Semantic Zooming for Ontology Graph Visualizations

Vitalis Wiens
University of Bonn
Bonn, Germany
wiens@cs.uni-bonn.de

Steffen Lohmann
Fraunhofer IAIS
St. Augustin, Germany
steffen.lohmann@iais.fraunhofer.de

Sören Auer
Technische Informationsbibliothek
Hannover, Germany
soeren.auer@tib.eu

ABSTRACT

Visualizations of ontologies, in particular graph visualizations in the form of node-link diagrams, are often used to support ontology development, exploration, verification, and sensemaking. With growing size and complexity of ontology graph visualizations, their represented information tend to become hard to comprehend due to visual clutter and information overload. We present a new approach of semantic zooming for ontology graph visualizations that abstracts and simplifies the underlying graph structure. It separates the comprised information into three layers with discrete levels of detail. The approach is applied to a force-directed graph layout using the VOWL notation. The mental map is preserved by using smart expanding and ordering of elements in the layout. Navigation and sensemaking are supported by local and global exploration methods, halo visualization, and smooth zooming. The results of a user study confirm an increase in readability, visual clarity, and information clarity of ontology graph visualizations enhanced with our semantic zooming approach.

CCS CONCEPTS

• **Human-centered computing → Visualization**; • **Information systems → Web Ontology Language (OWL)**;

KEYWORDS

Semantic Zooming, Ontology Graphs, Sensemaking, Navigation Support, Halo Visualization, Smart Expanding, VOWL

ACM Reference Format:
Vitalis Wiens, Steffen Lohmann, and Sören Auer. 2017. Semantic Zooming for Ontology Graph Visualizations. In *K-CAP 2017: K-CAP 2017: Knowledge Capture Conference, December 4–6, 2017, Austin, TX, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3148011.3148015

1 INTRODUCTION

A fundamental aspect of the Semantic Web is to create and communicate conceptualizations of the information and data in certain domains. Taxonomies, vocabularies, and ontologies serve this purpose by providing a formal, machine-readable representation of the domain knowledge. The larger and more interlinked such vocabularies and ontologies become, the more challenging it is for humans to explore and comprehend them. Visualizations of ontologies can offer a good starting point for exploration and can support sensemaking of the provided information [6]. In particular, graph visualizations in the form of node-link diagrams are often used to depict the structure of ontologies [20]. A node-link diagram represents the concepts of an ontology by a graph $G(N, L)$, where classes typically comprise the set of nodes N and their interrelations are described by the set of links L using the terms of the ontology.

Cognitive science shows that human capacity is limited in the aspect of storing and processing information [7]. Ontology graph visualizations tend to become hard to read with growing size and complexity, which is induced by visual clutter, crossing edges, and occlusion of rendering primitives. Additionally, every rendering primitive (e.g. circle, rectangle, text, link, arrow, and even color) can represent information. An information overload is a natural consequence when the amount of rendering primitives exceeds the cognitive and perceptive capacity of the user to comprehend the visual representation of the ontology. The conceptual elements of an ontology are visualized by multiple rendering primitives, because, for example, a depiction of a class requires at least two primitives (name and shape). This indicates that the amount of rendering primitives grows faster than the amount of nodes and links of the ontology graph visualizations, leading quickly to an information overload. Thus, the readability, information clarity, and visual clarity of the represented graph structure is not only affected by the number of classes and properties of the ontology, but in more general by the number of rendering primitives.

This paper proposes a semantic zooming approach for ontology graph visualizations that are depicted as node-link diagrams. Semantic zooming is a form of level of detail visualization [26]. In our approach, the information comprised in an ontology is partitioned and organized by three different information layers: 1) Topological Layer, 2) Aggregation Layer, and 3) Visual Appearance Layer. The simplification and abstraction for each information layer is realized in form of discrete levels of detail, which allow the user to focus on certain regions in the ontology graph visualization and explore its structure. Additional local exploration methods enable a focused investigation for class relations and attributes.

The **Topological Layer** describes a simplification of the topological graph structure. Discrete topological levels of detail are assigned to each class. These levels allow the user to adjust the visualization to a specified level of detail. Additionally, the number of presented nodes can be controlled using local exploration methods.

The **Aggregation Layer** is property-oriented and describes the aggregation of class attributes comprising datatype properties and object properties. Different aggregation methods are defined for multiple properties between distinct classes, datatype properties, and reflexive properties. Datatype properties and reflexive properties correspond to one class, thus we organize them on the same

---

K-CAP 2017, December 4–6, 2017, Austin, TX, USA

ACM ISBN 978-1-4503-5553-7/17/12…$15.00
https://doi.org/10.1145/3148011.3148015

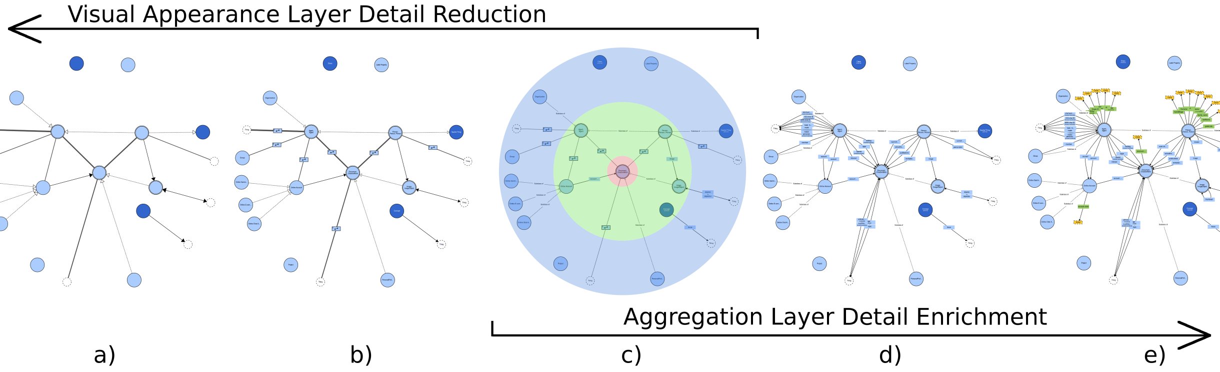


Figure 1: The cooperation of the different information layers with discrete levels of detail, and their influence on the graphical representation of the ontology *Friend of a Friend*. Images a) – c) show the incremental details of the visual appearance layer. In image c), the topological levels of detail are indicated by the colored circles. Red indicates the lowest, green the intermediate, and blue the highest topological detail level. Images c) – e) show the incremental levels of detail for the aggregation layer.

level of detail and enforce their collapse and expand operations to be performed simultaneously in the implementation of the semantic zooming approach for ontology graph visualizations.

The **Visual Appearance Layer** comprises removal of rendering primitives from the visualization (e.g. when the class name becomes to small to read) and style manipulations (i.e. line thickness of links and sizes of arrow elements). Thus, the visual representation of the depicted graph is adapted to the required level of detail. Formative expert interviews guided the definition of the discrete levels of detail with respect to the visual appearance of rendering primitives.

After reviewing the related work in Section 2, we introduce the approach of semantic zooming for ontology graph visualizations in Section 3 and apply it on a force-directed layout using the VOWL notation in Section 4. A summative user study in Section 5 shows the advantages of the new approach through a comparison of an existing ontology graph visualization framework (WebVOWL) and an extension of it with the semantic zooming approach. Already small and medium size ontologies can exhibit visual clutter, occlusion of rendering primitives, and information overload. While this paper focuses on a sophisticated method for these sizes of ontologies, the future work, as described in Section 6, aims to extend the semantic zooming for *large* and *very large* ontology graph visualizations, that introduce further challenges with respect to the responsiveness of a framework and sensemaking of the comprised information.

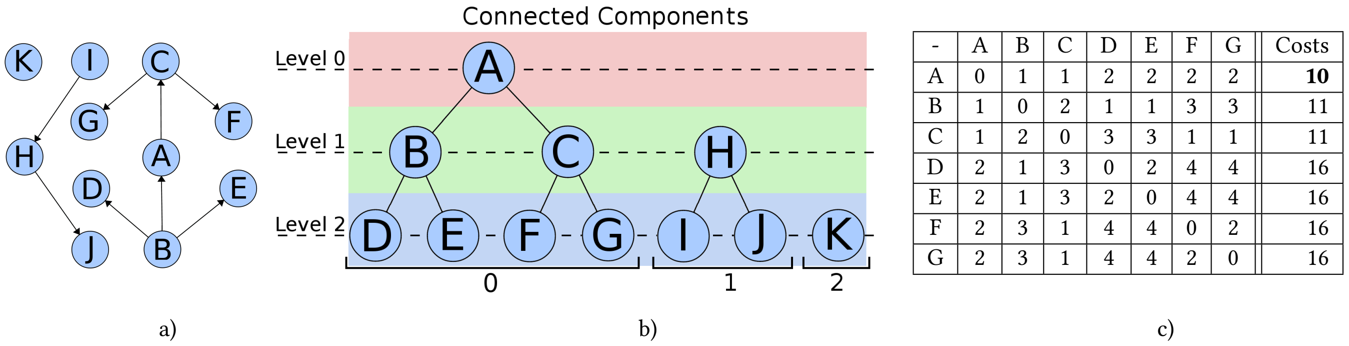
2 RELATED WORK

A number of ontology visualization techniques have been proposed in the last couple of years [8, 17, 18]. Many of the available approaches visualize ontologies as graphs using node-link diagrams. Typically, the nodes are the ontology classes and the links represent the property relations between them, as described above. While some approaches visualize only a particular type of property relations, such as the class hierarchy (e.g. OWLViz [16]), others aim at visualizing a larger set of property relations (e.g. OWLPropViz [25]).

The available approaches use different notations for the representations of nodes and links. On the one hand, popular notations are reused and adapted to ontologies, such as UML class diagrams [5] or Concept Maps [14]. On the other hand, new notations are proposed that have been specifically designed for the visual representation of ontologies, such as Graffoo [9] or VOWL [20]. Independently of the chosen visualization strategy and notation, all these approaches run into the problems of crossing links and primitive occlusion with an increasing size and complexity of the graph structures.

Some approaches apply techniques that help to reduce the occlusion of rendering primitives and decrease the number of crossing links. For instance, the implementations of SOVA [3] and WebVOWL [20] allow to filter certain property and class types on demand in order to reduce the number of nodes and edges in the graph visualization. Other works apply hierarchical edge bundling [1, 15] to group property links or apply force-directed graph layouts [3, 10, 20] to reduce the number of crossing links. OntoTrix [1] uses adjacency matrices to visualize dense parts of the graph structure underlying the ontology, which increases the visual scalability but does not show structural aspects in these dense graph parts any longer. While all these techniques can improve the readability of ontology graph visualizations, they cannot prevent that visual clutter, occlusion, and information overload still become a problem at some point, i.e., when the number of rendered elements is beyond the cognitive capacities of human perception.

In this paper, we present an approach that addresses this problem by letting the user dynamically control the number of elements and the amount of information that is shown in the visualization. The information comprised in an ontology is partitioned and organized by different information layers with discrete levels of detail. The abstraction and simplification is achieved using aggregation mechanisms and semantic zooming techniques. We are not aware of any other work that applies these techniques to the visualization of ontology graphs. While basic forms of semantic zooming have been used in the ontology visualizations OWL-VisMod [13] and Jambalaya [23], they are very limited compared to our approach: They simply let the users switch between different views and windows of the graphical user interface, but do not apply the semantic zooming to the ontology graph itself. Thus, this is the first approach presenting a sophisticated method for semantic zooming for ontology graph visualizations.



a) b) c)

-	A	B	C	D	E	F	G	Costs
A	0	1	1	2	2	2	2	**10**
B	1	0	2	1	1	3	3	11
C	1	2	0	3	3	1	1	11
D	2	1	3	0	2	4	4	16
E	2	1	3	2	0	4	4	16
F	2	3	1	4	4	0	2	16
G	2	3	1	4	4	2	0	16

Figure 2: Assignment of the global topological levels of detail: a) Input graph, b) Minimum spanning tree organization, c) Path matrix for the largest connected component of the input graph, indicating the computation of the costs for the exploration.

3 APPROACH

The semantic zooming approach for ontology graph visualizations presented in this paper partitions and organizes the terminological box of an ontology into three information layers. For each layer, discrete levels of detail control the amount of visualized primitives and thus also control the amount of information that is presented to the user. Following Shneiderman's visual information seeking mantra "overview first, zoom and filter, then details-on-demand" [22], these three information layers allow the user to focus on certain regions in the visualization and explore the structure of the ontology on different detail levels.

3.1 Topology Simplification

The topology simplification is applied on a simplified undirected graph $G'(N', L')$. In contrast to the graph $G(N, L)$ that reflects the full ontology graph structure, the set of nodes N' represents only the classes of the ontology (no datatypes). The set of links L' represents only object properties connecting two different classes. Additionally, multiple connections between two classes are reduced to a single undirected link. The excluded datatype properties and object properties are reintroduced to the visualization in the property-oriented aggregation layer.

A breadth-first search determines the set of connected components $C = \{c_0, \cdots, c_n\}$ of the simplified graph G'. Based on the topology of the connected components, a topological detail level D is assigned to each node. The graphical representation is adjusted to a specified level of detail by reducing or enriching the visualization accordingly to the detail levels of the class nodes N'. In our approach, we differ between *global* (D_{global}), and *local* (D_{local}) topological detail levels. The global detail levels are used for global exploration of the whole ontology graph, including the removal of connected components at the lower topological detail levels as indicated in Figure 2b. The local detail levels are used for the exploration of a particular connected component and allow a focused investigation of certain regions in the graph using collapsing of topological branches and expansion of adjacent nodes.

The local topological detail levels $D_{local}(c_i)$ of an individual connected component $c_i \in C$ are determined by identifying the pairwise relational distances between the classes. The distances are computed by investigating the object properties of the ontology that connect the classes of that connected component. The distances for each class are stored in a path matrix M_i. An entry in the matrix $M_i(j, k)$ describes the number of steps that are required for a node $n_j \in c_i$ to reach another node $n_k \in c_i$. A row-wise summation of the relational distances reflects a cost value $V(n_j) = \sum_k M_i(j, k)$ for the exploration of a connected component starting with a node n_j. An example for the cost values is shown in the "costs" column of the table in Figure 2c. The row r_j of the path matrix M_i with the minimal cost value $V(n_j)$ is selected to determine the local topology of the spanning tree. Similar to the works addressing the centrality measures in graphs [4, 11], we compute the pairwise relational distances between nodes. In contrast to a centrality measure for a node, we obtain directly its topological detail level through relational distances of the row r_j.

Several optimal paths may exist, this occurs when $V(n_q) = V(n_r)$ with $n_q, n_r \in c_i$ and $n_q \neq n_r$. Each optimal path corresponds to a row with the minimal cost value V. In our approach, we assign the topological detail levels based on the minimal relational distance corresponding to one of the paths. Thus, the lowest topological detail level does not include a single but several root nodes for the minimal spanning tree organization.

The computation of the minimal spanning tree determines the local topological detail levels D_{local} for a connected component and additionally achieves its exploration with the minimal amount of expansion operations.. A path matrix for the largest connected component of a synthetic input graph in Figure 2a is illustrated in the table of the Figure 2c. The local expansion operation of a class node adds all directly connected class nodes to the visualization. The local collapse operation identifies additionally all indirectly connected classes by computing a 'hiding chain', which recursively traverses and accumulates all directly connected class nodes with a higher local topological level of detail, starting from the node of operation. This enables the user to collapse a topological branch and reduce the visualization such that the focus can be directed on particular regions in the corresponding graph.

The global topological detail levels D_{global} are determined by aligning the spanning trees in such a way that their highest local topological detail levels are equal. Each spanning tree corresponds to a connected component $c_i \in C$ and has a tree depth t_i, which is determined by the highest local topological detail level of the

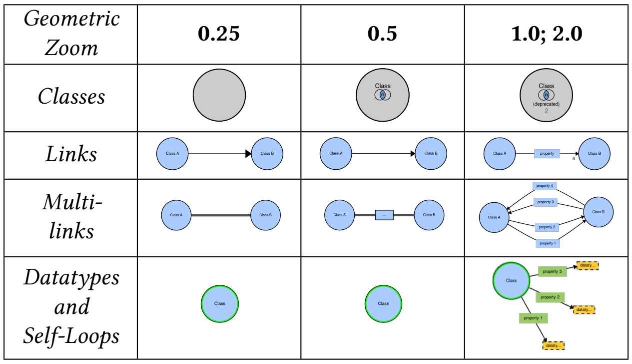


Figure 3: Definition of the discrete levels of detail for the visual appearance layer, defined by formative expert interviews for different geometric zoom levels.

class nodes in this connected component. Based on the maximal spanning tree depth $t_{max} = \max(t_i)$, the total number of global topological detail level is set to t_{max}. The spanning trees are inserted into D_{global} with an offset $h = t_{max} - t_i$ which is added to the local topological detail levels of the connected component. This alignment of the spanning trees enables the removal of complete connected components from the visualization. Additionally, large connected components, most likely providing more relevant information, will be visualized first, while smaller connected components are added to the visualization with increasing topological detail levels, as shown in Figure 1 and Figure 2b.

In the implementation of the approach, we additionally assign each leaf node the highest level t_{max} for the global topological detail levels. This enforces their introduction to the visualization on the highest detail level. A justification for this design decision is, for example, the VOWL notation, which multiplies the class `owl:Thing` when it is connected to different classes in the ontology, as shown in Figure 1. Additionally, the expansion of leaf nodes on the highest global detail level provides a visual indication to the user when the topological layer reaches its maximal level of detail.

3.2 Aggregation

The aggregation layer addresses class attributes comprising datatype properties, reflexive properties, and multiple properties between different classes. The open world assumption in the ontology domain allows an unlimited amount of these attributes. Thus, they can generate cluttered graph representations without providing more insight into the underlying structure of the ontology.

Multi-links enrich the simplified connection $l' \in L'$ between two nodes by introducing a directed link and visualizing multiple connections that are described by the different object properties. Datatype properties and reflexive properties are assigned to an individual class; thus, we enforce in our approach that these two have the same detail level, as they represent attributes corresponding to a particular class. In our approach, the aggregation layer consists of the following levels of detail: 1) Aggregation of all attributes, 2) Expansion of all multi-links, 3) Expansion of all attributes.

The first level reduces the visualization by removing all aggregated attributes, allowing the user to obtain an overview of the graph structure. The second level introduces more details by expanding all multi-links, and thus provides information about the different relations between the classes. The third level adds the remaining attributes (datatype properties and reflexive properties) to the visualization, and thus provides full details about the attributes of the different classes. The three discrete detail levels are shown in Figure 1c, d, and e, which illustrates the aggregation layer using the VOWL notation. Additionally, local exploration methods enable the user to collapse or expand the attributes for individual class nodes and multi-links, which allows to focus on certain exploration tasks.

3.3 Visual Appearance

The third layer addresses how the visual appearance of rendering primitives is modified. The observation that on a zoomed-out visualization of the graph, the class names and additional symbols become to small to read, raises the specifications for this particular information layer. No additional insight is obtained when the user can not clearly see or read the provided rendering primitive. Instead, these small primitives introduce noise and distraction to the visualization. Thus, excluding these unreadable rendering primitives results in a much clearer visualization, which additionally boosts the performance of the underlying framework, as fewer primitives stress the rendering pipeline.

We obtained the definitions of the discrete detail levels for this information layer from expert interviews with five participants, who had at least five years of experience in the ontology domain. However, finding the most appropriate number of discrete levels of detail still remains an open research question at this point. The interview was conducted with a framework of four graphical views with different geometric zoom levels (0.25, 0.5, 1.0, and 2.0). The highest geometric zoom level was chosen in order to fit the largest graph of the questionnaire on the screen. The questions were designed such that for the different geometric zoom levels, the visual appearance of classes, object properties, datatype properties, reflexive properties and multiple connections between classes, are identified and assigned to one level of detail. The ontology experts had to choose, for each geometric zoom level, from a set of options which modify the visual appearance of the primitives respectively. Based on the experts' decisions, the majority of votes defines for the individual primitives on which geometric zoom level these appear. The link thickness and arrow sizes are determined by the average value respectively. Based on the outcome of the expert interviews, the defined levels are illustrated in Figure 3.

As different screen resolutions might have affected the outcome of the formative expert interviews, all interviews had been performed on a 24" display with a resolution of 2560 × 1440 pixels.[1] The evaluation of the expert interviews shows that the geometric zoom levels 1.0 and 2.0 both have the same visualizations. Based on these results, we implemented only three levels of detail for the visual appearance layer in the summative user study. Despite the fact that this information layer describes the visual appearance of rendering primitives on a certain geometric zoom level, we do not enforce the manipulation of the rendering primitives to be performed together with a geometric zoom operation, but rather let

---

[1] The experiment is available online at http://vowl.visualdataweb.org/semzoom/config/

the user set the discrete level that fits to the underlying task. Nevertheless, a toggle option in the implementation of the approach allows the user to perform the manipulation of rendering primitives together with the geometric zoom, similar as in geographical map applications like Google Maps.

4 IMPLEMENTATION

We applied the approach of semantic zooming for ontology graph visualizations to node-link diagrams with force-directed layout using the VOWL notation. In particular, we extended the implementation provided by WebVOWL [19]. The extended implementation of the approach exploits and restricts different aspects of the ontology graph visualizations with respect to the VOWL notation. Disjoint properties (`owl:disjointWith`) are removed from the set of links L', that are used in the first information layer for the identification of the local topological detail levels of class nodes. The observation of real-world datasets indicates that these tend to generate cluttered visualizations with a lot of crossing edges, without comprising high information value. Additionally, in the implementation of the approach we exploit the VOWL notation for datatypes and datatype properties. Due to the fact that for each datatype property, a corresponding datatype node is generated in the graphical representation, these are connected to one particular class by the domain of the property, and thus they do not exhibit multiple connections. This allows a true conceptual separation in the aggregation layer between multiple object properties connecting two classes and datatype properties connecting a class node to a datatype node.

Ontologies are usually distributed in a textual format and do not carry any spatial information. In particular, the position of a node in the graphical representation is not part of the definition of the ontology. Force-directed layouts use a random node position initialization if no spatial information is provided. An optimization process iteratively realigns the positions of the nodes to generate a visual appealing representation of the ontology [12]. The forces between the nodes are optimized also with respect to the constrains provided by their links. Due to the random position initialization and the optimization process, the graphical representation exhibits different layouts which may interfere with the users mental map from a previous visualization of the same ontology.

Our approach introduces collapse and expand operations on the topological layer and on the aggregation layer. Different expansion operations add new elements to the force-directed layout in the implementation. The positions of these new elements are either random or not consistent with the current layout of the graph. Due to the optimization process, this non-optimal positions can introduce strong forces in the layout, leading to significant changes of the positions of the already present elements in the visualization. The mental map of the user could be distorted by this significant modifications to the ontology graph visualization. Additionally, as indicated in Figure 4b, the result of the optimization process may not always generate a visual appealing representation of the ontology because it is based on finding local optima. In the implementation of our approach, the mental map is preserved by using what we call *smart expanding* methods, which consider the order and positions of elements.

4.1 Multi-Link Expansion

In the WebVOWL application, the representation of multiple properties connecting two different classes introduces primitives that are used in the force-directed layout optimization. The expansion operation considers the relative positions (a and b) of the two participating nodes and determines the positions of the corresponding collective of rendering primitives. The relative distance and direction between two nodes is described by a vector $v = b - a$. An orthogonal vector $w \perp v$ describes the direction of the expansion for the rendering primitives. The initial positions of rendering primitives are stacked along the vector w. Additionally, the center of the stacked primitives is aligned with the center of the link connecting the two class nodes. A symmetric and smooth fanning out animation is generated by the optimization process due to the new initial positions which are near the optimal layout configuration from the previous computation. This technique does not only reduce the computation time for the optimization process, but also achieves a stable and deterministic expansion of the rendering primitives with a visual appealing fanning-out animation.

4.2 Single Class Attribute Expansion

The expansion operation for datatype properties and reflexive properties determines for their parent class node n_i two disjoint sets of elements ($T(n_i)$ and $E(n_i)$). The current topological detail level and the users interactions (expand and collapse operations on the topological layer) affect the set of class nodes that are present in the current layout of the graph. Based on the current configuration of the topological layer, the set of class nodes $T(n_i)$ represents the directly connected neighbors of the node n_i. The set of elements $E(n_i)$ comprises datatypes properties and reflexive properties that are added to the visualization by the expansion operation.

The positions of the nodes comprised in the set $T(n_i)$ are used to determine angles on a unit circle with respect to the position of the node n_i. These angles are sorted in the full 360 degree representation and their adjacent difference is computed. The pairwise difference indicates a free angular space for the expansion operation by providing start and end angles (α_s and α_e). The largest free angular space is a good candidate for the positions of the expanded elements. It exhibits the maximal free space in the graph layout and, additionally, the newly added elements interfere less with the already present class nodes in the force-directed layout optimization process. The elements, comprised in the set $E(n_i)$, are equally distributed in the largest free angular space with an angular offset $\alpha_h = \frac{|\alpha_s - \alpha_e|}{|E(n_i)|}$. A Class node n_i with an empty set of neighboring nodes ($T(n_i) = \emptyset$) uses the full 360° free angular space. A class node with only one neighboring node $|T(n_i)| = 1$ uses a free angular space of 90°, where the center of the free angular space is aligned with the direction from node n_i to the single node in $T(n_i)$ on the opposite of the unit circle. Based on the computed angles, the initial positions of the elements $E(n_i)$ are set relative to the position of the node n_i with a fixed distance, reflecting the half-length of the original link. A smooth expanding animation is achieved by this reduced relative distances of the expanded elements to the node n_i. These initial positions provide already a good but not optimal starting point for the force-directed layout optimization process.

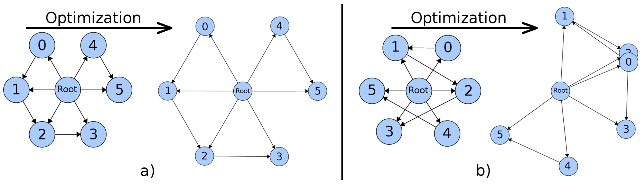


Figure 4: Order is crucial for multiple interconnected nodes. a) Correct ordering of nodes in the initialization and the resulting planar layout after optimization. b) Initialization with wrong ordering of nodes results in a non-optimal layout of the graph with crossing edges and reduced readability, because the optimization process ended in a local minimum.

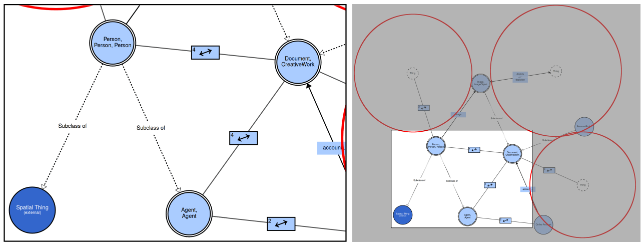


Figure 5: Halos for nodes of interest. Left: Screenshot of the approach implementing halo visualization. Right: Artificially generated image with halos and indicated viewport.

4.3 Topological Expansion

The expansion operations in the aggregation layer exploit for datatypes and self-loops the fact that these are not interconnected between each other in the VOWL notation. Each expanded element has only one property that connects it to the parent node n_i and thus the ordering of the elements in the expansion is not of importance. The order of expanded nodes becomes crucial when these have multiple interconnections as illustrated, for example, by the nodes with the labels 0, 1, 2, and 3, in Figure 4a. In order to achieve a visualization without crossing edges, if the set of expanded class nodes exhibits a planar graph layout organization, this expansion operation orders the set of expanded nodes $E(n_i)$ and initializes their positions in the same fashion as described in Section 4.2.

The interrelations in the set of expanded nodes $E(n_i)$ are identified and pairs of connected class nodes are built based on their IDs in the graph layout. Sequences of IDs are determined by concatenating and reordering the pairs such that only matching IDs are used in the concatenation, implementing a domino tiles like behavior that corresponds to a chain. The duplicated ID entries comprised in one chain are removed, providing the ordering of the corresponding class nodes. The final ordering of the expanded class nodes is achieved by a concatenation of all duplicate-free chains. Class nodes, of the expanded set $E(n_i)$, that do not participate in any chain are added after the last chain, so they do not interfere with the already correctly organized class nodes. Based on the ordering, the expanded class nodes are added into the free angular space. When the connection between the expanded nodes is more complex than an open or closed chain of IDs, this technique cannot generate crossing-free layouts. Neglecting the order of the interconnected nodes results in crossing edges and decreases the readability of the graph, as illustrated in Figure 4b.

4.4 Navigation and Exploration Support

Ontology graph visualizations in the form of node-link diagrams require search and localization functionalities to support navigation and exploration of the comprised information. We implemented a search bar which generates a dictionary that is based on the terms of the ontology. The search query is analyzed and the best matching entries of the dictionary are suggested to the user. The searched elements corresponding to the users selection are visually highlighted in the ontology graph visualization. Elements that are inside the viewport are highlighted by an outer border of the rendering primitive (i.e. a circle for class nodes and a rectangle for properties and datatype nodes in case of VOWL). Elements that are outside the viewport are visually indicated to the user using halos [2]. A halo is a circle having a dynamic radius. As illustrated in Figure 5, the radius of the halo is adjusted such that a part of the circle is rendered at the border of the viewport. The visible arc of the halo enables the user to virtually extend the screen space and estimate the location of the searched element, by completing the arc mentally to a circle and determining its center. Rectangular highlight elements are transferred into circles when they are located outside the viewport. Thus, their positions in the off-screen are also indicated by an arc of the halo element.

A localization functionality translates the viewport such that its center and the center of the element of interest are aligned. Additionally, the localization request preserves the overview and the context of the graph by a smooth geometric zooming and panning [24]. If multiple elements are highlighted by the selected search query (e.g. multiple representations of "Thing", as in Figure 5), the localization operation alternates between these multiple elements.

5 EVALUATION

A user study was conducted in order to evaluate the benefits of ontology graph visualizations enhanced with semantic zooming. Two versions of a visualization framework using a force-directed layout and the VOWL notation are directly compared with respect to the extended features of our semantic zooming approach. The user study included 12 participants. The participants comprised computer science students of the University of Bonn, and employees of the Fraunhofer IAIS institute. The age of the participants was in the range of 23 and 63 years, with a median of 29 years. The self-estimation of the participants' skills and the average experiences with graph visualizations and ontologies is shown in Table 1.

5.1 Study Design

The study consisted of two consecutive experiments for each participant. The first experiment was performed on the framework without the semantic zooming approach (version A), and the second experiment is performed on the framework for ontology graph visualization enhanced with semantic zooming, halos, smooth zooming, as well as search and locate functionality (version B). Both versions

Expertise level	Novice	Skilled	Expert	avg. Years
Graph Visualization	7	4	1	1.75
Ontologies	6	6	0	1.52

Table 1: Expertise levels of the study participants, and their average years of experience with graph visualizations and ontologies respectively.

#	Task	#	Task
1	Identify the main component of the ontology graph.	5	Find the object property R.
2	Name the key classes of the main component.	6	What are the domain and range of this property?
3	How many properties link the classes X and Y.	7	Press the button "reset" and relocate property R.
4	How many datatype properties has the class Z.	8	Freely explore the ontology graph.

Table 2: Performed tasks, where X,Y,Z,R refer to variable class or property names of the corresponding ontologies.

used in this study are available online.[2] All experiments were performed on a 24" display with a resolution of 1920 × 1200 pixels, and the implementations were executed in full-screen mode. Additionally, keyboard and mouse were used for the interaction with the ontology graph visualizations.

The participants were equally divided into two groups (G_A and G_B). Group G_A started with the ontology *Friend of a Fried (FOAF)*, and proceeded in the second experiment with the ontology *Semantically-Interlinked Online Communities (SIOC)*. Group G_B performed the same experiments, but the order of the ontologies was reversed. In both experiments, as indicated in Table 2, the participants had to search for classes, identify different relations, and count the number of links or datatypes respectively. The tasks have been designed to familiarize the participants with the functionalities of the two implemented versions.

After the use of each version, the participants had to rate them along various dimensions. The rating on each dimension used a scale ranging from one to five, where one maps to very low and five maps to very high. Additionally, the participants were asked to list positive and negative aspects of the corresponding version of the framework.

5.2 Results

The results of the user study reveal that the application of ontology graph visualizations enhanced with semantic zooming outperforms the one without it. This is reflected in the higher ratings, as shown in Figure 6, for *Readability*, *Visual Clarity*, *Information Clarity*, *Navigation Support*, and *Layout Stability*.

In version A, most of the participants liked the ontology graph visualization in form of a force-directed node-link diagram using the VOWL notation. The intuitive interaction with the visualization framework (navigation via panning and zooming using the mouse) was positively acknowledged by most participants.

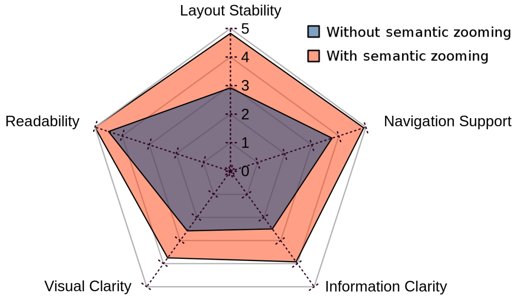


Figure 6: A star-plot visualization of the results obtained from the user study comparing the two versions.

In version B, which implements the presented approach, all of the participants positively acknowledged the control of information using the different information layers and their discrete levels of detail. Smart expanding was also positively remarked with the comments: "optimal organization of expanded nodes" and "deterministic behavior of the expansion methods". Search, locate, and halo visualization were appreciated by most of the participants. All participants gave a positive feedback for the implementation of the approach. However, most of the users wished for a tooltip functionality (or similar) for the different information layers, and additional visual indicators for the different levels of detail. The majority of the participants mentioned that the control mechanisms need first to be learned and fully understood in order to obtain the full potential of the semantic zooming for ontology graph visualizations. In the same context, the majority described the learning curve as steep and the control mechanisms as easy to learn.

However, the user study can only be considered a preliminary evaluation, indicating some benefits of the presented semantic zooming approach. The small and selective group of participants and the simple evaluation setup limit the validity of this user study. Additionally, the non-visualization related functionalities (search and locate) implemented in version B may have introduced a bias in the evaluation results.

6 CONCLUSION & DISCUSSION

In this paper, we have presented a semantic zooming approach for ontology graph visualizations. With growing size and complexity, the corresponding graph visualizations exhibit visual clutter, information overload, and occlusion problems, which make the underlying structure of the ontologies difficult to comprehend. Our approach partitions and organizes the information of an ontology into three information layers with discrete levels of detail, which allow the user to control the amount of the presented information.

An implementation of the new semantic zooming approach is evaluated on a force-directed layout using the VOWL notation. Control elements and different exploration methods allow the users a focused investigation by adjusting the visualization to their needs and examine parts of the ontology as it suits the required task: understanding and validating the ontology. Navigation in the ontology graph visualization is supported by search and locate functionalities. Additionally, halos enable the user to estimate the positions for searched off-screen elements. Localization of the searched elements

---

[2] Version A is available at: http://vowl.visualdataweb.org/semzoom/versionA/, version B at: http://vowl.visualdataweb.org/semzoom/versionB/.

is a smooth animation of the transition to the element of interest. The overview and context for the user is preserved by geometric zooming operations during the transition process.

Future work includes the extension of the new semantic zooming approach for *large* and *very* large ontology graph visualizations. The visualization of large ontologies and the integration with semantic zooming requires more investigation in order to achieve an intuitive and sophisticated exploration and navigation in their graphical representations. For example, in the current implementation of the approach, a collapsed element can neither be visually highlighted by a halo element, nor a localization operation can be performed on it. Only elements that are present in the current layout of the graph can be visually highlighted and located.

Large ontology graphs exhibit additional challenges. Only when the full ontology is transferred into the graph structure, a connected component analysis can be performed, which is the basis for the topological detail levels determination. In order to overcome this limitation, we plan to apply some kind of *key concept* analysis [21] on the ontology. Key concepts can provide additional information about the topology of the graph structure, in particular the assigned topological detail levels. In contrast, the current approach is based merely on the topology of the corresponding graph and does not comprise any information about the importance of a class nor does it integrate the semantics of the relations between classes (e.g. `rdfs:subClassOf`, `rdfs:subPropertyOf`, etc.). Additionally, key concepts are planned to be used for the ordering of connected components and their classes. Based on the importance of the connected components and their classes, this ordering can enable a streaming process of the underlying graph structure of the ontology. Thus, large ontologies will be processed in blocks of global and local topological levels of detail. A step-by-step generation of the detail levels for the topological layer and the aggregation layer can enable the user to explore the ontology already on a low detail level, while in the background, additional information is processed and added to the ontology graph visualization. Note that this streaming process also introduces new challenges in the synchronization of the global and local expansion operations.

Additionally, we plan to investigate methods that improve the performance and responsiveness of the framework for large ontology graph visualizations. The amount of rendering primitives affects the performance of every visualization framework (dynamic or static). Through the different levels of detail, a reduction of rendering primitives achieves already better performance because less elements stress the rendering system. Nevertheless, the performance will be the same when the framework shows the full details of the ontology graph visualization. Improving the performance will play a key role for the visualization of very large ontologies.

Overall, the discussion addresses only a portion of methods that advance the visualization of large ontologies. Apart from that, we hope that the semantic zooming approach for ontology graph visualizations will be useful to other researchers and developers with respect to sensemaking, exploration and validation of ontologies.

ACKNOWLEDGMENTS

This work has been supported by European Union's Horizon 2020 research and innovation programme (H2020 FET) to Fraunhofer IAIS for the GRACeFUL project under grant agreement no. 640954.

REFERENCES

[1] Benjamin Bach, Emmanuel Pietriga, Ilaria Liccardi, and Gennady Legostaev. 2011. OntoTrix: A hybrid visualization for populated ontologies. In *20th Int. Conf. on World Wide Web (WWW '11), Companion Volume*. ACM, 177–180.
[2] Patrick Baudisch and Ruth Rosenholtz. 1993. Halo: a technique for visualizing offscreen location. In *Human Factors in Computing Systems (CHI '03)*. 418–488.
[3] Tomasz Boinski, Anna Jaworska, Radoslaw Kleczkowski, and Piotr Kunowski. 2010. Ontology visualization. In *2nd Int. Conf. on Information Technology (ICIT '10)*. IEEE, 17–20.
[4] Ulrik Brandes. 2001. A faster algorithm for betweenness centrality. *Journal of mathematical sociology* 25, 2 (2001), 163–177.
[5] Sara Brockmans, Raphael Volz, Andreas Eberhart, and Peter Löffler. 2004. Visual Modeling of OWL DL Ontologies Using UML. In *3rd International Semantic Web Conference (ISWC '04)*. Springer, 198–213.
[6] Chaomei Chen. 2002. *Visualizing the Semantic Web: XML-Based Internet and Information Visualization*. Springer.
[7] Nelson Cowan. 2010. The Magical Mystery Four. *Current Directions in Psychological Science* 19, 1 (2010), 51–57.
[8] Marek Dudáš, Ondřej Zamazal, and Vojtěch Svátek. 2014. Roadmapping and Navigating in the Ontology Visualization Landscape. In *19th Int. Conf. on Knowledge Engineering and Knowledge Management (EKAW '14)*. Springer, 137–152.
[9] Riccardo Falco, Aldo Gangemi, Silvio Peroni, David Shotton, and Fabio Vitali. 2014. Modelling OWL Ontologies with Graffoo. In *The Semantic Web: ESWC 2014 Satellite Events*. Springer, 320–325.
[10] Sean Falconer. 2010. OntoGraf. http://protegewiki.stanford.edu/wiki/OntoGraf. (2010).
[11] Linton C Freeman. 1977. A set of measures of centrality based on betweenness. *Sociometry* (1977), 35–41.
[12] Thomas MJ Fruchterman and Edward M Reingold. 1991. Graph drawing by force-directed placement. *Software: Practice and experience* 21, 11 (1991), 1129–1164.
[13] Francisco José García-Peñalvo, Ricardo Colomo-Palacios, Juan García, and Roberto Therón. 2012. Towards an Ontology Modeling Tool. A Validation in Software Engineering Scenarios. *Expert Systems with Applications* 39, 13 (2012).
[14] Pat Hayes, Thomas C. Eskridge, Raul Saavedra, Thomas Reichherzer, Mala Mehrotra, and Dmitri Bobrovnikoff. 2005. Collaborative Knowledge Capture in Ontologies. In *3rd Int. Conf. on Knowledge Capture (K-CAP '05)*. ACM, 99–106.
[15] Walter Hop, Sven de Ridder, Flavius Frasincar, and Frederik Hogenboom. 2012. Using Hierarchical Edge Bundles to Visualize Complex Ontologies in GLOW. In *27th Annual ACM Symposium on Applied Computing (SAC '12)*. ACM, 304–311.
[16] Matthew Horridge. 2010. OWLViz. http://protegewiki.stanford.edu/wiki/OWLViz. (2010).
[17] Akrivi Katifori, Constantin Halatsis, George Lepouras, Costas Vassilakis, and Eugenia Giannopoulou. 2007. Ontology visualization methods – A survey. *ACM Computer Surveys* 39, 4 (2007).
[18] Monika Lanzenberger, Jennifer Sampson, and Markus Rester. 2009. Visualization in Ontology Tools. In *Int. Conf. on Complex, Intelligent and Software Intensive Systems (CISIS '09)*. IEEE, 705–711.
[19] Steffen Lohmann, Vincent Link, Eduard Marbach, and Stefan Negru. 2015. WebVOWL: Web-based Visualization of Ontologies. In *EKAW 2014 Satellite Events*. Springer, 154–158.
[20] Steffen Lohmann, Stefan Negru, Florian Haag, and Thomas Ertl. 2016. Visualizing Ontologies with VOWL. *Semantic Web* 7, 4 (2016), 399–419.
[21] Silvio Peroni, Enrico Motta, and Mathieu d'Aquin. 2008. Identifying key concepts in an ontology, through the integration of cognitive principles with statistical and topological measures. In *3rd Asian Semantic Web Conference (ASWC '08)*. Springer.
[22] Ben Shneiderman. 1996. The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations. In *IEEE Symposium on Visual Languages (VL '96)*. 336–343.
[23] Margaret-Anne Storey, Natasha F. Noy, Mark Musen, Casey Best, Ray Fergerson, and Neil Ernst. 2002. Jambalaya: An Interactive Environment for Exploring Ontologies. In *7th Int. Conf. on Intelligent User Interfaces (IUI '02)*. ACM, 239–239.
[24] Jarke J. van Wijk and Wim A. A. Nuij. 2003. Smooth and efficient zooming and panning. In *9th IEEE Symposium on Information Visualization (InfoVis '03)*. 15–23.
[25] Lutz Wachsmann. 2008. OWLPropViz. http://protegewiki.stanford.edu/wiki/OWLPropViz. (2008).
[26] Chris Weaver. 2004. Building Highly-Coordinated Visualizations in Improvise. In *IEEE Symposium on Information Visualization (INFOVIS '04)*. IEEE, 159–166.