Electronic Communications of the EASST
Volume 18 (2009)

Proceedings of the
Eighth International Workshop on
Graph Transformation and Visual Modeling Techniques
(GT-VMT 2009)

Euler Diagram Transformations

Andrew Fish

17 pages

Guest Editors: Artur Boronat, Reiko Heckel
Managing Editors: Tiziana Margaria, Julia Padberg, Gabriele Taentzer
ECEASST Home Page: http://www.easst.org/eceasst/ ISSN 1863-2122

Euler Diagram Transformations

Andrew Fish[1]

[1] andrew.fish@brighton.ac.uk, http://www.brighton.ac.uk/cmis/contact/details.php?uid=agf
School of Computing, Mathematical and Information Sciences
University of Brighton, UK

Abstract: Euler diagrams are a visual language which are used for purposes such as the presentation of set-based data or as the basis of visual logical languages which can be utilised for software specification and reasoning. Such Euler diagram reasoning systems tend to be defined at an abstract level, and the concrete level is simply a visualisation of an abstract model, thereby capturing some subset of the usual boolean logic. The visualisation process tends to be divorced from the data transformation process thereby affecting the user's mental map and reducing the effectiveness of the diagrammatic notation. Furthermore, geometric and topological constraints, called wellformedness conditions, are often placed on the concrete diagrams to try to reduce human comprehension errors, and the effects of these conditions are not modelled in these systems.

We view Euler diagrams as a type of graph, where the faces that are present are the key features that convey information and we provide transformations at the dual graph level that correspond to transformations of Euler diagrams, both in terms of editing moves and logical reasoning moves. This original approach gives a correspondence between manipulations of diagrams at an abstract level (such as logical reasoning steps, or simply an update of information) and the manipulation at a concrete level. Thus we facilitate the presentation of diagram changes in a manner that preserves the mental map. The approach will facilitate the realisation of reasoning systems at the concrete level; this has the potential to provide diagrammatic reasoning systems that are inherently different from symbolic logics due to natural geometric constraints. We provide a particular concrete transformation system which preserves the important criteria of planarity and connectivity, which may form part of a framework encompassing multiple concrete systems each adhering to different sets of wellformedness conditions.

Keywords: Euler Diagrams, Graph Transformations, Logical Reasoning

1 Introduction

Euler diagrams are used as the underlying structures in many application areas for the representation of set-based information, such as: non-hierarchical directories [DES03, DF07], complex genetic set relations [KMGB05], ontologies in semantic web applications [HES$^+$05], statistical data [CR03], and as the basis of logical specification and reasoning systems which can be used in software systems development (e.g. Constraint diagrams [FFH05, Ken97]). Logical reasoning systems based on Euler diagrams (e.g. Spider diagrams [HST05]) tend to apply to the abstract level, with the concrete level simply being a visualisation of the abstract level. Wellformedness conditions are often imposed on concrete diagrams in order to try to enhance comprehensibility of the diagrams, but this can lead to some abstract diagrams not being representable by concrete diagrams. Developing abstract level reasoning systems avoids the complexities of concrete level reasoning, but accordingly denies us the development of a truly diagrammatic reasoning system affected by geometric and topological constraints. If we are to define transformation rules at the concrete level then they should *lift* to the abstract level (in the sense that they can be viewed as an instantiation of an application of the abstract rules to their abstract models, but the constraints imposed at the concrete level can restrict the application of the rules). The development of broader transformation systems with more generic manipulations than logical reasoning rules will enable wider applicability of the systems.

We present a brief outline of the methodology used upfront, using Figure 1, for reference purposes; details and explanations of terminology will follow. The left hand part of Figure 1 depicts the generation process for wellformed Euler diagrams (see [FFH08] for details) which involves constructing an abstract labelled graph (called the superdual in [FFH08], or a closeness graph in [Cho07]) from an abstract diagram d_1, finding a planar spanning subgraph that satisfies connectivity conditions and embedding it in the plane so that it satisfies certain face conditions, yielding the "dual" of an appropriate concrete diagram. Varying the wellformedness conditions that are imposed on the system affects the conditions imposed on the graphs but the same general approach can be taken (this relaxation of some conditions was performed in [Cho07]).

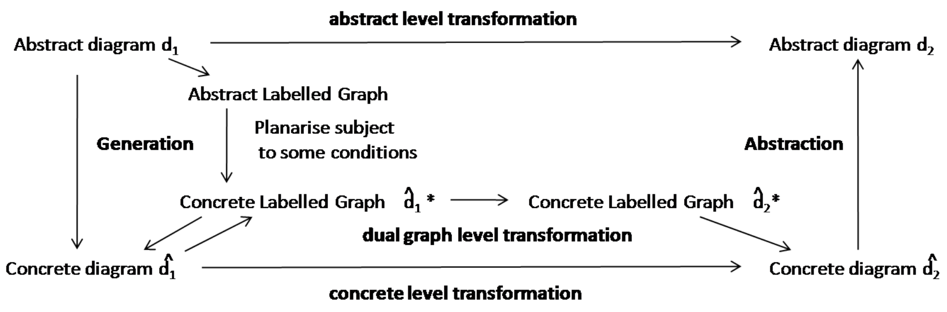
abstract level transformation
Abstract diagram d1
Abstract diagram d2
Abstract Labelled Graph
Generation
Planarise subject to some conditions
Abstraction
Concrete Labelled Graph d̂1*
Concrete Labelled Graph d̂2*
dual graph level transformation
Concrete diagram d̂1
Concrete diagram d̂2
concrete level transformation

Figure 1: An overview of the generation process and the transformation systems involved.

Transformations of diagrams at the abstract level can only alter the abstract sets of zones and contours, and these are straightforward to describe (typically they would be add or remove abstract contours or zones; see Definition 6 and Example 3). However, transformations of diagrams at the concrete level would involve geometric transformations which can be very hard to describe (see Section 3 for an example). Also, one must ensure that the transformations do not cause the violation of the wellformedness conditions that are imposed on the system. Therefore, we define a transformation system on the labelled dual graphs of the concrete diagrams. Note that we perform operations on the embedded labelled dual graphs because we want to preserve the mental map, but a similar approach can be taken at an abstract dual graph level; this may provide a system in which rules are more often applicable, but will lose some of the geometric information contained in the concrete dual graphs. So, as depicted in Figure 1, given an abstract diagram d_1 and a generated concrete diagram $\hat{d_1}$ with labelled dual graph $\hat{d_1}^*$, we apply transformations which take $\hat{d_1}^*$ and return $\hat{d_2}^*$, which is the dual of concrete diagram $\hat{d_2}$. We show that these transformations lift to abstract level transformations in the sense that the abstraction of the concrete diagram $\hat{d_2}$ is the abstract diagram d_2 obtainable from the abstract level transformation.

After discussing some related work in Section 1.1, we recall background definitions of Euler diagrams at both concrete and abstract levels, the notion of wellformedness conditions imposed on concrete diagrams, and develop further the notions of the graph of an Euler diagram and its labelled dual graph in Section 2. We develop transformation systems at the abstract diagram level and then at the concrete dual graph level (corresponding to concrete diagram transformations that lift to abstract diagram transformations), in Section 3. Conclusions and future work plans are discussed in Section 4.

1.1 Related work

In [MMM08] the authors present a general layout approach which operates on either the abstract or concrete syntax levels of a diagram language, using graph transformations at the concrete level and model transformations at the abstract level. They examined the relative benefits of each level of application using deterministic finite state automata as their example language, and indicated that concrete level specification was more natural, whereas abstract level specification was less error prone. Layout refinement, or beautification [CMP99], refers to the process of improving an initial layout via small changes measured by metrics designed to capture aesthetic qualities for example. In [RMF04], the problem of the production of pairs of Euler diagrams (with additional graphs overlaid) that were similar in appearance was addressed using layout refinement methods: they developed comparison measures for Euler diagrams, integrated them into a multicriteria optimizer, and applied a force model for the associated graphs that attempted to move nodes towards their positions in the original layout. This approach utilises existing generation techniques for each of the pair of diagrams independently, not taking account the other diagram. Therefore, this does not facilitate on-line transformations such as contour addition; an on-line approach is likely to be more efficient in this case due to the complexity of the generation procedure (in [Cho07] the Euler diagram generation problem is shown to be NP-complete under a given set of wellformedness conditions). In this paper, our approach inherently takes account of a diagram to be transformed and enables the on-line update of diagrams. Furthermore, the Euler diagram comparison measures and beautification methods of [RMF04] could be applied as a post-processing step after the creation of the transformed dual graphs, thereby avoiding the major computational difficulties but gaining the major benefits of their approach.

In [SHRZ08], the addition of curves to wellformed Euler diagrams is considered, adopting an ad-hoc approach utilising an extension of a dual graph, but no proof of correctness is provided. Their approach extends to nested wellformed Euler diagrams by using the method of decomposing Euler diagrams developed in [FF08]. This method for curve addition could be viewed as a slightly obscured version of a specific case of the methodology for curve addition developed in this paper. However, our method is generic, operating on and returning actual concrete dual graphs, and can be applied to diagrams which break several of the wellformedness conditions.

In [MELS95], the term layout adjustment is used for geometric transformations that adjust a layout after its initial creation. They suggest several models (orthogonal ordering, proximity relations, topology) for the mental map, which should be preserved if possible when adjusting graph-based diagrams; that is, when adjusting a diagram the relative ordering of nodes and the proximity of nodes of the original graph should be preserved as much as possible, as should the dual graph. In an interactive setting they consider node addition and identify techniques for ensuring node disjointness after the insertion. More recently, in [LLY06] simulated annealing is used to try to maintain the mental map, with cost function incorporating six criteria of [BT02] where similarity measures of graphs were defined and tested against human perception. In this paper we are considering substantial transformations to a diagram such as adding or removing contours, and we consider the primary notion of preserving the mental map to be realised by the preservation of as much the embedded dual graph of the original diagram as possible. Since the diagrams are constructed from the dual graphs in this context, this preservation encompasses the main concepts of the mental map models above. Further techniques for layout adjustment of the transformed dual graph could then be performed if necessary.

2 Euler Diagrams and Graphs

First of all we recall the definitions of Euler diagrams, separating the abstract and concrete models as usual, and then we provide a set of wellformedness conditions that are often imposed with the intention of reducing human comprehension errors; Definitions 1 and 2 are adapted from those in [FFH08].

Definition 1 An *abstract Euler diagram* is a pair: $d = \langle C(d), Z(d) \rangle$ where: $C(d)$ is a finite set whose members are called *(abstract) contours*, $Z(d) \subseteq \mathscr{P}C(d)$ is the set of *(abstract) zones* of d, where $\mathscr{P}X$ denotes the powerset of set X, and $\bigcup\limits_{z \in Z(d)} z = C(d)$. If $Z(d) \neq \mathscr{P}C(d)$ then the elements of $\mathscr{P}C(d) - Z(d)$ are called *missing zones*.

Definition 2 A *concrete Euler diagram* is a pair $\hat{d} = \langle C(\hat{d}), F_{\hat{d}} \rangle$ where: $C(\hat{d})$ is a finite set of closed curves, called *(concrete) contours*, in the plane, and $F_{\hat{d}} : C(\hat{d}) \to \mathscr{L}$ is a function associating with each contour a label drawn from an infinite alphabet of labels $\mathscr{L}$. The *label set* $\mathscr{L}(\hat{d})$ of $\hat{d}$ is the set of labels associated with $\hat{d}$: $\mathscr{L}(\hat{d}) = \{F_{\hat{d}}(\hat{c}) \,|\, \hat{c} \in C(\hat{d})\}$. A *minimal region* of a concrete Euler diagram $\hat{d}$ is a connected component of $\mathbb{R}^2 - \bigcup\limits_{\hat{c} \in C(\hat{d})} \hat{c}$. Let $X \subseteq C(\hat{d})$ be a set of contours, and let *interior* $(\hat{c})$ and *exterior* $(\hat{c})$ denote the interior and exterior [1] of a contour $\hat{c}$, respectively. If the set $\hat{z} = \bigcap\limits_{\hat{c} \in X} interior\,(\hat{c}) \cap \bigcap\limits_{\hat{c} \in C(\hat{d}) - X} exterior\,(\hat{c})$ is non-empty, then $\hat{z}$ is a *zone* of $\hat{d}$ (note that a zone is a union of minimal regions); the set of labels associated with the contours in X is the *zone label set* $\mathscr{L}(\hat{z})$ of $\hat{z}$: $\mathscr{L}(\hat{z}) = \{F_{\hat{d}}(\hat{c}) \,|\, \hat{c} \in X\}$.

We say $\hat{d}$ is *wellformed* if all of the following wellformedness conditions (WFCs) hold:

WFC 1 Simple contours: The contours are simple closed curves.

[1] In the transformation systems that we develop in this paper the usual meaning of interior and exterior of simple closed curves are adopted. Suitable extensions are used if non-simple curves are present; see [FS06] for example.

WFC 2 Unique contour labels: Each contour has a unique label; that is, $F_{\hat{d}}$ is injective.

WFC 3 Transverse intersections: Contours meet transversely. This can be subdivided into:

> **WFC 3a** No tangential intersections.
>
> **WFC 3b** No concurrency; that is contours meet at a discrete set of points.

WFC 4 No multiple points: At most two contours meet at a single point.

WFC 5 Connected concrete zones: Each concrete zone is a minimal region.

These are the most commonly considered wellformedness conditions but other constraints could be imposed, such as using fixed geometric shapes like circles or ellipses; for example, the generation of area proportional Euler diagrams with small numbers of circles was investigated in [Cho07].

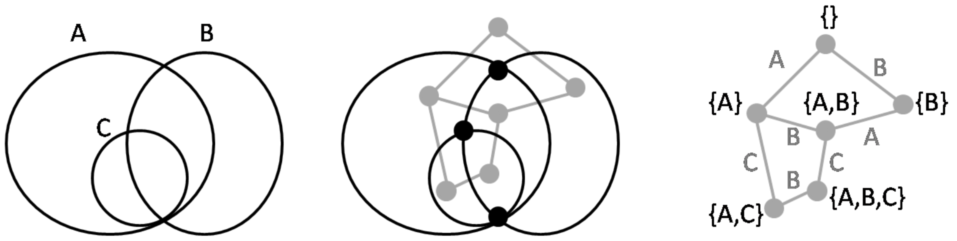

Figure 2: A concrete Euler diagram (left), its graph together with an unlabelled dual graph overlaid in grey (middle) and the labelled dual graph (right) where the vertex labels sets and the induced edge labels are shown (with set brackets on edge labels omitted for readability purposes).

Example 1 *The left of Figure 2 shows a concrete Euler diagram $\hat{d}$ with three contours labelled A, B and C, and six zones which can be informally described as: outside all contours; inside A and outside B and C; inside B and outside A and C; inside A and B but outside C, inside A and C but outside B; inside A, B and C. The associated abstract diagram d is a set of abstract contours, together with a set of abstract zones: $\langle\{A,B,C\},\{\{\},\ \{A\},\ \{B\},\ \{A,B\},\ \{A,C\},\ \{A,B,C\}\}\rangle$, where the abstract zones correspond to the set of contours that the concrete zone is "inside". The concrete diagram $\hat{d}$ fails WFC 3a and WFC 4 since it has a point of intersection of the three curves (i.e. a multiple point) where the contours labelled A and C meet tangentially. The diagram $\hat{d}$ forms a single component and has three branch points (see Definition 4), giving rise to the graph $G(d)$ shown in the middle of the figure. An unlabelled dual graph of $G(d)$ is overlaid, shown in grey, whilst the right of the figure shows $\hat{d}^*$, the dual of the diagram $\hat{d}$.*

Definition 3 Let $\mathscr{L}$ be an alphabet of labels. An *abstract labelled graph* G is a vertex-labelled graph, whose vertex labels are set of labels drawn from $\mathscr{L}$. The label set of G, denoted $\mathscr{L}(G)$, is the union of the vertex label sets of G. An abstract labelled graph which has been embedded in the plane is called a *concrete labelled graph* (i.e. this is a drawing in the plane with no edge crossings, which is sometimes called a *plane labelled graph*). A labelling on the edges of the graph is induced by taking the symmetric difference of the label sets of the incident vertices.

One can view a concrete diagram as a graph as follows; this definition generalises the cases given in [Cho07, FFH08]: the intuition is that branch points are either points of intersection of the curves or places where concurrent curves separate.

Definition 4 Let $C = \{C_1, \ldots, C_n\}$ be a set of curves in the plane, where we also refer to C_i as the images of the curves, as usual. Let x be a point on any curve in C and let $B_\varepsilon(x) = \{y \in \mathbb{R}^2 : |x-y| < \varepsilon\}$ denote a ball of radius ε around x. If $\exists \varepsilon > 0$ and $i \in \{1, \ldots, n\}$ such that $B_\varepsilon(x) \cap C_i$ is (topologically) a line, but $B_\varepsilon(x) \cap C_j = \emptyset$ for any $j \neq i$ then x is a *non-singular point*. If $\exists \varepsilon > 0$, $k > 1$ and $i_1 \neq \ldots \neq i_k \in \{1, \ldots, n\}$ such that $B_\varepsilon(x) \cap C_{i_1} = \ldots = B_\varepsilon(x) \cap C_{i_k}$ is a line but $B_\varepsilon(x) \cap C_j = \emptyset$ for any $j \neq i_1, \ldots, i_k$ then x is a *point of k-concurrency*. If x is not a non-singular point nor a point of k-concurrency then x is a *branch point*.

Let d be concrete diagram. Then a *graph of d* is a plane graph $G(d)$ whose vertex set consists precisely of one vertex at every branch point, together with one vertex on each component (i.e. a maximal set of images of curves that is connected) that has no branch points; and whose edges are the images of the curves joining these vertices. A *dual* of d, denoted d^*, is a concrete labelled graph that is a geometric dual graph of $G(d)$ such that: if vertex v of d^* is placed in zone $\hat{z}$ of d then v is labelled by $\mathscr{L}(\hat{z})$.

The Euler diagram generation process, in [FFH08], takes an abstract diagram and creates a concrete diagram realising it, utilising a "dual graph" of the Euler diagram as part of the construction process; Definition 5 and Theorem 1 are rephrased from [FFH08].

Definition 5 Let G be a labelled graph. For $l \in \mathscr{L}$, let $G^+(l)$ and $G^-(l)$ denote the subgraphs of G induced by deleting any vertices whose labels contain l, and induced by deleting any vertices whose labels exclude l, respectively. If $G^+(l)$ and $G^-(l)$ are connected then we say that l *satisfies the connectivity conditions*. If G is connected and l satisfies the connectivity conditions for all $l \in \mathscr{L}(G)$ then we say that G is *well connected*.

Let G be a well connected plane labelled graph. Then G passes the *face conditions* if each face cycle of G (with distinct vertices and edges) of length $2n$ has crossing index $n-1$, where the crossing index of a face is the number of pairs of labels that are non-nested in the edge-word around the face, read cyclically.

Example 2 *The dual graph d^* shown on the right of Figure 2 has three faces with edge words BABA, CBCB and BACBCA. A pair of labels x and y in an edge word are nested if and only if the occurrences of x (and the occurrences of y) are adjacent (reading the word cyclically) in the projection of the edge word onto the word containing only x and y. In each of the words BABA and CBCB we have only a single pair of labels and these are non-nested. Therefore, both of these internal faces have length $4 = 2n$ and crossing index $1 = n-1$, and so they pass the face conditions. For the outside face word, BACBCA, we have 3 projections onto its pairs of labels to consider: BABA, ACCA and BCBC. The occurrences of C are adjacent in ACCA and so the labels A and C are nested, whereas the other two pairs are non-nested. Therefore, we have*

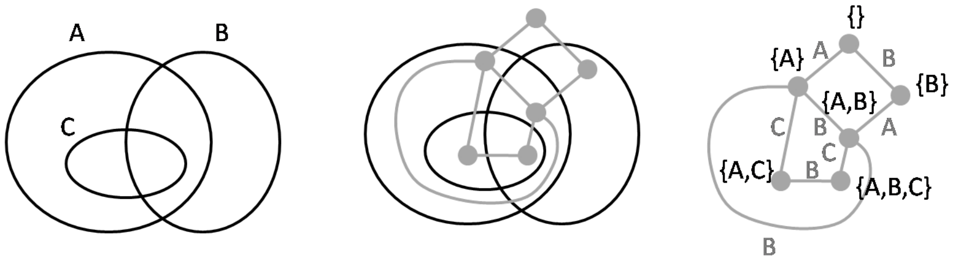

Figure 3: Understanding the face conditions: a concrete Euler diagram (left) and its dual graph (right) for comparison with Figure 2.

word length $6 = 2n$ and crossing index $2 = n-1$, and so d^* passes the face conditions. Since the process of taking the dual graph places exactly one vertex in each region, a word of length more than four causes the creation of a multiple point in the corresponding Euler diagram; this can be seen on the left of the Figure 2 where there is a multiple point incident with the outer zone. However, since d^* passes the face conditions, one can add edges which subdivide faces with edge word length greater than four, having the effect of modifying the Euler diagram by removing the multiple points, as shown in Figure 3. This observation follows directly from the work in [FFH08]. In contrast, if a well connected plane labelled graph G has a face cycle with word ABCABC then all three pairs of labels are non-nested and so the graph would fail the face conditions, indicating that the multiple point in the Euler diagram is necessary and cannot be removed by the above procedure.

Theorem 1 *Let d be an abstract diagram. Then there is a concrete diagram $\hat{d}$ which satisfies all of the WFCs and whose abstraction* [2] *is d if and only if there exists a concrete labelled graph G for d that has the vertex labeled $\{\}$ incident with the outer face, and the properties that: G is well connected (P1), G has unique vertex labels (P2); vertices of G whose label sets differ by more than one label are not adjacent (P3); and G passes the face-conditions (P4).*

The construction used in [FFH08] did not allow multiple edges between pairs of vertices, and such a concrete labelled graph G may require the addition of extra edges in order to construct the labelled dual graph d^*. In [Cho07] a dual graph approach to the generation problem was adopted and similar existence theorems provided. From the theory developed in [FFH08] or [Cho07]:

Proposition 1 *Let d be a concrete diagram and let d^* be a dual of d. If d satisfies (WFC1,2 and 5) then d^* satisfies (P1 and 2). If d^* satisfies (P1 and 2) then there is a concrete diagram d' which has dual d^* and which satisfies (WFC1,2 and 5).*

The idea is if d has only uniquely labelled simple contours with connected zones then unique-

[2] The abstraction refers to the natural mapping from concrete to abstract diagrams; see [FFH08] for details if required. A concrete labelled graph G for d is a concrete labelled graph with the property that there a bijection between between the set of vertices of G and the set of zones of d in which the vertex label sets correspond exactly to the "inside" set of contours of the zones.

ness of contour labels and connectedness of concrete zones in d ensure that the vertex label sets of d^* are all distinct. Then, since the regions in d which are "inside" and "outside" the contour labelled l are both connected, we have that the connectivity conditions for the label $l \in \mathscr{L}$ in d^* hold. Conversely, if d^* satisfies (P1 and 2) then one can take a dual d' which is a graph that can be viewed as an Euler diagram (i.e. we are utilising the usual diagram generation process). The fact that it only uses uniquely labelled simple closed curves follows from the connectivity conditions, whilst the connected zones condition then follows from the unique vertex label condition.

3 Transformation Systems

We first provide paramaterised transformation rules which enable both the generation and manipulation of diagrams at the abstract level, generalising the logical reasoning rules in [FHJT08].

Definition 6 Let $d = \langle C(d), Z(d)\rangle$ be an abstract Euler diagram. Define:

1. *RemoveContour*(l,d): If $\ell \in C(d)$, then d **with** ℓ **removed** is d' where $C(d') = C(d) - \{\ell\}$ and $Z(d') = \{Y - \{l\} : Y \in Z(d)\}$.

2. *AddContour*(l, Z_c, Z_s, d): Let Z_c and Z_s denote (possibly empty) disjoint subsets of $Z(d)$, and suppose that $\ell \notin C(d)$. Then, d **with** ℓ **added**, zones Z_c **covered** and zones Z_s **split** is d' where $C(d') = C(d) \cup \{l\}$ and $Z(d') = (Z(d) - Z_c) \cup \{x \cup \ell : x \in Z_s \cup Z_c\}$.

3. *AddZone*(z,d): If $z \in \mathscr{P}C(d) - Z(d)$ then d **with** z **added** is d' where $C(d') = C(d)$ and $Z(d') = Z(d) \cup \{z\}$.

4. *RemoveZone*(z,d): Let $z \in Z(d) - \{\}$ (so z is not the zone outside all contours). Let $X \subseteq Z(d)$ be the set of contours which are in zone z but are not in any other zone in $Z(d)$. Then d **with** z **removed** is d' where $C(d') = C(d) - X$ and $Z(d') = Z(d) - \{z\}$.

Example 3 *Firstly, let d be the abstract diagram $\langle\{A,B,C\},\{\{\}, \{A\}, \{B\}, \{A,B\}, \{A,C\}, \{A,B,C\}\}\rangle$. Then the operation RemoveZone$(\{A,B,C\},d)$ yields $d' = \langle\{A,B,C\},\{\{\}, \{A\}, \{B\}, \{A,B\}, \{A,C\}\}\rangle$. A concrete diagram with abstraction d is shown at the top middle of Figure 4, whilst a concrete diagram with abstraction d' is shown at the top left of Figure 4.*

Secondly, let d be the abstract diagram $\langle\{A,B,C\},\{\{\}, \{A\}, \{B\}, \{C\}, \{A,B\}, \{A,C\}, \{B,C\}, \{A,B,C\}\}\rangle$. Then the operation AddContour$(D, \{\{A,B,C\}\}, \{\{A\}, \{B\}, \{C\}, \{A,B\}, \{A,C\}, \{B,C\}\}, d)$ yields $d' = \langle\{A,B,C,D\},\{\{\}, \{A\}, \{B\}, \{C\}, \{A,B\}, \{A,C\}, \{A,D\}, \{B,C\}, \{B,D\}, \{C,D\}, \{A,B,D\}, \{A,C,D\}, \{B,C,D\}, \{A,B,C,D\}\}\rangle$. The effect of this operation is to copy the zones that are "split" (zones $\{A\}$, $\{B\}$, $\{C\}$, $\{A,B\}$, $\{A,C\}$, $\{B,C\}$ in this instance) and to add the new label to the copy as well as to the zones that are "covered" (zone $\{A,B,C\}$ in this instance). A concrete diagram with abstraction d is given in Figure 5 at the top middle (called d_2), whilst a concrete diagram with abstraction d' is shown at the top right (called d_3).

Remark 1 *In settings where Euler diagrams represent propositional logic (e.g. see [FHJT08]) such transformations can be restricted to those that induce logical inferences to provide a reasoning system (the transformations are then often called reasoning rules). For instance, the addition*

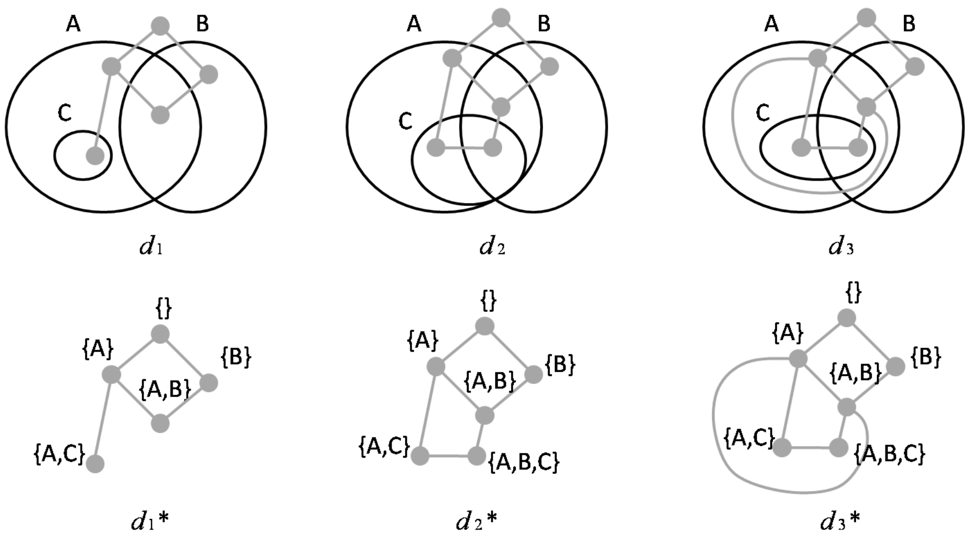

Figure 4: Left/middle: Addition/removal of a vertex to/from the dual graph and a zone to/from the diagram. Middle/Right: a post-processing graph transformation step for removing tangential intersections.

of a new contour by $AddContour(l, Z_c, Z_s, d)$ is a reasoning rule if $Z_s = Z(d)$ and $Z_c = \emptyset$; if shading is used in the system, then the reasoning rule for zone removal has the extra precondition that the zone is shaded, whilst zone addition has the extra postcondition that any missing zone that is added is shaded. The effects of altering rule sets within an automated reasoning environment for Euler diagram systems were investigated in [SMF+07].

Corresponding (geometric) transformations at the concrete level are very difficult to try to realise consistently, and this has not been managed in the literature. For example, there are different possible ways of attempting to realise zone removal: if a zone z has nice properties such as being star-shaped (i.e. there is a point p in z such that every other point in z is connectable to p via a straight line within z) then one could use radial contraction. More generally, an operation to squash the zone to a point could be used if the region is simply connected, but if the zone is not simply connected (e.g. an annulus) then one may desire a transformation that does not identify all of the boundary of the zone to a point. The application of these operations may cause the violation of some wellformedness constraints (such as the simplicity of the contours), thereby either preventing their application at the concrete level or requiring a different concrete representation to be recreated, if one exists, destroying the mental map and the utility of the visualisation even in the case that such a recreation exists. This contrasts with the effect of zone removal at the abstract level which can always be applied to a (non-outside) zone z in the syntactic transformation system.

Therefore, we wish to build concrete level transformation systems for Euler diagrams based on concrete dual graphs manipulations; any instance of a concrete transformation should lift to the appropriate instance of an abstract transformation of Definition 6. Accordingly, we develop

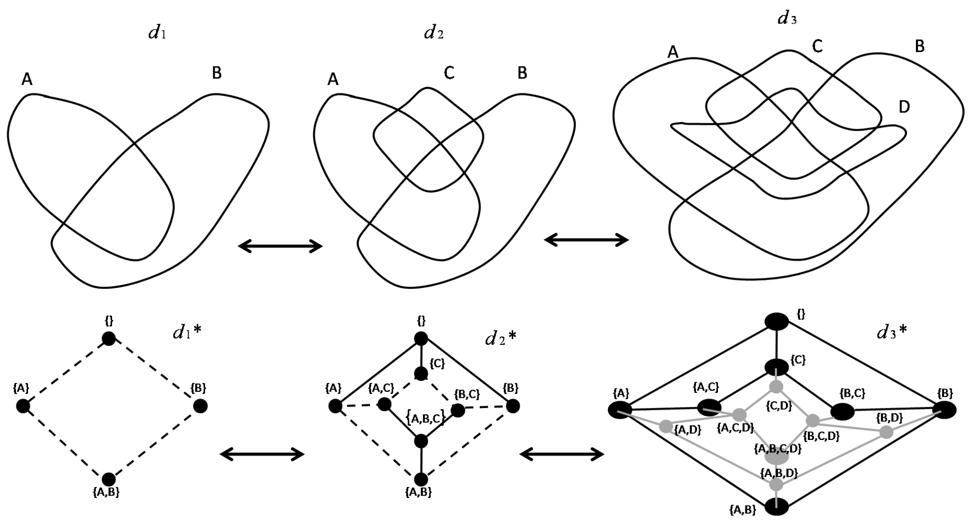

Figure 5: Transformations of the dual graph, and the corresponding Euler diagrams. The top left to top right diagrams show the addition of contour C then D, and the corresponding dual graph transformations are depicted below. Reading from right to left shows the reverse process of contour deletion.

a concrete dual graph transformation system in which all of the plane graphs are well connected and the vertex label set are unique. We note that any abstract diagram has a representation as a concrete diagram which are unions of regions with holes [MF94, RZF08], or Euler-like [Cho07], but commonly one wishes to keep the system as straightforward as possible for the users, and here we have chosen to enforce the use of uniquely labelled simple closed curves and connected zones. However, there are many other variations at the plane graph level that could be adopted, or one could consider abstract dual graph level transformations if one is willing to sacrifice some geometric information.

3.1 Adding and removing contours

We define operations at the dual graph level in order to realise contour addition at the concrete level. Intuitively a *collar* of a path, or a cycle, is a thickening of that path or cycle. However, in this context we incorporate the use of labels and we choose to alter the resulting graph so that we retain planarity, connectivity and uniqueness of vertex label sets. Note that as stated this collaring operation is non-deterministic; any choices involved when edges are to be crossed by the insertion of the collar can lead to different diagram layouts, but here we concentrate on the fundamental properties of planarity and connectivity.

Definition 7 Let d be a concrete diagram and let d^* be a dual of d. Let p be a path of distinct vertices and edges, except possibly for the first and last vertex (i.e. p can be a simple cycle), in d^*. Let p' be a new path (or cycle) which is a copy of p but which has an additional label l added to all of its vertex label sets. Suppose that $p_1, \dots, p_n$ is the vertex sequence of p and $p'_1, \dots, p'_n$ is the vertex sequence of p', where the label of p_i differs from the label of p'_i by the label l. Then $collar(p;d^*,l)$, a *collaring of* p, is a graph obtained from d^* by:

1. embedding p' in the plane such that:
   (a) for each i, the vertex p'_i is disjoint from d^* (i.e. the position of p'_i in the plane is not the same as any vertex and it does not lie on any edge of d^*).
   (b) the edges of p' are disjoint from the edges of p and the vertices of d^*.
   (c) each vertex p'_i is in a neighbourhood of vertex p_i (that is, if p_i has coordinates (a_i,b_i), then $\exists\varepsilon > 0$ such that $p'_i \in B_\varepsilon(p_i) = \{(x,y) \in \mathbb{R}^2 \,|\, (x-a_i)^2 + (y-b_i)^2 < \varepsilon\})$ that contains no other vertex of d^*, and
   (d) if p is a cycle then p' is in the interior of the bounded region that is bounded by p [3].
   (e) if p is not a cycle, then there is a vertex of d^*, labelled by $\{\}$, which is not in any region of the plane bounded by p' and d^*.
2. adding an edge labelled l between vertex p_i and vertex p'_i, for each $i \in \{1,\dots,n\}$.
3. for each edge $e = (p_i,v) \in E(d^*)$ which crosses the path p',
   (a) if $v \in V(d^*) - \{p_1,\dots,p_n\}$ then delete e and add an edge from the vertex p'_i to v.
   (b) if $v = p_j \in \{p_1,\dots,p_n\}$ then delete e and add an edge from the vertex p'_i to p'_j.

Example 4 *The dual graph d_1^* at the bottom left of Figure 5 has a simple cycle p highlighted using dashed edges. The effect of collaring p is shown in the middle dual graph d_2^*; in this case no edges of d_1^* were crossed by the insertion of p' and so no edges of d_1^* were deleted. The effect of the application of collaring on the highlighted cycle in d_2^* is shown by d_3^* at the bottom right; three edges of d_2^* were crossed by the insertion of the path p' and so these were deleted and three new edges were added (the alterations are shown in grey). However, note that the label D has also been added to the vertex labelled $\{A,B,C\}$, which was in the interior of the cycle p in d_2^*, after the collaring operation described; this corresponds to the new contour covering the zone in the diagram (see Theorem 2). The corresponding Euler diagrams are shown, starting at the top left with d_1, performing a transformation that lifts to $AddContour(C,\{\},Z(d_1),d_1)$ to give d_2, shown in the top middle, and then performing a transformation that lifts to $AddContour(D,\{\{A,B,C\}\}$, $\{\{A\}$, $\{A,C\}$, $\{C\}$, $\{B,C\}$, $\{B\}$, $\{A,B\}\},d_2)$ to give d_3 shown at the top right.*

Theorem 2 *Let d^* be a well connected plane labelled graph with unique vertex label sets which is the dual of a concrete Euler diagram d, let p be a path of distinct vertices and edges, or a simple cycle, in d^*, l be a label which is not in $\mathscr{L}(d^*)$ and let H denote $collar(p;d^*;l)$. Then:*

1. *if no edge of d^* is missing from H, then H is a well connected plane labelled graph.*
2. *if there are edges of d^* that are missing from H, then collaring p yields a labelled plane graph which is wellconnected if and only if $H^-(l)$ is connected.*

[3] Note that d^* is embedded in the plane and so the cycle p splits the plane into two regions by the Jordan curve theorem; of course, one can adapt the theory to the non-embedded dual graph level.

3. *if c is a simple cycle of d^* with no vertices in $int(c)$, the interior of the bounded region bounded by c, and the path p lies entirely on the cycle c (this includes the case that p is the entire cycle c), then collaring p yields a well connected plane labelled graph.*

4. *Let Z_s denote the set of vertex label sets of p and let Z_c denote the set of vertex label sets of the vertices in $int(p)$ if p is a cycle (and $Z_c = \{\}$ otherwise). Then $collar(p;d^*;l)$ followed by the addition of label l to all vertex labels of vertices in $int(p)$, if p is a cycle, yields a graph K which is the dual of a concrete diagram d' that differs from d by the addition of a new contour labelled l, covering zones Z_c and splitting zones Z_s; that is, the composite transformation lifts to $AddContour(l,Z_c,Z_s,d)$. In particular, if p is not a cycle then collaring p lifts to $AddContour(l,\{\},Z_s,d)$.*

Proof. A path p of distinct edges and vertices in d^* corresponds to a sequence of adjacent zones in d. Adding a collar of p splits each of these zones into two adjacent zones, one inside l and one outside l, where l is the new label. The definition of collaring ensures that the resultant graph is planar. If no edges of d^* were removed upon collaring then the connectivity conditions also hold: the subgraph $H^+(l)$ is the copy of p with the additional label l, embedded in the plane and viewed as a graph, and so is connected; the subgraph $H^-(l) = d^*$ is connected by hypothesis; the other induced subgraphs do not become disconnected by the addition of the collar. So case 1 holds.

However, if any edges of d^* were removed during collaring then the connectivity conditions for H could be broken. Now $H^+(l)$ is connected by definition, but $H^-(l)$ could be disconnected. The extra edges added during collaring ensure that there are no other obstructions to H being wellconnected, as follows. Suppose that we have a vertex labelled x adjacent to a vertex labelled y in d^* and this edge is removed by the collaring operation. Then, without loss of generality, there is either a path $x - xl - y$ or a path $x - xl - yl - y$ in H. Thus, if x and y had a label k in common in d^*, then all of the vertices in this new path in H have label k, and similarly if they both exclude a label m in d^* then so do all of the vertices in this new path in H. Therefore, the connectivity of $G^+(k)$ and $G^-(m)$ are preserved upon collaring, and part 2 holds.

Part 3 follows since if p lies entirely on a simple cycle c which has no vertices in its interior then $H^-(l)$ is connected. In fact, since the only obstruction to the collaring operation preserving the connectivity conditions is the connectivity of $H^-(l)$, we can ensure that it is preserved in the case of the path being a simple cycle by adding the new label l to any vertices in $int(p)$; this operation does not affect the other connectivity conditions. This composite transformation of dual graphs corresponds to the addition of a new contour, where the zones to be split into two correspond to vertices in the simple cycle p, whilst those to be covered lie in $int(p)$. The effect of lifting to the abstract level can be checked by considering the label sets of the vertices that were present before and are present after the transformation, and so part 4 follows. □

Definition 8 Let G be a concrete labelled graph, and let l be a label in $\mathscr{L}(G)$. Define the operation $RemoveLabel(l)$ to be the contraction of every edge labelled by exactly l, together with the identification of the corresponding vertices [4], followed by the removal of the label l

[4] The label set taken is the union of the two label sets; the intuition is that this edge contraction corresponds to stretching the contour labelled l so that it also covers the zones that it previously split.

from all vertex label sets.

Proposition 2 *Let d^* be a well connected plane labelled graph with unique vertex label sets which is the dual of a concrete diagram d, and suppose that c is a contour of d with label l. Then the operation $RemoveLabel(l)$ on d^* corresponds* [5] *to the removal of the contour c from d.*

Proof. Contraction of edges labelled by exactly l corresponds to the merging of each pair of adjacent zones whose label sets differ by the label l. The removal of the label l from all vertex label sets suitably updates all vertices. □

Example 5 *The dual graphs in Figure 5 from right to left show the removal of labels D and C corresponding to Euler diagram contour deletion. In this case we note that the RemoveLabel operation actually leaves multiple edges between vertices such that there are no other vertices in the interior of the region bounded by these vertices and edges. However, since such multiple edges have no significant effect on the diagrams constructed we can assume that these excess edges can be discarded.*

3.2 Adding and removing zones

When considering the operations of adding or removing zones at the concrete level, the natural addition or deletion of vertices of the dual graph may break the connectivity conditions. Since we wish to ensure that planarity and wellconnectedness are preserved, we consider extra edge additions on the neighbourhood of the vertices to be added or removed.

Definition 9 Let G be a labelled graph and let E be a set of edges in the complement of G. Then E is a *wellconnecting edge set for* G if G together with the edges in E is wellconnected. If G is a plane labelled graph then a *plane wellconnecting edge set for* G is a wellconnecting edge set E for G together with an embedding of E such that $G \cup E$ is a plane graph.

Definition 10 Let G be a well connected plane labelled graph, let v be a vertex of G, and let $G - v$ denote G with v removed. Suppose that E is a plane wellconnecting edge set for $G - v$. Then let $RemoveVertex(v, G)$ denote the operation of removing vertex v from G and adding E [6].

Lemma 1 *If v is a vertex of a well connected plane labelled graph, G, and v has vertex degree at most 3, then the set of edges of the complete graph on the set of vertices incident with v in G is a plane wellconnecting edge set for $G - v$.*

Proof. If vertices v_1 and v_k (not equal to v) in G are connected by a path $p = v_1 v_2 \dots v_k$ that passes through v (i.e. $v = v_i$ for some $i \in \{2, \dots k-1\}$) then they are still connected in $(G - v) \cup E$ by the path $v_1 v_2 \dots v_{i-1} v_{i+1} \dots v_k$. Planarity is ensured to be preserved since v had degree at most 3. □

[5] Note that the remove contour operation in our system is not applicable if the removal of the contour leaves a disconnected zone. This corresponds to the existence of vertices differing by label l which are not adjacent in d^*. Relaxing the zone connectedness condition would mean that contour removal is always applicable.

[6] The choice of edge set used effects the layout but we are primarily concerned with planarity and connectivity.

Definition 11 Let G be a well connected plane labelled graph and let Y be a set of labels which is not the label set of any vertex of G [7]. Suppose that G has a face F whose incident vertices have label sets that contain all of the labels of Y, and there is a plane wellconnecting edge set E for $G \cup v$, where v is a new vertex labelled by Y embedded in the interior of the face F. Then define $AddVertex(Y,G)$ to be the operation which inserts the vertex v with label Y in face F, and adds E.

Remark 2 *There is flexibility in the choice of rules developed and they could be chosen according to system requirements or user preference. The rule $RemoveVertex(v,G)$ is not applicable if there is no plane wellconnecting edge set for $G - v$, but if it is applicable then it preserves planarity and wellconnectivity. If one wanted a more relaxed system which always enabled the application of the rule then one could alter the rule so that the vertex v is always deleted. Furthermore, to attempt to improve the layout one could add a subgraph of the clique on the vertices incident with v that maintains planarity whilst minimising the number of connectivity conditions that are broken. Similarly, one could generalise the $AddVertex(Y,G)$ rule to be always applicable and to attempt to preserve the connectivity conditions for as many labels as possible.*

Proposition 3 *Let d^* be a well connected plane labelled graph with unique vertex label sets which is the dual of a concrete diagram d. Let Y be a set of labels corresponding to a zone z which is missing from d. Suppose that there is a face F of d^* which has every label of Y appearing in its incident vertices and there is a plane wellconnecting edge set E for $G \cup v$, where v is a new vertex labelled by Y embedded in the interior of the F. Then $AddVertex(Y,d^*)$ is a well connected plane labelled graph with unique vertex label sets, and the operation corresponds to the addition of zone z with label set Y to d yielding d'; i.e. when applicable, this lifts to the operation $AddZone(z,d)$. Let w be a vertex of d^* and let w' be its corresponding zone in d. Then $RemoveVertex(w,G)$ is a well connected plane labelled graph with unique vertex label sets and the operation corresponds to the removal of zone w' from d; this lifts to the operation $RemoveZone(w',d)$.*

Proof. The operations preserve the planarity and connectivity conditions as well as the uniqueness of vertex label sets by construction of the rules. Matching the vertex label sets with the zone sets of the abstract diagram before and after the transformation shows that the operations lift to the abstract transformations. □

Example 6 *In Figure 4, the middle dual graph d_2^* shows the effect of $AddVertex(\{A,B,C\},d_1^*)$ applied to the left hand dual graph d_1^*, using the outer face F and wellconnecting edge set as shown. This corresponds to the addition of the zone $\{A,B,C\}$ to d_1 to give d_2. Since d_1^* is wellconnected, the application of $RemoveVertex(v,G^*)$ to d_2^*, where v is labelled by $\{A,B,C\}$, returns d_1^* corresponding to the removal of the zone $\{A,B,C\}$ of the diagram d_2 giving d_1.*

The incorporation of techniques to alter the dual graph to remove various breaks in well-formedness conditions, or to exchange them, would facilitate the construction of different systems. For instance, in the right hand side of Figure 4, we see the insertion of an extra edge into the dual graph between $\{A\}$ and $\{A,B\}$ surrounding $\{A,C\}$ and $\{A,B,C\}$. Since we obtain ex-

[7] This condition can be relaxed if the system allowed disconnected zones.

actly one vertex in each face when taking a dual, this has the effect of moving the contour C away from the intersection point of A and B in the Euler diagram, thereby removing the tangentiality and the multiple point.

4 Conclusion

Logical reasoning systems based on Euler diagrams are commonly constructed at an abstract level, and the concrete level is merely a visualisation of the abstract level. Then, changes at the abstract level are not consistently reflected by changes at the concrete level that preserve the mental map (i.e. the usual approach is the complete regeneration of a new diagram after a transformation, which does not take account of the diagram prior to the transformation). Building transformation systems at the concrete level which are realisations of the abstract level transformations addresses this concern, enabling the presentation of diagrams that reflect change in a local manner when it is possible to do so. This enables strong control of the topological and geometric properties of the diagrams allowed in order to assist with user comprehension and preferences. In this paper we have viewed concrete diagrams as graphs and provided the first concrete level dual graph transformation system which can be utilised to transform concrete diagrams. Altering the graph theoretic properties of the system will enable the realisation of transformation systems of concrete diagrams satisfying different sets of wellformedness conditions and should facilitate interplay between such systems. The concrete level dual graph transformations preserve the mental map in the sense that will preserve much of the original layout of the diagram (because the diagram is generated from the embedded dual graph and much of the original embedded dual graph is preserved). Furthermore, if one desired systems in which the rules were applicable more often than at the concrete level, but which lose some of the geometric information, then one could build abstract level dual graph transformation systems.

In the future, specialisations of this theory could be used to provide fast computations of restricted classes of diagram transformations. This could be utilised in a graph transformation based library system for generation and manipulation of diagrams for instance, where a collection of initial graphs together with a set of transformation rules is used to attempt to generate a diagram requested by a user, whether that user be human or a system request.

Acknowledgements: Funded by UK EPSRC grant EP/E011160: Visualisation with Euler Diagrams. Thanks to John Taylor for helpful comments on the work, to John Howse for performing a typographic check, but especially to the anonymous reviewers whose detailed reviews helped to improve the paper substantially.

Bibliography

[BT02] S. Bridgeman, R. Tamassia. A User Study in Similarity Measures for Graph Drawing. *Journal of Graph Algorithms and Applications* 6:225–254, 2002.

[Cho07] S. Chow. *Generating and Drawing Area-Proportional Euler and Venn Diagrams*. PhD thesis, University of Victoria, 2007.

[CMP99] S. Chok, K. Marriott, T. Paton. Constraint-Based Diagram Beautification. In *Proceedings of IEEE Symposium on Visual Languages*. Pp. 12–19. IEEE Press, 1999.

[CR03] S. Chow, F. Ruskey. Drawing Area-Proportional Venn and Euler Diagrams. In *Proceedings of Graph Drawing 2003, Perugia, Italy*. LNCS 2912, pp. 466–477. Springer-Verlag, September 2003.

[DES03] R. DeChiara, U. Erra, V. Scarano. VennFS: A Venn Diagram file manager. In *Proceedings of Information Visualisation*. Pp. 120–126. IEEE Computer Society, 2003.

[DF07] R. DeChiara, A. Fish. EulerView: A non-hierarchical visualisation component. In *Proceedings of IEEE Symposium on Visual Languages and Human Centric Computing*. IEEE 134, pp. 145–152. 2007.

[FF08] A. Fish, J. Flower. Euler Diagram Decomposition. In *Proceedings of 5th International Conference on Diagrams 2008*. LNAI 5223, pp. 28–44. Springer-Verlag, 2008.

[FFH05] A. Fish, J. Flower, J. Howse. The Semantics of Augmented Constraint Diagrams. *Journal of Visual Languages and Computing* 16:541–573, 2005.

[FFH08] J. Flower, A. Fish, J. Howse. Euler Diagram Generation. *Journal of Visual Languages and Computing* 19:675–694, 2008.

[FHJT08] A. Fish, J. Howse, C. John, J. Taylor. A normal form for Euler diagrams with shading. In *Proceedings of Diagrams 08*. LNAI 5223, pp. 206–221. Springer, 2008.

[FS06] A. Fish, G. Stapleton. Formal Issues in Languages Based on Closed Curves. In *Proceedings of Distributed Multimedia Systems, International Workshop on Visual Languages and Computings*. Pp. 161–167. Knowledge Systems Institute, Grand Canyon, USA, 2006.

[HES$^+$05] P. Hayes, T. Eskridge, R. Saavedra, T. Reichherzer, M. Mehrotra, D. Bobrovnikoff. Collaborative Knowledge Capture in Ontologies. In *Proceedings of the 3rd International Conference on Knowledge Capture*. Pp. 99–106. 2005.

[HST05] J. Howse, G. Stapleton, J. Taylor. Spider Diagrams. *LMS Journal of Computation and Mathematics* 8:145–194, 2005.

[Ken97] S. Kent. Constraint Diagrams: Visualizing Invariants in Object Oriented Modelling. In *Proceedings of OOPSLA97*. Pp. 327–341. ACM Press, October 1997.

[KMGB05] H. Kestler, A. Muller, T. Gress, M. Buchholz. Generalized Venn Diagrams: A New Method for Visualizing Complex Genetic Set Relations. *Journal of Bioinformatics* 21(8):1592–1595, 2005.

[LLY06] Y. Lee, C. Lin, H. Yen. Mental Map Preserving Graph Drawing Using Simulated Annealing. In *Proceedings of the 2006 Asia-Pacific Symposium on Information Visualisation*. Pp. 179–188. Australian Computer Society, Inc., 2006.

[MELS95] K. Misue, P. Eades, W. Lei, K. Sugiyama. Layout Adjustment and the Mental Map. *Journal of Visual Languages and Computing* 6:183–210, 1995.

[MF94] E. C. M. Egenhofer, P. di Felice. Topological Relations between Regions with Holes. *International Journal of Geographical Information Systems* 8:129–144, 1994.

[MMM08] S. Maier, S. Mazanek, M. Minas. Layout Specification on the Concrete and Abstract Syntax Level of a Diagram Language. In *Proceedings of 2rd International Workshop on the Layout of Software Engineering Diagrams*. ECEASST 13, pp. 40–54. 2008.

[RMF04] P. Rodgers, P. Mutton, J. Flower. Dynamic Euler Diagram Drawing. In *Visual Languages and Human Centric Computing*. Pp. 147–156. IEEE Computer Society Press, 2004.

[RZF08] P. Rodgers, L. Zhang, A. Fish. General Euler Diagram Generation. In *Proceedings of 5th International Conference on Diagrams 2008*. LNAI 5223, pp. 13–27. Springer-Verlag, 2008.

[SHRZ08] G. Stapleton, J. Howse, P. Rodgers, L. Zhang. Generating Euler Diagrams from Existing Layouts. In *Proceedings of 2rd International Workshop on the Layout of Software Engineering Diagrams*. ECEASST 13, pp. 16–31. 2008.

[SMF$^+$07] G. Stapleton, J. Masthoff, J. Flower, A. Fish, J. Southern. Automated Theorem Proving in Euler Diagrams Systems. *Journal of Automated Reasoning* 39(4):431–470, 2007.