Exercise Sheet 2: RDF Modelling

Maximilian Marx, Markus Krötzsch

Knowledge Graphs, 2023-10-24, Winter Term 2023/2024

Exercise 2.1. Which of the following literals describe the same value? Explain your answer.

1. `"2"^^xsd:integer` vs. `"2.0"^^xsd:decimal`
2. `"2"^^xsd:decimal` vs. `"2"^^xsd:float`
3. `"2018-11-06T15:40:00+01:00"^^xsd:dateTime` vs. `"2018-11-06T14:40:00Z"^^xsd:dateTime`
4. `"2018-11-06T15:40:00+01:00"^^xsd:dateTime` vs. `"2018-11-06T14:40:00"^^xsd:dateTime`

A detailed description of each of the various XML Schema datatypes is given in the online specification: see https://www.w3.org/TR/xmlschema11-2/.

Exercise 2.2. Recall that blank nodes act as placeholders for arbitrary resources in RDF: they assert that there is something without saying what it is. Such an assertion might logically follow from other, stronger assertions, so that some triples in a graph might be redundant. For example, the second triple in the following dataset can be omitted without loss of information:

```
eg:s   eg:p   eg:o .
_:1    eg:p   _:2  .
```

More generally, an *instance* of an RDF graph G is a graph $\sigma(G)$ obtained by applying a function σ that maps blank nodes to arbitrary RDF terms. A graph is *lean* if it does not have any instance $\sigma(G) \subset G$ that is strictly contained in G. In the example, $\sigma = \{\texttt{\_:1} \mapsto \texttt{<s>}, \texttt{\_:2} \mapsto \texttt{<o>}\}$ shows that this graph is not lean.

Determine if the following graphs are lean:

(a)
```
eg:s   eg:p   eg:o .
_:1    eg:p   _:1  .
```

(b)
```
eg:s   eg:p   _:2  .
_:1    eg:p   eg:o .
```

(c)
```
eg:s   eg:p   eg:o .
_:1    eg:p   [ eg:p   [] ] .
```

(d)
```
eg:s   eg:p   eg:s .
_:1    eg:p   [ eg:p   [] ] .
```

∗ **Exercise 2.3.** Show that it is NP-complete to decide if an RDF graph is not lean.

Hint:
For hardness, find a reduction from 3-colorability. Making an RDF graph non-lean if a graph is colorable is not hard. Making it lean if it is not colorable requires some trick to prevent the encoded graph from embedding into itself.

Exercise 2.4. Write a program that reads a graph in N-Triples format and checks whether the graph is bipartite. Use it to decide whether `authorship.nt.gz`[1] and `coauthors.nt.gz`[1] are bipartite.

Hint: each of the uncompressed graphs is roughly 4 GiB in size. In Python, you can use `gzip.GzipFile`[2] to process the compressed file without decompressing it first. There is also `authorship-snippet.nt.gz`[1], a small part of the graph that you can use during development.

Please note: In order to get the correct data files, please install git-lfs[3] on your system, and then activate it in your local repository (git lfs install).

Exercise 2.5. From the `coauthors.nt.gz` graph[1], extract the *connected component* containing <http://dblp.uni-trier.de/pers/s/Studer:Rudi>, i.e, extract the induced subgraph that

- contains <http://dblp.uni-trier.de/pers/s/Studer:Rudi>,
- contains all nodes reachable from <http://dblp.uni-trier.de/pers/s/Studer:Rudi> by some path, and
- contains all edges that are present in the full graph between these nodes.

Note that, while an RDF graph is inherently directed, edges in `coauthors.nt.gz` are symmetric, i.e., the graph is essentially undirected.
Hint: `authorship-snippet.nt.gz`[1] contains <http://dblp.uni-trier.de/pers/s/Studer:Rudi> and can be used for testing during development.

Exercise 2.6. The bibliographic database DBLP[4] offers individual data records as RDF in N-Triples format. This data can be downloaded from the URL obtained by appending `.nt` to the URI. Use this interface to find all publications that have https://dblp.org/pid/s/RudiStuder.html as their only author.

- Download some RDF files in your browser to find out how this information is encoded.
- Write a program that crawls a small part of the data to answer the query.

Note: If your program sends too many requests in a short time, the server will deny the request and cancel the connection. Dirty trick: use `time.sleep(1)` before executing a request.

Hint: `requests`[5] provides a high-level API for making HTTP requests in Python, but you may need to install it, e.g., using `pip`.[6] A built-in alternative that provides a lower-level interface is `urllib.requests`.[7]

[1] https://github.com/knowsys/Course-Knowledge-Graphs/tree/main/data/dblp
[2] https://docs.python.org/3/library/gzip.html
[3] https://git-lfs.github.com/
[4] https://dblp.org
[5] https://requests.readthedocs.io/en/latest/
[6] https://pypi.org/project/pip/
[7] https://docs.python.org/3/library/urllib.request.html