Improving the experience of ontology design, management and enquiry with concept diagrams

Jim Burton
University of Brighton

***Abstract*—Ontology engineers use a variety of software tools to design, manage and interrogate their ontologies. These tools often include visualisation features which provide a graphical depiction of parts of the ontology. In this paper we describe a new tool, *ConceptViz*, which uses a novel notation called *concept diagrams*, based on Euler diagrams. Although work is at an early stage, ConceptViz is highly expressive, allowing information to be expressed in a single diagram that would normally need to be gathered from several parts of an ontology management tool. To further motivate the use of concept diagrams we highlight their balance of *iconic* and *symbolic* notation and show how symbolic features can be used to reduce clutter in diagrams.**

I. Introduction

Ontologies are used to represent knowledge bases in diverse fields such as the semantic web, bioinformatics and digital libraries. Each ontology consists of a hierarchical tree of concepts, relations between those concepts and individuals that inhabit the concepts. Since ontologies can become very large, visualising concepts and their interrelation is seen as essential and the field of ontology visualisation is an active one [11]. Existing visualisations lack several desirable features however, including support for reasoning about the ontology at a formal level. This paper describes work-in-progress on a novel visualisation technique and tool that makes use of the intuitive power and formal properties of the expressive Euler-based visual logic of *concept diagrams*. For details of concept diagrams, see Howse et al. [7]; in this paper we describe only those details of the syntax and semantics of the notation as are needed for our argument. Our aim is to justify the claim that concept diagrams provide a diagrammatic model for ontologies which is easy to understand and which exposes more information than existing techniques. In addition, the notation has the potential to bring the benefits of formal reasoning to a wide community of users.

The visualisation tool, called *ConceptViz*, is developed as a plugin for the ontology management tool Protégé [17]. Figure 1 gives a flavour of the notation. Amongst other information, it tells us that the concepts *InformationRealization* (*IR*) and *InformationObject* (*IO*) are subsumed by *InformationEntity* (*IE*), that *IR* and *IO* are disjoint (both facts coming from the placement of circles), and that every *IE* is either an *IR* or an *IO* (this fact coming from the shading). To produce this diagram, the plugin interacts with the Protege API to gather information from *asserted* and *inferred* hierarchies; the inferred hierarchy is provided by one of a range of theorem provers available with Protégé. Our tool provides a graphical, direct and explicit representation of information, to find which the user would otherwise have to interrogate several parts of the Protégé environment. In some cases, in fact, the user would need to make inferences of their own to arrive at the same information, since it does not appear explicitly outside of the diagram.

The secondary topic of this work is the problem of devising effective formal visual languages. In order to justify its existence, we believe that each such language should benefit the user by exploiting the particular cognitive properties of predominately iconic (as opposed to symbolic) languages. A solution to this problem has been the explicit goal of researchers in visual reasoning since the time of Peirce. A number of logicians and computer scientists have addressed the same problem since, such as Shin and Hammer [6], Gurr [13] and Shimojima [18]. Meanwhile, the same problem has been examined in depth by the cognitive science (e.g., [8]) and semiotic [1] communities. These studies have normally been carried out in a more ethnographic way (i.e. by studying found notations rather than devising new ones), and have not tended to focus on the domain of visual logic. This, and the fact that few cognitive scientists are also logicians and vice versa, has meant that the findings of each community are not always well-known or exploited enough outside of that community. Our analysis of the strengths and weaknesses of approaches to ontology visualisation, including our own, represents a attempt at reconciling these different bodies of knowledge.

II. Ontologies and visualisation

An ontology represents knowledge as a set of *concepts* (sets, classes) within a particular domain, along with *individuals* inhabiting the concepts and *roles* (relations) between concepts and/or individuals. Thus, an ontology is a "formal, explicit specification of a shared conceptualisation" [4]. We can also think of an ontology simply as a collection of axioms, or as a *taxonomy* plus some roles. The taxonomy distinguishes the *is-a* relation as central to understanding the structure of the ontology.

Ontologies have found many and diverse applications, some of the most well known being medicine and the semantic web. Figure 2 shows an informal visualisation of the semantic web ontology produced by the W3 standards organisation.

At the formal level, there are a number of standards for encoding ontologies, which tend to be XML-based (e.g. RDF, OWL2).

```
<Ontology ontologyIRI="http://example.com/tea.owl"
    ...>
   <Prefix name="owl" IRI="http://www.w3.org
       /2002/07/owl#"/>
```

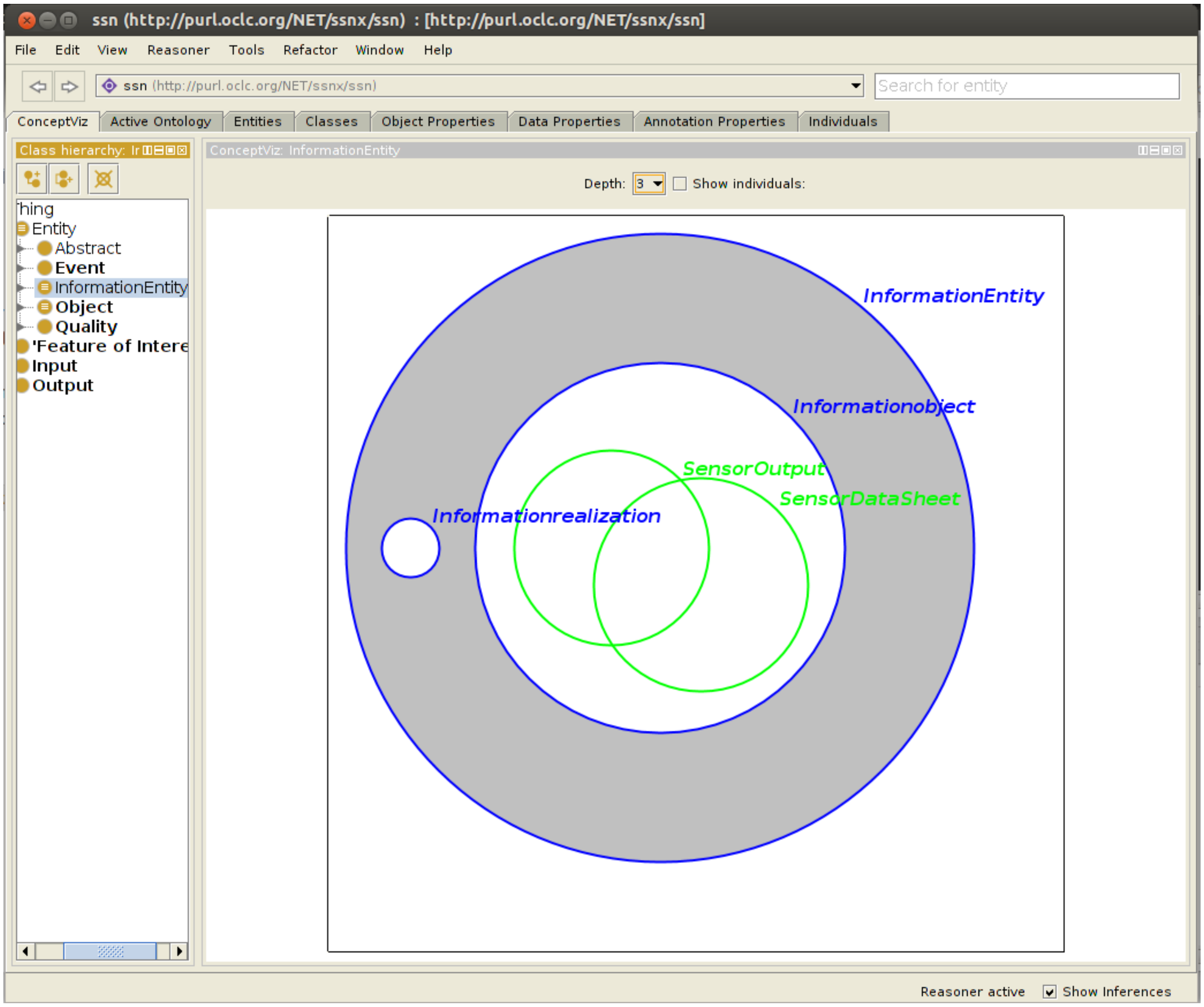

Fig. 1. The *ConceptViz* plugin

```
  <Declaration>
    <Class IRI="Tea"/>
  </Declaration>
</Ontology>
```

OWL semantics are based on *description logics* (DLs), decidable fragments of FOL with efficient SAT solvers. These have been incorporated into ontology management tools, allowing users to reason about their ontology.

There are a diverse range of tasks involved in the creation, use and maintenance of ontologies. These include design, debugging, comprehension/discovery (bottom-up and top-down), maintenance, query and reasoning. However, although visualisation tools have long been recognised as an essential part of ontology workflows, most visualisations focus almost entirely on comprehension.

Shneiderman [20] identified a number of high-level tasks a visualisation should support: overview, zoom, filter, details on demand (e.g. right click a concept to see properties, roles etc), and so on. There are a wide variety of ontology visualisations available, many described in a useful survey by Katifori [11], and ontology visualisation continues to be an active area, testifying to their importance to ontology users. Katifori categorises the visualisations according to their approach to information visualisation and user interaction. The most basic category is the *indented list* of the taxonomic structure, as seen in Protégé. This can be useful in getting a feel for the overall structure of an ontology but has several drawbacks. For instance, no roles or properties are shown, parts of the structure may be hidden, and it is not clear how multiple inheritance (concepts which appear in several parts of the taxonomy) should be handled.

The next most common representations are *tree-based*, such as *OntoGraf*, which is a Protégé plugin. OntoGraf provides a powerful and intuitive visualisation of the hierarchy, but one which makes poor use of space and can quickly become cluttered. OntoGraf can be used to visualise concepts, roles and individuals, making it more expressive than most other visualisations. It offers the user a great deal of flexibility about which parts of the ontology are displayed and which are hidden. This flexibility comes at a cost, however, since it makes it easy to create diagrams which may be misleading. OntoGraf users can reduce clutter in their diagrams by choosing not to show certain relations: in figure 3 the *is-a* relation between *Country* and *Thing* is not shown, which may surprise or confuse novice

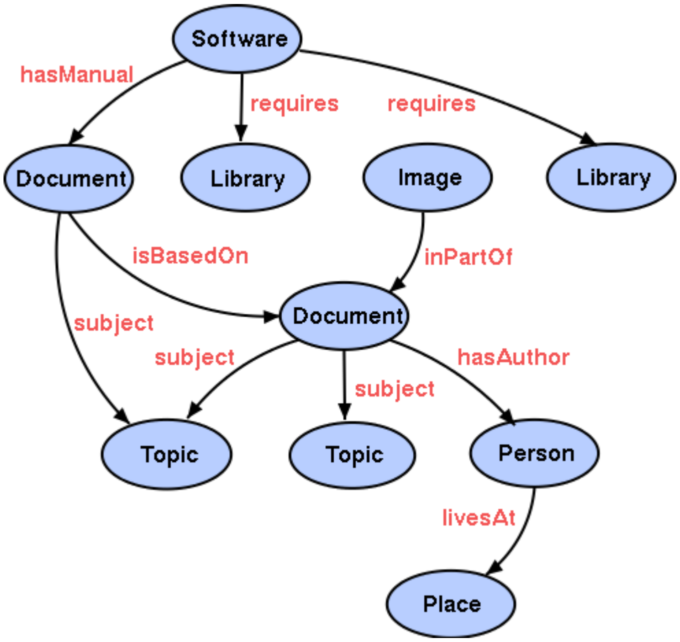

Fig. 2. An informal visualisation of an ontology fragment, © http://w3.org

users. In contrast, diagrams in ConceptViz are produced using an unambiguous visual logic (concept diagrams). Thus, the inferences a user may draw from a diagram are always valid, provided that they draw these inferences correctly.

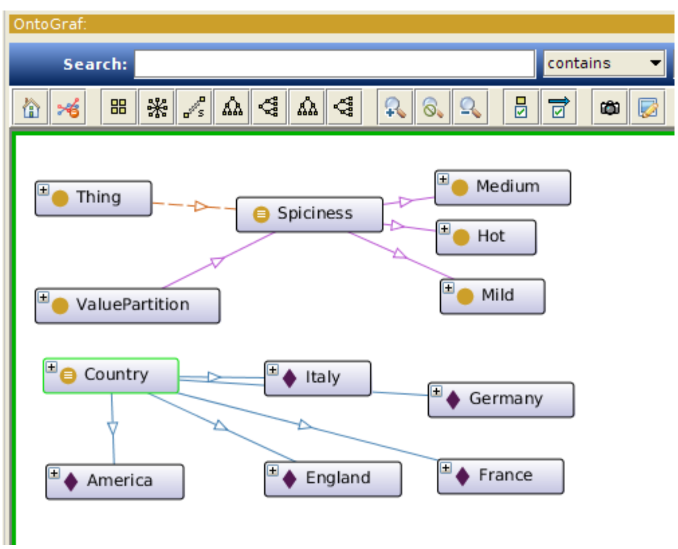

Fig. 3. A screenshot of the OntoGraf visualisation

One of the most important developments since Katifori's survey is the idea of *key concepts* [16]. Heuristics are applied to identify the concepts which are key to understanding an ontology. These include the number of sub-concepts, density of role-based information and coverage, so that sparse sections of the ontology are not ignored. Key concepts are used in KC-Viz [14], a plugin for the Swoon editor that uses key concepts to orient the user and convey a "gestalt" impression of the whole ontology. A recent tool that uses the key concepts idea in a more general setting is TriView [9]. Key concepts, which they call *landmarks*, are shown in an *overview window*. A second, larger window shows the local taxonomic sub-tree in detail, while a third shows roles and other axioms. The position of the local sub-tree, relative to the whole ontology, is indicated in the overview window. In this way, the user is helped to understand the overall structure of the ontology at the same time as focusing on a small area of it.

While all visualisations have strengths and weaknesses and are more suitable for some tasks than others, it seems fair to say that they all leave something to be desired. Key problems identified by Katifori include *clutter* and other *scalability* issues. There are also crucial features we may want which go beyond visualisation. *Reasoning* is not supported to any significant degree by any of the tools (e.g. OWLViz can indicate inconsistent concepts but that is about as far as it goes).

Also, *editing* is not supported: several tools allow the user to add a new concept, but then resort to forms etc to specify the details. Finally, *existing tools focus on taxonomy*: support for individuals is unusual, and support for roles and other axiom-based information more so.

III. The Affordances of Concept Diagrams

In the previous section we described some existing approaches to ontology visualisation and identified important gaps in functionality. *ConceptViz* is our work-in-progress ontology visualisation based on concept diagrams. In this section we aim to motivate the use of concept diagrams (CDs) as the basis of an alternative visualisation.

Our first observation is that CDs are more expressive than most existing visualisations. CDs are expressive enough to represent *entire* ontologies diagrammatically. Figure 4 shows a concept diagram with syntactic features that we intend to incorporate into ConceptViz. These features allow the representation of individuals (using solid dots), roles (using arrows) and quantification (using extra-diagrammatic symbolic expressions).

The second observation is that, alone among the visualisations we have considered, CDs are *formal* and come equipped with a framework for developing proofs and carrying out reasoning. This property enables a visualisation based on CDs to cover all stages of the ontology workflow described in the previous section, providing the potential for purely diagrammatic ontology management. This expressiveness and inferential capability doesn't imply, however, that CDs are fit for purpose. For that to be true, they must also be "easy" to understand relative to other approaches, for some measure of ease.

The question of the usability, or relative ease of comprehension, of a visualisation based on CDs can only be definitively answered by empirical studies. As well as "road testing" the usability of CDs, we can also attempt to show that their use is justified from a theoretical perspective. This requires an analysis of the *affordances* of CDs – the possible meanings of some piece of notation, and the ways in which an actor perceives and constructs that meaning. The starting point for

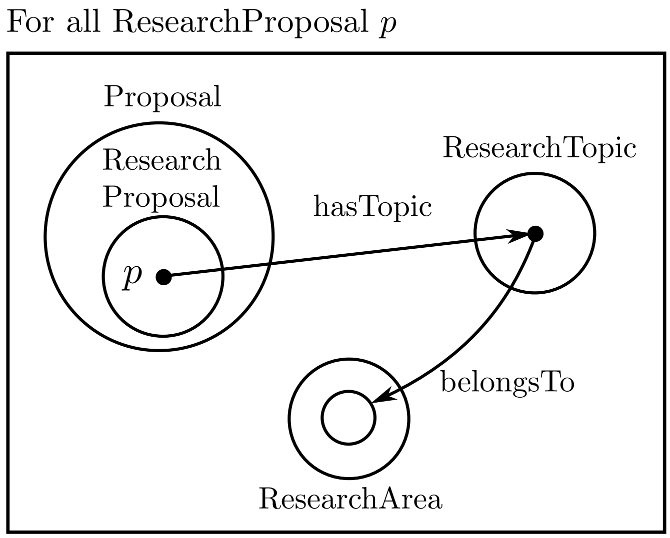

Fig. 4. A concept diagram

this analysis is the work of C.S. Peirce, who founded the field of semiotics. Peirce categorised the modes by which we construct the meaning of signs as *iconic*, *symbolic* and *indexical* [15]. Icons depict by resemblance, symbols by convention, and indices by "pointing" to their meaning. An important point to retain is that no sign is purely iconic, symbolic or indexical – each sign makes use of these modes to various degrees, and each mode relies on inferences gained using other modes. For Peirce, each mode is best suited to a different type of information: symbols are best suited to represent general rules, icons should be used to represent a particular hypothesis or "state of affairs", and indices are best used to represent the existence of an entity.

Although he stressed the interdependence of the modes, Peirce privileged the iconic mode as the most "natural" and in designing his systems of existential graphs, Peirce's aim was to create a system which was *as iconic as possible*. Iconicity is related to Gurr's notion of *well-matchedness* [5]. The Euler basis of CDs is *well-matched to meaning*, in that the syntax corresponds in a natural way to the semantics (e.g. topological separation means disjointness). Individuals are well-matched (relatively iconic) too. Placing a dot in a region asserts the existence of an individual in the concept modelled by that region and outside the others. Two separate dots assert the existence of two separate individuals.

Hammer and Shin [6] noted that the many of the additions made to Euler's original notation over the years do not always provoke the same natural associations in the reader. Shading, for instance, first introduced by Venn, bears no resemblance to the emptiness of a set and has a purely conventional meaning (apart from a slightly tenuous connection between shading, darkness and the emptiness of a void). Elsewhere Shin argues [19] that resemblance is not, in fact, inversely proportional to conventionality. Two cognitive properties of diagrams which are inversely proportionate to each other, however, are conventionality and the use of *perceptual inferences*. That is, the less a notation relies on convention, the more perceptual inferences are introduced. This seems convincing but it also seems likely that conventional symbols can be internalised, giving us indexical or immediate access to their meaning.

We can think of the number of perceptual inferences required by a given notation to convey a particular statement as its relative *efficiency* [1]. This efficiency is determined by the choices made by the designers of the notation; consider the observation made by Blackwell and Green, that "every notation highlights some kinds of information, at the cost of obscuring other kinds" [2]. For an analysis of the implications of these choices made by a comparison between spider diagrams and existential graphs, see Burton & Coppin's paper in *Euler Diagrams 2012* [3].

Furthermore, Shin points out that several of the conditions we might want to represent are incapable of depiction without convention, particularly disjunctive information and certain types of negative information. Thus, although advocates of Euler-based notations have typically argued for their effectiveness by an appeal to their well-matched features, an expressive notation needs to make use of non-iconic features. In order to understand the real nature of well-matchedness we need to focus on the correspondence between topological and semantic structure rather than any idea of physical resemblance. In recent work, Legg [12] makes the point that by doing so, we can see that the use of inference rules in any (visual) logic is iconic, since the application of the rule is defined via syntactic changes to the diagram which correspond to structural changes on the semantic level.

To expand on Peirce's dictum that symbolic notation should be used to represent "general rules", symbolic features have the capacity for a very compact expression, freed of the responsibility to depict structural correspondence. The designers of CDs add several symbolic features, neither of which adds to expressiveness, but which can both be used to reduce clutter in diagrams: *dashed arrows* and *nested bounding boxes* (also called nested universes). As well as inheriting the benefits of Euler diagrams, CDs inherit some limitations: an Euler diagram can quickly become cluttered [10]. In addition, we've seen that clutter is identified as a key usability problem in ontology visualisation. We explain the issues of clutter reduction in the following examples. The examples relate to communicative goals held by the diagram creators, in which the creators want to draw attention to a particular piece of information.

Each arrow in a concept diagram has a *source*, a *target* and a *label*. The arrow tells us that when the domain of the role represented by the label is restricted to the concept represented by the source, then the image of that role is the concept represented by the target. Informally, 5 tells us that members of the concept A are related to members of the unlabelled concept under f. The semantics of the arrow assert that elements of A are not related to anything else under f.

Assume that the creators of figure 5 want to focus on the presence of B in the target of the arrow. One way of drawing attention to B is to remove other concepts from the diagram, resulting in figure 6. This is a limited solution; the creators cannot now reintroduce C or D to the diagram without reintroducing clutter and losing the focus on B in the target of the arrow.

In contrast to arrows drawn with a solid line, dashed arrows provide partial information. The dashed arrow in figure 7 tells

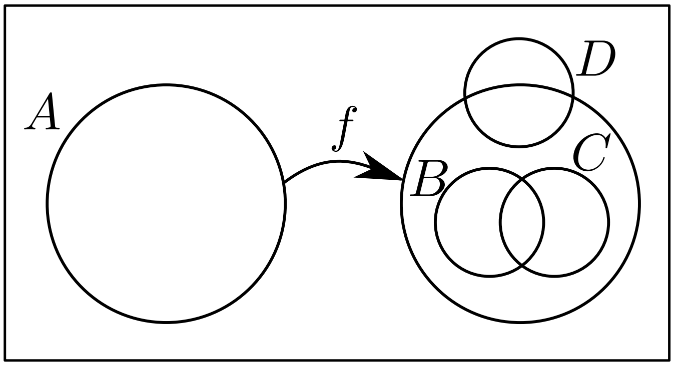

Fig. 5. Arrows in concept diagrams

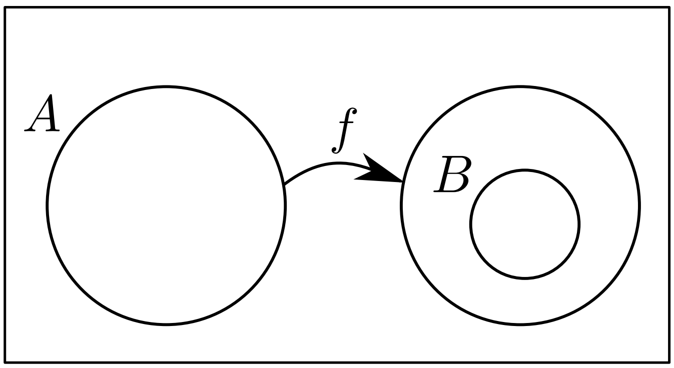

Fig. 6. Reducing the clutter in a concept diagram

us that elements of A are related to *at least* elements of B under f. The diagram creators could reintroduce C and D to figure 7 without needing to change the target of the arrow. Howse et al.[ISWC TUTORIAL] show several larger examples in which the "savings" (in terms of clutter reduction) are amplified further still.

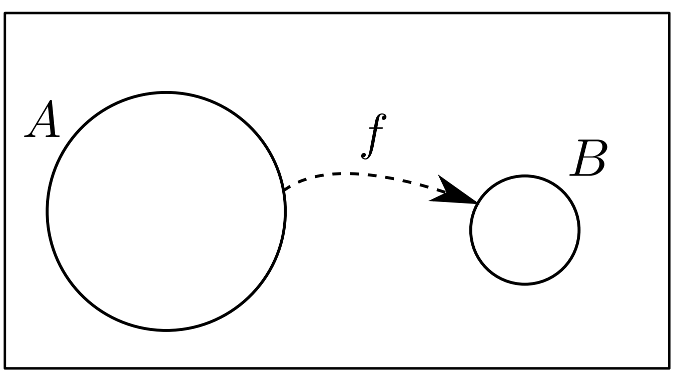

Fig. 7. Reducing the clutter in a concept diagram using dashed arrows

In our next example we consider nested bounding boxes. The bounding box of each concept diagram represents the concept *Thing* or, in set-theoretic terms, the universe of discourse. Diagrams may include several bounding boxes, each of which represents *Thing*. Suppose that we want to add a fourth concept, D, to figure 8, so that D is disjoint from B. Also, D is the source of an arrow which has C as its target. This arrow is the main focus of what the diagram creator wants to convey.

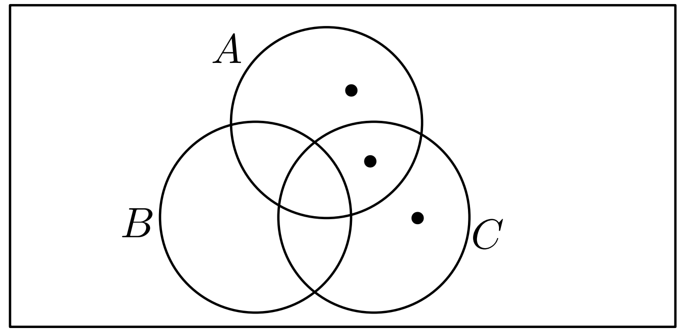

Fig. 8. A CD with three concepts and no disjointness information

The result of adding D to figure 8 in this way is shown in figure 9. We can see that the resulting diagram is rather cluttered. The diagram creators goal of highlighting the information provided by the arrow depends partly on the way we lay out the diagram, but is hard to achieve.

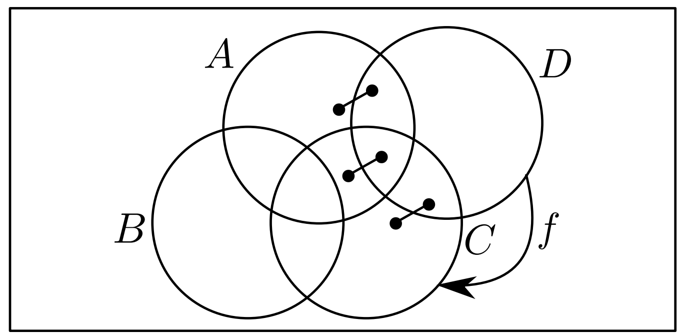

Fig. 9. A cluttered concept diagram

When a concept diagram includes several bounding boxes, the spatial relations between elements in separate boxes has no meaning. Thus, nested universes allow us to add D without specifying its disjointness or otherwise from A, B and C. This results in the diagram in figure 10, which is less cluttered than figure 9, making the communicative goal of focusing on the arrow easier to achieve.

Dashed arrows and nested bounding boxes can reduce clutter because they allow us to convey partial information in an unambiguous way. As stated above, OntoGraf users can reduce clutter by choosing not to show some relations in the diagram, but this may lead to results that are difficult to understand out of their original context.

IV. Conclusions and further work

We have seen that concept diagrams contain both iconic and symbolic features and noted that, historically, the main claim for the usability of Euler-based notations has been the iconic nature of the underlying Euler circles. However, symbolic features have great expressive potential and we have shown examples of symbolic syntax in concept diagrams that allows the compact representation of ontologies. Given that clutter is recognised as a common problem in ontology visualisation and that a medium-sized ontology may contain thousands of concepts, we see this as a strong point of concept diagrams. Although the current version of ConceptViz

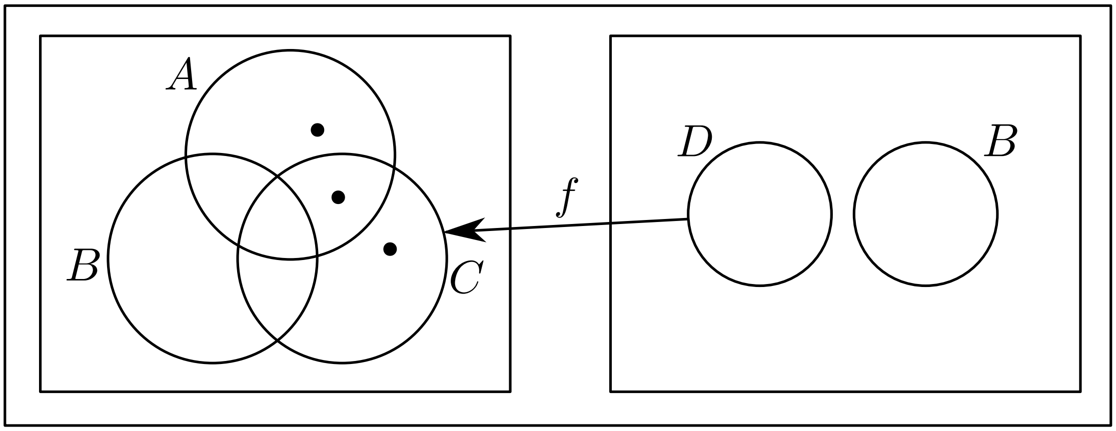

Fig. 10. The case for nested bounding boxes

contains only Euler diagrams, it provides an adequate basis for implementing more of the concept diagram notation.

We conclude by providing a few details of the implementation of ConceptViz and the goals for its future development. The plugin uses the *Inductive Circles* (iCircles) library by Jean Flower. iCircles uses ideas and algorithms developed by Stapleton, Flower, Rodgers and Howse (2012) [CITE] to draw Euler diagrams using circles. To ensure that the diagram can be drawn, it relaxes a topological property, normally desirable, which states that labels should be unique.

There is a growing number of ontology visualisation tools in existence, but none that support the entire ontology management workflow. A major goal for the plugin is to incorporate editing functionality, enabling ConceptViz to go beyond visualisation to ontology design and maintenance. Users will be able to generate and alter ontologies diagrammatically, observing the changes in other parts of the Protégé interface, and vice versa. This will result in "round-trip" diagrammatic ontology engineering. The interface for the tool will make a number of ontology design patterns available (such as applying a domain restriction to a role), giving users a toolkit for assembling ontologies from frequently used components.

One of the interesting problems is to ensure predictability when drawing the same or similar diagrams – when we reveal sub-concepts in one part of the diagram, we want the rest of the diagram to be as unchanged as possible. Unpredictability may cause serious usability problems (Herman, 1998) [CITE]. The layouts of diagrams produced by iCircles are currently rather unpredictable, since layout choices are based on heuristics. Work is underway to abstract these choices into reusable layout policies.

References

[1] Jacques Bertin. *Semiology of Graphics: Diagrams, Networks, Maps*. University of Wisconsin Press.

[2] Alan Blackwell and Thomas R. Green. Notational systems – the cognitive dimensions of notations framework. In John M. Carroll, editor, *HCI Models, Theories, and Frameworks: Toward a Multidisciplinary Science*, Interactive Technologies, chapter 5, pages 103+. Morgan Kaufmann, San Francisco, CA, USA, 2003.

[3] J. Burton and P. Coppin. Understanding and predicting the affordances of visual logics. In *submitted to 3rd International Workshop on Euler Diagrams*, 2012.

[4] Thomas R. Gruber. A translation approach to portable ontology specifications. *KNOWLEDGE ACQUISITION*, 5:199–220, 1993.

[5] C. Gurr and K. Tourlas. Towards the principled design of software engineering diagrams. In *Proceedings of 22nd International Conference on Software Engineering*, pages 509–518. ACM Press, 2000.

[6] E. Hammer and S. J. Shin. Euler's visual logic. *History and Philosophy of Logic*, pages 1–29, 1998.

[7] John Howse, Gem Stapleton, Kerry Taylor, and Peter Chapman. Visualizing ontologies: A case study. In Lora Aroyo, Chris Welty, Harith Alani, Jamie Taylor, Abraham Bernstein, Lalana Kagal, Natasha Noy, and Eva Blomqvist, editors, *The Semantic Web ISWC 2011*, volume 7031 of *Lecture Notes in Computer Science*, pages 257–272. Springer Berlin Heidelberg, 2011.

[8] James R. Hurford. The neural basis of Predicate-Argument structure. *Behavioral and Brain Sciences*, 23(6), 2003.

[9] Zong L. Jiao, Qiang Liu, Yuan-Fang Li, Kim Marriott, and Michael Wybrow. Visualization of large ontologies with landmarks. In Sabine Coquillart, Carlos Andújar, Robert S. Laramee, Andreas Kerren, and José Braz, editors, *GRAPP/IVAPP*, pages 461–470. SciTePress, 2013.

[10] C. John. Measuring and reducing clutter in euler diagrams. In *Proceedings of the First International Workshop on Euler Diagrams, Euler 2004*, volume 134 of *ENTCS*, pages 103–126. Elsevier, 2005.

[11] Akrivi Katifori, Constantin Halatsis, George Lepouras, Costas Vassilakis, and Eugenia Giannopoulou. Ontology visualization methods a survey. *ACM Comput. Surv.*, 39(4):10+, November 2007.

[12] Catherine Legg. What is a logical diagram? In Amirouche Moktefi and Sun-Joo Shin, editors, *Visual Reasoning with Diagrams*, Studies in Universal Logic, pages 1–18. Springer Basel, 2013.

[13] D. L. Moody. The "physics" of notations: Toward a scientific basis for constructing visual notations in software engineering. *Software Engineering, IEEE Transactions on*, 35(6):756–779, November 2009.

[14] Enrico Motta, Silvio Peroni, JoséManuel Gómez-Pérez, Mathieu d'Aquin, and Ning Li. Visualizing and navigating ontologies with KC-viz. In Mari C. Suárez-Figueroa, Asunción Gómez-Pérez, Enrico Motta, and Aldo Gangemi, editors, *Ontology Engineering in a Networked World*, pages 343–362. Springer Berlin Heidelberg, 2012.

[15] C. S. Peirce. *Collected Papers*, volume 4. Harvard University Press, 1933.

[16] S. Peroni, E. Motta, and M. d'Aquin. Identifying key concepts in an ontology, through the integration of cognitive principles with statistical and topological measures. In *The Semantic Web*, pages 242–256, 2008.

[17] The Pròtège Team. The Pròtège website. urlhttp://protege.stanford.edu/, September 2013.

[18] A. Shimojima. Inferential and expressive capacities of graphical representations: Survey and some generalizations. In *Proceedings of 3rd International Conference on the Theory and Application of Diagrams*, volume 2980 of *LNAI*, pages 18–21, Cambridge, UK, 2004. Springer.

[19] S. J. Shin. *The Logical Status of Diagrams*. CUP, 1994.

[20] B. Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Proceedings of the IEEE Symposium on Visual Languages, VL'96*, page 336. IEEE, 1996.