GEORGE-AUGUST-UNIVERSITÄT GÖTTINGEN

INSTITUTE FOR MATHEMATICS AND COMPUTER SCIENCE

Comparison Of two RDF-based Ontology

DBpedia vs. YAGO

Azadeh Amiri

Dorna Amiri

Project Seminar
Web Data Integration and Data Management

Summer 2015

TABLE OF CONTENTS

CHAPTER I	**DBpedia**	**1**
1.1	Introduction	1
1.2	Extraction Framework	2
1.3	Extractors	3
	1.3.1 Raw Infobox Extraction	3
	1.3.2 Mapping-Based Infobox Extraction	3
	1.3.3 Feature Extraction	4
	1.3.4 Statistical Extraction	4
	1.3.5 URI Schemes	4
1.4	DBpedia Live Synchronization	4
1.5	DBpedia Interlinking	5
1.6	DBpedia Ontology	5
1.7	DBpedia Knowledge base Usage on the Web	6
CHAPTER II	**YAGO (Yet Another Great Ontology)**	**7**
2.1	Introduction	7
2.2	YAGO-NAGA	8
2.3	YAGO Architecture	8
	2.3.1 YAGO Core Extractors	9
	2.3.2 YAGO Core Checker	9
	2.3.3 YAGO Gatherer	10
	2.3.4 YAGO Scritinizer	10
2.4	YAGO Data Structure	11
	2.4.1 Entities	11
	2.4.2 Relations	12
CHAPTER III	**QUERIES**	**14**
REFERENCES		

CHAPTER I

DBpedia

1.1 INTRODUCTION

Wikipedia is a widely deployed encyclopedia with the advantage of a collaborative authorship which developed it to a central knowledge source of mankind. The main part of Wikipedia articles consists of free text, but different types of structured data is also supported such as infobox templates, categorization data, images, tables and lists. The infobox contains some concise information about the described subject in the Wikipedia articles which is represented as attribute-value pairs.

Since Wikipedia only provides free text search restricted to keyword matching, the expressive queries such as “finding all rivers that flow into the Rhine and are longer than 100 miles” cannot easily be answered. Another problem of Wikipedia is the possibility of duplicated information occurrence on different pages and in different Wikipedia language editions which results in inconsistency.

The DBpedia project main objective is to extract different kinds of structured information from Wikipedia editions in multiple languages and constructs a multilingual knowledge base. “The DBpedia project maps Wikipedia infoboxes from 27 different language editions to a single shared ontology consisting of 320 classes and 1,650 properties.” For every page in Wikipedia, there would be a constructed Uniform Resource Identifier (URI) in DBpedia for the identification of an entity which has been depicted by the matching Wikipedia page. During the extraction process, structured information from the wiki turned into RDF triples and is added to the knowledge base as properties of the corresponding URI.

In addition to the DBpedia’s capability to answer the complicated and sophisticated expressive queries; one of the important characteristic of DBpedia is its powerful coverage of various topics enabling it to be used in further application domains.

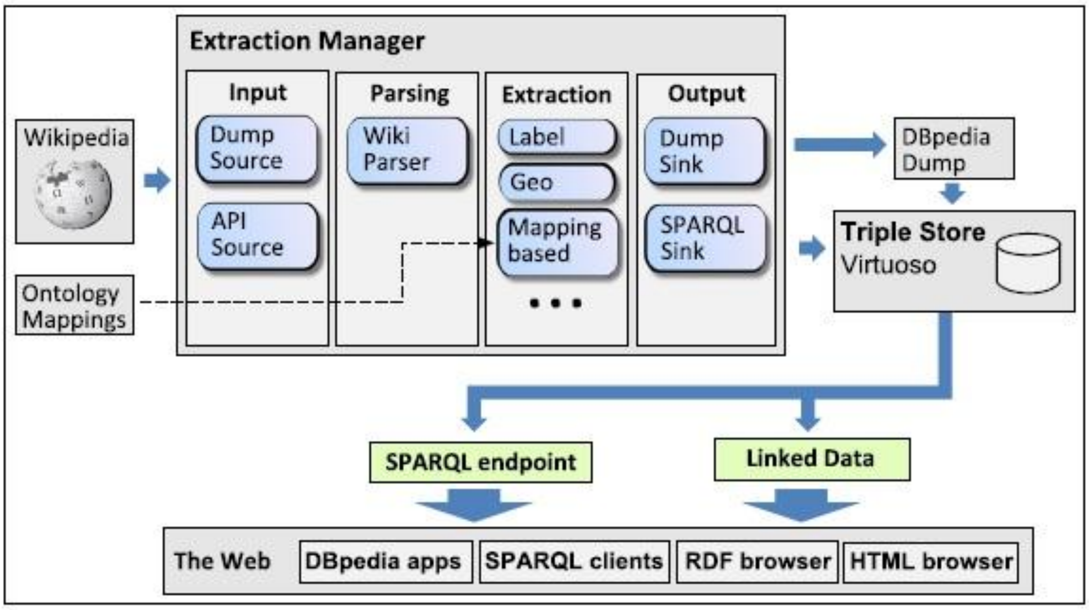
The DBpedia performs as a central interlinking hub in the web of Linked Data; there are RDF links from DBpedia directing into different external web data sources and also RDF links from Linked Data publishers’ datasets pointing to DBpedia.

From practical point of view, a large number of Tools have been constructed using DBpedia knowledge base: DBpedia Mobile, Relation Finder, Navigator and Query

Builder. Moreover, in order to enhance the applications and research approaches, the DBpedia has been used by many researchers as a conducting testbed.

1.2 EXTRACTION FRAMEWORK

The structur ed information in Wikipedia such as infobox templates, links to external web pages, disambiguation pages, redirects between pages, geo-coordinates, and links across different language editions of Wikipedia would be extracted and transformed into the DBpedia knowledge base by its extraction framework.



As it is shown in the above figure, the extraction in DBpedia consists of four stages:

- Input: The Wikipedia pages can be read from a Wikipedia dump or directly fetched from a MediaWiki installation.
- Parsing: The source code in the Wikipedia page would be transformed into an Abstract Syntax Tree by wiki parser. In fact, parsers help the extractor by datatypes identification, value conversion of different units and splitting the markup into lists.
- Extraction: The generated Abstract Syntax Tree is the input for the extractors. There are many extractors in DBpedia for labels, abstracts or for example geographical coordinates. The extractors receive the Abstract Syntax Tree and produce triples.
- Output: The collected RDF statements for example in N-Triples format will be written to a sink.

1.3 EXTRACTORS

The various extractors existing in the extraction manager plays an important role in the extraction framework by transforming different parts of Wikipedia pages to RDF statements. These extractors process the Wikipedia content such as: labels, images, interlanguage links, geo coordinates, disambiguation, external links, etc.

1.3.1 RAW INFOBOX EXTRACTION

The infobobxes in Wikipedia which present and highlight the valuable facts of the articles are crucial units for the DBpedia extraction. These infoboxes frequently appear on the *top right*-hand corner of the article is actually a table of attribute-value pairs. The common used infoboxes in Wikipedia describe persons, countries, organisations, automobiles.

In raw infobox extraction method, the infoboxes in Wikipedia is mapped directly to RDF. Since in the raw infobox extraction the synonyms attributes are not resolved, the query writing would be a difficult task. In addition, the basis of the datatypes determination of the properties is heuristics, leading to relatively high error rate. The mentioned problems cause the quality of the extracted data to be low.

1.3.2 MAPPING-BASED INFOBOX EXTRACTION

The main aim of Mapping-Based Infobox extraction method is to generate high quality data. The method solves the problem of heterogeneity in the Wikipedia infobox system. The Wikipedia templates were mapped to the ontology which was created manually. Therefore uniting the name variations and assigning specific datatypes to the values will homogenize the representation of the information in the knowledge base.

The mapping of Wikipedia templates to the DBpedia ontology is performed by DBpedia Mapping language using the MediaWiki templates that define DBpedia ontology classes and properties as well as template/table to ontology mappings. The allocation of a type from the DBpedia ontology to the represented entities in the related infobox will be done by a mapping. Furthermore, attributes in the infobox are mapped to the properties in the DBpedia ontology. The DBpedia mapping Wiki does not only consider the templates with a single language edition to be mapped, but also the templates from all Wikipedia language editions will be mapped to the shared DBpedia ontology.

1.3.3 FEATURE EXTRACTION

In this approach, a number of special extractors which are able to derive a single feature from an article, for example label or geographic coordinates would be used.

1.3.4 STATISTICAL EXTRACTION

DBpedia derive some data sets by the means of Natural Language Processing related extractors which do the task of collecting data based on statistical measures of page links or word counts.

1.3.5 URI SCHEMES

The extraction framework, for every Wikipedia article presents several URIs to display the concepts described on a particular page. The DBpedia URIs have been published under the http://dbpedia.org domain. The significant namespaces are described as follows:

- http://dbpedia.org/resource/: the namespace is used to show the article data and represented by dbr prefix. Based on the article title, there would be a one-to-one mapping between a Wikipedia page and a DBpedia resource.
- http://dbpedia.org/property/: the namespace is used to show the properties which are generated by the raw infobox extraction and represented by prefix dbp.
- http://dbpedia.org/ontology/: it is used to represent the DBpedia ontology and is shown by prefix dbo.

1.4 DBpedia LIVE SYNCHRONIZATION

The DBpedia dump-based extraction mode consumes the Database Wikipedia page collection and produces N-Triples files as the result. Generating new dumps is a time-consuming task because of its manual effort requirements. On the other hand, once the dumped files are published, any improvement to them is impossible. Since Wikipedia articles are uninterruptedly revised with great speed, the DBpedia traditional dump-based extraction generates the DBpedia dataset which could not show the current state of Wikipedia. In order to prevent the outdated dataset in DBpedia, a live extraction system based on a continuous stream of updates from Wikipedia and processes that stream on the fly was developed. It alleviates the problem of the outdated DBpedia datasets with a minimum delay of just a few minutes. The Wikimedia supports the DBpedia project’s access to the Wikipedia OAI-PMH live feed. The protocol provides permission to a programme to retrieve updates in XML via HTTP. A Java component, which performs as a proxy, continuously extracts new updates and supplies the DBpedia live framework. The update stream is deployed to gain access to new knowledge upon relevant changes in Wikipedia articles.

1.5 DBpedia INTERLINKING

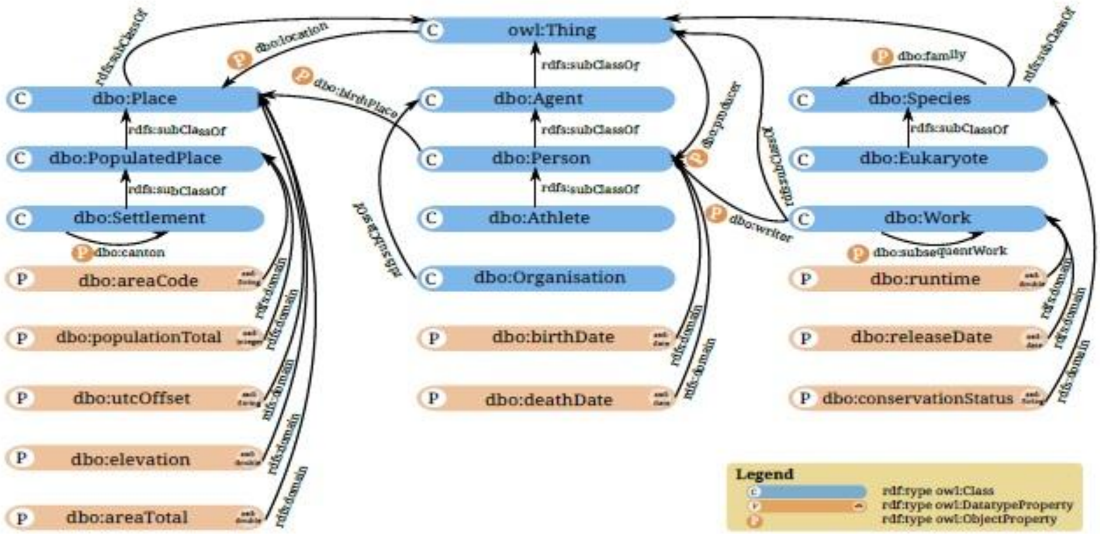
Since DBpedia knowledge base is interlinked with a number of data sources based on linked data principles, the DBpedia users have the ability to access more information. There are several outgoing links that point from DBpedia into other data sets, as well as outside data sets which publish their links pointing to DBpedia.

There is a link repository supported by DBpedia in which conventions for adding linksets and linkset metadata are described. The joined linkstes to the repository would be deployed for the subsequent official DBpedia release. For example, one of the added data sets is the GeoNames which DBpedia links to by using the predicate “owl:same as” And the overall number of links set between them is 86500. Furthermore, DBpedia has the capability of setting links on schema level. Links to other schemata can be set by using the “owl:equivalentClass” in class templates and owl:equivalentProperty in datatype or object property templates. There are also a number of data sets pointing their links to DBpedia. The Sindice has been used to ascertain the online datasets which have a link to DBpedia. It crawls the RDF resources on the web and indexes those resources. If the data set of the subject and object are different, the triple is identified as a link. Sindice has indexed 4 million links pointing to DBpedia.

1.6 DBpedia ONTOLOGY

The DBpedia user community supports the structure of the DBpedia knowledge base and produces mappings from Wikipedia information representation structures to the DBpedia ontology. The DBpedia ontology which based on OWL consists of 320 classes and 1,650 different properties. The classes construct a subsumption hierarchy in which the root node of the structure is the owl:Thing. In Figure , a part of DBpedia ontology is shown, representing the relationship between its top ten classes.

As mentioned before, each infobox is mapped to a class and each attribute of it would be represented as a property in DBpedia ontology. The DBpedia ontology and mappings are kept externally in Mappings-Wiki. In addition to adding more ontology classes or checking the ontology classes hierarchy, a DBpedia user would be able to alter the relation between an infobox and its corresponding ontology class and/or an infobox property and its corresponding ontology property. The DBpedia ontology wiki can be found at http://mappings.dbpedia.org and in order to edit, users should register and obtain the editor right by contacting the DBpedia maintainers.



The DBpedia ontology has grown over the time, but since the classes have a good coverage in the initial versions, there is not a significant increase in these ontology elements. On the other hand, the properties has been remarkably growing over time due to the collaboration on the DBpedia Mappings Wiki and the increase of more detailed information to infoboxes by Wikipedia editors.

1.7 DBpedia KNOWLEDGE BASE USAGE ON THE WEB

In addition to DBpedia downloadable data sets consists of the results of the extractors, DBpedia is served via a public SPARQL endpoint (http://dbpedia.org/snorql/). It also performs as a Linked Data Hub which returns resources in a number of various formats.

CHAPTER II

YAGO (Yet Another Great Ontology)

2.1 INTRODUCTION

Ontological background knowledge is mostly deployed in Semantic-Web related applications, but the usage is not limited to this field. Machine translation to resolve the word ambiguity problems, query expansion to improve the retrieval performance, and even document classification to present a more precise classifier, need to make use of lexical knowledge, taxonomies and ontologies. Above all, the importance of ontological background knowledge in data cleaning (detecting and removing errors and inconsistencies from data) , record linkage (finding records in a data set that refer to the same entity across different data sources) and information integration cannot be discarded.

There are many questions related to the way an ontology could be defined and deployed to provide the best results.

- How accurate and efficient is the utilized ontology?
- Does the utilized ontology satisfy the expected performance of an application?
- Are there other resources to improve the quality of an ontology?
- How could the utilized ontology be re-usable and application-independent?

The problem with the most existing applications is that they usually rely on one single source of background knowledge, and this decreases the quality of the ontology and finally degrades the performance.

Most of the knowledge sources are acceptable and successful from the quality point of view, because they are man-made and manually assembled. Apart from this benefit, there are some major disadvantages such as low coverage, high cost for assembly and fast aging.

Knowledge-based applications can easily foster their performance by using a relatively thorough ontology, instead of one-source-dependent ontology. The target ontology exploits several sources, which makes it somehow as powerful as an encyclopedia. It includes concepts, named entities and also the relations among them. Such ontology is extensible, can be re-used and applied in many applications without any dependency.

2.2 YAGO-NAGA

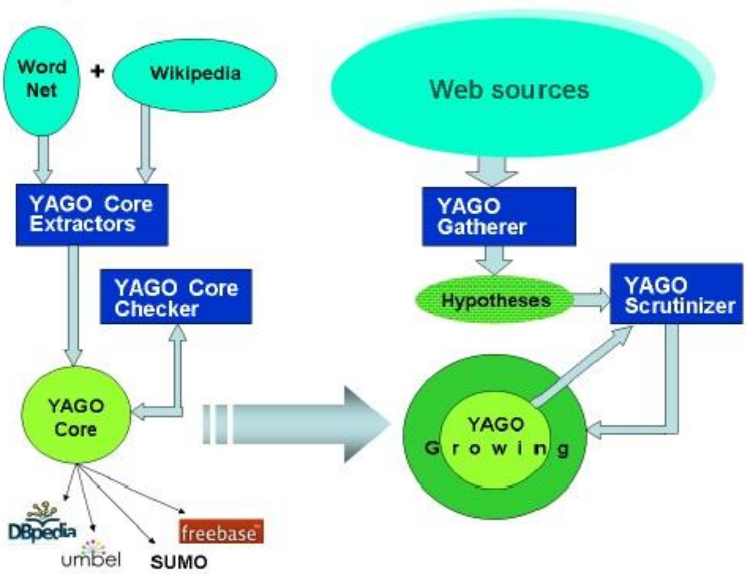
A comprehensive knowledge base is expected to know all individual entities of this world (e.g., Germany), their semantic classes (e.g., Germany isa Country), relationships between entities (e.g., Germany linksTo EU), as well as validity times and confidence values for the correctness of such facts. Besides, it should be logically rich enough to be applied by reasoning and provide necessities for querying.

The YAGO_NAGA project as an innovative application started in 2006 with the purpose of providing a conveniently searchable, large-scale, highly accurate knowledge base of common facts in a machine-processible representation.

2.3 YAGO ARCHITECTURE

Architecture of YAGO has been designed in order to meet both coverage and accuracy. Although high recall is one of the main objectives of YAGO knowledge base, high precision and the consistency of the extracted knowledge is the YAGO's first priority. The architecture includes four fundamental sections:

- YAGO Core Extractors
- YAGO Core Checker
- YAGO Gatherer
- YAGO Scrutinizer



2.3.1 YAGO CORE EXTRACTORS

The main duty of this section is to derive information from infoboxes and category system of Wikipedia, based on rule-based information extraction. Infoboxes are made of a set of property name-value pairs in a template format. For each type of entities (countries, companies, scientists, music bands, sports teams, etc.), there are some properties, which are filled with related values of individual entities. Due to many naming diversity and noise in infoboxes, YAGO only extracts frequently used properties, not all, and then normalizes the matching values. For example, property “spouse” in Elvis Presley’s infobox is mapped to “marriedTo” relation, and the fact is interpreted as (Elvis_Presley marriedTo Priscilla_Presley).

In the category system of Wikipedia, entities are categorized manually, and this leads YAGO to define classes in the instanceOf relations. Occasionally, some misleading category name make it difficult for YAGO to extract the proper class membership (instanceOf). To solve this problem, YAGO employs linguistic processing (noun phrase parsing) and also some heuristics (e.g.,the head word in the noun phrase should be in plural form) in the category system.

One important aspect of facts is that they may change by time. For example presidential election takes place every four years in some countries, so the value of the relation "presidentOf" will not be the same through all years. To ensure the validity of the knowledge, YAGO uses temporal annotation for facts. It adopts several relations such as "ValidSince" and "ValidUntil" to tackle the problem.

2.3.2 YAGO CORE CHECKER

Every single fact is checked for consistency before it is added to the knowledge base. YAGO core checker examines the facts from three points of view: relationals, class hierarchy, type conditions.

Information extraction form infoboxes and categories of Wikipedia cannot provide a through ontology with high quality, but even may result in a large incoherent collection of facts. YAGO intensively deployed Wordnet and integrates the facts from Wikipedia with the clean taxonomy of WordNet.

Each extracted individual entity is mapped to at least one of the YAGO classes, which are imported from WordNet classes along with their superclasses/subclasses. Otherwise, the entity is not allowed to be inserted in the knowledge base. The derived classes from Wikipedia category names also need to be mapped to at least one superclass, using subclassOf relationship. Satisfying these conditions guarantees the consistency of the knowledge base and acyclic subclassOf relation.

Every binary relation has a type signature with particular domain and range. This avoids wrong interpretation of infobox properties, since each domain and range includes entities of particular type.

Declaring the type of relations such as transitive and acyclic, is also an effective way for the consistency of extracted facts. The class hierarchy is one important usage case. Functional dependencies, inclusion dependencies, and inverse relations are other types which are still developing.

2.3.3 YAGO GATHERER

The gathering phase employs recall-oriented IE methods, and aims at high throughput.

The YAGO knowledge base can be kept up-to-date by periodically re-running the extractors on Wikipedia and WordNet. As the new facts are added, the old ones can also be kept, if they do not cause any inconsistencies (time validity). In addition to infoboxes and categories as the main sources of the YAGO core, it can be growed by adding natural-language Text Sources. LEILA is a powerful NLP tool that is used for this purpose. It uses a dependency-grammar parser for deep parsing of natural language sentences, and provides reasonably good accuracy. Other NLP methods are also supported by YAGO architecture, but not in large scale.

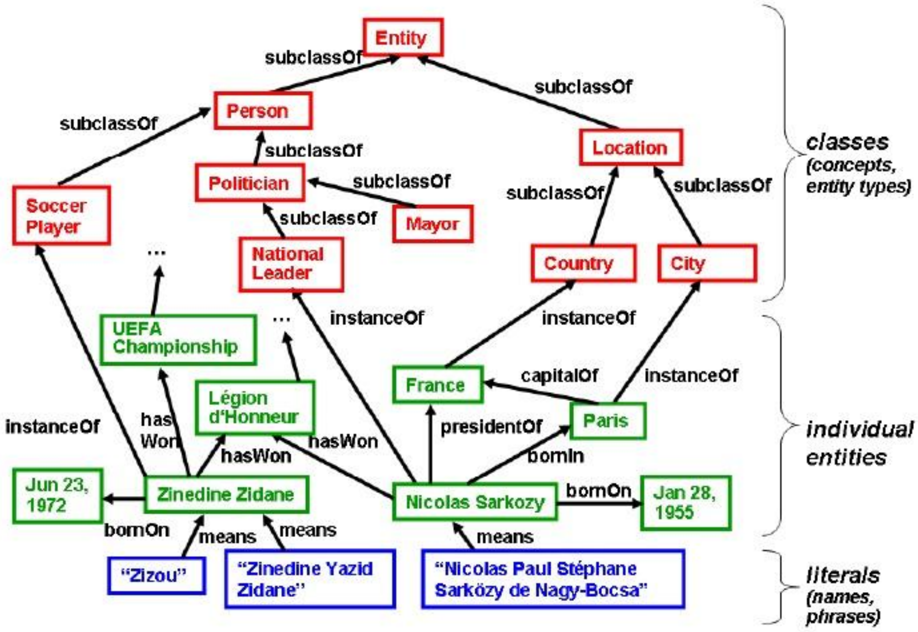
The possibility that every famous entity (such as politicians, athletes, pop stars, movies, cities, rivers, etc) has its own page in Wikipedia is very high, so YAGO core seems to contain almost all entities with their important classes. On the other hand, this does not hold for classes like computer scientists, medical drugs,etc. The YAGO has the capability of adapting the YAGO extractors to other sources such as DBLP or UMLS, in order to add one or two interesting entities. The output is interpreted as a set of fact hypotheses.

2.3.4 YAGO SCRUTINIZER

The scrutinizing phase assesses the hypotheses against the existing knowledge base. Adding the new facts for the purpose of growing the YAGO core, may disturb the consistency, so new derived facts are batched and checked by consistency checking procedures. Facts that show high indication of being inconsistent with essential invariants and prior knowledge will be filtered out.

2.4 YAGO DATA STRUCTURE

YAGO data model from expressiveness aspect is designed as complete as it could be. It is expected to be able of expressing entities, facts, relations between facts and properties of relations. OWL(Web Ontology Language) as the current best knowledge presentation utilizes RDFS as the basis, which is expressive enough to indicate the relations between facts but provides only very primitive semantics. For example, it does not know transitivity, which is crucial for partial orders such as subClassOf. YAGO data model is an improved version of RDFS. It has resolved the weaknesses of RDFS and also different variants of OWL in the field of expressiveness. Overall, YAGO is highly expressive, simple and decidable.



2.4.1 ENTITIES

The YAGO model uses the same knowledge representation as RDFS: All objects (e.g. cities, people, even URLs) are represented as entities in the YAGO model. Numbers, dates, strings and other literals are represented as entities as well. Two entities can stand in a relation. The triple of an entity, a relation and an entity is called a fact. The two entities are called the arguments of the fact. Each fact is given a fact identifier. As RDFS, the YAGO model considers fact identifiers to be entities as well.

2.4.2 RELATIONS

Wikipedia includes a lot of defined individuals in comparison with WordNet, So individuals are taken from Wikipedia. Each entity is an instance of at least one class. For each individual there is a unique Wikipedia page title, which is a candidate to become an individual. YAGO only considers conceptual category pages in Wikipedia among different types of categories, to assign classes to individuals. To identify conceptual categories, a light linguistic parsing of the category name is employed. It extracted the best word in a compound noun as a class, which is mostly a plural word.

The conceptual categories yield not only the type relation, but also, as its domain, the set of individuals. It also yields, as its range, a set of classes. The reason is that only individuals have conceptual categories, not all article pages such as hub pages.

The defined category system in Wikipedia has not a significant efficiency, since it only reflects the thematic structure of the Wikipedia pages. In YAGO, only leaf categories of Wikipedia are considered, and then the hierarchy of classes are made, using WordNet well-organized taxonomy of sysnsets. Each synset of WordNet becomes a class of YAGO. To prevent any conflicts between individuals in Wikipedia and common nouns in WordNet, the preference is always given to the WordNet. This ensures that all common nouns are classes and no entity is duplicated.

The lower classes extracted from Wikipedia have to be connected to the higher classes extracted from WordNet. The subClassOf relation refers to the hyponymy relation from WrodNet. If the first sysnset of class A is a hyponym of the class B, then class A is a subclass of class B.

Both Wikipedia and WordNet can be deployed to provide the word meaning, using redirect pages and sysnsets respectively. This allows us to deal with synonymy and ambiguity.

The YAGO hierarchy of classes allows us to identify individuals that are persons. Individuals of Person type follow the common pattern of a given name and a family name. Thus, the relations givenNameOf and familyNameOf are established. and the Name Parser is used to identify and decompose the person names.

Extraction of other relations is done by employing the relational Wikipedia categories. Some relations which are common between most entities are defined using categories end or begin with specific nouns. Others only exist between a group of entities with the same time. For all these relations, the domain and the range should be valid.

All the facts in YAGO knwledge base are indicated in the forms of unary (classes of individual entities) and binary (pairs of entities connected by specific relationship types) relations. YAGO data model can also be represented as a graph, in which vertices and edges refer to entities(and classes) and relations respectively. Every single fact in this graph can be interpreted as a RDF triple of (subject, predicate, object).

CHAPTER III

QUERIES (DBpedia vs. YAGO)

In this chapter, some geographical queries in DBpedia and YAGO were written and compared. Each SPARQL query's result in both DBpedia and YAGO was analyzed to answer the related questions; for example why some expected results were missing, duplicated, or not provided. The DBpedia was able to answer most of the queries, but not completely. since it covers a wide range of properties, it has a great power to generate the results, on the other hand because of low consistency between resource properties, most of the time the results are not complete.That makes writing queries in DBpedia is more complicated. In comparison with DBpedia, number of defined relations is a lot less, so that some queries cannot be answered. The positive point is that the existing relations are almost the same for all the entities, and that makes it easy to write queries. Entities of the same type always have the same relation (property), so if some answers are missing, they simply do not exist in the knowledge base.

- **Source countries of Danube river and the countries it passes by?**

DBpedia

```
PREFIX dbo: <http://dbpedia.org/ontology/>

PREFIX res: <http://dbpedia.org/resource/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
SELECT DISTINCT ?source ?country
WHERE {
            res:Danube dbo:sourceCountry ?source;
            dbo:country ?country.

     ?source rdfs:label ?sourcename.
    ?country rdfs:label ?countryname.
FILTER((langMatches(lang(?sourcename),"en"))&&(langMatches(lang(?countryname),"en")))}
```

YAGO (SPARQL endpoint http://lod2.openlinksw.com/sparql)

```
PREFIX : <http://yago-knowledge.org/resource/>
SELECT distinct ?country
WHERE {   :Danube :isLocatedIn ?country.
            ?country rdf:type :wikicategory_European_countries }
```

<table>
<tr><th>DBpedia Result</th><th>YAGO Result</th></tr>
<tr><td>
<table>
<tr><th>source</th><th>country</th></tr>
<tr><td>:Germany</td><td>:Austria</td></tr>
<tr><td>:Germany</td><td>:Bulgaria</td></tr>
<tr><td>:Germany</td><td>:Croatia</td></tr>
<tr><td>:Germany</td><td>:Germany</td></tr>
<tr><td>:Germany</td><td>:Hungary</td></tr>
<tr><td>:Germany</td><td>:Moldova</td></tr>
<tr><td>:Germany</td><td>:Romania</td></tr>
<tr><td>:Germany</td><td>:Serbia</td></tr>
<tr><td>:Germany</td><td>:Slovakia</td></tr>
<tr><td>:Germany</td><td>:Ukraine</td></tr>
</table>
</td><td>
<table>
<tr><th>country</th></tr>
<tr><td>http://yago-knowledge.org/resource/Moldova</td></tr>
<tr><td>http://yago-knowledge.org/resource/Austria</td></tr>
<tr><td>http://yago-knowledge.org/resource/Serbia</td></tr>
<tr><td>http://yago-knowledge.org/resource/Croatia</td></tr>
<tr><td>http://yago-knowledge.org/resource/Slovakia</td></tr>
<tr><td>http://yago-knowledge.org/resource/Germany</td></tr>
<tr><td>http://yago-knowledge.org/resource/Bulgaria</td></tr>
<tr><td>http://yago-knowledge.org/resource/Romania</td></tr>
<tr><td>http://yago-knowledge.org/resource/Hungary</td></tr>
<tr><td>http://yago-knowledge.org/resource/Ukraine</td></tr>
</table>
</td></tr>
</table>

Result: DBpedia and YAGO, both have obtained the same and correct results, but the YAGO in contrast to DBpedia was not able to find the source country of Danube. The relation between river and the country source is not defined in YAGO.

- **European countries having government type Constitutional monarchy.**

DBpedia

```
PREFIX dcterms: <http://purl.org/dc/terms/>

PREFIX dbpedia-owl: <http://dbpedia.org/ontology/>
PREFIX res: <http://dbpedia.org/resource/>
SELECT DISTINCT ?country
WHERE {
        ?country rdf:type umbel-rc:Country;
        dcterms:subject   <http://dbpedia.org/resource/Category:Countries_in_Europe>;
        dcterms:subject  <http://dbpedia.org/resource/Category:Constitutional_monarchies>.
       }
```

YAGO

```
PREFIX : <http://yago-knowledge.org/resource/>
SELECT distinct ?country
WHERE {
         :Monarchies_in_Europe :linksTo ?country.
        ?country rdf:type :wikicategory_European_countries }
```

DBpedia Result	YAGO Result
country :Andorra :Belgium :Denmark :Liechtenstein :Luxembourg :Monaco :Netherlands :Norway :Spain :Sweden :United_Kingdom	**country** http://yago-knowledge.org/resource/Monaco http://yago-knowledge.org/resource/San_Marino http://yago-knowledge.org/resource/Norway http://yago-knowledge.org/resource/Luxembourg http://yago-knowledge.org/resource/Belgium http://yago-knowledge.org/resource/Vatican_City http://yago-knowledge.org/resource/Denmark http://yago-knowledge.org/resource/Sweden http://yago-knowledge.org/resource/Spain http://yago-knowledge.org/resource/Italy http://yago-knowledge.org/resource/Andorra http://yago-knowledge.org/resource/Liechtenstein http://yago-knowledge.org/resource/Switzerland

Result: DBpedia generates the correct answers, since the rdf:type of the country "Norway" was not dbo:Country; rdf:type umbel-rc:Country was used; otherwise the country "Norway" was missing. In YAGO, the relation "linksTo" does not necessary provide the correct result; for example here the countries Italy, San marino and Vatican city are not constitutional monarchies, but appeared in the results. They are related to the class Monarchies_in_ Europe due to different reasons (such relations is not defined in YAGO).

- **Host cities of the summer Olympic Games as well as capitals in Europe.**

DBpedia

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

PREFIX dcterms: <http://purl.org/dc/terms/>
SELECT DISTINCT ?lcity
WHERE {
    ?city dcterms:subject <http://dbpedia.org/resource/Category:Capitals_in_Europe>;
        rdfs:label ?lcity.

    ?event dbp:title "Summer Olympic Games"@en.
    ?event dbp:title ?lcity.
FILTER ((langMatches(lang(?lcity),"en"))) }
order by ?lcity
```

YAGO

```
PREFIX : <http://yago-knowledge.org/resource/>
SELECT distinct ?city
WHERE {
        ?city rdf:type :wikicategory_Capitals_in_Europe.
        ?city rdf:type :wikicategory_Host_cities_of_the_Summer_Olympic_Games }
```

DBpedia Result	YAGO Result
lcity "Amsterdam"@en "Athens"@en "Berlin"@en "Helsinki"@en "London"@en "Paris"@en "Rome"@en "Stockholm"@en	**city** http://yago-knowledge.org/resource/Athens http://yago-knowledge.org/resource/Berlin http://yago-knowledge.org/resource/Amsterdam http://yago-knowledge.org/resource/Stockholm http://yago-knowledge.org/resource/Helsinki http://yago-knowledge.org/resource/London http://yago-knowledge.org/resource/Paris http://yago-knowledge.org/resource/Rome http://yago-knowledge.org/resource/Moscow

Result: In DBpedia the city “Moscow” was missing; the problem occurred because the property “title” of the event 1980_Summer_Olympics (which is related to Moscow) does only include "Summer Olympic Games"@en and “Host city” but not the city “Moscow”.In order to solve the problem in DBpedia:

rdf:type yago:HostCitiesOfTheSummerOlympicGames for the city was used which produced the correct answer.

DBpedia

```
PREFIX yago: <http://dbpedia.org/class/yago/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dcterms: <http://purl.org/dc/terms/>

SELECT DISTINCT ?city
WHERE {
    ?city dcterms:subject <http://dbpedia.org/resource/Category:Capitals_in_Europe>;
        rdfs:label ?lcity.

    ?city rdf:type yago:HostCitiesOfTheSummerOlympicGames
    FILTER ((langMatches(lang(?lcity),"en")))
}
```

DBpedia Result
city
:Amsterdam
:Helsinki
:Moscow
:Paris
:Stockholm
:London
:Rome
:Berlin
:Athens

- **European countries with capitals and their population of capitals.**

DBpedia

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

PREFIX dcterms: <http://purl.org/dc/terms/>

SELECT distinct ?cn ?cap ?pop
WHERE {{ ?cap dcterms:subject
          <http://dbpedia.org/resource/Category:Capitals_in_Europe>;
          dbpedia-owl:country ?c;
          dbpedia-owl:populationTotal ?pop.

          ?c rdfs:label ?cn }
UNION
{      ?c dcterms:subject
       <http://dbpedia.org/resource/Category:Countries_in_Europe>;
       dbpedia-owl:capital ?cap;
       rdfs:label ?cn.
       ?cap dbpedia-owl:populationTotal ?pop
        }

FILTER ((langMatches(lang(?cn),"en")))
}
order by ?c
```

YAGO

```
PREFIX : <http://yago-knowledge.org/resource/>
SELECT ?cn ?ncap ?pop
WHERE {  ?c rdf:type :wikicategory_European_countries;
                skos:prefLabel ?cn;
                :hasCapital ?cap.
           ?cap skos:prefLabel ?ncap;
```

```
            :hasNumberOfPeople ?pop
        }
Order by ?cn
```

DBpedia Result	Yago Result
• Result : 53 • Missing: 12 • Duplicate : 5	• Result: 38 • Missing: 15 • Duplicate : -

Results : In DBpedia the countries such as Belguim, Greece, Finland, Russia, Italy, Netherlands, Ukraine, United Kingdom, Lithuania and Luxemburg were missing; because their capitals do not have the property dbpedia-owl:populationTotal. The country “Armenia” was also missing; since its capital “Yerevan” does not belong to the “Category:Capitals_in_Europe”. The country “Malta” does not have the English label “Malta”@en, therefore it was missing, too.

In DBpedia there were also duplicate countries in the result, because there are for example for the capital “Nicosia”, the dbo:country includes Cyprus, Northern Cyprus, Turkey and North Nicosia. The country “Denmark” also appeared twice, because for the capital “Nuuk”, the dbo:country is Denmark. The Georgia and the Georgia(country) were also in the result; because the dbr:Tbilisi dbo:country dbr:Georgia and dbr:Georgia(country) dbo:capital dbr: Tbilisi.

Using YAGO, there are 50 countries in class wikicategory_European_countries, but the query returns only 38 records. The reason is that some counties in this class have no capitals or no population for the capitals. In addition, some countries do not exist in the class wikicategory_European_countries such as Bosnia and Herzegovina, Turkey and Northern Irland.

Country	:hasCapital	:hasNumberOf People
Belgium		No
Finland		No
Greece		No
Italy	No	
Luxemburg		No
Monaco	No	
Portugal		No
Republic Of Kosovo	No	
Russia		No
Slovakia		No

- **Name and population (ordered) of all European countries that have more than 10.000.000 inhabitants.**

YAGO

```
PREFIX : <http://yago-knowledge.org/resource/>
SELECT ?cn ?pop
WHERE { ?c  rdf:type :wikicategory_European_countries;
              skos:prefLabel ?cn.
          OPTIONAL { ?c :hasNumberOfPeople ?pop.FILTER (?pop > 10000000) }
          FILTER(BOUND(?pop))
        }
Order by ?cn
```

DBpedia

```
PREFIX dbpedia-owl: <http://dbpedia.org/ontology/>
PREFIX dcterms: <http://purl.org/dc/terms/>
SELECT distinct ?cn ?cp
WHERE {  { ?cap dcterms:subject
                 <http://dbpedia.org/resource/Category:Capitals_in_Europe>;
                 dbpedia-owl:country ?c.
           ?c rdfs:label ?cn }
UNION
         { ?c  dcterms:subject
                <http://dbpedia.org/resource/Category:Countries_in_Europe>;
               rdfs:label ?cn}

OPTIONAL { ?c dbpedia2:populationEstimate ?cp.Filter(?cp>10000000)}
OPTIONAL { ?c dbpedia2:populationCensus ?cp.Filter(?cp>10000000) }
FILTER (bound(?cp))
FILTER (langMatches(lang(?cn),"en"))
}
order by ?c
```

DBpedia Result

cn	cp
"Belgium"@en	11198638
"Czech Republic"@en	10513209
"Germany"@en	80716000
"Greece"@en	10816286
"Italy"@en	60782668
"Netherlands"@en	16912640
"Poland"@en	38483957
"Portugal"@en	10427301
"Romania"@en	19942642
"Russia"@en	143975923

YAGO Result

cn	pop
"Belgium"	11007020
"Germany"	81799600
"Greece"	11305118
"Hungary"	10014324
"Italy"	60681514
"Netherlands"	16847007
"Poland"	38186860
"Portugal"	10576252
"Russia"	143030106

<table>
<tr>
<td>
<table>
<tr><td>"Spain"@en</td><td>46704314</td></tr>
<tr><td>"Turkey"@en</td><td>77695904</td></tr>
<tr><td>"Ukraine"@en</td><td>44291413</td></tr>
<tr><td>"United Kingdom"@en</td><td>64100000</td></tr>
</table>
</td>
<td>
<table>
<tr><td>"Spain"</td><td>46030109</td></tr>
<tr><td>"Ukraine"</td><td>45888000</td></tr>
<tr><td>"United Kingdom"</td><td>62262000</td></tr>
</table>
</td>
</tr>
</table>

Result: The country “France” does not have the property “populationEstimate” or “populationCensus”; it has the dbp:frTotalPopulationEstimate; so it was missing in DBpedia.

Using YAGO, there are two missing countries: Czech Republic and Romania. Both of them have the relation :hasNumberOfPeople with values more than 10000000, according to the YAGO browser, but it seems that the links are broken.

Besides the result includes Hungary with the population of "10014324" ,which is wrong. In YAGO browser the population equals to 9935000 as well as in reality. Turkey is also not in the results, since it does not exist in the class wikicategory_European_countries.

- **Name of all european countries that have no membership in the European Union.**

DBpedia

```
PREFIX dbpedia-owl: <http://dbpedia.org/ontology/>
PREFIX dcterms: <http://purl.org/dc/terms/>
SELECT distinct ?country
WHERE {  ?country rdf:type umbel-rc:Country;
                    dcterms:subject
                        <http://dbpedia.org/resource/Category:Countries_in_Europe> .
          FILTER NOT EXISTS {?country dcterms:subject
          <http://dbpedia.org/resource/Category:Member_states_of_the_European_Union> .}
}
order by ?country
```

YAGO

```
PREFIX : <http://yago-knowledge.org/resource/>
SELECT ?country
WHERE { {?country rdf:type :wikicategory_European_countries}
           MINUS
          {?country rdf:type :wikicategory_European_countries;
                       rdf:type :wikicategory_Member_states_of_the_European_Union;
                      :linksTo :European_Union} }
ORDER BY ?country
```

DBpedia Result	YAGO Result
country	**country**
:Albania	http://yago-knowledge.org/resource/Albania
:Andorra	http://yago-knowledge.org/resource/Andorra
:Armenia	http://yago-knowledge.org/resource/Armenia
:Azerbaijan	http://yago-knowledge.org/resource/Azerbaijan
:Belarus	http://yago-knowledge.org/resource/Belarus
:Bosnia_and_Herzegovina	http://yago-knowledge.org/resource/Croatia
:Georgia_(country)	http://yago-knowledge.org/resource/European_microstates
:Iceland	http://yago-knowledge.org/resource/Georgia_(country)
:Kazakhstan	http://yago-knowledge.org/resource/Iceland
:Kosovo	http://yago-knowledge.org/resource/Liechtenstein
:Liechtenstein	http://yago-knowledge.org/resource/Moldova
:Moldova	http://yago-knowledge.org/resource/Monaco
:Monaco	http://yago-knowledge.org/resource/Montenegro
:Northern_Cyprus	http://yago-knowledge.org/resource/Northern_Cyprus
:Norway	http://yago-knowledge.org/resource/Norway
:Republic_of_Macedonia	http://yago-knowledge.org/resource/Republic_of_Kosovo
:Russia	http://yago-knowledge.org/resource/Republic_of_Macedonia
:San_Marino	http://yago-knowledge.org/resource/Russia
:Serbia	http://yago-knowledge.org/resource/San_Marino
:Switzerland	http://yago-knowledge.org/resource/Serbia
:Turkey	http://yago-knowledge.org/resource/Switzerland
:Ukraine	http://yago-knowledge.org/resource/Ukraine
:Vatican_City	http://yago-knowledge.org/resource/Vatican_City

Result : In DBpedia the country “Montenegro” was missing, because it does not belong to the Category:Countries_in_Europe.

Using YAGO, there are three missing countries: Bosnia and Herzegovina, Turkey and Kazakhstan. All of them are not in class wikicategory_European_countries.

In addition, Croatia exists in the results as a wrong answer, since it is a memeber of EU. Class wikicategory_Member_states_of_the_European_Union does not include Croatia, so here it is considered as non-member of EU.

- **Abbreviations of all organizations whose headquarter is located in the capital of a member country (together with the names of the country and the city).**

DBpedia

```
PREFIX dbpedia-owl: <http://dbpedia.org/ontology/>
PREFIX dcterms: <http://purl.org/dc/terms/>

SELECT  distinct  ?cn  ?capn  ?abr
WHERE {{
        ?cap dcterms:subject
            <http://dbpedia.org/resource/Category:Capitals_in_Europe>;
            dbpedia-owl:country ?c;
            rdfs:label ?capn.
        ?c rdfs:label ?cn}
    UNION
        { ?c dcterms:subject
            <http://dbpedia.org/resource/Category:Countries_in_Europe>;
            dbpedia-owl:capital ?cap;
            rdfs:label ?cn.
          ?cap rdfs:label ?capn}

             ?org rdf:type dbpedia-owl:Organisation;
             dbpedia-owl:headquarter ?cap;
             dbpedia-owl:abbreviation ?abr.

    FILTER ((langMatches(lang(?capn),"en")) && (langMatches(lang(?cn),"en"))) }
```

YAGO

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

SELECT ?oname ?cname ?cntname
WHERE { ?org rdf:type :wordnet_world_organization_108294696;
          skos:prefLabel ?oname;
          :isLocatedIn ?cty.

       ?cty rdf:type :wordnet_capital_108518505;
          skos:prefLabel ?cname.
       ?country rdf:type :wordnet_country_108544813;
            :hasCapital ?cty;
            skos:prefLabel ?cntname }

order by ?cntname
```

Results: YAGO was not able to generate abbreviation, because it does not have the related relation. That makes the comparison complicated. On the other hand, DBpedia produced huge number of records; in which some famous organization like

“EU”,”UNESCO” and ”NATO” were missing. The reason is behind missing and inconsistent properties; for example the resource “European_Union” does not have the properties dbpedia-owl:headquarter and dbpedia-owl:abbreviation.

References

- Auer, Sören, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. Springer Berlin Heidelberg, 2007.
- Bizer, Christian, Tom Heath, and Tim Berners-Lee. "Linked data-the story so far." Semantic Services, Interoperability and Web Applications: Emerging Concepts (2009): 205-227.
- Fabian, M. S., K. Gjergji, and W. Gerhard. "YAGO: A core of semantic knowledge unifying wordnet and wikipedia." In 16th International World Wide Web Conference, WWW, pp. 697-706. 2007.
- Kasneci, Gjergji, Maya Ramanath, Fabian Suchanek, and Gerhard Weikum. "The YAGO-NAGA approach to knowledge discovery." ACM SIGMOD Record 37, no. 4 (2009): 41-47.
- Lehmann, Jens, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann et al. "DBpedia-a large-scale, multilingual knowledge base extracted from wikipedia." *Semantic Web Journal* 5 (2014): 1-29.
- Morsey, Mohamed, Jens Lehmann, Sören Auer, Claus Stadler, and Sebastian Hellmann. "Dbpedia and the live extraction of structured data from wikipedia." *Program* 46, no. 2 (2012): 157-181.
- Suchanek, Fabian M., Gjergji Kasneci, and Gerhard Weikum. "Yago: A large ontology from wikipedia and wordnet." Web Semantics: Science, Services and Agents on the World Wide Web 6, no. 3 (2008): 203-217.