InK Browser – The Interactive Knowledge Browser

Joseph Zalewski, Lu Zhou, Cogan Shimizu ✉, and Pascal Hitzler

Data Semantics Lab, Kansas State University, USA
{jzalewski, luzhou, coganmshimizu, hitzler}@ksu.edu

Abstract. We present an improved implementation of the Interactive Knowledge Browser (InK Browser), a tool for exploring knowledge graphs visually, using a schema diagram.

1 Motivation

There are many ways to explore data in a knowledge graph (KG). One can run SPARQL queries, for instance. Thus, a set of specialized tools have been developed to streamline the process, such as knowledge graph *browsers* (e.g., Pubby[1]). These provide a user experience similar to browsing the Web itself: entity URIs become hyperlinks, and triples involving that entity are presented, with further clickable entities, allowing the user to traverse the graph, while seeing only a small, focused piece of it at a time. In some cases, we have found that such browsers may be *too* locally focused: they do not allow the user to orient themselves within the graph as a whole. We propose to use *schema diagrams* [5], graphical representations of an *ontology*, to provide this context when navigating the graph. These diagrams can be ingested quickly, typically having few enough elements that they can be fully displayed on a single page, and show the large-scale structure of the data in the graph. Of course some ontologies are vast, and a specimen like the Gene ontology, with thousands of classes, would not be a good choice of ontology for this purpose. Yet many useful ontologies applicable to large KGs are small, and the InK Browser exploits these kind of ontologies and their schema diagrams. In theory there is no limit to the size of the *knowledge graph* that can be explored this way, since the tool is built on the existing SPARQL technology, which shows good scalability to large graphs.

2 Related Work and Contributions

Visual knowledge graph browsers already exist, some at an enterprise deployment level and in practical use [1][2]. However, they mostly focus on displaying

[1] See http://wifo5-03.informatik.uni-mannheim.de/pubby/.

arbitrary nodes of a knowledge graph and therefore need to be highly dynamic, since it is usually impossible to meaningfully show all nodes at once. InK Browser differs from these products in that it does not display most entities visually, but only ontological concepts, thus avoiding the explosive complexity of visuals that often results from visual browsing, since a small static set of concepts can be used that fit on the screen all at once. The whole browsing experience in InK Browser is run through this static diagram. This work is a reimplementation and extension of [3]. In particular, the current version is implemented in an extensible object-oriented style, and supports access to graphs hosted on remote endpoints, as well as control of the layout of diagrams.

3 Implementation Details

The project is built using Flask (which powers the browsing functionality) and mxGraph (which renders our schema diagram).[2] Actions within the browser are tied to generated SPARQL queries, which are executed using the SPARQLWrapper Python library. The browser consists of a collection of views. Views communicate with each other using a simple, flexible message protocol in order to propogate user actions across multiple views. This extensible plug-in philosophy makes it easy to customize functionality, and thus the user's perspectives on the graph. We list the currently available modules below. Implementation and installation details for this tool and tool suite can be found in our online portal.[3]

- **Instance List:** This view displays a subset of instance data for a selected class and is shown in Figure 1a (left column). The class shown is determined when the user clicks on a new entity, such as an entry in the Instance Data view or node in the Schema Diagram view.
- **Instance Data:** This view displays the "triple" information of a selected entity in a traditional KG browser style, shown in Figure 1a (middle column).
- **Schema Diagram:** This view displays the schema diagram for the KG, as shown in Figure 1a (right column). This prototype uses an external script to extract the relevant schema diagrammatic information from a prepared OWL file, such as the annotations generated while using CoModIDE [6]. This schema diagram in this view is interactive and will allow a class-level navigation by clicking on the nodes of the graph.
- **Class Hierarchy:** This view displays the class hierarchy of the KG in a collapsible tree format, as mapped to a common schema, shown in Figure 1b in order to provide users with an overview of general topics in the KG. Child-parent relationships indicate subsumption. Currently, we display this mapping according to the Wikipedia Category Graph [4]. Mappings are generated via a pre-trained word2vec model to detect the most similar term in

[2] See `jgraph.github.io/mxgraph/` and `https://jgraph.github.io/mxgraph/`
[3] See `http://daselab.org/content/modular-ontology-engineering-portal`.

the Wikipedia category graph. We plan to support Schema.org[4] or SUMO ontology[5] as well.
- **Entity Annotation:** This view displays annotation assertions for a selected entity and is shown in Figure 1b. Currently we automatically display rdfs:label, rdfs:comment, and dcterms:provenance, but hope to make this customizable in the future.
- **Text Search:** The text search view provides users with a search box in the current workspace. The search view takes the user's input and returns a list of similar entities with their IRIs and useful annotations sorted by relevance. This can help users target their understanding of the KGs. An example is shown in the Figure 1d.
- **Graph Statistics:** The graph statistics view displays simple graph statistic (e.g., numbers of nodes, edges, and triples) and is shown in Figure 1e.

4 Demonstration

We will demonstrate the InK Browser live. The demo will include accessing a real KG developed as part of the KnowWhereGraph[6] project, and navigating it using the GUI functionality of InK Browser. The schema diagram used to organize the data will be one created from CoModIDE [6].

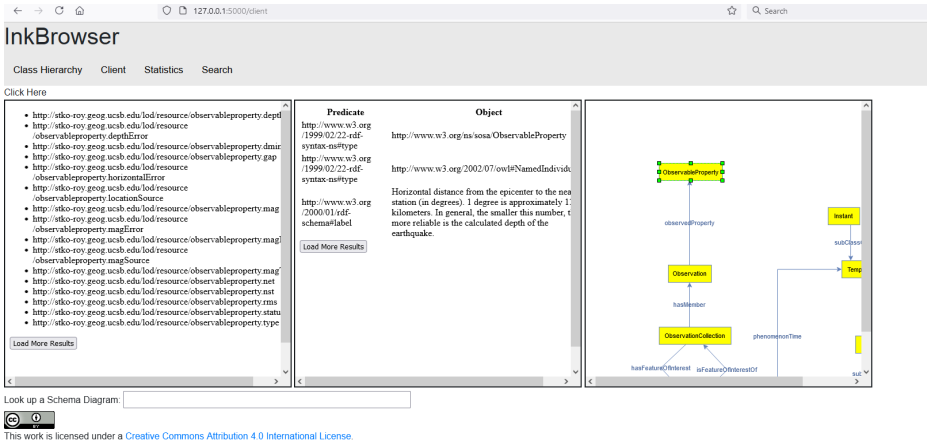
5 Conclusion and Future Work

InK Browser shows promise as a new paradigm in knowledge graph browsing, allowing the user to see individual triples and the structure of an entire graph at once. However there is much to do moving forward. We anticipate to improve the UI of the program (e.g., improving its visual look and feel), the UX (e.g., identifying an rdfs:label or concept name to use instead of an IRI), and the overall integration of Flask's features, develop additional modules (e.g., a map-based geographic data interface that detects and displays available spatially-explicit data), and finally, conduct a user study to more objectively determine whether the InK Browser browsing experience is any better than that of existing systems.
Acknowledgement. The authors acknowledge support by the National Science Foundation under Grant 2033521 A1: KnowWhereGraph: Enriching and Linking Cross-Domain Knowledge Graphs using Spatially-Explicit AI Technologies.

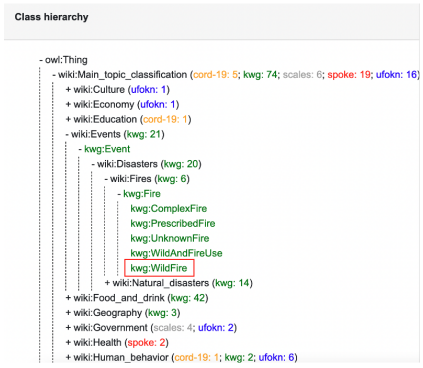
References

1. Gruff – new browser based version. `https://allegrograph.com/products/gruff/`, accessed: 2021-08-14

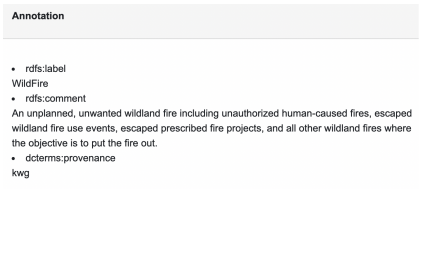
[4] See `https://schema.org/`.
[5] See `https://www.ontologyportal.org/`.
[6] See `https://knowwheregraph.org/`.



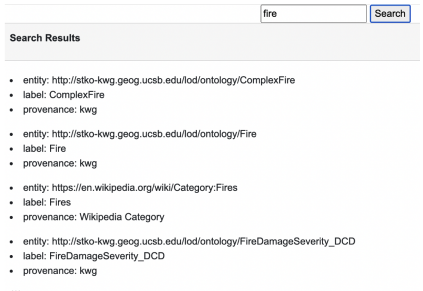
(a) A look at the InK Browser. The left column displays distinct instances of the class chosen in the schema diagram (right column) and specific instance data for a given choice of individual (center column).



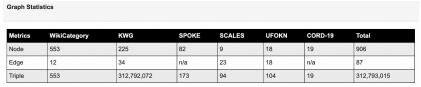
(b) Wildfire Class in Hierarchy



(c) Wildfire Annotations



(d) Text Search View



(e) Graph Statistics View

Fig. 1: Text Search and Graph Statistics View

2. Knowledge graph visual browser web application. https://github.com/linkedpipes/knowledge-graph-browser-frontend, accessed: 2021-08-14
3. Chittella, R.S.: Leveraging schema information for improved knowledge graph navigation (2019), http://rave.ohiolink.edu/etdc/view?acc_num=wright1564755327091243
4. Sarker, M.K., Schwartz, J., Hitzler, P., Zhou, L., Nadella, S., Minnery, B.S., Juvina, I., Raymer, M.L., Aue, W.R.: Wikipedia knowledge graph for explainable AI. In: Villazón-Terrazas, B., Ortiz-Rodríguez, F., Tiwari, S.M., Shandilya, S.K. (eds.) Knowledge Graphs and Semantic Web - Second Iberoamerican Conference and First Indo-American Conference, KGSWC 2020, Mérida, Mexico, November 26-27, 2020, Proceedings. Communications in Computer and Information Science, vol. 1232, pp. 72–87. Springer (2020). https://doi.org/10.1007/978-3-030-65384-2_6, https://doi.org/10.1007/978-3-030-65384-2_6
5. Shimizu, C., Eberhart, A., Karima, N., Hirt, Q., Krisnadi, A., Hitzler, P.: A method for automatically generating schema diagrams for modular ontologies. In: 1st Iberoamerican Conference on Knowledge Graphs and the Semantic Web (2019), to Appear.
6. Shimizu, C., Hammar, K., Hitzler, P.: Modular ontology modeling. Tech. rep. (2021), https://daselab.cs.ksu.edu/publications/modular-ontology-modeling, under Review.