OneQL: Una arquitectura basada en ontologías para consultar recursos eficientemente en la Web Semántica

Tomás Lampo, Edna Ruckhaus, Javier Sierra, María Esther Vidal, Amadís Martínez

Universidad Simón Bolívar

Motivación del Problema

1. **Tendencia en la Web Semántica:** más documentos RDF publicados y con mayor cantidad de datos. Necesidad de probar OneQL con datasets de gran tamaño (44 millones de tripletas) y así realizar consultas más interesantes.
2. **Experimentos previos:** dataset de las votaciones en el Congreso de USA (4MB y 67.392 tripletas) y consultas hasta 7 patrones en el WHERE.

Ejemplo previo Seleccionar todas aquellas leyes y sus títulos donde 'Nay' sea el ganador y que al menos exista un votante que haya votado por la misma opción que el votante 'L000174'

Consulta

```
PREFIX vote: <tag:govshare.info,2005:rdf/vote/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX people: <http://www.rdfabout.com/rdf/people/>
SELECT ?E ?T FROM <http://example.org/votes>
WHERE { ?E vote:winner ' Nay ' . ?E dc:title ?T .
?E vote:hasBallot ?I . ?I vote:option ?X .
?J vote:option ?X . ?E vote:hasBallot ?J .
?J vote:voter 'people:L000174' }
```

¿Qué es OneQL?

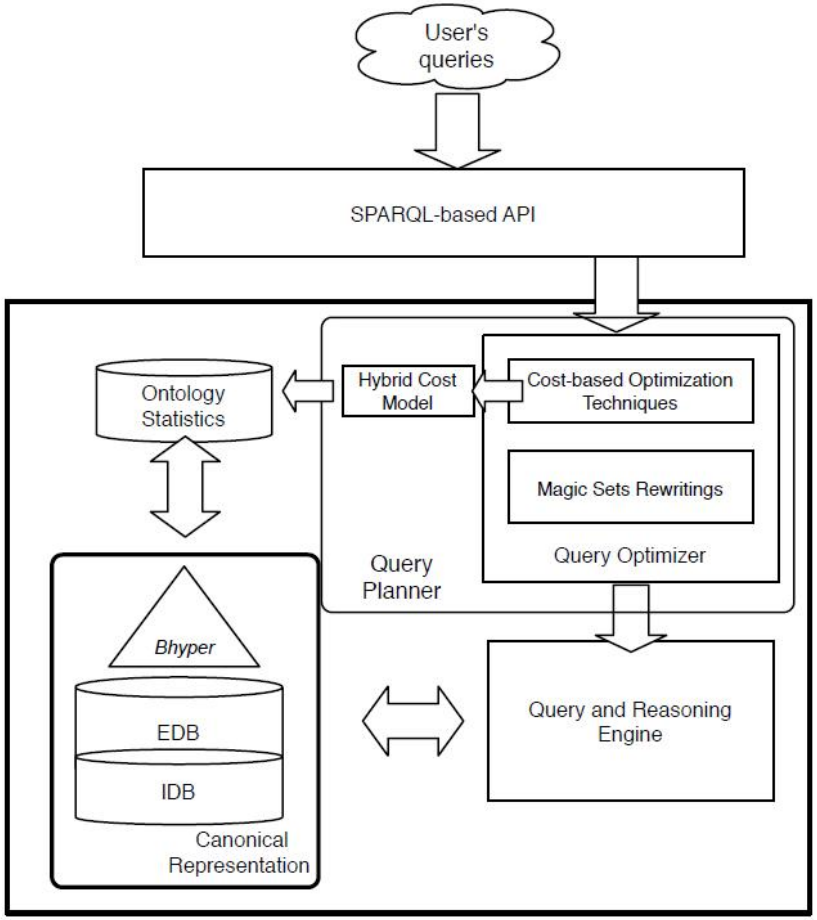
OneQL

Proporciona técnicas de optimización y evaluación de *queries* SPARQL para manejar grandes documentos RDF/RDFS y *queries* complejos. Implementado en Datalog, Prolog.

Eficiencia depende de:

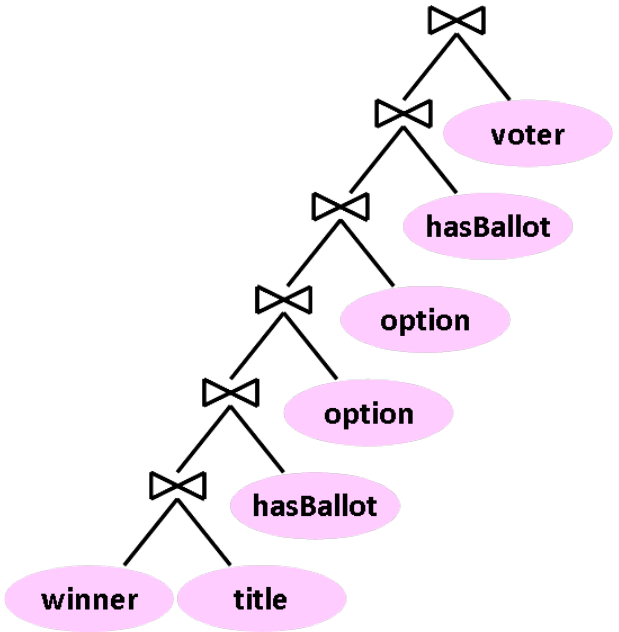
- **Ténicas de optimización y evaluación:** estimación de costo por modelo de costo y muestreo; y búsqueda de planes de cualquier forma lineales o *bushy* (algoritmo Simulated Annealing)
- **Bypher:** Representación de documentos RDF basada en hipergrafos dirigidos. Permite el acceso a tripletas que compartan el mismo sujeto y predicado, o el mismo predicado y objeto: $subject(S, P, Lo)$ y $object(O, P, Ls)$ definidos por extension

OneQL: Arquitectura

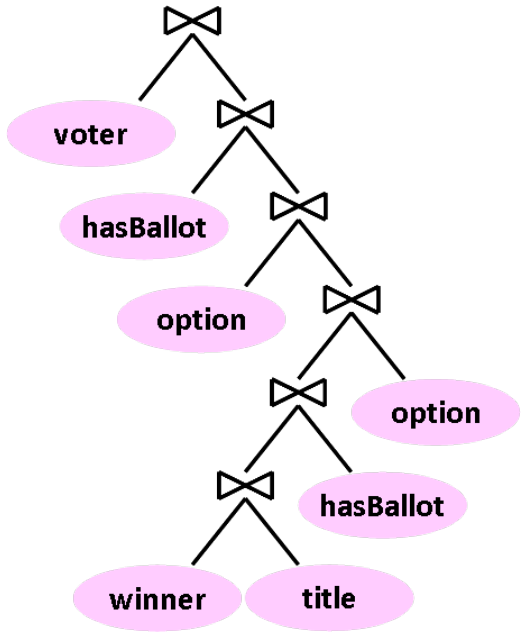


OneQL: Optimización SPARQL *Queries*

Left-linear tree (8466 s)



Bushy tree (122 s)



OneQL Optimización: Simulated Annealing

Algoritmo Simulated Annealing

- Aleatorio, realiza cambios aleatorios en cada etapa sobre el espacio de búsqueda de planes *bushy*
- Objetivo: proporcionar un buen plan (no necesariamente el mejor) en un tiempo aceptable.
- En cada etapa, un plan de ejecución del *query* es creado y luego sucesivas transformaciones son aplicadas
- Probabilidad de transformar un plan p en uno p' es $P(p, p', T)$, donde: T refleja el número de etapas ejecutadas y aunque $cost(p') > cost(p)$, P puede ser distinto de 0.
- Condición de parada: $T = 0$.

Objetivo

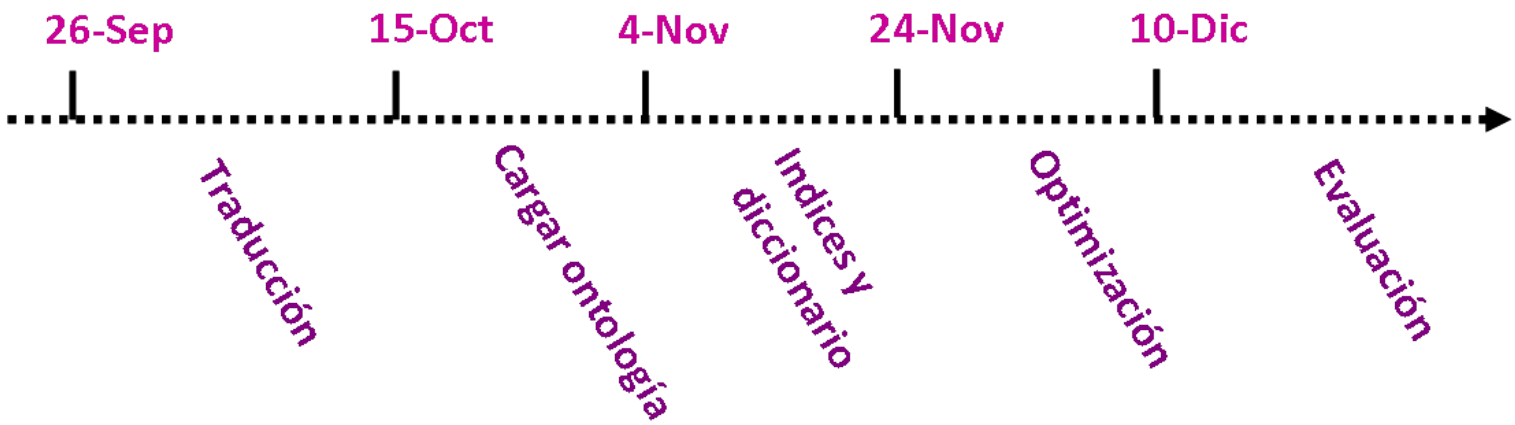
Objetivo del proyecto

Estudiar el comportamiento de OneQL con datasets de gran tamaño como: YAGO (44 millones de tripletas)

Retos del proyecto

Experimentar con datasets muy grandes:

1. Descargar Yago y traducir los documentos RDFS a formato OneQL
2. Cargar la ontología a OneQL y crear el diccionario
3. Generar los índices Bypher de sujetos y objetos y el diccionario
4. Optimización de consultas
5. Evaluación de consultas



Retos del proyecto: traducir a OneQL

Descargar Yago y traducir los documentos RDFS a formato OneQL:

1. Compilar el traductor RDF a OneQL (8-Octubre)
2. Ejecutar para los 143 documentos (19GB) el Traductor utilizando la maquina virtual de Java de 64 bits y 16 GB de RAM.
3. Fallas del Traductor: caracteres no reconocidos por la aplicación en los documentos y falta de memoria
4. Se creó un programa en Python para identificar los patrones de caracteres que el Traductor no reconocía en estos y eliminarlos.
5. Se lograron traducir 118 documentos, los que tenían un tamaño mayor a 300 MB no pudieron ser traducidos. (15-Octubre)

Retos del proyecto: cargar ontología y crear diccionario

Cargar la ontología en OneQL y crear el diccionario de Yago (Analyze para hallar las estadísticas de los datos).

1. Consolidar documentos traducidos a OneQL
2. Eliminación de posibles duplicados
3. Carga de la ontología: problemas con las tripletas, no podían ser entendidas por Swi-Prolog (tenían caracteres que hacían que el interpretador se confudiera). Aprox. 44 millones de tripletas y muchas con errores de este tipo.
4. Creación de programas en AWK y Python para arreglar las tripletas hasta lograr cargarlas en Prolog. (28-Octubre)
5. Generar diccionario: problemas por falta de memoria, se necesitaban más de 14 GB. Se ejecutó por más de 5 días, se decidió terminarla y crear el diccionario de otra forma. (4-Noviembre)

Retos del proyecto: crear índices y diccionario

Creación de los índices Bypher y el diccionario de Yago:

1. Intento de creación de los índices Bypher con la ontología completa (falló por falta de memoria)
2. Separación de la creación de los índices en dos instrucciones, una para crear el índice de sujetos y otra para el de objetos.
3. Indice de sujetos: particionando el archivo de la ontología en grupos de 5 millones de tripletas cada uno (9 archivos) y a c/u se le creó el índice.
4. Indice de objetos: particionando el archivo de la ontología en grupos de 5 millones, 1 millón y 500 mil tripletas (58 archivos) y con c/u se creó el índice. (22-Noviembre)
5. Se agruparon los fragmentos de ambos índices.
6. Creación del diccionario OneQL: a partir de los índices de sujetos y objetos. Utilizado en la fase de optimización. (24-Noviembre)

Retos del proyecto: optimización

Traducción de consultas (SPARQL - OneQL) sobre Yago y optimización de las mismas:

1. Traducción de consultas: 40 consultas de SPARQL a OneQL. Primer grupo de consultas ejecutado en JENA, resultaron ser muy sencillas. Se escogió el último grupo de 15 consultas para realizar los experimentos.
2. Optimización de consultas: (problemática)
    1. Faltaban datos necesarios para el proceso de optimización
    2. Con la temperatura (700) e iteraciones (20) por defecto del SA fallaba el proceso por falta de memoria
    3. Se tuvo que cargar sólo con la fración de la ontología utilizada por las consultas
    4. Se modificaron los parámetros del SA hasta finalmente lograr optimizar 12 consultas. (10-Diciembre)

Retos del proyecto: evaluación

Evaluación de las consultas sobre Yago:

1. Se evaluaron las 12 consultas optimizadas y sin optimizar, utilizando y sin utilizar los índices Bypher
2. Se revisaron los logs de la evaluación de estas consultas, pero ninguna arrojó respuestas y el tiempo de ejecución sólo comprendía la carga de la ontología y de los índices.
3. Se debe revisar el evaluador para encontrar la falla.

Solución Propuesta

Estudiar con 15 consultas el comportamiento de OneQL con Yago:

1. Carga y traducción del dataset Yago
2. Creación las estructuras Bypher para indexar las tripletas de Yago
3. Creación el diccionario de OneQL para Yago
4. Optimización de 12 consultas
5. Intento de evaluación

Resultados

Dataset, Indices y configuración de las consultas

- Dataset: Yago, tamaño: 4 GB, tripletas: aprox. 44 millones
- Indice de sujetos: tamaño: 3,3 GB, tripletas: 35.004.239
- Indice de objetos: tamaño: 1,35 GB, tripletas: 7.989.604
- 15 consultas con 13 a 35 patrones en el WHERE.

Métricas

- Costo estimado del plan
- Simulated Annealing: 5 iteraciones y temperatura inicial de 10; y 20 iteraciones y temperatura inicial de 400.

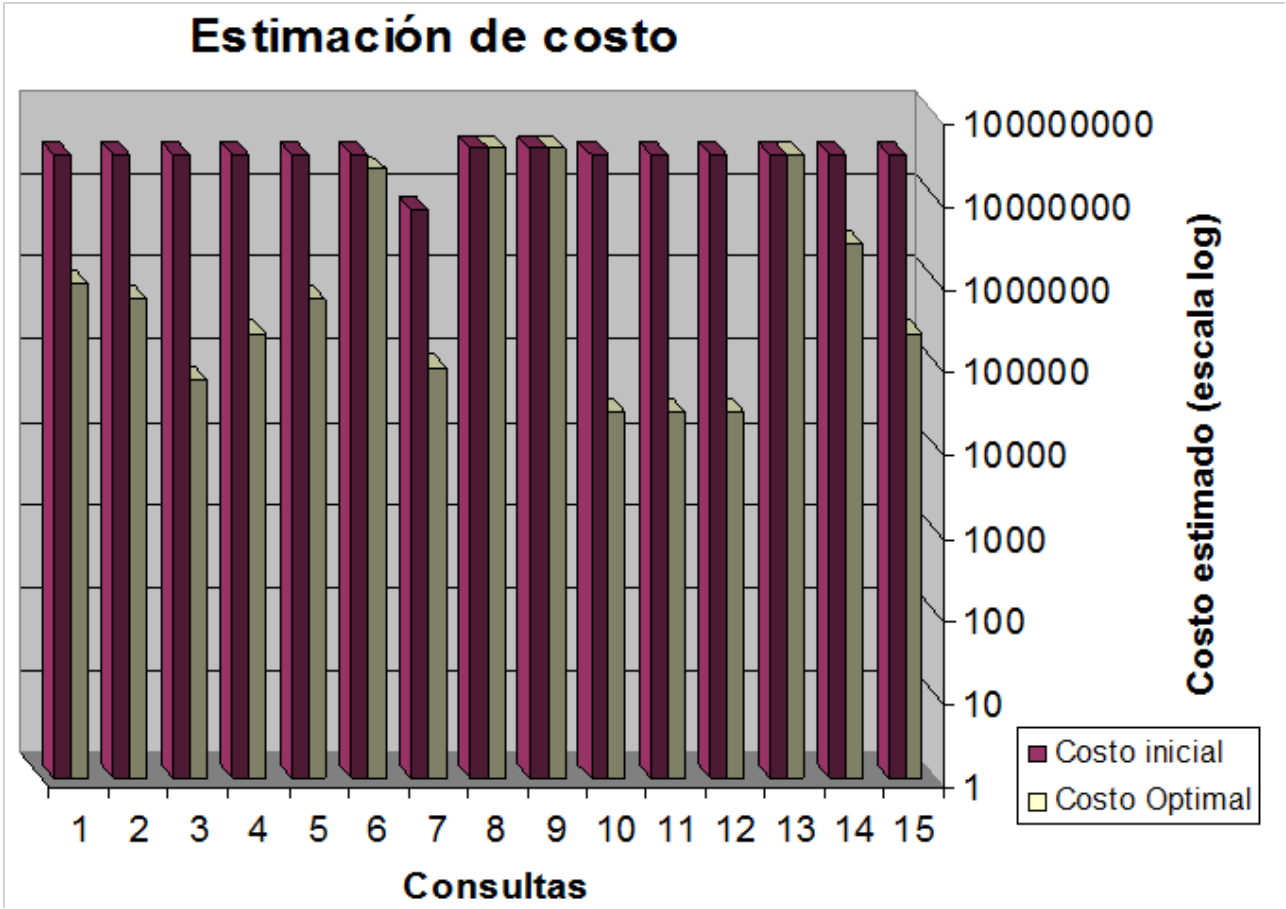
Resultados: cardinalidad y número de valores distintos

Propiedad	Tripletas	Valores Sujetos	Valores Objetos
object	8.299.663	7.073.574	1.869.809
hasFamilyName	1.138.612	569.306	250.544
hasGivenName	1.137.516	568.758	58.368
locatedIn	120.312	60.125	4.520
bornIn	72.374	36.187	16.818
actedIn	57.670	14.089	27.479
directed	47.444	5.859	23.710
hasWonPrize	45.280	16.692	4.409
livesIn	29.420	14.707	6.508
influences	17.404	3.029	4.958
isMarriedTo	8.416	4.208	4.121
hasCurrency	700	347	265
discovered	174	75	83

Resultados: consultas

Consulta	Patrones	Temperatura	Iteraciones	Costo inicial	Costo optimal
1	19	10	5	3.3019.808	915.943
2	14	10	5	3.3019.803	595.216
3	18	10	5	3.3019.818	62.614
4	23	400	20	3.3019.819	229.459
5	15	10	5	33.019.803	595.216
6	18	10	5	33.019.810	21.231.308
7	13	10	5	6.831.699	87.841
8	35	X	X	37.574.276	37.574.276
9	35	X	X	37.574.276	37.574.276
10	24	400	20	33.019.819	26.053
11	24	400	20	33.019.819	26.053
12	24	400	20	33.019.819	26.053
13	26	X	X	33.019.822	33.019.822
14	26	400	20	33.019.822	2.680.911
15	25	400	20	33.019.819	229.459

Resultados: optimización



Conclusiones y Trabajo Futuro

1. Debido a que OneQL no posee un servicio de consulta sobre el Cloud de Linked Data, si se desea experimentar con algún dataset se debe **tomar en cuenta el tiempo que lleva y los recursos que se necesitan para preparar los datos** para que éstos puedan ser entendidos por OneQL.
2. Para 12 de 15 consultas el optimizador **encontró un plan cuya estimación de costo es menor que la del plan inicial**.
3. Se debe **comprobar** mediante la evaluación de consultas que los **planes escogidos por el optimizador son mejores que los planes iniciales**.
4. Se debe realizar un estudio para **hallar las ventajas de OneQL vs Jena y RDF-3X** para datasets de gran tamaño