Jambalaya Express:
Ontology Visualization-On-Demand

Robert Lintern
rlintern@uvic.ca

Margaret-Anne Storey
mstorey@uvic.ca

Chisel Group
University of Victoria,
Victoria, BC, Canada

Abstract

In this talk we explore the use of visualization as a cognitive aid for managing ontologies and knowledge representations. We focus on our most recent work to provide visualization "on demand" for supporting specific editing, maintenance and understanding tasks. This contrasts with our previous approach of providing overviews of ontologies for supporting exploration of unfamiliar ontologies. We have made efforts to not only reduce the amount of visual information shown in Jambalaya's views, but also to help ensure that this information is more focused on a user's current task or concept of interest.

1. Introduction

Over the past several years the Chisel group at the University of Victoria (www.thechiselgroup.org) has been working in the area of ontology and knowledge-base (KB) visualization. Our main KB visualization tool, Jambalaya, is an extension of the SHriMP graph visualization toolkit (www.thechiselgroup.org/shrimp) reconfigured as a plug-in for Protégé with additional features to help increase understanding of both regular Protégé projects and now RDF/OWL projects.

In short, Jambalaya's unique visualization approach includes nesting of nodes based on any slot or property types (or no nesting), smoothly animated zooming and layouts, and the embedding of Protégé's class and instance/individual editors within the visualization itself. Jambalaya also includes many of the base features found in other graph or KB visualizations, such as filtering, colouring and "styling" nodes and arcs by type, and various graph layout algorithms.

While we are still concerned with the *why* and *what* of visualizing ontologies, we have made an effort recently to tackle some of the *where*, *when* and *how* of visualization. **Where** should the visualization be present in the user interface? **When** would the user want to see a Jambalaya-like visualization? **How** can we ensure that a useful visualization is easily accessible and repeatable?

We do not presume to have solved these fundamental "visualization" issues, but have hopefully made some changes to Jambalaya in the last year that lead us in the right direction.

2. Scalability Challenges

Visualization-on-demand requires that Jambalaya remain nimble; this requires some effort in tackling scalability issues.

The scalability challenges that Jambalaya has been faced with are not unlike those faced by many other applications including the base Protégé system itself. Large ontologies, especially those stored in remote databases, bring these scalability issues to the forefront. In particular, we have been working with the NCI Thesaurus (ncicb.nci.nih.gov/core/EVS), a large OWL database project stored on a remote server with over eighty thousand classes, restrictions, and annotations; it has been an excellent test-bed for bottlenecks in our code. Examples in this presentation will be taken from the NCI Thesaurus.

Fortunately we already have some techniques in place to handle scalability issues. The general architecture of SHriMP, Jambalaya's underlying visualization engine, has been planned from the beginning to load data incrementally from large data sources, strictly on an as-needed basis. It has also been designed from the start to load data from a variety of sources with no assumptions made on the speed at which data can be gathered. We have endeavoured to keep this in mind while working in the ontology domain, and provide confirmation dialogs to the user whenever this constraint is likely to be violated. For example, the user is warned before opening Jambalaya's Attribute Panel as it requires all concepts to be loaded into memory. We have also attempted to show progress for slow operations, and allow the user to cancel them part way through.

The approach of nesting child nodes within their parents affords incremental loading since information on child nodes is obtained only when a parent is opened to show its children.

We've made attempts to reduce not only what is brought into Jambalaya's display, but also to reduce the amount of data coming into Jambalaya's own data buffers by implementing a layer of "data" filters.

3. Working Set, Express Views, and Drag-n-Drop

Recently, we have introduced the concept of "root classes" to Jambalaya. This allows the user to essentially choose a smaller working set and reduce the scope of data shown in Jambalaya. For large ontologies, a user may be only interested in visualizing the classes and restrictions defined within and between a few subtrees of the ontology. For example, in Figure 1 the scope of the classes shown in Jambalaya has been restricted to the *Perhipheral_Nervous_System* concept and the *Peripheral_Nervous_System_Part* subtree.

In order to reduce the number of steps a user must perform to elicit a useful view, Jambalaya's main view now provides some "express view" buttons to facilitate easy switching between different views of a project. (See fig. 1) For example, in the OWL domain, the user can click the "Domain-Range" express view button to quickly see how the OWL classes are related by the domains and ranges of the project's properties. Another express view shows the *is-a* class hierarchy in a vertical tree.

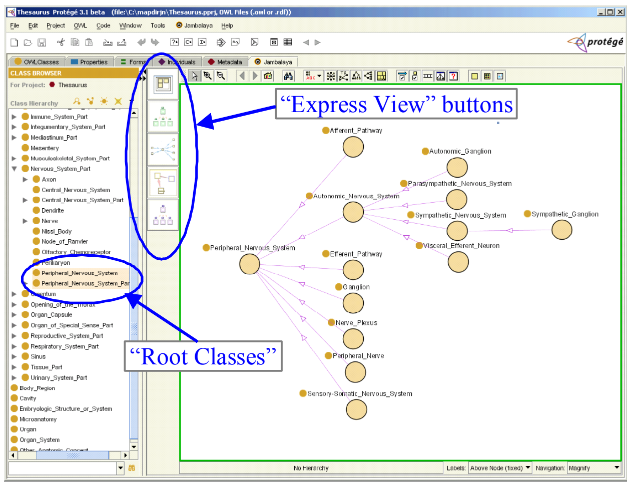


Figure 1. The Jambalaya Tab displays a portion of the "has part" hierarchy of the NCI Thesaurus.

In addition to providing some default express views, we will be adding the facility for users to define their own express views by allowing them to create new buttons and attach their own scripts – currently supported by Jambalaya – to these new buttons. For example, in Figure 1, one of the express view buttons is specific to the NCI Thesaurus and lays out the concepts in a horizontal tree according to the "has part" hierarchy.

The choosing of root classes and the launching of express views have been simplified by adding drag-and-drop functionality to Jambalaya. The user can simply drag classes from the Classes tree (a familiar view from Protégé), on the left of the Jambalaya tab, and drop them into Jambalaya's main view to make these the root classes of the main view. The user can also drag and drop these classes onto the express view buttons themselves to change the root classes and immediately invoke a express view on this new "working set."

4. Query View

In an attempt to make a departure from the Jambalaya's heavy-weight top-down visualizations we have implemented a simpler graphical view that can be quickly popped-up from anywhere in Protégé's GUI (see fig 2). This view takes a different approach than Jambalaya's main view to visualizing data in that it displays only classes of interest to the user and their "neighbours" (i.e. classes related to these classes of interest).

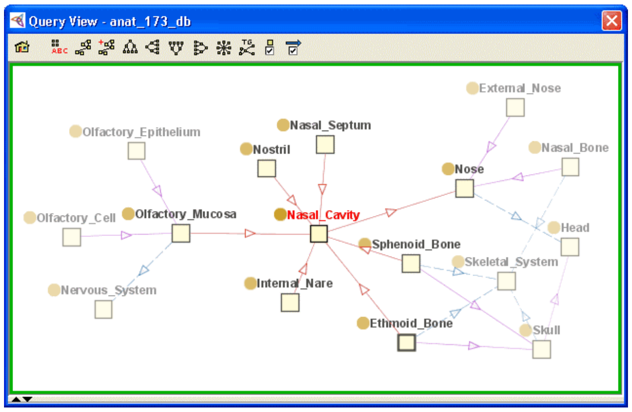
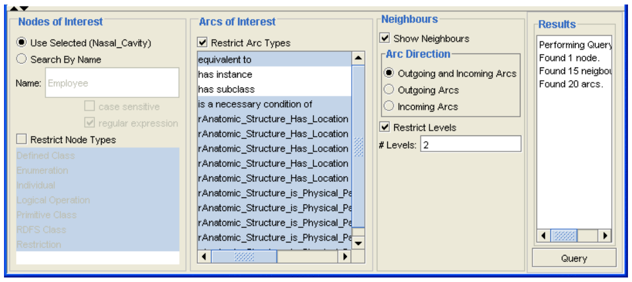


Figure 2. Jambalaya's Query View displaying the *Nasal_Cavity* concept and its "neighbours."



Nodes of Interest
(●) Use Selected (Nasal_Cavity)
( ) Search By Name
Name: Employee
[ ] case sensitive
[✓] regular expression
[ ] Restrict Node Types
Defined Class
Enumeration
Individual
Logical Operation
Primitive Class
RDFS Class
Restriction

Arcs of Interest
[✓] Restrict Arc Types
equivalent to
has instance
has subclass
is a necessary condition of
rAnatomic_Structure_Has_Location
rAnatomic_Structure_Has_Location
rAnatomic_Structure_Has_Location
rAnatomic_Structure_Has_Location
rAnatomic_Structure_Has_Location
rAnatomic_Structure_is_Physical_Pa
rAnatomic_Structure_is_Physical_Pa
rAnatomic_Structure_is_Physical_Pa
rAnatomic_Structure_is_Physical_Pa

Neighbours
[✓] Show Neighbours
Arc Direction
(●) Outgoing and Incoming Arcs
( ) Outgoing Arcs
( ) Incoming Arcs
[✓] Restrict Levels
Levels: 2

Results
Performing Query
Found 1 node.
Found 15 neigbou
Found 20 arcs.
Query

Figure 3. A GUI is provided to change the parameters of the query used to populate the Query View.

We call this new view the "Query View" since it provides a means for the user to specify various parameters for determining classes of interest - for example, searching class names using regular expressions - and parameters for specifying how much of their neighbourhood to show. (see Fig. 3) The user can restrict the query to certain node and arc types, whether or not to use incoming arcs, outgoing arcs, or both, and how many levels of neighbours (i.e. the neighbours of neighbours etc.) to show.

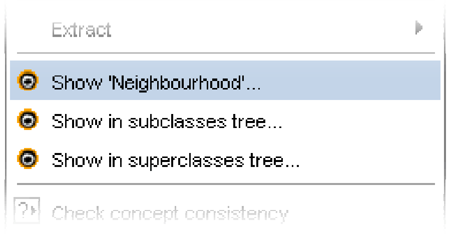

Extract
Show 'Neighbourhood'...
Show in subclasses tree...
Show in superclasses tree...
Check concept consistency

Figure 4. Jambalaya's Query View can be popped-up from a concept's right-click menu.

In order to make the Query View more accessible it can be popped-up from a class' right-click menu (see fig 4) and the parameters of the Query View are set automatically to show how this selected class relates to the rest of the ontology. For example, from the NCI Thesaurus we can see all concepts related to the *Nasal_Cavity* concept, such as *Nose* and *Nasal_Septum*, by invoking the "Show Neighbourhood" action on *Nasal_Cavity* (see fig. 2 and 4).

5. Ongoing and Future Work

We have always been interested in improving the configurability of Jambalaya since it is difficult for us to predict what would be considered a "useful" view for all ontologies. Different domains, and the ontologies that describe them, have different classes and slots (or properties) of interest. For example, in the anatomy domain it is important to observe the "has part" hierarchy. Furthermore, ontologies can be designed in very different ways depending on their intended usage and can be described with many different "dialects," especially in Protégé with the use of meta-classes. Because of this variability we cannot always predict a correct configuration for Jambalaya. We envision in the future having a user-specified mapping from ontology objects and their attributes to Jambalaya's visual components and visual variables. For example, the user may want classes that inherit from a particular superclass to be shown in a certain colour.

We also envision support for certain user-defined "graph transformations." One simple example of a graph transformation is to change the way that a reified relation is displayed. In fact, Jambalaya already does this automatically for subclasses of *:DIRECTED-BINARY-RELATION*. Instead of showing the reified relation as a node with *:TO* and *:FROM* arcs, these three items are collapse into a single arc with a more descriptive type and tool-tip. More support is needed for users to set-up and configure these types of transformations themselves.

6. Discussion

At the end of our presentation, we will lead a discussion on how visualization can be helpful (or not) for representing and managing ontologies. In particular we will focus on the challenges of providing task-specific visualizations. The discussion may also delve into the pros and cons of visualizing OWL ontologies using a graph metaphor.

7. Acknowledgements

The Chisel group's visualization work has previously been supported by grants from the National Cancer Institute's Center for Bioinformatics.

8. References

1. Alani, H., TGVizTab: An Ontology Visualisation Extension for Protege. in *Knowledge Capture 03 - Workshop on Visualizing Information in Knowledge Engineering*, (Sanibel Island, FL, 2003), ACM, 2-7.
2. Ernst, N.A. and Storey, M.-A. A Preliminary Analysis of Visualization Requirements in Knowledge Engineering Tools, University of Victoria, Victoria, 2003, 6.
3. EZOWL Tab, The EZOWL website, Intelligent Web Technology Research Team, Electronics and Telecommunications Research Institute, iweb.etri.re.kr/ezowl/
4. Kremer, R., Visual Languages for Knowledge Representation. in *11th Workshop on Knowledge Acquisition, Modelling, and Management (KAW '98)*, (Banff, Alberta, 1998).
5. Piccolo, The Piccolo website, Human-Computer Interaction Lab, University of Maryland, www.cs.umd.edu/hcil/piccolo/
6. Protégé, The Protégé website, Stanford Medical Informatics, protege.stanford.edu
7. Sintek, M., OntoViz Tab: Visualizing Protege Ontologies,protege.stanford.edu/plugins/ontoviz/ontoviz.html
8. OWLViz Tab, Medical Informatics Group, University of Manchester, www.co-ode.org/downloads/owlviz/
9. Storey, M.-A.D., Musen, M.A., Silva, J., Best, C., Ernst, N., Fergerson, R. and Noy, N.F., Jambalaya: Interactive visualization to enhance ontology authoring and knowledge acquisition in Protege. *Workshop on Interactive Tools for Knowledge Capture, K-CAP-2001*, (Victoria, B.C. Canada, 2001).
10. Storey, M.-A.D., Wong, K., Fracchia, F. and Müller, H., On Integrating Visualization Techniques for Effective Software Exploration. in *InfoVis '97*, (Phoenix, AZ, 1997), 38-45.
11. TouchGraph, The TouchGraph website, www.touchgraph.com/
12. Treemap, The Treemap website, Human-Computer Interaction Laboratory, University of Maryland, www.cs.umd.edu/hcil/treemap/