Towards Imaging Large-Scale Ontologies for Quick Understanding and Analysis*

KeWei Tu, Miao Xiong, Lei Zhang, HaiPing Zhu, Jie Zhang, and Yong Yu

Department of Computer Science and Engineering,
Shanghai JiaoTong University, Shanghai, 200030, P.R.China
{tkw,xiongmiao,zhanglei,zhu,zhangjie,yyu}@apex.sjtu.edu.cn

Abstract. In many practical applications, ontologies tend to be very large and complicated. In order for users to quickly understand and analyze large-scale ontologies, in this paper we propose a novel ontology visualization approach, which aims to complement existing approaches like the hierarchy graph. Specifically, our approach produces a holistic "imaging" of the ontology which contains a semantic layout of the ontology classes. In addition, the distributions of the ontology instances and instance relations are also depicted in the "imaging". We introduce at length the key techniques and algorithms used in our approach. Then we examine the resulting user interface and find it facilitates tasks like ontology navigation, ontology retrieval and ontology instance analysis.

1 Introduction

Ontologies play a key role in the Semantic Web. More and more ontologies have been developed to formalize the conceptualization of a domain. In some applications, such conceptualization is so large and complicated that the resulting ontology will contain hundreds to tens of thousands of concepts and relations. For instance, the OWL version of the National Cancer Institute Ontology [1] contains more than twenty-seven thousand classes and seventy properties.

In order for users to understand and hence make use of such large-scale ontologies, there must be effective ways to present ontologies and facilitate user browsing and searching. Presenting a large ontology in plain text (e.g. XML files) is obviously unacceptable, as in this way users can learn nothing more than some individual statements. Almost all ontology engineering tools provide form-like UI to present interrelated statements together. For example, a typical "class view" gives an orderly list of the name, description, sub-classes, super-classes and properties of a certain class. However, such a view only presents to users a local view of the ontology, leaving users unaware of the ontology's holistic content and structure. A widely used ontology visualization approach is to draw an ontology as a graph (as in [2, 3]), with nodes representing classes and edges representing relations in most cases. In this way, users gain a much larger view of the ontology and could more conveniently navigate between different ontological

* This work was supported by IBM China Research Lab.

elements. In spite of these advantages, however, for large ontologies users' visual field in such an approach is often rather local. This is because when scaling the whole ontology graph to the screen size, one can often see nothing but a mess of lines.

An advisable visualization principle is to organize an ontology by its class hierarchy. Most ontology tools use trees to present the ontology hierarchy, although it is actually a directed acyclic graph (DAG). A hierarchy structure is a semantic organization of an ontology, so users can gain a holistic sense of an ontology when looking at the top levels of the hierarchy. In addition, seeking a certain concept in a hierarchy is much easier, as one can gradually narrow his searching scope when going deeper in the hierarchy. Apart from these merits, however, our practical experience shows that there are several shortcomings of hierarchical visualization.

- Hierarchy can not visualize some important information of an ontology such as the relations between classes.
- When users go deeper into a hierarchy, it is very easy for them to get lost and be unable to find the way home to the root.
- There is no semantic organization for sibling classes (i.e. the direct subclasses of a certain class). Typically they are listed alphabetically, and when the list is very long it becomes somewhat unreadable.
- Browsing a hierarchy is kind of inconvenient as it requires sometimes too many mouse-clicks to expand the nodes.

In this paper we propose a novel visualization approach to attack the deficiencies of current approaches. The main feature of our approach is that it presents a large-scale ontology by a holistic "imaging" which is semantically organized for quick understanding of the subject and content of the ontology. The semantic organization takes into account not only the class hierarchy information, but also other information such as relations and class similarities. Easy and friendly browsing and searching functions are provided on top of our visualization approach.

Another concern for ontology visualization is upon instances. In practice the number of instances populated in an ontology is often orders of magnitude larger than the class/relation number. A good visualization approach should also facilitate the analysis of ontology instances and instance relations. While most existing approaches do little on this aspect, our approach aims to meet this request.

It should be noticed that our approach is to complement existing approaches, instead of replacing them. In other words, our approach aims to facilitate the quick understanding and analysis of large-scale ontologies, while functions such as the details display of ontological elements are not provided. So one should use our approach together with existing approaches to achieve the best efficiency.

The rest of this paper is organized as follows. Section 2 presents the main techniques and algorithms used in our approach. Section 3 demonstrates the resulting visualization system by a few user scenarios. Finally we discuss the related work in section 4 and conclude the paper in section 5.

2 Techniques and Algorithms

In this section we present our approach of generating from an ontology its imaging. We use a simple RDF ontology (Figure 1) as the sample ontology throughout the procedure. Notice that, however, (i) our approach could be applied to other kinds of ontology as well, such as OWL ontology, and (ii) our approach is specially designed for large-scale ontology.

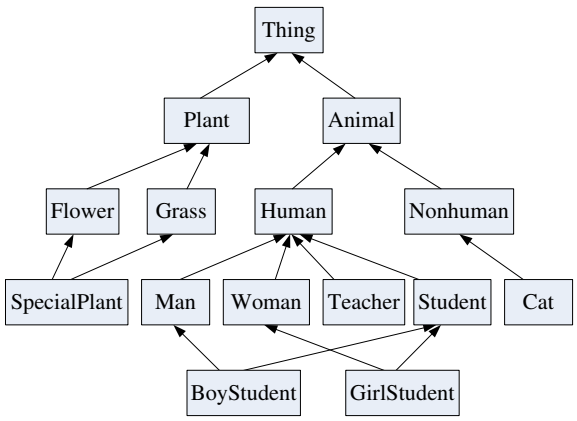


Fig. 1. The sample ontology (only the class hierarchy is shown, while the properties are omitted)

2.1 Semantic Connection Calculation

While an ontology is mainly composed of a set of classes, we need to lay them out in an Euclidean space to facilitate visualization. In order for such layout to be semantically meaningful, we first need to calculate the semantic connections between the classes.

As we know, a higher semantic similarity implies a stronger semantic connection, so we first calculate the semantic similarities between classes. The most classic semantic similarity calculation is based on the class hierarchy. For example, [4] proposed a formula as follows.

$$Sim_1(c_1, c_2) = \frac{2 \times N_3}{N_1 + N_2 + 2 \times N_3}$$

where N_1 and N_2 are the numbers of sub-class relations from c_1 and c_2 to their most specific common superclass C, and N_3 is the number of sub-class relations from C to the root of the class hierarchy.

Another kind of similarity calculation is based on the feature of the classes, such as the parts and attributes. For our sample RDF ontology, the only feature

is the properties of a class. The following formula is proposed by [5].

$$Sim_2(c_1, c_2) = \frac{\|C_1 \cap C_2\|}{\|C_1 \cap C_2\| + \alpha\|C_1 - C_2\| + (1-\alpha)\|C_2 - C_1\|}$$

where C_1 and C_2 are the feature sets of c_1 and c_2, and for our sample ontology they are the sets of properties of c_1 and c_2. α is defined as $\min(N_1, N_2)/(N_1 + N_2)$.

Besides class similarity, one may agree that the number of relations defined between two classes also contributes to the semantic connection, because more relations between two classes implies that they are more relative, and hence have a stronger semantic connection. This is depicted by the following formula.

$$Rel(c_1, c_2) = \frac{1 - e^{-n}}{1 + e^{-n}}$$

where n is the number of relations between c_1 and c_2.

In our approach, we take the weighted average of the three above-calculated values as the final evaluation of the semantic connection. The weights are customized by users, so that they can control what information would be presented in the visualization result. In order to speed up the subsequent processing, we set a connection threshold and discard those connections that are too weak. This threshold is also customizable, specifying the compromise between running speed and result quality.

For our sample ontology, the following table shows a part of the resulting semantic connections, i.e. those between the Student class and the other classes, with the threshold set to 0.6.

Student – Teacher	:	0.7257
Student – Animal	:	0.6333
Student – Human	:	0.8214
Student – Man	:	0.7175
Student – Woman	:	0.7175
Student – BoyStudent	:	0.8497
Student – GirlStudent	:	0.8497

2.2 Layout

After semantic connections are calculated, we can now lay classes out in a 2D plane, with strong connections implying small distances and vice versa. There are several algorithms that can be used to lay out a set of elements based on their preferred distances, such as Multidimensional Scaling (MDS) [6] and Force Directed Placement (FDP, also called Spring Embedder) [7]. In our approach we choose the FDP algorithm, because although it only achieves local optimality of the layout, it has some fast variations and could be used incrementally.

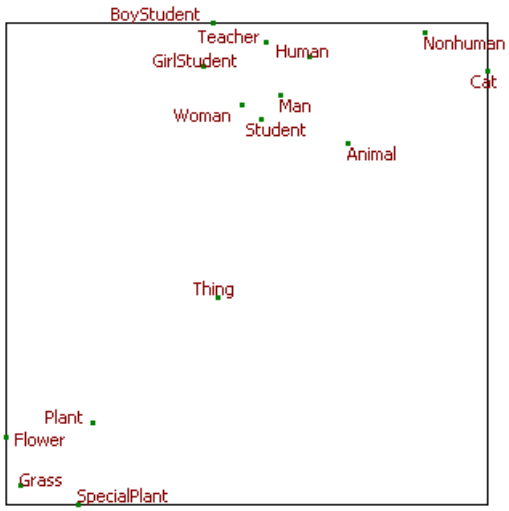


Fig. 2. Sample ontology: the result of FDP.

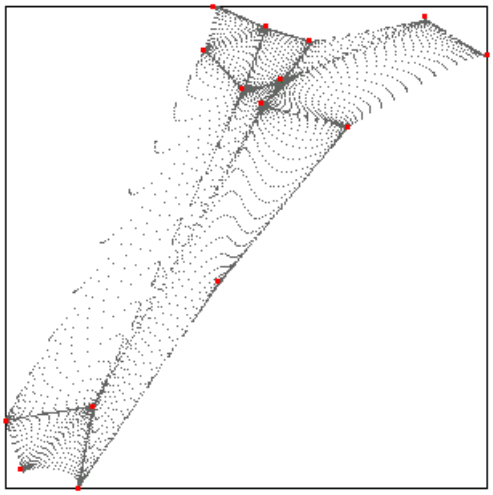
In standard FDP, elements are first randomly placed. With the forces between elements defined by their preferred and actual distances, they are moved according to the combined forces on them, until convergence. To speed up this procedure, we adopt a preprocessing method proposed in [8] before the standard FDP.

Figure 2 shows the result of running FDP on our sample ontology. As we see, classes with strong semantic connections, such as those human classes, are clustered together, while classes with weak connections, such as the plant classes and the animal classes, are far away from each other.

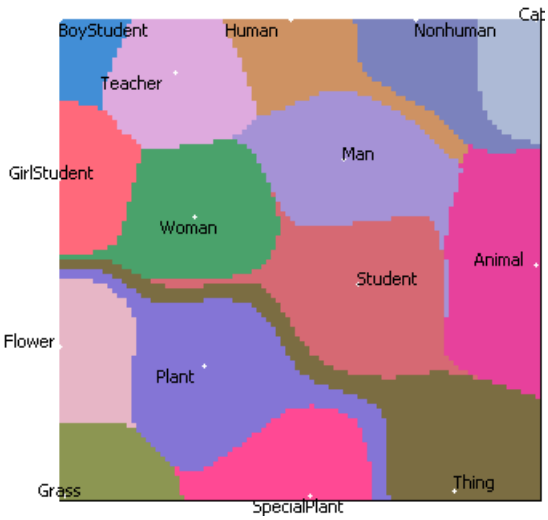
2.3 SOM

The layout produced by FDP reflects fairly well the semantic organization of the ontology. However, similar classes are often placed together in a small area, while there are often large areas with few classes in it. In most cases the class clusters are more important to users' understanding of the ontology, but the small space those clusters occupy make class labelling/annotation very inefficient, or even impossible. So we try to alleviate this problem by distorting the layout using Self-Organizing Maps (SOM) [9].

A SOM network is composed of $n \times n$ neurons, where n is determined by the number of classes, so that on average each class will own tens of SOM neurons. The neurons are first randomly placed in the 2D plane where the FDP layout is. In training, they will first be gradually organized to form a smooth network (the self-organizing phase), and then be converged to the positions where the classes are (the convergence phase).



(a)



(b)

Fig. 3. Sample ontology: the result of SOM

Figure 3(a) shows the resulting SOM network in the original 2D plane. By spreading the $n \times n$ network to form a grid, we get Figure 3(b), the resulting layout, where a class is represented by its nearest neuron. We can see in the resulting layout that classes in clusters tend to occupy much larger space than before, thus could be labelled clearly.

Notice that for each neuron, we can assign it to its nearest class. In this way each class would own a set of neurons. In other words, in the resulting layout each class would occupy an area, in addition to a point, as shown in Figure 3(b). The semantic organization of the ontology is now exactly visualized by *the neighborhood of these areas*: two neighboring areas implies a strong semantic connection between the two corresponding classes. For example, in Figure 3(b), Woman and Man are neighboring because they have a strong semantic connection, while Woman and Plant are separated by two other classes, which indicates that their semantic connection is not so strong. One may also find out from the figure that sometimes this layout can not perfectly visualize the semantic connections: some classes with strong connections are not neighboring. This is inevitable, however, as we have to display an ontology in a two-dimensional space.

There is another merit of representing a class by an area in addition to a point: classes can be colored much more clearly. This facilitates our instance visualization, as in section 2.5.

2.4 Labelling

Although by applying SOM classes tend to occupy much larger space, for large-scale ontolgies it is still impossible to label all classes when the layout is displayed in a screen with a limited size. So we have to assess the importance of each class, and when displaying (a part of) the layout, we label only the most important classes that fall into the screen. Currently we simply compute the importance based on the class hierarchy.

$$Importance(c) = {\gamma_1}^{depth(c)} + \gamma_2 \sum_{c_i \in DirectChildren(c)} Importance(c_i)$$

where γ_1 and γ_2 are two customizable constants in $[0, 1]$ (their default values are 0.5 and 1 respectively in our system), $depth(c)$ is the depth of c in the class hierarchy, and $DirectChildren(c)$ is the set of the direct children of c in the hierarchy. The first part of the formula gives more importance to the classes higher in the hierarchy, while the second part gives more to the classes with more descendants.

2.5 Visualizing Instance

In our approach we visualize the distribution of the ontology instances and instance relations.

For ontology instances, we use different colors to represent different instance numbers of classes. Since a class could be represented by an area, we just fill the area with the corresponding color. Two modes are provided to count the instance number of a class. The first mode only takes into account those instances whose types are explicitly declared as the class, while the second also includes those that are declared as the instances of the sub-classes of that class. Figure 4 is the resulting imaging.

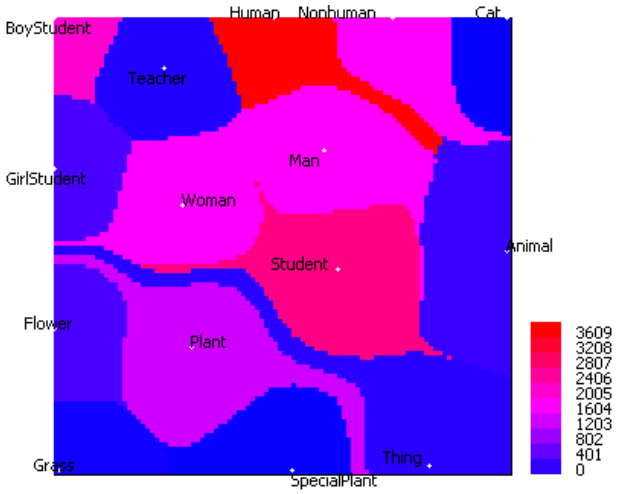


Fig. 4. Sample ontology: the instance distribution.

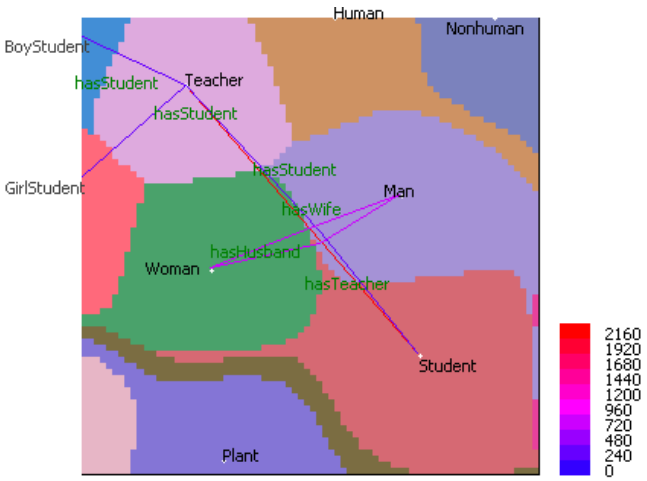


Fig. 5. Sample ontology: the instance relations (zoomed-in vision).

For instance relations, we draw a line between two classes to represent a relation between the instances of the two classes. Like in visualizing the instance distribution, we use the line color to indicate the number of instance pairs that have the relation. Notice that here we use the point representation of classes. Also there are two modes to count the number, based on whether the instances of a class's sub-classes should be included into the instances of that class. To prevent showing too many lines simultaneously, we select only those with the largest numbers of instance pairs. Figure 5 shows the sample visualization of the instance relations.

3 User Interface

Now we will review the user interface of our visualization approach under different tasks, i.e. ontology navigation, ontology retrieval and ontology instance analysis.

3.1 Ontology Navigation

Ontology navigation, i.e. navigating the content of an ontology, is the most basic task for an ontology visualization tool. For large-scale ontologies, traditional visualization approaches become inefficient, as discussed in section 1. Unlike those approaches, we present to users a holistic imaging of the ontology. In the imaging, the placement of classes are semantically arranged according to the measurements discussed in section 2.1, so that neighboring classes typically have stronger semantic connections. Such placement may facilitate user understanding of the ontology.

Since the ontology being visualized may be very large, we present the imaging with multiple *levels-of-detail*. In other words, we first present the whole imaging at low resolution and with only limited information directly labelled, but upon users' request partial imagings with more details can be presented, and the more local an partial imaging is, the more detailed information it will present. In this way, at first users could see several most important classes of the ontology marked in the imaging, thus roughly understanding the subject of the ontology. If one is interested in one of these major classes, by clicking that class or a nearby area he will get a partial imaging presenting the secondly important classes that are related to the selected major class, thus he could roughly understand the related part of the ontology. This procedure could go on until the most detailed information is presented.

Figure 6 gives an example of this procedure. The ontology visualized is the Semantic Web Technology Evaluation Ontology (SWETO)[1], which contains more than one hundred classes.

When a user is viewing a partial imaging, he could choose to switch to another partial imaging near the current one. This is useful because that, as the imaging is semantically organized, based on the current partial imaging the user may guess the position of a class and wish to seek it in a nearby area. Besides, the user could also choose to go back to a higher level-of-detail (i.e. with lower resolution).

We also provide a thumbnail of the imaging to facilitate the navigation. The area that is currently displayed is highlighted in the thumbnail, so as to help the user locate himself. Another function of thumbnail is that, by clicking on the thumbnail the user could switch between different areas conveniently.

Notice that our visualization approach is designed to complement other approaches instead of replacing them, so we integrate our visualization tool into an ontology engineering environment, i.e. ORIENT [10], which provides most kinds

[1] Available at http://lsdis.cs.uga.edu/Projects/SemDis/sweto/

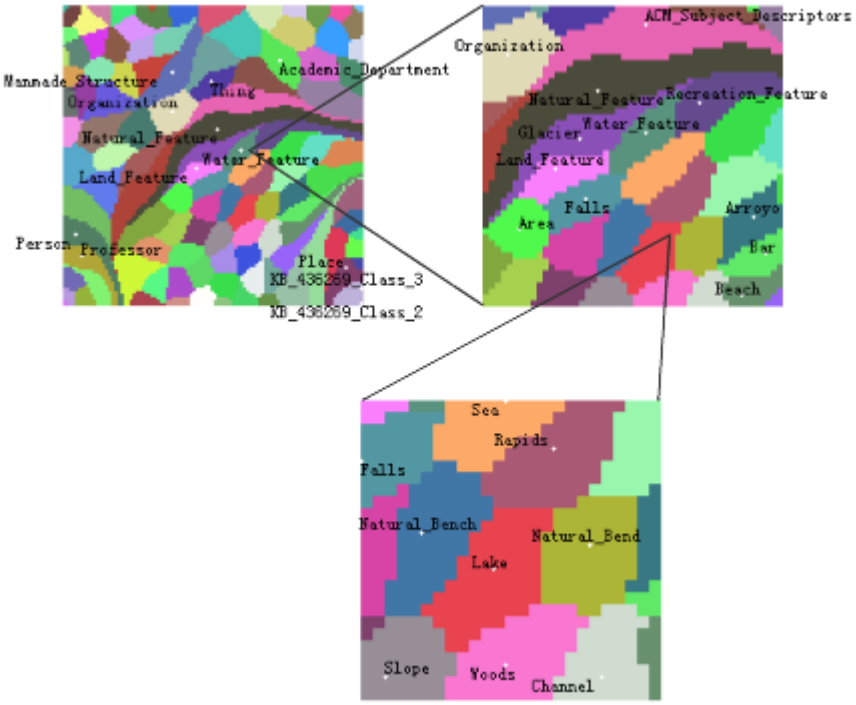


Fig. 6. Multiple levels of detail.

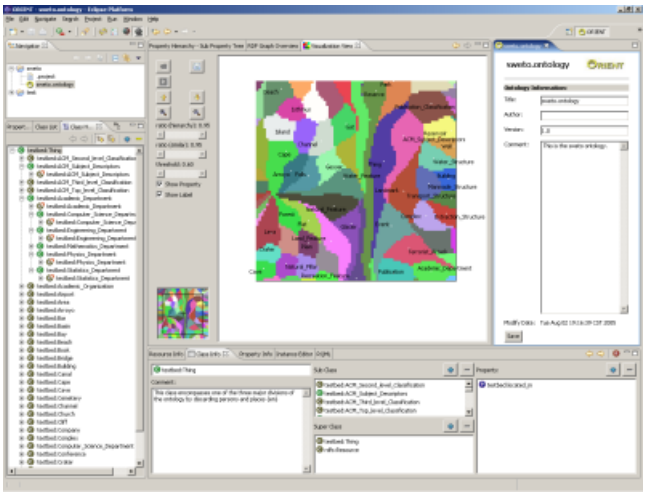


Fig. 7. Integrated into an ontology engineering environment

of traditional visualization tools like the form view, tree view and graph view (Figure 7). Such integration doubtless maximizes the overall navigation utility. For example, one could first get an overall understanding of the ontology from our visualization approach, and then scrutinize the formal details of several interesting classes using the traditional views.

3.2 Ontology Retrieval

In some applications, such as semantic annotation, users need to find from an ontology a class or classes that meet certain conditions. For large-scale ontologies, however, the searching process may be quite laborious and time-consuming. So using ontology visualization to facilitate the searching process is advisable and useful.

Our visualization approach is intrinsically applicable to searching. First, the classes are semantically organized in the resulting imaging, so finding a certain class in it would be easier than in a mere list. Second, the *level-of-detail* technique introduced in the previous section enables users to gradually narrow the searching scope and finally find the target class in the finest level-of-detail. Figure 6 is exactly an example procedure of searching for the water-related classes in the SWETO ontology.

Notice that our visualization tool is integrated with a set of traditional ontology engineering tools. Using them combinatorially could further facilitate the searching process. For example, one may have already found a class by the traditional tools and hence have gotten the class's position in our imaging, so he could then find the related classes just by looking into the nearby area of the position. It is also possible for users to get a set of class candidates by traditional tools (e.g. an RDF query answerer), then he could map these classes to the imaging and find the desired one based on their positions.

3.3 Instance Analysis

Facing an ontology with large-scale instances, people are usually at a loss in the "ocean" of information it provides. However, by "imaging" the distribution of the ontology instances and instance relations using our visualization approach, we are able to make quick analysis, at the instance level, about which topics in the ontology are "hot", and which classes are more actively inter-connected and thus strongly associated.

Take for example the imaging of a university ontology shown in Figure 8. In the areas depicting professors (the upper part), Naval_Architecture_and_Ocean_Engineering_Professor, Mechanical_and_Power_Engineering_Professor, Electronics_and_Electric_Engineering_Professor, and Material_Science_and_Engineering_Professor are assigned hotter colors than other professor classes, which means they own more instances. Analogous situation happens to the student classes (the lower part), as the classes such as Naval_Architecture_and_Ocean_Engineering_Student, Mechanical_and_Power_Engineering_Student, Electronics_and_Electric_Engineering_Student, and Material_Science_and_Engineering_Student are hotter than the others. So users

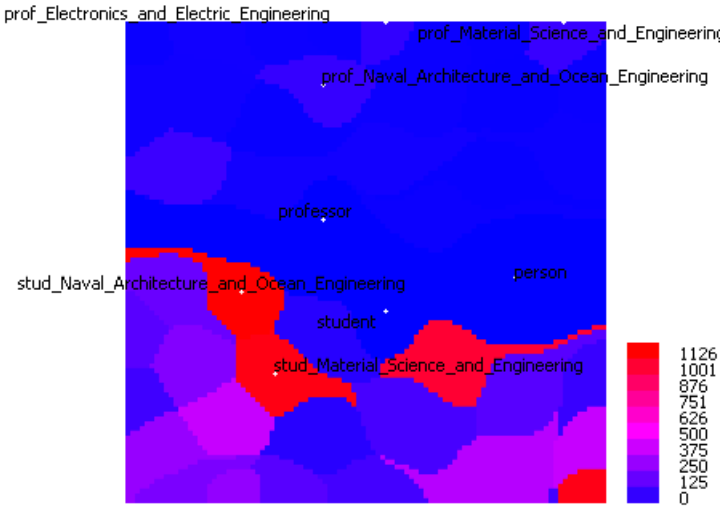


Fig. 8. Instance analysis for a university ontology.

could quickly infer from the above facts that these engineering-related schools are of more importance in that university.

Inter-relationship among different classes is another important and interesting factor that can be easily investigated using our visualization approach. For instance, in the imaging of an online store's knowledge base of its customers' purchasing records, by observing the Middle-Aged class one can find that, the two lines representing the Purchased property connecting the class respectively to the Classical_Literature class and to the Traditional_Opera class are more prominent (which means more property instances within) than those Purchased lines connecting Middle-Aged to Pop_Music and to Animation. For the Youth class, the situation is just the opposite. Therefore the store holder can learn from the imaging how to associate and recommend proper categories of commodities to the customers of different age. Similarly, other shopping habits among buyers of different gender, occupation, region, etc., can also be analyzed in our ontology visualization.

The level-of-detail, searching and overview functions provided by our visualization approach can be used to further enhance the instance analysis process.

4 Related Work

A number of previous work have contributed to the ontology visualization field. OntoViz[2] targets at the precise visualization of the ontology structure, using rectangles for classes/instances and lines for relations, while exploring ontology in OntoViz is somewhat inconvenient. TGVizTab[3] also tries to draw a precise graph of the ontology with concepts as nodes and relations as edges, and

it employs a spring-embedding algorithm to implement a customizable layout. Jambalaya[11] displays information in a similar way, but adds in a level-of-detail feature, allowing users to browse the ontology detail at several levels. In a word, all these methods try to draw a precise graph of the ontology, so they have a common drawback that if the ontology is very large, the visualization result will become unreadable. In contrast, our approach presents a holistic view which is applicable to large ontologies.

Currently most ontology visualization methods do not well support the analysis of large-scale instances. Our approach, however, manages to visualize the distribution of ontology instances and instance relations. There are also some other ontology visualization methods that aim to integrate instance information into the visualization. The Spectacle system[12] is designed to present a large number of instances with a simple ontology to users. Each instance is placed into a cluster based on its class membership. Instances which are members of multiple classes are placed in overlapping clusters. This visualization provides a clear and intuitive depiction of the relationships between instances and classes. Our approach can not visualize the instance overlap like Spectacle, which is the cost of our choosing to present a holistic view of large-scale ontologies.

The visualization techniques and algorithms that we use in our approach are also widely employed in other knowledge visualization fields. Infosky[13] is a system that tries to enable the exploration of large and hierarchically structured knowledge spaces, and it presents a two-dimensional graphical representation with variable magnification, much like providing users a real-world telescope. One of the key algorithms employed in InfoSky is a spring-embedding algorithm, which is used to cluster documents or document collections. WEBSOM[14] is developed for visualizing document clusters based on the Self-Organizing Map (SOM). Similar documents become close to each other on the map, like the books on the shelves of a well-organized library. Some other systems, like Themescape by Cartia Inc. and ET-Map[15], also visualize large numbers of documents or websites using the SOM algorithm.

5 Conclusion and Future Work

In this paper we present a novel ontology visualization approach that aims to facilitate user understanding and analysis of large-scale ontologies. Specifically, our approach produces a holistic imaging of the ontology that contains a semantic layout of the ontology classes, with the distribution of instances and instance relations also visualized. We scrutinize the resulting user interface and find it facilitates tasks like ontology navigation, ontology retrieval and ontology instance analysis.

As the next step, we are planning to conduct a comprehensive user-based study, so as to better evaluate our approach. In addition, while currently our system can visualize an ontology with hundreds of classes in minutes, we will try to further speed up our approach by using more optimization techniques, especially for the FDP and SOM algorithms, which are the main source of the

time complexity of our approach. We also plan to extend our approach to visualize ontology evolution. Since both the FDP and the SOM algorithm could be adapted to run incrementally, it is possible to produce two comparable imagings of the ontology before and after changing, thus highlighting the ontology changes.

References

1. Available at http://www.mindswap.org/2003/CancerOntology/.
2. Sintek, E.: OntoViz Tab: Visualizing protege ontologies. Available at http://protege.stanford.edu/plugins/ontoviz/ontoviz.html (2003)
3. Alani, H.: TGVizTab: An ontology visualization extension for protege. In: in Knowledge Capture 03 - Workshop on Visualizing Information in Knowledge Engineering, Sanibel Island, FL (2003)
4. Wu, Z., Palmer, M.: Verbs semantics and lexical selection. In: Proceedings of the 32nd conference on Association for Computational Linguistics, Morristown, NJ, USA, Association for Computational Linguistics (1994) 133–138
5. Rodríguez, M.A., Egenhofer, M.J., Rugg, R.D.: Assessing semantic similarities among geospatial feature class definitions. In: Interoperating Geographic Information Systems (Interop'99). LNCS 1580, Springer-Verlag (1999) 189–202
6. Kruskal, J.B., Wish, M.: Multidimensional Scaling. Beverly Hills and London: Sage Publications (1984)
7. Fruchterman, T.M.J., Reingold, E.M.: Graph drawing by force-directed placement. Software - Practice and Experience **21** (1991) 1129–1164
8. Mutton, P., Rodgers, P.: Spring embedder preprocessing for www visualization. In: Proceedings of Information Visualisation 2002 (IV02). (2002)
9. Kohonen, T.: Self-Organizing Maps. Springer (1995)
10. Zhang, L., Yu, Y., Lu, J., Lin, C., Tu, K., Guo, M., Zhang, Z., Xie, G., Su, Z., Pan, Y.: ORIENT: Integrate ontology engineering into industry tooling environment. In: Proceedings of the 3rd International Semantic Web Conference (ISWC2004). (2004)
11. Storey, M.A.D., Noy, N.F., Musen, M.A., Best, C., Fergerson, R.W., Ernst, N.: Jambalaya: an interactive environment for exploring ontologies. In: International Conference on Intelligent User Interfaces (IUI). (2002) 239–239
12. Fluit, C., Sabou, M., van Harmelen, F.: Ontology-based information visualization. In: Proceedings of Information Visualization '02. (2002)
13. Andrews, K., Kienreich, W., Sabol, V., Becker, J., Droschl, G., Kappe, F., Granitzer, M., Auer, P., Tochtermann, K.: The infosky visual explorer: exploiting hierarchical structure and document similarities. Information Visualization **1** (2002) 166–181
14. Kaski, S., Honkela, T., Lagus, K., Kohonen, T.: Websom - self-organizing maps of document collections. Neurocomputing **21** (1998) 101–117
15. Chen, H.C., Schuffels, C., Orwig, R.: Internet categorization and search: A self-organizing approach. Journal of Visual Communication and Image Representation **7** (1996) 88–102