Leskinen, Petri; Hyvönen, Eero; Tuominen, Jouni

Sparql2graphserver: A server-side tool for extracting networks from linked data for data analysis

Published in:
CEUR Workshop Proceedings

Published: 01/01/2021

Document Version
Publisher's PDF, also known as Version of record

Sparql2GraphServer: a Server-side Tool for Extracting Networks from Linked Data for Data Analysis

Petri Leskinen[1][0000−0003−2327−6942] and Eero Hyvönen[1,2][0000−0003−1695−5840] and Jouni Tuominen[1,2][0000−0003−4789−5676]

[1] Semantic Computing Research Group (SeCo), Aalto University, Finland
[2] HELDIG – Helsinki Centre for Digital Humanities, University of Helsinki, Finland
http://seco.cs.aalto.fi, http://heldig.fi, firstname.lastname@aalto.fi

Abstract. This paper presents a server-side tool for constructing graph representations from a Linked Data for network analysis. The main features for the tool are: 1) Support to read data from a SPARQL endpoint, 2) easy adaptation to virtually any dataset and data model, and 3) easy usage in web portals for data analysis. The tool is in use.

Keywords: Linked Open Data · Network Visualization · Data Analysis
Demo paper

1 Introduction

Visualizing a network provides new insights about its structure and the underlying data [10,9]. A network can be built from Linked Data (LD) publication using SPARQL [3], where extraction patterns [1] allow a researcher to extract various types of connections (*links, edges*) between the *nodes*.

Many portals in the domain of Digital Humanities use network visualizations for data analysis, such as Six Degrees of Francis Bacon[3], Linked Jazz[4], and the co-citation graph and correspondent network in ePistolarium[5]. LodLive[6] allows the user to browse individual triples of a LD database. For front-end visualizations there are libraries such as Cytoscape.js[7], D3.js[8], Sigma[9], and 3D Force-Directed Graph[10] for rendering a network in 3D using WebGL. An overview of the evolution of LD visualization is presented in [8].

[3] http://sixdegreesoffrancisbacon.com
[4] https://linkedjazz.org/network/
[5] http://ckcc.huygens.knaw.nl/epistolarium
[6] http://en.lodlive.it/
[7] https://js.cytoscape.org/
[8] https://d3js.org/
[9] http://sigmajs.org/
[10] https://github.com/vasturiano/3d-force-graph

This paper introduces an online service, Sparql2GraphServer, for constructing networks based on LD knowledge graphs. The tool facilitates constructing either 1) an egocentric network [7] around *ego*, a specific knowledge graph URI, or 2) a general network, e.g., a group of nodes with specified properties. The service is used in practise in the biographical semantic portals AcademySampo[11] (in use) [5,6] and LetterSampo[12] (prototype).

2 Using the Tool

Backend Process The queries to the SPARQL endpoint are performed in the server side, and the process consists of three sequential steps. Firstly, a query for the network links is performed as a sequence of breadth-first searches until a desired amount of links is achieved. Secondly, the graph is constrained so that if the number of links exceeds the limit, the nodes having the lowest degrees are removed. This aims to retrieving more dense connections, e.g., a higher triadic closure between nodes of higher centrality measures, instead of start-shaped structures around a few central nodes. This pruning also aims to retrieving a more clustered graph structure to ease, e.g., the visual analysis. The constraining is optional and adjusting the query parameters allow for querying for an unfiltered graph, too. Finally, the node metadata is queried calculating simultaneously the predefined network statistics, e.g., the network diameter, average degree, and the total number of edges, nodes, and connected components.

The API query parameters are shown in Table 1; the queries are of the form `https://example.org/query?links=""&nodes=""&limit=""&id=""`. Example queries using Wikidata are shown in Tables 2 and 3. The link query requests for teacher-student relationships starting from the sources nodes. In the query, the placeholder `<ID>` is replaced with a list of source set nodes, initially containing only the *ego* given with the query parameter `id`. The link weight is defined as the count of common occupations, fields of work, and academic memberships between two nodes.

Server Response The JSON-formatted server response is depicted in Table 4. The result field `elements` contains two arrays for `edges` and `nodes`. The API adds the data fields defined in the result clauses in the corresponding SPARQL queries to the result. The node data contains metrics like `degree`, `in_degree`, `pagerank`, and `distance`, e.g., the shortest path length from *ego*. Finally, the element `metrics` contains, e.g., network diameter, average degree, and the numbers of nodes, edges, and connected components for the entire network. The response format described is compatible with Cytoscape.js, a library for network visualization in a web portal. Furthermore, the application also supports the GraphML[13] format.

Visualization in a Front-End Portal The API response can directly be an input for a Cytoscape.js component in a portal. Fig. 1 depicts a network

[11] https://seco.cs.aalto.fi/projects/yo-matrikkelit/en/
[12] https://seco.cs.aalto.fi/projects/rrl/
[13] http://graphml.graphdrawing.org/

Table 1. Query parameters of API

PARAMETER	DESCRIPTION	TYPE
endpoint	Server endpoint	required
prefixes	SPARQL prefixes	optional, default ""
links	SPARQL query for edge information	required
nodes	SPARQL query for node details	required
id	*ego* url for an egocentric network	optional, default None
limit	Limit the number of nodes	optional, default 1000
format	Response format: 'cytoscape' or 'graphml'	default 'cytoscape'
optimize	Drops out a proportion of low-degree nodes	optional, default 1.0
removeMultipleLinks	show only one link between nodes	optional, default True
customHttpHeaders	Headers, e.g. 'Authorization', of the query	optional, default None

Table 2. Query for links

```
PREFIX rdfs:
  <http://www.w3.org/2000/01/rdf-schema#>
PREFIX wdt:
  <http://www.wikidata.org/prop/direct/>

SELECT DISTINCT ?source ?target ?label
    (COUNT(DISTINCT ?link)+1 AS ?weight)
WHERE {
    VALUES ?source { <ID> }

    # teacher-student relations
    VALUES (?rel ?label) {
        (wdt:P802 "Student")
        (wdt:P185 "Doctoral student")
    }
    ?source ?rel ?target

    # links by common occupations, fields
        of work, or memberships
    OPTIONAL {
        VALUES ?prop
          { wdt:P101 wdt:P463 wdt:P106 }
        ?source ?prop ?link . ?target ?
       prop ?link
    }
} GROUP BY ?source ?target ?label
```

Table 3. Query for nodes

```
PREFIX rdfs:
  <http://www.w3.org/2000/01/rdf-schema#>
PREFIX wdt:
  <http://www.wikidata.org/prop/direct/>

SELECT DISTINCT ?id ?name
WHERE {

    VALUES ?id { <ID_SET> }
    ?id rdfs:label ?name .
    FILTER (LANG(?name) = "en").

} GROUP BY ?id ?name
```

Table 4. Server Response

```
{'elements': {
  'edges': [
    {'data':
      {'source':
        'www.wikidata.org/entity/Q9047',
      'target':
        'www.wikidata.org/entity/Q76510',
      'label':  'doctoral student',
      'weight':   7}
    }, {'data': ... }, ... ],
  'nodes': [
    {'data': {'id':
      'www.wikidata.org/entity/Q9047',
      'name':'Gottfried Wilhelm Leibniz',
      'distance': 0, 'degree':  5, '
       degree_weighted':  22,
      'in_degree': 0, 'out_degree':5,
      'in_degree_weighted': 0,
      'out_degree_weighted':22,
      'pagerank':  0.01214}
    }, {'data': ... }, ... ] },
'metrics':
  {'average_degree': 2.48,
  'diameter': 10,
  'number_connected_components': 1,
  'number_of_edges':  62,
  'number_of_nodes':  50},
'directed':   True
}
```

visualization based on the example result in Table 4. The node and edge labels are extracted from the query results while the visual appearance is configured in the front-side application code. Here the node color on a red-blue palette is based on the path distance from the center node, the node sizes are based on the out-degree, and the edge strengths on the count of common occupations, fields of work, and academic memberships between two nodes as defined in the query for edges (Table 2).

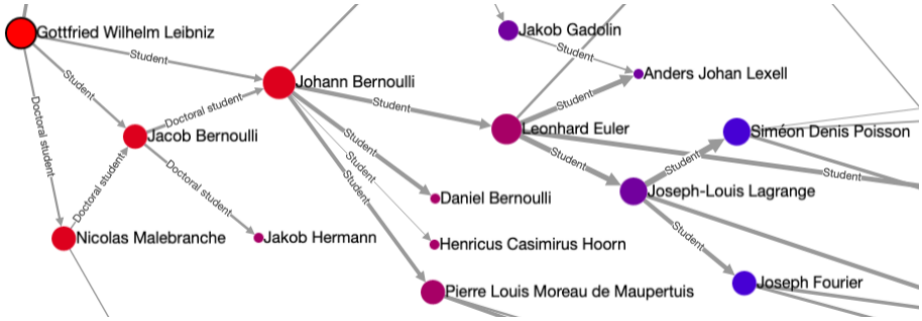

Fig. 1. Social network of mathematician Gottfried Wilhelm Leibniz in Wikidata

The tool Sparql2GraphServer is based on Flask[14], microframework application written in Python 3.8 using modules NetworkX 2.4[15] [2], NumPy[16], RDFLib[17], and SparqlWrapper[18]. The source code is available in GitHub[19]. The source code repository contains a Dockerfile recipe for running and deploying the application as a container which allows for portability and scaling.

3 Discussion

Integration to Sampo-UI Framework The Sparql2Network API is integrated to the Sampo-UI Framework [4]; the API was developed and is used in the AcademySampo and LetterSampo portals with Sampo-UI. The Sampo-UI framework includes functions for scaling and constraining the numeric result values into, e.g., edge widths, node sizes, as well as into RGB or HSL color ranges. For example, this page[20] in AcademySampo shows a connection network around the Finnish composer Jean Sibelius, the page[21] demonstrates using Sparql2Network to create a sociocentric network, possibly filtered by using faceted search.

Evaluation Performing several SPARQL queries to the database might have longer request times, especially in cases of smaller extracted networks. An alternative would be using a single query empowered with, e.g., property paths or nested selection blocks. However, that approach is not guaranteed to perform in feasible time in a more complex case and would require customizing the queries for each specific database. The chosen solution to have a pair of simple and quick queries appears to be more efficient. A custom SPARQL query can adapt to cases

[14] https://flask.palletsprojects.com/en/2.0.x/
[15] https://networkx.org/
[16] https://numpy.org/
[17] https://rdflib.readthedocs.io
[18] https://rdflib.dev/sparqlwrapper/
[19] https://github.com/SemanticComputing/Sparql2GraphServer
[20] https://akatemiasampo.fi/en/people/page/p21762/connections
[21] https://akatemiasampo.fi/en/people/faceted-search/network

where the linking property is not explicitly provided but reasoned from the data, e.g., a connection between two people who have lived in the same location at the same time.

Acknowledgements This work was funded by the Academy of Finland project SEMPARL, the EU project InTaVia[22], and is related to the EU COST action Nexus Linguarum[23]. CSC – IT Center for Science provided computational resources for the work. Discussions with Mikko Kivelä, Javier Ureña-Carrion, Minna Tamper, and Esko Ikkala are acknowledged.

References

1. Ghawi, R., Pfeffer, J.: Extraction Patterns to Derive Social Networks from Linked Open Data Using SPARQL. Information 11(7), 361 (2020).
2. Hagberg, A.A., Schult, D.A., Swart, P.J.: Exploring network structure, dynamics, and function using networkx. In: Proceedings of the 7th Python in Science Conference. pp. 11–15. Pasadena, CA, USA (2008).
3. Harris, S., Seaborne, A.: SPARQL 1.1 Query Language. https://www.w3.org/TR/sparql11-query/, accessed: 2021-06-28.
4. Ikkala, E., Hyvönen, E., Rantala, H., Koho, M.: Sampo-UI: A full stack JavaScript framework for developing semantic portal user interfaces. Semantic Web (2021), http://www.semantic-web-journal.net/, accepted.
5. Leskinen, P., Hyvönen, E.: Linked open data service about historical Finnish academic people in 1640–1899. In: DHN 2020 Digital Humanities in the Nordic Countries. Proceedings of the Digital Humanities in the Nordic Countries 5th Conference. pp. 284–292. CEUR Workshop Proceedings, Vol. 2612 (2020).
6. Leskinen, P., Hyvönen, E.: Reconciling and using historical person registers as linked open data in the AcademySampo knowledge graph. In: Proceedings of the 20th International Semantic Web Conference (ISWC 2021). Springer–Verlag (2021), https://seco.cs.aalto.fi/publications/2021/leskinen-hyvonen-reconciling-2021.pdf.
7. Marsden, P.V.: Egocentric and sociocentric measures of network centrality. Social networks 24(4), 407–422 (2002).
8. Po, L., Bikakis, N., Desimoni, F., Papastefanatos, G.: Linked Data Visualization: Techniques, Tools, and Big Data. Morgan & Claypool, Palo Alto, CA (2020). https://doi.org/10.2200/S00967ED1V01Y201911WBE019.
9. Schlögl, M., Lejtovicz, K.: A Prosopographical Information System (APIS). In: BD. pp. 53–58 (2017).
10. Warren, C.N., Shore, D., Otis, J., Wang, L., Finegold, M., Shalizi, C.: Six Degrees of Francis Bacon: A Statistical Method for Reconstructing Large Historical Social Networks. DHQ: Digital Humanities Quarterly 10(3) (2016).

[22] https://intavia.eu/
[23] https://nexuslinguarum.eu/the-action