BioHackrXiv

Knowledge graphs and wikidata subsetting

Jose Emilio Labra Gayo[1], Ammar Ammar[2], Dan Brickley[3], Daniel Fernández Álvarez[1], Alejandro González Hevia[1], Alasdair Gray[4], Eric Prud'hommeaux[5], Denise Slenter[2], Harold Solbrig[6], Seyed Amir Hosseini Beghaeiraveri[4], Benno Fünkfstük[8], Andra Waagmeester[7], Egon Willighagen[2], Liza[3], Guillermo[3], Roberto García[9], and Leyla Jael Castro[10]

1 WESO research group, University of Oviedo, Spain **2** Maastricht University **3** Google, London, UK **4** Heriot Watt University, UK **5** Janeiro Digital, W3C/MIT **6** Johns Hopkins University **7** Micelio/Gene Wiki **8** TU Dresden **9** Universitat de Lleida, Spain **10** ZB MED Information Centre for Life Sciences, Germany

[TOC]

BioHackathon series:
BioHackathon Europe 2020
Virtual conference 2020
Wikidata subsetting

Submitted: 22 Jan 2021

Published by BioHackrXiv.org

Abstract

Knowledge graphs have successfully been adopted by academia, governement and industry ref? to represent large scale knowledge bases. Open and collaborative knowledge graphs such as Wikidata capture knowledge from different domains and harmonize them under a common format, making it easier for researchers to access the data while also supporting Open Science... > [name=Labra] I just want to say that wikidata is big and important (and to try comments :)

Wikidata keeps getting bigger and better, which subsumes integration use cases. Having a large amount of data such as the one presented in a scopeless Wikidata offers some advantages, e.g., unique access point and common format, but also poses some challenges, e.g., performance. Regular wikidata users are not unfamiliar with running into frequent timeouts of submitted queries. Due to its popularity, limits have been imposed to allow for fair access to many. However this suppreses many interesting and complex queries that require more computational power and resources. Replicating Wikidata on one owns infrastructure is a solurtion > [name=Andra] reference to Addshore blog on copying wikidata (/micwGyuUQqmZYvAkJsQ5pw) However, with the sheer size of wikidata and its ongoing growth, this can be expensive to do. In most, if not all of those complex use-cases only subsets of Wikidata are actually needed. With smaller subsets it is possible to replicate those parts on less expensive infrastructure. During a series of Hackathon > [name=Andra] references to Biohackathon 2019, Swat4HCLS hackathon 2019, virtual COVID biohackathon, virtual barcelona hackathon, virtual SWAT4HCLS hackathon 2021 > creating those subsets have emerged as an alternative to reduce the amount and spectrum of data offered by Wikidata. Subsets are extracted from Wikidata, restricting their data to, for instance, a particular domain or use case. Less data makes more complex queries possible while still keeping the compatibility with the whole Wikidata as the model is kept. > [name=Labra] Short sentence about why we would like subsets... > In this paper we describe our Wikidata subsetting efforts during the Virtual BioHackathon 2020 and SWAT4LS , including some approaches to create subsets and some susbsets from the Life Sciences domain. > [name=Labra] Short sentence about what we did... >

Introduction

[name="Labra"] Short motivation (use cases will be in a separate section later)

There are several benefits and use cases derived from extracting knowledge graphs subsets. For example, it is possible to extract datasets for a specific research task from a given knowledge graph with multiple data. It also allows users to store these subsets locally and reduce database scaling and costs. The extracted datasets can be a useful resource for researchers which can analyze the evolution of the data and develop on-the-fly transformations and subsets for their specific needs.

Background

During the last years there were multiple biohackathon efforts to develop mechanisms which extract subsets from linked data. One example would be the G2G language created in a 2019 Biohackathon-Europe project, which used a mixture of SPARQL expressions and Cypher patterns to extract property graphs. > [name="Labra"] The reference to G2G is a bit ugly... it points to just an example of the G2G... I though about including a reference to the github repo but it doesn't mention the G2G language... do we have a better reference to cite?

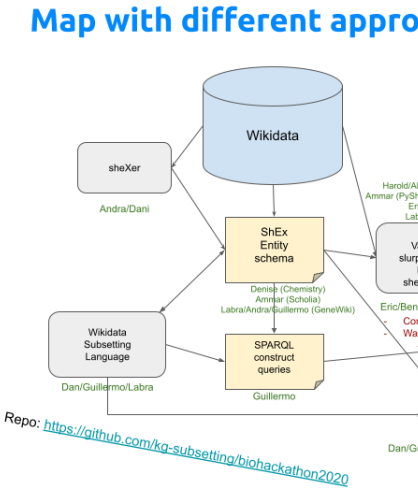
An initial effort started during SWAT4(HC)LS 2019 to define the topical use cases and main methods which was collected as a some informal notes that were later added to a Wikidata project. Later, at the virtual Covid-19 biohackathon in April, an initial prototype was started to use ShEx schemas to define the subsets to be extracted from a wikibase instance.

This report collects the main advances developed during the virtual Biohackathon 2020, project 35 which were complemented with the SWAT4HCLS virtual hackathon in january 2021.

Description of activities

In this section we report the different activities that were done during the Biohackathon and the SWAT4LS events.

General overview



[name="Labra"] TODO: Add a better version of the diagram

As a running example we departed from the GeneWiki project following (Waagmeester et al., 2020). That paper a figure with a UML-like data model that represents the main concepts related with life sciences in the GeneWiki project. That figure was taken as the initial point and the goal was to obtain a wikidata subset that followed the data model represented in that figure.

> [name=Labra] I am not sure if we can add a copy of that figure here or just refer to it... should we ask the publishers?

We classify the activities in 4 parts: - Describing the subsets: how do we describe what subset of wikidata we are interested in? - Extraction techniques: which approach can we follow to extract the subset from wikidata? - Publishing and using the subset: once we have the subset, how do we publish it so it can be used? - Use cases: what use cases did we identify?

Describing the subsets

A first step to obtain a Knowledge Graph subset is to describe what we expect to extract. Given that we are talking about really big data, this process must be done in a machine-processable way. Also, given that those subsets are intended to be used by people, they should be easily described by some domain experts.

A common use case is to define the boundaries of the subset by some topic identifying the type of entities or the properties that are of interest for a given use case. Next, it is necessary to describe the bounderies of that subset in its larger context of all of Wikidata.

We identified several approaches to describe the subsets: - SPARQL Construct queries - Filtering by rule patterns - Shape Expressions and entity schemas - Defining a domain specific language

SPARQL construct queries

> [name=Labra] We could use a running example which would help readers to follow the paper better. I chose "anatomical_structures" because it was the first one I had at hand... maybe a better one is better?

One approach to extract data from any SPARQL endpoint is to use SPARQL construct queties. As an example, the following query:

```
CONSTRUCT {
?anatomical_structure wdt:P31 wd:Q4936952.
?anatomical_structure wdt:P361 ?part_of.
?anatomical_structure wdt:P527 ?has_part.
} WHERE {
 ?anatomical_structure wdt:P31/wdt:P279* wd:Q4936952
} UNION {
 ?anatomical_structure wdt:P31/wdt:P279* wd:Q4936952.
 ?anatomical_structure wdt:P361 ?part_of.
} UNION {
 ?anatomical_structure wdt:P31/wdt:P279* wd:Q4936952.
 ?anatomical_structure wdt:P527 ?has_part.
}
```

Can be used to retrieve anatomical structures from Wikidata.

SPARQL construct queries can also transform the data retrieved on-the-fly. For example, the previous query could be expressed as (link to the original query:

```
CONSTRUCT {
?anatomical_structure a schema:AnatomicalStructure.
?anatomical_structure schema:partOfSystem ?part_of.
?anatomical_structure schema:subStructure ?has_part.
} WHERE {
 ?anatomical_structure wdt:P31/wdt:P279* wd:Q4936952
} UNION {
 ?anatomical_structure wdt:P31/wdt:P279* wd:Q4936952.
 ?anatomical_structure wdt:P361 ?part_of.
} UNION {
 ?anatomical_structure wdt:P31/wdt:P279* wd:Q4936952.
 ?anatomical_structure wdt:P527 ?has_part.
}
```

which not only retrieves the data but also transforms it using properties from the schema.org vocabulary.

Filtering by rule patterns

WDumper is a tool created by Benno Fünkfstük that generates Wikidata RDF dumps on demand. The tool is based on wikidata Toolkit and allows the user to select the desired entities and properties according to rule patterns, as well as other settings like labels, descriptions, aliases, sitelinks, etc. Upon request the service creates the RDF dumps which can later be downloaded.

Internally, the rules are represented by a JSON configuration file.

> [name="labra"] We could go into more details and describe the JSON configuration file... but not sure if it is necessary...

ShEx and entity schemas

ShEx was created in 2014 as a human-readable and concise language for RDF validation and description (Prud'hommeaux, Labra Gayo, & Solbrig, 2014). In 2019, ShEx was adopted by Wikidata to define entity schemas (Thornton et al., 2019). > [name=Labra] TODO... > - ShEx created manually > - Automatically generating entity schemas: sheXer

Wikidata subsetting language

ShEx can be too expressive and it may contain constructs that are

- TODO: Describe the idea... the language can generates SPARQL Construct queries

Extraction of subsets from wikidata

Slurping

> [name=Labra] TODO: describe the slurping process shex.js and pyshex already support slurp (shex-s should do it soon...)

The following example uses a Wikidata ShEx definition to construct a minimal conforming graph from Wikidata using PyShEx slurper. It has been deployed as a Jupyter notebook.

Talk about issue with blank nodes and endpoints... Phabricator ticket: https://phabricator.wikimedia.org/T267782 2 appearances of blank nodes: - to represent ontological constructs like `owl:complentOf` which may be justified. - to express unknown and no values. This use could be replaced TODO: Extend this explanation?

WDumper

[name=Labra] We could describe the process followed by WDumper if we have more information/details about it @(**???**) could describe here how he combined the SPARQL construct queries with WDumper

Linking to other graphs

[name=Labra] Talk here about the process to link the extracted contents to other contents... for example, in our running example, they linked it to scrapped data from mobidb and disprot @(**???**) could add here some description of the scrapping github repo: https://github.com/elizusha/scraper

Creating a Wikibase instance from RDF dumps

[name=Labra] Talk about - Wikibase - WikidataIntegrator?

Publishing and using the subsets

Documenting the subset

[name=Labra] - Extracting the ShEx schema using sheXer

RDF HDT

[name=Labra] Talk about RDF HDT dumps... we had a long discussion at SWAT4LS in which Lydia, Javier Fernández and Wouter Beek participated...

- Dump using RDF HDT
- Publish RDF HDT file using Fuseki through a SPARQL endpoint
  - Available as Docker container: https://github.com/rogargon/fuseki-hdt-docker

Interactive visualizations/explorations

- Interactively explore the data available through the SPARQL endpoint using Rhizomer
  - Available as Docker containers: https://github.com/rhizomik/rhizomerEye

SPARQL queries and demos

[name=Labra] We talked about providing nice SPARQL queries and/or demos about the subset we obtained... In this section it would be nice to at least describe one very nice SPARQL query showing what we accomplished... ideas? - Description of https://github.com/athalhammer/danker-hdt-docker by athalhammer ?

Use cases

GeneWiki

[name=Labra] This use case is probably part of the running example along the whole paper... so we may remove it from here and title the section "Other use cases"

Scholia

[name=Labra] I think this use case comes mainly from Ammar

Chemistry

[name=Labra] This use case comes from Denise

Fact-checking

[name=Labra] We talked a bit about this use case and it think it makes sense... maybe add some prose or a real use case?

Discussion

[name=Labra] TODO... if we don't have nothing to say, we could merge this section with the conclusions section... - A link with the resulting docker image? - Future work: Wikidata subsetting as a service?

Conclusions

[name=Labra]

[TOC]

References

Prud'hommeaux, E., Labra Gayo, J. E., & Solbrig, H. (2014). Shape expressions: An RDF validation and transformation language. In *Proceedings of the 10th international conference on semantic systems, SEMANTICS 2014* (pp. 32–40). ACM.

Thornton, K., Solbrig, H., Stupp, G. S., Labra Gayo, J. E., Mietchen, D., Prud'hommeaux, E., & Waagmeester, A. (2019). Using shape expressions (ShEx) to share RDF data models and to guide curation with rigorous validation. In *The semantic web* (pp. C1–C1). Springer International Publishing. doi:10.1007/978-3-030-21348-0_40

Waagmeester, A., Stupp, G., Burgstaller-Muehlbacher, S., Good, B. M., Griffith, M., Griffith, O. L., Hanspers, K., et al. (2020). Wikidata as a knowledge graph for the life sciences. *eLife*, *9*. doi:10.7554/elife.52614