Rev. Téc. Ing. Univ. Zulia. Vol. 34, Nº 2, 171 - 180, 2011

Efficient processing of RDF documents with directed hypergraphs

Amadís Antonio Martínez-Morales[1], María-Esther Vidal[2]

[1]Departamento de Computación, Facultad Experimental de Ciencias y Tecnología, Universidad de Carabobo. Valencia, Venezuela. aamartin@uc.edu.ve

[2]Departamento de Computación y Tecnología de la Información, Universidad Simón Bolívar. Valle de Sartenejas, Baruta, Venezuela. mvidal@ldc.usb.ve

Abstract

Resource Description Framework (RDF) is a proposal of the WWW Consortium (W3C) to express metadata about resources in the Web. The RDF data model has been formalized using different graph-based representations, each one with its own limitations with respect to expressive power and support for the tasks of query answering and semantic reasoning. In this paper, we show the advantages of the directed hypergraph-based representation for RDF documents, and analyze space complexity required to store an RDF document, and the impact of this representation on the time complexity of the query answering task. In addition, we empirically compare the DH approach with respect to the labeled directed graphs and bipartite graphs representations. Experimental results show that the time/space tradeoff of the DH-based representation outperforms the other two approaches.

Key words: data model, directed hypergraphs, Resource Description Framework (RDF), semantic web.

Procesamiento eficiente de documentos RDF con hipergrafos dirigidos

Resumen

Resource Description Framework (RDF) es una propuesta del *WWW Consortium* (W3C) para expresar metadatos acerca de recursos en el *Web*. RDF ha sido formalizado utilizando diversas representaciones basadas en grafos, cada una tiene sus propias limitaciones con respecto a poder expresivo y soporte para las tareas de responder consultas y razonamiento semántico. En este trabajo se muestran las ventajas de la representación basada en hipergrafos dirigidos (HD) para documentos RDF, en función de la cantidad de espacio de almacenamiento requerido y el tiempo de evaluación de consultas. A tal fin, se analiza la complejidad en espacio requerido para almacenar un documento RDF y el impacto en la complejidad en tiempo de la tarea de responder consultas. Además, se reportan los resultados de un estudio empírico donde se compara el enfoque basado en HD con respecto a representaciones basadas en grafos etiquetados dirigidos y grafos bipartitos. Los resultados experimentales obtenidos dan indicios de que la relación tiempo/espacio favorece a la representación basada en HD.

Palabras clave: hipergrafos dirigidos, modelo de datos, *Resource Description Framework* (RDF), *web* semántica.

1. Introducción

Resource Description Framework (RDF) es un lenguaje propuesto por el W3C para describir recursos de información de una manera comprensible por máquinas y proporcionar soporte para que estas descripciones puedan ser procesadas por aplicaciones.

La estructura básica en el modelo de datos de RDF es una terna de la forma (*s*, *p*, *o*), denomi-

nada terna RDF, donde *s* (sujeto) es el recurso que se está describiendo, *p* (predicado) es una propiedad del recurso *s* y *o* (objeto) es el valor de la propiedad *p*. Intuitivamente, un conjunto de ternas RDF puede verse como un grafo: los recursos son los nodos y las propiedades son los arcos que los conectan. Por esta razón, un conjunto de ternas RDF se denomina grafo RDF [1].

Un sistema de administración para RDF es un *software* de propósito general que permite la creación, almacenamiento y manipulación de documentos RDF. Un sistema de administración para RDF debe, además, proporcionar soporte para dos tareas fundamentales: (1) responder consultas realizadas por usuarios y agentes de *software* sobre documentos RDF y (2) razonamiento semántico sobre documentos RDF, para descubrir interrelaciones entre los recursos descritos y poder inferir nuevas ternas RDF a partir de ternas RDF existentes [2]. Se debe destacar que las características particulares de RDF influyen en la complejidad de estas tareas, las cuales, en el caso general, son problemas NP-completos [3, 4].

El modelo de datos de RDF permite diversas representaciones para la misma información: grafos etiquetados dirigidos (GED) [1, 4], hipergrafos no dirigidos (HND) [5], grafos bipartitos (GB) [5, 6] e hipergrafos dirigidos (HD) [2, 7]. Cada una de estas representaciones tiene sus propias limitaciones con respecto a: (1) el poder expresivo del modelo de datos de RDF y (2) el soporte para las tareas de responder consultas y razonamiento semántico. En este trabajo se muestran las ventajas del modelo formal basado en HD para la representación de documentos RDF [2, 7]. A tal fin, se estudia la complejidad en espacio de este enfoque para el almacenamiento de información y el impacto de esta representación sobre la tarea de responder consultas (Sección 2). Así mismo, se reportan comparaciones empíricas con respecto a otras representaciones existentes, específicamente GED y GB, para la evaluación de la calidad del modelo planteado (Sección 3). Finalmente, se presentan las conclusiones y el trabajo futuro (Sección 4). Este artículo es una versión resumida de los resultados finales obtenidos en [2].

2. Marco teórico

Un grafo RDF se define formalmente como sigue [3, 4, 8]: sea $\boldsymbol{V}$ un conjunto infinito de términos, particionado en tres conjuntos infinitos: $\boldsymbol{U}$ (identificadores universales de recursos), $\boldsymbol{B} = \{b_j : j \geq 0\}$ (identificadores de nodos existenciales) y $\boldsymbol{L}$ (literales). Una terna $t = (s, p, o) \in (\boldsymbol{U} \cup \boldsymbol{B}) \times \boldsymbol{U} \times (\boldsymbol{U} \cup \boldsymbol{B} \cup \boldsymbol{L})$ se denomina terna RDF, donde *s* es el sujeto, *p* es el predicado y *o* es el objeto de *t*. Un conjunto de ternas RDF $T = \{(s, p, o) : (s, p, o) \in (\boldsymbol{U} \cup \boldsymbol{B}) \times \boldsymbol{U} \times (\boldsymbol{U} \cup \boldsymbol{B} \cup \boldsymbol{L})\}$, se denomina grafo RDF. El universo de *T*, $univ(T)$, es el conjunto de valores de $\boldsymbol{U} \cup \boldsymbol{B} \cup \boldsymbol{L}$ que ocurren en las ternas de *T*. El tamaño de *T*, $|T|$, es el número de ternas RDF en *T*. Sea *Var* un conjunto infinito de variables disjunto dos a dos de $\boldsymbol{U}$, $\boldsymbol{B}$ y $\boldsymbol{L}$. Una terna $(v_1, v_2, v_3) \in (\boldsymbol{U} \cup Var) \times (\boldsymbol{U} \cup Var) \times (\boldsymbol{U} \cup \boldsymbol{L} \cup Var)$ es un patrón basado en ternas. Un conjunto de patrones basados en ternas $P = \{(v_1, v_2, v_3) : (v_1, v_2, v_3) \in (\boldsymbol{U} \cup Var) \times (\boldsymbol{U} \cup Var) \times (\boldsymbol{U} \cup \boldsymbol{L} \cup Var)\}$ se denomina patrón basado en grafos, siendo $var(P)$ el conjunto de variables en *P*. Una consulta *Q* es una expresión de la forma $Q: H \leftarrow B$, donde *B* es un patrón basado en grafos y $H = \langle H_1, \ldots, H_n \rangle$ es una lista de variables tal que $\langle \forall i: 1 \leq i \leq n: H_i \in var(B) \rangle$. *H* se denota por $head(Q)$ y *B* se denota por $body(Q)$. Si $|B| = 1$ entonces *Q* es una consulta básica, si $|B| > 1$ entonces *Q* es una consulta conjuntiva. Dado un grafo RDF *T*, la complejidad en espacio para almacenar el documento RDF representado por *T* es $O(|T|)$. Si *T* no contiene identificadores de nodos existenciales, la complejidad de la tarea de responder una consulta *Q* sobre *T* es $O(|T|^k)$, donde $k \geq 1$ es el número de patrones basado en ternas en $body(Q)$ [9]. Por otra parte, el problema de responder consultas sobre un grafo RDF con identificadores de nodos existenciales es NP-completo [4], debido a que los identificadores de nodos existenciales representan información incompleta [10].

El modelo de datos de RDF permite diversas representaciones para la misma información: grafos etiquetados dirigidos (GED) [1, 4], hipergrafos no dirigidos (HND) [5], grafos bipartitos (GB) [5, 6] e hipergrafos dirigidos (HD) [2, 7]. En este trabajo se muestran las bondades del modelo basado en HD para RDF. Básicamente, un hi-

pergrafo dirigido está definido por un conjunto de nodos W y un conjunto de hiperarcos E. Cada hiperarco $e \in E$ conecta un conjunto de nodos origen ($orig(e)$) con un conjunto de nodos destino ($dest(e)$). Formalmente, un hipergrafo dirigido RDF se define como sigue [2, 7]:

Definición 1

Sea T un grafo RDF. El hipergrafo dirigido RDF asociado con T es una terna $\boldsymbol{H}(T) = (W, E, \rho)$ tal que:

- $W = \{ w: w \in univ(T) \}$ es el conjunto de nodos.
- $E = \{ e_i: 1 \leq i \leq |T| \}$ es el conjunto de hiperarcos.
- $\rho: W \times E \rightarrow \{'s', 'p', 'o'\}$ es la función de rol de los nodos con respecto a los hiperarcos. Sea $t \in T$ una terna RDF, $e \in E$ un hiperarco, y $w \in orig(e) \cup dest(e)$ un nodo. Se tiene que:
  - $(\rho(w, e) = 's') \Leftrightarrow (w \in orig(e)) \wedge (w \in sub(\{t\}))$
  - $(\rho(w, e) = 'p') \Leftrightarrow (w \in orig(e)) \wedge (w \in pred(\{t\}))$
  - $(\rho(w, e) = 'o') \Leftrightarrow (w \in dest(e)) \wedge (w \in obj(\{t\}))$

Dado un grafo RDF T, cada nodo de $\boldsymbol{H}(T) = (W, E, \rho)$ corresponde con un elemento $w \in univ(T)$. De esta manera, la información sólo es almacenada en los nodos, y los hiperarcos sólo se encargan de preservar el rol de cada nodo y el concepto de dirección de los grafos RDF. Por lo tanto, este enfoque requiere menor cantidad de memoria que otras representaciones basadas en grafos para RDF [2, 7]. Dado un grafo RDF T y $\boldsymbol{H}(T) = (W, E, \rho)$ el hipergrafo dirigido RDF asociado con T, se tiene entonces que $|W| = |univ(T)|$ y $|E| = |T|$. Por lo tanto, la complejidad en espacio requerido para almacenar un documento RDF bajo este enfoque es $O(\text{máx}(|univ(T)|, |T|))$.

La transformación de un grafo RDF a un hipergrafo dirigido RDF se muestra en el Algoritmo 1. Dado un grafo RDF T, el Algoritmo 1 recorre todas las ternas RDF en T (línea 4). Por cada terna RDF $t = (s, p, o) \in T$, el algoritmo realiza las siguientes operaciones: (1) agrega s, p y o al conjunto de nodos W (línea 5), (2) agrega el identificador del hiperarco correspondiente a t al conjunto de hiperarcos E (línea 6), y (3) agrega los roles (sujeto, predicado u objeto) de cada nodo con respecto al hiperarco a la función de rol ρ (líneas 7-9). Al final, retorna el hipergrafo dirigido RDF asociado con T, $\boldsymbol{H}(T)$ (línea 11). Nótese que $\boldsymbol{H}(T)$ define un índice implícito basado en posición sobre las ternas RDF en T, que puede tener impacto positivo en la tarea de responder consultas sobre T. Los costos asociados con esta transformación se definen en la Proposición 1.

Proposición 1

$\boldsymbol{H}(T)$ puede ser construido con complejidad $O(|T|)$.

Intuitivamente, la operación más costosa que se ejecuta en el ciclo **for** del Algoritmo 1 es la inserción de elementos en el conjunto de nodos W. Sin embargo, si W se implementa como un conjunto basado en *hash*, esta operación de inserción puede ser ejecutada con complejidad $O(1)$. Por lo tanto, la construcción del hipergrafo dirigido asociado con un grafo RDF T tiene complejidad $O(|T|)$. De esta manera, si las estructuras de datos para el almacenamiento de un HD están basadas en *hash*, se garantiza complejidad en tiempo constante para las operaciones de inserción y consulta de la estructura. Este resultado reduce la complejidad del algoritmo de construcción del grafo bipartito asociado con un grafo RDF T, $O(|T| \lg |T|)$, presentada en [5, 6]. Con respecto a la tarea de responder consultas, los Algoritmos 2 y 3 introducen las tareas de evaluación de consultas básicas y conjuntivas, respectivamente, sobre un grafo RDF T sin identificadores de nodos existenciales, utilizando la representación propuesta (Figura 1).

Si Q: $H \leftarrow B$ es una consulta básica puede asumirse, sin pérdida de generalidad, que $var(H) = var(B)$, dado que $|B| = 1$. De esta manera, la evaluación de Q se traduce en responder $B = body(Q)$. El Algoritmo 2 introduce el caso en el cual $|var(B)| = 1$, es decir, cuando B contiene una sola variable. Para evaluar una consulta de este tipo, el Algoritmo 2 procede como sigue: el primer paso consiste en determinar el rol de la variable en B: sujeto (línea 1), predicado (línea 7) u objeto (línea 12). Por ejemplo, en caso de que la variable tenga el rol de sujeto según la función ρ, se procede a construir los conjuntos E_p y E_o de hiperarcos incidentes en los nodos instanciados p y o, los cuales tienen los roles de predicado y objeto, respectivamente (líneas 2-3). En la línea 4 se intersectan los conjuntos E_p y E_o obtenidos ante-

```
GETHYPERGRAPH(T)
1  W ← ∅
2  E ← ∅
3  i ← 1
4  for each (s, p, o) ∈ T do:
5     W ← W ∪ {s, p, o}
6     E ← E ∪ {e_i}
7     ρ(s, e_i) ← 's'
8     ρ(p, e_i) ← 'p'
9     ρ(o, e_i) ← 'o'
10    i ← i + 1
11 return H(T) = (W, E, ρ)
```

Algoritmo 1. Transformación de un grafo RDF a un hipergrafo dirigido RDF

```
ANSWERCONJUNCTIVEQUERY(Q, (W, E, ρ))
1  for each C_j ∈ body(Q) do:
2     S_j ← ANSWERBASICQUERY(C_j, (W, E, ρ))
3  ans ← S_1
4  for j ← 2 to |body(Q)| do:
5     ans ← ans ⋈ S_j
6  return ans
```

Algoritmo 3. Evaluación de consultas conjuntivas acíclicas

```
BASICQANSONEVAR((s, p, o), (W, E, ρ))
1  if VARIABLE(s):
2     E_p ← {e ∈ E : ρ(p, e) = 'p'}
3     E_o ← {e ∈ E : ρ(o, e) = 'o'}
4     E_Q ← E_p ∩ E_o
5     ans ← {(x, p, o) : e ∈ E_Q ∧ ρ(x, e) = 's'}
6  else
7     if VARIABLE(p):
8        E_s ← {e ∈ E : ρ(s, e) = 's'}
9        E_o ← {e ∈ E : ρ(o, e) = 'o'}
10       E_Q ← E_s ∩ E_o
11       ans ← {(s, y, o) : e ∈ E_Q ∧ ρ(y, e) = 'p'}
12    else
13       E_s ← {e ∈ E : ρ(s, e) = 's'}
14       E_p ← {e ∈ E : ρ(p, e) = 'p'}
15       E_Q ← E_s ∩ E_p
16       ans ← {(s, p, z) : e ∈ E_Q ∧ ρ(z, e) = 'o'}
17 return ans
```

Algoritmo 2. Evaluación de consultas básicas, una variable

Figura 1. Algoritmos de construcción y consulta de un hipergrafo dirigido RDF utilizando la representación propuesta.

riormente con el objetivo de determinar aquellos hiperarcos que son relevantes para generar el conjunto solución de la consulta, el cual finalmente se construye en la línea 5. De manera análoga se procede en los otros dos casos (cuando la variable tenga el rol de predicado o de objeto en B según la función ρ). El Algoritmo 2 puede modificarse sin dificultad cuando $|var(B)| = 0$ ó $|var(B)| > 1$.

Si Q es una consulta conjuntiva, se parte de la suposición de que Q es acíclica, lo cual permite que la evaluación de Q sobre T pueda realizarse en un número polinomial de pasos con respecto a $|T|$ [11]. El problema general de responder una consulta conjuntiva Q es W[1]-completo en el tamaño de Q [12]. Según el Algoritmo 3, la evaluación de Q se realiza como sigue: para cada sub-objetivo $C_j \in body(Q)$ $(1 \leq j \leq m = |body(Q)|)$ se calcula el conjunto solución S_j sobre $var(C_j)$, utilizando un algoritmo para la evaluación de consultas básicas, dependiendo del número de variables en C_j. Una vez obtenidos todos los conjuntos solución S_j $(1 \leq j \leq m)$, el resultado final de la consulta está definido por el conjunto $\pi_H(S_1 \bowtie S_2 \bowtie \ldots \bowtie S_m)$.

3. Resultados experimentales

La implementación de los algoritmos y estructuras de datos se realizó utilizando el lenguaje de programación Python, y las pruebas se realizaron en un computador con procesador AMD Opteron y 16 GiB de memoria RAM bajo el sistema operativo Linux CentOS versión 5.0.

Los experimentos se realizaron sobre distintos conjuntos de datos, reales y sintetizados. El prototipo desarrollado se evaluó sobre cinco conjuntos de datos reales [13]: Webcrawl (www.activerdf.org/webcrawl_10k.nt), FOAF (rdfweb.org/2003/02/28/cwm-crawler-output.rdf), Mindswap (www.cs.umd.edu/~hendler/2003/MindPeople4-30.rdf), ontoworld.org Semantic Wiki (ontoworld.org/RDF/ontoworld.xml) y Wikipedia[3], una conversión de Wikipedia en inglés a RDF (labs.systemone.at/wikipedia3). Los datos sintetizados fueron generados por el Lehigh Uni-

versity *Benchmark* (LUBM) [14]. Este *benchmark* está basado en la ontología Univ-Bench, que describe universidades, departamentos y las actividades académicas que ocurren en ellos. Cada conjunto de datos está representado por la expresión *Univ*(*N*, *S*), que denota el conjunto de datos que contiene *N* universidades generadas a partir de la semilla *S*, comenzando por el índice 0. Para efectos de este estudio, se generaron cinco conjuntos de datos: *Univ*(1, 0), *Univ*(5, 0), *Univ*(10, 0), *Univ*(20, 0) y *Univ*(50, 0), que contienen datos de 1, 5, 10, 20 y 50 universidades, respectivamente. Las características de estos conjuntos de datos se muestran en la Tabla 1.

Por otra parte, LUBM ofrece 14 consultas, que toman en cuenta los siguientes factores: tamaño de la entrada, selectividad, complejidad, utilización de información jerárquica y capacidad de inferencia. Estas consultas fueron diseñadas con énfasis en tamaño grande de la entrada y alta selectividad [14]. En este estudio experimental sólo se consideraron aquellas consultas que no involucran la tarea de razonamiento semántico para ser respondidas, ocho en total del conjunto original de 14 consultas. Adicionalmente fueron diseñadas 12 consultas, respetando los criterios de diseño establecidos, para complementar el conjunto total de consultas.

Se consideraron varias métricas para medir el desempeño de las representaciones basadas en hipergrafos dirigidos (HD), grafos etiquetados dirigidos (GED) y grafos bipartitos (GB): (1) tiempo de almacenamiento (en segundos) de los conjuntos de datos en las estructuras de datos definidas para la representación correspondiente (tiempo de carga), (2) tamaño en memoria (en MiB) de las estructuras de datos, una vez cargados los conjuntos de datos en ellas (tamaño de la estructura), (3) relación entre el tiempo de carga y el tamaño de la estructura de datos resultante (*time/space tradeoff*) y (4) tiempo promedio de respuesta (en segundos) de las consultas, cada una fue ejecutada tres veces sobre los conjuntos de datos (tiempo de respuesta de las consultas).

La Figura 2 muestra los resultados con respecto al tiempo de carga de los conjuntos de datos reales en las estructuras de datos definidas para cada representación. Se debe destacar que, en todos los casos, el tiempo de carga requerido por el modelo basado en HD es inferior al de los modelos basados en GED y GB. El tiempo de carga requerido por la representación basada en HD es, en promedio, 1/3 del tiempo requerido por los otros dos enfoques.

En la Figura 3 se presentan los resultados con respecto al tamaño (MiB) de las estructuras de datos, definidas para cada representación, asociadas con los conjuntos de datos reales. Como puede notarse, el espacio de almacenamiento requerido por el modelo basado en HD es inferior al del modelo basado en GED, y ligeramente superior al espacio requerido por el modelo basado en GB. Esto se debe a que los índices requeridos por una estructura de datos basada

Tabla 1
Características de los conjuntos de datos

Tipo de datos	Conjunto de datos	Número de ternas RDF	Tamaño (MiB)
Reales	Mindpeople	1082	0,13
	Webcrawl	10000	1,30
	FOAF	9758	1,38
	Ontoworld	55619	9,20
	Wikipedia[3]	47054407	6882,20
Sintetizados	*Univ*(1, 0)	103074	17,27
	Univ(5, 0)	645649	108,28
	Univ(10, 0)	1316322	220,79
	Univ(20, 0)	2781322	468,68
	Univ(50, 0)	6888642	1163,49

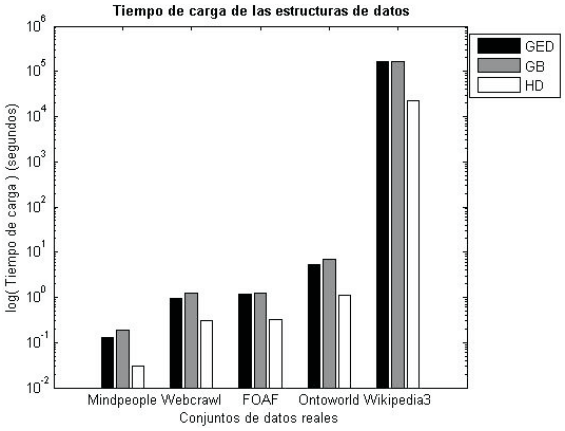

Figura 2. Tiempo de carga (segundos) de los conjuntos de datos reales.

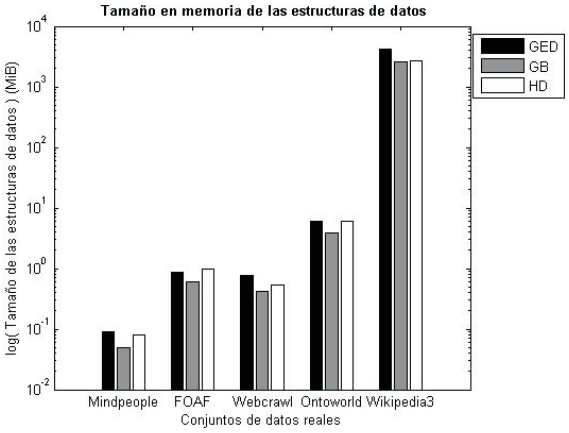

Figura 3. Tamaño (MiB) de las estructuras asociadas con los conjuntos de datos reales.

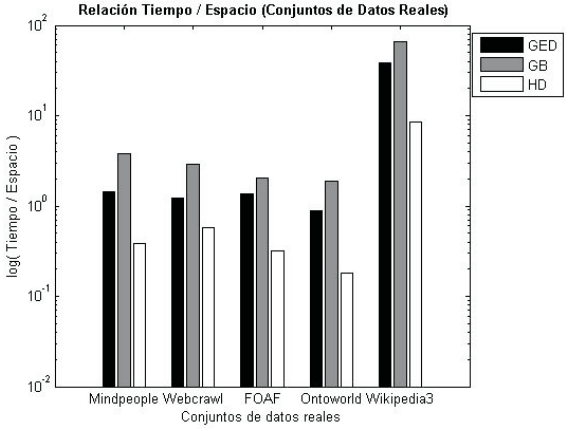

Figura 4. Relación tiempo de carga/espacio requerido (conjuntos de datos reales).

en *hash* son más complejos que los requeridos para indizar una estructura de datos ordenada.

La Figura 4 reporta los resultados obtenidos para cada representación y conjunto de datos real, con respecto a la relación tiempo de carga / tamaño de la estructura resultante (*time / space tradeoff*). Se debe destacar que, en todos los casos de prueba, esta relación favorece a la representación basada en HD con respecto a las otras dos representaciones. Esto significa que, bajo el enfoque basado en HD, el tiempo de carga de un conjunto de datos RDF puede reducirse significativamente utilizando sólo un poco más de memoria en la estructura de datos resultante.

En la Figura 5 se presentan los resultados obtenidos con respecto al tiempo de carga de los conjuntos de datos sintetizados en las estructuras de datos definidas para cada representación. Puede verse nuevamente que, en todos los casos de prueba, el tiempo de carga requerido por el modelo basado en HD es inferior al de los modelos basados en GED y GB. El tiempo de carga requerido por la representación propuesta es, en promedio, menor que 1/3 del tiempo de carga requerido por las otras dos representaciones. Este cociente va disminuyendo a medida que se incrementa el tamaño de los conjuntos de datos. El mejor caso se presenta con el conjunto de datos *Univ*(50, 0), en el que el tiempo de carga requerido por el modelo basado en HD fue aproximadamente 1/8 del tiempo de carga requerido por los otros dos modelos.

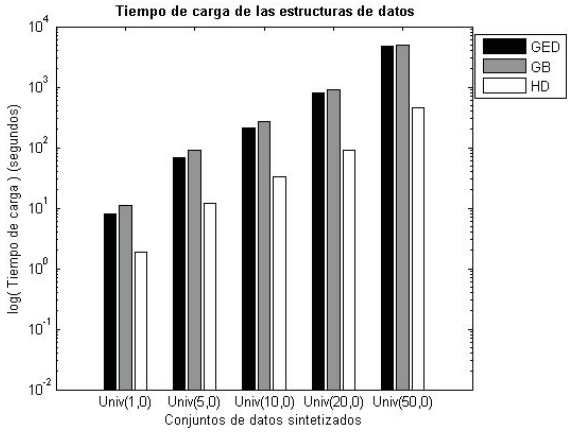

Figura 5. Tiempo de carga (segundos) de los conjuntos de datos sintetizados.

La Figura 6 muestra los resultados obtenidos con respecto al tamaño (MiB) de las estructuras de datos asociadas con los conjuntos de datos sintetizados. Nuevamente, el espacio de almacenamiento requerido por el modelo basado en HD es inferior al del modelo basado en GED, y ligeramente superior al espacio requerido por el modelo basado en GB. Como se mencionó anteriormente, esto se debe a que los índices requeridos por una estructura de datos basada en *hash* son más complejos que los requeridos para indizar una estructura de datos ordenada.

En la Figura 7 se reportan los resultados obtenidos para cada representación y conjunto de datos sintetizado, con respecto a la relación tiempo de carga / tamaño de la estructura resultante (*time/space tradeoff*). En todos los casos de prueba, esta relación favorece a la representación propuesta basada en HD con respecto a las otras dos representaciones.

La Figura 8 muestra los resultados con respecto al tiempo promedio de respuesta de las consultas sobre los conjuntos de datos sintetizados. En todos los casos, el tiempo de respuesta de las consultas sobre el modelo basado en HD es inferior al de los modelos basados en GED y GB. Los resultados obtenidos reflejan un comportamiento más estable de los algoritmos de evaluación de consultas sobre el esquema basado en HD que sobre los otros dos enfoques. Esto se debe a que el proceso de búsqueda sobre la estructura definida para HD (basada en *hash*) tiene complejidad $O(1)$; mientras que, bajo las otras dos representaciones (basadas en conjuntos ordenados), tiene complejidad $O(\lg |T|)$, equivalente al de la búsqueda binaria. Esta diferencia en complejidad tiene como consecuencia que el tiempo de respuesta de las consultas sea menos sensible al tamaño de los conjuntos de datos sobre el modelo basado en HD que sobre los otros dos modelos. De esta manera queda corroborado, de manera experimental, el comportamiento estable del algoritmo de evaluación de consultas sobre la representación basada en HD, el cual está asociado con la complejidad de la tarea de responder consultas. También queda comprobado, empíricamente, que el tiempo de respuesta de las consultas es menos sensible al incremento del tamaño de los conjuntos de datos en el modelo basado en HD que en las otras dos representaciones.

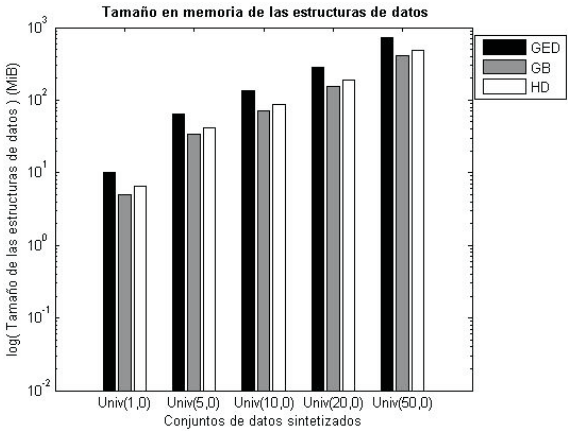

Figura 6. Tamaño (MiB) de las estructuras asociadas con los datos sintetizados.

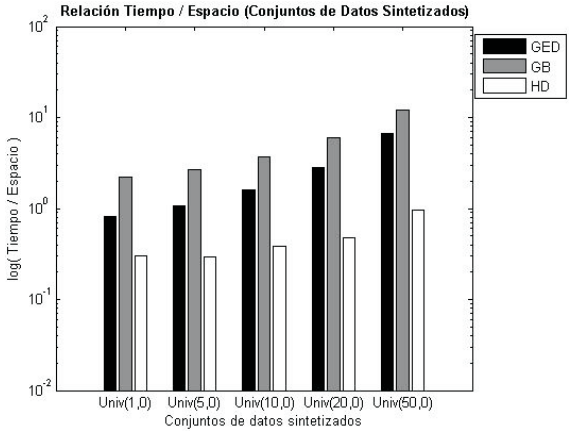

Figura 7. Relación tiempo de carga/espacio requerido (conjuntos de datos sintetizados).

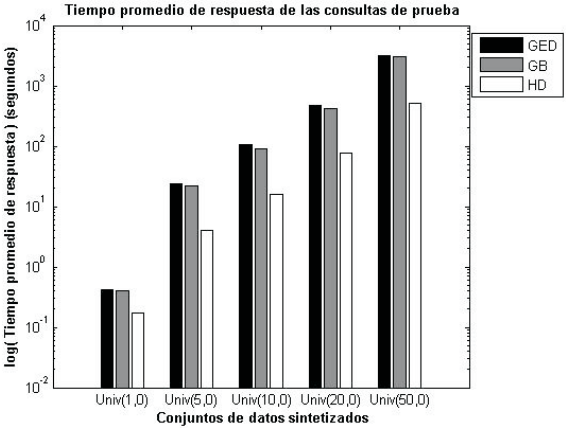

Figura 8. Tiempo promedio de respuesta (segundos) de las consultas.

Finalmente, se presentan comparaciones del enfoque propuesto con el *framework* Jena [15], desarrollado para la creación y manipulación de documentos RDF. La Figura 9 muestra los resultados con respecto al tiempo de carga de los conjuntos de datos sintetizados en las estructuras de datos. Se debe destacar que, en todos los casos, el tiempo de carga requerido por la representación basada en HD es menos de 1/4 del tiempo de carga requerido por Jena.

En la Figura 10 se reportan los resultados con respecto al tamaño (MiB) de las estructuras de datos asociadas con los conjuntos de datos sintetizados. Como puede notarse, el espacio de almacenamiento requerido por el modelo basado en HD es inferior al de Jena. El espacio utilizado por la representación basada en HD es menos de 2/3 del espacio requerido por Jena.

La Figura 11 presenta los resultados con respecto al tiempo promedio de respuesta de las consultas de prueba sobre los conjuntos de datos sintetizados. En todos los casos, el tiempo de respuesta de las consultas sobre el modelo basado en HD es inferior al de Jena.

4. Conclusiones y trabajo futuro

En este trabajo se analizaron las bondades del modelo basado en hipergrafos dirigidos (HD) para RDF, con el objetivo de ofrecer una alternativa para el almacenamiento eficiente de documentos RDF. En un hipergrafo dirigido la información sólo es almacenada en los nodos, y los hiperarcos sólo se encargan de preservar el rol de cada nodo y el concepto de dirección de los grafos RDF. Las estructuras de datos diseñadas para el almacenamiento de HD, basadas en hash, garantizan complejidad en tiempo constante para las operaciones de inserción y consulta de la estructura. Una vez planteada la representación y definidas las estructuras de datos, se estudió la complejidad en espacio de este enfoque para el almacenamiento de información y el impacto de esta representación sobre la tarea de responder consultas. Así mismo, se realizaron comparaciones con respecto a las representaciones existentes, específicamente grafos etiquetados dirigidos (GED) y grafos bipartitos (GB), para la evaluación de la calidad del modelo planteado.

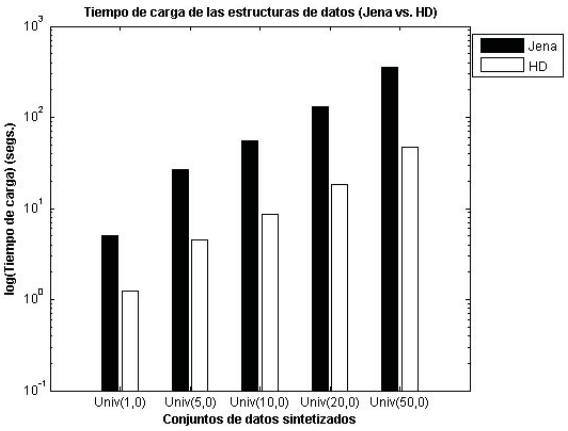

Figura 9. Tiempo de carga (segundos) de los conjuntos de datos sintetizados.

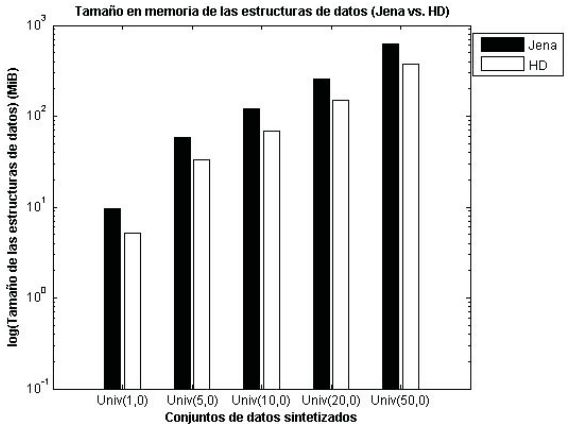

Figura 10. Tamaño (MiB) de las estructuras asociadas con los datos sintetizados.

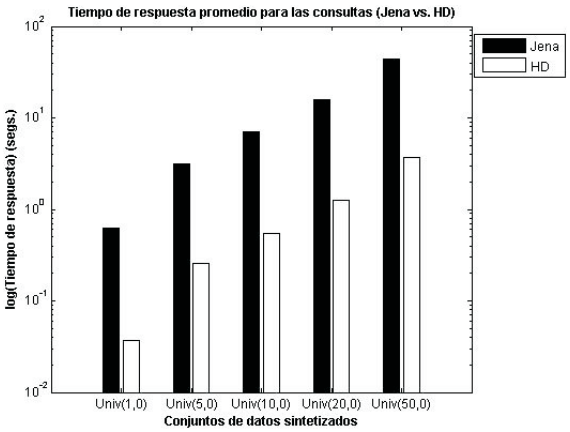

Figura 11. Tiempo promedio de respuesta (segundos) de las consultas.

Los resultados experimentales obtenidos demostraron empíricamente que, en todos los casos de prueba: (1) el tiempo de carga requerido por el modelo basado en HD es inferior al de los modelos basados en GED y GB, (2) los algoritmos propuestos de evaluación de consultas tienen mejor desempeño, en el caso promedio, utilizando el enfoque basado en HD, (3) la relación tiempo/espacio (*time/space tradeoff*) favorece a la representación basada en HD con respecto a los otros dos enfoques. Además, en casi todos los casos de prueba, el espacio de almacenamiento requerido por el modelo basado en HD es inferior al del modelo basado en GED, y ligeramente superior al espacio requerido por el modelo basado en GB. Esto puede atribuirse al proceso de indización de las estructuras asociadas con cada representación. Así mismo, la comparación realizada con el *framework* Jena [15] permitió comprobar, de manera experimental, que el enfoque propuesto requiere menos recursos (tiempo de carga, espacio de almacenamiento y tiempo de respuesta de consultas) que este sistema de administración para RDF.

Como trabajo futuro se propone considerar las siguientes extensiones al prototipo desarrollado: (1) desarrollar los algoritmos necesarios para proporcionar soporte para RDFS [10, 16] y la tarea de razonamiento semántico, (2) proporcionar soporte para consultas expresadas en el lenguaje SPARQL [17], ya que éste se ha convertido en el lenguaje estándar de consultas para RDF, (3) estudiar técnicas de optimización de consultas bajo este ambiente, y (4) realizar comparaciones de desempeño con respecto a otros sistemas existentes.

Referencias

1. Klyne, G. and Carroll, J. J. (2004). Resource Description Framework (RDF): Concepts and Abstract Syntax. Technical Report Recommendation, W3C.
2. Martínez Morales, A. A. (2008). Modelo basado en Hipergrafos Dirigidos para Resource Description Framework (RDF). Trabajo de Ascenso, Universidad de Carabobo, Venezuela.
3. Gutiérrez, C., Hurtado, C. A., and Mendelzon, A. O. (2003). Formal aspects of querying RDF databases. In Cruz, I. F., Kashyap, V., Decker, S., and Eckstein, R., editors, Proceedings of SWDB 2003, pages 293-307, Berlin, Germany.
4. Gutiérrez, C., Hurtado, C. A., and Mendelzon, A. O. (2004). Foundations of Semantic Web Databases. In Deutsch, A., editor, Proc. of PODS 2004, pages 95-106, Paris, France.
5. Hayes, J. (2004). A Graph Model for RDF. Master's thesis, Technische Universität Darmstadt, Department of Computer Science, Darmstadt, Germany. In collaboration with the Computer Science Department, University of Chile, Santiago de Chile.
6. Hayes, J. and Gutiérrez, C. (2004). Bipartite Graphs as Intermediate Model for RDF. In McIlraith, S. A., Plexousakis, D., and van Harmelen, F., editors, Proceedings of ISWC 2004, volume 3298 of Lecture Notes in Computer Science, pages 47-61, Hiroshima, Japan.
7. Martínez-Morales, A. A. and Vidal, M. E. (2007). A Directed Hypergraph Model for RDF. In Simperl, E., Diederich, J., and Schreiber, G., editors, Proceedings of KWEPSY 2007, volume 275 of CEUR Workshop Proceedings, pages 96-97, Innsbruck, Austria.
8. Baget, J.-F. (2005). RDF Entailment as a Graph Homomorphism. In Gil, Y., Motta, E., Benjamins, V. R., and Musen, M. A., editors, Proceedings of ISWC 2005, volume 3729 of Lecture Notes in Computer Science, pages 82-96, Galway, Ireland.
9. Zaniolo, C., Ceri, S., Faloutsos, C., Snodgrass, R. T., Subrahmanian, V., and Zicari, R. (1997). Advanced Database Systems. Morgan Kaufmann, San Francisco, USA, first edition.
10. Hayes, P. (2004). RDF Semantics. Technical Report Recommendation, W3C.
11. Yannakakis, M. (1981). Algorithms for Acyclic Database Schemes. In Proceedings of VLDB 1981, pages 82-94, Cannes, France.
12. Papadimitriou, C. H. and Yannakakis, M. (1997). On the Complexity of Database Queries. In Proceedings of PODS 1997, pages 12-19, Tucson, Arizona, USA.

13. Oren, E., Gerke, S., and Decker, S. (2007). Simple Algorithms for Predicate Suggestions using Similarity and Co-Occurrence. In Franconi, E., Kifer, M., and May, W., editors, Proceedings of ESWC 2007, volume 4519 of LNCS, pages 160-174, Innsbruck, Austria.

14. Guo, Y., Pan, Z., and Heflin, J. (2004). An Evaluation of Knowledge Base Systems for Large OWL Datasets. In McIlraith, S. A., Plexousakis, D., and van Harmelen, F., editors, Proc. of ISWC 2004, volume 3298 of LNCS, pages 274-288, Hiroshima, Japan.

15. Wilkinson, K., Sayers, C., Kuno, H., and Reynolds, D. (2003). Efficient RDF Storage and Retrieval in Jena2. In Cruz, I. F., Kashyap, V., Decker, S., and Eckstein, R., editors, Proceedings of SWDB 2003, pages 131-150, Berlin, Germany.

16. Brickley, D. and Guha, R. V. (2004). RDF Vocabulary Description Language 1.0: RDF Schema. Technical Report Recommendation, W3C.

17. Prud'hommeaux, E. and Seaborne, A. (2008). SPARQL Query Language for RDF. Technical Report Recommendation, W3C and Hewlett-Packard Laboratories, Bristol.

Recibido el 20 de Enero de 2010
En forma revisada el 30 de Mayo de 2011