ASPIA, un Repositorio Federado como Base Semántica de Wikipedia

Marcia Kaminose, Francisco Blanco, Agustin Parmisano, Diego Torres, and Alicia Díaz

Laboratorio de Investigación y Formación en Informática Avanzada, LIFIA.
Facultad de Infórmatica, UNLP,
{yukari.kaminose, blanco.jose.francisco, agustinparmisano}@gmail.com
{diego.torres, alicia.diaz}@lifia.unlp.edu.ar

Abstract. Wikipedia es una gran enciclopedia en la que colaboran miles de usuarios alrededor de todo el mundo. Por otro lado se encuentra DBpedia, un repositorio semántico que extrae información de Wikipedia. La información semántica extraída por DBpedia facilita la búsqueda de información, la cual a veces no es posible obtener navegando por Wikipedia. Sin embargo, la información que contiene DBpedia no representa la totalidad del contenido de Wikipedia. En este artículo se verá cuál puede ser la razón de esta falta de datos, y cómo se podría evitar este problema en una búsqueda, implementando un repositorio federado.

Keywords: Repositorio federado, DBpedia, Wikipedia, Yago

1 Introducción

La Web Semántica ha ido tomando mayor importancia durante los últimos años, ya que permite mejoras en la navegación y la búsqueda de contenido en la Web. La información de la Web Semántica es extraída, en gran medida, de la información producida por la Web Social. Un buen ejemplo es el caso de DBpedia[1], una base semántica de conocimiento que obtiene el contenido de los infobox y markups de Wikipedia[1]. En DBpedia es posible realizar consultas semánticas de una forma sencilla, como por ejemplo obtener aquellas "personas que nacieron en París después del año 1980".

La base semántica de DBpedia está conformada por recursos relacionados entre sí por propiedades semánticas. De esta forma, la cualidad semántica de DBpedia permite deducir información que no está presente en Wikipedia. Por ejemplo, la consulta anterior produce nombres de personas que no pueden obtenerse navegando en Wikipedia desde la página de la ciudad París. Estas diferencias generan una brecha semántica entre DBpedia y Wikipedia.

En Improving Wikipedia with DBpedia[10], los autores presentan una alternativa para poder suplir esta diferencia de información utilizando DBpedia para mejorar el contenido de Wikipedia. El trabajo de Torres et al. está basado en

[1] http://www.wikipedia.org/

representar en Wikipedia una relación semántica p de DBpedia, respetando las convenciones y la estructura de Wikipedia. Para ello, presentan el Path Index Algorithm (PIA). PIA selecciona aquellos pares de páginas que se encuentran relacionadas en DBpedia por p y que también lo estan en Wikipedia por un camino navegacional (navegando por medio de links entre páginas). Luego, utiliza esta selección para aprender y detectar la forma en que la comunidad de Wikipedia mejor representa estas relaciones en la misma Wikipedia. Al finalizar, PIA retorna la mejor forma general para representar la relación semántica de DBpedia en Wikipedia. Como ejemplo, los autores muestran que la mejor forma de representar la relación *birthPlace* entre Francia y filósofos franceses es mediante el camino navegacional: `France / Category:France / French_People / French_People_by_Occupation / French_Philosophers / <philosopher>`.

Sin embargo, el conjunto de elementos que retorna DBpedia para ejecutar PIA es notoriamente pequeño: solamente 34 filósofos franceses como fuente de aprendizaje de PIA. Como fue indicado anteriormente, DBpedia construye su base semántica extrayendo información desde Wikipedia utilizando los infobox. Sin embargo, existen otras bases semánticas que obtienen información de Wikipedia utilizando diferentes reglas de extracción de conocimiento[9, 5, 2].

El objetivo de este artículo es extender la fuente de información semántica que utiliza como entrada el algoritmo PIA, combinando diferentes bases semánticas basadas en Wikipedia. La combinación de diferentes bases semánticas es un campo de investigación con un largo recorrido[6, 3]. Aunque las fuentes de extracción de conocimiento coincidan, la representación de las ontologías generalmente difieren en nombres y también en significado. El desafío de combinar diferentes bases de conocimiento incluye la combinación de ontologías. En ellas se suceden problemas como diferencias en los nombres de las propiedades (`lugarDeNacimiento` por `birthPlace`), categorizaciones diferentes (Paris type City, Paris type Capital) o dominios diferentes (birthPlace entre Personas y Lugares; birthPlace entre Personas y Ciudades), por nombrar algunos.

El aporte de este artículo es mostrar como es posible construir un repositorio federado, el cual llamamos Abstract Semantic PIA (ASPIA), que combine diferentes bases semánticas basadas en Wikipedia en una sola. Para ello, el artículo detalla un prototipo en funcionamiento que permite combinar DBpedia con Yago, agregando una capa semántica de abstracción, e incluye un caso testigo en la cual se verifica el resultado de ejecutar PIA utilizando nuestra propuesta, abriendo la posibilidad de continuar con este enfoque.

El artículo está organizado de la siguiente forma. En la sección 2 se describe la motivación y un ejemplo. Luego, en la sección 3 presentamos el repositorio federado ASPIA. En la sección 4 presentamos el prototipo con la implementación de ASPIA. Luego, realizamos una prueba del prototipo en la sección 5. En la sección 6 describimos otros trabajos relacionados y para finalizar, en la sección 7, las conclusiones y trabajos a futuro.

2 Motivación

DBpedia transforma la información de Wikipedia en una base semántica formada por recursos relacionados entre sí por propiedades semánticas. La transformación de Wikipedia a DBpedia se realiza mediante reglas de mapeo definidas en forma colaborativa[2]. En términos generales, DBpedia mapea cada página de Wikipedia a un recurso de DBpedia, y toma de los infoboxes las propiedades que relacionan los recursos[1].

La Tabla 1 muestra la traducción realizada por DBpedia para el caso del artículo del filósofo francés Montesquieu[3]. En la columna *Infobox* se indica el detalle del infobox *philosopher* y los valores para indicar el lugar de nacimiento (*birth_place*), mientras que en la columna *DBpedia* se muestran las ternas generadas como producto del mapeo del infobox. Para una mayor claridad, se hizo uso de los *PREFIX*.

Infobox	DBpedia
{{ Infobox philosopher (1) |birth_place =..., [[France]] (2) }}	PREFIX db-owl:<http://dbpedia.org/ontology/> PREFIX :<http://dbpedia.org/resource/> PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#> :Montesquieu rdf:type db-owl:Philosopher. (3) :Montesquieu db-owl:birthPlace :France. (4)

Table 1. Infobox de Wikipedia y su correspondiente mapeo en DBpedia para Montesquieu.

Según las reglas de mapeo establecidas por DBpedia, el tipo del infobox *philosopher* genera que el recurso dentro de DBpedia *http://dbpedia.org/resource/Montesquieu* sea de tipo *Philosopher*, generando la terna (3), mientras que la propiedad del lugar de nacimiento del infobox *birth_place* se corresponde con la propiedad del lugar de nacimiento en DBpedia *birthPlace*, dando lugar a la terna (4).

Gracias a este mecanismo de mapeo, es posible realizar una consulta semántica que nos permita obtener todos los filósofos franceses que se encuentran en DBpedia. La siguiente consulta en lenguaje SPARQL nos permite obtener dicha información:

```
PREFIX db-owl:<http://dbpedia.org/ontology/>
PREFIX :<http://dbpedia.org/resource/>

SELECT :France ?philosopher WHERE {
 ?philosopher a db-owl:Philosopher .
 ?philosopher db-owl:birthPlace :France .}
```

[2] http://mappings.dbpedia.org/
[3] http://en.wikipedia.org/wiki/Montesqieu, el día 09 de Marzo del año 2012.

Esta consulta realizada en el endpoint de DBpedia[4] retorna 34 resultados[5]. De acuerdo a Torres et al., la comunidad de usuarios de Wikipedia utiliza la categoría *French Philosophers* para agrupar a los filósofos franceses. Al navegar en Wikipedia a dicha categoría notamos que la cantidad de artículos categorizados son 346[6]. Es decir, la comunidad de Wikipedia considera que existen 346 filósofos franceses en Wikipedia, que son muchos más que aquellos que enumera DBpedia.

Esto se debe a que muchos de los artículos en la categoría *French_Philosophers* no poseen un infobox con la información referida al tipo *philosopher* o que algunos de estos no tienen asociado el valor del lugar de nacimiento *birth_place*.

Como alternativa a DBpedia, existen otras bases de conocimiento. Tomemos como referencia a Yago. La principal diferencia de ésta con DBpedia es la forma en la que realiza la traducción de información desde Wikipedia. Por ejemplo, Yago utiliza como una de sus fuentes las categorías que utiliza Wikipedia para organizar sus artículos[9], dándole otra perspectiva con respecto a DBpedia. Además, actúa en conjunto con WordNet[7], un diccionario léxico que le ayuda a interpretar dichas categorías, o extraer categorías de ella.

Una cosa para destacar de la información obtenida en Yago es que comparte los mismos IDs de los recursos con DBpedia. Por otro lado, en Yago podemos hallar que la clase que definiría a los filósofos franceses puede verse como la combinación de las clases filósofos y franceses pertenecientes a DBpedia, tomando el nombre de *frenchphilosophers*, lo que dejaría en claro su equivalencia semántica. Por lo tanto se desprende que existen elementos en la ontología de Yago que son comunes a la definida por DBpedia.

La consulta SPARQL equivalente a la realizada en DBpedia para obtener el listado de filósofos franceses en Yago es:

```
PREFIX yago:<http://yago-knowledge.org/resource/>

SELECT yago:France, ?philosopher WHERE {
 ?philosopher a yago:wikicategory_French_philosophers .
 ?philosopher yago:hasWikipediaUrl ?link .}
```

Al igual que en la consulta realizada en DBpedia, la tupla devuelta sólo corresponde a los filósofos franceses, por lo especificado en la cláusula *where*. Y es en este bloque donde la diferencia es notoria entre ambos repositorios, ya que no se verifica que el elemento buscado en Yago sea filósofo y luego que sea francés, al estar esta información unida en la propiedad *yago:wikicategory_French_philosophers*. Además, lo que podemos notar en esta última consulta es que se debe cerciorar que dichos artículos posean su conexión con Wikipedia que, como podemos observar, es su URL.

[4] http://dbpedia.org/sparql
[5] Consulta realizada el día 10 de Marzo del año 2012.
[6] Consulta realizada en Wikipedia el día 7 de Marzo del año 2012.

La cantidad de filósofos franceses de Wikipedia obtenidos por medio del endpoint de Yago[7] luego de realizar la consulta fue 304[8], de los cuales la mayor parte fueron corroborados manualmente. Podemos notar una mayor cantidad de filósofos dentro de la base de conocimiento de Yago, esto se debe a, como se dijo anteriormente, las diferentes formas de extraer información que tiene Yago, por ende, muchos de los elementos de Wikipedia que no poseen infobox igual podrían encontrarse en él.

A partir de estos ejemplos, podemos decir que el utilizar una sola base como única fuente de información nos puede generar una escasez de datos. El desafío entonces involucra definir un repositorio semántico federado que permita combinar diferentes fuentes de información semántica detrás de una estructura abstracta. En este repositorio semántico federado debe ser posible obtener, mediante una única consulta semántica, elementos relacionados en las diferentes bases particulares, ocultando al usuario que realiza la consulta el orígen de los mismos.

En la siguiente sección detallaremos el framework de federación de repositorios semántincos llamado ASPIA.

3 Repositorio federado: ASPIA

Nuestro enfoque consiste en crear un repositorio federado que combine la información de diferentes repositorios basados en Wikipedia, para que puedan ser utilizadas como fuente para proveer a PIA de información. La ventaja que obtenemos a partir de este enfoque está directamente relacionada con la definición de una base de datos federada.

Una base de datos federada es aquella que unifica diferentes bases de datos autónomas. Esto quiere decir que cada una de las bases de datos se manejan de forma independiente, con su propio conjunto de datos, pero permiten que todos o algunos de sus datos sean compartidos. Entonces, una base de datos federada no es una base de datos en sí, sino una colección de ellas, las cuales están dispuestas a proveer información a diferentes usuarios[8, 4].

Entonces, partiendo de la definición de lo que es una base de datos federada, podemos decir que la ventaja que obtenemos de un repositorio federado es el contar con diferentes fuentes de información para recolectar la mayor cantidad de datos, y evitar posibles problemas de insuficiencia de datos. Sin embargo, esta solución nos puede traer otra clase de problema, como la redundancia de datos, ya que los repositorios trabajan de forma independiente y no llevan un control de la información que tienen los otros repositorios. Además, al manejar cada uno su propio conjunto de datos, estos podrían no estar estandarizados, por lo que puede que una misma información tenga diferentes formas de representarse para las diferentes bases.

Para mitigar los problemas enunciados, se define una capa semántica de abstracción, Abstract Semantic PIA (ASPIA), la cual se encarga de brindar in-

[7] http://lod.openlinksw.com/sparql
[8] Consulta realizada en Yago el día 10 de Marzo del año 2012.

formación a PIA. Para obtener dicha información, ASPIA recibe una consulta semántica para poder ejecutarla en los diferentes repositorios de los cuales queremos extraer información. Uno de los problemas que surgen a partir de esto es que no todas las consultas van a ser las mismas para los diferentes repositorios, o no todos los datos devueltos por los repositorios van a ser compatibles con PIA. Por esta razón, ASPIA y su capa de abstracción deben ser capaz de transformar una consulta específica para que sea posible ejecutarla en las diferentes bases semánticas. Además de, una vez generados los resultados de las consultas, poder eliminar aquellos datos que se encuentren repetidos, y convertirlos al estándar definido por PIA.

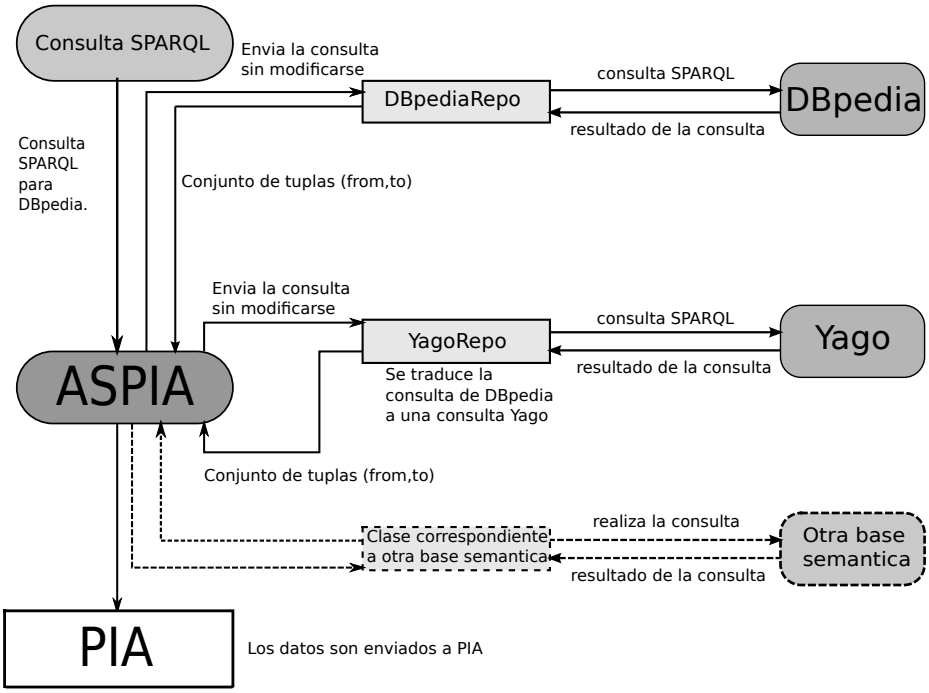

Fig. 1. Muestra el flujo de datos desde DBpedia y Yago hasta llegar a PIA.

Como muestra la Figura 1, la capa de abstracción recibe una consulta semántica apta para ser ejecutada en DBpedia. Dicha consulta puede no ser comprendida por otras bases semánticas directamente, entonces para evitar este problema se traduce la consulta a un formato comprensible por cada base semántica. Un ejemplo es el mostrado en la misma Figura 1, donde una consulta en formato de DBpedia es traducida y enviada a Yago. Estos resultados son devueltos a ASPIA para que éste pueda eliminar aquellos resultados repetidos y evitar la

redundancia. Al completar esta tarea, debe transformar los datos a un formato entendible para PIA, para finalmente enviárselos.

En este artículo proponemos crear una capa de abstracción sobre las distintas fuentes de conocimiento semántico, y en un principio estaremos trabajando solo con DBpedia y Yago. Los datos devueltos por las distintas bases semánticas son procesados por el algoritmo de búsqueda con el fin de filtrar los resultados repetidos y poder armar con ellos una estructura de datos de manera apta para enviarlos hacia PIA. Un ejemplo de una estructura válida sería:

```
DBpedia -> {(France, Michael_Foucault), (France, Jean_Paul_Sartre)}
Yago -> {(France, Michael_Foucault), (France, Monstesqieu)}
```

4 Implementación

Para poder llevar a cabo la implementación de la capa de abstracción, hemos implementado un algoritmo haciendo uso del lenguaje JAVA. Como se muestra en la Figura 2, contamos con una clase ASPIA, encargada de realizar la abstracción de datos, diferentes clases Repo que representan a los distintos repositorios, en este caso DBpediaRepo y YagoRepo, la clase RepoConnector, que es la responsable de efectuar la consulta en el repositorio correspondiente utilizando las APIs provistas por cada uno, y clases que heredan de YagoTranslator, encargadas de transformar una consulta DBpedia a una apta para Yago definiento un mapping para los valores posibles.

En la Figura 3 se muestra un diagrama de secuencia simplificado de cómo ASPIA lleva a cabo la tarea de extracción de datos en las diferentes ontologías en las cuales pretendemos buscar.

El algoritmo ASPIA toma como parámetro una consulta escrita para DBpedia y delega la consulta a las clases DBpediaRepo y YagoRepo, las cuales a su vez están asociados con la clase RepoConnector. Los resultados de las consultas ejecutadas por RepoConnector le son devueltas a DBpediaRepo y YagoRepo según corresponda, para luego pasar estos resultados a ASPIA. El inconveniente en este caso es que, como se vió en los ejemplos de las consultas anteriores, las consultas de DBpedia y Yago difieren en su sintaxis, por lo que la clase YagoRepo debería estructurar la consulta a un formato válido antes de enviársela al conector, haciendo uso de los traductores.

Estos traductores reciben parte de la consulta escrita para DBpedia, el bloque `where` comprendido entre {}. Cada uno de estos traductores internamente toman cada condición del bloque recibido y, mediante una comparación, determinan si al menos una parte del bloque puede ser traducido. En caso negativo, el bloque es descartado y tomado por otro traductor, que vuelve a realizar la misma comprobación. Pero en caso de determinar que la traducción es posible, tomará aquella o aquellas condiciones que este traductor comprende, para transcribirlas a una o unas condiciones que el repositorio de Yago sea capaz de entender. Este proceso se repite para todas las condiciones, hasta que el bloque haya pasado por todos los traductores. Una vez que el bloque esté traducido, se arma una nueva

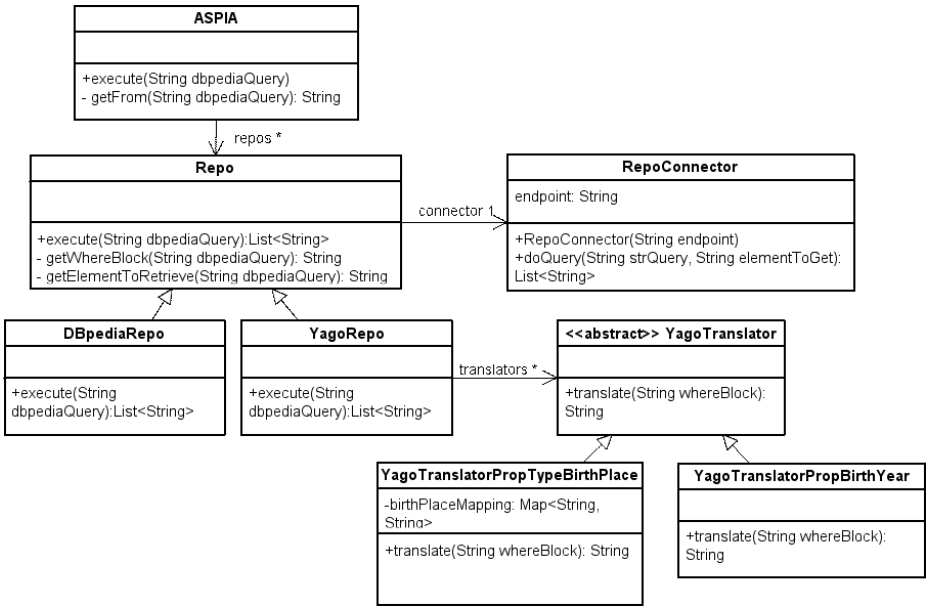

Fig. 2. Diagrama de clases del algoritmo ASPIA.

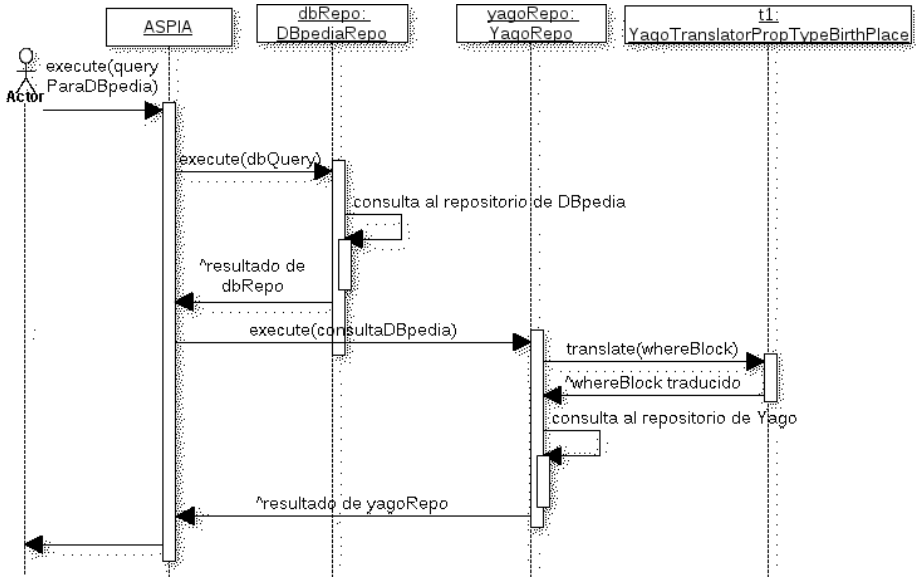

Fig. 3. Diagrama de secuencia del algoritmo ASPIA.

consulta con el nuevo bloque, le es pasada al RepoConnector correspondiente a Yago, y su resultado, a YagoRepo.

Una vez que se hayan realizado las consultas, los datos devueltos se estandarizan en cada clase que representa al repositorio para que sean compatibles con PIA. Finalmente, los resultados estandarizados le son pasadas a ASPIA, que se encarga de evitar la redundancia, eliminando aquellos datos que se encuentran repetidos, es decir, si alguno de los recursos ya se encontraba en el conjunto de datos de ASPIA, este no volverá a incluirse al conjunto final de resultados de ASPIA. Con los datos restantes, se arma un conjunto de tuplas entendibles por PIA.

5 Caso Testigo

Se quiere demostrar que en ciertos casos la información contenida en DBpedia sobre algún elemento no es suficiente para poder realizar evaluaciones con PIA. Para poder demostrarlo, se trabaja sobre el problema de los filósofos franceses. Sabiendo que la cantidad de filósofos franceses existentes en DBpedia es menor a la existente en Wikipedia, se compararan dos ejecuciones distintas de PIA y se analizan los resultados.

Para poder llevar a cabo nuestra evaluación hemos utilizado el framework ASPIA, con el fin de generar una base de datos con las tuplas *from - to*. Como parámetro de entrada seleccionamos la consulta con el formato de DBpedia:

```
SELECT <http://dbpedia.org/resource/France> ?philosopher
WHERE {
 ?philosopher a <http://dbpedia.org/ontology/Philosopher> .
 ?philosopher <http://dbpedia.org/ontology/birthPlace>
<http://dbpedia.org/resource/France> .}
```

Se han ejecutado las consultas en sus respectivas bases semánticas al día 24 de Abril del 2012. Una vez que los datos repetidos extraídos por ASPIA fueron eliminados, la cantidad de datos restantes fue de 319 filósofos franceses. Estos datos fueron almacenados en una base de datos MySQL en forma de tuplas {(France, Michael_Foucault), (France, Jean_Paul_Sartre), ...}. Posteriormente se generó un dump de dicha base de datos. El dump generado fue tomado como entrada en PIA, y se obtuvo el siguiente resultado: de los 319 pares (France, ?philosopher), se encontraron 317 coincidencias en las cuales existe al menos un camino que vayan de Francia al filósofo, 1 no fue encontrado, y en 1 caso no se encontró en Wikipedia la entrada de DBpedia, debido a un error de sincronización. Como parte del resultado, se ha obtenido la tabla 1 que muestra los caminos que halló PIA para navegar desde Francia a los filósofos. La primer columna hace referencia a la categorización preferida por parte de los usuarios de Wikipedia, la segunda el camino y el tercero, a la cantidad de filósofos a los que se pudo llegar a través de ese camino.

Si comparamos el resultado de ejecutar los datos obtenidos por PIA junto a ASPIA, y los datos obtenidos en *Improving Wikipedia with DBpedia*[10] por PIA,

Pos	Path Query	#
1	[from]/French_people/French_people_by_occupation/French_philosophers/[to]	289

Table 2. Resultado de la ejecución de la consulta de los filósofos en PIA.

podemos observar que el camino que tiene una mayor cantidad de ocurrencias sigue siendo el mismo y, como era de esperarse, la cantidad de ocurrencias para dicho camino es mayor en esta última ejecución, de PIA junto con ASPIA.

6 Trabajos Relacionados

Uno de los principales sistemas de búsqueda semántica en la web que trabaja de modo similar a ASPIA es Sindice[11].

Sindice es una plataforma diseñada para ayudar a la creación de aplicaciones que trabajen por encima de los datos semánticos distribuidos por la web. Sindice reune los datos en la web de distintas bases semánticas, siguiendo los estandares de la web actuales, ofreciendo búsqueda y consultas a través de estos datos, actualizando en tiempo real dichos datos en pocos minutos.

De acuerdo a la naturaleza de Sindice, ASPIA podría combinarse con Sindice para incluir los datos de su resultado a PIA.

Una de las diferencias que se puede encontrar en Sindice es la utilidad final que brinda. Sindice ofrece la posibilidad de trabajar como nexo entre distintas aplicaciones y los datos semánticos distribuidos por la web, generando de este modo una capa nueva por sobre Sindice, las distintas aplicaciones que lo utilizan. Otra de sus características distintivas es el poder utilizar alguna de sus herramientas con el fin de obtener los datos en caso de ser un usuario final.

ASPIA a diferencia de Sindice, transforma cualquier recurso extraído de una base semántica a un tipo de dato que pueda ser procesado directamente por PIA. Otra de las diferencias es que Sindice no filtra los recursos extraídos que son repetidos, en cambio los diferencia por la base origen y se los otorga al usuario o aplicación que lo haya solicitado, mientras que ASPIA elimina los recursos repetidos en las distintas bases semánticas. Estos datos son obtenidos de las bases de conocimiento semántico y almacenados.

7 Conclusión y Trabajos Futuros

Se ha presentado el algoritmo ASPIA, una capa de abstracción encargada de englobar información de diferentes bases semánticas en un solo conjunto de resultados, sin elementos repetidos, e indiferente de la fuente de información. Experimentamos con ASPIA, realizando consultas SPARQL sobre las bases semánticas DBpedia y Yago, y con el resultado devuelto, vimos que en conjunto tienen más información que cada una de ellas por separado. Si bien la utilización de dos repositorios semánticos que utilizan estrategias distintas para la extracción de

sus datos de Wikipedia mejoró el resultado final, se cree que al ampliar la cantidad de repositorios se podría mejorar el conjunto de datos obtenidos por ASPIA. Uno de los trabajos a futuro será la incorporación de nuevos repositorios, tal como Freebase[2], y el análisis de su aporte sobre ASPIA. También se ampliará el campo de consultas para que no sólo abarque a los filósofos franceses, se podrá realizar cualquier consulta semántica en DBpedia, Yago y otro repositorio que se desee agregar implementando nuevos traductores encargados de realizar los mapeos.

Agradecimientos *Este trabajo fue financiado por el PAE 37279-PICT 02203 el cual es esponsoreado por la ANPCyT, Argentina.*

References

1. Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann. Dbpedia - a crystallization point for the web of data. *J. Web Sem.*, 7(3):154–165, 2009.
2. Kurt D. Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In Jason Tsong-Li Wang, editor, *SIGMOD Conference*, pages 1247–1250. ACM, 2008.
3. Jérôme Euzenat. An api for ontology alignment. In Sheila McIlraith, Dimitris Plexousakis, and Frank van Harmelen, editors, *The Semantic Web – ISWC 2004*, volume 3298 of *Lecture Notes in Computer Science*, pages 698–712. Springer Berlin / Heidelberg, 2004. 10.1007/978-3-540-30475-3_48.
4. Dennis Heimbigner and Dennis Mcleod. A federated architecture for information management. *ACM Transactions on Office Information Systems*, 3:253–278, 1985.
5. Raphael Hoffmann, Saleema Amershi, Kayur Patel, Fei Wu, James Fogarty, and Daniel S. Weld. Amplifying community content creation with mixed-initiative information extraction. In *Proceedings of CHI09*, 2009.
6. Prateek Jain, Pascal Hitzler, Amit P. Sheth, Kunal Verma, and Peter Z. Yeh. Ontology alignment for linked open data. In Peter F. Patel-Schneider, Yue Pan, Pascal Hitzler, Peter Mika, Lei Zhang, Jeff Z. Pan, Ian Horrocks, and Birte Glimm, editors, *International Semantic Web Conference (1)*, volume 6496 of *Lecture Notes in Computer Science*, pages 402–417. Springer, 2010.
7. George A. Miller. Wordnet: A lexical database for english. *Communications of the ACM*, 38:39–41, 1995.
8. Amit P. Sheth and James A. Larson. Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Comput. Surv.*, 22(3):183–236, September 1990.
9. Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. YAGO: A core of semantic knowledge - unifying WordNet and Wikipedia. pages 697–706, Banff, Canada, 2007. ACM.
10. Diego Torres, Pascal Molli, Hala Skaf-Molli, and Alicia Diaz. Improving wikipedia with DBpedia. In *SWCS - Semantic Web Collaborative Spaces Workshop 2012 in 21st WWW Conference 2012*, Lyon, France, 2012. ACM.
11. Giovanni Tummarello, Renaud Delbru, and Eyal Oren. Sindice.com: Weaving the open linked data. In *In Proceedings of the International Semantic Web Conference (ISWC)*, 2007.