Formal Issues in Languages Based on Closed Curves

Andrew Fish and Gem Stapleton
Visual Modelling Group
University of Brighton
Brighton, UK
www.cmis.brighton.ac.uk/research/vmg
{andrew.fish,g.e.stapleton}@brighton.ac.uk

Abstract

Three important questions arise when using visual languages: for any given piece of information can we draw a diagram representing that information, can we reliably interpret the diagrams and can we reason diagrammatically about that information? The desirable answer to all three questions is yes, but these desires are often conflicting; for example, well-formedness conditions can be enforced to assist diagram interpretation but this can result in drawability problems. In this paper, we focus on visual languages based on closed curves, which are used in numerous computing applications. Many such languages effectively use spatial properties such as containment and disjointness. We consider the consequences of enforcing various well-formedness conditions, such as simplicity and connectedness of minimal regions, in relation to the above questions. We suggest refinements of the conditions in order to find a balance between the conflicting desires.

1 Introduction

We will use the term Euler diagram in a very general sense, to mean any finite collection of closed curves which express information about intersection, containment or disjointness. Often, well-formedness conditions are imposed on Euler diagrams. These conditions are usually chosen in order to alleviate mental difficulties in the user's interpretation of the diagrams but they may also be due to application domain requirements. An example of one such condition, which is often enforced, is that the closed curves must be simple. We aim is to raise awareness of consequences that emerge as a result of well-formedness conditions.

Euler diagrams have numerous applications, including the visualization of statistical data [6], displaying the results of database queries [34] and representing non-hierarchical computer file systems [8]. They have been used in a visual semantic web editing environment [24, 35] and for viewing clusters which contain concepts from multiple ontologies [16]. Closed curves are a basis for many of the UML notations, including class diagrams and statecharts [9]. Another major application area is that of logical reasoning [5, 11, 15, 19, 20, 21, 26, 28, 30, 31, 33]. For example, the constraint diagram logic [21] is based on Euler diagrams and is used for formal object oriented specification [18, 22].

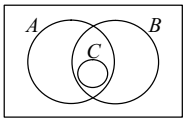

Figure 1. An Euler diagram.

Any Euler diagram can be described by listing the set intersections that are present in the diagram. For example, the diagram in figure 1 can be described by: $A \cap \overline{B} \cap \overline{C}$, $A \cap B \cap \overline{C}$, $B \cap \overline{A} \cap \overline{C}$, $A \cap B \cap C$ and $\overline{A} \cap \overline{B} \cap \overline{C}$ (where $\overline{A}$ is the complement of A etc). Each of the items in this list corresponds to a region in the diagram; for example $A \cap \overline{B} \cap \overline{C}$ corresponds to the region which is inside the curve A, but outside the curves B and C. A natural question arises: given an Euler diagram description, does there exist an Euler diagram with that description? In other words, is the Euler diagram description *drawable*? The answer depends on the well-formedness conditions enforced. Classifying which descriptions are drawable under each set of well-formedness conditions is important: it will allow us to inform users as to which conditions permit their information to be visualized. In this paper, we specify well-formdness conditions that allow all descriptions to be drawn.

When visualizing information, we may want to perform some logical reasoning or transform a diagram if information is updated. Ideally, transformation rules (of which reasoning rules are a special case) will modify diagrams in such a way that users can easily identify the change, pre-

serving their mental map as much as possible. Whether transformation rules can be applied in such a way is, again, affected by the well-formedness conditions enforced.

The accuracy of diagram interpretation is also intertwined with the well-formedness conditions. We demonstrate that not enforcing any well-formedness conditions renders some diagrams ambiguous unless care is taken when defining semantics.

2 Describing Euler Diagrams

We describe Euler diagrams in terms of the labels associated to their curves. Given a finite collection of labels, $L = \{L_1, L_2, ..., L_n\}$, let W be a subset of $\mathbb{P}L - \{\emptyset\}$. The pair (W, L) is an **Euler diagram description**. The elements of W describe the minimal regions; a **minimal region** (called a *zone* in [11]) is a maximal set of points in the plane that are interior to some set of curves and exterior to the remaining curves. An element of W is the set of labels of the curves that contain the corresponding minimal region. For example, $(W = \{\{A\}, \{B\}, \{A, B\}, \{A, B, C\}\}, L = \{A, B, C\})$ describes the Euler diagram in figure 1. The minimal region which is inside A but outside B and C corresponds to the set $\{A\}$ in W. If $W = \mathbb{P}L - \{\emptyset\}$ then (W, L) describes a Venn diagram. All Venn diagram descriptions are drawable with simple closed curves and connected minimal regions [25].

3 Simple Closed Curves

The closed curves in Euler diagrams are often required to be simple [4, 6, 15, 19, 32, 34] but not always [2, 5, 9, 28, 30, 33]. Formally, a **curve** is a continuous function, f, defined on the interval $[0, 1]$. If $f(0) = f(1)$ then f is **closed**. If, for all $x, y \in [0, 1]$, $f(x) = f(y)$ implies $x = y$ or $|x - y| = 1$ then f is **simple** [3]; that is, simple closed curves do not self-intersect.

3.1 Consequences of Enforcing Simplicity

The Jordan Curve Theorem states that any simple closed curve with codomain $\mathbb{R}^2$ splits $\mathbb{R}^2$ into precisely two pieces, one bounded and the other unbounded [1]. Given the image of a simple closed curve, we can easily identify the curve's interior because the interior is the bounded piece. The identification of the interior of closed curves (simple or otherwise) is crucial in order to be able to interpret Euler diagrams. When formalizing the semantics of Euler diagrams, it is usually stated that the interior of each curve represents the set denoted by that curve's label. Many definitions given in the literature rely on the notion of interior [15, 19, 30, 33]; an example is given below.

Definition 3.1 *The set of all points interior to a closed curve is a* ***basic region****. A* ***region*** *is defined using operations union, intersection, difference and complement on basic regions.*

For example, in figure 1, the basic region interior to A consists of the three minimal regions inside A. We can safely use definitions that rely on the interiors of simple closed curves because there is a well-defined (and intuitive) notion of what constitutes the interior.

Unfortunately, enforcing simplicity has the consequence that not every Euler diagram description is drawable [23, 34]. This has implications when defining transformation rules. An example of such a rule allows the removal of a minimal region; this is used when reasoning (see [19, 33], for example) and is also useful when information being visualized is updated. There is no guarantee that such a region can be nicely removed from a diagram (under a continuous transformation of the plane), maintaining the simplicity of curves. A naive technique to remove a minimal region, which does not work in general, is to shrink the region to a point, thus ensuring that the resulting diagram looks similar to the original diagram. For example, in figure 2, the shaded minimal region can be nicely removed from d_1 to give d_2.

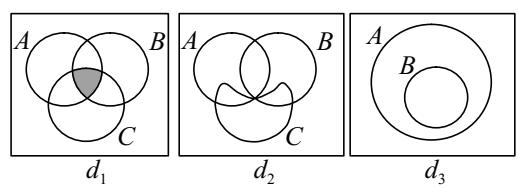

Figure 2. Shrinking minimal regions.

The ability to shrink a region to a point depends upon the local geometry; for example a star like region, unlike a non-simply connected region, can be contracted to a point. In figure 2, d_3 contains a non-simply connected minimal region – interior to A but exterior to B – which cannot be shrunk to a point. Even if it is possible to shrink a given minimal region to a point, this may yield a diagram, d, that contains non-simple curves. In some cases, it is not even possible to draw a diagram using only simple closed curves that has same description as d. For example, Venn-(9) is drawable with simple closed curves, but removing all minimal regions except for those described by $\emptyset, \{A, B, C\}, \{D, E, F\}, \{G, H, I\}, \{A, D, G\}, \{B, E, H\}, \{C, F, I\}$ leaves a diagram whose description is not drawable when simplicity is enforced [23, 34].

The undrawability of some collections of set intersections can have profound effects on logical reasoning systems. Reasoning rules are examples of transformation rules which specify, syntactically, when a diagram, d_2, can be obtained from another diagram, d_1. A reasoning rule is

valid if the semantics of d_2 can be deduced from the semantics of d_1. Such rules are usually defined in terms of a pre-condition and a post-condition (sometimes implicitly), as in [19, 31, 33]. The expectation of such a contract is that if the pre-condition holds for a well-formed diagram d_1 then there exists a well-formed diagram d_2 that satisfies the post-condition. Therefore, the pre-condition should be made strong enough, or the post-condition weak enough, so that this is indeed the case.

However, being able to specify a strong enough pre-condition can be difficult. This is because it is currently unknown which diagram descriptions are drawable under simplicity: to specify a strong enough pre-condition we need to know that applying the transformation rule will produce a well-formed diagram in return. Furthermore, specifying a weak enough post-condition so that a well-formed d_2 exists can result in the rule not being valid, meaning that the logic is unsound.

A **proof** is a sequence of diagrams $\langle d_1, d_2, ..., d_n \rangle$ such that, for all $1 \leq i < n$ each d_{i+1} is obtained from d_i by applying a reasoning rule [13]. Some reasoning systems allow a set of premise diagrams (as opposed to a single premise diagram), as in [30, 33], and the definition of a proof can be adapted. Ideally, reasoning systems will be **complete**: for any diagrams d_1 and d_n, if the semantics of d_n can be inferred from the semantics of d_1 then there is a proof of d_n from d_1. Suppose that the semantics of d_2 can be inferred from those of d_1 and we wish to write a proof from d_1 to d_2. It is possible that the undrawability of some diagram descriptions can prevent there from being a proof from d_1 to d_2 (if any 'proof' of d_2 from d_1 has to pass through a non-wellformed diagram), thus making the system incomplete. Even if the system is complete, it may be the case that some 'proofs' would naturally pass through diagrams that fail the well-formedness conditions and so some natural 'proofs' are unobtainable.

Many completeness proof strategies are constructive, giving a sequences of reasoning rule applications that transform the premise diagram into the conclusion diagram, such as those in [15, 19, 30, 31, 33]. A potential source of error in this type of completeness proof arises if the pre-conditions are not strong enough; in this case rules are not always applicable even when the pre-condition is satisfied but such completeness proofs sometimes ignore this issue. It is possible to overcome the undrawability issue and its consequences by not enforcing simplicity.

3.2 Consequences of Not Enforcing Simplicity

Theorem 3.1 shows, as a benefit of not enforcing simplicity, that all descriptions are drawable. This removes the potential problem of reasoning systems being incomplete because of the non-applicability of reasoning rules.

Theorem 3.1 *Let $(W, L = \{L_1, ..., L_n\})$ be an Euler diagram description. Then there exists an Euler diagram, d, with description (W, L).*

Proof To construct d proceed as follows. For each $S_i \in W$, draw one simple closed curve C_i labelled by S_i, such that if $|W| > 1$ then

1. there exists a unique point, p, in the image of all of the C_i's and
2. no other point is in the image of two (or more) distinct C_i's and
3. the interiors of the C_i's are pairwise disjoint.

For each $L_j \in L$ such that there is no $S_i \in W$ satisfying $L_j \in S_i$, draw a line in the plane, C_i, labelled by $\{L_j\}$, so that C_i does not intersect any curve already drawn. The diagram d consists of $|L|$ closed curves K_j (with label L_j) where the image of K_j is the union of the images of the C_i's for which $L_j \in S_i$. The resulting diagram has description (W, L) by construction, since each required minimal region appears in exactly one of the curves C_i and the only other minimal region present is outside all the $K_j's$.

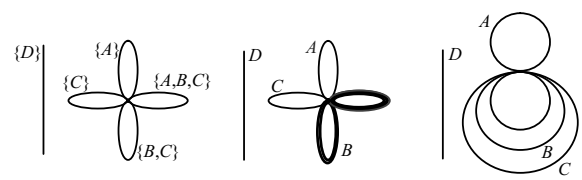

Figure 3. Diagram construction.

An example of this construction process can be seen in figure 3. Starting with the description $(W = \{\{A\}, \{C\}, \{B, C\}, \{A, B, C\}\}, L = \{A, B, C, D\})$, we draw four curves labelled appropriately (the C_i's, shown in the lefthand diagram). To construct the required Euler diagram d (the middle diagram) we use the images of these four curves. For example, the image of the curve in d labelled A is obtained by taking the union of the images of the simple curves labelled $\{A\}$ and $\{A, B, C\}$. The rightmost diagram, where the curve labelled A is a figure of eight, also has description (W, L) but better exploits the containment properties of Euler diagrams.

From the construction process given in the proof of theorem 3.1 we can extract a refined set of well-formedness conditions and maintain the drawability of every Euler diagram description: the closed curves either self-intersect at most once or are lines and, in addition, each minimal region is connected. Furthermore, we can weaken these well-formedness conditions and still maintain drawability: the closed curves either self-intersect a finite number of times or are lines.

Our focus now turns to the interpretation of diagrams when simplicity is not enforced. Formally, the interior of

a closed curve is defined by using *winding numbers* [3]. Knowing the image of a closed curve is not sufficient information to determine its interior. In practice we are only given the images so a method to identify interior points is required (discussed later in this section).

Various reasoning systems based on Euler diagrams use additional syntax to represent elements or individuals [5, 11, 19, 30, 33]. For example, the Euler/Venn system uses *constant sequences* to represent individuals [33]. The Euler/Venn diagram d_1 in figure 4 is ambiguous: is Jean a software engineer? It may well be the case that many people would say Jean is a software engineer because Jean is seemingly placed inside Software Engineers. Furthermore, it might appear that there are two curves, one of which is labelled by Jean; we are not necessarily able to correctly identify the syntactic components of diagrams in the non-simple case, causing further semantic ambiguity.

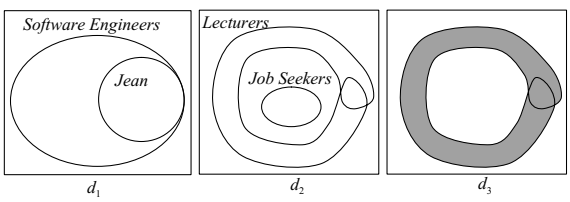

Figure 4. Ambiguous diagrams.

Ambiguity also arises when using self-intersecting closed curves in Euler diagrams (even without any extra syntax). The Euler diagram d_2 in figure 4 is ambiguous for a similar reason: are all job seekers also lecturers or are none of them lecturers? In this case, even though the curve labelled Job Seekers is placed in a bounded component of $\mathbb{R}^2$ minus the image of the curve labelled Lecturers, there is not an intuitive notion that the Job Seekers curve is 'inside' the Lecturers curve: the shaded part of d_3 indicates a likely 'interior' of the Lecturers curve.

We conclude that additional information is essential for the unambiguous interpretation of d_2 (and d_1). One may identify the interior and exterior points given only the images using some canonical method. Such a method needs to be clearly stated because there are various choices that give rise to different interiors; we now give two such methods.

Method 1 If we only allow curves, C, to have a finite number of self intersections then we can define the interior of the image of C as follows. Pick a point, p, in the unbounded region and let x be a point not equal to p and not in the image of C. Let γ be a path from p to x that does not pass through any point where C self-intersects and γ only intersects C transversely and a finite number of times. If γ intersects C an odd number of times then x is interior to the image of C (this is well-defined because the parity of the number of crossings is the same for any such path γ). Using this method, no job seekers also happen to be lecturers, agreeing with our intuitive interpretation of d_2, but, for d_1, Jean is not identified as a software engineer.

Method 2 A point, x, is interior to the image of C if x is in a bounded face of $\mathbb{R}^2$ minus the image of C. This method agrees with the intuitive interpretation of d_1 that Jean is a software engineer, but gives the non-intuitive interpretation of d_2 identifying all job seekers as lecturers.

Our two methods give different notions of the interiors of the image of closed curves. Together with our examples in figure 4, this highlights the importance of specifying such a method when not enforcing simplicity. No method for identifying interior points from the images has been stated in the publications on Euler diagram reasoning where simplicity is not enforced.

As well as affecting semantic problems, not enforcing simplicity affects the way we can define reasoning systems. In figure 5, the basic region interior to A in d_1 is the set of shaded points. This shading frequently is used to assert that A represents the empty set. In d_2, A has no interior and, therefore, also represents the empty set.

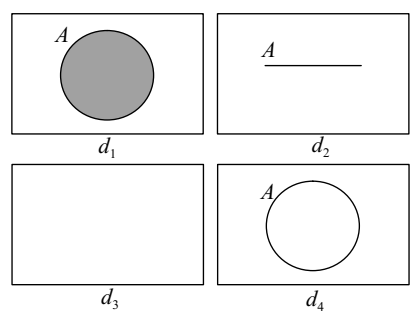

Figure 5. Reasoning with Euler diagrams.

A reasoning rule in the Euler/Venn system (which is similar to a rule in [19] for spider diagrams) states that any shaded minimal, but not basic, region can be removed from a diagram [33]. So, the *removing shaded regions* rule cannot be applied to the diagram d_1 in figure 5 to give d_2 because the only shaded region in d_1 is also a basic region. The pre-condition ensures that after an application of the rule there is at least one minimal region inside each curve because of the 'not basic' constraint. Since the diagrams d_1 and d_2 are semantically equivalent, if the reasoning system is to be complete then we must be able to write a proof of d_2 from d_1. There is a sequence of rule applications to deduce d_2 from d_1: erase A from d_1, giving d_3; introduce A to d_3 giving d_4; unify d_1 and d_4 to give d_2 (see [33] for details of these rules). In this instance, completeness may not have been affected by the overly strong pre-condition, but a complex sequence of rule applications is required in order to make a seemingly trivial deduction.

A reason for not allowing the removal of basic regions in [19] where simplicity is enforced is to avoid applications of rules yielding non-wellformed diagrams like d_2. When allowing non-simple closed curves, as in [33], we can relax the pre-condition of the removing shaded regions rule to allow the removal of basic regions.

When applying reasoning or, more generally, transformation rules it is a great advantage, from a usability perspective, if the rules can be applied in such a way that a user's mental map is minimally disrupted. The following theorem tells us that, when no well-formedness conditions are enforced, a minimal region can be removed under a continuous transformation of the plane, helping to preserve a user's mental map.

Theorem 3.2 *Let d be an Euler diagram. Any minimal region, m, can be removed from d by shrinking m.*

4 Connected Minimal Regions

One well-formedness condition, nearly always enforced, is that minimal regions must be connected components of the plane. If we enforce connectedness then we cannot necessarily delete any curve, which is a typical reasoning rule [15, 19, 30, 33], and maintain well-formedness. For example, in figure 6, the deletion of any curve will result in disconnected minimal regions.

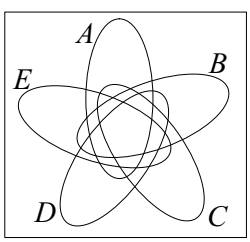

Figure 6. Venn-5.

Enforcing connectedness comes with a price: to get complete reasoning systems, such as Venn-II [30], we must redraw the diagram where necessary after deleting a curve; for example, deleting a curve from Venn-(5) gives Venn-(4) which is drawable, but we may have to completely redraw the diagram rather than simply deleting the curve. This places a larger burden on the user in terms of understanding (due to mental map difficulties when presented with a redrawing) or in terms of effort (when the user is forced to redraw). In fact, there are few choices of curve that can be deleted from Venn-(n) whilst maintaining well-formedness.

Theorem 4.1 *Let d be a Venn diagram that has connected minimal regions. There are at most three choices of curve that can be deleted from d without leaving disconnected minimal regions [29].*

5 Inductive Definitions

Some definitions of Euler diagrams are inductive [5, 28, 30, 33] and we can think of this as placing another well-formedness condition on Euler diagrams.

Definition 5.1 *An Euler diagram is **inductive** if it can be constructed as follows. A rectangle is an inductive Euler diagram. If d_1 is an inductive Euler diagram and d_2 results by drawing a closed curve, C, completely within the rectangle of d_1 so that all of the minimal regions of d_1 are split by C into at most two new minimal regions then d_2 is an inductive Euler diagram.*

The diagrams in figure 7 are 'inductive diagrams'. For example, d_3 in figure 7 is obtained from d_2 by adding B. The curve B splits the minimal region inside A into two minimal regions, one of which is disconnected: the minimal region interior to A but exterior to B in d_3 consists of two components of the plane minus the images of the curves. In definition 5.1, it is stated that the new curve splits each existing minimal region into at most two new minimal regions. We observe that this condition is redundant: a curve can only split minimal regions into at most two regions. We believe there is confusion between the notion of minimal regions in the usual combinatorial sense (a minimal region is interior to some set of closed curves and exterior to the remaining curves) and in a topological sense (a connected component of the plane minus the images of the curves). It is likely that the intention of the definition writer is to enforce the connectedness of minimal regions; if this is the case d_3 should not be an inductive diagram.

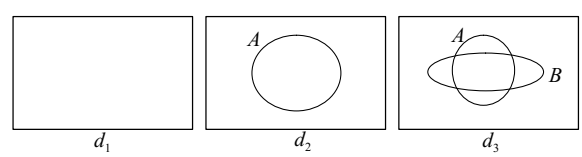

Figure 7. Inductive Euler diagrams.

The diagram in figure 6 does not comply with the inductive definition when connectedness is enforced since deleting any curve leaves disconnected minimal regions. Given any diagram, it is not immediately obvious whether it satisfies the inductive definition. This is likely to affect the usability of the notation because testing to see if a diagram is inductive can be time consuming and some visually pleasing images (such as that in figure 6) are not inductive.

Perhaps the motivation for inductive definitions of Euler diagrams came from inductive Venn diagram definitions where there is no drawability problem (all Venn diagram descriptions are drawable under the inductive definition). However, it is not even clear whether it is possible to delete

an arbitrary curve from an inductive Euler diagram and produce an inductive diagram in return (even if you redraw).

6 Precise Language and Abstraction

Concrete (drawn) level formalisms are difficult to work with, causing a tendency to use imprecise or informal language. This can easily lead to ambiguities or oversights such as relying on being able to identify the interiors of non-simple closed curves given only their images. By contrast, the use of precise language necessitates a thorough understanding of the problem domain and permits a proper analysis of any consequences arising. The act of formalization often brings to light issues that are not apparent when working informally.

In order to understand the consequences of enforcing well-formedness conditions in their entirety, specialist knowledge of the application domain and some knowledge of geometry and topology is required. For example, a logician is likely to be focused on identifying a complete set of reasoning rules and, in order to achieve this goal, may choose not to enforce simplicity whilst being unaware of the topological consequences.

An alternative to the difficult approach of formalizing at the concrete level is to use an abstract syntax [10, 17] (like an Euler diagram description). Defining transformation rules at an abstract level facilitates a greater level of rigour and precision, which is essential in order to be certain that the reasoning systems are sound and that the rules can be applied correctly in the sense that our expectation of the pre/post contract is met. Using an abstract level is advantageous in that it separates the problems of reasoning from drawability: the rules can be defined independently of any concrete level well-formedness conditions. The reasoning systems in [19, 31] are defined using an abstract syntax but many are defined at the concrete level, as in [5, 15, 30, 33].

A major problem with defining reasoning rules (or any transformation rules) at the concrete level is that specifying the pre-condition completely will require a classification of drawable diagrams under the chosen well-formedness conditions. When transformation rules are defined at the concrete level, sometimes we must redraw a diagram after the application of a rule in which case one may as well have used an abstract syntax so working at the abstract level in general is sensible.

Of course, the concrete diagrams must be generated from the abstract syntax and this may not always be possible under a specified set of well-formedness conditions. However, reasoning at the abstract level allows us to be certain that proofs exist and, because of theorem 3.2, we know that all proofs can be visualized when no conditions are enforced. Furthermore, it is computationally less expensive to apply rules at the abstract level. This computational efficiency is essential when automating the search for proofs; even in some simple cases many thousands of diagrams are produced that are not part of the proof [13], so they do not need to be drawn. There is ongoing work in the drawing community attempting to address the issue of being able to generate diagrams under various well-formedness conditions [7, 6, 12, 14, 34] and also attempting to make diagrams look similar [27].

7 Conclusion

This paper provides a discussion of various issues that arise in the many visual languages based on closed curves, of which there are numerous examples in computing [8, 9, 16, 21, 22, 24, 35]. We have investigated the consequences of enforcing various well-formedness conditions that appear in the literature, especially in the areas of drawability, semantic interpretation and reasoning. Not enforcing the simplicity condition causes problems interpreting Euler diagrams. Completely removing this condition is not sensible: it allows any closed curves ranging from space filling curves [1] to straight lines or even points to be used. This not only causes difficulties in the formalization of, for example, semantics but may lead to confusion, especially when lines represent disjunction as in [5, 11, 19, 30, 33] and dots represent elements as in [11, 19]. If the simplicity condition is not enforced or refined then it is necessary to specify a method for determining the interior of curves given just the images.

We suggest that refinements of the conditions are used (such as refining simplicity to allow only a finite number of self-intersections) in order to provide a balance between conflicting desires. From a reasoning perspective, permitting a finite number of self-intersections allows rules to be applied to more diagrams by enabling the relaxation of the pre-conditions. This may enhance usability in certain circumstances; for example, a minimal region can be removed using a natural process so that the resulting diagram looks similar to the original diagram.

We have provided a construction process which generates a diagram from any Euler diagram description. This will enable the visualization of proofs in logical reasoning systems which are defined at an abstract level; the use of an abstract level separates the issues of drawability from those of reasoning. Further work on the generation of diagrams under various sets of well-formedness conditions is underway, and will enable the visualization of proofs containing diagrams which satisfy conditions specified by a user.

There are other standard well-formedness conditions and in future work we plan to conduct an analysis of them. This may lead to further refinements of the conditions as well as provide a better understanding of their effects. We expect that issues similar to those raised in this paper will occur

in other visual languages even if different well-formedness conditions are imposed.

Acknowledgement Author Stapleton is supported by a Leverhulme Trust Early Career Fellowship. Thanks to John Taylor for his comments on an earlier draft of this paper.

References

[1] M. Armstrong. *Basic Topology*. Springer-Verlag, 1979.

[2] J. Barwise and E. Hammer. Diagrams and the concept of logical system. In G. Allwein and J. Barwise, editors, *Logical Reasoning with Diagrams*. OUP, 1996.

[3] D. Blackett. *Elementary Topology*. Academic Press, 1983.

[4] B. Bultena and F. Ruskey. Venn diagrams with few vertices. *Electronic Journal of Combinatorics*, 5:1–21, 1998.

[5] L. Choudhury and M. K. Chakraborty. On extending Venn diagrams by augmenting names of individuals. In *Proceedings of Diagrams 2004, Cambridge, UK*, volume 2980 of *LNAI*, pages 142–146. Springer-Verlag, March 2004.

[6] S. Chow and F. Ruskey. Drawing area-proportional Venn and Euler diagrams. In *Proceedings of Graph Drawing 2003, Perugia, Italy*, volume 2912 of *LNCS*, pages 466–477. Springer-Verlag, September 2003.

[7] S. Chow and F. Ruskey. Towards a general solution to drawing area-proportional Euler diagrams. In *Proceedings of Euler Diagrams*, volume 134 of *ENTCS*, pages 3–18, 2005.

[8] R. DeChiara, U. Erra, and V. Scarano. VennFS: A Venn diagram file manager. In *Proceedings of Information Visualisation*, pages 120–126. IEEE Computer Society, 2003.

[9] H. Dunn-Davies and R. Cunningham. Propostional statecharts for agent interaction protocols. In *Proceedings of Euler Diagrams 2004, Brighton, UK*, volume 134 of *ENTCS*, pages 55–75, 2005.

[10] A. Fish and J. Flower. Abstractions of Euler diagrams. In *Proceedings of Euler Diagrams 2004, Brighton, UK*, volume 134 of *ENTCS*, pages 77–101, 2005.

[11] A. Fish, J. Flower, and J. Howse. The semantics of augmented constraint diagrams. *Journal of Visual Languages and Computing*, 16:541–573, 2005.

[12] J. Flower and J. Howse. Generating Euler diagrams. In *Proceedings of Diagrams 2002*, pages 61–75, Georgia, USA, April 2002. Springer-Verlag.

[13] J. Flower, J. Masthoff, and G. Stapleton. Generating readable proofs: A heuristic approach to theorem proving with spider diagrams. In *Proceedings of Diagrams 2004*, volume 2980 of *LNAI*, pages 166–181, Cambridge, UK, 2004. Springer.

[14] J. Flower, P. Rodgers, and P. Mutton. Layout metrics for Euler diagrams. In *7th International Conference on Information Visualisation*, pages 272–280. IEEE, 2003.

[15] E. Hammer. *Logic and Visual Information*. CSLI Publications, 1995.

[16] P. Hayes, T. Eskridge, R. Saavedra, T. Reichherzer, M. Mehrotra, and D. Bobrovnikoff. Collaborative knowledge capture in ontologies. In *Proc. of the 3rd International Conference on Knowledge Capture*, pages 99–106, 2005.

[17] J. Howse, F. Molina, S.-J. Shin, and J. Taylor. Type-syntax and token-syntax in diagrammatic systems. In *Proceedings 2nd International Conference on Formal Ontology in Information Systems, USA*, pages 174–185. ACM Press, 2001.

[18] J. Howse and S. Schuman. Precise visual modelling. *J. of Software and Systems Modeling*, 4:310–325, 2005.

[19] J. Howse, G. Stapleton, and J. Taylor. Spider diagrams. *LMS J. of Computation and Mathematics*, 8:145–194, 2005.

[20] C. John. Reasoning with projected contours. In *Proceedings of Diagrams 2004*, volume 2980 of *LNAI*, pages 147–150, Cambridge, UK, 2004. Springer.

[21] S. Kent. Constraint diagrams: Visualizing invariants in object oriented modelling. In *Proceedings of OOPSLA97*, pages 327–341. ACM Press, October 1997.

[22] S.-K. Kim and D. Carrington. Visualization of formal specifications. In *6th Aisa Pacific Software Engineering Conference*, pages 102–109, Los Alamitos, CA, USA, IEEE, 1999.

[23] O. Lemon and I. Pratt. Spatial logic and the complexity of diagrammatic reasoning. *Machine GRAPHICS and VISION*, 6(1):89–108, 1997.

[24] J. Lovdahl. *Towards a Visual Editing Environment for the Languages of the Semantic Web*. PhD thesis, Linkoping University, 2002.

[25] T. More. On the construction of Venn diagrams. *Journal of Symbolic Logic*, 23:303–304, 1959.

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

[27] P. Rodgers, P. Mutton, and J. Flower. Dynamic Euler diagram drawing. In *Visual Languages and Human Centric Computing, Rome, Italy*, pages 147–156. IEEE Computer Society Press, September 2004.

[28] H. Sawamura and K. Kiyozuka. JVenn: A visual reasoning system with diagrams and sentences. In *Proceedings of Diagrams 2000*, volume 1889 of *LNAI*, pages 271–285, Edinburgh, UK, 2000. Springer-Verlag.

[29] P. Scotto di Luzio. Patching up a logic of Venn diagrams. In *Proceedings 6th CSLI Workshop on Logic, Language and Computation*. CSLI Publications, 2000.

[30] S.-J. Shin. *The Logical Status of Diagrams*. Cambridge University Press, 1994.

[31] G. Stapleton, J. Howse, and J. Taylor. A decidable constraint diagram reasoning system. *Journal of Logic and Computation*, 15(6):975–1008, December 2005.

[32] N. Swoboda and G. Allwein. Using DAG transformations to verify Euler/Venn homogeneous and Euler/Venn FOL heterogeneous rules of inference. *J. on Software and System Modeling*, 3(2):136–149, 2004.

[33] N. Swoboda and G. Allwein. Heterogeneous reasoning with Euler/Venn diagrams containing named constants and FOL. In *Proceedings of Euler Diagrams 2004*, volume 134 of *ENTCS*. Elsevier Science, 2005.

[34] A. Verroust and M.-L. Viaud. Ensuring the drawability of Euler diagrams for up to eight sets. In *Proceedings of Diagrams 2004*, volume 2980 of *LNAI*, pages 128–141, Cambridge, UK, 2004. Springer.

[35] Y. Zhao and J. Lövdahl. A reuse based method of developing the ontology for e-procurement. In *Proceedings of the Nordic Confernce on Web Services*, pages 101–112, 2003.