Node-Link and Containment Methods in Ontology Visualization

Julia Dmitrieva and Fons J. Verbeek

Universiteit Leiden, Leiden Institute of Advanced Computer Science,
Niels Bohrweg 1, 2333 CA Leiden, The Netherlands
{jdmitrie,fverbeek}@liacs.nl
http://bio-imaging.liacs.nl/

Abstract. OWL Ontology language can be very expressive. This could provide difficulty in ontology understanding process. We belief, that an ontology visualization equipped with intuitive interactions can simplify this process, and help the user during ontology exploration and development.
We introduce an approach representing an ontology with two different tree visualization techniques: the node-link technique, and the containment technique. These two representations show the structure of an ontology differently. The former, represents an ontology as a graph structure. This graph structure, based on the ontology hierarchy and properties, can be explored in different geometries: Euclidean, hyperbolic and spherical.
The second representation shows only the hierarchical structure. The design of the containment approach is implemented in a non-standard way. In place of traditional two-dimensional space-filling techniques, we elaborate on the sphere-packing approach and make our hierarchy visualization three-dimensional. We augment this technique with the semantic zoom functionality, where the level of detail is a function of a distance from the viewer.

Key words: knowledge representation, ontology visualization, graph visualization, treemaps, node-link

1 Introduction

Currently, scientists have recognized the importance of modeling and representation their knowledge by means of ontologies. The OWL [1] ontology language, which is the W3C recommendation, provides the possibility for modelers to make their knowledge machine understandable, sharable and amenable for reasoning tools. Currently, there are different tools available for modeling and editing ontologies, e.g. Protégé [2] and OntoEdit [3].

Although a lot of work has been done in the field of the ontology modeling, editing, and reasoning, the area of ontology visualization is still insufficiently investigated. From a comprehensive survey [4] we can conclude that there is no visualization tools available which can visualize a given ontology in different geometrical models, instead the visualization is mostly limited to a single geometry and dimension, e.g. OntoRama [5] is a RDF browser visualizing an ontology as a graph laid out in a Poincaré-like Disk (two-dimensional, hyperbolic), TGVizTab [6] is an ontology visualization plugin for Protégé based on TouchGraph [7] technology (two-dimensional, Euclidean), OntoSphere [8] proposes a node-link visualization build around a selected concept in three-dimensional space (three-dimensional, Euclidean).

From the survey [4] it also follows that most visualization tools are devoted to representation of the hierarchical part of ontology. However, most domains are not so simple that they can be modeled as a taxonomy, because domain's concepts can be related to each other by means of other kind of relations, besides the sub/super class.

In this work we elaborate on our multi-view ontology visualization method [9], and combine two graph drawing techniques, e.g. node-link and containment. In the former, the ontology is represented as a graph structure, which is visualized in Euclidean, hyperbolic and spherical geometry. These geometrical models are visualized in two- and three-dimensional space. Beside the different geometry representation, we provide a possibility to visualize a knowledge base with all properties not restricted to taxonomic sub/super class relations. Each concept can be visualized with different level of depth and with selected subset of properties.

With the containment approach we visualize the hierarchical tree structure of an ontology. We drew the inspiration from CropCircles [10], but we base our visualization on sphere-packing technique instead of two-dimensional space-filling techniques as it was done in previous methods [10–12]. The visualization is further augmented with the semantic zoom, where the user by zooming in/our can control the level of detail (level of hierarchy) in the visualization.

2 Requirements

1. **Hierarchy.** Because the backbone of an ontology is the hierarchy, a visualization tool should represent the hierarchical structure.
2. **Properties representation.** Most ontologies are more than a hierarchy. Frequently the concepts are described by using restrictions on properties.
3. **Level of detail.** It should be possible to select the level of detail in a visualization. Ontologies from medical and biological domains can be very large and can require extensive computational and memory resources, this could slow down the application. Moreover, it can be difficult to understand the whole ontology shown in one visualization window. In our approach we provide the possibility to choose till which level an ontology will be visualized.
4. **History.** The user can return to the concepts that were chosen in the previous steps.
5. **Filtering.** Ontologies could contain hundreds of properties, e.g. NCI Thesaurus [13], interconnecting the concepts. The user can be interested in only

the subset of the ontology, based on the central concept and the properties of the user's choice.

6. **Multiple geometrical views.** The representation of the graph in different geometrical models can help the user to better understand the structure of an ontology. The evidence for this assertion is already shown [14].
7. **Zoom semantic/geometric.** An visualization tool equipped with semantic and geometric zoom interactions could help the user to see more or less details during ontology exploration. With the geometric zoom the visualized object is scaled when the user zooms in/out. The semantic zoom provide the possibility to see more/less details of the object by zooming in/out. With our containment approach the user can see the higher/lower levels of the hierarchy, when zooming in/out.

Further we will describe our visualization approach in context of these requirements.

3 Transformation to Visual Representation

In order to represent an ontology in a structure amenable for a visual representation we need to transform it to a graph or a tree data structure. In our approach, the graph structure is the basis for the node-link representation, and the hierarchical tree structure is the basis for the containment representation.

3.1 Graph Generation

The formal basis for OWL ontologies is provided by Description Logics [15] constructors. These constructors can be very complicated. In medical domains, e.g. NCI Thesaurus, ontology concepts are frequently defined as union of intersections of other concepts, e.g. $C \sqsubseteq A \sqcap ((\exists R_1.B_1 \sqcap B_2) \sqcup (C_1 \sqcap \exists R_2.C_2))$. Because of the diversity and complexity of DL constructors, we need to find a way to represent both the taxonomy as well as other kinds of *connections* between the concepts. In our node-link approach we represent ontology as a graph structure. We use OWL-API [16] and reasoner services [17] to generate a taxonomy and infer other kind of relationships between concepts. In this approach the graph nodes with multiple parents are not cloned, but introduced only once and connected with the parents when it is needed. The transformation to the graph structure happens as follows:

1. The reasoner finds all subclasses of the given concept C: If $B \sqsubseteq C$ then C is connected to B with a red edge.
2. The reasoner finds all superclasses of the given concept C: If $C \sqsubseteq B$ then C is connected to B with a green edge.
3. Inferring that C and E are related to each other via R requires extraction of properties from axioms of the concept. Then the reasoner can be asked whether the concept $C \sqcap \neg\exists R.E$ is satisfiable, if not then it is necessary for the class C to have the property R with the filler E.

This procedure begins with the root concept and recursively calls itself till the certain level of depth is reached. This level of depth is a parameter of the user interface. Although the properties make sense in context of individuals, we use them to describe the classes in an ontology. We can justify our method, because we choose only the properties which are necessary for the given class.

3.2 Hierarchical Tree Representation

The skeleton for the containment visualization approach is the subsumption hierarchy that represents subclass relationships between concepts in an ontology. This hierarchical graph can be generated from the *told hierarchy*[1] or from the *inferred hierarchy*[2]. However, the hierarchy is not a tree but a directed acyclic graph, because classes in ontologies can have multiple parents. To adapt this graph structure to the tree representation, we clone each node when it has more than one parent. The tree generations begins with the root node (*Thing*). The procedure visit direct subclasses of *Thing* and add them to the children set of the root node. Then, the procedure call itself recursively on each node. When this walk reach a node that is already visited from some other parent, the node will be cloned, but the subtree of this cloned node will not be visited/visualized for the second time. Although the introduction of clones of classes can be confusing for the user, because the visualization shows more classes than there are actually present in an ontology, it is unavoidable by this representation.

4 Visualization

Two different visualization techniques represent ontology structure in different ways. In this section we describe this approaches separately.

4.1 Visualization of Node-Link Representation

The visualized graph data-structure is directly generated from the ontology (OWL/OBO serialization) by means of the transformation process (see Sect. 3.1). The graph can be generated for different levels of depth. In the visualization a user can navigate through the graph and expand each concept. When the concept is expanded, the graph structure for this concept is generated. The level of depth is the parameter provided by the user interface. During the ontology exploration the user can return to the previously chosen concepts.

In addition, a simple *keyword search* interface is provided, where the user can retrieve the concepts which are syntactical related to the given query term. The search is implemented as a string comparison between the query and annotated properties of all concepts in the ontology. The concepts that match the query

[1] Told hierarchy reasoning is very simple form of the reasoning, where the subsumption hierarchy is directly extracted from ontology axioms, this simple reasoner is part of the OWL-API [16].

[2] Inferred hierarchy is generated with the Pellet [17] reasoner.

term are presented to the user in a list (list box element). From this list a concept can be chosen (mouse interaction) and the corresponding graph structure for this concept will be generated.

Because with this method we are visualizing the graph structure, we have equipped the user interface with a *path search* functionality. After the user have inserted two query terms the paths between these terms will be searched in the ontology. All paths with the length less than 4 will be represented in a list. After that, the path of interest can be chosen.

4.2 Representations of Views Based on Different Geometry

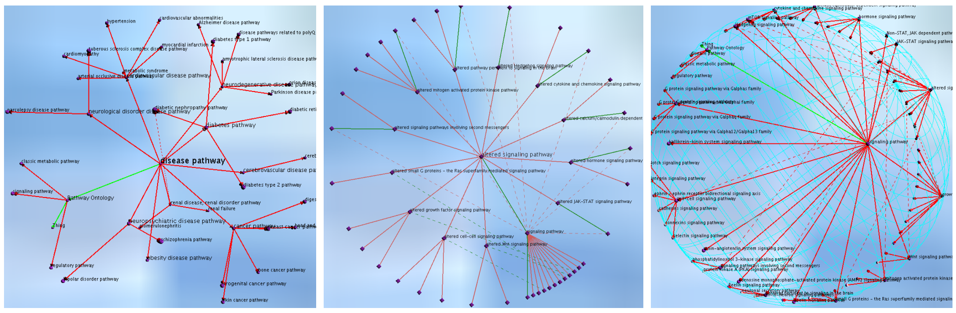

Fig. 1: Visualization of ontology with Sphere, Disk and Stereographic models.

We have implemented two Euclidean views; i.e. *Sphere* and *Disk*. In the *Sphere* visualization, the root concept is placed in the center of the visualization window. The sub/super classes and related concepts surround the root concept and are evenly spaced on the imaginary sphere surface (see Fig. 1). Two other representations, i.e. the Klein Model, and the Poincaré Disk model (see Fig. 2), are based on the hyperbolic geometry and realize the so called ”focus + context” techniques [18, 19]. In addition, we have also implemented the *Stereographic* view (see Fig. 1), where the graph structure is laid out at the surface of a sphere. All the views are augmented with the corresponding geometrical transformations. In Euclidean view the Euclidean transforms are implemented, whereas for the Klein model the hyperbolic transforms [20] are applied, and in Poincaré model the Möbius transformations are used.

4.3 Visualization with Containment Approach

The tree skeleton for the containment approach is built as previously described (see Sect. 3.2). The nodes are places on the surface of a sphere in the hierarchical layers. The fist layer is formed with the direct children of *Thing*. The second layer is formed by the children nodes of the nodes of the first layers. Each node is represented as are spherical cap, where the spherical cap area depends on the leaf number of the corresponding node. The children nodes are sorted in

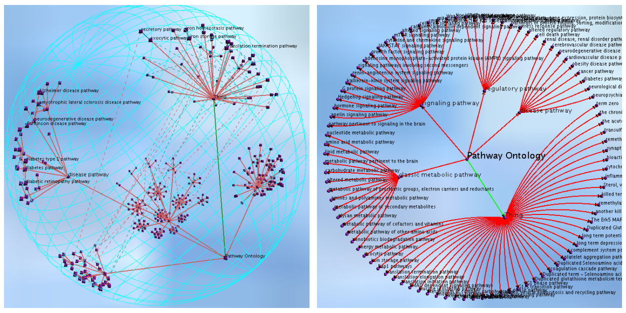

Fig. 2: Visualization of ontology with hyperbolic geometry. The left figure shows the Klein model, and the right figure shows the Poincaré model.

descending order on the basis of the leaf number. The corresponding children caps are nested in the parental cap, and are placed at the surface of the sphere, which has a bigger radius, to give an impression of a relief in the visualization. We can compare the nodes of the first layer with the continents on the surface of the Earth. The nodes of the second layers, then, will represent countries inside the continents. The nodes of the third layer will represent the provinces or republics.

Semantical Zoom In our approach we introduce the semantical zoom technique which augments the standard mouse zoom functionality with the semantic zoom. In the case of the geometrical zoom, when the visualized object approaches the viewer the representation of that object is not changed, but only scaled. With the semantic zooming the distance change triggers the activation or deactivation of the details. In the case of the visualization of a hierarchy, the closer the sphere representing an ontology is, the more levels of the hierarchy are visualized (see Fig. 3.) Using this technique the user can explore a hierarchy in a simple and an intuitive manner.

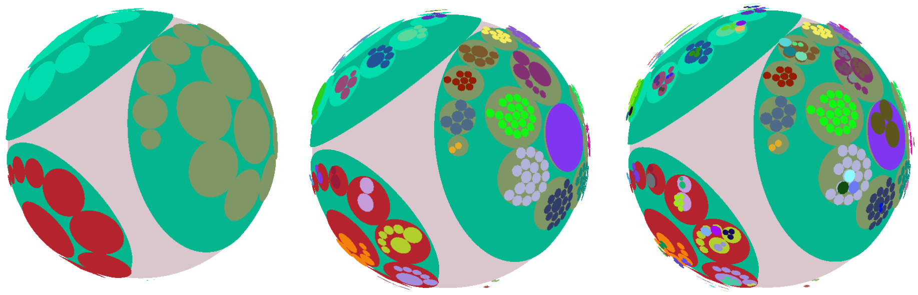

Fig. 3: Consecutive changes of the levels of ontology during zooming in. The labels of nodes can be retrieved on demand and are not shown here.

5 Conclusion and Discussion

In this paper we have described our ontology visualization approach. We have elaborated on two tree visualization techniques, e.g. node-link and containment, and combined them in one ontology visualization tool.

Each geometrical representation has its advantages and shortcomings. Therefore, the question which view fits best for an ontology does have more than one answer. Representation of an ontology with multiple geometrical views can help the user to better understand the ontology, because different geometries emphasize the graph structures differently.

The global information, i.e. what this domain is about, what are concepts in this domain, is well visualized with the hyperbolic geometries. Alternatively, if we want to see the local context of the concept of interest, then the Euclidean views are more suitable.

In Euclidean geometry each graph node needs the equivalent space for its visual representation. However, the backbone of the graph is a spanning tree structure, and number of nodes increases exponentially in the tree. This is the reason why Euclidean views are only sufficient for a small number of layers; from our experience not more than three.

The hyperbolic views offer a possibility to use more space, because the circumference of the circle in the hyperbolic plane increases exponentially with the radius. This is the reason why the hyperbolic geometries are more suitable for the representation of global visualization of larger structures.

With the stereographic view the user can benefit from $3D$ representation of the graph, laid out at the plane, because this graph is mapped at the surface of the hemisphere with the stereographic projection. This view, however, shares the disadvantages with the Euclidean view, because the individual nodes in the deeper layers are difficult to distinguish in a crowd of concepts.

The containment representation in our approach is suitable for representation of a hierarchical structure of an ontology. We assert this on the basis of the fact that other containment ontology visualization approaches, like Jambalaya [11] are very popular in ontology community, however, Jambalaya shows only part of the hierarchy, and needs multiple interaction steps in order to comprehend the hierarchy. In our approach the hierarchy of an ontology can be explored intuitively, by using standard zoom functionality, which is enhanced with semantic zoom technique. Although our method is meant to visualize the whole hierarchy, at this moment it is only suitable for small ontologies. We will work further on solving this problem.

Our visualization methodology is available on-line [21] for probing and inspection as a Java Web Start application.

Acknowledgements

This work has been partially supported by the BioRange program of the Netherlands BioInformatics Centre (NBIC, BSIK grant).

References

1. Patel-Schneider, P.F., Hayes, P., Horrocks, I.: OWL Web Ontology Language: Semantics and Abstract Syntax, W3C Recommendation, February 10, 2004. `http://www.w3.org/TR/owl-semantics/`.
2. Protégé, ontology editor and knowledge-base framework, `http://protege.stanford.edu/`
3. OntoEdit an ontology engineering environment, `http://www.ontoknowledge.org/tools/ontoedit.shtml`
4. Katifori, A., Halatsis, C.: Ontology visualization methods - a survey. AMC Computing Surveys, 39 (2007).
5. Eklund, P., Roberts, N., Green, S.: OntoRama: browsing RDF ontologies using a hyperbolic-style browser. In: Proceedings of the First International Symposium on Cyber Worlds (2002).
6. A TouchGraph Visualization Tab for Protégé, `http://users.ecs.soton.ac.uk/ha/TGVizTab/`
7. ToughGraph Navigator, `http://www.touchgraph.com/navigator.html`
8. Bosca, A., Bonino, D., Pellegrino, P.: OntoSphere: more than a 3D ontology visualization tool. In: SWAP 2005, the 2nd Italian Semantic Web Workshop, Trento, Italy, December 14-16, 2005, CEUR Workshop Proceedings, ISSN 1613-0073, online http://ceur-ws.org/Vol-166/70.pdf.
9. Dmitrieva, J., Verbeek, J.F.: Multi-view ontology visualization. The 11th International Protégé Conference, June 23-26, 2009 - Amsterdam.
10. Wang, T.D., Parsia, B.: CropCircles: topology sensitive visualization of OWL class hierarchies. In: Cruz et al. (eds.) ISWC 2006, LNCS, vol. 4273, pp. 695–708, Springer, Heidelberg (2006).
11. Storey, M., Musen, M., Silva, J., Ernst, N., Fergerson, R., Noy, N.: Jambalaya: interactive visualization to enhance ontology authoring and knowledge acquisition in Protégé. In: Workshop on Interactive Tools for Knowledge Capture (K-CAP-2001).
12. Shneiderman, B.: Tree visualization with tree-maps: 2-d space-filling approach. ACM Transactions on Graphics 11(1), pp. 92–99 (1992).
13. National Cancer Institute. Enterprise Vocabulary Services, `http://ncit.nci.nih.gov/`
14. Soon, T.T.: A study on multiple views for tree visualization. In: Proc. SPIE, Vol. 6495, 64950B (2007).
15. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P., editors. The Description Logic Handbook, Cambridge University Press, (2002).
16. The OWL-API: `http://owlapi.sourceforge.net/index.html`
17. Pellet: `http://pellet.owldl.com/`
18. Lamping, J., Rao, R.: The hyperbolic browser: a focus + context technique for visualizing large hierarchies. Journal of Visual Languages and Computing, 7, pp. 33–35 (1996).
19. Munzner, T.: H3: laying out large directed graphs in $3D$ hyperbolic space. In: Proceedings of the 1997 IEEE Symposium on Information Visualization, October 20-21 1997, Phoenix, AZ, pp. 2–10
20. Phillips, M., Gunn, C.: Visualizing hyperbolic space: unusual uses of 4×4 matrices (1991).
21. Multi-View Ontology Visualization, `http://www.liacs.nl/~jdmitrie/ontVis/OntologyVisualization.html`