Visualizing Populated Ontologies with OntoTrix

Benjamin Bach, Gennady Legostaev, Emmanuel Pietriga

Visualizing Populated Ontologies with OntoTrix

Benjamin Bach[1,3], Gennady Legostaev[2,1], and Emmanuel Pietriga[1]

[1] INRIA and LRI (Université Paris-Sud & CNRS), Orsay, France
`benjamin.bach@lri.fr, emmanuel.pietriga@inria.fr`
[2] St Petersburg State University, Saint Petersburg, Russia
`glegostaev@gmail.com`
[3] Dresden University of Technology, Germany

Abstract. Most tools for visualizing Semantic Web data structure the representation according to the concept definitions and interrelations that constitute the ontology's vocabulary. Instances are often treated as somewhat peripheral information, when considered at all. The visualization of instance-level data poses different but significant challenges as instances will often be orders of magnitude more numerous than the concept definitions that give them machine-processable meaning. We present a visualization technique designed to visualize large instance sets and the relations that connect them. This visualization uses both node-link and adjacency matrix representations of graphs to visualize different parts of the data depending on their semantic and local structural properties, exploiting ontological knowledge to drive the layout of, and navigation in, the visualization.

1 Introduction

By making use of the rich capabilities of graphical representations and by abstracting from the complex syntactic details of textual formats, visual tools aim at providing better cognitive support to users, from knowledge engineers to domain-expert end-users. They provide them with interactive representations of the data based upon state-of-the art information visualization techniques, better supporting tasks such as ontology understanding, discovery, search, comparison and mapping [3].

Most tools structure the visualization according to the concept definitions and interrelations that constitute the ontology's vocabulary. While many of them do support the visualization of instance data, instances are often treated as somewhat peripheral information. Employing Description Logics terminology, the visualization is mainly structured according to the TBox, instances that constitute the ABox being treated as leaf nodes in this tree or graph structure. Exceptions to this general observation exist, but either give a limited view of the ABox [1,6] or use conventional node-link diagram representations that hardly scale beyond a few hundred nodes at best [4].

Instances represent an essential part of the overall knowledge base. Compared to the definition of concepts based on OWL constructs, instance data are at a lower level of abstraction. But more importantly, instance datasets are often orders of magnitude larger (see, e.g., many of the datasets currently part of the *Linking Open Data* graph). As such, the visualization of instance-level data poses different but real challenges that remain to be addressed. We present OntoTrix, a visualization technique designed to enable users to visualize, and navigate in, large instance sets and their relations.
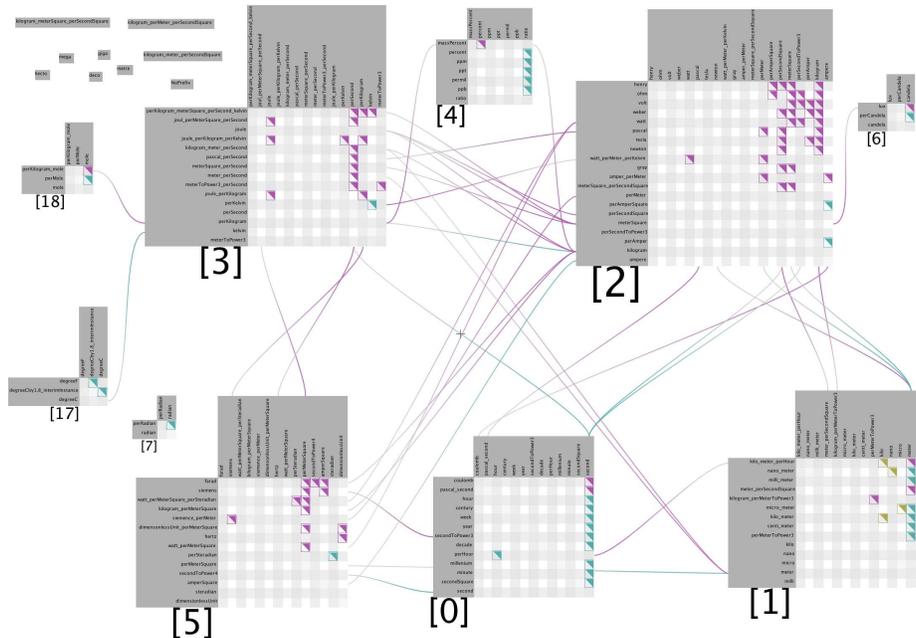
Fig. 1. Overview of SWEET's units.owl ontology with OntoTrix.

2   OntoTrix

Ontology graphs contain many nodes and edges, and are often non planar. Two main issues with node-link diagram representations of such graphs are their inefficient use of screen real-estate and edge crossings that make dense regions difficult to read, both eventually causing scalability problems. A well-known alternative to node-link diagrams for graph visualization are adjacency matrices. Nodes are represented as rows and columns, and edges as filled cells at the intersection of connected rows and columns. While node-link diagrams are good at showing the structure of relatively small and sparse graphs, adjacency matrices are very effective at showing large (better use of screen real-estate) and dense (no edge crossing) graphs. However, adjacency matrix representations are much less familiar to users than node-link diagrams, and make tasks that involve following paths in the graph more difficult [2].

OntoTrix is inspired by a recent hybrid network visualization technique, NodeTrix [2], that uses both node-link and adjacency matrix representations to visualize different parts of the data depending on their semantic and structural properties (Figure 1). The technique has proven successful at handling large networks, being very efficient at visualizing locally dense but globally sparse networks. It displays the overall structure of the network using a node-link diagram, and the dense subgraphs that represent communities using matrices. While the graph structure of ontologies might not always share the small-world characteristics of social networks, such a hybrid representation, combined with appropriate interaction techniques, can be an efficient means to perform exploratory visualization of large ontology instance sets.
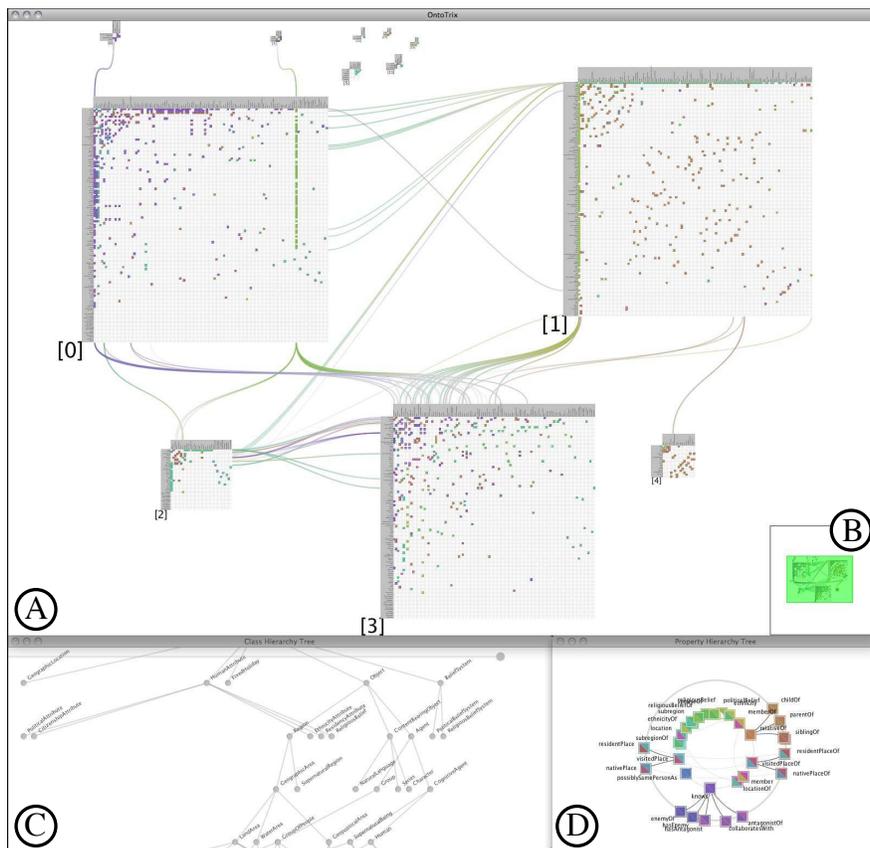
Fig. 2. OntoTrix Interface Overview: (A) Main NodeTrix view, (B) Bird's eye view, (C) Class hierarchy view, (D), Property hierarchy view.

NodeTrix was originally devised for undirected social network structures that only feature one type of node and one type of relation. We extend NodeTrix to handle the much richer and complex graph structure of populated ontologies, exploiting ontological (TBox) knowledge to drive the layout of, and navigation in, the representation. Embedded in a smooth zoomable interface [5], and coupled with visual representations of the class and property hierarchies that enable interactive navigation and filtering of instance data, this visualization produces more compact and legible representations than node-link diagram approaches, thus better scaling to large instance sets.

The OntoTrix environment features four main views (Figure 2). The main view (A), contains the OntoTrix representation of the instance set. (B) provides an interactive bird's eye view of (A). The class hierarchy is visualized in (C), the property hierarchy in (D). These views are highly synchronized. For instance, hovering a node in the class hierarchy view highlights all corresponding instances in the OntoTrix view. Hovering a node in the property hierarchy view highlights all corresponding elements in matrices, as well as corresponding edges between matrices. Classes declared as being in the domain or range of the property are highlighted in the class hierarchy, and conversely.

Matrices in NodeTrix basically correspond to highly-connected groups of actors, i.e., dense subgraphs that represent social communities. In OntoTrix, we propose different methods for grouping instances into matrices, yielding very different perspectives on the dataset. Instances can be clustered in matrices based on edge density, taking into account all types of relations between instance nodes (Figure 1). A second method groups instances into matrices according to class membership. A third method represents a tradeoff between the previous two. From an initial grouping based on density, all nodes are grouped together on a per-matrix basis according to class membership as described above (reordering columns and rows). Each of the original matrices can then be split into smaller matrices corresponding to class membership groups. The last method groups all instances involved in statements based on selected object property type(s).

Beyond improvements in terms of readability of a graph's structure, NodeTrix features an interesting property that is related to the above view of matrices as aggregates of instance nodes for the layout process. Often, these aggregates will by themselves represent interesting entities, not explicitly represented in the ontology, but that bear semantics. When grouping by class membership in OntoTrix, matrices obviously represent groups of similar instances. We thus label each matrix with the associated class name. When grouping by other methods, matrices cannot easily be tagged as there is no explicit information about the grouping that stems from the purely structural clustering of the graph. Matrices might still represent interesting entities, but will have to be labeled manually and are currently assigned a random identifier. For instance, the matrices in Figure 1 illustrate interesting grouping patterns: matrix [0] contains mostly time-related units, matrix [1] mostly energy-related units, matrix [6] units related to measuring light, etc. These groupings are very different from what is obtained when grouping by class membership, and give an interesting perspective on the dataset. By supporting dynamic, smoothly-animated transitions between grouping methods, OntoTrix allows for rapid switching between these perspectives.

OntoTrix is implemented in Java, using the ZVTM zoomable user interface toolkit, LinLogLayout for layout and clustering, Jena 2 for parsing ontologies, and the TDB backend for storing. Jena's OWL transitive reasoner provides a complete classification of the ontology. Additional reasoning can be performed using other reasoners.

References

1. Fluit, C., van Harmelen, F., Sabou, M.: Supporting user tasks through visualisation of lightweight ontologies. In: Handbook on Ontologies in Info. Systems. Springer-Verlag (2003)
2. Henry, N., Fekete, J.D., McGuffin, M.J.: Nodetrix: a hybrid visualization of social networks. IEEE Transactions on Visualization and Computer Graphics 13(6), 1302–1309 (2007)
3. Katifori, A., Halatsis, C., Lepouras, G., Vassilakis, C., Giannopoulou, E.: Ontology visualization methods—a survey. ACM Computing Surveys 39(4), 10:1–10:42 (2007)
4. Noppens, O., Liebig, T.: Interactive Visualization of Large OWL Instance Sets. In: International Workshop on the Semantic Web and User Interaction (SWUI) (2006)
5. Pietriga, E.: A toolkit for addressing hci issues in visual language environments. In: Symp. on Visual Languages and Human-Centric Computing (VL/HCC'05). pp. 145–152. IEEE (2005)
6. Tu, K., Xiong, M., Zhang, L., Zhu, H., Zhang, J., Yu, Y.: Towards imaging large-scale ontologies for quick understanding and analysis. In: ISWC. pp. 702–715. Springer-Verlag (2005)