Extending YAGO4 Knowledge Graph with Geospatial Knowledge

Maria Despoina Siampou[a], Nikolaos Karalis[b] and Manolis Koubarakis[a]

[a]National and Kapodistrian University of Athens
[b]DICE group, Department of Computer Science, Paderborn University

Abstract

We present an extension of YAGO4, the latest version of the YAGO knowledge graph, with qualitative geospatial information representing the administrative organization of Greece. To achieve this, we derive new geospatial information from the Greek Administrative Geography (GAG) dataset, an official source of the administrative divisions of Greece. Our goal has been to extend entities already existing in YAGO4 with geospatial information and add missing entities without introducing duplicate knowledge. Our study should be viewed as a demonstration of how YAGO4 can be extended with administrative geospatial information for the whole world. In addition, it has allowed us to uncover certain issues that exist with the representation and querying of geospatial information in schema.org and YAGO4 which we discuss in detail.

Keywords

linked open data, knowledge graphs, YAGO

1. Introduction

Many intelligent applications are driven today by knowledge graphs (KGs) such as the Google KG[1], DBpedia [1] and YAGO [2]. YAGO was one of the first research projects to build a knowledge graph automatically. The main idea of YAGO was to harvest information about entities from the infoboxes and categories of Wikipedia, and to combine this data with an ontological backbone derived from classes in WordNet [3]. Since Wikipedia is an excellent repository of entities, and WordNet is a widely used lexical resource, the combination proved useful.

YAGO2 [4, 5], the second version of YAGO, was released in 2011. YAGO2 adds geospatial and temporal information to the YAGO knowledge graph by introducing `geo-entities`. Here, geospatial information does not only come from Wikipedia, but also from GeoNames[2] and is represented with the properties `hasLongitude` and `hasLatitude`. These properties represent the longitude and latitude of the center of a geo-entity, respectively. For example, the

GeoLD 2022: 5th International Workshop on Geospatial Linked Data co-located with ESWC, May 30 2022, Hersonissos, Greece

m.siampou@di.uoa.gr (M. D. Siampou); nikolaos.karalis@uni-paderborn.de (N. Karalis); koubarak@di.uoa.gr (M. Koubarakis)

[1]https://developers.google.com/knowledge-graph/
[2]http://geonames.org/

coordinates of Greece are represented with the following triples: `<Greece> <hasLatitude> "39.00"^^<degrees>` and `<Greece> <hasLongitude> "22.00"^^<degrees>`.

To enrich YAGO2 with more geospatial information, YAGO2geo [6] was developed. YAGO2geo is an extension of YAGO2 with precise geospatial information represented by geometries (including lines, multipolygons etc.) encoded using Open Geospatial Consortium[3] standards. The extension draws geospatial information from multiple sources. First, it utilizes geographical administrative data provided by official sources of three countries: Greece, the United Kingdom and the Republic of Ireland. Additionally, it extracts geospatial information about the administrative units of every country from the Global Administrative Areas dataset (GADM)[4] and introduces information for other types of geographical features, such as lakes, from the OpenStreetMap (OSM)[5]. Apart from geometries, YAGO2geo includes additional information provided by the aforementioned data sources, for instance the population of cities.

Following the evolution of YAGO, YAGO3 [7], was released in 2015. YAGO3 is multilingual, as it combines information from Wikipedias in multiple languages.

The latest version of YAGO, YAGO4 [8], takes its taxonomy from schema.org[6] [9] and the entities from Wikidata [10]. Additionally, its consisent ontology allows semantic reasoning with OWL 2 description logics. Geospatial information in YAGO4 is represented with the `schema:geo` property, which expects its values to be of `schema:GeoCoordinates` and `schema:GeoShape` types. By illustration, the coordinates of Greece are expressed as `schema:geo <geo:38.5,23>`. This representation indicates that each location is represented, again, by a longitude/latitude pair. However, there are various cases in which representing locations with more precise information, like polygons or linestring, seems more reasonable. If we have such information available e.g., about cities, municipalities and rivers, then we can answer questions like *"Which is the city of Germany where two streams meet at a lake?"*, or *"Which are the neighboring municipalities of the municipality of Athens?"*.

Following the approach of YAGO2geo, we seek to extend the YAGO4 knowledge graph with more precise geospatial information derived from the GAG dataset. Such information can be both geometries and other properties such as population, which are not present in YAGO4. The contributions of the paper are the following:

• We carry out a detailed study regarding what kind of administrative information about Greece is available in YAGO4. We find several issues including the inability of representing complex geometries and performing SPARQL queries. We expect similar issues to exist regarding how the administrative divisions of other countries are represented in YAGO4.

• We extend YAGO4 with new classes that capture the current administrative organization of Greece and precise geospatial information about these divisions.

• We point out inadequacies in the representation of certain types of geospatial information (e.g., multilines and multipolygons) in schema.org although these types have been available in standards of the Open Geospatial Consortium for many years.

•We present partial solutions to these inadequacies so that such geospatial information can be represented in YAGO4.

[3] https://www.ogc.org/standards/geosparql

[4] https://gadm.org/

[5] https://www.openstreetmap.org/

[6] https://schema.org/

•We have publicly released our work accompanied with the datasets used.[7]

The rest of the paper is structured as follows: Section 2 discusses related works, while Section 3 gives a detailed overview of the YAGO4 knowledge graph. Section 4 presents the GAG dataset and discusses what kind of administrative information about Greece is already present in YAGO4. Section 5 presents the methodology followed in order to extend YAGO4 with geospatial information. Section 6 concludes the paper and makes certain recommendations for schema.org and YAGO4.

2. Related Work

Geospatial and Temporal Information in YAGO2. YAGO2 integrates a spatial and a temporal dimension into the YAGO knowledge base in a consistent way. Therefore, while YAGO knowledge is encoded in SPO triples, where S is the subject, P is the predicate and O is the object, YAGO2 extends this model and uses SPOTL, where T stands for time and L stands for location. Regarding its spatial dimension, YAGO2 introduces geo-entities to represent the entities with a permanent physical location on Earth. The position of each geo-entity is described by a single geo-coordinate pair, consisting of latitude and longitude values. To that extent, the yagoGeoEntity class was created to serve as the common super class for all geo-entities. Furthermore, the special data type yagoGeoCoordinates was introduced to represent the location coordinates. Each instance of yagoGeoEntity is directly connected to its geographical coordinates by the hasGeoCoordinates relation. Geo-entities are drawn from two main sources: Wikipedia and GeoNames. Wikipedia is a rich source of information, containing numerous spatial entities (like mountains, rivers, cities, etc.) most of them accompanied with their geographical coordinates. The GeoNames dataset was utilized for locations that did not meet the aforementioned criterion. In addition to geographical coordinates, GeoNames provides information regarding location, such as alternate names and neighboring countries. YAGO2 takes advantage of this plethora of information, presenting algorithms that match individual geo-entities existing in both sources, without introducing duplicated entities. The matching process results in over 7 million geo-entities and 320 million new facts from GeoNames that were added to YAGO2.

The knowledge graph YAGO2geo. YAGO2geo [6] was developed with the purpose of extending YAGO2 with precise geospatial information. Here, the new geospatial information is drawn from multiple sources. First, administrative data are derived from the official datasets of three countries: the GAG dataset, the administrative divisions dataset of the United Kingdom and the administrative divisions dataset of the Republic of Ireland. The dataset of the United Kingdom was obtained from the Ordnance Survey[8] and Ordnance Survey Northern Ireland[9], while the dataset of the Republic of Ireland was obtained from Ordnance Survey Ireland[10]. To represent countries from all over world, the latest (2018) version of the GADM is also used. Lastly, YAGO2geo introduces geospatial information from the biggest volunteered, crowdsourced and

[7] https://github.com/msiampou/yago4-ext.

[8] https://www.ordnancesurvey.co.uk/

[9] https://www.nidirect.gov.uk/campaigns/ordnance-survey-of-northern-ireland

[10] https://www.osi.ie/

open dataset with geospatial information, OSM. To avoid introducing duplicate information into YAGO2geo, the authors developed a matching algorithm consisting of two filters: (i) *the label similarity filter* and (ii) *the geometry distance filter*. This approach was based on the methodologies presented in YAGO2 and LinkedGeoData [11]. In detail, the goal of the first filter is to match entities that share similar labels. For two resources to be matched, the string similarity of their respective labels should exceed a predefined threshold th_a. The geometry distance filter is performed as a second step and is applied on the matches produced by the label similarity filter. Its purpose is to eliminate the false matches. More precisely, for every pair of matched entities, the geometry distance filter checks whether the Euclidean distance -in the `WGS:84` coordinate system[11] - between the two entities is less than a specific threshold th_b. While adding more precise geospatial information, YAGO2geo also retains the existing one, including the "old" coordinate pairs. The extension introduces 137 thousand linestrings and 640 thousand polygons and multipolygons, creating a geospatial KG much richer, in terms of geospatial knowledge, compared to DBpedia and Wikidata. YAGO2geo is publicly available[12] and its knowledge can be queried using GeoSPARQL[13] as well as visualised using the linked spatiotemporal data visualization tool `Sextant`[14].

The knowledge graph WorldKG. WorldKG [12, 13, 14] was released in 2021, presenting a new comprehensive geographic knowledge graph built from the OpenStreetMap dataset. Its ontology was created using the key-value pairs of the OSM schema. Each class in the ontology is a subclass of `wkgs:WKGObject` and each entity belonging to this class can be related to a `geo:SpatialObject` via the property `wkgs:spatialObject`. A `geo:SpatialObject` can be either a point, a linestring or a polygon. Namely, the WorldKG entity representing the country of Greece is linked to its respective `geo:SpatialObject` (`wkg:geo432424989`) via the property `wkgs:spatialObject`. The `geo:SpatialObject` represents the entity's type of geometry (`sf:Point`) and the coordinates of the geometry (`POINT(21.9877132 38.9953683)`). To link the ontology of WorldKG with other existing ontologies, the authors developed the Neural Class Alignment (NCA) [12] approach, to obtain the alignments between OSM tags and the classes of the Wikidata and DBpedia knowledge graphs. NCA is an unsupervised framework consisting of two steps. First, a neural classification model is built and trained on a dataset of linked entities in OSM and a selected KG. Then, the resulting model is probed to identify the captured alignments. In the end, the class and tag combinations that are linked, are the ones with class activation threshold higher than a value th_c. To avoid any wrong mappings, the erroneous alignments are manually identified and discarded. The resulting alignments are included in the ontology using the `owl:equivalentClass` property. WorldKG contains more than 820 million triples associated with geographic data for 188 countries and 7 continents. Additionally, the number of geographic entities is two orders of magnitude higher than in Wikidata and DBpedia, being more than 110 million. WorldKG is publicly available[15], and the

[11] A coordinate reference system (CRS) is a coordinate system that is related to an object (e.g., the Earth, a planar projection of the Earth) through a so-called datum which specifies its origin, scale, and orientation. WGS84 is the latest version of the World Geodetic System (WGS) and was established in 1984.

[12] http://yago2geo.di.uoa.gr

[13] http://test.strabon.di.uoa.gr/yago2geo

[14] http://test.strabon.di.uoa.gr/SextantOL3/?mapid=m95dp4hsgkafoe40_

[15] http://www.worldkg.org/

site provides - amongst other things- a WorldKG SPARQL endpoint that supports GeoSPARQL functions.

The spatial dimension of schema.org. Schema.org started as an initiative of Microsoft, Yahoo!, Google and Yandex and it is now a collaborative, community activity. Schema.org improves the Web by providing a structured data markup schema supported by major search engines, that also covers a wide range of topics including people, places, events, products, health and medical types, reviews etc. The idea was to present webmasters with a single vocabulary, making it easier for them to decide on a markup schema and reap the benefits across multiple consumers of the markup. Schema.org was initially launched with 297 classes (or types) and 187 relations (or properties) which over the past few years have grown to 792 and 1447 respectively. The classes are organized into a hierarchy, where each class may have one or more super-classes. Relations are polymorphic, as a relation can have one or more domains and ranges. As for the class hierarchy, it serves more as an organizational tool to assist browsing the vocabulary rather than a "universal ontology". Regarding its spatial dimension, schema.org defines the property `geo` to model the geo-coordinates of a place and its values are expected to be either of a `GeoCoordinates` or a `GeoShape` type. The `GeoCoordinates` type represents a latitude/longitude pair of coordinates of a place or event in the `WGS 84` coordinate system. The `GeoShape` type represents the geographic shape of a place. A `GeoShape` can be described using several properties whose values are based on latitude/longitude pairs. Some of these properties are the `line`, `polygon`, `circle` and `box` properties. Lately, schema.org proposed the full integration of an additional type named `GeospatialGeometry` as a super-type of `GeoShape`, to enable the statement of geospatial relations covered by geospatial standards e.g., by the Open Geospatial Consortium. `GeospatialGeometry` models the topological relations between two geometries as defined in `DE-9IM` model [15]. More precisely, these relations are represented using the following properties: `geoContains`, `geoCoveredBy`, `geoCovers`, `geoCrosses`, `geoDisjoint`, `geoEquals`, `geoIntersects`, `geoOverlaps`, `geoTouches` and `geoWithin`. To conclude, schema.org aims to provide a simple representation, covering properties and types that are frequently used by webmasters. Thus, it does not represent well-known spatial literal formats, (e.g. KML, WKT) complex geometries (e.g. multipolygons, geometry collections) and multiple coordinate reference systems.

3. The YAGO4 Knowledge Graph

Taxonomy. YAGO4 takes advantage of the stable class hierarchy of schema.org, as well as, the fine-grained classes that Wikidata offers. As a result, the top-level classes in YAGO4 come from schema.org combined with bioschemas.org[16] [17] [16]. Additionally, the leafs of the YAGO4 taxonomy correspond to Wikidata classes. Having these inputs, the YAGO4 taxonomy is then constructed as follows. First, for each instance in Wikidata, each possible path in the Wikidata taxonomy to the root node is taken into consideration. If the first class on the path has a

[16] https://bioschemas.org

[17] Bioschemas is a community project built on top of schema.org. Its goal is to improve interoperability in Life Sciences, for resources to better communicate and work with each other by using a common markup on their websites.

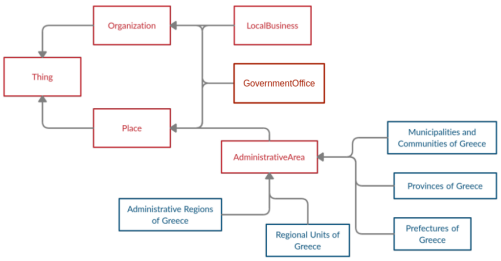

Figure 1: The structure of the administrative organization of Greece in YAGO4 taxonomy. Red color indicates classes that were taken from `schema.org`, whereas blue color indicates Wikidata classes. `GovernmentOffice` class leads to -among others- the instances of the 7 decentralized administrations of Greece.

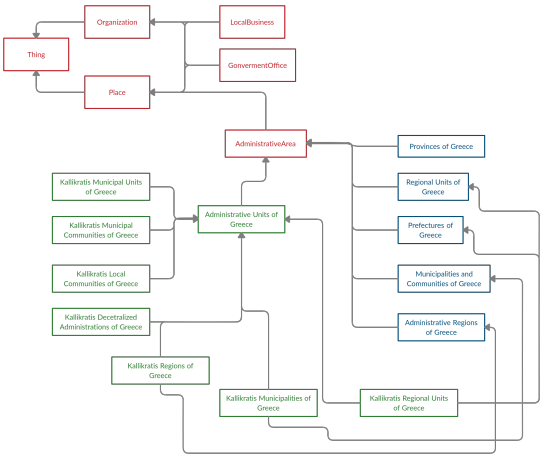

Figure 2: The structure of the administrative organization of Greece in YAGO4 extended taxonomy. Green color indicates the newly inserted classes.

Wikipedia article, it is then included in YAGO4. The path to the root in the Wikidata taxonomy is continued, discarding all classes on the way, until a class that has been mapped to schema.org is found. The path to the root in the schema.org taxonomy is then continued, adding all classes on the way to YAGO4. If a class that has been mapped to schema.org is not found, the entire path is discarded. All Wikidata classes that have less than 10 direct instances are also discarded. The final taxonomy contains only 10 thousand classes out of the 2.4 million original Wikidata ones, shrinking the taxonomy by 99.6%. For our purposes, we focus on the administrative organization of Greece, whose structure in the taxonomy in YAGO4 can be seen in Figure 1.

Entities. YAGO4 is stored in RDF data format and all entities consist of human-readable URIs to make the KB more accessible for interactive use. The name of each entity corresponds to its Wikipedia page title. For the cases that an entity does not have a corresponding Wikipedia page, its name is the concatenation of its English label with its Wikidata identifier. For example, the name of the entity which describes the Region of Crete is `Crete_Region_Q1267522`. If an English label is not available, then the entity is named after its Wikidata identifier. The methodology followed by the authors of YAGO4, assigns human-readable names to the vast majority of entities, without introducing duplicates or ambiguity.

Well-Typed Values. YAGO4 defines well-typed literals, having each data values of Wikidata translated to RDF terms. Furthermore, globe coordinates are mapped to `schema:GeoCoordinates` resources, expressing location as a longitude/latitude pair. By illustration, the municipality of Andros is represented as `<http://yago-knowledge.org/resource/Andros> <http://schema.org/geo> <geo:37.83333333333409, 24.93333333333383>`. In Section 5.4, we introduce our methodology to express the geographic coordinates of each location with more complex geometries such as polygons, lines and multi-polygons.

Relations and Constraints. The relations in YAGO4 come from schema.org and are mapped to Wikidata predicates. Moreover, YAGO4 has hand-crafted semantic constraints for ensuring

Table 1
Comparison of the number of instances of each class related to the administrative divisions of Greece in GAG and YAGO4.

Administrative Unit	# instances in GAG	# instances in YAGO4
Decentralized Administrations	7	7
Regions	13	13
Regional Units	74	77
Prefectures	0	55
Provinces	0	70
Municipalities	325	340
Municipal Units	1034	0
Municipal Communities	3	0

the quality of the data and allowing logical reasoning. These constraints are modeled in the W3C standards SHACL [17] and OWL. In general, YAGO4 assumes that no other properties are allowed for each class, thereby interpreting the SHACL constraints under a "closed world assumption". These constraints are automatically enforced during the construction of the KB, hence the data of YAGO4 satisfy all constraints. Lastly, the generated YAGO4 ontology uses the OWL 2 axioms `DisjointClasses`, `ObjectPropertyDomain`, `DataPropertyDomain`, `ObjectPropertyRange`, `DataPropertyRange`, `ObjectUnionOf`, `FunctionalDataProperty`, `FunctionalObjectProperty`, and falls into the OWL DL subset. In our extension, we added several extra properties to YAGO4, presented in Section 5.2.

4. The administrative organization of Greece

The Kallikratis and Kleisthenis Programme. The current administrative organization of Greece is the result of the Kallikratis Programme[18] and its recent reform, the Kleisthenis Programme[19] that came into effect in 2019. According to them, the first level of government consists of municipalities which are further subdivided into municipal units. Furthermore, municipal units consist of municipal and local communities. The second level of government is composed by regions that are divided into regional units. A group of municipalities composes a regional unit. The third level consists of decentralized administrations that consist of a group of regions. There are, in total, 7 decentralized administrations, 13 regions, 74 regional units and 325 municipalities in Greece. Regarding the last ones, the Kleisthenis Programme introduced the creation of 12 new municipalities, increasing their total number from 325 to 332. Additionally, in contrast to the administrative organization of the country (1997 Kapodistrias Reform[20]), provinces are totally abolished and prefectures are replaced by regions.

Administrative divisions of Greece instances in YAGO4. After evaluating the quality of the instances of the classes related to the administrative divisions of Greece in YAGO4, we noticed that the regional units are scattered between prefectures and regional units, while some of them are instances of both classes. For example, `Pella_(regional_unit)` is both a type of `Regional_units_of_Greece` and a type of `Prefectures_of_Greece`. Additionally,

[18] https://en.wikipedia.org/wiki/Kallikratis_Programme
[19] https://el.wikipedia.org/wiki/Πρόγραμμα_Κλεισθένης_Ι
[20] https://en.wikipedia.org/wiki/Kapodistrias_reform

the class of Provinces_of_Greece, leads to 70 instances of past-existing provinces of Greece, which possibly still exist in Wikidata, such as Imathia_Province. Table 1 compares the number of instances of each class related to the administrative divisions of Greece in GAG and YAGO4. We denote that the YAGO4 taxonomy is not fully consistent with the administrative organization of Greece. In particular, there are three classes representing provinces, prefectures and municipalities and communities of Greece in YAGO4, while classes representing municipal units, municipal communities and local communities do not exist at all. Nevertheless, contrasting the number of instances of each YAGO4 class with the number of instances of their respective GAG class, we observe that the classes of YAGO4 contain more entities than their corresponding official administrative divisions, mostly due to duplication (e.g., regional units and prefectures) as well as outdated information (e.g., provinces). However, YAGO4 contains the information about the new municipalities introduced with the Kleisthenis reform. Lastly, the number of instances of municipal units, local and municipal communities are non-existent due to the taxonomy deficiencies that were mentioned.

5. Extending YAGO4

Taxonomy Extension. As mentioned in Section 3.1, the part of YAGO4 taxonomy pertaining to the administrative divisions of Greece is not fully consistent with the administrative organization of Greece. We addressed this problem by introducing new leaf classes in YAGO4. More precisely, we propose the insertion of a super class that represents the administrative units of Greece and is a subclass of the AdministrativeArea. We further inserted 7 additional classes that represent each of those administrative units separately. Each class is a subclass of the Administrative_Units_of_Greece class, as well as a subclass of its respective YAGO4 class, provided it exists. For example, the Regions_of_Greece is both a subclass of Administrative_Units_of_Greece and Administrative_Regions_of_Greece, whereas Municipal_Units_of_Greece is only a subclass of Administrative_Units_of_Greece. This, allows us to extend YAGO4 with missing entities without violating its taxonomy and constraints. The part of the extended taxonomy is illustrated in Figure 2.

Addition of Properties. The available properties of YAGO4 do not allow us to represent useful information of administrative units. Hence, we extend the taxonomy of YAGO4 with the object property has_seat which connects a municipality with a municipal or local community and the data properties has_population and has_code, without violating the strict constraints that are in place.

Matching Phase. In order to extend each pre-existing YAGO4 entity with qualitative geospatial information, we tried to match each entity with one in the GAG dataset. For example, the YAGO4 entity Athens_Municipality_Q1224979 and the GAG entity with identifier 9186, represent the same location, therefore they should be matched. For this purpose, we used the (i) *label similarity filter* and (ii) the *geometry distance filter*, as described on Section 2 and [6].

Results. Our methodology enabled us to match most of the entities existing in GAG dataset. In fact, many GAG entities were matched with more than one entities, yielding the existence of duplicate entities in YAGO4. For that reason, the number of YAGO4 matches in Table 2 is greater than the number of GAG matches. More precisely, all decentralized administration and region

Table 2
Comparison between the number of instance matches in GAG and YAGO4.

GAG Classes	YAGO4 Classes	Number of matched GAG entities
Decentralized Administrations	Government Offices	7/7
Regions	Regions of Greece	13/13
Regional Units	Regional Units and Prefectures of Greece	67/74
Municipalities	Municipalities and Communities of Greece	322/325

entities have been matched with exactly one YAGO4 entity. In addition, 67 regional units and 325 municipalities were matched with a YAGO4 entity, while 11 regional unit and 9 municipality duplicate instances were spotted. Finally, due to the absence of the municipal units, local and municipal communities classes, the corresponding GAG entities failed to be matched. For our extension purposes, we discard duplicate entities, as we do not wish to introduce duplicate information in the extended knowledge graph.

Comparison with previous work. With regards to the GAG dataset, in this extension we managed to achieve a greater number of matches, on average, compared to YAGO2geo. More precisely, in YAGO2geo we managed to match the majority of the decentralized administration and region entities, but not all of them. Additionally, exactly 21 regional unit entities were matched. However, the number of matches regarding municipalities, municipal units and communities beat our current results. The different outcomes in both cases are mostly due to the different class hierarchies between the two versions of YAGO. In YAGO2 the information related with the administrative divisions of Greece lie upon the `geoclass_administrative_division`, first to fifth `geoclass_X-order_administrative_division` and `populated_place` and `locality` classes. As we can observe, the taxonomy of YAGO4, which was presented in Section 3.1, is more compatible with the current administrative organization of Greece, in contrast with the taxonomy of YAGO2, a fact that renders the detection of the desired entities easier. In addition, YAGO2 is built automatically from Wikipedia, GeoNames and WordNet, resulting in a KB that contains 447 million facts about 9.8 million entities. Yet, YAGO4, which is entirely built of the rich instance data of Wikidata, contains 2 billion type-consistent triples for 64 million entities. According to the design choices made for the two versions of YAGO, as well as the results from our works, we came to the following conclusions. First, the high number of unmatched regional units entities in YAGO2 is due to their absence from the knowledge graph. Second, YAGO4 contains many duplicate entities, especially regional units and municipalities. For example `<https://yago-knowledge.org/resource/Andros>` and `<https://yagoknowledge.org/resource/Andros_(town)>` are both referring to Andros Municipality. Last but not least, there is a lack of information regarding municipal communities in YAGO4, resulting from the absence of the relating Wikidata instances.

Location Coordinates. As previously mentioned, the geospatial information in YAGO4 is represented with the `schema:geo` property, which expects its values to be of `schema:GeoCoordinates` and `schema:GeoShape` types. A `GeoShape` can be described using several properties whose values are based on latitude/longitude pairs. The properties of `GeoShape`, that are related to our task, are the `line`, `polygon`, `circle` and `box`. In general, YAGO4 maps its coordinates to `schema:GeoCoordinates` resources. Regarding the `GeospatialGeometry` type

```
POLYGON((
21.71163060727338 40.85096274759539,
21.71163660172085 40.85091782978087,
... ,
21.71163060727338 40.85096274759539))
```

Figure 3: WKT polygon

```
"21.71163060727338,40.85096274759539
 21.71163660172085,40.85091782978087
 ...
 21.71163060727338,40.85096274759539"
```

Figure 4: schema polygon

mentioned in Section 2.4, it is not integrated in the YAGO4 knowledge graph. However, the geographic information in the GAG dataset is expressed in WKT format, using polygons and multipolygons to indicate the area of each location. Since schema.org does not, yet, support complex geometries like these supported by WKT, YAGO4 inherits this shortcoming. To make the geographic information provided by GAG compatible to the design of schema.org, we enriched the spatial dimension of YAGO4 only with polygons by converting WKT polygons to `schema:polygon` in the obvious way. Figures 3 and 4 present an example of such conversion. Yet, since multipolygons represent the general case of mapped areas, the inability to convert such geometries to a schema.org type indicates a strong limitation.

There are two ways according to which we could somehow address this issue. The first approach is to split the multipolygons into separate shapes. Since a multipolygon is a multi-surface whose elements are polygons, we could divide each geometry to multiple polygons. However, schema.org does not allow logical `AND` and `OR` operators over lists of type `GeoShape`. To address this issue the addition of the `excludesGeoShape` property has been suggested[21]. The insertion of this property would provide a mechanism for excluding an area using the supported shapes. For our case, we could use lists of values to denote the aggregation of polygons and utilize the `excludesGeoShape` property to exclude areas and therefore define our multipolygons. Even though this approach would assist the modeling of complex geometries, it also suggests an extra effort from the webmasters in regards to representing their data. This suggestion hasn't yet been accepted in the schema.org repository. The second approach is to model our multipolygons using the `additionalProperty` property on `GeoShape`. Generally, schema.org offers the property `additionalProperty` to represent additional characteristics of an entity. `additionalProperty` expects its values to be of `PropertyValue` type, indicating a property-value pair. Although this approach would enable the addition of extra geographical information to the entities of the YAGO4, this information will not end up being properly exploited.

While both of the approaches suggest a way of adding geographical information represented by complex geometries into the YAGO4, they still cannot address the issues of processing these geometries. More precisely, due to the fact that schema.org has its own geometry serialization and does not adopt one of the widely used ones (eg., WKT, GeoJSON), common GeoSPARQL queries that can be performed in WordKG and YAGO2geo, cannot be performed in YAGO4. Listing 1 illustrates such an example. This query asks for the municipalities whose geometries intersect with the geometry of the municipality of Athens. Taking all the above into consideration, even though our methodology has resulted in the augmentation of YAGO4 with geospatial

[21] https://github.com/schemaorg/suggestions-questions-brainstorming/issues/210

information, this information cannot be queried by GeoSPARQL given the incompatibility of the data models of schema.org (hence YAGO4) and GeoSPARQL.

6. Conclusions and Recommendations

In this work, we presented an extension of YAGO4 with more precise geospatial information regarding the administrative organization of Greece. Through our study we were able to uncover certain issues that exist with the representation and querying of geospatial information in schema.org and therefore YAGO4. More precisely, even though schema.org provides some spatial modeling, such as representing point locations and providing widely used shapes like polygons and lines, it does not cater for well-known spatial literal formats, complex geometries and multiple coordinate reference systems. Consequently, utilizing the class hierarchy of schema.org resulted in a less powerful geometry representation in the YAGO4 ontology. Currently, there are several open issues[22] [23] [24] on the schema.org repository, discussing the importance of supporting the processing of complex geometries as well as ways of integrating that support. If the community of schema.org decides to move forward with the suggested changes, YAGO4 will be able to overcome all the aforementioned limitations and extensions like the one presented here will be easily developed. Even though schema.org chooses a more simple representation for general use, the interest in spatial data has been increasing, and so does the need for shared vocabularies including also this case. We believe that the W3C Schema.org Community Group[25] should consider addressing the aforementioned limitations, contributing into making geospatial data widely discoverable and accessible.

Listing 1: GeoSPARQL query in YAGO2geo

```
PREFIX uom: <http://www.opengis.net/def/uom/OGC/1.0/>
PREFIX geo: <http://www.opengis.net/ont/geosparql#>
PREFIX geof: <http://www.opengis.net/def/function/geosparql/>
PREFIX y2geo: <http://kr.di.uoa.gr/yago2geo/ontology/>
SELECT ?x ?y
WHERE {
         ?x y2geo:hasGADM_Name "Athens"@en.
         ?x rdf:type y2geo:GAG_Municipalities.
         ?x geo:hasGeometry ?xGeom.
         ?xGeom geo:asWKT ?xWKT .
         ?y rdf:type y2geo:GAG_Municipalities .
         ?y geo:hasGeometry ?yGeom .
         ?yGeom geo:asWKT ?yWKT .
FILTER (geof:sfIntersects(?xWKT, ?yWKT)) }
```

Acknowledgments

The research work was supported by the Hellenic Foundation for Research and Innovation

[22] https://github.com/schemaorg/schemaorg/issues/1375
[23] https://github.com/ESIPFed/science-on-schema.org/issues/105
[24] https://github.com/schemaorg/schemaorg/issues/1548
[25] https://www.w3.org/community/schemaorg/

(H.F.R.I.) under the "First Call for H.F.R.I. Research Projects to support Faculty members and Researchers and the procurement of high-cost research equipment grant" (Project Number: HFRI-FM17-2351).

References

[1] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, Z. Ives, Dbpedia: A nucleus for a web of open data, ISWC'07/ASWC'07, Springer-Verlag, Berlin, Heidelberg, 2007, p. 722–735.

[2] F. M. Suchanek, G. Kasneci, G. Weikum, Yago: A core of semantic knowledge, WWW '07, Association for Computing Machinery, New York, NY, USA, 2007, p. 697–706.

[3] G. A. Miller, Wordnet: A lexical database for english, Communications of the ACM 38 (1995) 39–41.

[4] J. Hoffart, F. M. Suchanek, K. Berberich, G. Weikum, YAGO2: A spatially and temporally enhanced knowledge base from wikipedia, Artif. Intell. 194 (2013) 28–61.

[5] J. Hoffart, F. M. Suchanek, K. Berberich, G. Weikum, YAGO2: A spatially and temporally enhanced knowledge base from wikipedia: Extended abstract, IJCAI/AAAI, 2013, pp. 3161–3165.

[6] N. Karalis, G. Mandilaras, M. Koubarakis, Extending the YAGO2 knowledge graph with precise geospatial knowledge, in: ISWC '19, volume 11779 of *Lecture Notes in Computer Science*, Springer, 2019, pp. 181–197.

[7] F. Mahdisoltani, J. Biega, F. M. Suchanek, Yago3: A knowledge base from multilingual wikipedias., in: CIDR, 2015.

[8] T. Pellissier Tanon, G. Weikum, F. Suchanek, Yago 4: A reason-able knowledge base, in: The Semantic Web, Springer International Publishing, Cham, 2020, pp. 583–596.

[9] R. V. Guha, D. Brickley, S. Macbeth, Schema.org: Evolution of structured data on the web, Commun. ACM 59 (2016) 44–51.

[10] D. Vrandečić, M. Krötzsch, Wikidata: A free collaborative knowledgebase, Commun. ACM 57 (2014) 78–85.

[11] C. Stadler, J. Lehmann, K. Höffner, S. Auer, LinkedGeoData: A core for a web of spatial open data, Semant. Web 3 (2012) 333–354.

[12] A. Dsouza, N. Tempelmeier, E. Demidova, Towards Neural Schema Alignment for OpenStreetMap and Knowledge Graphs, in: ISWC '21, volume 12922 of *Lecture Notes in Computer Science*, Springer, 2021, pp. 56–73.

[13] A. Dsouza, N. Tempelmeier, R. Yu, S. Gottschalk, E. Demidova, WorldKG: A World-Scale Geographic Knowledge Graph, in: CIKM '21, ACM, 2021.

[14] N. Tempelmeier, E. Demidova, Attention-Based Vandalism Detection in OpenStreetMap, in: WWW '22, ACM, 2022.

[15] E. Clementini, P. D. Felice, P. van Oosterom, A small set of formal topological relationships suitable for end-user interaction, in: SSD '93, volume 692 of *Lecture Notes in Computer Science*, Springer, 1993, pp. 277–295.

[16] A. J. G. Gray, C. A. Goble, R. Jimenez, Bioschemas: From potato salad to protein annotation, in: ISWC '17 (Posters, Demos & Industry Tracks), 2017.

[17] Shapes constraint language (SHACL), Technical Report, W3C, 2017. URL: https://www.w3.org/TR/shacl/.