RDF Reasoning on Large Ontologies: A Study on Cultural Heritage and Wikidata

Nuno Freire[1] (✉) and Diogo Proença[1,2]

[1] INESC-ID, Rua Alves Redol 9, 1000-029 Lisbon, Portugal
{nuno.freire,diogo.proenca}@tecnico.ulisboa.pt
[2] IST, Universidade de Lisboa, Avenida Rovisco Pais, 2, 1049-001 Lisbon, Portugal

Abstract. Large ontologies are available as linked data, and they are used across many domains, but to process them considerable resources are required. RDF provides automation possibilities for semantic interpretation, which can lower the effort. We address the usage of RDF reasoning in large ontologies, and we test approaches for solving reasoning problems, having in mind use cases of low availability of computational resources. In our experiment, we designed and evaluated a method based on a reasoning problem of inferring Schema.org statements from cultural objects described in Wikidata. The method defines two intermediate tasks that reduce the volume of data used during the execution of the RDF reasoner, resulting in an efficient execution taking on average 10.3 ± 7.6 ms per RDF resource. The inferences obtained in the Wikidata test were analysed and found to be correct, and the computational resource requirements for reasoning were significantly reduced. Schema.org inference resulted in at least one *rdf:type* statement for each cultural resource, but the inference of Schema.org predicates was below expectations. Our experiment on cultural data has shown that Wikidata contains alignment statements to other ontologies used in the cultural domain, which with the application of RDF and OWL reasoning can be used to infer views of Wikidata expressed in cultural domain's data models.

Keywords: Data volume · Reasoning · Wikidata · Schema.org · Semantic web

1 Introduction

Nowadays, large ontologies are available as linked data and with open licenses that allow for their reuse in a wide variety of applications across all domains of knowledge. Some examples are DBpedia[1] and Wikidata[2]. The usefulness of these ontologies is clearly acknowledged in many domains, but due to their high data volume, their reuse requires the commitment of considerable human resources for acquiring the knowledge about the ontologies' data models and for the development of the information systems for their processing.

[1] https://wiki.dbpedia.org/.

[2] https://www.wikidata.org/wiki/Wikidata:Main_Page.

Published by Springer Nature Switzerland AG 2020
I. Maglogiannis et al. (Eds.): AIAI 2020, IFIP AICT 583, pp. 381–393, 2020.
https://doi.org/10.1007/978-3-030-49161-1_32

The Resource Description Framework (RDF) provides possibilities for automation of data processing and semantic interpretation that can lower the effort for reusing large ontologies and for the development of general-purpose data processing tools. In our work, we address the usage of RDF reasoning for automated discovery of new facts about the ontologies' data.

RDF reasoning processes are based on a set of rules. Both the RDF Schema (RDF(S)) and Web Ontology Language (OWL) specifications include entailment rules to derive new statements from known ones. RDF(S) entailments are included in its semantics specification [1]. OWL has specifications for its RDF-based semantics [2] and its direct semantics [3], which include an extensive set of entailment rules.

In cultural heritage, the domain where we conduct our research, semantic data is highly valued and applied for descriptions of cultural objects. Reasoning is often mentioned and implemented but is mostly put into practice with pragmatic implementations that use ad-hoc data processing and querying of triple stores or SPARQL endpoints.

Our research focuses on defining a method for reasoning on large ontologies that can be systematically applied to varied reasoning contexts. We studied the problem using Wikidata as the target ontology for reasoning, and a reasoning problem with potential application in culture domain. Our approach aims to be lightweight, due to the lower capacity of information technologies employed by most cultural institutions, in contrast to other domains.

Our work provides three main scientific contributions:

- It identifies some limiting computational aspects of applying RDF reasoning to large volumes of data;
- It defines, tests and evaluates a method for RDF reasoning in large ontologies;
- It provides observations and evidence of Wikidata's potential to provide alignments from its data modes to other ontologies which can, with the use of RDF and OWL reasoning, be used to infer views of Wikidata expressed in cultural domain's ontologies.

We follow, in Sect. 2, by describing related work on linked data and reasoning, and also the research on reasoning in the cultural domain. Section 3 presents our proposed method for reasoning in large ontologies. The setup for the evaluation on Wikidata of our method is presented in Sect. 4. Section 5 presents the results from the evaluations and their analysis. Section 5 finalizes by summarizing the method, highlights the conclusions of the study and describes future work.

2 Related Work

Linked data has a large diversity of research topics related to our work. Scalability is one of the most addressed topics, with many facets such as indexing, federated querying, aggregation and reasoning. The reuse of published linked data by third parties has revealed data quality to be a challenge as well, both at the level of semantics and at the level of syntax [4–6]. Reuse of linked data is one of the concerns of our work, in which data quality is a relevant aspect. In this area, significant work has been done to facilitate the reuse of linked data by aggregation and data cleaning [7, 8].

Reasoning on linked data is also an active research topic. A comprehensive analysis and description of techniques has been published in [9]. Regarding the particular aspect of scalable reasoning, the related work employs techniques based on high computational capacity [10–13], which are beyond the capacity of most cultural institutions. The application of reasoning in large volumes of RDF data was addressed by [14], but with a different target use case – data streams, which is a problem with characteristics that differ from those of reasoning in large ontologies.

Regarding cultural heritage, although the use of linked data has been the focus of research, most of the published work addresses mainly the aspect of the publication of linked data [15–17]. Large ontologies have been published and are maintained by cultural organizations, mostly by national libraries that have built and maintained these ontologies to support the information needs from bibliographic data. This ontology development has been a long-term practice, and started much earlier than when the semantic web emerged. However, the scalability of the application of RDF(S) or OWL reasoning on cultural ontologies has not been studied.

3 Method

We are addressing reasoning problems where an ontology is used to make inferences about a target dataset. To cope with the large amount of data for reasoning, we tested the reduction of data volume used for reasoning (we will mention in Sect. 4 the approaches we attempted but were unable to run).

Subsection 3.1 describes the general process of our approach, including the main tasks, software components and data flow. Subsection 3.2presents how we applied and evaluated the process in an experiment on Wikidata.

3.1 The Process

Figure 1 shows an overview of the process followed in our method. It consists of the following tasks:

1. A domain expert defines the reasoning problem. The expert makes the following specifications:

    1.1. Specification of the triple patterns required from the ontology for the target reasoning ruleset. This specification allows the RDF reasoner to reduce the number of statements used during the reasoning process;
    1.2. Specification of the SPARQL endpoint(s) of the ontology(ies). The endpoints are used to collect the triple patterns specified in the previous item;
    1.3. Specification of the RDF reasoning ruleset to be applied;
    1.4. Define partitions of the target dataset into sub datasets where the reasoning problem can be applied independently from the rest of the dataset. This specification allows the execution of the RDF reasoner to be done on less data by executing it independently on each of the dataset partitions.

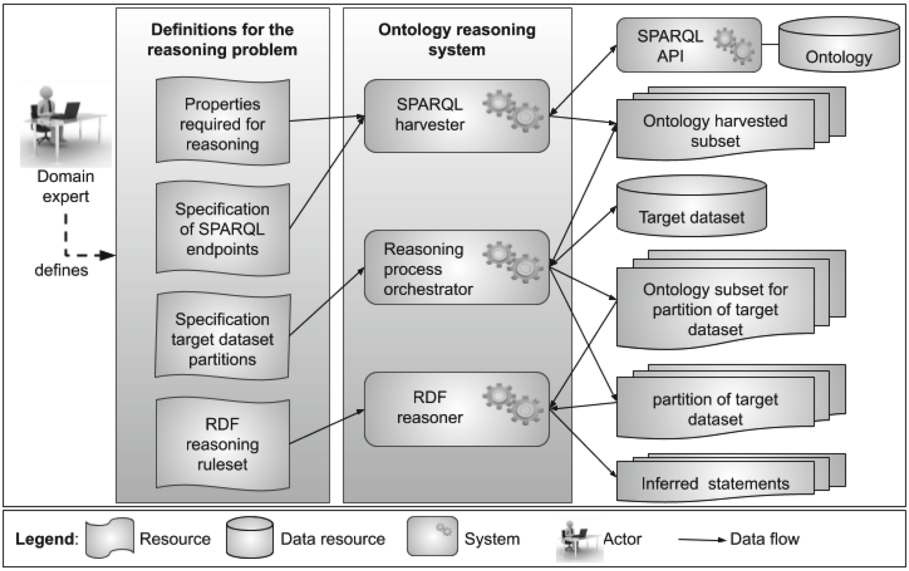
Definitions for the reasoning problem
Ontology reasoning system
Domain expert
defines
Properties required for reasoning
Specification of SPARQL endpoints
Specification target dataset partitions
RDF reasoning ruleset
SPARQL harvester
Reasoning process orchestrator
RDF reasoner
SPARQL API
Ontology
Ontology harvested subset
Target dataset
Ontology subset for partition of target dataset
partition of target dataset
Inferred statements
Legend:
Resource
Data resource
System
Actor
Data flow

Fig. 1. Overview of the applied method for RDF reasoning in large ontologies.

2. RDF reasoning software, adapted to this process, executes the reasoning process following the specifications prepared by the domain expert:

   2.1. A SPARQL harvester collects all triples from the ontology that use the specified triple patterns, storing them in a local triple store;

   2.2. The RDF reasoner applies the reasoning ruleset to each sub dataset:

      2.2.1. Before applying the reasoning rules, this RDF reasoner further reduces the subset of harvested statements from the ontology, selecting all resources about the specified properties, and all subjects and objects of statements using these properties as predicates. This selection of statements is applied recursively to all referred resources;

      2.2.2. The reasoner executes using the ontology as supporting data, and the target subset as the main model for reasoning where all inferred statements are added;

3.2 Evaluation on Wikidata

We applied the reasoning method using Wikidata as the target ontology. The tested reasoning problem is for a use case of the culture domain. Wikidata contains RDF statements that align its classes and properties to several other ontologies. Alignments are not available for all classes and properties but their availability is high and has shown potential to support automatic interpretation of Wikidata [18]. One of these ontologies is Schema.org, which is also applied for cultural heritage objects.

For evaluating our method, we defined a use case that can be solved by RDF reasoning on Wikidata and Schema.org. We formulate it as follows: *"as a data re-user, I would like to obtain RDF data about a Wikidata entity represented as Schema.org"*.

Although Wikidata makes some use of Schema.org for its RDF output, it is only used for a limited set of properties that include only human-readable labels [18]. Wikidata's RDF output predominantly uses Wikidata's properties and classes, and Wikibase classes (Wikibase is the software on which Wikidata runs). The reasoning rules defined by RDF Semantics [1] and OWL [2, 3], enable the inference of Schema.org statements. By reasoning on the alignment statements that exist in Wikidata's RDF resources for its classes and properties, combined with the statements in RDF resources of Schema.org and Wikidata classes and properties, it is possible to infer the Schema.org properties and also the *rdf:type* properties using Schema.org classes as object.

The subset of RDF(S) and OWL reasoning rules that is required for our use case, and which we have setup in our RDF reasoner, are listed in Table 1[3]. To solve the reasoning problem, the reasoner requires that the statements match any of the triple patterns that appear in the rule trigger conditions, therefore for our use case, the reasoner requires data from Wikidata and the definition of OWL itself. The reasoner also requires these triple patterns from Schema.org in order to fulfil our use case. Note that RDF resources from RDF(S) do not have to be included. This is because RDF(S) is the foundational semantics for reasoning, therefore the base implementations of reasoners along with the RDF(S) inference rules have all the implicit meaning required for reasoning.

Our SPARQL harvester collected all statements using the required triple patterns from Wikidata and stored them locally in a triple store. Similarly, for Schema.org we collected the statements according to the triple patterns but given the much smaller size of Schema.org we simply harvested them from Schema.org's OWL definition file[4]. For the applied ruleset, we collected statements with the following properties: *rdfs:subclassOf, rdfs:subPropertyOf, owl:equivalentProperty, owl:equivalentClass* and *owl:sameAs*. For all the resources appearing as subjects in the harvested statements, we also harvest their *rdf:type* statements.

It is important to point out that Wikidata's RDF output is using almost exclusively Wikidata's properties and classes. In an earlier study, where we analysed Wikidata's RDF output about cultural heritage resources [18], we have observed only two properties from RDF in use: *rdf:type* and *rdf:label*. In the form that Wikidata's RDF is available, the application of the reasoning rules of RDF(S) and OWL would not be triggered. But Wikidata defines equivalent properties for all the necessary properties to perform the required reasoning, therefore, we harvested these equivalent properties instead of the RDF(S) and OWL ones.

The RDF resources of Wikidata's properties do not state their equivalence to RDF(S) and OWL. To allow the reasoning rules to trigger on the Wikidata properties, we added

[3] For readability purposes, in this text we abbreviate namespaces as follows: rdf for http://www.w3.org/1999/02/22-rdf-syntax-ns#; rdfs for http://www.w3.org/2000/01/rdf-schema#; owl for http://www.w3.org/2002/07/owl#; schema for http://schema.org/; wdt for http://www.wikidata.org/prop/direct/ .

[4] Schema.org OWL definition is made available at https://schema.org/docs/developers.html . Its web page states that currently, it is in an experimental stage.

Table 1. The subset of RDFS and OWL reasoning rules required for our use case.

Source of rule	Rule trigger condition	Rule entailment
RDFS	`(? A ?P ?B), (?P rdfs:subPropertyOf ?Q)`	`(?A ?Q ?B)`
RDFS	`(?X rdfs:subClassOf ?Y), (?A rdf:type ?X)`	`(?A rdf:type ?Y)`
OWL	`(?P owl:equivalentProperty ?Q)`	`(?P rdfs:subPropertyOf ?Q), (?Q rdfs:subPropertyOf ?P)`
OWL	`(?P rdfs:subPropertyOf ?Q), (?Q rdfs:subPropertyOf ?P)`	`(?P owl:equivalentProperty ?Q)`
OWL	`(?P owl:sameAs ?Q), (?P rdf:type rdf:Property), (?Q rdf:type rdf:Property)`	`(?P owl:equivalentProperty ?Q)`
OWL	`(?P owl:equivalentClass ?Q)`	`(?P rdfs:subClassOf ?Q), (?Q rdfs:subClassOf ?P)`
OWL	`(?P owl:sameAs ?Q), (?P rdf:type rdfs:Class), (?Q rdf:type rdfs:Class)`	`(?P owl:equivalentClass ?Q)`
OWL	`(?A owl:sameAs ?B) (?B owl:sameAs ?C)`	`(?A owl:sameAs ?C)`
OWL	`(?X owl:sameAs ?Y), (?X rdf:type owl:Class)`	`(?X owl:equivalentClass ?Y)`
OWL	`(?X owl:sameAs ?Y), (?X rdf:type rdf:Property)`	`(?X owl:equivalentProperty ?Y)`

owl:equivalentProperty statements to the harvested dataset. Table 2 lists the equivalent properties and the respective alignment statements. With the alignment statements in the data available for reasoning, the RDFS and OWL rules make complete inferences.

Once we reach this step, the reasoner's setup is complete and the reasoner is ready to be executed on Wikidata entities. For this evaluation, we identified Wikidata resources about cultural heritage objects by querying its SPARQL API, and checking for Wikidata entities containing the property *wdt:P727* (Europeana ID[5]). This property stands for the identifier assigned by Europeana to cultural heritage objects described in its dataset, therefore we consider it a reliable form of identifying cultural heritage objects in Wikidata. We collected a total of 11,928 resources from Wikidata in our sample and executed the reasoner.

We partitioned the sample by individual RDF resources that represent a cultural heritage object. We applied the final tasks of our reasoning to each partition. All the data from the ontologies required for the reasoning problem is collected first. For each partition, we selected from the ontologies all RDF resources on the predicates present in the

[5] https://www.wikidata.org/wiki/Property:P727.

Table 2. The property alignment statements we applied to allow the reasoning rules to trigger on the Wikidata properties.

Wikidata property	Alignment statements
Equivalent class (*wdt:P1709*)	`(wdt:P1709 owl:equivalentProperty owl:equivalentClass)`
Equivalent property (*wdt:P1628*)	`(wdt:P1628 owl:equivalentProperty owl:equivalentProperty)`
Subclass of (*wdt:P279*)	`(wdt:P279 owl:equivalentProperty rdfs:subClassOf)`
Subproperty of (*wdt:P1647*)	`(wdt:P1647 owl:equivalentProperty rdfs:subPropertyOf)`
Instance of (*wdt:P31*)	`(wdt:P31 owl:equivalentProperty rdf:type))`
External superpropery (*wdt:P2235*)	`(wdt:P2235 owl:equivalentProperty rdfs:subPropertyOf)`

partition, and all RDF resources of subjects and objects of the selected statements. This selection of statements is applied recursively to all referred RDF resources ensuring that the reasoner will execute with all necessary statements from Wikidata and Schema.org properties, which are required for an individual partition.

Finally, the reasoner was executed. We collected all the inferred statements and logged the running time of the reasoner. We create a data profile of the inferred statements for analysis. The results and their analysis are presented in Sect. 4.

4 Evaluation Results

For evaluating our method, we measured the number of statements used for reasoning at three stages: (1) the original ontologies; (2) after selecting triples necessary for the reasoning rules; and (3) for the statements about the cultural heritage objects. The reasoner execution time was also measured at the same three stages. Our third evaluation measured the number and characterization of the statements inferred in the final result.

Our experiments applied the same execution environment for all tests. It ran in a server with a Intel(R) Core (TM) i7-3770 CPU at 3.40 GHz. We have not applied any parallel processing, therefore the experiments run with one thread only, and the Java runtime environment was set for a limit of 16 GB memory usage. The software we implemented to support the experiment was a Java application that used Apache Jena[6] for the required RDF processing components: the RDF reasoner, triple store and RDF programming interface.

Table 3 summarizes the results we obtained on the number of statements for reasoning at the three stages. It breaks down the result by the three ontologies we used for the study on Wikidata. We cannot estimate the reduction of statements as a percentage of the

[6] https://jena.apache.org/.

original collection because the number of statements for Wikidata is unknown to us, but clearly it is a very small fraction of the original size, judging by its final average obtained for the final selection of triples and from the measurements obtained from Schema.org and OWL.

Table 3. The results in number of statements used for reasoning at three stages of the analysis: the original ontologies, after selecting triples necessary for the reasoning rules, and for the statements about the cultural heritage objects.

Ontology	Original size	Subset for rules	Subset for reasoning on an RDF resource (average per RDF resource)
Wikidata	unknown (latest dump file in Turtle format has 16 GB compressed)	3,010,212 (3,009,062 about classes plus 1,150 about properties)	256.9 ± 73.4
Schema.org	29.715	984	12.8 ± 5.3
OWL	450	6	2.0 ± 0.1
Total RDF statements	Unknown	3,011,202	271.8 ± 77.4

We attempted to execute the reasoner at all the three stages of the experiment and measured the execution time for each one. With the available computational resources, it was not possible to successfully execute it whenever too many statements from the ontologies were used. Using the complete ontologies exceeded the memory capacity. After reducing the reasoning data to the necessary statements for the reasoning rules, the reasoning time was too long for real-world applicability.

Reasoning ran successfully when executed at the final stage of our method where the ontologies statements used were the specific ones required for the reasoning rules and the RDF resource that was the target of reasoning. Table 4 presents the results for the time taken to execute the RDF reasoner, broken down in two operations: (1) the selection of statements from the ontologies; and (2) the execution of the RDF reasoner. For our complete sample of 11,928 Wikidata resources, the total runtime was of approximately two minutes.

Our third measurement was performed on the statements inferred in the final stage. We measured the amount of statements by predicates and their namespaces. Since our evaluation included the inference of *rdf:type* statements based on the transitivity property of *rdf:subClassOf*, we have measured the number of *rdf:type* statements inferred as well, grouping by the namespaces of the objects of statements.

From the sample of 11,928 Wikidata resources, the reasoning inferred 1,785,227 statements, averaging approximately 150 statements per resource. These statements contained predicates from 43 different namespaces, and the most frequently found are listed in Table 5. Most of the top ones where expected since they were related to those found in the reasoning rules applied. It was surprising, however, the large amount of statements

Table 4. The Results of execution time.

Operation	Execution on ontologies original size	Execution on the statement subset required by the rules	Execution on the statement subset for reasoning on a specific RDF resource
Ontology subset selection	N/A	N/A	6.8 ± 5.0 ms
Reasoning time	Reasoner failure (high memory requirements)	Reasoning not successful (too long - several hours)	3.5 ± 5.1 ms
Total time - ontology subset selection plus reasoning time	Unknown	Unknown	10.3 ± 7.6 ms

inferred from other namespaces, which amounted to 36% of the inferred statements. These results make it evident that Wikidata's alignments to properties and classes of other ontologies are frequently available, and that they can support the automatic semantics processing by general purpose RDF processing tools.

Table 5. The average number of inferred statements from a Wikidata RDF resource.

Namespace of predicates	Statements
http://d-nb.info/standards/elementset/gnd#	473,076
http://www.w3.org/1999/02/22-rdf-syntax -ns#	427,361
http://schema.org/	114,782
http://www.w3.org/2000/01/rdf-schema#	33,037
http://xmlns.com/foaf/0.1/	32,188
http://purl.org/dc/terms/	23,343
http://id.loc.gov/ontologies/bibframe/	20,019
http://www.cidoc-crm.org/cidoc-crm/	16,894
http://www.w3.org/2006/vcard/ns#	15,184
http://purl.obolibrary.org/obo/	14,224

Regarding the inference of *rdf:type* statements, we observed that 12 namespaces had at least 1 inference of *rdf:type* for more than 99% of the Wikidata RDF resources. The details for these namespaces are shown in Table 6. Schema.org had the highest average of inferences per resource, which is not surprising since the complete class structure of Schema.org was included in the source data for reasoning. These results support the conclusion that Wikidata's alignments to classes of other ontologies are frequently available.

Table 6. The namespaces with *rdf:type* inferences for at least 99% of the Wikidata RDF resources, and their respective averages per resource.

Namespace of rdf:type object	RDF resources with 1+ inferences	Average inferences per RDF resource
http://schema.org/	11,928	4.0 ± 0.5
http://www.cidoc-crm.org/Entity/e1-crm-entity/	11,928	1.0 ± 0.0
http://id.loc.gov/ontologies/bibframe/	11,927	3.1 ± 0.4
http://www.cidoc-crm.org/entity/e25-man-made-feature/	11,927	1.0 ± 0.0
http://purl.org/dc/dcmitype/	11,926	1.0 ± 0.1
https://d-nb.info/standards/elementset/gnd#	11,925	1.0 ± 0.0
http://dbpedia.org/ontology/	11,925	1.0 ± 0.3
http://www.cidoc-crm.org/Entity/e2-temporal-entity/	11,918	1.0 ± 0.0
http://def.seegrid.csiro.au/isotc211/iso19108/2002/temporal#	11,918	1.0 ± 0.0
http://www.cidoc-crm.org/entity/e18-physical-thing/	11,908	1.0 ± 0.0
http://purl.org/dc/terms/	11,908	1.0 ± 0.1
http://www.cidoc-crm.org/entity/e24-physical-man-made-thing/	11,894	1.0 ± 0.1

We have further analysed the inferred statements containing predicates or objects with Schema.org URIs. Table 7 focuses on the inference results of Schema.org. Regarding the inference of *rdf:type* properties with Schema.org classes, at least one inference was always made from each Wikidata resource, and on average, 4.0 ± 0.5 *rdf:type* statements were inferred per resource. Regarding the inference of statements having Schema.org predicates, an average of 9.6 ± 2.5 statements were inferred. Altogether, an average of 13.6 ± 2.5 Schema.org statements were inferred.

5 Conclusion and Future Work

We have tested several approaches for solving RDF reasoning problems in large ontologies with limited computational resources. We identified that with a high volume of input data for reasoning, the memory requirements of the RDF reasoner become very demanding leading to an extremely long runtime or even to the impossibility of a successful execution.

Table 7. The average amount of inferred statements from a Wikidata RDF resource and the breakdown for those that contain Schema.org predicates or in objects of *rdf:type* properties.

Statements in Wikidata source resources	Total inferred statements	Inferred statements with Schema.org		
		In predicates	In *rdf:type* objects	Total
271.8 ± 77.4	149.7 ± 33.5	9.6 ± 2.5	4.0 ± 0.5	13.6 ± 2.5

Our method defines two intermediate tasks that reduce the volume of data used during reasoning. The first task is executed in advance of the reasoning, and it creates a subset of the ontology that contains only the statements that match the triple patterns included in at least one of the reasoning rules. The second reduction is performed when a reasoning request is invoked for a fragment of the target dataset, and it selects the statements of the ontology that are needed for the reasoning. The RDF reasoner runs efficiently when using only the resulting ontology subset and the target data.

Besides the evaluation of our reasoning method, it was also possible to evaluate Wikidata's potential for automatic semantics processing by general purpose RDF tools. Our conclusions pertain mainly to the context of cultural heritage data. We found that Wikidata's classes and properties frequently contain alignments to other ontologies, which are nowadays in use by the cultural domain.

We tested the inferences from Wikidata's Schema.org alignments. The inference of *rdf:type* statements was very positive, with at least one statement inferred for each cultural resource. However, the inference of statements with Schema.org predicates were not as high as we initially expected. We believe the difference between the different results is explained by the extensive class hierarchy of Wikidata along with the reasoning rules defined for *rdf:type*. These rules infer statements using all the super classes of the original statement, which makes an equivalence to at least one Schema.org class to be available in most cases. The reasoning rule for *owl:equivalentProperty*, however, does not infer the *owl:equivalentProperty* from super properties, it is inferred only from the particular Wikidata's property in the predicate leading to fewer alignments being available.

Our experiment on cultural data from Wikidata provided evidence supporting that the need for high resources may be mitigated since its data model contains alignments statements to several other ontologies in use by the cultural domain. Along with the application of RDF and OWL reasoning these alignments can be used to infer views of Wikidata expressed in cultural domain's data models. In addition, our method allows to lower the computational resources required for reasoning on Wikidata.

The positive results obtained with Wikidata motivate further work for maturing our method into a generic software framework to solve reasoning problems in large volumes of RDF data. The prototype should be redesigned into a framework supporting machine-readable definitions of large-scale reasoning problems. This definition of a reasoning problem must allow the configuration of all the subtasks of our method: the data source(s) for the ontology(ies); the triple patterns, or fragments, from the ontology for the reasoning problem; the reasoning rules; and triple fragments for the target dataset. We will start by an investigation of available standard vocabularies that address some of these configuration requirements. We expect that DCAT [19] and VoID [20] might support the configuration of data sources, and SHACL [21] or ShEx [22] might support the configuration of triple fragments.

Regarding Wikidata, our experiment supported that its Schema.org view may fulfil the requirements of some applications in culture. The use of alignment statements in Wikidata for ontologies of other domains should also be investigated.

Acknowledgments. We would like to acknowledge Antoine Isaac from the Europeana Foundation for his contribution to the preliminary discussion of our work regarding RDF reasoning and Wikidata. We also acknowledge the contribution of João Cardoso from INESC-ID for his review of the article.

This work was partly supported by Portuguese national funds through Fundação para a Ciência e a Tecnologia (FCT) with reference UIDB/50021/2020 and by the European Commission under contract number 30-CE-0885387/00-80.

References

1. Hayes, P., Patel-Schneider, P.F (eds.): RDF 1.1 Semantics. W3C Recommendation, W3C (2014)
2. Schneider, M. (eds.): OWL 2 Web Ontology Language RDF-Based Semantics, 2nd edn. W3C Recommendation, W3C (2012)
3. Motik, B., Patel-Schneider, P.F., Grau, B.C. (eds.).: OWL 2 Web Ontology Language Direct Semantics, 2nd edn. W3C Recommendation, W3C (2012)
4. Rietveld, L.: Publishing and Consuming linked data: optimizing for the unknown. In: Studies on the Semantic Web, vol. 21. IOS Press (2016)
5. Radulovic, F., Mihindukulasooriya, N., García-Castro, R., Gomez-Pérez, A.: A comprehensive quality model for Linked Data. In: Semantic Web, vol. 9, no. 1. IOS Press (2018)
6. Beek, W., Rietveld, L., Ilievski, F., Schlobach, S.: LOD lab: scalable linked data processing. In: Pan, J.Z., et al. (eds.) Reasoning Web 2016. LNCS, vol. 9885, pp. 124–155. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-49493-7_4
7. Beek, W., Rietveld, L., Schlobach, S., van Harmelen, F.: LOD laundromat: why the semantic web needs centralization (even if we don't like it). In: IEEE Internet Computing, vol. 20, no. 2. IEEE (2016)
8. Fernández, J.D., Beek, W., Martínez-Prieto, M.A., Arias, M.: LOD-a-lot. In: d'Amato, C., et al. (eds.) ISWC 2017. LNCS, vol. 10588, pp. 75–83. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68204-4_7
9. Hogan, A.; Reasoning techniques for the web of data. In: Studies on the Semantic Web, vol, 19. IOS Press (2014)
10. Oren, E., Kotoulas, S., Anadiotis, G., Siebes, R., ten Teije, A., van Harmelen, F.: Marvin: distributed reasoning over large-scale Semantic Web data. J. Web Semant. **4**(7), 305–316 (2009). https://doi.org/10.1016/j.websem.2009.09.002
11. Stuckenschmidt, H., Broekstra, J.: Time – space trade-offs in scaling up RDF schema reasoning. In: Dean, M., et al. (eds.) WISE 2005. LNCS, vol. 3807, pp. 172–181. Springer, Heidelberg (2005). https://doi.org/10.1007/11581116_18
12. Gu, R., Wang, S., Wang, F., Yuan, C., Huang, Y.: Cichlid: efficient large scale RDFS/OWL reasoning with spark. In: 2015 IEEE International Parallel and Distributed Processing Symposium, pp. 700–709 (2015). https://doi.org/10.1109/ipdps.2015.14

13. Ravindra, P., Deshpande, V.V., Anyanwu, K.: Towards scalable RDF graph analytics on MapReduce. In: Proceedings of the 2010 Workshop on Massive Data Analytics on the Cloud (MDAC 2010). ACM (2010). https://doi.org/10.1145/1779599.1779604
14. Komazec, S., Cerri, D.: Valle, E.D., Horrocks, I., Bozzon, A. (eds.) 1st International Workshop on Ordering and Reasoning (OrdRing 2011). CEUR-WS (2011)
15. Simou, N., Chortaras, A., Stamou, G., Kollias, S.: Enriching and publishing cultural heritage as linked open data. In: Ioannides, M., Magnenat-Thalmann, N., Papagiannakis, G. (eds.) Mixed Reality and Gamification for Cultural Heritage, pp. 201–223. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-49607-8_7
16. Hyvönen, E.: Publishing and using cultural heritage linked data on the semantic web. In: Synthesis Lectures on the Semantic Web: Theory and Technology. Morgan & Claypool (2012)
17. Jones, E., Seikel, M. (eds.): Linked Data for Cultural Heritage. Facet Publishing (2016)
18. Freire, N., Isaac, A.: Technical usability of Wikidata's linked data. In: Abramowicz, W., Corchuelo, R. (eds.) BIS 2019. LNBIP, vol. 373, pp. 556–567. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-36691-9_47
19. Maali, F., Reikson, J.: Data Catalog Vocabulary (DCAT). W3C Recommendation, W3C (2019)
20. Alexander, K., Cyganiak, R., Hausenblas, M., Zhao, J.: Describing Linked Datasets with the VoID Vocabulary. W3C Interest Group Note, W3C (2011)
21. Knublauch, H., Kontokostas, D. (eds.): Shapes Constraint Language (SHACL). W3C Recommendation, W3C (2017)
22. Prud'hommeaux, E., Boneva, I., Gayo, J.E.L., Kellog, G. (eds.): Shape Expressions Language 2.1. W3C Draft Community Group Report, W3C (2018)