de dimensiones de cada tabla de hechos (eje *x*), desde

[8] http://www.heath-e-child.org/

una dimensión hasta ocho. El eje *y* muestra el tiempo que tarda en generarse la tabla de hechos en segundos. La complejidad temporal es lineal con respecto al número de dimensiones involucradas, lo cual demuestra la escalabilidad de nuestra propuesta.

TABLA III
TABLA DE HECHOS GENERADA PARA LA INSTANCIA FOCO PATIENT_0

FACT TABLE								
Focus inst.	**Gender**	**hasAge**	**Gene**	**date**	**dIndex**	**Disease**	**BloodIndicant**	**DrugTherapy**
Patient_0	Female	30	HBB	30122008	29	PACJ	CProtein+	Naproxen
Patient_0	Female	30	HBB	30122008	29	PACJ	NeutrophilLow	Naproxen
Patient_0	Female	30	HBB	18122008	50	PACJ	NeutrophilLow	Naproxen
Patient_0	Female	30	HBB	18122008	50	PACJ	NeutrophilLow	Infliximab
Patient_0	Female	30	HBB	18122008	50	PACJ	NeutrophilLow	Etacernept
Patient_0	Female	30	HBB	18122008	50	PACJ	PlateletLow	Naproxen
Patient_0	Female	30	HBB	18122008	50	PACJ	PlateletLow	Infliximab
Patient_0	Female	30	HBB	18122008	50	PACJ	PlateletLow	Etacernept
Patient_0	Female	30	CFRT	30122008	29	PACJ	CProtein+	Naproxen
…	…	…	…	…	…	…	…	

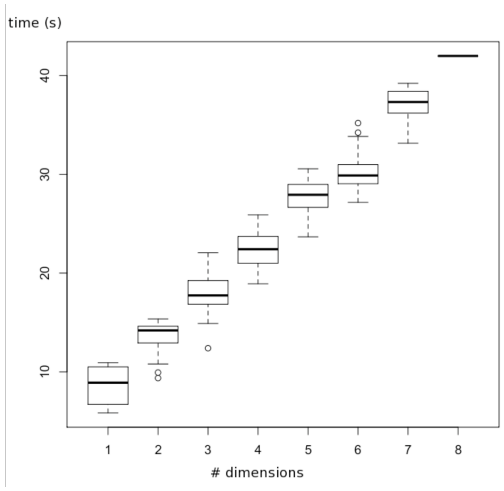

Fig. 5. Tiempo empleado en la generación de tablas de hechos con respecto al número de dimensiones que involucra cada tabla de hechos.

VI. Conclusiones y trabajo futuro

En este trabajo se ha presentado un propuesta para la identificación y extracción automática de hechos a partir de un repositorio de instancias expresadas en RDF/(S) y OWL. Según nuestros conocimientos, este es el primer trabajo que aborda el problema de identificar y extraer hechos a partir de instancias ontológicas. Por tanto, creemos que nuestro trabajo abre nuevas perspectivas de investigación, ya que combina las técnicas de almacenes de datos y OLAP con las de la Web Semántica. Se ha demostrado que el método propuesto es escalable incluso con repositorios de instancias con una estructura compleja y heterogénea. Como trabajo futuro se considera explorar nuevas técnicas de indexación de los datos que aceleren el proceso. Otro tema pendiente es el estudio de la agregabilidad de las medidas sobre las tablas de hechos obtenidas, así como la población de las jerarquías de dimensión a partir de fuentes semánticas. Otra línea de investigación abierta es el estudio de los *mappings* entre el diseño multidimensional del analista y la ontología de aplicación cuando el primero ha sido diseñado de manera independiente a las ontologías de dominio.

REFERENCIAS

[1] S. Chaudhuri, and U. Dayal, An overview of data warehousing and OLAP technology. *SIGMOD Rec.* 26(1), pp. 65-74, 1997.

[2] J.M. Pérez, R. Berlanga, M.J. Aramburu, and T.B. Pedersen, Integrating Data Warehouses with Web Data: A Survey. *IEEE Trans. Knowl. Data Eng.* 20(7), pp. 940-955, 2008.

[3] C. Chen, X. Yan, F. Zhu, J. Han, and P.S. Yu, "Graph OLAP: Towards Online Analytical Processing on Graphs," in *International Conference of Data Management (ICDM)*, pp. 103-112, 2008.

[4] DBpedia Project: http://dbpedia.org

[5] BioRDF Project: http://esw.w3.org/topic/BioRDF_Top_Level_Task

[6] V. Nebot, R. Berlanga, J.M. Pérez, and M.J. Aramburu, Multidimensional Integrated Ontologies: A Framework for Designing Semantic Data Warehouses, *Journal on Data Semantics, JoDS XIII: Special Issue "Semantic Data Warehouses"*, vol. 5530, pp.1-35, 2009.

[7] O. Romero, and A. Abello, Automating multidimensional design from ontologies, in *Proc. of the 10th Int. Workshop on Data Warehousing and OLAP (DOLAP),* pp. 1-8, 2007.

[8] M. Niinimaki and T. Niemi, An ETL Process for OLAP Using RDF/OWL Ontologies, *Journal on Data Semantics, JoDS XIII: Special Issue "Semantic Data Warehouses"*, copyright (c) Springer, 2009.

[9] K. Garwood, T. McLaughlin, C. Garwood, S. Joens, N. Morrison, C.F. Taylor, K. Carroll, C. Evans, A.D. Whetton, S. Hart, D. Stead, Z. Yin, A.J. Brown, A. Hesketh, K. Chater, L. Hansson, M. Mewissen, P. Ghazal, J. Howard, K.S. Lilley, S.J. Gaskell, A. Brass, S.J. Hubbard, S.G. Oliver, and N.W. Paton, PEDRo: a database for storing, searching and disseminating experimental proteomics data. *BMC Genomics*, 5(68), 2004.

[10] F. Baader, D. Calvanese, D.L. McGuinness, D. Nardi, and P.F. Patel-Schneider, *The Description Logic Handbook: Theory, Implementation, and Applications,* in Description Logic Handbook. Cambridge University Press 2003.

[11] V. Nebot, and R. Berlanga, Building tailored ontologies from very large knowledge resources, in *11th Int. Conference on Enterprise Information Systems* (ICEIS), vol. 2, pp. 144-151, 2009.

[12] T.B. Pedersen, C.S. Jensen, and C.E. Dyreson, A foundation for capturing and querying complex multidimensional data. *Information Systems*, 26(5), pp. 383-423, 2001.

[13] V. Nebot, and R. Berlanga, Efficient Retrieval of Ontology Fragments Using an Interval Labeling Scheme, in *13th edition of the Spanish Conference on Software Engineering and Databases (JISBD) 2008.*

[14] V. Nebot, and R. Berlanga, Efficient Retrieval of Ontology Fragments Using an Interval Labeling Scheme, *Inf. Sci.,* 179(24), pp.4151-4173, 2009.

Victoria Nebot nació en Castellón, España, el 21 de Junio de 1984. Se graduó en Ingeniería Informática en la Universidat Jaume I de Castellón en el 2007.

Actualmente sus esfuerzos se encaminan hacia la obtención del título de doctora. Para ello, cuenta con una beca de formación predoctoral financiada por el Ministerio de Educación y Ciencia de España. Entre sus campos de interés destacan la explotación y análisis de datos semi-estructurados o con estructura compleja derivados de la Web Semántica, así como las tecnologías de almacenes de datos y OLAP.

Durante sus estudios universitarios, recibió diversos premios como el Premio al Rendimiento Académico en el 2005 y el Premio Extraordinario de Ingeniería Informática en el 2007.

Rafael Berlanga nació en Valencia, España, el 8 de Octubre de 1969. Se graduó en la Facultad de Ciencias Físicas de Valencia en el año 1992, donde obtuvo el grado de doctor en el año 1996.

Ejerce como profesor Titular de Universidad en la Universitat Jaume I de Castellón, desde el año 1999, aunque se incorporó a dicha universidad en el año 1992 como profesor ayudante. Desde 1997, dirige el grupo de investigación Bases de Conocimiento Temporal (Temporal Knowledge Bases Group), en el cual se han formado a ocho doctores. Durante ese tiempo, ha dirigido varios proyectos de investigación del Programa Nacional, dando lugar a una notable productividad científica, de la que destacan varias publicaciones en revistas de reconocido prestigio como IEEE Transaction on Knowledge Engineering, Information Processing & Management e Information Sciences.

Los campos de interés previos fueron el razonamiento temporal y la planificación en Inteligencia Artificial, y los actuales son Bases de Datos, Recuperación de la Información y la Web Semántica.