# Drawing Euler Diagrams with Circles: The Theory of Piercings

Gem Stapleton, Leishi Zhang, John Howse and Peter Rodgers

**Abstract**—Euler diagrams are effective tools for visualizing set intersections. They have a large number of application areas ranging from statistical data analysis to software engineering. However, the automated generation of Euler diagrams has never been easy: given an abstract description of a required Euler diagram, it is computationally expensive to generate the diagram. Moreover, the generated diagrams represent sets by polygons, sometimes with quite irregular shapes which make the diagrams less comprehensible. In this paper we address these two issues by developing the theory of piercings, where we define single piercing curves and double piercing curves. We prove that if a diagram can be built inductively by successively adding piercing curves under certain constraints then it can be drawn with circles, which are more aesthetically pleasing than arbitrary polygons. The theory of piercings is developed at the abstract level. In addition, we present a Java implementation that, given an inductively pierced abstract description, generates an Euler diagram consisting only of circles within polynomial time.

## 1 Introduction

An Euler diagram is a collection of closed curves that partition the plane into connected subsets, called regions, each of which is enclosed by a set of the curves. Typically, Euler diagrams are used to visualize set-theoretic relationships where each curve in the Euler diagram represents a set and each region represents the intersection of a number of sets. The term 'Euler diagram' is often confused with the term 'Venn diagram' – in fact, the latter can actually be seen as a subclass of Euler diagrams. The reason for this is that whereas Venn diagrams have to represent all possible set intersections, Euler diagrams only need to represent a subset of the possible intersections. For example, the lefthand diagram in Fig. 1 shows an Euler diagram with 3 sets and 5 intersections (including that outside all of the curves) whereas the righthand diagram in Fig. 1 is a both an Euler diagram and a Venn diagram with 3 sets and includes all 8 possible intersections.

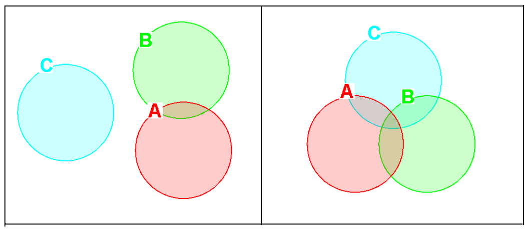

Fig. 1. Euler diagrams.

Euler diagrams are attractive visualization tools because they are able to represent set intersection and enclosure in an easy to understand way. However, despite the benefits as a visualization method, the practical use of Euler diagrams has been held back by the difficulties in their automated generation. All generation approaches start with an abstract description of the diagram to be embedded. Typically, these descriptions state which curves are to be present and which set intersections must be represented. In order to transform abstract descriptions into diagrams effectively, various research efforts have been devoted to the automated generation of Euler diagrams [1], [2], [3], [4].

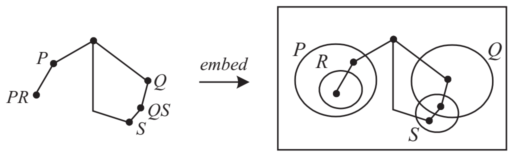

Fig. 2. Generation using a dual graph.

Some existing generation approaches, such as [5], [6], construct a so-called dual graph from the abstract description, which is embedded in the plane, and 'wrap' closed curves around the dual graph, as illustrated in Fig. 2. Each node in the graph represents a required set intersection. For instance, $PR$ represents the set $P \cap R \cap \overline{Q} \cap \overline{S}$ and the node with no label represents $\overline{P} \cap \overline{R} \cap \overline{Q} \cap \overline{S}$. For space reasons we omit the details of these generation approaches.

As the number of required sets and intersections increases, the number of vertices and edges in the dual graph can increase dramatically, with the graph having at most $2^n$ vertices each of which represents a set intersection. Two vertices are joined by an edge whenever the set intersection they represent differs by exactly one set (e.g. nodes for $A \cap B \cap \overline{C}$ and $A \cap \overline{B} \cap \overline{C}$ will be joined by an edge). Generating and manipulating such graphs

• *Gem Stapleton and John Howse are with Visual Modelling Group, University of Brighton, Brighton, UK.*
*E-mail: {g.e.stapleton,john.howse}@brighton.ac.uk*
• *Peter Rodgers and Leishi Zhang are with University of Kent, Kent, UK.*
*E-mail: {p.j.rodgers,l.zhang}@kent.ac.uk*

*Manuscript received September 10, 2008; revised January 11, 2007.*

can involve a huge amount of computation. Stages of the drawing process typically involve finding a large planar subgraph of the dual that has an embedding with certain properties. The subgraph and its embedding are chosen depending on the wellformedness conditions that the to-be-drawn diagram is required to possess. Moreover, the diagrams generated usually represent sets by polygons, sometimes with quite irregular shapes [7], [8], which make the diagrams less comprehensible and not necessarily appealing to users who are familiar with the idea of using circles to represent set-theoretic relationships.

In this paper, we propose a method that is capable of generating a class of Euler diagrams using circles in polynomial time. In part, the polynomial time algorithm exists because the number of set intersections to be represented is constrained to be at most $4 \times (n-1)$, where $n$ is the number of sets. However, this constraint on the number of set intersections alone is not necessarily sufficient to ensure that the aforementioned algorithms run in polynomial time, as we will further discuss below. In order to define our class of Euler diagrams, we identify two types of curves, which we have termed 'single piercing curves' and 'double piercing curves' respectively, at the abstract description level. Second, we show that if a diagram can be drawn by successively adding these piercing curves then it can be drawn with circles in an efficient manner. For simplicity, in the remaining part of this paper, we say a curve which is either a single piercing or a double piercing is a piercing of a diagram. Although we will define the terms single and double piercing later, for now we refer the reader to Fig. 3 which illustrates a diagram that can be built inductively by adding piercing curves. Diagrams that can be generated inductively using piercing curves can be identified at the abstract level and it is at this level that we develop the theory of piercings.

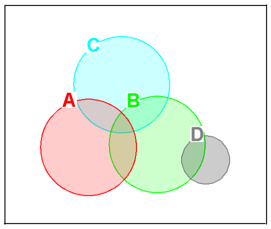

Fig. 3. An inductively pierced diagram.

'Pierced' Euler diagrams can be thought of as being sparse, and are typical of those seen to represent lots of subset and disjointness relationships between sets. This is indicative of the type of situations where Euler diagrams excel at representing information. Particular examples include their application as a basis for software modelling notations, such as class diagrams, state charts, and constraint diagrams [9]; see [10] for an example of a software model produced using constraint diagrams. Indeed, Euler diagrams are suitable for forming the basis of logics which are capable of ontology specification [11], [12]; here, one may often want to specify that the classes (concepts) in the ontology are either disjoint or in a subset relationship. Euler diagrams have a wide range of other application areas such as statistical data analysis and logical reasoning [13], [14], [15], [16], [17], [18], [19].

We start, in section 2, by providing some examples of Euler diagrams that have been automatically drawn using previous generation methods. This allows for comparison with those produced using the methods of this paper. Section 3 overviews the syntax of Euler diagrams and other necessary background material. Abstract descriptions of Euler diagrams are detailed in section 4. Section 5 defines the notions of a single piercing and a double piercing. Inductively pierced descriptions are defined in section 6. We prove that all inductively pierced descriptions can be embedded with circles in section 7. Some limitations of the theory are discussed in section 8. To demonstrate the utility of the theoretical results, section 9 provides an implementation that embeds inductively pierced descriptions as diagrams drawn with circles; the software is freely available from `www.eulerdiagrams.com/piercing.htm`. Many of the diagrams in this paper were generated by the software, including those in Figs. 1 and 3. The complexity of our drawing method is identified in section 10, where we also present some discussions around the complexity of other drawing algorithms. Finally, we conclude in section 11 and discuss future directions for this research.

## 2 Other Drawing Methods

The first generation method, developed by Flower and Howse [5], provides an algorithm that is theoretically capable of drawing an Euler diagram given any abstract description, $D$, provided $D$ has a so-called completely wellformed drawing. The associated software implementation can produce drawings for diagrams with at most 4 curves. An illustration of the output can be seen in Fig. 4. The abstract description for this diagram specifies the labels present, $A, B, C$ and $D$, and the regions (called *zones*) to be present (i.e. which set intersections are to be represented): $\emptyset$ (outside all of the curves), $A$ (inside just $A$), $B$, $AB$ (inside exactly $A$ and $B$), $AC$, $ABC$, $ACD$, and $ABCD$. All of the diagrams in this section have this abstract description to allow for easy comparison (although some use lower case curve labels).

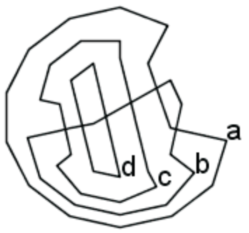

Fig. 4. Generation using the methods of [5].

The techniques of Flower and Howse [5] were extended to enhance the layout [20]. First, some modifications were made to the implementation of the generation method; in our running example this gives the lefthand diagram in Fig. 5, although the labels are not shown, as opposed to the diagram in Fig. 4. Also Flower et al. [20] used layout metrics and hill climbing algorithms to improve the diagrams' aesthetic qualities; the result of the layout improvements applied to the lefthand diagram in Fig. 5 can be seen on the right.

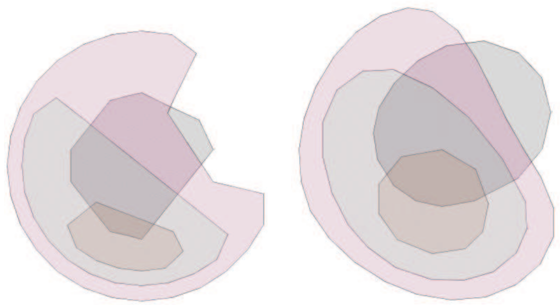
Fig. 5. Using the layout improvement methods of [20].

Further extensions to the generation methods of Flower and Howse allow the drawing of abstract descriptions that need not have a completely wellformed embedding. This was done by Rodgers et al. [21], where techniques to allow the generation of any abstract description were developed; output from the software of Rodgers at al. can be seen in Fig. 6. All of the methods described so far use a dual graph based approach and are computationally complex, having an NP-complete step. This means that some diagrams take a significant time to draw.

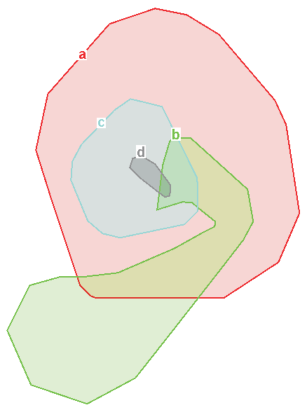

Fig. 6. Generation using the methods of [21].

Indeed, the dual graph method requires one to choose a dual graph from the infinitely many that are capable of generating the required Euler diagram. The chosen graph directly impacts the aesthetic quality of the drawn diagrams and finding a suitable dual can be difficult. A substantial part of Rodgers et al. [21] focusses on the task of finding a dual that minimizes the number of times wellformedness conditions are broken and guarantees the absence of certain conditions (such as no non-simple curve and no 'disconnected' zones). An alternative method for choosing a dual graph is developed by Simonetto and Auber [22], which has been implemented [6]. Output from that implementation can be seen in Fig. 7, where the labels have been manually added post drawing[1].

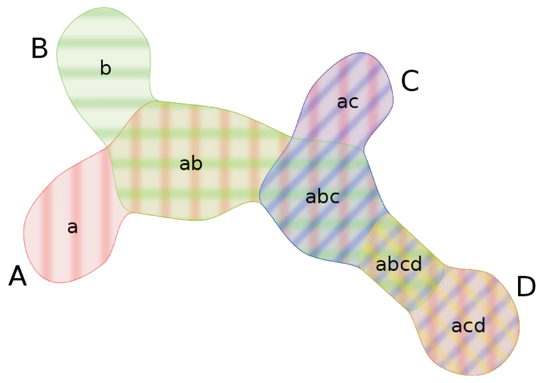

Fig. 7. Generation using the methods of [6].

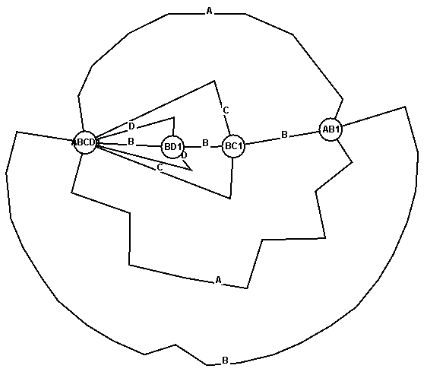

Fig. 8. Inductive generation using the methods of [23].

A different method was developed by Chow [24], that draws so-called *monotone* Euler diagrams. Amongst other restrictions, monotone diagrams must have the intersection between all curves in the to-be-generated Euler diagram being present; such diagrams are called *monotone*. Many 'pierced' diagrams do not have this intersection present so our method is complementary to that of Chow. We do not have access to Chow's software implementation of his generation method, so we refer the reader to `http://apollo.cs.uvic.ca/euler/DrawEuler/index.html` for images of automatically drawn diagrams that can be compared, in terms of aesthetics, with those in this paper.

Most recently, an inductive generation method has been developed [23], which draws Euler diagrams by

1. We thank Paolo Simonetto for supplying this image.

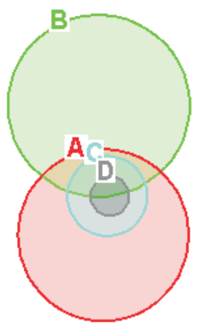

Fig. 9. Generation using the methods of this paper.

adding one curve at a time; see Fig. 8 for an example of the software output. This method has the advantage that it can draw diagrams under arbitrary sets of the wellformedness conditions (where possible) but it also has an NP-complete step since it searches through graphs for cycles with certain properties. The layout metrics of [20] could be applied to the diagrams drawn using this method to improve their aesthetic qualities.

Using the techniques we develop in this paper, the diagram in Fig. 9 can be generated. We identify a class of abstract descriptions that can be drawn quickly, in polynomial time, entirely with circles in a completely wellformed manner.

## 3 Euler Diagrams

An Euler diagram is a collection of closed curves drawn in the plane. Each curve has a label, chosen from a set $\mathcal{L}$. The closed curves essentially provide a partition of the plane into minimal regions. A zone is a union of minimal regions determined by being inside certain curves and outside the other curves. To illustrate, Fig. 10 shows an Euler diagram with three curves $A$, $B$ and $C$. There are seven zones in this diagram. Note that zone $c$ (the regions inside curve $C$ but outside the other two curves) consists of two minimal regions and is, therefore, disconnected. Recall that a closed curve in the plane is a continuous function of the form $c\colon [a,b] \to \mathbb{R}^2$ where $c(a) = c(b)$. Given an arbitrary function, $f\colon A \to B$, we write $image(f)$ to denote the set of elements in $B$ to which $f$ maps.

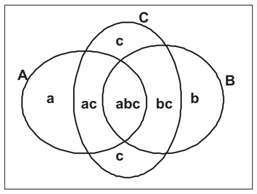

Fig. 10. An Euler diagram.

*Definition 3.1:* An **Euler diagram** is a pair, $d = (Curve, l)$, where

1) $Curve$ is a finite collection of closed curves each with codomain $\mathbb{R}^2$, and
2) $l\colon Curve \to \mathcal{L}$ is an injective function.

*Definition 3.2:* A **minimal region** of an Euler diagram $d = (Curve, l)$ is a connected component of

$$\mathbb{R}^2 - \bigcup_{c \in Curve} image(c).$$

*Definition 3.3:* A **zone** in an Euler diagram $d = (Curve, l)$ is a non-empty set of minimal regions that can be described as being interior to certain curves (possibly no curves) and exterior to the remaining curves.

For the interpretability and classification of diagrams, a range of diagram properties have been defined, which are sometimes called wellformedness conditions. Throughout this paper, we assume the following set of wellformedness conditions:

1) All of the curves are simple.
2) No pair of curves runs concurrently.
3) There are no triple points of intersection between the curves.
4) Whenever two curves intersect, they cross.
5) Each zone is connected (i.e. consists of exactly one minimal region).

To illustrate, Fig. 11 shows some examples of non-wellformed Euler diagrams. Formal definitions of the wellformedness conditions can be found in [25]. Any Euler diagram which satisfies all of the conditions is said to be *completely wellformed*.

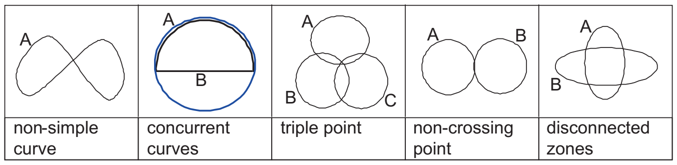

Fig. 11. Non-wellformed diagrams.

## 4 Abstraction of Euler Diagrams

In order to generate an Euler diagram, we start with a description of that diagram. To illustrate, the diagram in Fig. 12 can be described as having four curves, $A$, $B$, $C$, and $D$. These curves divide the plane in such a manner that there are six zones present. For instance, there is one zone inside $A$ only and another zone inside precisely $A$ and $B$. Thus, each present zone can be described by the labels of the curves that the zone is inside.

*Definition 4.1:* An **abstract description**, $D$, is a pair, $(L, Z)$ where $L$ is a subset of $\mathcal{L}$ (i.e. all of the labels in $D$ are chosen from the set $\mathcal{L}$) and $Z \subseteq \mathbb{P}L$ such that $\emptyset \in Z$. Elements of $Z$ are called **abstract zones** (or, simply, zones). Given $D = (L, Z)$, we define $L(D) = L$ and $Z(D) = Z$.

In Fig. 12, the diagram, $d$, has abstract description $L = \{A, B, C, D\}$ and $Z = \{\emptyset, \{A\}, \{B\}, \{A, B\}, \{C\}, \{C, D\}\}$. Sometimes we will write the zones using lower case letters and as strings, such as $ab$ instead of $\{A, B\}$; when writing zones as strings, using lower case distinguishes the zone $a = \{A\}$ from the curve label $A$, for instance.

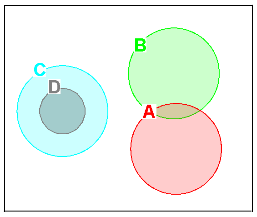

Fig. 12. Abstractions.

*Definition 4.2:* Given an Euler diagram $d = (Curve, l)$, we map $d$ to the abstract description $abstract(d) = (image(l), Z)$, called the **abstraction** of $d$, where $Z$ contains exactly one abstract zone for each zone in $d$; in particular, given a zone, $z$, in $d$, the set $Z$ contains the abstract zone

$$abstract(z) = \{l(c) : c \in C(z)\}$$

where $C(z)$ is the set of curves in $d$ that contain $z$. Given an abstraction, $D = (L, Z)$, if $abstract(d) = D$ then we say $d$ is an **embedding** of $D$.

Note that $\emptyset$ represents the zone that is contained by no curves. It is called the outside zone and is present in every abstract description. The Euler diagram generation problem can be summarized as: given an abstract description, $D = (L, Z)$, find an Euler diagram, $d$, such that $abstract(d) = D$ and such that $d$ satisfies some chosen wellformedness conditions.

## 5 Piercing Curves

A class of abstract descriptions that can be drawn with circles in an efficient manner can be built by successively adding *piercing curves*. We define two types of piercing curves (strictly, curve labels). If an abstract description can be built inductively, entirely of this type of curve, under certain constraints then it can be drawn with circles in polynomial time.

To illustrate, in Fig. 3, the curve labelled $D$ is what we define to be a *single piercing* of $B$. The curve $D$ contains exactly two zones, that inside $D$ and that inside both $B$ and $D$. *Double piercings* pierce two other curves that themselves intersect. So, in Fig. 3, the curve $C$ is a double piercing of $A$ and $B$. We will see later that it is not always the case that double piercings can be drawn with circles and we have to place restrictions on the manner in which we inductively construct our abstract descriptions.

To add a single piercing as a circle, it is necessary is that the two zones through which the circle is to pass are topologically adjacent. Indeed, by the wellformedness conditions, we know that if these two zones are topologically adjacent then they 'separated' by the curve through which the to-be-added single piercing is to pass. In the case of a double piercing, we need a point, $p$, that is an intersection point of the two curves being pierced. In Fig. 3, for example, we can add a double piercing of $A$ and $B$ in $d_2$ by finding a disc around one of their (two) intersection points. Key to our construction is that, if we build a diagram, $d$, by successively adding piercing curves then any pair of zones in $d$ whose abstractions differ only by one label are topologically adjacent. Our results below hold since we are working in $\mathbb{R}^2$ and, under the standard metric, the neighbourhood of a point, $p$, is a disc, so $p$ can easily be enclosed by a circle.

The zones that a piercing curve passes through are strongly related to each other and we call these zones a cluster. The following definition formalizes this concept at the abstract level.

*Definition 5.1:* Let $z$ be an abstract zone and let $L \subseteq \mathcal{L} - z$. The set $\{z \cup L' : L' \in \mathbb{P}L\}$ is an **$L$-cluster** for $z$, denoted $C(z, L)$. Given an $L$-cluster, $C(z, L)$, and a label, $\lambda \in \mathcal{L} - (z \cup L)$, the $L$-cluster $C(z \cup \{\lambda\}, L)$ is called a **$\lambda$-partner** for $C(z, L)$.

For example, given the zone $ab$ (formally, $\{A, B\}$) and the label set $\{C, D\}$, the set

$$C(ab, \{C, D\}) = \{ab, abc, abd, abcd\}$$

is a $\{C, D\}$-cluster for $ab$. Since the label $E$ is not in $ab$ or in $\{C, D\}$, the cluster

$$C(abe, \{C, D\}) = \{abe, abce, abde, abcde\}$$

is an $E$-partner for $C(ab, \{C, D\})$.

Since piercing curves contain specific sets of zones, it is useful to define the set of zones contained by a curve label in an abstract description. We also observe that, in a diagram, some curves properly contain other curves. This concept is also helpful at the abstract level.

*Definition 5.2:* Let $D = (L, Z)$ be an abstract description and let $\lambda_1$ and $\lambda_2$ be distinct curve labels in $L$. If $\lambda_1 \in z$ and $z \in Z$ then we say $\lambda_1$ **contains** $z$ in $D$ with the set of such zones denoted $Z_c(\lambda_1)$. If $Z_c(\lambda_1) \subset Z_c(\lambda_2)$ then $\lambda_2$ **contains** $\lambda_1$ in $D$. The set of curves that contain $\lambda_1$ in $D$ is denoted $L^c(\lambda_1)$.

To compute $L^c(\lambda_1)$, we can make use of the following result.

*Lemma 5.1:* Let $D = (L, Z)$ be an abstract description and let $\lambda$ be a curve label in $L$. Then $L^c(\lambda) = \bigcap_{z_i \in \{z \in Z : \lambda \in z\}} z_i - \{\lambda\}$.

### 5.1 Single Piercings

As illustrated previously, single piercing curves are those curves that intersect with exactly one other curve.

*Definition 5.3:* Let $D = (L, Z)$ be an abstract description and let $\lambda_1, \lambda_2 \in L$ be distinct curve labels. Then $\lambda_1$ is a **single piercing** of $\lambda_2$ in $D$ if there exists a zone, $z$, such that

1) $\lambda_1 \notin z$ and $\lambda_2 \notin z$,
2) $Z_c(\lambda_1) = C(z \cup \{\lambda_1\}, \{\lambda_2\})$, and
3) $C(z, \{\lambda_2\}) \subseteq Z$.

The zone $z$ is said to **identify** $\lambda_1$ as a piercing of $\lambda_2$.

To illustrate, the diagram $d_1$ in Fig. 13 has a single piercing curve labelled $C$. At the abstract level,

$abstract(d_1)$ has zone set $Z = \{\emptyset, a, b, ab, bc, abc\}$. The zone $b$ identifies $C$ as a piercing of $A$, since

$$Z_c(C) = \{bc, abc\} = C(bc, \{A\})$$

and

$$C(b, \{A\}) = \{b, ab\} \subseteq Z.$$

Neither $A$ nor $B$ are single piercing curves in $d_1$ but they are both single piercing curves in $d_2$; removing a single piercing curve can create single piercing curves. The curve $C$ can be thought of as 'piercing' the single curve $A$, hence the terminology 'single piercing'.

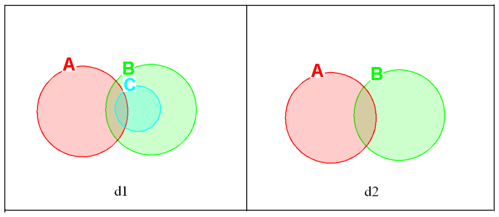

Fig. 13. Identifying single piercings.

### 5.2 Double Piercings

Double piercing curves are curves that pierce two other curves and split four zones. This is a clear generalization of a single piercing curve and one can proceed to define triple piercings and so forth. We further discuss triple piercings below, in the context of generation with circles. Our generation method works for abstract descriptions that are built using single and double piercings but not $n$-piercings, where $n \geq 3$.

*Definition 5.4:* Let $D = (L, Z)$ be an abstract description and let $\lambda_1, \lambda_2, \lambda_3 \in L$ be distinct curve labels. Then $\lambda_1$ is a **double piercing** of $\lambda_2$ and $\lambda_3$ in $D$ if there exists a zone $z$ such that

1) $\lambda_1 \notin z$, $\lambda_2 \notin z$, and $\lambda_3 \notin z$,
2) $Z_c(\lambda_1) = C(z \cup \{\lambda_1\}, \{\lambda_2, \lambda_3\})$, and
3) $C(z, \{\lambda_2, \lambda_3\}) \subseteq Z$.

The zone $z$ is said to **identify** $\lambda_1$ as a piercing of $\lambda_2$ and $\lambda_3$.

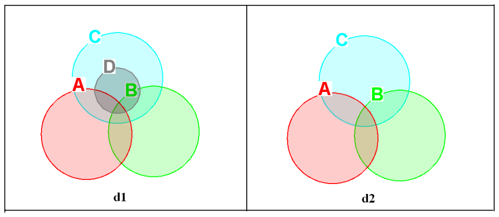

Fig. 14. Identifying double piercings.

To illustrate, the diagram in Fig. 14 has a double piercing curve labelled $D$ and abstract zone set $\{\emptyset, a, ab, b, c, ac, bc, abc, cd, acd, bcd, abcd\}$. The abstract zone $c$ identifies $D$ as a double piercing of $A$ and $B$; we have

$$Z_c(D) = C(cd, \{A, B\})$$

and

$$C(c, \{A, B\}) = \{c, ac, bc, abc\} \subseteq Z.$$

Removing $D$ from $d_1$ creates a new double piercing, $C$, of $A$ and $B$; $C$ was not a piercing in $d_1$. In $d_2$, all of the curves are double piercings and the removal of any one of them turns the remaining curves into single piercings.

## 6 Inductively Pierced Descriptions

There are some diagrams that can be produced by inductively adding piercings. We call them piercing diagrams and the abstract description of this type of diagram is called an inductively pierced description. Fig. 15 illustrates the process, starting with the 'empty' diagram, and successively adding curves. Notice that the first two diagrams do not contain single or double piercing curves. The curve $B$ is a single piercing in $d_2$, we add the double piercing $C$ to give $d_3$, then add $D$ and $E$ to give $d_4$ and $d_5$ respectively. There are different sequences of diagrams that result in $d_5$ by adding piercing curves.

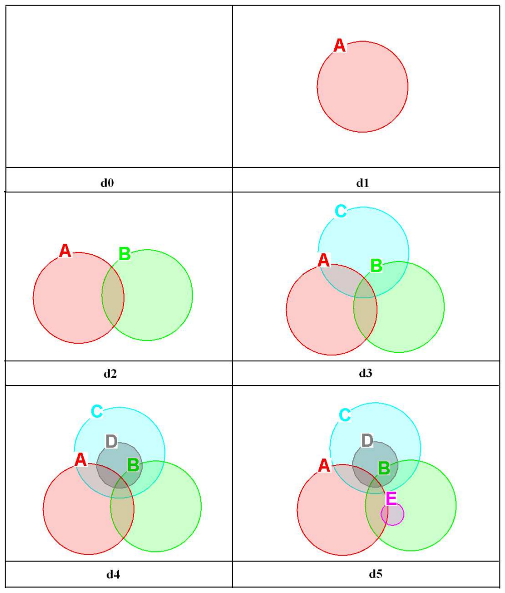

Fig. 15. Inductively adding piercings.

All of the diagrams in Fig. 15 are *connected* since their curves form a connected component of the plane. We can also add curves that are not connected to any other curve, shown in Fig. 16, and then pierce these new curves. Thus, curves that do not intersect with any other curves act as a base case to which we can add piercings; we call these curves *base piercings*.

*Definition 6.1:* Let $D = (L, Z)$ be an abstract description and let $\lambda \in L$ be a curve label. Then $\lambda$ is a **base piercing** in $D$ if there exists a zone, $z$, such that

1) $\lambda \notin z$,
2) $Z_c(\lambda) = C(z \cup \{\lambda\}, \emptyset)$, and
3) $C(z, \emptyset) \subseteq Z$.

The zone $z$ is said to **identify** $\lambda$ as a base piercing.
Note that in the above definition, $C(z \cup \{\lambda\}, \emptyset) = \{z \cup \lambda\}$ and $C(z, \emptyset) = \{z\}$. Whilst this alternative presentation

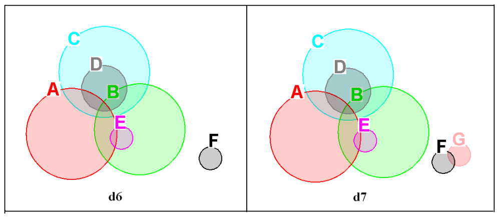

Fig. 16. Disconnected diagrams.

may make the concepts in the definition more immediately apparent, our chosen presentation of the definition readily matches the definitions of single piercings and double piercings.

We are aiming to identify a class of abstract descriptions that can be drawn with circles, since these are aesthetically pleasing, in an efficient manner. If we can build an abstract description, $D$, starting from $(\emptyset, \{\emptyset\})$ by successively adding piercings then it is not necessarily possible to draw $D$ using only circles. Assume that we have an abstract description, $D$, drawn in a completely wellformed manner by successively adding piercings. If we want to add a single piercing, $\lambda$, to obtain some particular abstract description then it is reasonably obvious that we can add a circle labelled $\lambda$ to that drawing and obtain the required abstraction. We note, however, that a single piercing cannot necessarily be added as a circle to an arbitrary diagram, which will be demonstrated below.

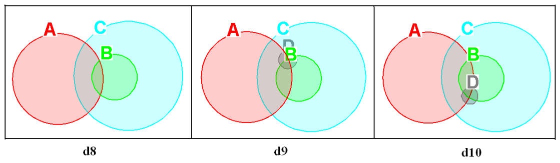

Fig. 17. Choices of embedding.

The case of double piercings is more interesting. We observe that, in any embedding drawn with circles, any pair of curves intersects exactly twice or not at all. If $\lambda_1$ is a double piercing of $\lambda_2$ and $\lambda_3$ then $\lambda_2$ and $\lambda_3$ intersect exactly twice, meaning that if $\lambda_1$ is to be embedded as a circle in a wellformed manner then it has to contain exactly one of these intersection points. Therefore, there is a choice of at most two places where $\lambda_1$ can be embedded. In Fig. 15, there was only one choice for the location of $D$ since it has to be enclosed by $C$. If we are to add more double piercings to $d_5$ then we must ensure that if they pierce $A$ and $B$ and are to be enclosed by $C$ then they must also be enclosed by $D$. Fig. 17 illustrates a scenario where we have two choices for how to add $D$ to $d_8$.

Further, consider the abstract description with labels $L = \{A, B, C, D, E\}$ and zones $Z = \{\emptyset, a, b, ab, c, ac, bc, abc, d, ad, abd, e, ae, abe\}$. This abstraction has three double piercings of $A$ and $B$, namely $C$, $D$, and $E$. Fig. 18 shows how we could embed $A$ and $B$, then add $C$ and $D$ as piercings. It is

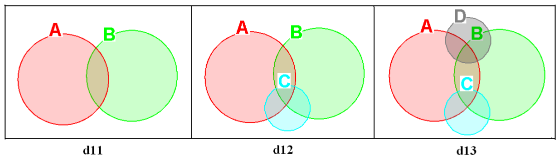

Fig. 18. Limitations.

then obviously not possible to add $E$ as a circle to $d_{14}$. Therefore, we must restrict the manner in which we add double piercings at the abstract level, for which we make use of the following definition.

*Definition 6.2:* Let $C_1 = C(z, \{\lambda_1, \lambda_2\})$ and $C_2 = C(z \cup \{\lambda_3\}, \{\lambda_1, \lambda_2\})$ be partner clusters given $\lambda_3$. Let $D = (L, Z)$ be an abstract description. If $C_1 \cup C_2 \subseteq Z$ then $\lambda_3$ is **outside-associated** with $C_2$ in $D$ and is **inside-associated** with $C_1$ in $D$.

Given a double piercing, $\lambda_3$, of $\lambda_1$ and $\lambda_2$ identified by $z$, $\lambda_3$ is outside-associated with $C(z, \{\lambda_1, \lambda_2\})$ and inside-associated with $C(z \cup \{\lambda_3\}, \{\lambda_1, \lambda_2\})$. The zones in $C(z \cup \{\lambda_3\}, \{\lambda_1, \lambda_2\})$ are all inside $\lambda_3$, hence the terminology 'inside associated'.

Finally, to define an inductively pierced description, we need an operation to remove curve labels from abstractions. Removing curve labels involves two steps: we remove the curve label, $\lambda$, from the label set and then we update the zone list, making sure $\lambda$ is removed from each zone; we transform $D$ into the abstract description that we define to be $D - \lambda$.

*Definition 6.3:* Given an abstract description, $D = (L, Z)$, and $\lambda \in L$, we define $D - \lambda$ to be $D - \lambda = (L - \{\lambda\}, \{z - \{\lambda\} : z \in Z\})$.

*Definition 6.4:* Let $D = (L, Z)$ be an abstract description. Then $D$ is an **inductively pierced description** if either

1) $D = (\emptyset, \{\emptyset\})$, or
2) $D$ has a base piercing, $\lambda_1$, such that $D - \lambda_1$ is inductively pierced, or
3) $D$ has a single piercing, $\lambda_1$, of $\lambda_2$ such that $D - \lambda_1$ is inductively pierced, or
4) $D$ has a double piercing, $\lambda_1$, of $\lambda_2$ and $\lambda_3$ identified by $z$, and either
    a) no other curve label, $\lambda_4$, in $D$ is outside-associated with $C(z, \{\lambda_2, \lambda_3\})$ or
    b) exactly one other curve label, $\lambda_4$, in $D$ is outside-associated with $C(z, \{\lambda_2, \lambda_3\})$ and we have either
        i) $L^c(\lambda_1) = L^c(\lambda_4) = L^c(\lambda_2)$ or
        ii) $L^c(\lambda_1) = L^c(\lambda_4) = L^c(\lambda_3)$.

    and $D - \lambda_1$ is inductively pierced.

Note that given a piercing curve, $\lambda_1$, we have $L^c(\lambda_1) = z$ where $z$ identifies $\lambda_1$. Thus, in 4(b), we seek to establish whether $L^c(\lambda_4) = z$ and whether $L^c(\lambda_2) = z$ or $L^c(\lambda_3) = z$. From lemma 5.1, we can compute $L^c(\lambda_i)$ using $L^c(\lambda_i) = \bigcap_{z_j \in \{z \in Z : \lambda_i \in z\}} z_j - \{\lambda_i\}$.

| $\|L\|$ | IPD | AD | percentage |
|---|---|---|---|
| 0 | 1 | 1 | 100 |
| 1 | 1 | 2 | 50 |
| 2 | 3 | 6 | 50 |
| 3 | 13 | 40 | 33 |
| 4 | 133 | 1992 | 7 |

TABLE 1
The proportion of inductively pierced abstract descriptions.

The space of inductively pierced descriptions represents a relatively small, but not insignificant, fraction of all abstract descriptions. The number of abstract descriptions with label set $L$ is $T = 2^{2^{|L|}} - 1$, since there are $2^{|L|}$ zones and any non-empty set of zones gives rise to an abstract description. In general, the number of inductively pierced descriptions with label set $L$ is bounded above by the number of non-empty subsets of $\mathbb{P}L$ with cardinality at most $4\times(|L|-1)$, since there are at most $4\times(|L|-1)$ zones in such a description: observing that any inductively pierced description containing only one curve label has two zones, and that there is such description containing exactly two curve labels with four zones, the maximum number of zones present when $|L| \geq 2$ is:

$$4\times(|L|-1)$$

since adding a double piercing increases the number of zones by four. For any $|L|$ it is relatively easy to show that there is an inductively pierced description with label set $L$ containing $4\times(|L|-1)$ zones. We can also provide a lower bound on the number of zones present, namely $|L|+1$ since every time we add a piercing curve the number of zones increases by at least one. Again, it is easy to show that there is an inductively pierced description with label set $L$ containing $|L|+1$ zones. Thus, we can give an upper bound on the number of inductively pierced abstract descriptions containing at least two curve labels:

$$UB = \sum_{i=|L|+1}^{4\times(|L|-1)} {}^{2^{|L|}}C_i.$$

However, the number of abstract descriptions and the upper bound placed on the number of inductively pierced descriptions is not hugely insightful. First, there are many abstract descriptions that are isomorphic to each other: $D_1 = (L_1, Z_1)$ is **isomorphic** to $D_2 = (L_2, Z_2)$ if there exists a bijection $\sigma\colon L_1 \to L_2$ that induces a bijection $\gamma\colon Z_1 \to Z_2$ such that for each $z_1 \in Z_1$, $\lambda \in z_1$ if and only if $\sigma(\lambda) \in \gamma(z_1)$. The table below summarizes the number, IPD, of inductively pierced descriptions and the number, AD, of abstract descriptions for fixed label set, containing up to four curve labels reduced up to isomorphism:

## 7 Embedding Inductively Pierced Descriptions

The manner in which we embed inductively pierced descriptions reflects the inductive definition. Given an abstract description that we wish to embed, we start with the empty diagram and successively add curves in the appropriate manner until we obtain a diagram with the required abstraction. In order to specify how to add a curve label $\lambda$ to an abstract description $D-\lambda$ to give $D$ we need to describe $\lambda$'s effect on the zones.

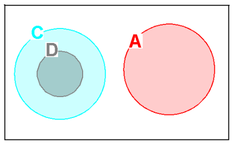

Fig. 19. Adding a curve.

The diagram in Fig. 12 can be obtained from that in Fig. 19, by adding a curve $B$. The abstraction of the diagram in Fig. 19 consists of $L = \{A, C, D\}$ and $Z = \{\emptyset, a, c, cd\}$. To describe how to add $B$, all we need is to know which zone identified $B$ as a piercing and which curves $B$ pierces. In this example, $B$ is identified by the zone $\emptyset$ and pierces $A$. The cluster $C(\emptyset \cup \{B\}, \{A\})$ allows us to create the abstraction of the diagram in Fig. 12 from the abstraction, $D$, of the diagram in Fig. 19: we take $L \cup \{B\}$ and $Z \cup C(\emptyset \cup \{B\}, \{A\})$. We can write this abstraction as $D + (B, C(\emptyset \cup \{B\}, \{A\}))$. For simplicity, we will write $D + B$, when the cluster is clear from the context or not relevant.

So, when we identify a curve as a piercing curve, we store $C(z, L)$ in order to know how to reconstruct $D$ when starting with the empty diagram (which is an embedding of $(\emptyset, \{\emptyset\})$). Our embedding method adds curves successively to drawn Euler diagrams until we obtain an Euler diagram with the required abstraction. For instance, if we want to embed the abstract description $D_5 = (L, Z)$ where $L = \{A, B, C, D, E\}$ and $Z = \{a, b, ab, c, ac, bc, abc, cd, acd, bcd, abcd, be, abe\}$ then our first task is to identify whether $D_5$ has a piercing curve. Here, $E$ is a single piercing of $A$, identified by $z = b$ with $C(z, \{A\}) = \{b, ab\}$; note that $D_5$ is the abstraction of $d_5$ in Fig. 15. Removing $E$ from $D_5$ yields $D_4 = D_5 - E$ with $L(D_4) = \{A, B, C, D\}$ and $Z(D_4) = \{a, b, ab, c, ac, bc, abc, cd, acd, bcd, abcd\}$, the abstraction of $d_4$. Continuing in this manner, we identify $D$ as a piercing of $D_4$, $C$ and a piercing of $D_3 = D_4 - D$, and so forth, until we obtain $(\emptyset, \{\emptyset\})$. Successively removing peircings gives a sequence of abstract descriptions that mirrors our inductive process of adding circles at the drawn diagram level. In this case, that sequence is $\langle D_0, D_1, ..., D_5 \rangle)$.

*Definition 7.1:* Given an abstract description, $D = (L, Z)$, a **pierced decomposition** of $D$ is a sequence, $dec(D) = \langle D_0, D_1, ..., D_n \rangle$ where each $D_{i-1}$ $(0 < i \leq n)$ is obtained from $D_i$ by the removal of some piercing, $\lambda_i$ (so, $D_{i-1} = D_i - \lambda_i$) and $D_n = D$. If $D_0$ contains no labels then $dec(D)$ is a **total pierced decomposition**.

The notion of a decomposition is similar to an abstraction of Euler diagrams developed in [26]. Establishing whether an arbitrary abstraction, $D$, is an inductively pierced description produces, as a bi-product, a total pierced decomposition. Trivially, we have the following lemma.

*Lemma 7.1:* Every inductively pierced description has a total pierced decomposition.

So, the first step in our embedding process is to create a total pierced decomposition. The following theorem allows us to identify whether an abstract description is inductively pierced in a relatively efficient manner. It establishes that the order in which we remove piercings is not important when determining whether a description is inductively pierced.

*Theorem 7.1:* Let $D$ be an inductively pierced description with piercing $\lambda_1$. Then $D - \lambda_1$ is also inductively pierced.

To prove the above theorem, one uses the following lemma.

*Lemma 7.2:* Let $D$ be an inductively pierced description with at least two distinct piercings, $\lambda_1$ and $\lambda_2$. Then $\lambda_2$ is a piercing of $D - \lambda_1$.

We can easily identify some curve labels as not being piercings, using the following lemma. Moreover, we can also quickly identify some descriptions as not being inductively pierced.

*Lemma 7.3:* Let $D = (L, Z)$ be an abstract description such that $L \neq \emptyset$ with $\lambda \in L$.

1) If $|Z_c(\lambda)| \neq 1$ and $|Z_c(\lambda)| \neq 2$ and $|Z_c(\lambda)| \neq 4$ then $\lambda$ is not a piercing curve.
2) If $|Z(D)| > 4 \times (|L(D)| - 1)$ then $D$ is not inductively pierced.
3) If $|Z(D)| < |L(D)|$ then $D$ is not inductively pierced.

We now present a key result of this paper: all inductively pierced descriptions can be embedded in a completely wellformed manner where all curves are circles.

*Theorem 7.2:* Let $D$ be an inductively pierced description. Then there is an embedding of $D$ that possesses all of the wellformedness conditions and all of whose curves are circles.

*Proof:* The strategy to prove this theorem is to use an induction argument. Given a total pierced decomposition, $(D_0, ..., D_n)$, we assume that $D_i = (L_i, Z_i)$ is embedded as $d_i$ in the appropriate manner (i.e. wellformed and with circles) and such that for any pair of zones, $z_1$ and $z_2$, in $d_i$ if their abstract descriptions have a symmetric difference containing exactly one label then they are topologically adjacent. In any wellformed diagram (in particular, $d_i$), we also have the property that any pair of zones that are topologically adjacent have exactly one label in the symmetric difference of their abstractions.

Consider, then, $D_{i+1}$ which is obtainable from $D_i$ by adding a piercing, $\lambda_i$. We know, therefore, that $D_{i+1} = D_i + (\lambda_i, C(z, \hat{L}))$, for some $\hat{L} \subseteq L_i$ and zone $z \in Z_i$. We can show it is possible to add a circle to $d_i$ to give $d_{i+1} = d_i + \lambda_i$ in such a manner that

1) $abstract(d_i + \lambda_i) = D_{i+1}$,
2) $d_i + \lambda_i$ is wellformed, and
3) any pair of zones in $d_i + \lambda_i$ whose abstractions have a symmetric difference that contains exactly one label are topologically adjacent.

Clearly, if $\lambda_i$ is a base or single piercing, this is trivial given the topological adjacency property. Thus, we sketch the argument where $\lambda_i$ is a double piercing.

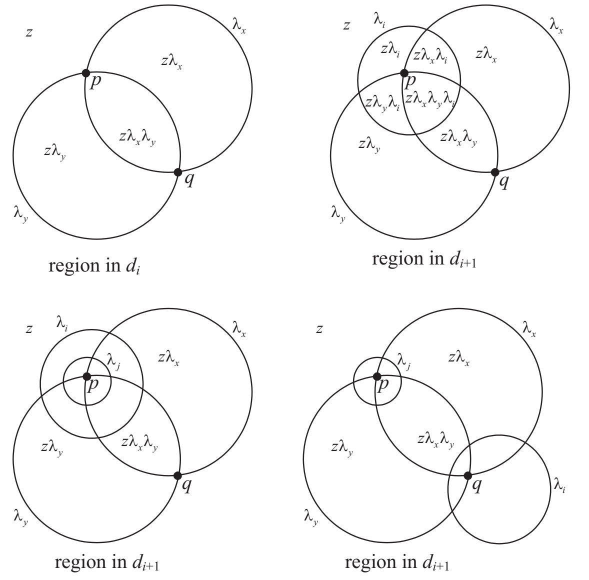

Fig. 20. Drawing with circles.

Suppose that $\lambda_i$ pierces $\lambda_x$ and $\lambda_y$ identified by $z$. Then, in $d_i$, the curves labelled $\lambda_x$ and $\lambda_y$ are drawn with circles and overlap (like Venn-2, the Venn diagram with two curves); see the representative region in $d_i$ in Fig. 20 which includes $\lambda_x$ and $\lambda_y$ but may omit curves in $d_i$ that also pass through the illustrated region. Clearly, we can add a curve labelled $\lambda_i$ to $d_i$, as shown in the representative region of $d_i + \lambda_i$, by our assumption about topological adjacency. However, we need to ensure that the curve labelled $\lambda_i$ only intersects $\lambda_x$ and $\lambda_y$ and does not contain any other curves in $d_i + \lambda_i$, so that we get the correct abstraction (i.e. $abstract(d_i + \lambda_i) = D_i + \lambda_i$); since $d_i$ is embedded using only circles, this is the only way the curve labelled $\lambda_i$ can be embedded in such a manner that we have the wrong abstraction.

We assume that $d_i + \lambda_i$ has the wrong abstraction. We further assume, without loss of generality, that the curve labelled $\lambda_i$ does not contain any curves that are base piercings or are single peircings of $\lambda_x$ or $\lambda_y$ (we can always route $\lambda_i$ to achieve this, given topological adjacency and wellformedness). Similarly, we can always route $\lambda_i$ through $d_i$ in such a manner that it only intersects with $\lambda_x$ and $\lambda_y$. Hence, the curve labelled $\lambda_i$ contains another curve because we have the wrong abstraction. Therefore, it must contain at least one curve labelled, say, $\lambda_j$ that is outside-associated with

$C(z, \{\lambda_x, \lambda_y\})$ in $D_i$ since $d_i$ is a wellformed embedding of $D_i$.

Since $d_i + \lambda_i$ has the wrong abstraction, $\lambda_j$ is also outside associated with $C(z, \{\lambda_x, \lambda_y\})$ in $D_i + \lambda_i$. We know that $D_{i+1}$ is inductively pierced, so there is no other curve label in $D_{i+1}$ that is outside associated with $C(z, \{\lambda_x, \lambda_y\})$. Moreover, if an additional curve label, $\lambda_k$ was outside associated with $C(z, \{\lambda_x, \lambda_y\})$ in $D_i$ then it to would be outside associated with $C(z, \{\lambda_x, \lambda_y\})$ in $D_i + \lambda_i$. This implies that in $D_i$, $\lambda_j$ is the only curve label that is outside associated with $C(z, \{\lambda_x, \lambda_y\})$.

We know, again without loss of generality, from the definition of an inductively pierced description that $L^c(\lambda_i) = L^c(\lambda_j) = L^c(\lambda_x)$ in $D_i + \lambda_i$. Therefore, in $D_i$, $L^c(\lambda_j) = L^c(\lambda_x)$. It then follows that the four zones around the point $q$ are exactly those in $C(z, \{\lambda_x, \lambda_y\})$ and we can, therefore, draw a circle around $q$ labelled $\lambda_i$ to give $d_i + \lambda_i$ with abstraction $D_i + \lambda_i$. Hence $D_i + \lambda_i$ can be embedded using circles. Trivially, we can draw the circle labelled $\lambda_i$ sufficiently small to ensure wellformedness. Finally, it is also trivial that the constructed embedding ensures that zones whose abstractions have a one label symmetric difference are topologically adjacent, and the result then follows by induction. $\square$

We note that, in the above proof, we argued that we could draw a circle sufficiently small in order to add it in the correct manner; this was to ensure that the circle, $c$, was enclosed by the correct set, $C$, of other circles. This need to draw curves with a small area inside them is not particular to using circles to represent the sets. When we want to draw one curve inside another, it is necessary that it has a smaller area inside it. This feature of Euler diagrams may make them difficult to read at a single scale.

In the definition of an inductively pierced description, the assertion in condition 4 that at most one other curve label $\lambda_4$ exists (satisfying the specified properties in 4(b)) is necessary for ensuring that we can draw an appropriate diagram with circles.

*Theorem 7.3:* Let $D$ be an abstract description with double piercing, $\lambda_1$, of $\lambda_2$ and $\lambda_3$ given $z$. Suppose that $D - \lambda_1$ is inductively pierced and that there exists two distinct curve labels, $\lambda_4$ and $\lambda_5$, that are outside-associated with $C(z, \{\lambda_2, \lambda_3\})$ in $D - \lambda_1$ and for each $i \in \{4, 5\}$, either

1) $L^c(\lambda_1) = L^c(\lambda_i) = L^c(\lambda_2)$ or
2) $L^c(\lambda_1) = L^c(\lambda_i) = L^c(\lambda_3)$.

Then $D$ cannot be drawn with circles.

The strategy for the proof is to take any embedding of $D - \lambda_1$ drawn with circles, show that $\lambda_4$ encloses one point where $\lambda_2$ and $\lambda_3$ intersect and that $\lambda_5$ encloses the other point. It then follows that $\lambda_1$ cannot be drawn as a circle to give a diagram with abstraction $D$.

## 8 Limitations

We have already seen that we cannot necessarily draw double piercings as circles, unless we are considering an abstract description that is inductively pierced. It is natural to ask whether, if an abstract description has a single piercing, that piercing can be drawn as a circle. In general, the answer to this is no. Fig. 21 shows an example: if we want to add a single piercing, $E$, of $C$ identified by zone $ab$ then we cannot do so using a circle. There are alternative (not wellformed) embeddings of $abstract(d)$ that allow us to draw $E$ as a circle. This example proves the following lemma.

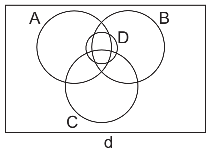

Fig. 21. Single piercings cannot always be drawn as circles.

*Lemma 8.1:* Let $D$ be an abstract description with a single piercing curve $\lambda$. Then it is not necessarily the case that a curve labelled $\lambda$ can be added as a circle to an embedding of $D - \lambda$ to give an embedding of $D$.

Perhaps the most obvious question is whether this work extends to the case of triple piercings. A triple piercing, $\lambda_1$, of $\lambda_2$, $\lambda_3$ and $\lambda_4$ identified by $z$ contains exactly the zones in $C(z \cup \{\lambda_1\}, \{\lambda_2, \lambda_3, \lambda_4\})$. To illustrate, $D$ is a triple piercing of $\{\emptyset, a, b, ab, c, ac, bc, abc, d, ad, bd, abd, cd, acd, bcd, abcd\}$. Visually, if we were to draw this abstract description using circles, we would need to add $D$ to a diagram like $d_{12}$ in Fig. 18. Intuitively, $D$ cannot be added as a circle and we cannot extend the results to allow triple piercings and still draw the diagrams with circles.

Finally, whilst we have identified a significant class of abstract descriptions that can be embedded using circles in an efficient manner, there are still many descriptions that can be drawn with circles that do not have inductively pierced descriptions, such as that in Fig. 21. Future work will extend the techniques developed in this paper to identify a larger class of abstract descriptions that can be embedded with circles.

## 9 Implementation

We have implemented a tool that allows the automated embedding of inductively pierced abstract descriptions, including functionality to identify whether a description has this property. It is relatively straightforward to establish whether any given abstract description, $D = (L, Z)$ is inductively pierced. As stated previously, if $D$ is inductively pierced then establishing this produces a total pierced decomposition, $dec(D) = (D_0, ..., D_n)$. When we generate $dec(D)$, we also create a list of piercings $lp = (\lambda_0, ..., \lambda_{n-1})$, where $D_i + \lambda_i = D_{i+1}$, and a list of clusters, $lc = (C(z_o, \hat{L}_0), ..., C(z_{n-1}, \hat{L}_{n-1}))$, where $\lambda_i$ pierces the curves in $\hat{L}_i$ identified by $z_i$. The two lists $lp$ and $lc$ completely determine $dec(D)$ and we use these two lists to construct our embedding.

Once an abstract description, $D$, is confirmed to be an inductively pierced description, the list of piercings, $lp$, and list of clusters, $lc$, can be used to generate the layout. Since we are building up our diagram inductively, at each key stage in the embedding process we need to add a circle, $C_i$, that is to be labelled $\lambda_i$. The program will establish what type of piercing is $\lambda_i$. For each $\lambda_i$, it is either contained by other circles that have already been drawn or not contained by any other circles.

We first sketch the case of how to add $C_i$ when $\lambda_i$ is not contained by any other curves. If $\lambda_i$ is a base piercing, the program will calculate the occupied region in the current drawing (see the dotted rectangle area in Fig. 22) and place the curve $C_i$ outside that region. The size of the curve is determined by the number of curves with which $C_i$ is to intersect in an embedding of $D$; the more intersections, the bigger we make the curve, to ensure that the still to be added curves are not too small in the final diagram.

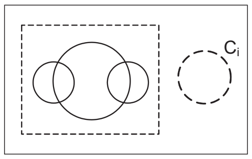

Fig. 22. Adding a base piercing curve that is inside no other curves.

Suppose now that $C_i$ is a single piercing of $C_p$ in the current diagram (i.e. the embedding of $D_i$). If $C_i$ is not fully contained by any other curves, the method will first find the occupied sectors of curve $C_p$ and generate a list of available sectors (see the grey areas in Fig. 23 as an example). The algorithm will then find the most suitable available sector in which to fit the curve. The most suitable sector is selected by measuring the "size" of each available sector (see the dotted line in Fig. 23 as an example) and finding one whose size is closest to twice the desired radius, $r_i$, of $C_i$. Once the most suitable sector is found, the algorithm will fit the new curve. The size and centre of the new curve will be adjusted to make sure the resulting diagram has abstraction $D_i + \lambda_i$.

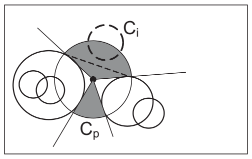

Fig. 23. Adding a single piercing curve that is inside no other curves.

If $C_i$ is a double piercing curve, the algorithm will find the two curves that $C_i$ pierces, say $C_{p1}$ and $C_{p2}$. It calculates the two intersection points of $C_{p1}$ and $C_{p2}$ and selects a valid one as the centre of $C_i$. An invalid intersection point is detectable by checking whether that point has been used as the centre of another dual piercing, $C_x$, of $C_{p1}$ and $C_{p2}$ and, if such a $C_x$ exists, whether $C_x$ is to contain $C_i$. Once a valid intersection point is found, it can be used as the centre of the new curve $C_i$. The radius of the curve can be determined by the area which is already marked as available between $C_{p1}$ and $C_{p2}$, (see the grey area of Fig. 24 as an example). Again, the radius of $C_i$ will be adjusted to make sure the adding of new curve results in a diagram with abstraction $D_i + \lambda_i$.

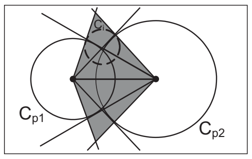

Fig. 24. Adding a double piercing curve that is inside no other curves.

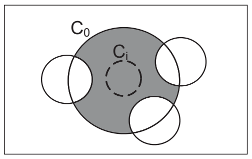

Fig. 25. Adding a base piercing curve inside other curves.

Suppose now that $C_i$ is contained by some other curve in $D_i + \lambda_i$. If $C_i$ is fully contained by other curves, the method will find the inner most outside curve of $C_i$, say $C_0$, and fit $C_i$ inside $C_0$. If $C_0$ is a base piercing, the program will calculate the unoccupied area (see the grey area in Fig. 25) in $C_0$ and then fit $C_i$ inside this area. If $C_i$ is a single piercing curve, the algorithm will calculate the sector in curve $C_p$ which is occupied by $C_0$ (see Fig. 26). A list of available sub-sectors will then be generated (see the grey sectors in Fig. 26). Similar to adding single piercing curves that are not contained by other curves, $C_i$ will be fitted to the most suitable sector and the radius will be adjusted to generate a valid layout. If $C_i$ is a double piercing curve, the method will find the two curves that $C_i$ pierces, say, $C_{p1}$ and $C_{p2}$. The two intersection points of the two curves will be calculated and one that is inside curve $C_0$ will be used as the centre of $C_i$; see Fig. 27.

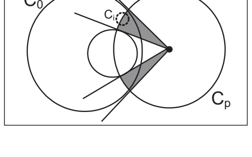

Fig. 26. Adding a single piercing curve inside other curves

To prototype the generation mechanism, we have implemented the method as a Java program, available from `www.eulerdiagrams.com/piercing.htm`. The program takes an abstract description of an Euler diagram as input and checks whether the abstract description is

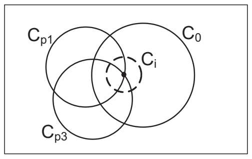

Fig. 27. Adding a double piercing curve inside other curves

inductively pierced. Once the program confirms that an abstract description is inductively pierced, it will draw the diagram by inductively fitting circles to available spaces in the reverse order of the sequence of removal. Figs. 1, 3, 9, 12, 13, 14, 15, 16, 17, 18, and 19 were all drawn using our software. Fig. 28 shows some nice automatically generated layouts for Euler diagrams containing many curves. Fig 29 shows a diagram containing 52 circles, which took 6.641 seconds to draw (Intel Pentium CPU with 2GB RAM, under Windows XP operation system, Java Version 1.6.0_03 from Sun Microsystems Inc.). As one can see in this diagram, the circles are sometimes a little small.

The method generates the basic drawing of a piercing diagram with circles. The layout may not be optimal but various methods can be applied to adjust the size of circles after the initial drawing. In future work, we plan to develop force directed methods that will move the circles' centre points and will alter the radii to improve the layout whilst maintaining the abstract description.

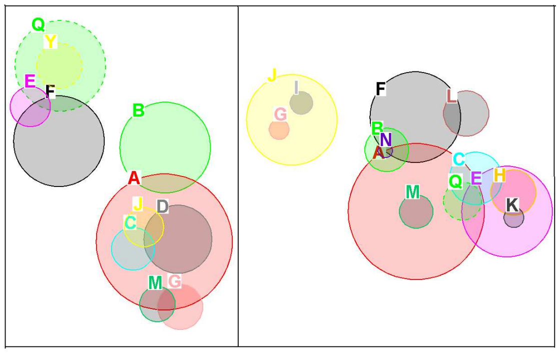

Fig. 28. Output from the software.

## 10 Complexity

The class of inductively pierced descriptions can be drawn efficiently, even under the requirement that the diagrams produced are completely wellformed. A naive algorithm to identify whether an abstract description, $D$, is inductively pierced first checks the cardinality of the zone set: $D$ is not inductively pierced if $|Z(D)| > 4 \times (|L(D)| - 1)$. If the zone set is small enough then for each curve label, $\lambda$, we determine whether it is a piercing curve. As soon as we identify a piercing curve, we check whether the conditions of definition 6.4 are satisfied; in the worst case, this takes $|L(D)| \times |Z(D)|$ which is $O(|L(D)|^2)$, since $|Z(D)| \leq 4 \times (|L(D)| - 1)$. Again, in the worst case, we have to iterate through all the curve labels in $L(D)$ performing this process. Hence the time complexity of this naive algorithm is $O(|L(D)|^3)$. The generation process is efficient because the program only needs to calculate the radius and centre of each circle; the embedding stage (i.e. after we have produced a decomposition) of the implemented algorithm is of $O(|L(D)|^2)$. Therefore, the time complexity of the entire generation algorithm is that of the process of seeking a decomposition, $O(|L(D)|^3)$.

We note that there are trivial, highly efficient, methods of drawing any abstract description but they pay no regard to wellformedness; for example, draw one circle for each zone, join these circles at a single point (creating a 'wedge of circles'), then traverse the circles to form each curve [27]. It is perhaps natural to ask whether other drawing methods that produce completely wellformed diagrams have similar efficient properties to our method, given the restriction to the class of inductively pierced descriptions. Thus, we will consider in more detail the dual graph based method of Flower and Howse [5], since this method guarantees to draw a wellformed diagram whenever this is possible; this drawing process was illustrated in figure 2.

Given an abstract description, $D = (L, Z)$, the first stage of the Flower and Howse method constructs the so-called superdual: the **superdual** of $D$ is a graph, $G$, with vertex set $Z$ and with an edge, $e$, between two zones (i.e. vertices) whenever the symmetric difference of the zones contains exactly one curve label. The next stage is to find a subgraph of the superdual that is planar, *well-connected* and has an embedding which passes the *face conditions*; we do not have space to formally define these conditions for space reasons. The superdual has a planar subgraph that possess these two conditions if and only if the abstract description $D$ has a completely wellformed embedding. Moreover, Flower and Howse use such a subgraph to produce a completely wellformed embedding. Checking whether an arbitrary abstract description has a superdual with such a subgraph is known to be NP-complete.

If we consider only inductively pierced descriptions then it is relatively easy to establish that the superdual is planar, well-connected, and that any embedding of it passes the face conditions. Hence the Flower and Howse method when applied to inductively pierced description does not have this NP-complete step. Therefore, the complexity of this method is dependent on (a) the algorithm used to find a plane embedding of the superdual, and (b) the algorithm used to construct the curves of the diagram given the embedding of the superdual. There are known polynomial time algorithms to construct plane embeddings of planar graphs, so (a) is of polynomial complexity.

Regarding (b), whether the algorithm used to draw the curves as presented in [5] is of comparable complexity to our drawing method (i.e. at most $O(|L(D)|^2)$, giving an overall complexity of $O(|L(D)|^3)$) or has worse time

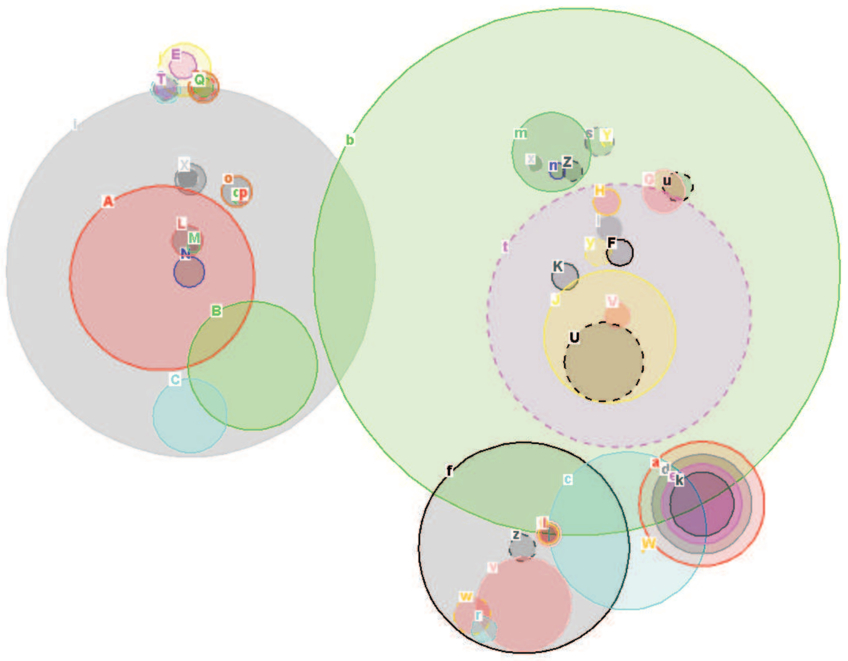

Fig. 29. A diagram containing 52 circles.

complexity is unknown. The method presented in [3] for drawing the curves from a suitable subgraph of the superdual is polynomial. However, typically the curves are not circles, since the layout of superdual impacts the possible routings for the curves.

It may also be natural to ask, if we consider the larger class of abstract descriptions consisting of all those whose number of zones is polynomial in the number of labels, whether the drawing method of Flower and Howse is of polynomial time complexity. We suspect that the answer to this may well be no: it is easy to show that there are abstract descriptions whose number of zones is *linear* (e.g. $4 \times (|L(D)| - 1)$) in the number of zones where the superdual is non-planar and, therefore, a large planar subgraph must be found which is well connected and has an embedding which passes the face conditions. Known algorithms to find well-connected planar subgraphs are exponential and the potential need to consider all different plane embeddings of each such subgraph cannot be overlooked.

## 11 Conclusion

In this paper, we have identified a class of abstract descriptions of Euler diagrams that can be drawn efficiently with circles. Using circles brings an aesthetic quality to the automatically generated diagrams. The implemented software that we have developed, which is freely available from `www.eulerdiagrams.com/piercing.htm`, demonstrates the practical utility of the research. The many areas in which Euler diagrams can be, and are, used to visualize information serves to demonstrate the significance of the work.

Our results improve on previous contributions in a number of ways. Existing generation approaches tend to embed Euler diagrams using polygons, sometimes with quite irregular shapes. Moreover, generating an embedding of an Euler diagram from an arbitrary abstract description can be very computationally expensive: the complexity of the computation often grows exponentially as the number of curves in the diagram increases. We have identified a class of inductively pierced descriptions can be drawn in completely wellformed manner in polynomial time.

In the future, we plan to investigate an amalgamation of embedding methods. A goal is to be able to identify a 'maximally pierced sub-description' of an abstract description. If we are able to do this, then we can embed that sub-description using the method presented in this paper and then use known techniques to add the remaining curves to the diagram [8], [23]. To illustrate, if we want to embed the abstraction $D$ with zones $Z(D) = \{\emptyset, a, b, ab, c, ac, bc, abc, ad, acd, e, be, ce, bce, abe, abce\}$, we can remove $E$, giving $D - E$ which is inductively pierced, embed $D - E$ and then add $E$, as shown in Fig. 30.

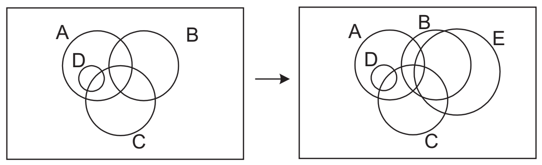

Fig. 30. Embedding maximal sub-diagrams.

## ACKNOWLEDGMENTS

This work is supported by the UK EPSRC grants EP/E011160/1 and EP/E010393/1 for the Visualization with Euler Diagrams project. We thank the anonymous reviewers for their very helpful comments which lead to an improved version of this paper.

## REFERENCES

[1] S. Chow and F. Ruskey, "Drawing area-proportional Venn and Euler diagrams," in *Proceedings of Graph Drawing 2003, Perugia, Italy*, ser. LNCS, vol. 2912. Springer-Verlag, September 2003, pp. 466–477.

[2] J. Flower, J. Howse, and J. Taylor, "Nesting in Euler diagrams: syntax, semantics and construction," *Software and Systems Modelling*, vol. 3, pp. 55–67, March 2004.

[3] P. Rodgers, L. Zhang, G. Stapleton, and A. Fish, "Embedding wellformed Euler diagrams," in *12th International Conference on Information Visualization*. IEEE, 2008, pp. 585–593.

[4] A. Verroust and M.-L. Viaud, "Ensuring the drawability of Euler diagrams for up to eight sets," in *Proceedings of 3rd International Conference on the Theory and Application of Diagrams*, ser. LNAI, vol. 2980. Cambridge, UK: Springer, 2004, pp. 128–141.

[5] J. Flower and J. Howse, "Generating Euler diagrams," in *Proceedings of 2nd International Conference on the Theory and Application of Diagrams*. Georgia, USA: Springer, April 2002, pp. 61–75.

[6] P. Simonetto, D. Auber, and D. Archambault, "Fully automatic visualisation of overlapping sets," *Computer Graphics Forum*, vol. 28, no. 3, 2009.

[7] J. Flower, A. Fish, and J. Howse, "Euler diagram generation," *Journal of Visual Languages and Computing*, available online, 2008.

[8] G. Stapleton, J. Howse, P. Rodgers, and L. Zhang, "Generating Euler diagrams from existing layouts," in *Layout of (Software) Engineering Diagrams*. ECEASST, 2008.

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

[10] S.-K. Kim and D. Carrington, "Visualization of formal specifications," in *6th Aisa Pacific Software Engineering Conference*. Los Alamitos, CA, USA: IEEE Computer Society Press, 1999, pp. 102–109.

[11] I. Oliver, J. Howse, G. Stapleton, E. Nuttila, and S. Törmä, "Expressing ontologies using diagrammatic logics," in *International Semantic Web Conference (Posters and Demos)*. http://kcap09.stanford.edu/share/posterDemos.

[12] 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*, 2003, pp. 101–112.

[13] P. Artes and B. Chauhan, "Longitudinal changes in the visual field and optic disc in glaucoma," *Progress in Retinal and Eye Research*, vol. 24, no. 3, pp. 333–354, 2005.

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

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

[16] H. Kestler, A. Muller, J. Kraus, M. Buchholz, T. Gress, H. L. abd D. Kane, B. Zeeberg, and J. Weinstein, "Vennmaster: Area-proportional Euler diagrams for functional go analysis of microarrays," *BMC Bioinformatics*, vol. 9, no. 67, 2008.

[17] T. Quick, C. Nehaniv, K. Dautenhahn, and G. Roberts, "Sensorimotor information flow in genetic regulatory network driven control systems," University College London, Tech. Rep. Research Note RN/05/29.

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

[19] N. Swoboda and G. Allwein, "Using DAG transformations to verify Euler/Venn homogeneous and Euler/Venn FOL heterogeneous rules of inference," *Journal on Software and System Modeling*, vol. 3, no. 2, pp. 136–149, 2004.

[20] J. Flower, P. Rodgers, and P. Mutton, "Layout metrics for Euler diagrams," in *7th International Conference on Information Visualisation*. IEEE Computer Society Press, 2003, pp. 272–280.

[21] P. Rodgers, L. Zhang, and A. Fish, "General Euler diagram generation," in *International Conference on the Theory and Application of Diagrams*. Springer, September 2008.

[22] P. Simonetto and D. Auber, "An heuristic for the construction of intersection graphs," in *13th International Conference on Information Visualisation*, 2009.

[23] G. Stapleton, P. Rodgers, J. Howse, and L. Zhang, "Inductively generating Euler diagrams," *Accepted for IEEE Transactions of Visualization and Computer Graphics*, 2009.

[24] S. Chow, "Generating and drawing area-proportional Euler and Venn diagrams," Ph.D. dissertation, University of Victoria, 2007.

[25] G. Stapleton, P. Rodgers, J. Howse, and J. Taylor, "Properties of Euler diagrams," in *Proceedings of Layout of Software Engineering Diagrams*. EASST, 2007, pp. 2–16.

[26] A. Fish and J. Flower, "Abstractions of Euler diagrams," in *Proceedings of Euler Diagrams 2004, Brighton, UK*, ser. ENTCS, vol. 134, 2005, pp. 77–101.

[27] A. Fish and G. Stapleton, "Formal issues in languages based on closed curves," in *Proceedings of Distributed Multimedia Systems, International Workshop on Visual Languages and Computings*. Grand Canyon, USA: Knowledge Systems Institute, 2006, pp. 161–167.

**Gem Stapleton** is a Senior Research Fellow, with interests including the theory of diagrammatic logics and developing automated diagram layout techniques. She received the Best Paper Award at Diagrams 2004, was Runner-Up for the British Computer Society Distinguished Dissertation Award 2005, and was the only UK Finalist for the Cor Baayen Award 2006, presented by ERCIM to the most promising young researcher in Computer Science and Applied Mathematics. She was General Chair of Diagrams 2008.

**Leishi Zhang** is a Research Associate with a PhD in bioinformatics visualization from the University of Brunel (funded by the EPSRC) and an MSc in computer science from University of Dundee. Her main research interests include information visualization, graph theory, artificial intelligence and data analysis. She has published her research in a number of international journals and conferences relating to the area of data analysis and visualization.

**John Howse** is Professor of Mathematics and Computation and he is leader of the Visual Modelling Research Group. His main research interests are diagrammatic reasoning and the development of visual modelling languages. He is on the program committee for several international conferences, was General Chair of Visual Languages and Human-Centric Computing 2006, is on the steering committee for the Diagrams conference series and was Program Chair for Diagrams 2008. He received the Best Paper Award at Diagrams 2002.

**Peter Rodgers** is a Senior Lecturer and his main research interests are in diagrammatic visualization, including graph and Euler diagram layout techniques. He has led several research projects supported by national and international funding bodies. He sits on the program committee of various international conferences.