Generating and Drawing Area-Proportional Euler and Venn Diagrams

by

Stirling Christopher Chow
B.Sc., University of Victoria, 1997

A Dissertation Submitted in Partial Fulfillment of the
Requirements for the Degree of

Doctor of Philosophy

in the Department of Computer Science

Generating and Drawing Area-Proportional Euler and Venn Diagrams

by

Stirling Christopher Chow
B.Sc., University of Victoria, 1997

Supervisory Committee

Dr. Frank Ruskey, (Department of Computer Science)

Supervisor

Dr. John A. Ellis, (Department of Computer Science)

Departmental Member

Dr. Margaret-Anne Storey, (Department of Computer Science)

Departmental Member

Dr. Gary MacGillivray, (Department of Mathematics and Statistics)

Outside Member

Supervisory Committee

Dr. Frank Ruskey
Supervisor

Dr. John A. Ellis
Departmental Member

Dr. Margaret-Anne Storey
Departmental Member

Dr. Gary MacGillivray
Outside Member

Abstract

An Euler diagram $C = \{c_1, c_2, \ldots, c_n\}$ is a collection of n simple closed curves (i.e., Jordan curves) that partition the plane into connected subsets, called regions, each of which is enclosed by a *unique* combination of curves. Typically, Euler diagrams are used to visualize the distribution of discrete characteristics across a sample population; in this case, each curve represents a characteristic and each region represents the sub-population possessing *exactly* the combination of containing curves' properties. Venn diagrams are a subclass of Euler diagrams in which there are 2^n regions representing *all* possible combinations of curves (e.g., two partially overlapping circles). In this dissertation, we study the Euler Diagram Generation Problem (EDGP), which involves constructing an Euler diagram with a prescribed set of regions. We describe a graph-theoretic model of an Euler diagram's structure and use this model to develop necessary-and-sufficient existence conditions. We also use the graph-theoretic model to prove that the EDGP is NP-complete. In addition, we study the related Area-Proportional Euler Diagram Generation Problem (ω-EDGP), which involves constructing an Euler diagram with a prescribed set of regions, each of which has a prescribed area. We develop algorithms for constructing area-proportional Euler diagrams composed of up to three circles and rectangles, as well as diagrams with an unbounded number of curves and a region of common intersection. Finally, we present implementations of our algorithms that allow the dynamic manipulation and real-time construction of area-proportional Euler diagrams.

Table of Contents

List of Tables

List of Figures

Acknowledgements

Researching and writing this dissertation has been a challenging, but very rewarding task. My time at the University would not have been as enjoyable or as great a learning experience without the constant support and guidance of my supervisor, Dr. Frank Ruskey. I would also like to thank Dr. Peter Rodgers and the other RWD project researchers for inviting me to England and sharing with me their research and ideas. In addition, the financial support provided to me through NSERC's Canada Graduate Scholarship program made it much easier to concentrate on my research. Of course, there is life outside of research, and I am grateful that as my research has evolved, so has my life. To this end, I would like to thank my wife Sarah for her support and understanding during the many long evening hours when I was huddled away working. Finally, with a new daughter in my life, I would not have had the time to write this dissertation were it not for the immeasurable help of my parents, Beverley and Wally.

Dedication

For Rowan.

Chapter 1

Introduction

While working at the Berlin Academy, the renowned Swiss mathematician Leonard Euler was asked to tutor Frederick the Great's niece, the Princess of Anhalt-Dessau, in all matters of natural science and philosophy [31]. Euler's tutelage of the princess continued from 1760 to 1762 and culminated in the publishing of the popular and widely-translated "Letters to a German Princess" [13]. In the letters, Euler eloquently wrote about diverse topics ranging from why the sky was blue to free will and determinism.

In his lesson on categorical propositions and syllogisms, Euler used diagrams composed of overlapping circles; these diagrams became known as Eulerian circles, or simply Euler diagrams. In an Euler diagram, a proposition's classes are represented as circles whose overlap depends on the relationship established by the proposition. For example, the propositions

All birds are animals
Some animals are carnivores

can be represented by Fig. 1.1.

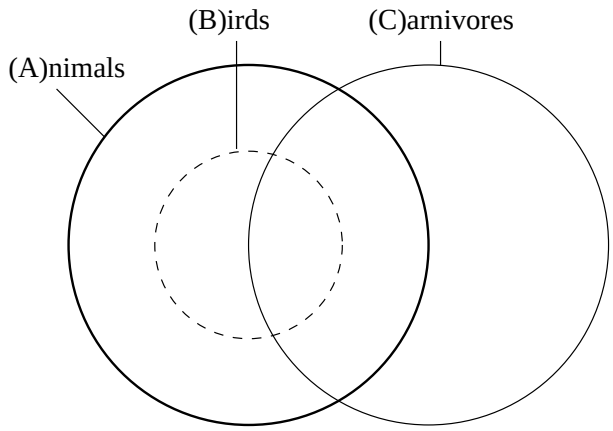


Figure 1.1: An example of an Euler diagram.

In 1880, John Venn, a Cambridge priest and mathematician, published a paper studying special instances of Euler diagrams in which the classes overlap in all possible ways [46]; although originally applied to logical reasoning, these "Venn diagrams" are now commonly used to teach students about set theory. For example, the Venn diagram in Fig. 1.2 shows all the ways in which three sets can intersect. The primary difference between Venn and Euler diagrams is how they represent empty sets (e.g., the set of birds that are *not* animals in the example of Fig. 1.1). In an Euler diagram, regions representing empty sets are omitted, while in Venn diagrams they are included but denoted by shading.

One of the most common uses of Euler and Venn diagrams is to visualize the distribution of discrete characteristics across a sample population. For example, suppose researchers catalog all plant and animal organisms on an island with particular attention being paid to the bird and carnivore populations. Table 1.1 shows the results of the study.

Based on the three characteristics of being an animal, a bird, and/or a carni-

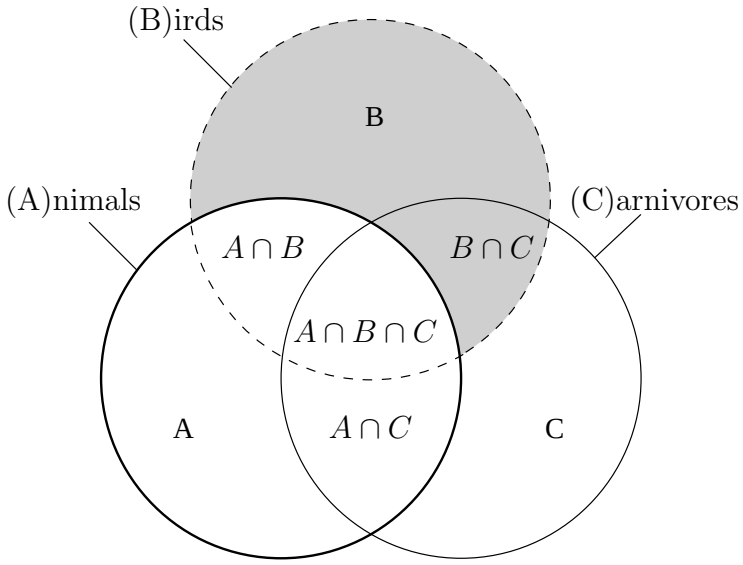


Figure 1.2: A Venn diagram that represents the Euler diagram in Fig. 1.1 by shading the missing regions.

Description	Count
Organisms	1000
Animals	237
Plants	763
Carnivores	83
Birds	21
Carnivorous Birds	8
Carnivorous Plants	11

Table 1.1: Sample study results of plant and animal populations.

vore, the study's results can be represented by the Euler diagram in Fig. 1.3. In the diagram, each characteristic is represented by a circle whose interior represents those organisms with the characteristic and whose exterior represents those organisms without the characteristic. In addition, a surrounding rectangle, called the *universe* represents the entire sample space. Each region is labeled with the number of organisms with *exactly* the characteristics represented by the circles containing the region. For example, since 21 birds were counted, the sum of the labels of the two regions inside the 'Birds' circle totals 21. Since 8 birds were carnivorous, we can deduce that 13 were not carnivorous and therefore the region inside the 'Birds' circle, but outside the 'Carnivores' circle is labeled with 13. The universe is labeled with the number of organisms that are neither animals, birds, nor carnivores. Additional labels could be added, for example to provide sums for the numbers of animals, birds, and carnivores, but these values can be derived from the existing labels so they are omitted for simplicity.

The Euler diagram in Fig. 1.3 enhances the data from Tab. 1.1 by explicitly representing the subset relationship between birds and animals, and by providing immediate access to the finer granularity population counts, which would otherwise have to be derived. One of the reasons why Fig. 1.3 is better than Tab. 1.1 at conveying the study's results is that it leverages both the reader's analytical ability (e.g., to understand the numbers) and the reader's perceptual ability (e.g., to see that birds are a subset of animals). An information visualization method that employs a reader's perceptual capabilities can reduce the amount of mental effort required to understand the conveyed data.

In Fig. 1.3, the population sizes are represented numerically and are not linked

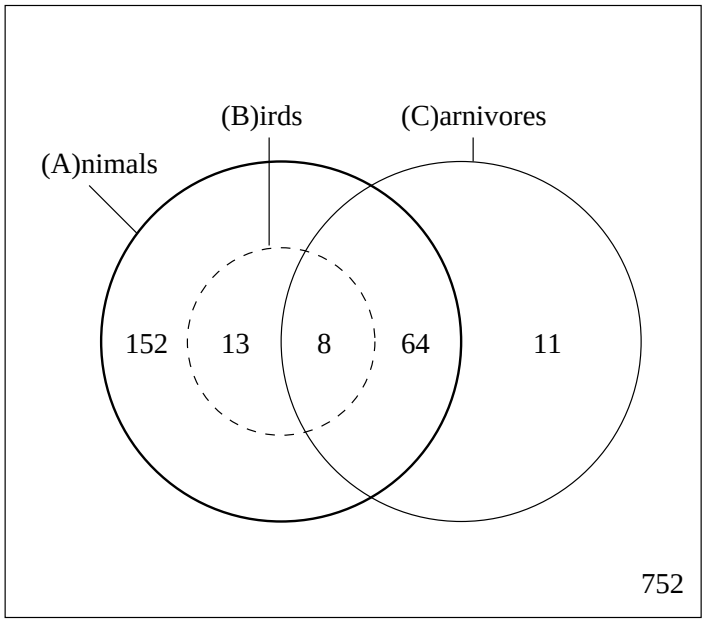


Figure 1.3: An Euler diagram representing the data from Tab. 1.1.

to any visual cues; the corresponding regions' areas bear no relation to their labeled sizes. For example, the region representing the 11 non-animal carnivores (i.e., carnivorous plants) has a greater area than the region representing the 13 non-carnivorous birds. Figure 1.4 shows a variation of Fig. 1.3 where the regions' areas have the same proportions as their respective labeled sizes (i.e., the region representing non-animal carnivores now has an area just slightly smaller than the region representing non-carnivorous birds); such diagrams are said to be *area-proportional*. The proportionality of individual region's areas also applies to subsets of regions. For example, it is clear from Fig. 1.4 that the number of carnivores is less than half the number of animals just by comparing the size of their respective circles; such an observation is not immediately obvious from 1.3 without summing the individual labels and is an example of how perceptual qualities can be used to improve data understanding.

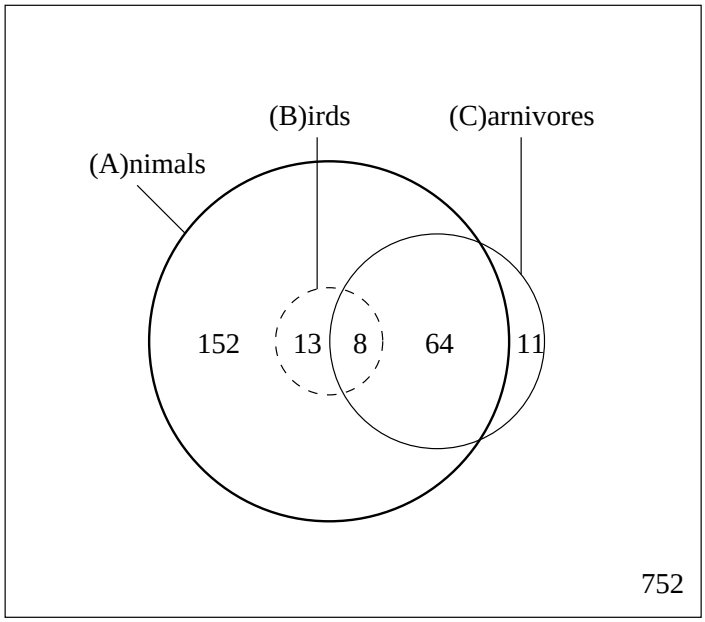


Figure 1.4: An area-proportional version of the Euler diagram in Fig. 1.3.

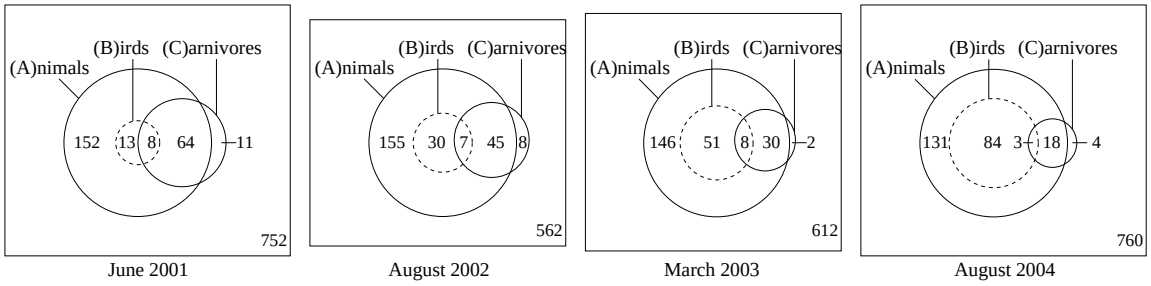


Figure 1.5: A time-based sequence of area-proportional Euler diagrams with a constant scaling factor, which allows direct comparison of diagrams.

In addition to enhancing a diagram's understanding, the area metric provides a common reference for comparing Euler diagrams. Figure 1.5 shows a sequence of four area-proportional Euler diagrams representing the results of studies similar to 1.1, but taken over the course of three years. Because the diagrams share the same scale (i.e., a region with labeled size 10 would have the same area in each diagram), they can be directly compared to discover trends without having to analyze the labels. For example, by noting the size of the universe rectangle, the reader can quickly ascertain that the study size decreased in the second year and then increased in subsequent years; using a similar process, the following trends can be seen:

1. The animal population remained relatively constant,
2. the bird population steadily increased,
3. the carnivore population steadily decreased,
4. and, in the final year, there was a marked decrease in carnivorous birds.

Because they are well-defined and have a rigid definition, the mathematical properties of Venn diagrams have been extensively studied [41]. The existence of Venn diagrams for an arbitrary number of sets was proved by John Venn [46] via a general construction method, and later by another construction due to Anthony Edwards [10].

Unfortunately, the flexible nature of Euler diagrams has resulted in considerably less mathematical scrutiny than Venn diagrams; in spite of this, Euler diagrams continue to be used in an ad hoc fashion to visualize population distributions, particularly in the biological and medical sciences fields [1].

The purpose of this dissertation is to formalize our understanding of Euler diagrams and to present algorithms for generating and drawing both general and area-proportional Euler diagrams such as those shown in Fig. 1.5. Before presenting

a formal mathematical framework for describing Euler diagrams and characterizing their important features, we provide on overview of this dissertations' chapters.

1.1 Chapter Overview

Chapter 2 describes previous Euler diagram research within the context of our framework and details how this dissertation goes beyond these existing results. The remaining chapters describe original research contributed by this author. In some cases, the research is in collaboration with other individuals, and these cases will be so noted.

Chapter 3 introduces algorithms for drawing area-proportional Euler and Venn diagrams for up to three curves, where the curve shapes are restricted to being circles and rectangles. In addition, Section 3.4 presents an interesting result proving there is a limit to the type of shapes that can be used to draw even small instances of Venn diagrams.

Chapters 4 and 5 relate Euler diagrams to the well-studied area of graph theory and provide several significant necessary-and-sufficient conditions for the existence of Euler diagrams. Using the existence conditions, Chapter 6 explores the effect of several restrictions that may be placed on Euler diagrams; the result is a hierarchy of Euler diagram types according to their expressiveness. Chapter 7 uses the existence conditions, as well, to prove that the problem of generating Euler diagrams is NP-complete; in addition, it presents some important subproblems whose computational complexity remains unknown.

Chapter 8 presents theory and algorithms related to how smaller instances of Euler diagrams can be joined to create larger Euler diagrams; the results of this

chapter provide a heuristic solution to the NP-complete problem of generating Euler diagrams.

Lastly, in the same vein as Chapter 3, Chapter 9 returns to the problem of drawing special instances of area-proportional Euler diagrams and presents an algorithm for generating and area-proportionally drawing the class of *monotone* Euler diagrams.

We now present a formal mathematical framework for Euler diagrams that provides a foundation for the results of this dissertation.

1.2 Basic Definitions

This section is divided into three subsections. The first subsection introduces some terminology for describing the topology of the plane and introduces the Jordan Curve Theorem. The second subsection applies the previously developed terminology to formally define Euler diagrams and the problems that will be investigated in this dissertation. The third and final subsection introduces a generalization of Euler diagrams which, while not being the focus of this dissertation, has appeared often enough in the related literature to warrant inclusion.

1.2.1 Subsets of the Euclidean plane $\mathbb{R}^2$

The manifestation of an Euler diagram is as a drawing in the Euclidean plane $\mathbb{R}^2$ (or just "the plane"), so we begin our formal definition of Euler diagrams by considering some properties of subsets of $\mathbb{R}^2$. Many of the following definitions are based on intuitive notions and should be sufficient to establish a foundation for understanding Euler diagrams; for more formal definitions, the reader is referred to one of the many

books written on the topics of metric and topological spaces [36].

The definition of the plane as the infinite set of points

$$\mathbb{R}^2 = \{(x, y) | x, y \in \mathbb{R}\}$$

leads to a natural definition of subsets of the plane. In fact, in this dissertation we define curves as subsets of $\mathbb{R}^2$. For example, we can think of the unit circle C as the subset

$$C = \{(x, y) \in \mathbb{R}^2 | \sqrt{x^2 + y^2} = 1\}.$$

For Euler diagrams, we usually treat plane subsets as *labeled* sets. For example, we may have two circles C_x and C_y that are completely overlapping, but represent distinct entities x and y in the diagram. Although C_x and C_y are *not* equal as labeled sets, they are equal as unlabeled sets, and thus we say that C_x and C_y are *equivalent*. Sometimes it is useful to ignore a plane subset C's label, or to emphasize its set definition; for this we use the notation $pts(C)$. Returning to our example, C_x and C_y are *equivalent* since $pts(C_x) = pts(C_y)$.

Given a plane subset S and point $(x, y) \in S$, we are often interested in considering all the points within a given radius of (x, y); this set is referred to as the *neighbourhood* of (x, y). Depending on the neighbourhood's radius, a property P may or may not hold in the neighbourhood (e.g., P may be the property that the neighbourhood is not in S). Most importantly, we are interested in whether or not P holds as we get arbitrarily close to (x, y); in this case, we say that P holds in the *immediate neighbourhood* of (x, y). The following definitions formalize these concepts.

Definition 1.2.1. Let S be a plane subset and ϵ be a positive real number. For any

point $(x, y) \in S$, the *neighbourhood* of (x, y), denoted $N_\epsilon(x, y)$, is defined as

$$N_\epsilon(x, y) = \{(x', y') \in \mathbb{R}^2 | \sqrt{(x' - x)^2 + (y' - y)^2} < \epsilon\}.$$

Definition 1.2.2. Let S be a plane subset and P be some property that applies to a neighbourhood. We say that P holds in the *immediate neighbourhood* of (x, y) if there is a positive real number ϵ' such that for all $0 < \epsilon < \epsilon'$, P holds in N_ϵ.

The notion of neighbourhood leads to the following definitions for open and closed plane subsets.

Definition 1.2.3. A plane subset S is *open* if every point $(x, y) \in S$ has a neighbourhood in S.

Definition 1.2.4. A plane subset S is *closed* if its complement $\overline{S} = \mathbb{R}^2 \backslash S$ is open.

Returning to the unit circle C, it *partitions* the plane into three distinct subsets: the closed set C itself, the open interior of C (denoted $int(C)$), and the open exterior of C (denoted $ext(C)$); these subsets are shown in Fig. 1.6(a) and have the following formal definitions:

$$\begin{aligned} C &= \{(x, y) \in \mathbb{R}^2 | \sqrt{x^2 + y^2} = 1\} \\ int(C) &= \{(x, y) \in \mathbb{R}^2 | \sqrt{x^2 + y^2} < 1\} \\ ext(C) &= \{(x, y) \in \mathbb{R}^2 | \sqrt{x^2 + y^2} > 1\}. \end{aligned}$$

We also have the notion of the *boundary* of a plane subset.

Definition 1.2.5. Let S be a plane subset. The *closure* of S, denoted S_c, is the unique smallest closed set such that $S \subseteq S_c$.

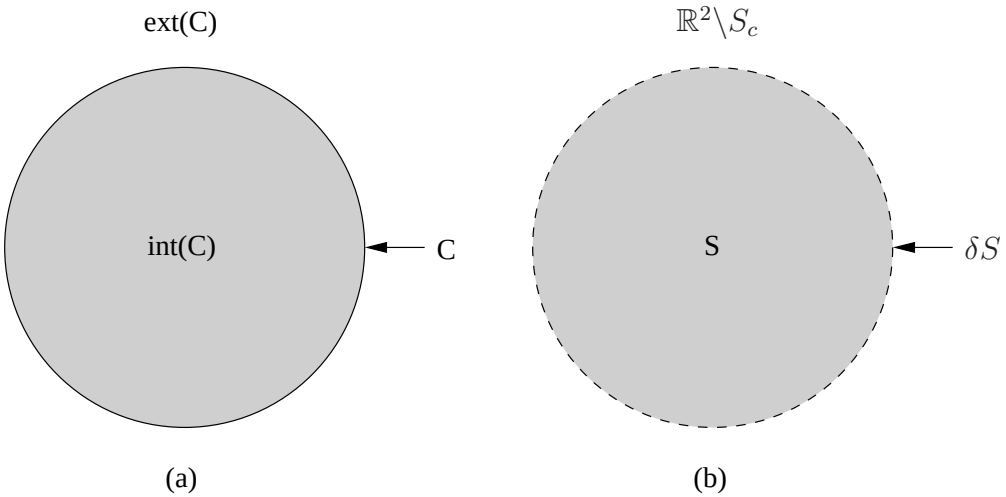


Figure 1.6: Two equivalent partitions of the plane, one defined by (a) a closed set C (the circle), and the other by (b) an open set S (the circle's interior).

Definition 1.2.6. Let S be a plane subset. The *boundary* of S, denoted δS, is defined as

$$\delta S = S_c \cap \overline{S}_c.$$

As an example of these definitions, consider Fig. 1.6(b), which shows the open set S (shaded and previously defined to be the interior points of the unit circle), its boundary δS (dashed), and the complement $\mathbb{R}^2\backslash S_c$ of its closure. Note the correspondence between Figs. 1.6(a) and (b) where we have

$$\begin{aligned} C &= \delta S, \\ int(C) &= S, \text{ and} \\ ext(C) &= \mathbb{R}^2\backslash S_c. \end{aligned}$$

Informally, the open set S in Fig. 1.6(b) is *connected* because for any two points

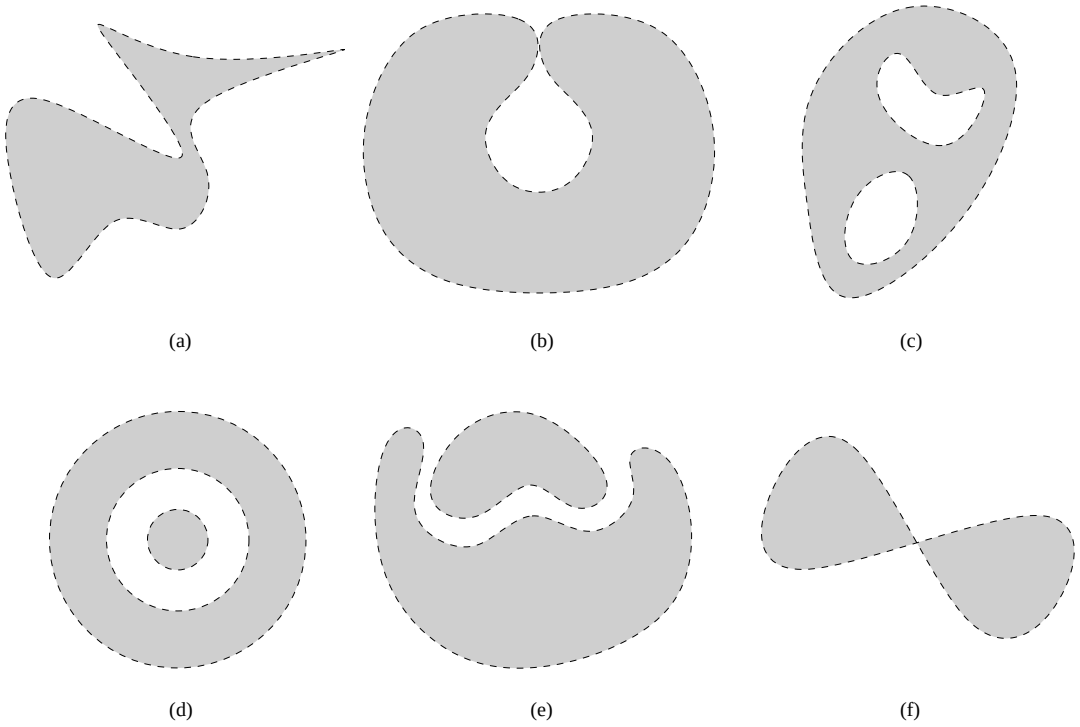


Figure 1.7: Examples of open sets (shaded) and their boundaries (dashed); (a)–(c) are connected and (d)–(f) are disconnected.

$u, v \in S$, there is a curve with endpoints u and v that is contained in S. Formally, an open set S is *disconnected* if there are non-empty open sets S_1 and S_2 such that

1. $S_1 \cup S_2 = S$, and
2. $S_1 \cap S_2 = \emptyset$.

An open set is *connected* if it is *not* disconnected.

Figure 1.7 shows several examples of connected and disconnected open sets. Of particular note is Fig. 1.7(c), which shows that a connected open set can have a disconnected boundary, and Fig. 1.7(f), which shows the inverse, that a disconnected open set can have a connected boundary.

In the example Euler diagrams from the introduction, each item (e.g., animals, birds, or carnivores in Fig. 1.1), is represented by a circle, or more specifically, the

open set whose boundary is a circle. A circle is an example of a simple closed curve: simple because it does not cross itself and closed because it has no endpoints. Simple closed curves are also known as Jordan curves.

The definition of a Jordan curve is very broad and includes common geometric shapes (e.g., triangles, squares, ellipses, and polygons), as well as fractals such as the Hilbert Curve, which is a space-filling curve, and the Koch Snowflake, which has finite area and infinite perimeter. For Euler diagrams, we are interested in curves that are "reasonable" to draw (i.e., exhibit no infinite phenomena). A common practice in computer graphics is to restrict curves to being "smooth" or C^2-continuous (i.e., having first and second derivatives, assuming a parametric curve definition) [17]. As such, we explicitly limit Jordan curves to being piecewise smooth.

Definition 1.2.7. A *Jordan curve* is a simple (i.e., not self-intersecting) closed curve that is piecewise C^2-continuous (i.e., piecewise doubly-differentiable). □

Example. In Fig. 1.7, (a) is the only plane subset with a Jordan curve boundary. The boundaries of (b) and (f) are not a Jordan curves because they self-intersect, and the boundaries of (c)–(e) are not Jordan curves because they are disconnected. □

We now have enough terminology to proceed with a fundamental result that generalizes our previous discussion of how a circle partitions the plane; this result is known as the Jordan Curve Theorem and, although apparently obvious, was not correctly proved until 1905:

Theorem 1.2.1 (Jordan Curve Theorem)**.** *The complement* $\mathbb{R}^2 \backslash C$ *of the points of a Jordan curve* C *is composed of two* disjoint *open sets:*

1. *the connected bounded interior* $int(C)$*, and*

2. the connected unbounded exterior $ext(C)$. □

We are now ready to consider how a set of Jordan curves partitions the plane in order to form an Euler diagram.

1.2.2 Euler Diagrams

Let $C = \{c_{x_1}, c_{x_2}, \ldots, c_{x_n}\}$ be a set of n Jordan curves where $x_i = x_j \implies i = j$ (i.e., the curves are *uniquely* labeled, which we will assume from now on). The set $\{x_i | 1 \leq i \leq n\}$ is referred to as the labels of C and denoted $labels(C)$.

In the previous Euler diagram examples, the curves have been labeled with the real-world entities that they represent; however, for the purpose of mathematical abstraction, we will generally label the curves with unique integers. In most of our examples, we let $x_i = i$, in which case $labels(C) = \{1, 2, \ldots, n\}$.

We use a lowercase c_{x_i} to represent each Jordan curve because it is an element of the set C; however, the reader is reminded that in the following definitions, we also consider c_{x_i} a subset of $\mathbb{R}^2$. Because of the plane subset interpretation of curves, C also has an implied plane subset interpretation, denoted $pts(C)$, and defined as

$$pts(C) = \bigcup_{c_i \in C} c_i.$$

The following definitions describe how two Jordan curves interact with each other by first classifying their intersections and then using this classification to define their relationship.

Definition 1.2.8. Let c_i and c_j be two Jordan curves and let the set of *intersections* between c_i and c_j, denoted $I^*(c_i, c_j)$, be the partition of $c_i \cap c_j$ into (maximal)

connected subsets.

The set of *point intersections* between c_i and c_j, denoted $I_P(c_i, c_j)$, is defined as

$$I_P(c_i, c_j) = \{S \in I^*(c_i, c_j) || S| = 1\}.$$

The set of *concurrent intersections* between c_i and c_j, denoted $I_C(c_i, c_j)$, is defined as

$$I_C(c_i, c_j) = I^*(c_i, c_j) \backslash I_P(C).$$

A point intersection $p \in I_P(c_i, c_j)$ is *transverse* if c_i is both interior and exterior to c_j within an immediate neighbourhood of p.

If c_i and c_j are not equivalent, a concurrent intersection $c \in I_C(c_i, c_j)$ is a simple curve with two endpoints; in this case, c is *transverse* if c_i is interior to c_j within the immediate neighbourhood of one endpoint and c_i is exterior to c_j within the immediate neighbourhood of the other endpoint.

An intersection is *tangential* if it is not transverse. □

Example. Figure 1.8 shows an example of two Jordan curves c_1 (solid) and c_2 (dashed) that exhibit the four possible types of intersections. $I^*(c_1, c_2)$ has four elements:

1. a singleton set containing the point marked "transverse point",
2. a singleton set containing the point marked "tangential point",
3. a set containing the points in the concurrent intersection marked "tangential concurrent", and
4. a set containing the points in the concurrent intersection marked "transverse concurrent".

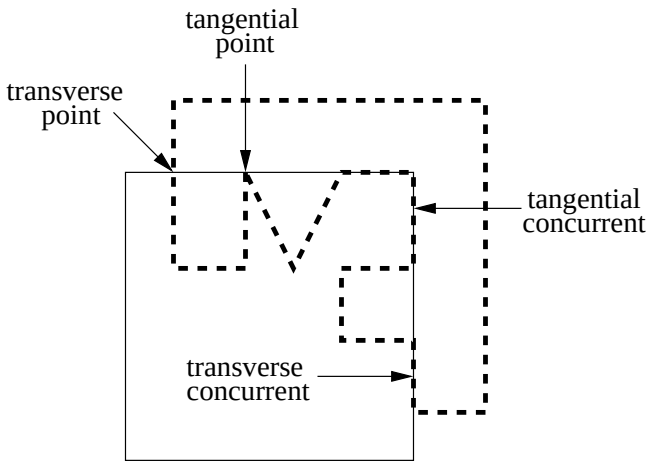


Figure 1.8: The four different types of intersections between a pair of Jordan curves.

$I_P(c_1, c_2)$ contains elements 1 and 2 and $I_C(c_1, c_2)$ contains elements 3 and 4.

In terms of point intersections, element 1 describes a transverse intersection because c_1 and c_2 cross each other at this point. On the other hand, element 2 describes a tangential intersection because although c_1 and c_2 intersect at this point, c_2 remains interior to c_1. In terms of concurrent intersections, element 3 describes a tangential intersection because c_2 is interior to c_1 at both endpoints of the concurrent intersection. On the other hand, element 4 describes a tangential intersection because c_2 is interior to c_1 at one endpoint and exterior to c_1 at the other endpoint of the concurrent intersection. □

Definition 1.2.9. Let c_i and c_j be two Jordan curves. We define the following relationships between c_i and c_j:

- c_i and c_j are *equivalent* if $pts(c_i) = pts(c_j)$,
- c_i and c_j *intersect* if $c_i \cap c_j \neq \emptyset$,
- c_i and c_j are *disjoint* if $c_i \cap c_j = \emptyset$,
- c_i and c_j intersect *transversely* if they have a transverse intersection,

- c_i and c_j intersect *tangentially* if they have a tangential intersection, and
- c_i and c_j are *concurrent* if they have a concurrent intersection. □

Example. Figure 1.9 shows several examples of pairs of Jordan curves that exhibit various relationships. Note that any equivalent curves, such as those in Fig. 1.9(a), are by definition concurrent; in addition, since the concurrent intersection has no endpoints, the criteria for transverse intersections does not apply, and by default, the intersection is tangential. Figure 1.9(b) is an example of how the "intersect transversely" and "intersect tangentially" relationships are not mutually exclusive; each relationship applies as long as there is at least one qualifying intersection. Finally, the pair of curves in Fig. 1.9(c) demonstrates that the definition of disjoint is based on curve boundaries and *not* curve interiors. □

Now that we've considered the interactions between a pair of Jordan curves, we can move on to sets of Jordan curves.

Definition 1.2.10. Let C be a set of Jordan curves. An *intersection* of C is a point in $\mathbb{R}^2$ shared by two or more of C's curves and is said to be *pairwise* if it is shared by *exactly* two curves and *non-pairwise* otherwise.

The *convergence points* of C, denoted $I_{CP}(C)$, are a special subset of C's intersections. Informally, a convergence point is a point in $\mathbb{R}^2$ where two or more curves begin to intersect. Formally, I_{CP} is the union of the point intersections and concurrent intersection endpoints over all pairs of curves in C; that is,

$$I_{CP}(C) = \bigcup_{c_i, c_j \in C, i \neq j} \left(I_p(c_i, c_j) \cup \bigcup_{c \in I_C(c_i, c_j), c \text{ open}} \{c_a, c_b\} \right)$$

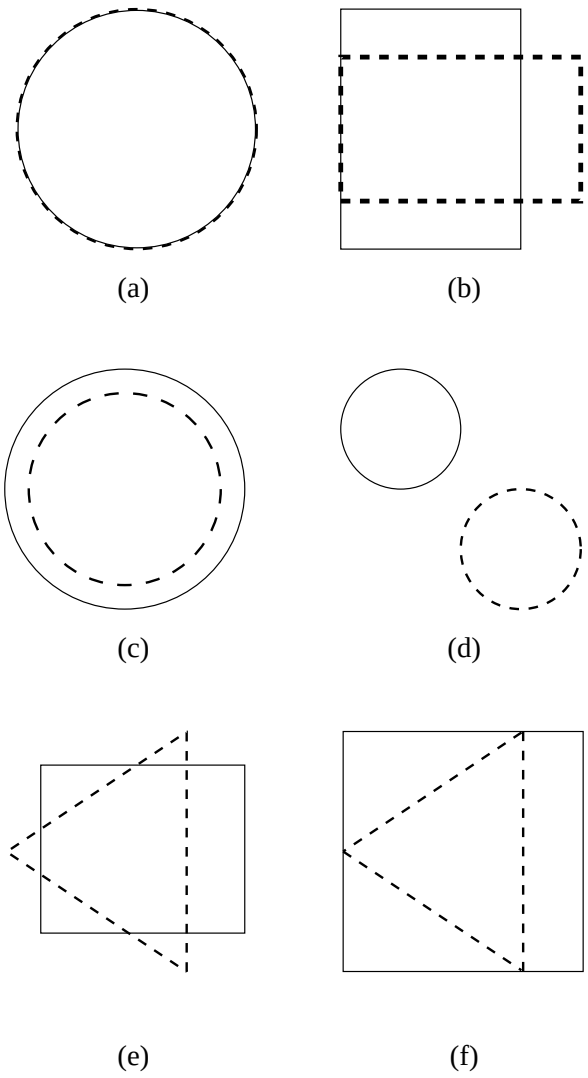


Figure 1.9: Examples of pairs of Jordan curves that exhibit the following relationships: (a) equivalent, intersect tangentially, concurrent, (b) intersect transversely, intersect tangentially, concurrent, (c) disjoint, (d) disjoint, (e) intersect transversely, and (f) intersect tangentially.

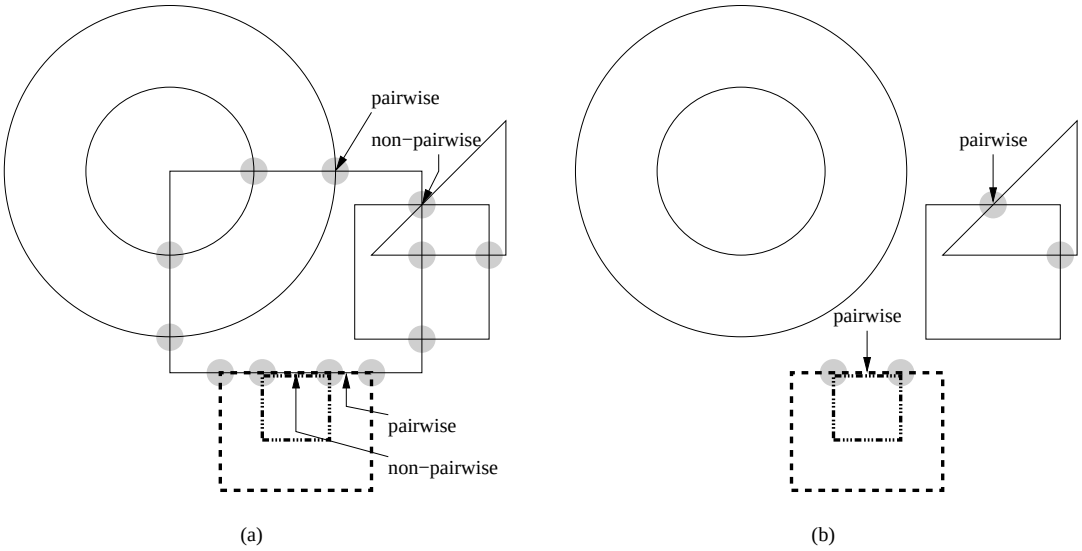


Figure 1.10: Example of two sets of Jordan curves where the convergence points are shaded. (a) is connected, concurrent, and non-pairwise, and (b) is disconnected, concurrent, and pairwise.

where "c open" means that c is not a closed curve and therefore has endpoints c_a and c_b.

Lastly, we classify C according to the following properties:

- C is *connected* if $pts(C)$ is a connected set,
- C is *concurrent* if it has a concurrent pair of curves, and
- C is *pairwise* if *all* its intersections are pairwise. □

Example. Figure 1.10 shows two examples of sets of Jordan curves; in both cases, the convergence points are shaded. The first set, Fig. 1.10(a), is connected because its points form a connected plane subset; in addition, it demonstrates that a connected set of Jordan curves can still have pairs of disjoint curves (e.g., the two circles). Figure 1.10(a) is also concurrent because it has several pairs of concurrent curves (e.g., the dashed rectangles), and non-pairwise because of the isolated point shared by the

triangle and two rectangles and the concurrent curve segment shared by the solid rectangle and two dashed rectangles. The second set, Fig. 1.10(b), is disconnected, concurrent, and pairwise; it is an example of how the pairwise property applies equally to point and concurrent intersections.

From these examples, we can also deduce that a point $(x, y) \in \mathbb{R}^2$ is a convergence point for a set C of Jordan curves if the complement of $pts(C)$ has more than two connected open sets within an immediate neighbourhood of (x, y). □

We now look beyond the intersection of curve points and consider the intersection of curve interiors and exteriors.

Definition 1.2.11. Let C be a set of Jordan curves. A subset $X' \subseteq labels(C)$ is called a *region label*. Each region label X' identifies a (possibly disconnected or empty) open subset of $\mathbb{R}^2$ called *region* X', denoted $r(X')$, that is interior to the curves identified in X' *and* exterior to the remaining curves; that is,

$$r(X') = \left(\bigcap_{i \in X'} int(c_i) \right) \cap \left(\bigcap_{j \in labels(C) \setminus X'} ext(c_j) \right).$$

Since the interior of each curve is bounded, every region is bounded except for $r(\emptyset)$, which is exterior to all curves.

The *non-empty regions* of C, denoted $R(C)$, is the set of *region labels* corresponding to all non-empty regions; that is,

$$R(C) = \{X' \subseteq labels(C) | r(X') \neq \emptyset\}.$$

Since $r(\emptyset) \neq \emptyset$, $R(C)$ always contains $\emptyset$. □

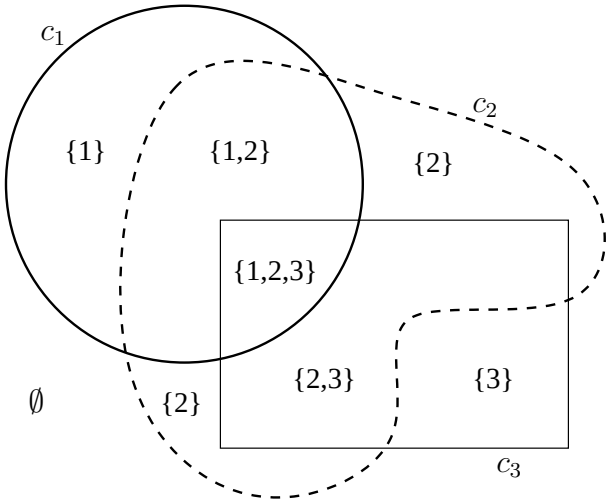


Figure 1.11: Jordan curves $C = \{c_1, c_2, c_3\}$ with each non-empty region $r(X')$ labeled by X'.

Although the term "region X'" and $r(X')$ refer to a subset of $\mathbb{R}^2$, as we did with a curve C and $pts(C)$, we may specifically use $r(X')$ to emphasize the set nature of a region.

Example. Figure 1.11 shows how three Jordan curves divide the plane into seven non-empty regions. Note how $r(\emptyset)$ is unbounded, $r(\{1,3\})$ is empty, and $r(\{2\})$ is disconnected. In this example,

$$R(C) = \{\emptyset, \{1\}, \{2\}, \{3\}, \{1,2\}, \{2,3\}, \{1,2,3\}\}.$$

□

Now that we have a framework for describing how a set of Jordan curves divides the plane into regions, we can define an Euler diagram as a set of Jordan curves with additional constraints.

Definition 1.2.12. A set C of n Jordan curves is an *Euler diagram* if and only if

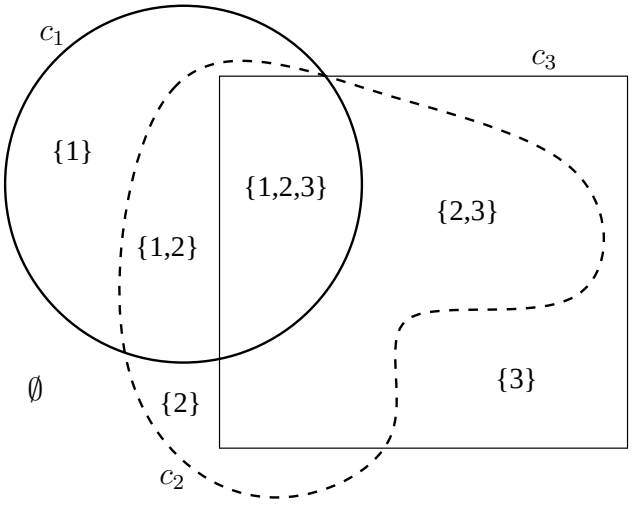


Figure 1.12: An Euler diagram representing $\{\emptyset, \{1\}, \{2\}, \{3\}, \{1,2\}, \{2,3\}, \{1,2,3\}\}$.

every non-empty region is connected; that is, $r(X')$ is a connected subset of $\mathbb{R}^2$ for all $X' \in R(C)$.

To emphasize the number of curves, we may refer to C as an *n-Euler diagram*.

In addition, we say that C *represents* $R(C)$. □

Although $C = \{\}$ (i.e., the empty plane) is technically an Euler diagram, the need to consider empty Euler diagrams can lead to the inclusion of trivial, but obfuscating, clauses in formal statements. For clarity of exposition, unless specifically noted, we assume that all Euler diagrams are non-empty.

Example. The set C of Jordan curves in Fig. 1.11 does not constitute an Euler diagram because $r(\{2\})$ is disconnected. On the other hand, the set C' of Jordan curves in Fig. 1.12 is an Euler diagram even though $R(C) = R(C')$. The requirement for regions to be connected is important for readability since it allows the number of non-empty regions to be quickly ascertained. In addition, labeling disconnected regions with quantities can be confusing because each connected subset is a candidate

for labeling. For example, if only one connected subset is labeled, then the others are left blank, and if all connected subsets are labeled, are they labeled with the region's overall quantity or a portion thereof? □

Since an Euler diagram C is a set of Jordan curves, all the properties of Def. 1.2.10 apply to C; that is, an Euler diagram may be connected or disconnected, concurrent or non-concurrent, and pairwise or non-pairwise. We are particularly interested in what effect these properties have on the algorithmic aspects of Euler diagram generation as well as on the expressiveness of Euler diagram in terms of which combinations of non-empty regions are possible. The following definition identifies a subset of Euler diagrams that are historically significant because they categorize Euler's original examples as well as being algorithmically interesting because of their stringent requirements.

Definition 1.2.13. An Euler diagram is *simple* if it is non-concurrent *and* pairwise. □

Example. Figure 1.13 shows examples of simple and non-simple Euler diagrams. The Euler diagram in Fig. 1.13(a) is simple because it has no concurrent curves and all its intersections are pairwise. On the other hand, the Euler diagrams in Figs. 1.13(b)–(d) are non-simple because they have either a non-pairwise intersection, a pair of concurrent curves, or both. □

Having developed a formal definition of Euler diagrams, we can now define Venn diagrams as the class of Euler diagrams that arises when the curve interiors overlap in all possible ways.

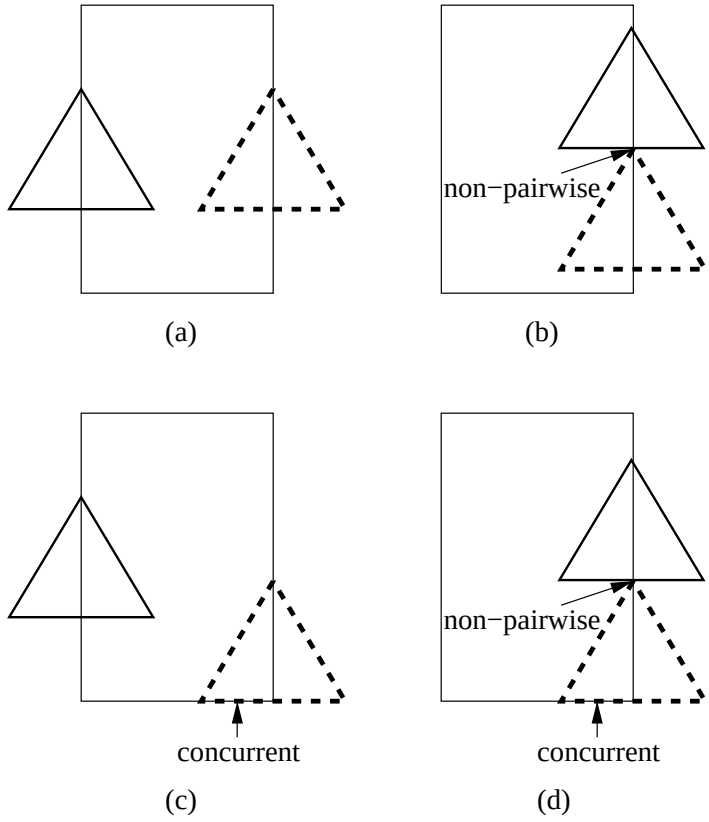


Figure 1.13: Examples of (a) a simple Euler diagram, (b) a non-simple Euler diagram with a non-pairwise intersection, (c) a non-simple Euler diagram with a concurrent intersection, and (d) a non-simple Euler diagram with both non-pairwise and concurrent intersections. Note how all these diagrams represent the same non-empty regions.

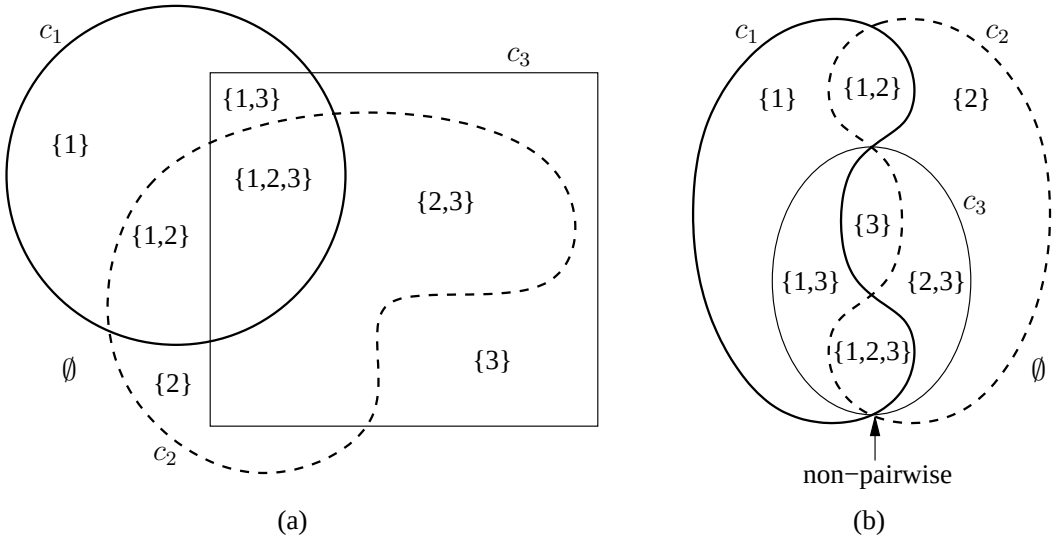


Figure 1.14: Two 3-Venn diagrams representing $\mathcal{PS}(\{1,2,3\})$; (a) is simple and (b) is non-simple.

Definition 1.2.14. An n-Euler diagram C is a *Venn diagram* if and only if $R(C) = \mathcal{PS}(labels(C))$ where $\mathcal{PS}$ is the power set (i.e., the set of all subsets).

As before, to emphasize the number of curves, we may refer to C as an *n-Venn diagram*. □

Note that Def. 1.2.13 also applies to Venn diagrams so we may refer to a *simple* Venn diagram.

Example. The Euler diagram in Fig. 1.12 is *not* a Venn diagram because $r(\{1,3\})$ is empty. On the other hand, the Euler diagram C in Fig. 1.14(a) is a 3-Venn diagram because

$$\begin{aligned} R(C) &= \{\emptyset, \{1\}, \{1,2\}, \{1,2,3\}, \{1,3\}, \{2\}, \{2,3\}, \{3\}\} \\ &= \mathcal{PS}(\{1,2,3\}). \end{aligned}$$

Since C has no concurrent curves and only pairwise intersections, it is an example of a simple Venn diagram. In contrast, the Venn diagram in Fig. 1.14(b) has two non-pairwise intersections, so it is non-simple. □

The previous definitions have all dealt with the topological relationships of Jordan curves. In order to define area-proportional Euler diagrams, we have to consider area metrics for the regions. Since we think of an Euler diagram C as a drawing in the plane, each region X' of C has an *area*, denoted $area(X')$, that is expressed in some unit system (e.g., cm^2 or in^2); the choice of unit system is unimportant as long as it remains constant. An empty region X' has no area, so $area(X') = 0$, while a non-empty region $X' \in R(C)$ has $area(X') > 0$. Since all regions except for $r(\emptyset)$ are bounded, $area(X') = \infty$ if and only if $X' = \emptyset$.

In addition to region areas, we are also interested in the total area of C, denoted $area(C)$; that is,

$$area(C) = \sum_{X' \in R(C) \setminus \{\emptyset\}} area(X').$$

The definitions for region and diagram area have omitted any consideration for the area required by the curve boundaries. How boundaries are represented is an implementation detail that must be considered by all Euler diagram visualization software. Any choice to draw boundaries will affect region areas; the boundary of a curve is the limit between its interior and exterior, but in the reality of a pixelated display, limits do not exist and the boundary representation will encroach on either the curve's interior or exterior. However, since boundary representation does not affect the results of this dissertation, we idealize Euler diagrams as being drawn with infinitely thin curve boundaries.

With area-proportional Euler diagrams, we are interested in the *relative* areas of the regions rather than their absolute areas; this is why we can ignore the specific units of area measurement. In other words, whether or not an Euler diagram is area-proportional is invariant under scaling; the following definition captures this meaning.

Definition 1.2.15. Let C be an Euler diagram and let $\omega : R(C)\backslash\{\emptyset\} \to \mathbb{R}^+$ be a *weight function* for the non-empty bounded regions of C with ω_{tot} the sum of ω over its domain.

C is *area-proportional* with respect to ω, or simply *ω-proportional*, if and only if there is a positive constant $\alpha \in \mathbb{R}^+$ such that for all $X' \in R(C)$,

$$\frac{area(X')}{area(C)} = \alpha \frac{\omega(X')}{\omega_{tot}}.$$

□

Up to now, we have taken an Euler diagram C and derived its non-empty regions $R(C)$; in fact, the opposite is usually the case (i.e., the desired non-empty region labels are specified from which a corresponding Euler diagram is derived). The problem of generating an Euler diagram that has a specific set of non-empty regions is called the *Euler Diagram Generation Problem*, and its area-proportional variant is called the *Area-Proportional Euler Diagram Generation Problem*. Within the context of the generation problems, the non-empty region labels specified in advance of an Euler diagram are referred to as a set system; the following definitions formalize the notion of a set system and the generation problems.

Definition 1.2.16. Let $X = \{x_1, x_2, \ldots, x_n\}$ be a set of abstract objects (e.g., integers, letters, or words), called *items*. A set S is a *set system* on X if:

1. $S \subseteq \mathcal{PS}(X)$,
2. $\emptyset \in S$, and
3. $\bigcup_{s \in S} s = X$.

To emphasize the number of items in a set system (that is, $|X|$ and not $|S|$), we may refer to S as an *n-set system*. In addition, we may refer to S's items as $X(S)$ and the set $\{x_1, x_2, \ldots, x_n\}$ as the *fullset* (in contrast to the empty set $\emptyset$). □

In the above definition, the third rule ensures that each item appears at least once in S; in other words, X is the *smallest* set of items on which S is a set system. In mathematics, X is also referred to as a *ground set* for S, but we opt to use the term "items" for X so that we can clearly differentiate between an "element" of S and an "item" of X. Lastly, as we did with Euler diagram labels, for the purpose of mathematical abstraction, we will use positive integers for set system items (i.e., $X \subset \mathbb{Z}^+$).

Example. The following are examples of set systems:

$$\begin{aligned} S &= \{\emptyset, \{1\}, \{1,2\}, \{1,2,3\}\} \\ X &= \{1,2,3\}, \end{aligned}$$

$$\begin{aligned} S &= \{\emptyset, \{a\}, \{b\}, \{c\}, \{a,b,c,d\}\} \\ X &= \{a,b,c,d\}, \text{ and} \end{aligned}$$

$$\begin{aligned} S &= \{\emptyset, \{Animals\}, \{Animals, Birds\}, \{Animals, Birds, Carnivores\}, \\ &\quad \{Animals, Carnivores\}, \{Carnivores\}\}, \\ X &= \{Animals, Birds, Carnivores\}. \end{aligned}$$

Given $S = \{\{1\}, \{1,2\}, \{1\}\}$ and $X = \{1,2,3\}$, S is *not* a set system on X for two reasons:

1. $\emptyset \notin S$ and
2. $3 \in X$, but 3 does not appear in any of the elements of S. □

With the definitions for Euler diagrams and set systems established, we have a basis for defining two of the principle problems addressed in this dissertation.

Definition 1.2.17. Euler Diagram Generation Problem (EDGP)

INPUT: A set system S.

OUTPUT: An Euler diagram C with $R(C) = S$.

□

Definition 1.2.18. Area-Proportional EDGP (ω-EDGP)

INPUT: A set system S and weight function $\omega : S\backslash\{\emptyset\} \to \mathbb{R}^+$.

OUTPUT: An ω-proportional Euler diagram C with $R(C) = S$.

□

Besides representing S and, if applicable, being ω-proportional, the generation problems do not specify any further properties that the Euler diagrams must have. Requiring the diagrams to have certain properties may be desirable for diagram readability as well as mathematical interest. Besides connected, non-concurrent, and pairwise, there are additional restrictions that we have yet to discuss; for example, one might want the curves to be certain geometric shapes (e.g., circles). The restricted variants of the EDGP and ω-EDGP will be considered in subsequent chapters of this dissertation. In addition, we also consider the associated decision problem (i.e., "*Is* there an Euler diagram representing S?"), particularly with respect to its computational complexity.

Before continuing with the next chapter, we wish to consider a relaxed version of Euler diagrams; although not a focus of this dissertation, they do play a role in

previous Euler and Venn diagram research as well as providing a counterpoint for considering the expressiveness of Euler diagrams as we have defined them.

1.2.3 Euler-like Diagrams

As we saw in Fig. 1.6, we can specify a partition of the plane using either a Jordan curve or a connected open plane subset whose boundary is a Jordan curve. Accordingly, rather than specifying an Euler diagram as a set of Jordan curves, we could just as easily have said that it is a set of connected open plane subsets, each of which is bounded by a Jordan curve. In other words, an Euler diagram

$$C = \{c_1, c_2, \ldots, c_n\}$$

is equivalent to a set

$$C' = \{s_1, s_2, \ldots, s_n\}$$

of connected open subsets $s_i \subset \mathbb{R}^2$ where $\delta s_i = c_i$ for all $1 \leq i \leq n$.

Suppose we relax the condition that each s_i be bounded by a Jordan curve and instead only require that s_i be a *bounded* connected open plane subset; Fig. 1.15 shows several examples of such sets. As we did with Jordan curves, we assume that the boundary δs_i is well-behaved in the sense that it is the union of one or more piecewise C^2-continuous closed curves. If we let $C' = \{s_1, s_2, \ldots, s_n\}$, then Defs. 1.2.10, 1.2.11, and 1.2.12 (that defined the properties of a set of Jordan curves, how they divide the plane into regions, and under what circumstances they form an Euler diagram), can still apply to C' if we use the following equivalent terms in the definitions:

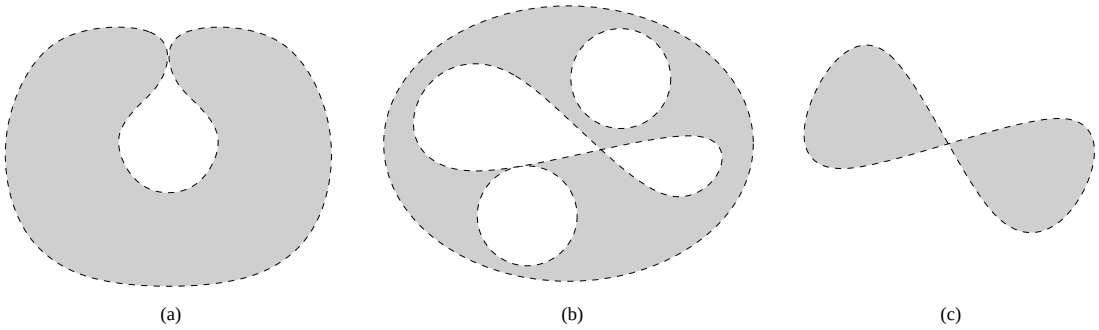


Figure 1.15: Examples of bounded open subsets of the plane; (a) and (b) are connected, while (c) is disconnected.

- $c_i \rightarrow \delta s_i$,
- $int(c_i) \rightarrow s_i$, and
- $ext(c_i) \rightarrow \mathbb{R}^2 \backslash (s_i \cup \delta s_i)$.

We say that C' is an *Euler-like* diagram when it satisfies Def. 1.2.12 (subject to the previously-listed equivalent terms). For example, Fig. 1.16 shows several Euler-like diagrams with the same type of properties (i.e., connected, concurrent, pairwise), that we previously saw for Euler diagrams; in each diagram, the interior of c_1 is shaded to show that its boundary is *not* a single Jordan curve.

The choice to specify an Euler-like diagram by its curve interiors rather than the curves themselves is important; such a specification removes the ambiguity of determining where the interior of a set of non-simple and/or disjoint closed curves lies. Another conscious decision was to specify Euler-like diagrams separately from Euler diagrams. An alternative approach would have been to define Euler diagrams as a subclass of Euler-like diagrams. We opted for the former approach in order to emphasis this dissertations' focus on Euler diagrams; we made this decision for two reasons:

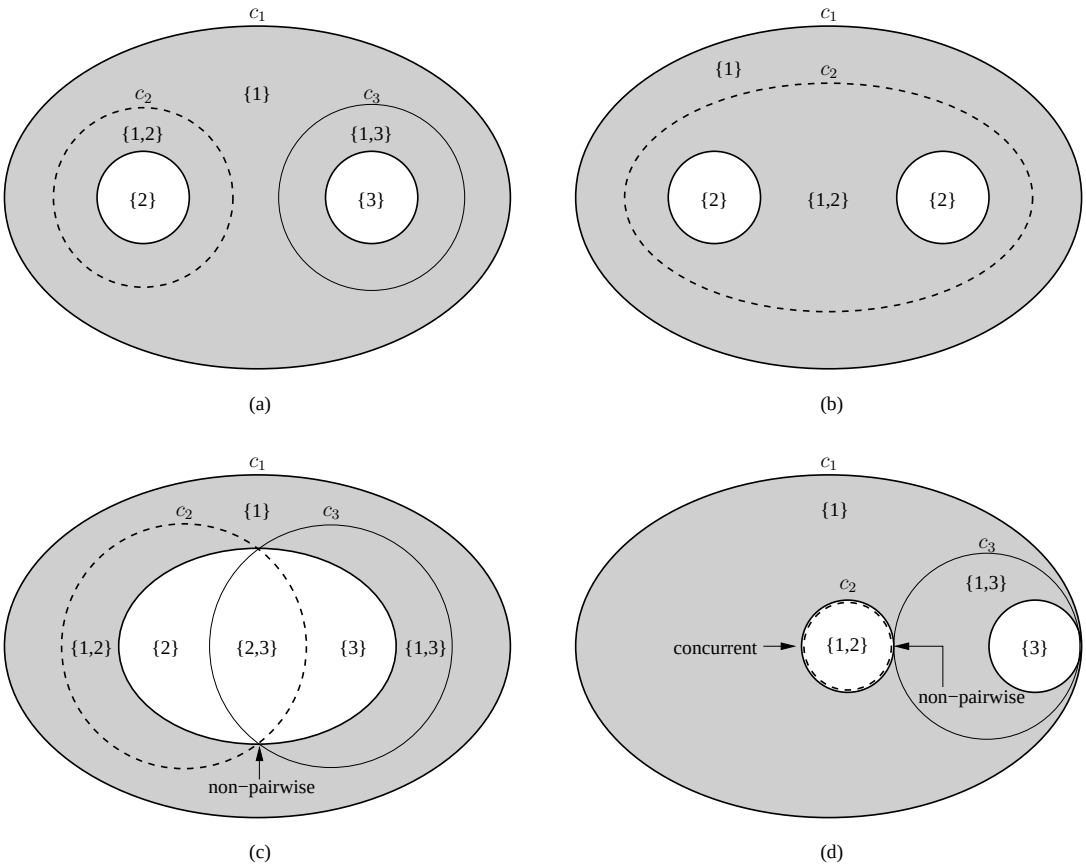


Figure 1.16: Examples of Euler-like diagrams (except (b)), which have the interior of c_1 shaded to indicate its "holes". (a) is an Euler-like diagram that is disconnected, non-concurrent, and pairwise, (b) is not an Euler-like diagram because $r(2)$ is disconnected, (c) is an Euler-like diagram that is disconnected, non-concurrent, and non-pairwise, and (d) is an Euler-like diagram that is connected, concurrent, and non-pairwise.

- based on the examples Euler provided (with, unfortunately, a noticeably lacking formal definition), we believe the definition of Euler diagram to be closer to Euler's original intent than Euler-like diagrams, and certainly more in-line with Venn's definition of Venn diagrams, and
- the visual complexity of Euler diagrams is proportional to the number of items (i.e., an n-Euler diagram has n Jordan curves), whereas the potentially disconnected boundaries of an Euler-like diagram's subsets could result in exponentially more Jordan curves (e.g., compare Fig. 1.12 to Fig. 1.16(a)).

Now that we have some formal definitions and a framework for describing Euler and Euler-like diagrams, we can proceed to the next chapter in which we consider how previous Euler and Venn diagram research applies to this dissertation. An effort has been made to express the existing research in terms of the formalisms established by this section, but where not possible, the differences will be noted.

Chapter 2

Previous and Related Work

In this chapter, we survey existing research related to the problem of Euler diagram generation and area-proportional drawing. We compare and contrast previous work to the results of this dissertation and describe how they influenced our work. In some sections, we may need to refer to results not yet presented in this dissertation; in these cases, the reader is encouraged to skim the section and return to it after reading the relevant parts of the dissertation.

The reader is also forewarned that previous research may refer to certain combinatorial objects as Venn or Euler diagrams, but these objects may not necessarily conform to the formal definitions we provided in Section 1.2. In the research, there is more agreement about the definition of Venn diagrams than Euler diagrams, most likely due to the fact that Venn provided a general construction for his namesake diagrams, while Euler presented his Eulerian circles "by example". As a result, researchers have been left to extrapolate a general definition of Euler diagrams from the few examples of two and three set diagrams that Euler provided. Our primary

motivation was to define Euler diagrams so that the most common definition for Venn diagrams was a special case (i.e., that in which all possible combinations of the sets are represented). In the following sections, we will clearly explain how any previous definitions of Venn and Euler diagrams differ from our definitions and the resulting implications of these differences.

2.1 n-Venn Diagram Constructions

When John Venn introduced the concept of Venn diagrams over a century ago [46], he included a "proof-by-construction" that n-set Venn diagrams exist for all $n \geq 1$. For $1 \leq n \leq 3$, Venn used the standard circle representation of n-set Venn diagrams. For $n = 4$, Venn wove a curve through the circles so that each region was bisected exactly once. For $n > 4$, Venn used an iterative approach whereby each subsequent curve followed a path along the inside and outside of the previous curve. Figure 2.1 shows an example of Venn's general construction of n-set Venn diagrams for $1 \leq n \leq 5$.

Although proving the existence of n-set Venn diagrams, the diagrams that result form Venn's construction lack an aesthetic appeal and are difficult to decipher. In 1989, Anthony Edwards [10] developed an alternative iterative construction for n-set Venn diagrams that produced more symmetrical and easier-to-read diagrams than Venn's construction. In Edwards' construction, the n^{th} curve, $n \geq 4$, weaves around the central circle bisecting all regions. Figure 2.2 shows an example of Edwards' general construction of n-set Venn diagrams for $1 \leq n \leq 5$; contrast this with Venn's construction in Fig. 2.1.

Our ultimate goal would be to develop Euler diagram analogs of Venn and Ed-

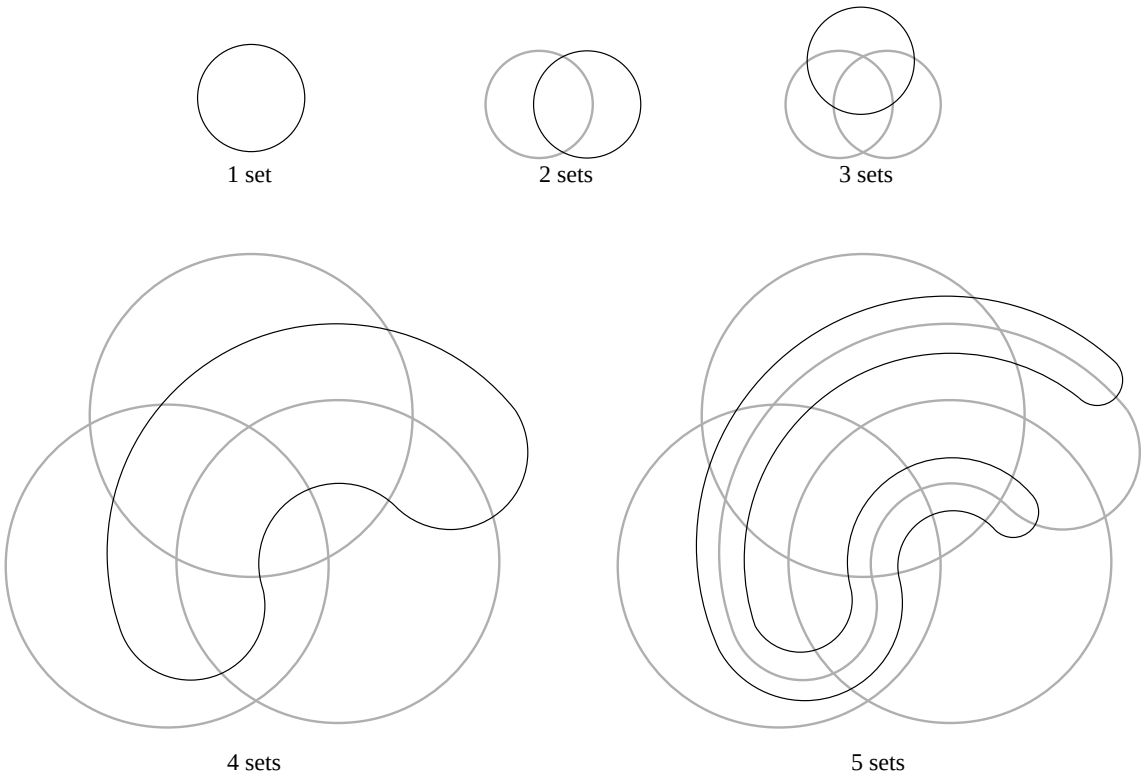


Figure 2.1: John Venn's iterative construction of n-set Venn diagrams for $n = 1, 2, \ldots, 5$.

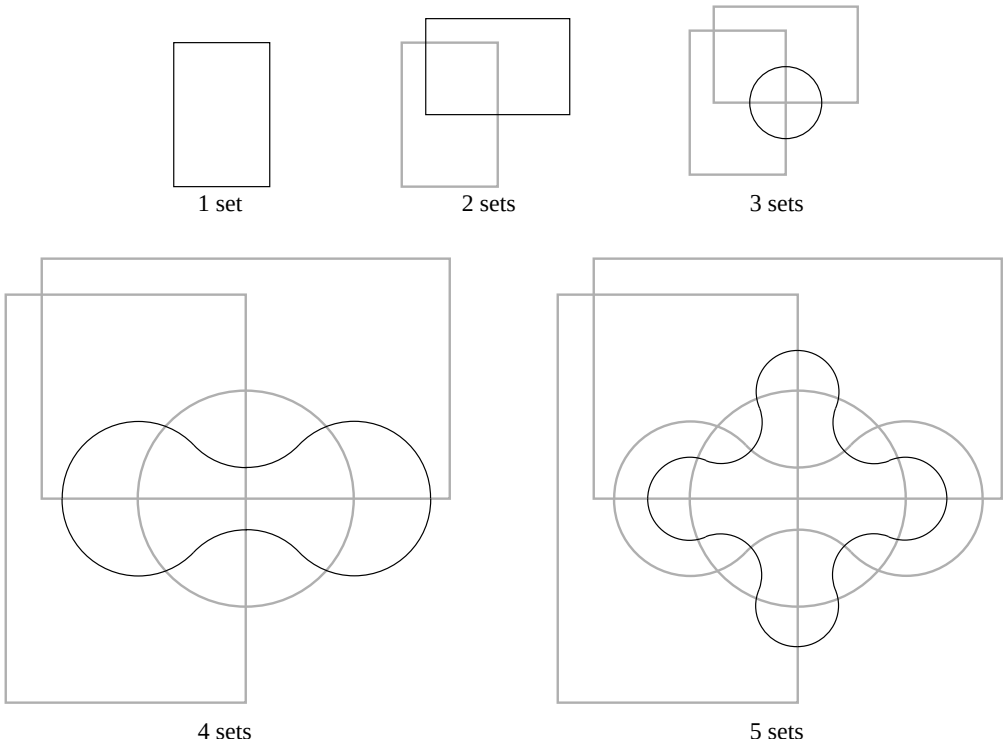


Figure 2.2: Anthony Edwards' iterative construction of n-set Venn diagrams for $n = 1, 2, \ldots, 5$.

wards' constructions that could be used as solution to both the regular and area-proportional Euler Diagram Generation Problems (EDGP and ω-EDGP); however, as we shall see in Chapter 7, an efficient solution to this problem is unlikely to exist. That being said, in Chapter 9, we show how Edwards' construction can be used as the starting point for an area-proportional drawing algorithm that can be applied to a large subclass of Euler diagrams.

2.2 Shape-constrained Venn Diagrams

In a series of articles [24, 25, 26, 27, 28], Grünbaum popularized the question, "What curve shapes can be used to create Venn diagrams?". For example, is there are 4-set Venn diagram composed of circles? The following is a simple argument that proves no four circle Venn diagram can exist: a 4-set Venn diagram composed of circles can have at most $\binom{4}{2}2 = 12$ intersections since any two circles intersect at most twice; however, when viewed as a graph, the diagram must satisfy Euler's formula ($|V| - |E| + |F| = 2$ with $|E| = 2|V|$ and $|F| = 2^4$), which implies the need for 14 points of intersection. In contrast, Fig. 2.3 shows that a 4-set Venn diagram does exist if ellipses are used in lieu of circles.

There are many results and open problems related to shape-constrained Venn diagrams, and the interested reader is referred to Ruskey and Weston's comprehensive survey of Venn diagrams [41] for further information. In Chapter 3, we consider the area-proportional variant of shape-constrained diagrams and develop several efficient algorithms for generating area-proportional Venn and Euler diagrams for up to three sets.

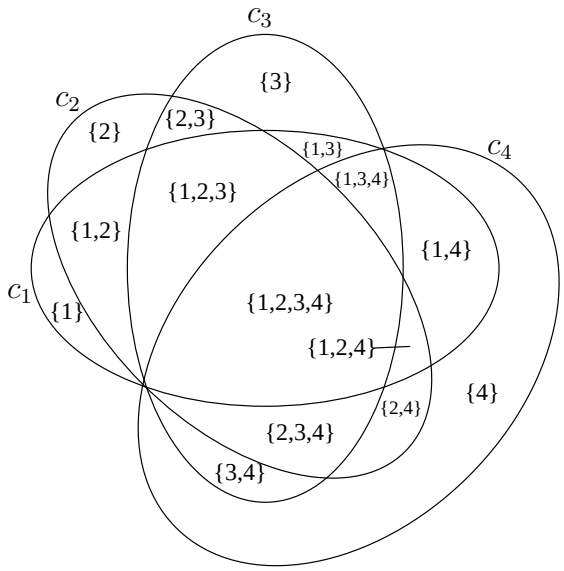


Figure 2.3: A 4-set Venn diagram composed of ellipses.

2.3 Topological Inference and String Graphs

Suppose A, B, and C are connected plane subsets with Jordan curve boundaries, and we are told that A is inside B and that B and C are disjoint. How is A related to C? The study of how to deduce complete topological relationships given partial information is known as topological inference and has applications in the areas of geographic information systems (GIS), spatial databases, and circuit layout, amongst others [23, 44].

Given a topological expression like the one that began this section, Grigni et al. [23] identified two interacting subproblems: *relational consistency* and *planarity*. The relational consistency problem is similar to satisfiability for Boolean expressions, and its purpose is to determine whether the topological expression "makes sense". For example, the expression "A is inside B, B is inside C, and A and C are disjoint" is not relationally consistent because the first two clauses *infer* that A is inside C.

The planarity problem is more subtle in that a topological expression may be relationally consistent, but not topologically realizable. For example, Fig. 2.4(a) shows a graph of a topological expression: each vertex represents a connected plane subset with a Jordan curve boundary, and subsets overlap if and only if their respective vertices are adjacent. The topological expression is relationally consistent (i.e., nothing can be inferred that would make the statement unsatisfiable), but any attempt to draw it in the plane as shown by Fig. 2.4(b) is futile; the relationships indicated by solid edges can be realized, but trying to draw the relationships between subsets A, B, and C as indicated by the dashed edge necessitates having an overlap between subsets that are supposed to be disjoint. On the other hand, one might think that a topological expression has a drawing only if its graph is planar; this is false as shown by the graph and drawing in Fig. 2.5 (note that the additional overlap that was disallowed by the previous graph *is allowed* by this graph, and that there may be multiple disconnected overlaps between the same pair of subsets).

The previous examples show that not all topological expressions have planar realizations, so is there an efficient algorithm for determining when a planar realization is possible? In general, a topological expression may contain up to eight types of topological relationships between each pair of subsets (e.g., "inside" and "equals") [11]. To make the problem scope manageable, researchers have limited the topological relationships to just "overlaps" and "disjoint" and refer to the planar realization of such a topological relationship as an "Euler" diagram; however, as shown in Fig. 2.5(b), such a diagram does not meet our definition of an Euler diagram since there are many disconnected regions. Sinden [44] showed that the problem of determining whether or not one of these limited topological expressions has a planar realization

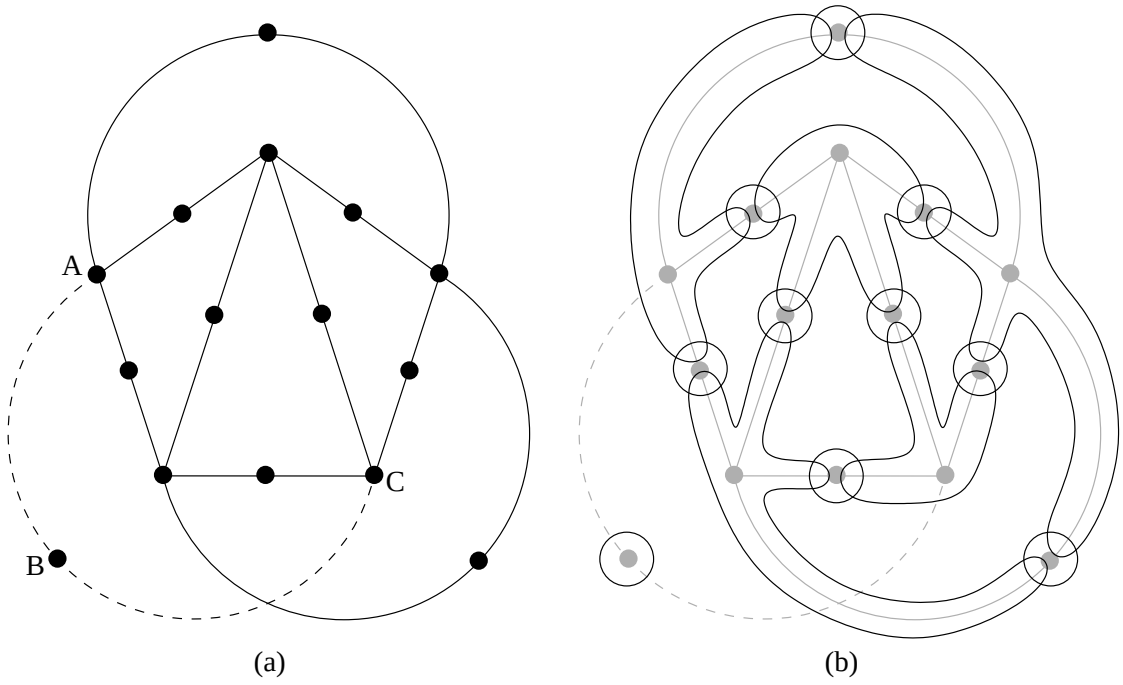


Figure 2.4: (a) A graph of a topological expression where each vertex represents a connected plane subset with a Jordan curve boundary and adjacent subsets overlap while non-adjacent subsets are disjoint, and (b) an attempt at a planar realization of the expression cannot succeed because the additional overlaps specified by the dashed edge result in a disallowed overlap.

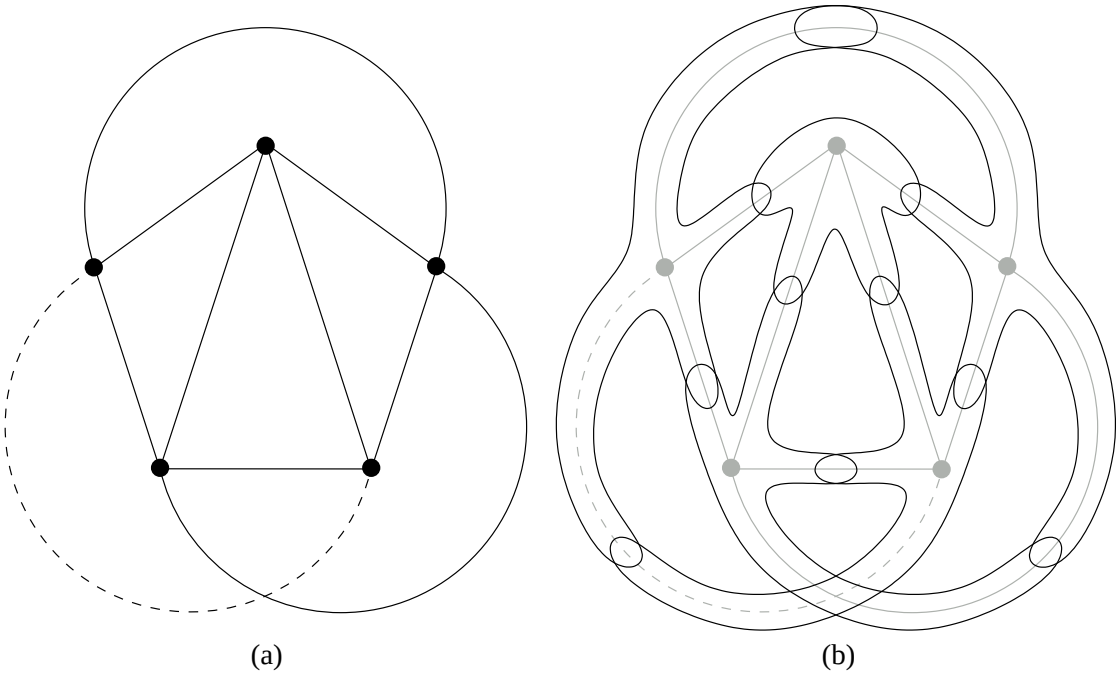


Figure 2.5: (a) A graph of a topological expression where each vertex represents a simply connected plane subset and adjacent subsets overlap while non-adjacent subsets are disjoint, and (b) a planar realization of the expression.

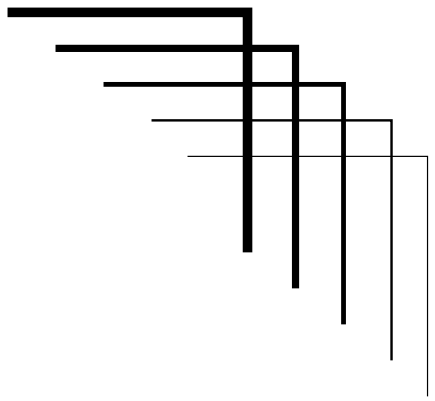


Figure 2.6: A set of five continuous curves (strings), whose intersection graph is K_5.

is equivalent to the *string graph* problem.

A string graph is the intersection graph [12] of a set of continuous open curves (strings) in the plane. For example, Fig. 2.6 is a drawing of five strings whose intersection graph is K_5 since all pairs of strings intersect. The string graph problem involves determining whether a specified graph is isomorphic to the intersection graph of a set of strings. For example, the graph in Fig. 2.4(a) is *not* a string graph because it impossible to draw 15 strings that intersect in the prescribed way (for the same reason that this graph did not represent a topological expression with a planar realization).

The string graph problem was first introduced in 1966 by Sinden [44] as a consequence of studying integrated circuit layout, and popularized as a combinatorial problem by Graham in 1976 [21]. In 1991, Kratochvíl and Jiří Matoušek [35] conjectured that any string graph could be realized by a drawing with an exponential number of intersections, but it was not until 2001 that Schaefer and Štefankovič [43] and Pach and Tóth [40] independently proved this upper bound, and thus that the string graph problem is decidable. Shortly afterwards, Schaefer et al. [42] finally

proved that the string graph problem is NP-complete.

Although there has been considerable study of string graphs, as we previously mentioned, their equivalency to "Euler" diagrams is not directly applicable because of the less-strict definition. In fact, because string graphs focus on the intersection of curves while Euler diagrams focus on the intersection of connected plane subsets, it is not evident how to apply the string graph results to Euler diagrams. As a result, we note the work in string graphs, but do not use any string graph results in this dissertation.

2.4 Hypergraph Planarity

A hypergraph $H = (V, E)$ is a generalization of a graph where, rather than being a pair of vertices, each edge is a subset of vertices. Unlike the singular notion of graph planarity, there are several definitions of planarity for hypergraphs depending on the chosen planar representation. Johnson and Pollak [33] defined two types of hypergraph planarity based on dual interpretations of an n-set Venn diagram; these interpretations are best explained through an example.

Example. Consider the 3-set Venn diagram C in Fig. 2.7(a) and define the hypergraph $H = (V, E)$ with

$$\begin{aligned} V &= \{1, 2, 3\} \text{ and} \\ E &= \{\{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}\}. \end{aligned}$$

If we represent each hyperedge by a region and each hypergraph vertex v by a

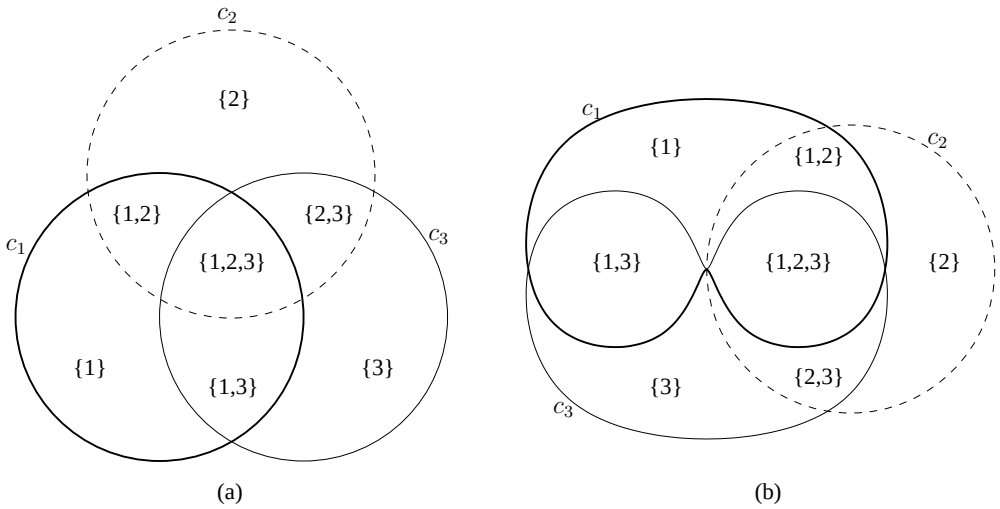


Figure 2.7: Examples of 3-set Venn diagrams.

Jordan curve enclosing the regions corresponding to hyperedges containing v, then C can be interpreted as a hyperedge-based planar representation of H.

Conversely, suppose we define the hypergraph $H' = (V', E')$ with

$$\begin{aligned} V' &= \{\{1\},\{2\},\{3\},\{1,2\},\{1,3\},\{2,3\},\{1,2,3\}\} \text{ and} \\ E' &= \{\{\{1\},\{1,2\},\{1,2,3\}\}, \\ &\quad \{\{2\},\{2,3\},\{1,2,3\}\}, \\ &\quad \{\{3\},\{1,3\},\{2,3\},\{1,2,3\}\}\}. \end{aligned}$$

If we represent each hypergraph vertex by a region and each hyperedge e by a Jordan curve enclosing the regions corresponding e's vertices, then C can be interpreted as a vertex-based planar representation of H'. □

The following definitions formalize these two interpretations of a Venn diagram. The reader is forewarned that the Venn diagrams referred to by these definitions *are*

not the same as the Venn diagrams defined in this dissertation; Johnson and Pollak's Venn diagrams are actually closer to Euler diagrams, but we will explore this issue after the definitions are given.

Definition 2.4.1 (Definitions 1 and 2 from Johnson and Pollak [33])**.** Given a hypergraph $H = (V, E)$, a *hyperedge-based Venn diagram representing* H consists of a plane graph G and a one-one map from the set E of hyperedges of H to the set of faces of G, such that for each $V' \subseteq V$, the union of the faces corresponding to hyperedges of H containing V' has a connected interior.

A hypergraph is *hyperedge-planar* if there is a hyperedge-based Venn diagram that represents it. □

Definition 2.4.2 (Definitions 3 and 4 from Johnson and Pollak [33])**.** Given a hypergraph $H = (V, E)$, a *vertex-based Venn diagram representing* H consists of a plane graph G and a one-one map from the set V of vertices of H to the set of faces of G, such that for each hyperedge $e \in E$, the union of the faces corresponding to the vertices in e has a connected interior.

A hypergraph is *vertex-planar* if there is a vertex-based Venn diagram that represents it. □

The concepts of hyperedge-planarity and vertex-planarity are distinct; as Johnson and Pollak showed [33, Section 3], there exist hypergraphs that are hyperedge-planar, but *not* vertex-planar, and vice versa.

How do hyperedge-based and vertex-based Venn diagrams compare to Euler diagrams? We must take care when comparing these diagrammatic concepts because they represent differing domains (i.e., hypergraphs vs. set systems). That being said,

there is a natural bijection between hypergraphs and set systems; each hyperedge is a set of vertices in the same way that each set system element is a set of items.

Definition 2.4.3. Let φ : set system $\rightarrow$ hypergraph be the bijection

$$\varphi(S) = (X(S), S\backslash\{\emptyset\})$$

where S is a set system on the items $X(S)$ and $(X(S), S\backslash\{\emptyset\})$ is a hypergraph with vertex set $X(S)$ and edge set $S\backslash\{\emptyset\}$. □

In the previous example, we used φ to derive the hypergraph H from the 3-set Venn diagram. Through φ, we can directly compare hyperedge-based and vertex-based Venn diagrams to Euler diagrams; that is, we can consider which set systems each diagram type can represent (we call this the diagram type's *representability class*). The following propositions show that hyperedge-based and vertex-based Venn diagrams are distinct from Euler diagrams in the sense that neither has a representability class that is a subset of the Euler diagram representability class, and vice versa. We leave the proofs for the appendix because they are based on concepts that will be covered in later chapters.

Proposition 2.4.1. *There exists a set system S that is representable by an Euler diagram, but for which $\varphi(S)$ is* not *hyperedge-planar.*

Proof. See Appendix A.2.4, pg. 255. □

Proposition 2.4.2. *There exists a set system S that is* not *representable by an Euler diagram, but for which $\varphi(S)$ is hyperedge-planar.*

Proof. See Appendix A.2.4, pg. 256. □

Proposition 2.4.3. *There exists a set system S that is representable by an Euler diagram, but for which $\varphi(S)$ is* not *vertex-planar.*

Proof. See Appendix A.2.4, pg. 258. □

Proposition 2.4.4. *There exists a set system S that is* not *representable by an Euler diagram, but for which $\varphi(S)$ is vertex-planar.*

Proof. See Appendix A.2.4, pg. 260. □

The previous example showed how a 3-set Venn diagram could be interpreted as both a hyperedge-planar and vertex-planar Venn diagram (with respect to different hypergraphs). In the hyperedge-planar interpretation of the 3-set Venn diagram, each Jordan curve c_v represents the hypergraph vertex v and encloses the regions corresponding to the hyperedges containing v; looking at this the other way, the union of the regions corresponding to hyperedges containing v is a connected plane subset whose boundary is a Jordan curve. However, according to Def. 2.4.1, the union of the faces corresponding to hyperedges of H containing $V' \subseteq V$ form a region of the plane whose interior is connected. If we let $V' = \{v\}$ then in a hyperedge-based Venn diagram, the union of regions corresponding to hyperedges containing v need only be a connected plane subset and not *necessarily* have a Jordan curve boundary. As a result, hyperedge-based Venn diagrams are similar to the Euler-like diagrams described in Section 1.2.3. In Chapter 6, we study the representability classes for Euler and Euler-like and conclude the following:

Proposition 2.4.5. *The representability class for Euler diagrams is a* proper subset *of the representability class for Euler-like diagrams.* □

In other words, Euler-like diagrams are more *expressive* than Euler diagrams, so in a sense hyperedge-base Venn diagrams are less restrictive than Euler diagrams; however, in another sense they are more restrictive. Not only does Def. 2.4.1 state that the union of regions corresponding to hyperedges containing each vertex must be connected, but so must the union of regions corresponding to hyperedges containing any *subset* of vertices. How does this translate to an Euler diagram $C = \{c_1, c_2, \ldots, c_n\}$? As we know from the previous paragraph, the requirement for the union of regions corresponding to hyperedges containing a vertex is analogous to the requirement that the interior of c_x be connected, and this is an implied part of the Euler diagram definition since c_x is a Jordan curve. The requirement for the union of regions corresponding to hyperedges containing any subset of vertices is analogous to the requirement that the *intersection* of the interiors of any subset of curves be connected, and this is *not* part of the Euler diagram definition.

Suppose we were to strengthen the definition of an Euler diagram so that the intersection of the interiors of any subset of curves had to be connected. Does this added restriction further limit which set systems can be represented? The following proposition states the answer to this question.

Proposition 2.4.6. *The representability class for Euler diagrams in which the intersection of the interiors of any subset of curves is connected is a* proper subset *of the representability class for Euler diagrams.*

Proof. See Appendix A.2.4, pg. 262. □

In other words, this *subset intersection restriction* is a restriction on representability and not merely aesthetic. The previous propositions show that Johnson and Pollak's notion of hyperedge-based and vertex-based Venn diagrams differs from our notion of Euler diagrams, but there are enough similarities that, with modification, some of their results can be applied to Euler diagrams. In particular, Johnson and Pollak proved that the decision problems corresponding to hyperedge-planarity and vertex-planarity (i.e., is a given a hypergraph H hyperedge(vertex)-planar?), are NP-complete. In Chapter 7, we use a variation of Johnson and Pollak's NP-complete reduction to prove that the decision problem corresponding to Euler diagrams (i.e., is a given set system S representable by an Euler diagram?), is also NP-complete.

2.5 Euler-like Diagram Generation

In France, the Institut National de l'Audiovisuel (INA) has a mandate to archive the nation's audio-visual productions and provide a publicly-accessible catalogue via the Internet. In collaboration with France's main computer science research institute, INRIA, researchers at INA are investigating the usage of Euler diagrams for database query visualization. For example, suppose someone wished to search INA's database for newspaper articles containing the keywords "Paris", "World War II", and "Peace". A traditional database query would either specify a conjunction or disjunction of the keywords, in which case only those articles containing *all* or *any*, respectively, of the keywords would be returned. As shown in Fig. 2.8, a 3-Venn diagram can be used to show the results of all possible combinations of the keywords, including the conjunction and disjunction, simultaneously; in this example, each region is labeled

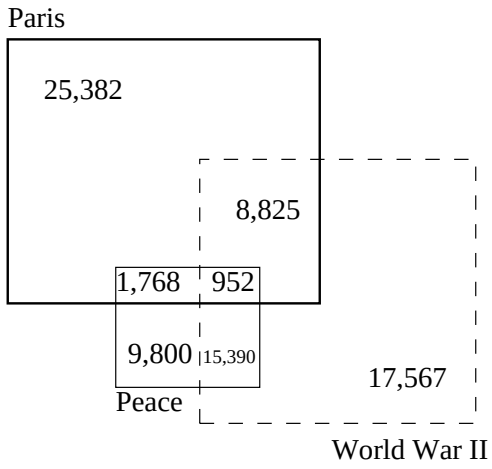


Figure 2.8: A non-area-proportional 3-Venn diagram showing the result of a database query for all possible conjunctions of the keywords "Paris", "World War II", and "Peace".

with the number of articles that exclusively contain all the keywords represented by the enclosing curves. As a result, the conjunctive results are represented by region $\{\text{Paris}, \text{World War II}, \text{Peace}\}$ and the disjunctive results are represented by the union of all regions. Visualizing database queries with Euler diagrams gives the searcher a sense of the distribution and density of database content, as well as providing the opportunity to perform peripheral searches (e.g., to focus on specific keyword combinations).

At present, and as shown in Fig. 2.8, INA/INRIA have not yet considered area-proportional drawings of query results; their focus has been on generating the underlying Euler diagram. That being said, the Set'Graph company (`www.setgraph.com`) has developed a similar database query visualization tool that employs area-proportional Venn diagrams composed of rectangles. Set'Graph's technology is limited to 3-Venn diagrams and portions are patented, so no further information is available concerning how their algorithms work.

With respect to Euler diagram generation, the main researchers, Verroust and Viaud [47], have developed a very nice iterative construction for what they refer to as Extended Euler Diagrams (EEDs). EEDs are equivalent to Euler-like diagrams (i.e., they allow the connected plane subsets representing each item to have "holes"); as a result, more than n Jordan curves may appear in an n-EED. A consequence of the EED construction algorithm is that Verroust and Viaud were able to prove that there exists an EED/Euler-like diagram for *any* set system on less than 9 items. Verroust and Viaud also showed that the existence of EEDs is equivalent to the existence of vertex-planar hypergraphs (Corollary 3 of [47]), so the results of the previous section suffice to show that EEDs and Euler diagrams are distinct (the nature of their differences will be explored further in Chapter 6).

2.6 Euler Diagram Generation

The most extensive work in Euler diagram generation was performed by a collaboration of UK-based researchers at the Universities of Brighton and Kent as part of a 3-year project (2002-2005) entitled "Reasoning With Diagrams (RWD)". The purpose of the RWD project was to develop methods "... to reason with a combination of diagrammatic and textual constraint notations, in the context of modeling software intensive systems." [39]. In particular, the researchers were interested in combining Euler diagrams with a symbolic logic language to create software constraint diagrams [32]. For example, Fig. 2.9 shows a disconnected 5-Euler diagram representing objects and states in a library loan system; the additional markings specify object properties and state transitions.

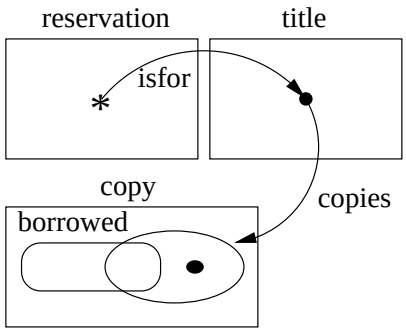


Figure 2.9: An example constraint diagram adapted from Fig. 1 of [14].

In order to draw a constraint diagram, one needs an underlying Euler diagram, and so Euler diagram generation was one of the first topics addressed by RWD researchers Flower and Howse [14]. Flower and Howse limited their investigation to Euler diagrams meeting certain "well-formedness" criteria; such diagrams are equivalent to simple Euler diagrams with the additional restriction that only *transverse* intersections are allowed (i.e., concurrent curves, non-pairwise intersections, and tangential pairwise intersections are disallowed). Using the terminology of Section 1.2, Flower and Howse's algorithm generates an Euler diagram C from a set system S via the following steps:

1. construct the closeness graph $G_c(S)$ (see Def. 5.2.1)
2. search for a connectivity graph G for S (see Def. 5.0.1) that is a subgraph of $G_c(S)$ and has a plane embedding G_e satisfying certain face conditions (Cor. 2 of [14]),
3. using the circularization algorithm, draw a plane dual of G_e to realize the Euler diagram.

Flower and Howse's algorithm embodies some of the same concepts that we describe in Sections 5.2 and 5.4; particularly, the use of the closeness graph to limit the search space for G_e. The face-cycle conditions that they describe are simplified

in Thm. 5.4.1 because we do not differentiate between transverse and tangential pairwise intersections.

Finding a connectivity subgraph is the fundamental step of any generation algorithm that begins with a closeness graph; in Chapter 7, we conjecture that this is an NP-complete problem. As a result, Flower and Howse state the following about Step 2 of their algorithm, "In the current implementation of the algorithm, an iterative planarising step is used which is not optimal, but works in small cases."

The final step, circularization, is a graph drawing algorithm where a new, but topologically equivalent, embedding of G_e is drawn and its plane dual constructed. In part of this dissertation, we study how to generate the topological structure of Euler diagrams irrespective of area-proportionality; however, we do not consider the related drawing problem except in the context of area-proportionality. The reason for this decision is based on Thm. 5.0.4; according to this theorem, *every* plane dual of a graph meeting certain criteria (for which G_e is such a graph), produces the desired Euler diagram either directly or via a stereographic projection. As a result, many existing planar graph drawing algorithms [38] may be used to produce an Euler diagram from G_e.

In further work [16], Flower et al. developed a hill-climbing optimizer that transforms Euler diagrams produced by the circularization algorithm into (hopefully) more aesthetically-pleasing drawings. The optimizer is based on maximizing a diagram's "fitness" according to various criteria (e.g., curve smoothness, curve convexity, balanced region areas, etc.), that are believed to contribute to the diagram's readability. Subsequently, Benoy and Rodgers [3], conducted a study that focused on three of the aesthetic criteria and found that the quality of each contributed, in varying degrees,

to Euler diagram readability.

If one is able to measure the quality of a diagram with respect to a particular criteria, then this criteria can be fairly easily added to the optimizer. One obvious choice is to measure the amount of deviation between the areas of an Euler diagram's regions and a prescribed weight function ω; the optimizer could then be used to transform an Euler diagram into an (approximately) ω-proportional Euler diagram. Like cartograms, which we consider in the next section, this technique requires a solution to the (as yet unsolved) Euler Diagram Generation Problem (EDGP). However, in collaboration with one of the RWD researchers, Peter Rodgers, we did successfully apply a similar optimization strategy for ω-proportional 3-Venn diagrams composed of circles; this work is described in Section 3.3.

Flower et al. [15] also considered Euler diagrams with a natural nested structure as shown in Fig. 2.10; they developed criteria for identifying a nested Euler diagram based on the closeness graph of the represented set system and studied the semantics of logic systems represented by nested Euler diagrams. In Chapter 8, we consider a similar, but broader class of structured Euler diagrams called *composite* Euler diagrams. Because we are interested in Euler diagrams that may not be well-formed by Flower and Howse's definition, we develop algorithms for identifying composite Euler diagrams on the basis of set systems rather than closeness graphs, which have an implied non-concurrent Euler diagram restriction.

Whereas the RWD research focused on well-formed Euler diagrams with significant restrictions, we have approached the Euler diagram generation problem in as general a fashion as possible, and only considered restricted instances when necessary or beneficial/interesting. As such, many of our results, particularly those in Chapter

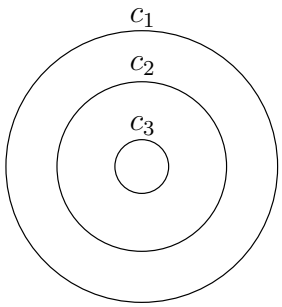


Figure 2.10: A nested Euler diagram as described in [15].

5 can be thought of as extensions and generalizations of the RWD work in Euler diagram generation.

2.7 Cartograms

A geographic map is a scale drawing of a land mass where certain geographic regions (e.g., Canadian provinces) are delineated. Additional data may be integrated into a map via colour, symbols, or labels (e.g., shading the provinces according to their average annual temperatures). An areal cartogram (from now on, simply cartogram), is a transformation of a map such that the geographic regions have areas that are proportional to some quantity other than physical area. For example, Fig. 2.11(a) shows a map of Canada and Fig. 2.11(b) shows the map's transformation to a cartogram where the provinces and territories are scaled by population.

There are three main types of computer-generated cartograms: non-contiguous, Dorling, and contiguous [18]. In a non-contiguous cartogram, the geographic regions are separated from each other, resized, and then positioned so that they are near their original locations. In a Dorling cartogram, the map is completely abstracted by replacing each geographic region with a uniform shape (e.g., a circle or rectangle),

(a)

(b)

Figure 2.11: (a) A map of Canada's provinces and territories and (b) an associated contiguous cartogram scaled according to 1996 populations. The cartogram was created with Frank Hardisty's Cartogram Generator [6] based on algorithms by Gastner and Newman [20] and "Nunavut" map data by ESRI Canada [29].

that is resized and then positioned to create a diagram that looks similar to the original map. In a contiguous cartogram, each geographic region is transformed to the necessary size while maintaining the map's overall topology (i.e., a geographic region cannot be moved so that it loses or acquires new neighbours). The requirement for preserving topology can cause contiguous cartograms to become distorted and no longer resemble the original map; the goal of contiguous cartogram algorithms is to limit the distortion so that recognizable features of the original map are preserved. The cartogram in Fig. 2.11(b) is an example of a contiguous cartogram.

Of the cartogram types, contiguous cartograms most resemble area-proportional Euler diagrams. It is conceivable that given an Euler diagram C and a weight function ω, C could be transformed by a contiguous cartogram algorithm into an ω-proportional diagram; however, this strategy has a number of problems.

First, by their nature, contiguous cartogram algorithms are region-centric, and beyond having a bias towards preserving the recognizability of the input map, they have no notion of the Jordan curves that form an Euler diagram. As a result, out-of-the-box, they would not be suitable for shape-constrained drawings, and as we shall see in Chapter 3, a priori knowledge of the curve shapes allows us to create efficient area-proportional Euler diagram generation algorithms.

Second, contiguous cartogram algorithms are computationally intensive; they require several iterations, each involving the computation of a solution to a system of many differential equations. For example, the contiguous cartogram in Fig. 2.11(b) took 15 minutes to compute on a Centrino 1.7GHz processor. In contrast, many of the algorithms we develop in this dissertation are fast enough to allow a user to dynamically vary the weight function and generate an updated area-proportional Euler

diagram in real-time.

Third, and most importantly, we must have a solution to the Euler Diagram Generation Problem (EDGP) in order to produce an input diagram for the cartogram algorithm. As we shall see in Chapter 9, there can also be a benefit to considering the topology generation and area-proportional drawing of an Euler diagram simultaneously. In the meantime, the interested reader is referred to Tobler's excellent summary of the history of computer-generated cartograms [45].

2.8 Families of Intersecting Simple Curves

Bultena et al. [4] studied methods for drawing Families of Intersecting Simple Curves (FISCs). A FISC is similar to an Euler diagram in that it is a set $C = \{c_1, c_2, \ldots, c_n\}$ of Jordan curves, but it differs in that region $\{1, 2, \ldots, n\}$ *must* be non-empty while other regions may be disconnected. Bultena et al. determined necessary-and-sufficient conditions for when a FISC can be drawn with convex curves and developed an algorithm to produce such drawings. In Chapter 9, we modify the FISC algorithm so that for any weight function ω and any set system S containing the fullset, it can generate an ω-proportional Euler diagram representing S.

Chapter 3

Preliminary Results (diagrams with two or three curves)

In this chapter, we consider the ω-proportional Euler diagram problem (ω-EDGP) when restricted to n-Euler diagrams with $n \leq 3$. These small cases are interesting not only because they are tractable, but also because they are well-suited for the exploration of shape-constrained diagrams. As we shall see, even these small cases lead to open problems and non-trivial algorithms.

Clearly, the ω-EDGP is solvable when restricted to 1-Euler diagrams (each diagram is just a scaled circle), so in the next section we begin by considering the ω-EDGP when $n = 2$.

3.1 Two Circle Euler Diagrams

Figure 3.1 shows an area-proportional 2-Venn diagram whose curves are circles. Given a weight function ω, we can construct such a diagram $C = \{c_1, c_2\}$ by first calculating

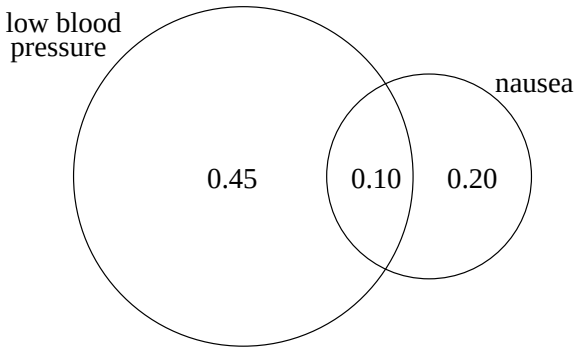


Figure 3.1: An area-proportional 2-Venn diagram showing the probabilities of certain complications for a trial drug.

each circle's radius so that its area equals the sum of the weights of the regions enclosed by the circle:

$$r_1 = \sqrt{\frac{\omega(\{1\}) + \omega(\{1,2\})}{\pi}}, \quad r_2 = \sqrt{\frac{\omega(\{2\}) + \omega(\{1,2\})}{\pi}}\ .$$

Without loss of generality, let $r_1 \geq r_2$; otherwise, we can just flip the diagram across the vertical. We choose a canonical orientation for the diagram by centering c_1 in the cartesian plane at $(0,0)$ and centering c_2 at $(r_1 + r_2, 0)$ as shown in Fig. 3.2(a). In this orientation, there is no overlap between c_1 and c_2; however, as shown in Fig. 3.2(b), if $\omega(\{1,2\}) > 0$, we need to recenter c_2 at $(d,0)$, where d is chosen so that the area of overlap between c_1 and c_2 is $\omega(\{1,2\})$. Since πr_1^2 and πr_2^2 are at least $\omega(\{1,2\})$, there is always a d that yields the necessary overlap.

In order to compute d, we use Euclidean geometry and trigonometry to derive a function $\sigma(d)$ whose value is the area of overlap between c_1 and c_2 for the given distance d between centers:

$$\sigma(d) = \frac{1}{2} r_1^2 \left(\alpha - \sin(\alpha)\right) + \frac{1}{2} r_2^2 \left(\beta - \sin(\beta)\right)$$

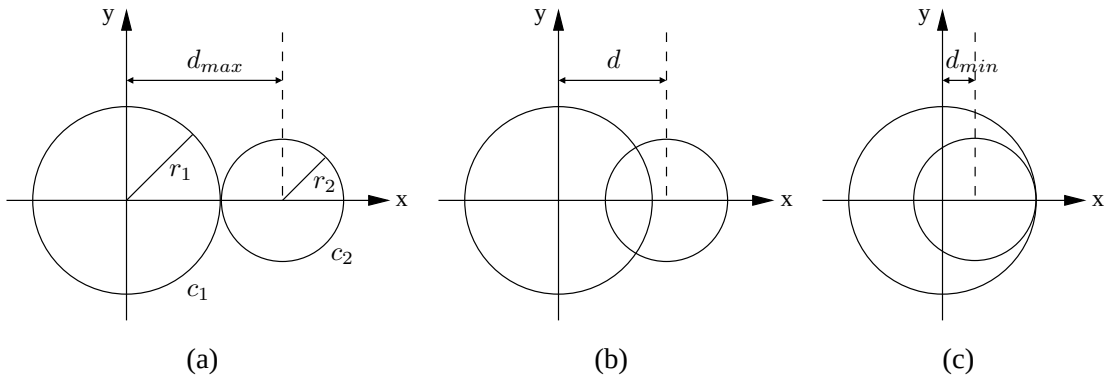


Figure 3.2: In the two circle algorithm, the area of overlap monotonically increases from (a) a minimum at (a) $d = r_1 + r_2$ to (c) a maximum at $d = r_1 - r_2$, and (b) the appropriate value for d is somewhere between these extremes.

where

$$\alpha = 2 \arccos\left(\frac{d^2 + r_1^2 - r_2^2}{2r_1 d}\right) \qquad \beta = 2 \arccos\left(\frac{d^2 + r_2^2 - r_1^2}{2r_2 d}\right)$$

are in radians.

The function σ is not invertible; however, we can use numerical methods to solve $\sigma(d) = \omega(\{1, 2\})$. We first note that $\sigma(d)$ is minimal at $d = r_1 + r_2$, maximal at $d = r_1 - r_2$, and monotonically increasing within this range as shown in Fig. 3.2. Because of σ's monotonicity, we can employ a simple bisection algorithm [5] in the range $[r_1 - r_2, r_1 + r_2]$ to efficiently compute d.

This algorithm can also construct Euler diagrams. If $\omega(\{1\}) > 0$, $\omega(\{2\}) > 0$, and $\omega(\{1, 2\}) = 0$, the algorithm will produce a diagram such as Fig. 3.2(a). If either $\omega(\{1\}) = 0$ or $\omega(\{2\}) = 0$, but not both, and $\omega(\{1, 2\}) > 0$, the algorithm will produce a diagram such as Fig. 3.2(c). Finally, if $\omega(\{1\}) = \omega(\{2\}) = 0$ and $\omega(\{1, 2\}) > 0$, the algorithm will produce two equal-sized circles centered at $(0, 0)$.

The values of r_1, r_2, and d fully specify a $1 : 1$ scale ω-proportional 2-Euler diagram whose curves are circles. Any scaling transformation can be applied to resize

region r	$\omega(r)$
$\{1,2,3\}$	2
$\{1,2\}$	6
$\{1,3\}$	6
$\{2,3\}$	1
$\{1\}$	5
$\{2\}$	9
$\{3\}$	7

Table 3.1: A sample 3-Venn weight function.

the diagram for device output (e.g., printer or monitor), while preserving the ω-proportionality. The following theorem is a direct result of this two circle algorithm.

Theorem 3.1.1. *Let S be a set system on the items X, $|X| = 2$, and ω be a weight function for S. There is an ω-proportional 2-Euler diagram that represents S and whose curves are circles.*

3.2 Three Rectangle Euler Diagrams

In the two circle algorithm, the curves are first constructed and then positioned so as to create the properly-sized regions. For the three rectangle algorithm, we take the opposite approach: the properly-sized/positioned regions are constructed one-by-one and the curves are a resulting byproduct. The three rectangle algorithm is best described through an example, so consider the same weight function ω shown in Tab. 3.1. In the following discourse, when we refer to a "rectangle" or a "6-gon", we implicitly mean an *orthogonal* one (i.e., one whose sides are horizontal and vertical).

Each region r is represented by an orthogonal polygon P_r whose area is $\omega(r)$. We begin by constructing $P_{\{1,2,3\}}$ as a rectangle whose aspect ratio may be arbitrarily

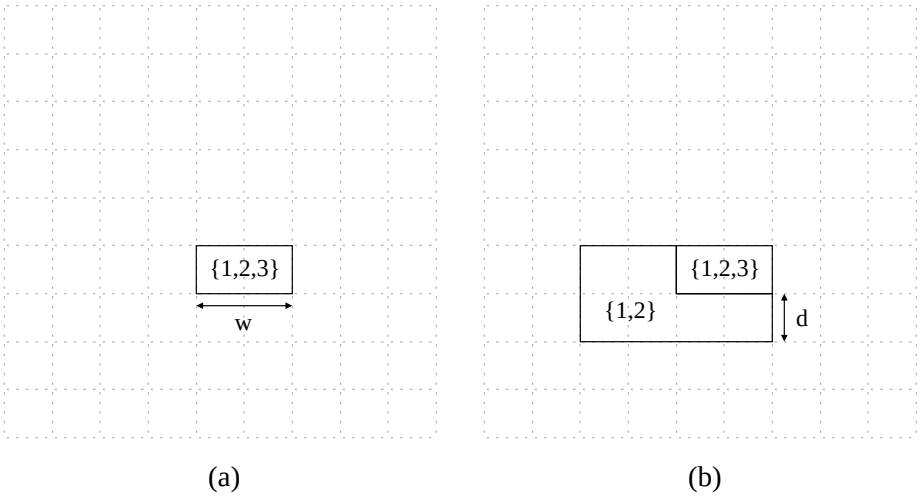


Figure 3.3: In the three rectangle algorithm, (a) the first step is to construct $P_{\{1,2,3\}}$ so that it has arbitrary width w, and (b) the second step is to construct $P_{\{1,2\}}$ so that it extends below $P_{\{1,2\}}$ by the arbitrary (within limits) distance d.

chosen, and we let w be its width as shown in Fig. 3.3(a). Next, we construct $P_{\{1,2\}}$ so that it extends outwards from the left and bottom sides of $P_{\{1,2,3\}}$ as shown in Fig. 3.3(b); the distance d between the bottom of $P_{\{1,2,3\}}$ and the bottom of $P_{\{1,2\}}$ may be arbitrarily chosen as long as it allows $P_{\{1,2\}}$'s area to be $\omega(\{1,2\})$.

We can now construct $P_{\{1,3\}}$ and $P_{\{2,3\}}$ so that they extend outwards from the top and right sides, respectively, of $P_{\{1,2,3\}}$ as shown in Fig. 3.4(a). Note that unlike $P_{\{1,2,3\}}$ and $P_{\{1,2\}}$, there is *no* freedom in how $P_{\{1,3\}}$ and $P_{\{2,3\}}$ are constructed.

At this point, the *core* of the diagram has been constructed. The next step is to construct $P_{\{1\}}$ so that it fills the space between $P_{\{1,2\}}$ and $P_{\{1,3\}}$ as shown in Fig. 3.5. In this example, there are 6 units that need to be filled by $P_{\{1\}}$ in order for c_1 to be rectangle; however, $\omega(\{1\}) = 5 < 6$, so c_1 will *not* be rectangle. That being said, there are numerous ways in which $P_{\{1\}}$ can be constructed. Figure 3.5(a) shows an example where $P_{\{1\}}$ extends directly up from $P_{\{1,2\}}$, and the resulting non-

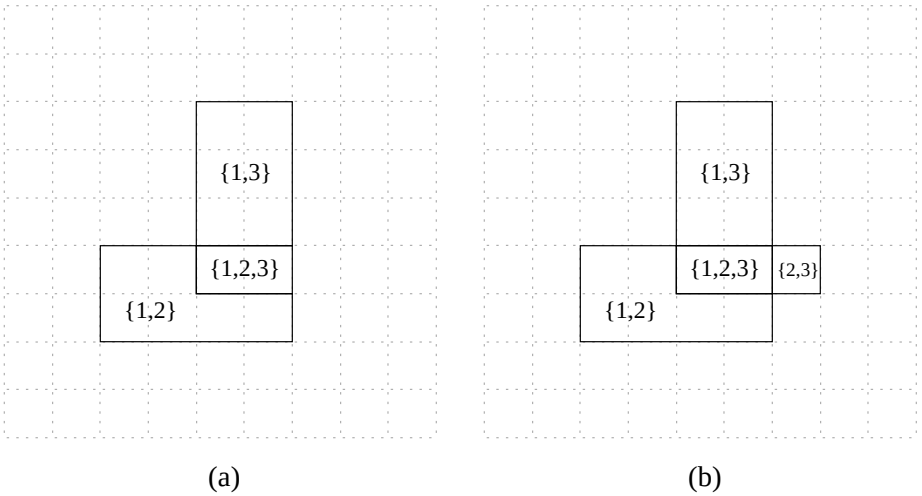


Figure 3.4: Once $P_{\{1,2,3\}}$ and $P_{\{1,2\}}$ are constructed, there is no freedom in how $P_{\{1,3\}}$ and $P_{\{2,3\}}$ are constructed.

rectangular c_1 is shown in thick black. Similarly, Fig. 3.5(b) shows an example where $P_{\{1\}}$ extends uniformly up from $P_{\{1,2\}}$ and outwards from the left and top sides of $P_{\{1,3\}}$. In both cases, c_1 is non-rectangular; however, in the former, there is a concurrent curve segment along the top side of $P_{\{1,3\}}$, so we favour the later.

In the case of $P_{\{3\}}$, $\omega(\{3\}) = 7$ is large enough to fill in the 3 units between $P_{\{1,3\}}$ and $P_{\{2,3\}}$, so we just extend $P_{\{3\}}$ outwards from the right sides of $P_{\{1,3\}}$ and $P_{\{2,3\}}$ as shown in Fig. 3.6(a). If $\omega(\{3\}) \leq 3$, we would opt for a similar construction as was used for $P_{\{1\}}$ and uniformly extend it outwards.

The construction of $P_{\{2\}}$ is slightly different than for $P_{\{1\}}$ and $P_{\{3\}}$ because $P_{\{2\}}$ needs to extend outwards from the left side of $P_{\{1,2\}}$ in order to prevent a concurrent curve segment. However, there is still the requirement that $\omega(\{2\})$ be large enough to fill in the 1 unit between $P_{\{2,3\}}$ and $P_{\{1,2\}}$. In this example, $\omega(\{2\}) = 9$ is sufficient to allow $P_{\{2\}}$ to extend outwards from the bottom of $P_{\{2,3\}}$ and $P_{\{1,2\}}$ and an arbitrary distance outwards from the left side of $P_{\{1,2\}}$ as shown in Fig. 3.6(b). If $\omega(\{2\})$ were

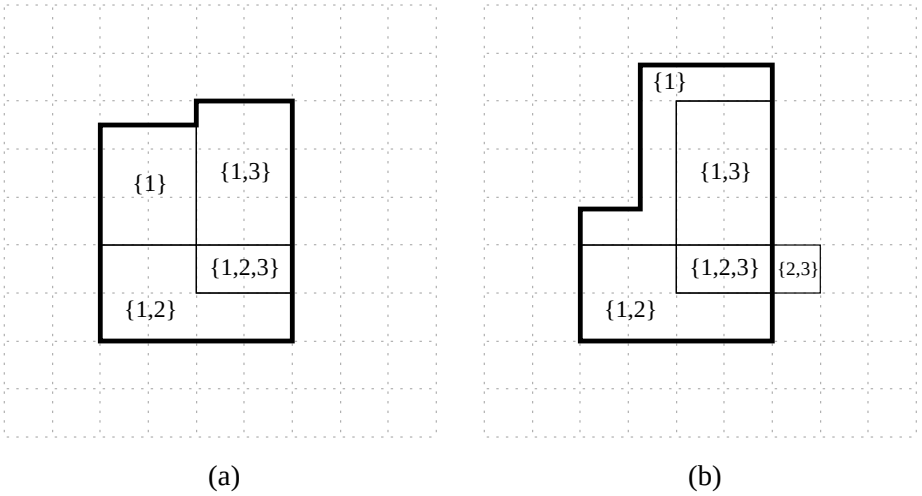


Figure 3.5: In constructing $P_{\{1\}}$, $\omega(\{1\}) = 5$ is too small to prevent c_1 from being non-rectangular; however, (a) in one construction there is a concurrent curve segment, while (b) in another construction there is no concurrent curve segment.

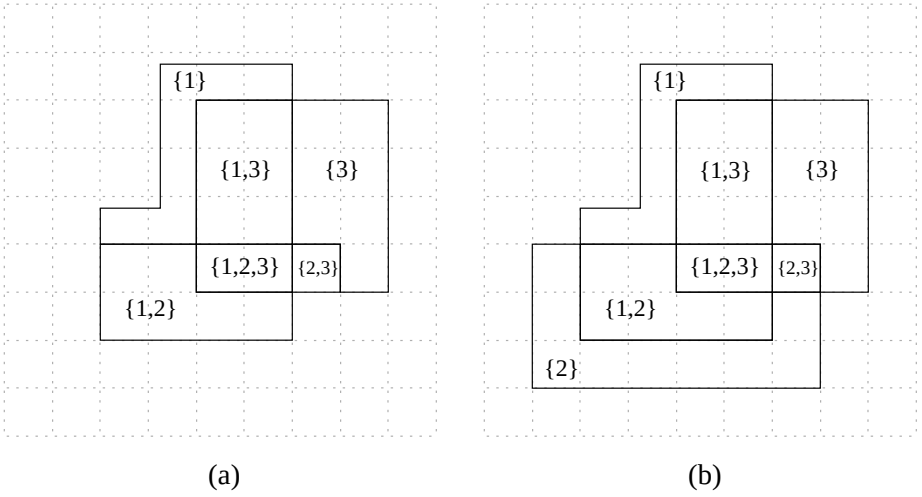


Figure 3.6: (a) In constructing $P_{\{3\}}$, $\omega(\{3\}) = 7$ is large enough to fill in the space between $P_{\{1,3\}}$ and $P_{\{2,3\}}$, (b) as is the case for $P_{\{2\}}$ where $\omega(\{2\})$ is large enough to fill in the space between $P_{\{2,3\}}$ and $P_{\{1,2\}}$.

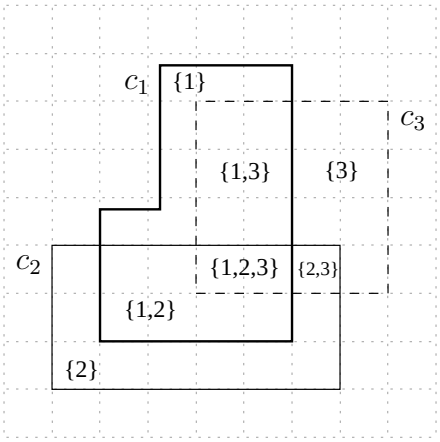


Figure 3.7: The ω-proportional simple 3-Venn diagram constructed by the three rectangle algorithm for the weight function in Tab. 3.1; since $\omega(\{1\})$ was too small, c_1 is a 6-gon.

too small, we could just uniformly extend it outwards as we did for $P_{\{1\}}$.

At this point, all the region's polygons have been constructed, and the curves can be determined by tracing the boundary of the union of the enclosed regions. As shown in Fig. 3.7, since $\omega(\{1\})$ was too small, c_1 is a 6-gon, while c_2 and c_3 are rectangles; if $\omega(\{2\})$ or $\omega(\{3\})$ were also too small, c_2 or c_3 would also be 6-gons. This three rectangle algorithm provides a constructive proof to the following theorem.

Theorem 3.2.1. *Let $S = \mathcal{PS}(X)$ where $|X| = 3$, and let ω be a weight function for S. There is an ω-proportional* simple 3-*Venn diagram that represents S and whose curves are orthogonal rectangles or orthogonal* 6*-gons.*

As we have described the three rectangle algorithm, once $P_{\{1,2,3\}}$ and $P_{\{1,2\}}$ are constructed, the diagram is determined; as a result, w and d are the algorithm's only variable parameters. It turns out that the choices for w and d can affect whether or not the curves of the resulting diagram are rectangles. Referring to Fig. 3.8, suppose we fix d and double w. The height of $P_{\{1,2,3\}}$ decreases, which increases the

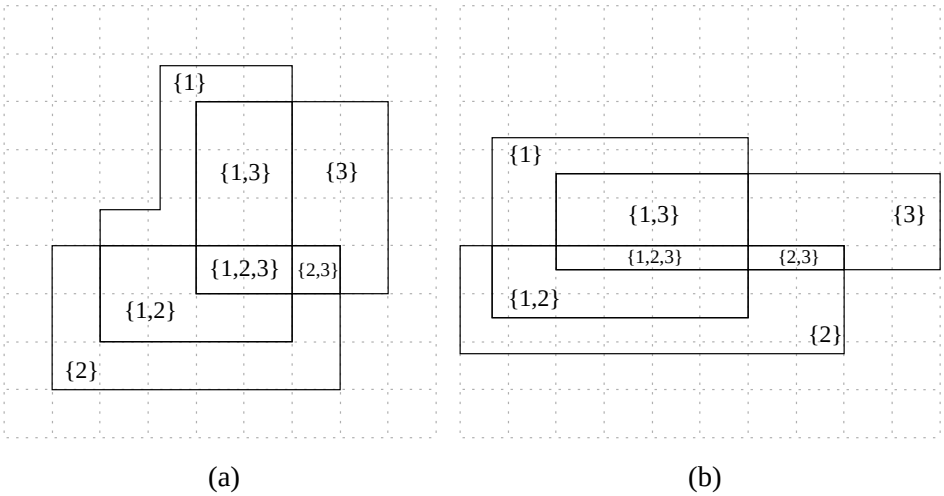


Figure 3.8: (a) For certain values of w and d, the three rectangle algorithm produces non-rectangular curves such as c_1; however, (b) by adjusting w and d (in this case only doubling w), it *may* be possible for the three rectangle algorithm to produce rectangular curves.

amount $P_{\{1,2\}}$ extends outwards from the left side of $P_{\{1,2,3\}}$; however, the height of $P_{\{1,3\}}$ decreases sufficiently to allow $P_{\{1\}}$ to fill the space and c_1 to be rectangle. The change to w does not affect the area that $P_{\{3\}}$ must fill in order for c_3 to be rectangle (i.e., 3 units), since the decrease in the height of $P_{\{1,3\}}$ is countered by a commensurate increase in the width of $P_{\{2,3\}}$. For $P_{\{2\}}$, the change to w causes an increase in the area that must be filled from 1 unit to 2 units; however, since $\omega(\{2\})$ is large enough, the increase is inconsequential. As shown in Fig. 3.8(b), by doubling w, the resulting diagram now has all rectangular curves.

Similarly, fixing w and doubling d decreases the minimum size required by $\omega(\{1\})$ in order for c_1 to be a rectangle, but increases the minimum size required by $\omega(\{2\})$ in order for c_2 to be a rectangle. From these examples, it is clear that there is a trade-off that must be balanced when determining appropriate values for w and d. In fact, since any diagram created by the three rectangle algorithm can be non-uniformly

scaled in order to make $P_{\{1,2,3\}}$ a square while preserving ω-proportionality and any rectangular curves, d is really the only parameter of consequence. It seems possible that one could solve a system of inequalities in terms of d and ω in order to yield the conditions under which the three rectangle algorithm produces only rectangles. At present, we state this as the following open problem.

Open Problem 3.2.2. Determine the necessary-and-sufficient conditions (with respect to a weight function ω), for the three rectangle algorithm to produce an ω-proportional simple 3-Venn diagram whose curves are orthogonal rectangles.

We also make the observation that all simple 3-Venn diagrams whose curves are rectangles have isomorphic layouts for regions $\{1,2,3\}$, $\{1,2\}$, $\{1,3\}$, and $\{2,3\}$; therefore, since the layout of these regions determines the minimum requirements of $\omega(\{1\}$, $\omega(\{2\}$, and $\omega(\{3\}$ for c_1, c_2, and c_3 to be rectangles, it seems that an answer to the above open problem would provide necessary-and-sufficient conditions for the existence of *any* such diagram.

Up to this point, we have considered the three rectangle algorithm within the context of Venn diagrams; what happens when we apply the algorithm to the generation of ω-proportional Euler diagrams? Consider the weight function ω shown in Tab. 3.1, but suppose $\omega(\{1,2\}) = \omega(\{1,3\}) = 0$; in other words, ω is a weight function for the set system

$$S = \{\emptyset, \{1\}, \{1,2,3\}, \{2\}, \{2,3\}, \{3\}\}.$$

As before, we construct $P_{\{1,2,3\}}$. Although $P_{\{1,2\}}$ is empty (since $\omega(\{1,2\}) = 0$), we still construct it so that it has the same height as $\omega(\{1,2,3\})$, but zero width. Similarly, we construct $P_{\{1,3\}}$ so that it has the same width as $\omega(\{1,2,3\})$, but zero

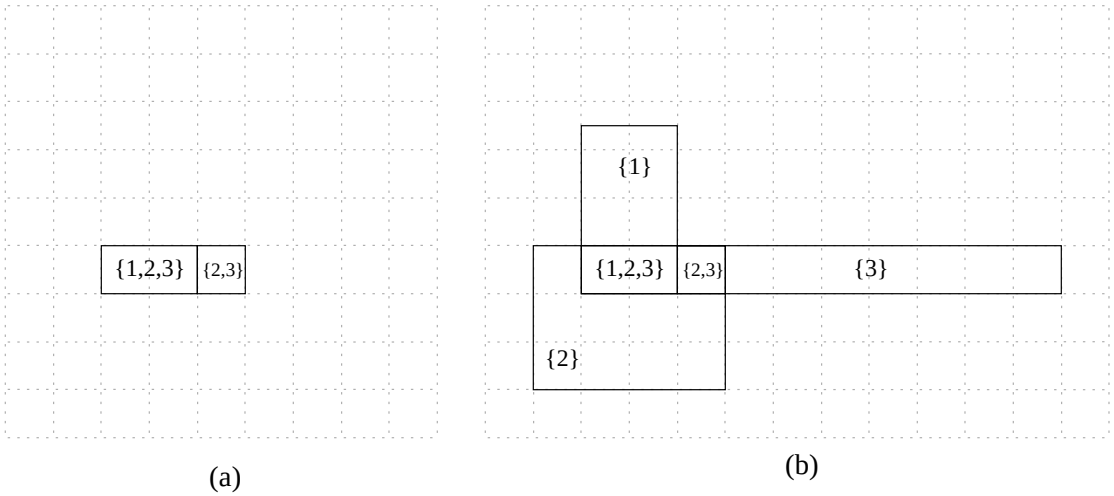


Figure 3.9: When the three rectangle algorithm is applied to an Euler diagram, as long as region $\{1, 2, 3\}$ is non-empty, all the necessary positional information is present for the diagram to be constructed.

height. Since $\omega(\{2, 3\}) > 0$, we construct $P_{\{2,3\}}$ as before. Figure 3.9(a) shows an example of the current state of the diagram.

In the previous Venn diagram example, $P_{\{1\}}$ was constructed so that its left side was aligned with the left side of $P_{\{1,2\}}$ and its right side was aligned with the right side of $P_{\{1,3\}}$. Even though $P_{\{1,2\}}$ and $P_{\{2,3\}}$ are empty, they still have positional information that is sufficient to define the x-coordinate of their left and right sides, respectively. As a result, $P_{\{1\}}$ can be constructed in the same manner as before, but there is no minimum size that $\omega(\{1\})$ must have in order for c_1 to be a rectangle. Similarly, $P_{\{3\}}$ and $P_{\{2\}}$ can be constructed as shown in Fig 3.9(b). As before, the resulting curves will be either rectangles or 6-gons; however, unlike the case for Venn diagrams, there may be concurrent curve segments (e.g., the boundary between region $\{2\}$ and region $\{1, 2, 3\}$ in Fig. 3.9(b)).

The only time that the three rectangle algorithm lacks enough information to

construct the diagram is when region $\{1, 2, 3\}$ is empty; in this case, the algorithm has no starting point. As in the Venn diagram case, the three rectangle algorithm provides a constructive proof to the following theorem.

Theorem 3.2.3. *Let S be a set system on the items X with $X \in S$, and let ω be a weight function for S. There exists an ω-proportional 3-Euler diagram that represents S and whose curves are orthogonal rectangles or orthogonal 6-gons.*

We leave the case when region $\{1, 2, 3\}$ is empty as an open problem with the observation that one would likely need to consider non-orthogonal rectangles.

3.3 Three Circle Venn Diagrams

As we saw in the previous section, orthogonal rectangles are limited in terms of which weight functions they can represent in a 3-Venn diagram. Since circles have fewer degrees of freedom than orthogonal rectangles, we expect them to be similarly limited in their ability to form an ω-proportional 3-Venn diagram; however, probably for the same reason that Euler chose to use circles in his original diagrams, they have an undeniable appeal. In fact, it is our experience that the majority of users who wish to create ω-proportional Euler diagrams attempt to do so with circles. For this reason, we consider the problem of how to construct ω-proportional simple 3-Venn diagrams whose curves are circles (note that there are no such *non-simple* diagrams since two circles have a concurrent intersection if and only if they are equivalent).

This section is based on joint work with Peter Rodgers (a UK-based Reasoning With Diagrams researcher, see Section 2.6), and is excerpted (with modifications made for consistency with Section 1.2), from a conference paper presented by Rodgers and

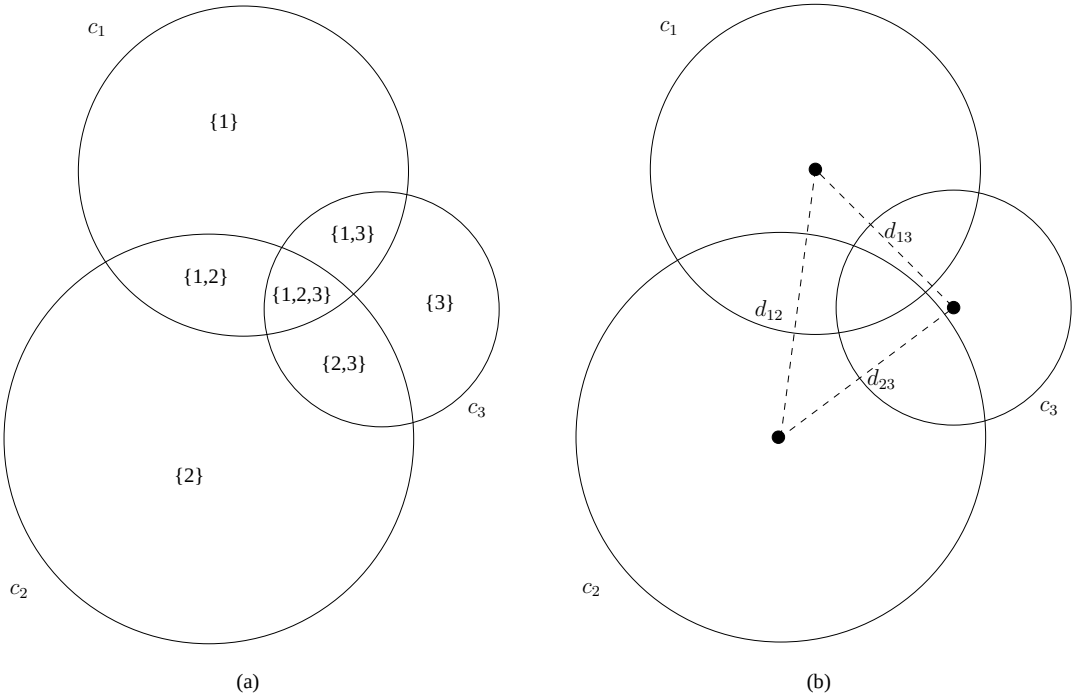


Figure 3.10: (a) An example of an ω-proportional three circle Venn diagram, and (b) the triangle formed by joining the circle's centers.

this author [7]. Rodgers designed and implemented the hill climbing optimizer while this author was responsible for the "Existence" and "Optimization" subsections.

3.3.1 Existence

Figure 3.10(a) shows an ω-proportional 3-Venn diagram $C = \{c_1, c_2, c_3\}$ whose curves are circles. Without loss of generality, let the diagram's scale be $1 : 1$ so, for example, the area of region $\{1\}$ is $\omega(\{1\})$. As a result, the area of overlap between each pair of circles c_i and c_j is given by

$$area_{ij} = \omega(\{i, j\}) + \omega(\{1, 2, 3\}).$$

If we consider each pair of circles as its own two circle Venn diagram, then from Section 3.1, the distance d_{ij} between the center of c_i and the center of c_j is uniquely determined by $area_{ij}$. As a result, the centers of the circles of an ω-proportional three circle Venn diagram form a triangle with sides d_{12}, d_{13}, and d_{23} as shown in Fig. 3.10(b). Modulo translations, rotations, and reflections, there is no freedom in positioning the circles; the area of overlap between each pair of circles will be correct, but the area of overlap between all three circles (i.e., the area of region $\{1, 2, 3\}$), is predetermined and in most cases will be incorrect. Based on this observation, given a weight function ω, the question of whether or not there exists an ω-proportional three circle Venn diagram can be answered using the following steps:

1. Compute d_{12}, d_{13}, and d_{23} using the bisection method described in Section 3.1.
2. If d_{12}, d_{13}, and d_{23} satisfy the *triangle inequality* [48], continue; otherwise, no diagram exists.
3. Compute the radii of c_1, c_2, and c_3 and place their centers at the vertices of a triangle with sides d_{12}, d_{13}, and d_{23}.
4. If the area of region $\{1, 2, 3\}$ is $\omega(\{1, 2, 3\})$ then the diagram exists; otherwise, no diagram exists.

The following theorem, which is based on the previous discussion, states that in general, the ω-EDGP is not solvable with three circles.

Theorem 3.3.1. *There exists a weight function ω for which there is* no *ω-proportional three circle Venn diagram.*

Proof. Let $C = \{c_1, c_2, c_3\}$ be an ω-proportional three circle Venn diagram. Define the weight function ω' as follows:

$$\begin{aligned}
\omega'(\{1\}) &= \omega(\{1\}) - 1 \\
\omega'(\{2\}) &= \omega(\{2\}) - 1 \\
\omega'(\{3\}) &= \omega(\{3\}) - 1 \\
\omega'(\{1,2\}) &= \omega(\{1,2\}) + 1 \\
\omega'(\{1,3\}) &= \omega(\{1,3\}) + 1 \\
\omega'(\{2,3\}) &= \omega(\{2,3\}) + 1 \\
\omega'(\{1,2,3\}) &= \omega(\{1,2,3\}) - 1
\end{aligned}$$

Suppose there is an ω'-proportional three circle Venn diagram $C' = \{c'_1, c'_2, c'_3\}$. c'_i will have the same radius as c_i since they enclose the same area. In addition, the area of overlap between a pair of circles c'_i and c'_j will be the same as for c_i and c_j since

$$\begin{aligned}
\omega'(\{i,j\}) + \omega'(\{1,2,3\}) &= \omega(\{i,j\} + 1 + \omega(\{1,2,3\} - 1 \\
&= \omega(\{i,j\}) + \omega(\{1,2,3\}).
\end{aligned}$$

As a result, C''s circles will be centered on the vertices of the same triangle as C's circles (since by the SSS rule, all such triangles are equivalent); therefore, they will have the same area for region $\{1,2,3\}$, but this is a contradiction since $\omega'(\{1,2,3\} \neq \omega(\{1,2,3\})$. So there cannot be an ω'-proportional three circle Venn diagram. □

Even though there may be a weight function ω for which there is no ω-proportional three circle Venn diagram, we might still be able to approximate a solution; in the remaining subsections, we consider such approximations.

3.3.2 Optimization

Let ω be a weight function for which there is no ω-proportional three circle Venn diagram. Suppose we wish to construct a three circle Venn diagram $C = \{c_1, c_2, c_3\}$ that is as "close" as possible to being ω-proportional. Since C is an approximation, its circles may be larger or smaller than desired and/or its regions may have areas that deviate from ω. In this subsection, we define a metric for specifying how close C is to being ω-proportional and state the approximation problem in terms of a constraint satisfaction problem (CSP). The next subsection describes a strategy for solving the CSP.

Let each circle c_i be defined by center (x_i, y_i) and radius r_i. Without loss of generality, we set $(x_1, y_1) = (0, 0)$, align c_2's center along the positive x-axis, and c_3's center in the positive y quadrants as shown in Fig. 3.11.

Circles c_1 and c_2 intersect at (d, h) and $(d, -h)$ where

$$\begin{aligned} h^2 &= {r_1}^2 - d^2 = {r_2}^2 - (x_2 - d)^2 \\ \Rightarrow d &= \frac{{r_1}^2 - {r_2}^2 + {x_2}^2}{2x_2} \\ \Rightarrow h &= \sqrt{{r_1}^2 - ({r_1}^2 - {r_2}^2 + {x_2}^2)^2/(2x_2)^2} \end{aligned}$$

In order for the three circles to form a valid Venn diagram, c_1 and c_2 must intersect non-tangentially and c_3 must bisect regions $\{1\}$, $\{2\}$, and $\{1, 2\}$. The following conditions are necessary and sufficient for the proper intersection of the three circles according to this layout:

- $x_1 = y_1 = 0$
- $x_2 > 0$ and $y_2 = 0$
- $y_3 > 0$
- $|r_1 - r_2| < x_2 < r_1 + r_2$

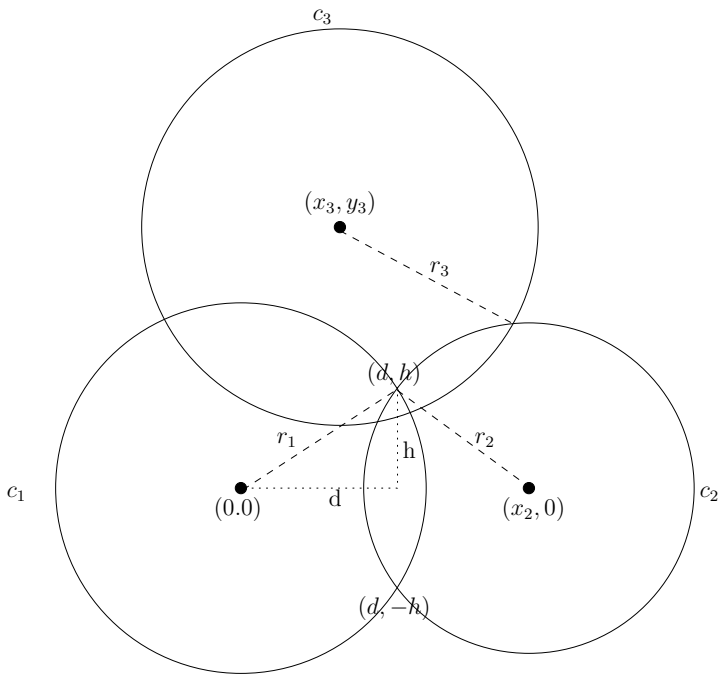


Figure 3.11: The constraint satisfaction problem layout and parameters.

- $(x_3 - d)^2 + (y_3 - h)^2 < {r_3}^2 < (x_3 - d)^2 + (y_3 + h)^2$

The total *desired* area of C is ω_{tot} where

$$\omega_{tot} = \sum_{s \in \mathcal{PS}(\{1,2,3\}) \setminus \{\emptyset\}} \omega(s).$$

Since C is an approximation, let $\alpha(s)$ be the *actual* area of region s and, correspondingly, α_{tot} the actual total area of C. In addition, let $s_1, s_2, \ldots, s_7$ be an ordering of C's regions such that $i < j \Rightarrow \omega(s_i) \leq \omega(s_j)$.

The goal of the optimization is to determine values for x_2, x_3, y_3, r_1, r_2, and r_3 that minimize the "difference" between C's actual region areas and the desired areas. For our purposes, the difference/fitness metric is a combination of both variance and "out-of-orderness" (i.e., we account for when $\omega(s_i) \leq \omega(s_j)$, but $\alpha(s_i) > \alpha(s_j)$). As a

result, the goal is to minimize the following function:

$$\Delta = \sum_{s \in \mathcal{PS}(\{1,2,3\}) \setminus \{\emptyset\}} (\omega(s)/\omega_{tot} - \alpha(s)/\alpha_{tot})^2 + \sum_{i<j} [\alpha(s_i) > \alpha(s_j)]$$

The above fitness function is idealistic and equally weights the two components; in practice, there are many interpretations of what makes a 'good' diagram. In the following subsection we describe a hill climbing approach for optimizing a more complex, but ad hoc fitness function, and present examples of its effectiveness.

3.3.3 Hill Climbing

The triangle layout described in Subsection 3.3.1 is used as the starting point for the search; this produces a layout where the areas of combinations of some regions are exact, but the areas of the single regions themselves may be far from an acceptable solution. In order to improve the layout further, we have developed a metric that attempts to quantify our perception of what is a good relationship between region areas. The optimization process changes the layout and so improves the diagram as measured by the metric.

The metric is based on our perception that it is not necessary to ensure that region areas are exactly proportional to their required size. In fact, users will be more concerned with relative sizes. In particular, when comparing the appearance of two regions, the one with the higher weight should have a larger area, and it is not particularly important how much bigger it is. Also, two regions that are more or less equal in weight should appear roughly equal in size in the diagram.

In order to measure this by a metric, we perform a pairwise comparison of all

regions. We calculate a value for each pair of regions. The values are summed to produce the metric that forms the fitness for the diagram. A lower number means a 'better' diagram. For each pair, we calculate an allowable range for the relative areas. If the areas fall in this range, the value is zero, and when they fall outside the range, the value increases with the square of the distance from the allowable range. This squaring factor penalizes region pairs falling far from the allowable range harshly, and so these pairs are more likely to be corrected than pairs with areas lying just outside the allowable range.

A pair of regions is considered to be equal if their weights are within 10% of each other. In this case, the allowable range for the area ratios is 10%. Hence regions with weight 50 and 53 would have a value of 0 if their areas were 93 and 100, but if their areas were 90 and 110 then the value would be 100.

Where the regions are not considered to be equal, the allowable range for the area ratios is between $1+0.3rp$ and $1+2rp$, where rp is how much greater the large region weight is than the small region weight; that is,

$$rp = (largeweight/smallweight) - 1.$$

For example, in the case where region weights were 10 and 20, ratios for the areas between 1.3 and 3 would result in a value of 0.

In addition, we consider the two cases where regions are not equal in area when their weights are equal and where two unequal regions have an area ratio less than the allowable range (i.e. the region with the large weight has too small an area), to be of greater importance than the case where the two unequal regions have an area

ratio greater than the allowable range (i.e. the region with the large weight has a too large an area). Hence we multiply the values for the first two cases by 100.

Clearly, the above metric is ad hoc. Both the computational method and the constants used in the calculation were developed after examining numerous example cases and comparing various alternative metrics. In particular, the 10% value for equal regions and the range of area ratios for unequal regions are not scientifically validated nor are they intended to generalize to other applications.

In order to change the diagram to improve the metric, we take a hill climbing approach. This consists of a number of iterations that move the circle centers and change the circle radii. The movement is controlled by a cooling schedule, where the amount of movement reduces on each iteration, allowing large changes at the start of the search process and refinement of the diagram layout towards the end of the process. On each iteration, each circle is modified. First, the circle is moved successively in 8 possible directions: horizontally, vertically and diagonally. Then the radius is expanded and contracted. When a move improves the diagram layout as measured by the metric, the move is kept and the next circle is tried.

Moves are restricted so that they do not modify the structure of the diagram; that is, moves that add or remove regions are not made.

3.3.4 Examples

This section illustrates the hill climbing method described in the previous subsection. Figure 3.12(a) shows a three circle Venn diagram after the initial layout. Each region is labeled by its desired weight and actual area (separated by ';'). ω-proportional diagrams would have all regions with their desired weights equal to their actual areas.

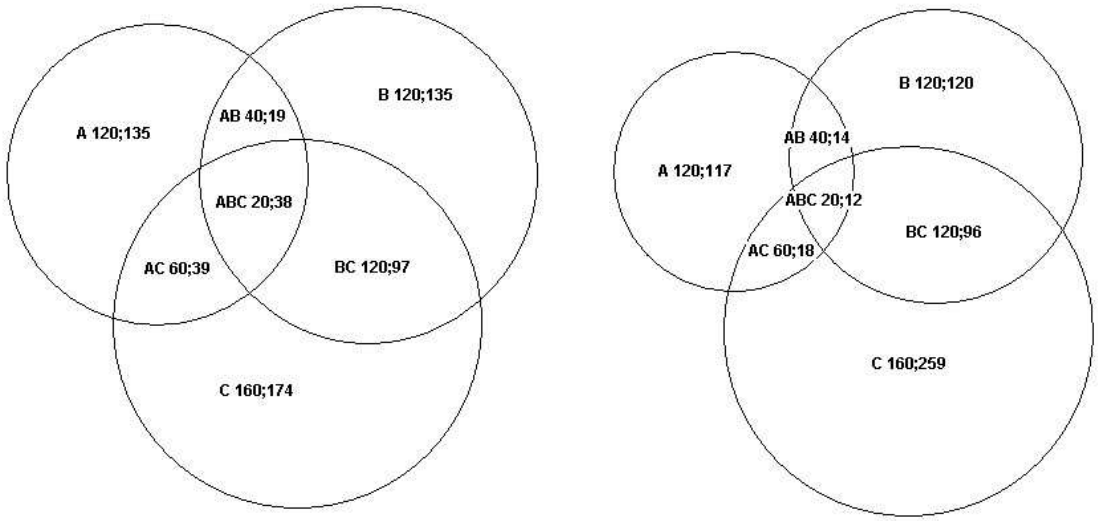


Figure 3.12: (a) An example of an initial layout (b) that is improved by the hill climber.

Figure 3.12(a) illustrates a layout where some of the region areas are bad for their desired weights. AB and AC have weights much larger than ABC, but AB has a smaller area and AC has a nearly equal area. A, B, and BC have the same weight, but whilst A and B have equal areas, BC's area is smaller. C has a larger weight and area than the other weights, so this is a reasonable layout for C.

Figure 3.12(b) shows the diagram after running the hill climber. The areas of AB and AC are now bigger than ABC, and although the relative size of their weights is much larger than would be indicated by the their areas, we consider that having area relationships in correct order to be a reasonable result. A and B now have slightly different areas, but the percentage error is small and unlikely to be noticeable. BC still has a smaller area than A and B, but the difference has been reduced significantly. C's area is still, correctly, larger than the rest, although it has increased in size. In

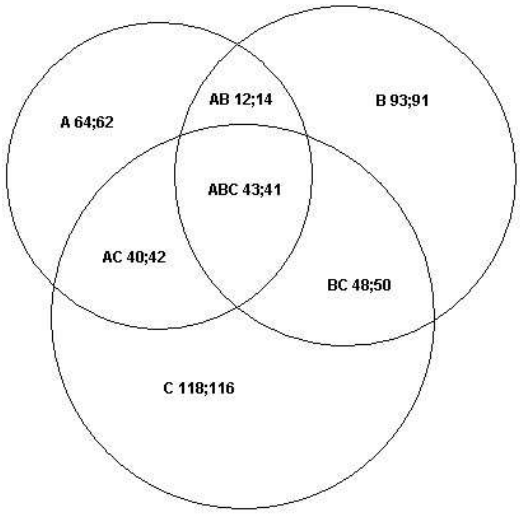


Figure 3.13: An example of a good initial layout that does not need improving.

terms of fitness values, Fig. 3.12(a) scores 275 and Fig. 3.12(b) scores 18.

Figure 3.13 shows a diagram that has a good initial layout since most of the areas are close to their weights. Such diagrams often occur is cases where the outer regions (i.e., A, B, and C) have nearly equal weights that are large compared to the inner regions' weights. The inner regions are similarly close in weight, except for AB, which is smaller; the layout method can usually adapt to single variations of this sort. The fitness value of the initial layout is approx. 0 and running the hill climber has very little impact on the diagram.

Figure 3.14(a) shows a diagram with a bad initial layout whose fitness value is 3277. When comparing region pairs, often the region with a larger weight has a smaller area. After running the hill climber, the diagram in Fig. 3.14(b) still has a bad layout whose fitness value is 2009. As with this example, when some of the 2-regions' weights are large and the fullset region's weight is small, often no good layout exists; this demonstrates a limitation of three circle Venn diagrams.

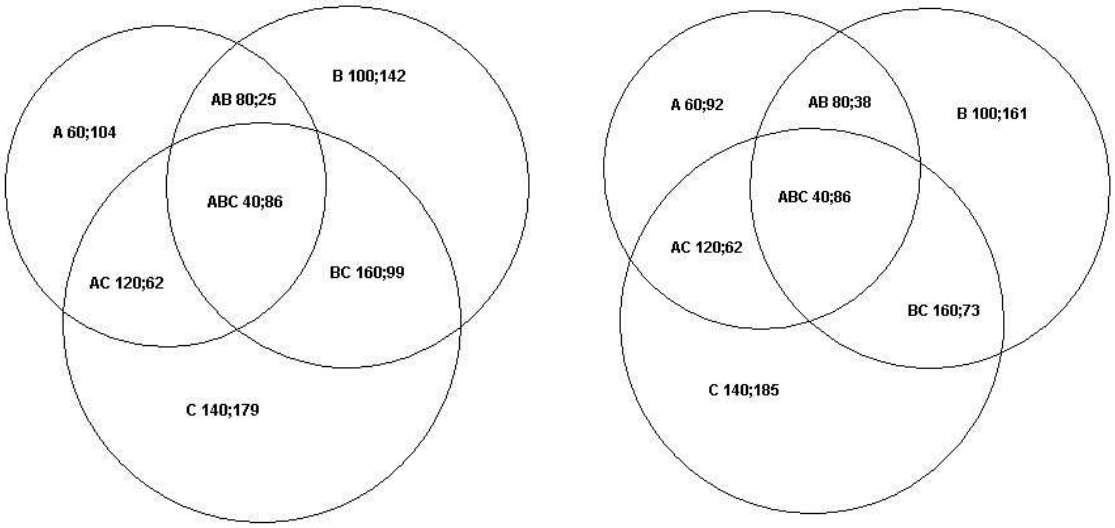


Figure 3.14: (a) An example of a bad initial layout (b) that is not improved by the hill climber.

3.4 Three Convex Curve Venn Diagrams

In the two previous sections, we considered the problem of drawing ω-proportional 3-Venn diagrams with axis-aligned rectangles and with circles, and in both cases we found that the problem has no general solution (i.e., one that works for all weight functions). A natural progression is to consider curve shapes with more degrees of freedom (e.g., ellipses and regular n-gons). In particular, many of the shapes that are commonly used in Venn diagrams are convex and can range in complexity from simple circles to convex Jordan curves. We are interested to know what effect, if any, curve convexity has on the existence of a general solution to the problem of drawing ω-proportional 3-Venn diagrams. As we shall see in this section, under the additional restriction that the ω-proportional 3-Venn diagrams be simple, curve convexity *does* preclude a general solution; however, without additional restrictions, the problem

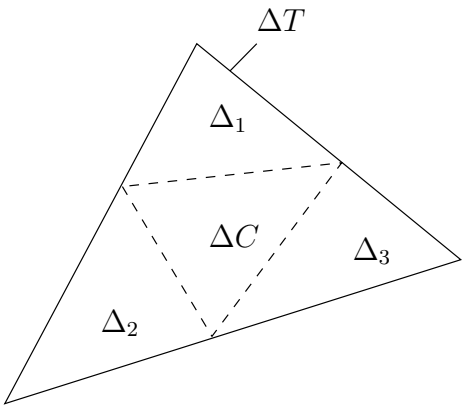


Figure 3.15: If triangle ΔC is inscribed in triangle ΔT so as to induce three triangles Δ_1, Δ_2, and Δ_3, then at least one Δ_i must have an area less than or equal to the area of Δ_T.

remains open.

Before presenting this section's main theorem, we begin by considering a lemma that plays a key role in the theorem's proof.

Lemma 3.4.1. *Let C be a convex Jordan curve. If ΔC is a triangle inscribed in C so that $int(C) \cap ext(\Delta C)$ is partitioned into three non-empty connected subsets R_1, R_2, and R_3, then at least one R_i must have an area less than or equal to the area of ΔC (e.g., see Fig. 3.16(a)).*

Proof. See Appendix A.3.4, pg. 265. □

Example. Lemma 3.4.1 is an extension of a result by Morgantini [37] where it was shown that if triangle ΔC is inscribed inside triangle ΔT so as to divide ΔT into four triangles (of which ΔC is one), then the areas of the other three triangles cannot all be greater than the area of ΔC (see Fig. 3.15). Figure 3.16 shows an example of Lem. 3.4.1 in action. In Fig. 3.16(a), the area of R_2 is less than the area of ΔC. As shown

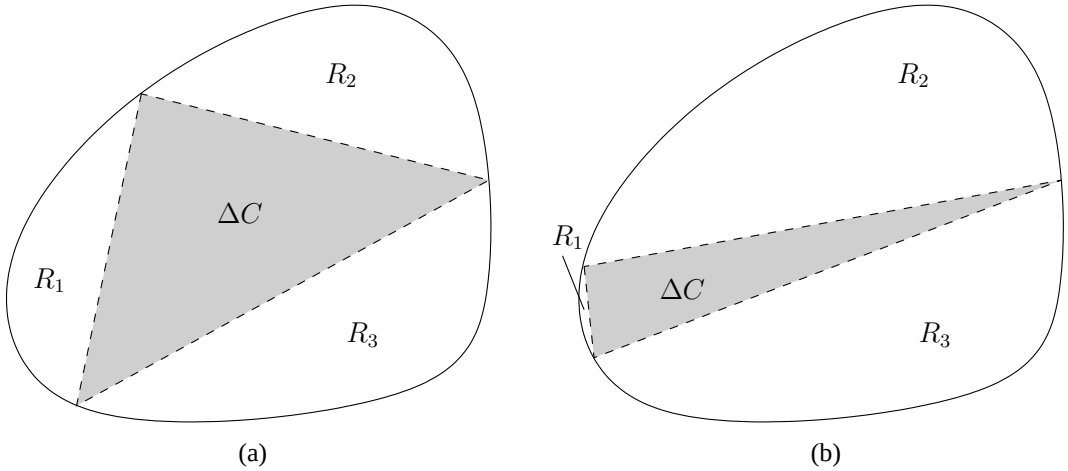


Figure 3.16: Morgantini's result [37] can be extended to a triangle ΔC inscribed in a convex curve. (a) At least one of the connected plane subsets R_i must have an area less than or equal to the area of ΔC because (b) any attempt to increase one region's area necessarily decreases another.

in Fig. 3.16(b), in attempting to increase the area of R_2, the area of R_1 is necessarily decreased. On the other hand, if we imagine fixing ΔC and then transforming C so that each R_i has an area greater than ΔC, before this inequality is achieved, C is forced to have an inflection point and thus become non-convex. □

We wish to prove that there is no general solution to the problem of drawing ω-proportional *simple* 3-Venn diagrams with convex curves; in other words, we would like to find a weight function ω for which there is no such ω-proportional diagram. How can we apply Lem. 3.4.1 to help with the proof?

Chilakammari et al. [34] determined that all simple 3-Venn diagrams are topologically equivalent to the three circle diagram shown in Fig. 3.17(a) (i.e., their regions and point intersections have the same relative positions). Suppose we assume that *every* weight function has an ω-proportional simple 3-Venn diagram whose curves are convex. Based on this assumption, we can prove that there is some weight function

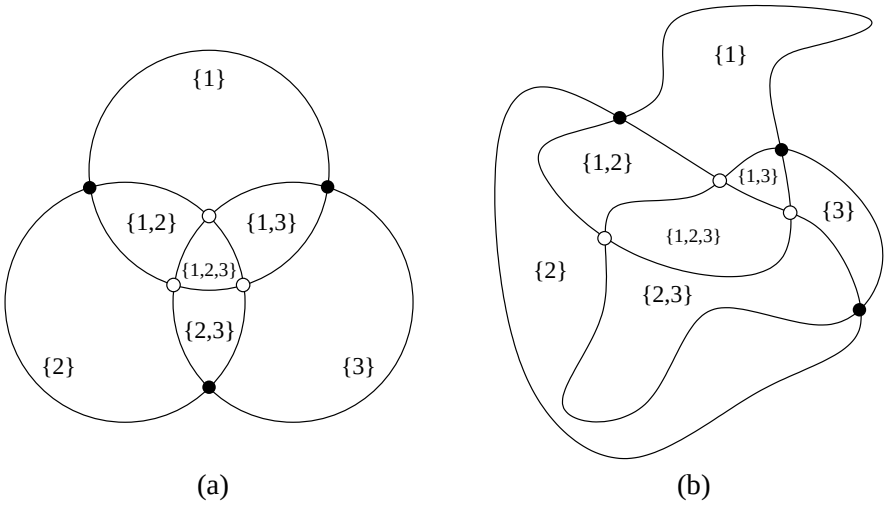


Figure 3.17: All simple 3-Venn diagrams are topologically equivalent to the classic three circle diagram in the sense that their regions and point intersections have the same relative positions.

ω' for which $\omega'(\{1,2,3\}$ is less than each of $\omega'(\{1,2\})$, $\omega'(\{1,3\})$, and $\omega'(\{2,3\})$, and which *must* have an ω'-proportional diagram whose union of regions $\{1,2,3\}$, $\{1,2\}$, $\{1,3\}$, and $\{2,3\}$ is a convex plane subset as shown in Fig. 3.18. Suppose we inscribe a triangle ΔC within region $\{1,2,3\}$ as shown by the dashed lines in Fig. 3.18. Since region $\{1,2,3\}$ is convex, the area of ΔC is less than or equal to $\omega'(\{1,2,3\})$ which, in turn, is less than each of $\omega'(\{1,2\})$, $\omega'(\{1,3\})$, and $\omega'(\{2,3\})$; however, this arrangement contradicts Lem. 3.4.1. As a result, there must be some weight function for which there is no ω-proportional simple 3-Venn diagram whose curves are convex. The most subtle part of the proof is to show the existence of ω'. The following theorem states this result, and the formal proof is included in the appendix.

Theorem 3.4.2. *There exists a weight function* ω *for which there is* no ω*-proportional simple* 3*-Venn diagram whose curves are convex.*

Proof. See Appendix A.3.4, pg. 270. □

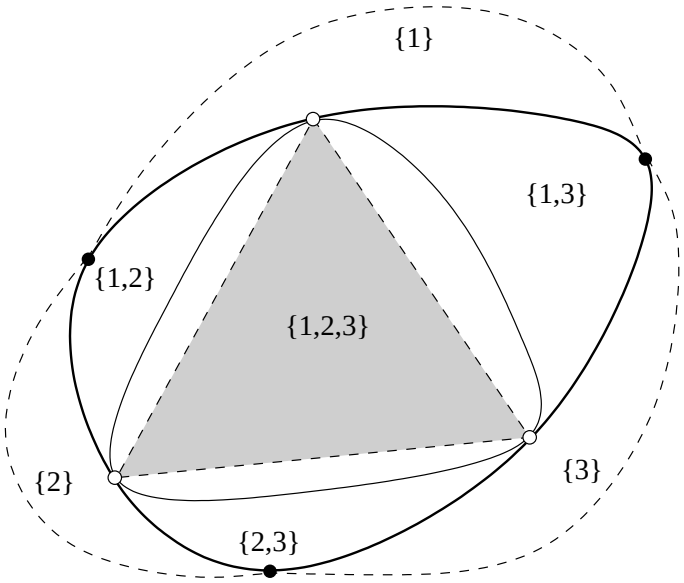


Figure 3.18: An ω-proportional simple 3-Venn diagram whose curves are convex and for which the boundary (thick black) of regions $\{1,2,3\}$, $\{1,2\}$, $\{1,3\}$, and $\{2,3\}$) is convex.

Theorem 3.4.2 tells us that in the presence of simple 3-Venn diagrams, convexity is a restrictive enough property to affect ω-proportionality. Extending the theorem to include *all* 3-Venn diagrams poses some challenges because the proof of Thm. 3.4.2 relies on the fact all simple 3-Venn diagrams are topologically equivalent, but this is not the case for non-simple 3-Venn diagrams where there are 13 topological equivalence classes [41]. As a result, we leave this as an open problem.

Open Problem 3.4.3. Is there a weight function ω for which there does *not* exist an ω-proportional 3-Venn diagram whose curves are convex. □

It is also interesting to consider the implications of Thm. 3.4.2 with respect to higher order Venn diagrams. Does Thm. 3.4.2 imply that there is no general solution to the problem of drawing ω-proportional simple n-Venn diagrams with convex curves when $n > 3$? The answer to this question is unclear. Certainly, if *every* $n > 3$ simple

n-Venn diagram were actually a simple $(n-1)$-Venn diagram with an additional curve, the answer would be "Yes" since any general solution for $n > 3$ would imply a general solution for $n = 3$. Grünbaum [27] observed (and later Chilakamarri et al. [34] proved), that there are only two topologically distinct simple 4-Venn diagrams, and each is derived by adding a curve to the simple 3-Venn diagram; therefore, Thm. 3.4.2 can be extended to $n = 4$. It turns out that, at least for $n = 5$, there is a simple 5-Venn diagram composed of ellipses for which the removal of *any* ellipse does *not* result in a simple 4-Venn diagram [41]. Additionally, the mere existence of a simple n-Venn diagram that is not reducible to an $(n-1)$-Venn diagram does not preclude the theorem from being true because, as is the case for $n = 5$, there are reducible 5-Venn diagrams. We presently leave further extension of Thm. 3.4.2 as an open problem.

Open Problem 3.4.4. Does the result of Thm. 3.4.2 apply to ω-proportional simple n-Venn diagrams with convex curves when $n > 4$? □

A final interesting problem is to consider whether the reduction to Lem. 3.4.1 used in the proof of Thm. 3.4.2 can be reversed to provide a construction for ω-proportional simple 3-Venn diagrams whose curves are convex when at least one of $\omega(\{1,2\})$, $\omega(\{1,3\})$, or $\omega(\{2,3\})$ is less than or equal to $\omega(\{1,2,3\})$. For example, as shown in Fig. 3.19, suppose we were able to inscribe a triangle with area $\omega(\{1,2,3\})$ within another triangle so that the three induced triangles had area $\omega(\{1,2\})$, $\omega(\{1,3\})$, and $\omega(\{2,3\})$; such an arrangement does not violate Morgantini's result [37]. The remaining regions could then be drawn as "leaf" triangles whose area can be as large or small as necessary. As shown by the thick black line in Fig. 3.19, the resulting curves are 5-gons. This idea inspires the following open problem.

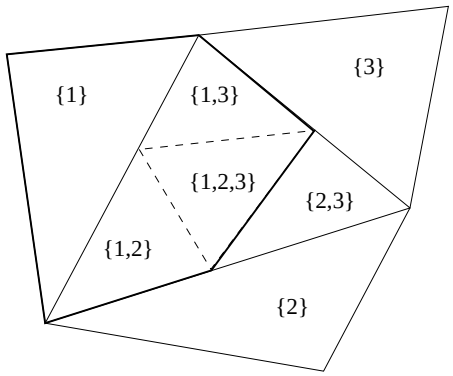


Figure 3.19: An example of a construction of an ω-proportional simple 3-Venn diagram whose curves are convex 5-gons. The construction begins by inscribing the dashed triangle with area $\omega(\{1,2,3\}$ inside a larger triangle so that the three induced triangles have area $\omega(\{1,2\})$, $\omega(\{1,3\})$, and $\omega(\{2,3\})$. The remaining regions are drawn with "leaf" triangles.

Open Problem 3.4.5. If ω is a weight function where at least one of $\omega(\{1,2\})$, $\omega(\{1,3\})$, or $\omega(\{2,3\})$ is less than or equal to $\omega(\{1,2,3\})$, does there *always* exist an ω-proportional simple 3-Venn diagram whose curves are convex (or, specifically, convex 5-gons)? □

In this chapter, we considered the problem of drawing ω-proportional Euler and Venn diagrams that were limited to 2 and 3 curves of specific shape. Although it is fairly easy to generate diagrams for these cases without regard for region areas, as we've seen, constructing an ω-proportional drawing is not so easy. It is obvious that we cannot draw an ω-proportional Euler diagram without being able to generate the topological structure of its underlying Euler diagram, so for the next few chapters, we will put aside ω-proportional Euler diagrams and focus on the Euler Diagram Generation Problem (EDGP). As we will see, unlike the small cases that we have looked at, the problem of generating Euler diagrams for greater numbers of sets is not trivial.

Chapter 4

Graph-Theoretic Foundations

Up to now, we have considered Euler diagrams as a composition of Jordan curves, which we have directly manipulated to create diagrams with certain desired properties. As we shall see in this chapter, Euler diagrams have an inherent plane graph structure and, more importantly, a plane dual structure, which gives rise to a graph-theoretical approach for describing and manipulating the diagrams. In fact, because of their discrete nature, the underlying graph structures tend to be easier to manipulate and reason with than their continuous Jordan curve counterparts in the plane-geometric realm. Before delving into the relationship between graph theory and Euler diagrams, we wish to introduce some basic graph theory notation and terminology that will be used throughout the remainder of this dissertation. By no means are the definitions in the following section intended to be exhaustive, and the reader is assumed to already be familiar with basic graph theory; their purpose is to specify notation and clarify the meaning of terms that are either subtle or not consistently defined in general use.

4.1 Graph Notation and Terminology

We use the term *graph* to refer to the most general type of combinatorial graph, one which may have multiple edges between the same pair of vertices as well as loops. The following definition describes the notation and terminology applicable to graphs.

Definition 4.1.1. A *graph* G is composed of vertices, denoted $V(G)$, and edges, denoted $E(G)$. We may write $G = (V, E)$ in which case $V = V(G)$ and $E = E(G)$.

If an edge $e \in E(G)$ connects vertices $u, v \in V(G)$ then u and v are *adjacent*, e is *incident* to u and v, and u and v are e's *endpoints*. If $u = v$ then e is a *loop*. If u and v are connected by two or more edges then these edges are referred to as *parallel* edges.

If G is *undirected*, we may write either $e = (u, v)$ or $e = (v, u)$. If G is *directed*, we say that G is a *digraph* and write $e = (u, v)$ if the edge is directed from u to v and $e = (v, u)$ if the edge is directed from v to u; in either case, e is directed from the *head* vertex to the *tail* vertex.

If G is *edge-labeled* then each edge $e \in E(G)$ has a label $l(e)$ whose contents will be defined by G's context.

The *degree* of a vertex v, denoted $deg(v)$, is the number of times v appears as the endpoint of an edge. By this definition, a loop contributes twice to its endpoint's degree. □

Since we are usually interested in the vertices and edges of a graph G, when the context is clear, we may write $v \in G$ instead of $v \in V(G)$ and $e \in G$ instead of $e \in E(G)$ to refer to an edge or vertex in G, respectively. In addition, we often use a graph G to represent the relationships between the elements of some set S; in such

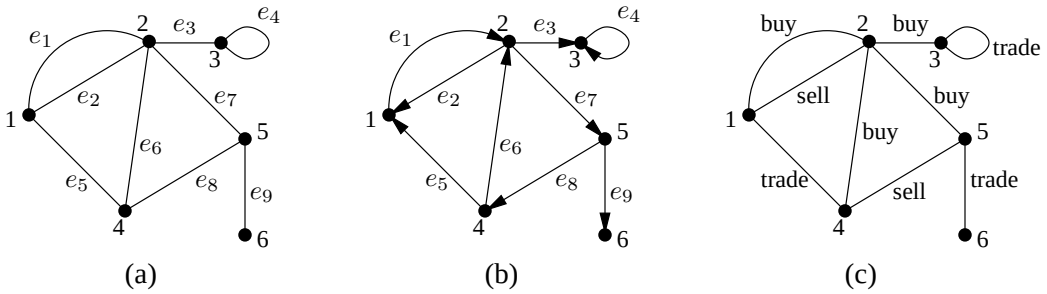


Figure 4.1: (a) An undirected graph G on $S = \{1, 2, 3, 4, 5, 6\}$, (b) a corresponding digraph $\vec{G}$ on $S = \{1, 2, 3, 4, 5, 6\}$, and (c) an edge-labeled version of G.

cases, there is an implied bijection between $V(G)$ and S. To emphasize this bijection, we use the phrase, "G is a graph *on* S," and when the context is clear, we may refer to a vertex of G by its corresponding element in S, or vice versa, as demonstrated in the following example.

Example. Figure 4.1(a) shows an undirected graph G on $S = \{1, 2, 3, 4, 5, 6\}$. G has six vertices and nine edges. Vertices 1 and 2 are the endpoints of an edge $e_1 = (1, 2) = (2, 1)$; therefore, 1 and 2 are adjacent and e_1 is incident to 1 and 2. In addition, there is a parallel edge $e_2 = (1, 2) = (2, 1)$, which also connects 1 and 2. Lastly, edge $e_4 = (3, 3)$ is a loop.

Figure 4.1(b) shows a digraph $\vec{G}$ created by orienting the edges of G. Unlike in G where we can write $e_1/e_2 = (1, 2)$ or $e_1/e_2 = (2, 1)$, in $\vec{G}$ we specifically write $e_1 = (1, 2)$ and $e_2 = (2, 1)$. In G and $\vec{G}$, the vertex degrees are $deg(1) = 3$, $deg(2) = 5$, $deg(3) = 3$, $deg(4) = 3$, $deg(5) = 3$, and $deg(6) = 1$.

Finally, Fig. 4.1(c) shows an edge-labeled graph corresponding to G. Note that the edge labels need not be unique. For example, $l(e_1) = l(e_6) =$ buy. □

In most cases, the graphs that we are interested in have no parallel edges or loops; the following definition assigns a specific name to such graphs.

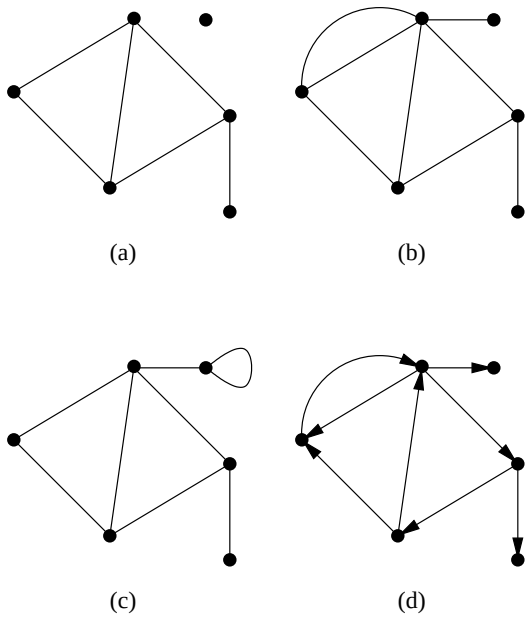


Figure 4.2: (a) A disconnected simple graph, (b) a connected non-simple graph that is loop-free, (c) a connected non-simple graph that is *not* loop-free, and (d) a weakly connected non-simple digraph that is loop-free.

Definition 4.1.2. A graph G is *simple* if it has no parallel edges or loops; otherwise, it is *non-simple*. If G has no loops then it is *loop-free*. □

Connectivity is also a fundamental property of graphs; the following definition formalizes what it means for both undirected and directed graphs to be connected.

Definition 4.1.3. An undirected graph is *connected* if there is a path between every pair of vertices.

A directed graph is *weakly connected* if its underlying undirected graph is connected and *strongly connected* if there is a directed path between every pair of vertices. □

Example. Figure 4.2(a) shows an undirected graph that is simple since it has neither parallel edges nor loops, but is disconnected due to the isolated vertex. Figure 4.2(b)

shows an undirected graph that is connected, but non-simple due to a pair of parallel edges; however, although the graph is non-simple, it is loop-free. Figure 4.2(c) shows an undirected graph that is connected, but although it has no parallel edges, it is non-simple due to the loop. Finally, Fig. 4.2(d) shows a digraph that is weakly connected since its underlying undirected graph is the same as Fig. 4.2(b), but is *not* strongly connected due to the pendant vertices; note as well that this digraph is non-simple due to the pair of parallel edges, which are still parallel even though they have opposing orientations. □

Every graph defines a set of subgraphs, but we need to be careful with how we define subgraphs of digraphs and graphs with parallel edges. The following definition specifies the properties that constitute a subgraph.

Definition 4.1.4. Let G and G' be two graphs both of which are either undirected or directed. G' is a *subgraph* of G, denoted $G' \subseteq G$, if G' can be derived by removing vertices and/or edges from G. We may also use the notation $G' \subset G$ to emphasize the case when G' is not isomorphic to G.

G' is a *pseudo-subgraph* of G if $V(G') \subseteq V(G)$ and $(u, v) \in E(G')$ implies $(u, v) \in E(G)$.

G' is a *spanning* (pseudo-)subgraph of G if $V(G') = V(G)$. □

Example. Figure 4.3(a) shows an undirected graph G with a spanning subgraph $G' \subset G$ in Fig. 4.3(b). The graph G' in Fig. 4.3(c) is a non-spanning pseudo-subgraph of G because for every pair of adjacent vertices $u, v \in G'$ is adjacent in G; note that this definition of pseudo-subgraphs in terms of adjacent vertices is specific to the case when G is undirected.

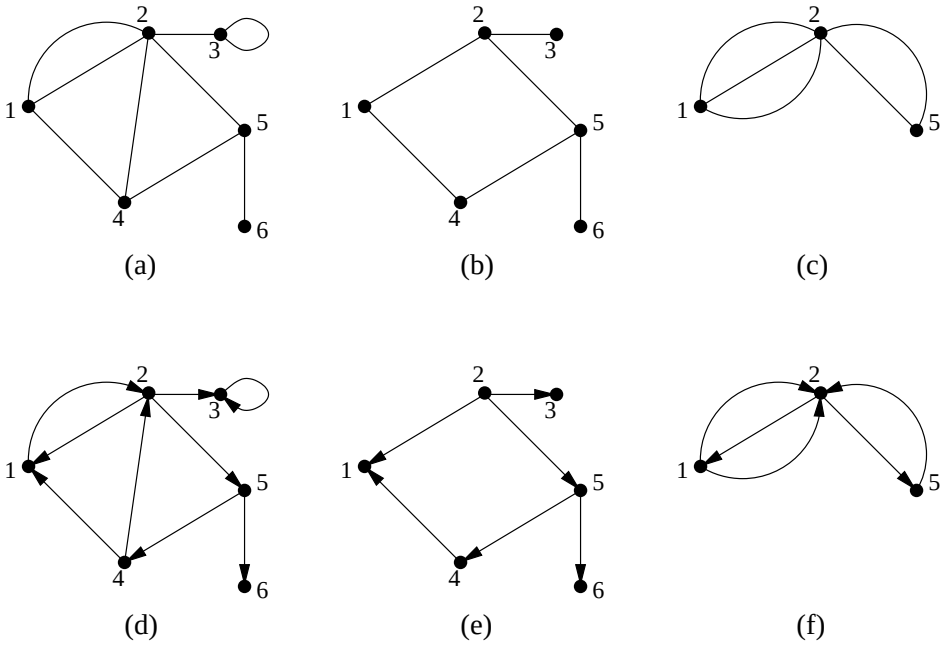


Figure 4.3: (a) A undirected graph for which (b) is a spanning subgraph and (c) is a non-spanning pseudo-subgraph, and (c) a digraph for which (d) is *not* a subgraph (since edge $(2, 4)$ is not in (c)), and (d) is *not* a pseudo-subgraph (since edge $(5, 2)$ is not in (c)).

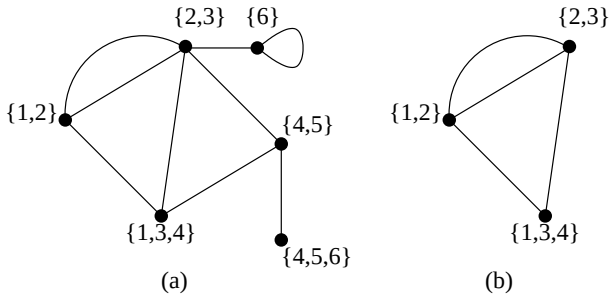


Figure 4.4: (a) A graph G on $S = \{\{1,2\},\{1,3,4\},\{2,3\},\{4,5\},\{4,5,6\},\{6\}\}$, and (b) the subgraph $G\backslash V'$ where $V' = \{x \in V(G) | x \cap \{5,6\} \neq \emptyset\}$.

Figure 4.3(d) shows a digraph $\vec{G}$. The digraph in Fig. 4.3(e) is *not* a subgraph of $\vec{G}$ since it cannot be derived by removing vertices and/or edges from $\vec{G}$; in particular, the edge $(2,4)$ is oriented the wrong way. Similarly, the digraph in Fig. 4.3(f) is *not* a pseudo-subgraph of $\vec{G}$; the parallel edges between vertices 1 and 2 are fine, but the parallel edge $(5,2)$ is not in $\vec{G}$. □

A common way to define a subgraph of G is to specify the vertices and/or edges that are removed from G. The following definition formalizes this subgraph notation.

Definition 4.1.5. Let $G = (V, E)$ be a graph and let $V' \subseteq V$ and $E' \subseteq E$ be subsets of its vertices and edges, respectively.

The subgraph derived from G by removing each vertex in V' and its incident edges is denoted $G\backslash V'$ and the subgraph derived from G by removing each edge in E' is denoted $G\backslash E'$. □

Example. Figure 4.4(a) shows an undirected graph G whose vertices are sets of integers. If we let $V' = \{x \in V(G) | x \cap \{5,6\} \neq \emptyset\}$ (i.e., V' is the set of vertices containing 5 or 6), then Fig. 4.4(b) shows the subgraph $G\backslash V'$. If we let $E' = \{(\{1,3,4\},\{4,5\}),(\{2,3\},\{4,5\}),(\{2,3\},\{6\}),(\{4,5\},\{4,5,6\}),(\{6\},\{6\})\}$ then the

subgraph $G\backslash E'$ would look similar to $G\backslash V'$ except that it would contain the three additional isolated vertices. □

Although in the previous examples we have represented graphs via drawings, a graph has no implicit plane representation; instead, it is an abstract representation of the relationships between its vertices. That being said, there are many times when we wish to manipulate graphs or reason about them on the basis of specific drawings; the following definition introduces the terminology for this application.

Definition 4.1.6. A *plane embedding* of a graph G is a drawing of G in the plane where each vertex $v \in V(G)$ has a unique location, denoted $pt(v)$, and each edge $e = (u, v) \in E(G)$ is represented by a simple curve, denoted $c(e)$, that extends from $pt(u)$ to $pt(v)$ and does not intersect any other curve except at its endpoints.

The term *plane graph* refers to a graph and an associated plane embedding.

A plane embedding of G defines a closed subset $pts(G) \subset \mathbb{R}^2$ where

$$pts(G) = \bigcup_{v \in V(G)} pt(v) \cup \bigcup_{e \in E(G)} c(e).$$

$pts(G)$ divides the plane into a set of maximal connected open subsets called the *faces* of G and denoted $F(G)$. Exactly one face is unbounded and referred to as the *outer face* or f_∞

The vertices and edges of G whose representative plane embedding points and curves are in the boundary δf_∞ are said to be *outer*.

If the boundary δf of a face $f \in F(G)$ is connected then when the context is clear, we may use δf to refer to a *traversal* of the edges comprising δf; the orientation of this traversal will be explicitly noted when required. The *degree* of a face f, denoted

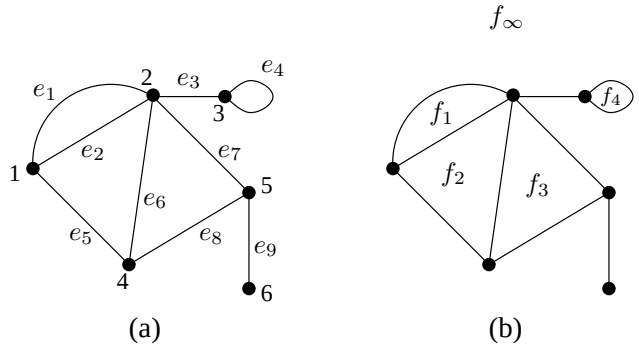


Figure 4.5: A plane embedding G with vertices and edges labeled separately from the faces for clarity.

$deg(f)$, is the length (in terms of edges) of the traversal δf. If the traversal δf is undefined then so is $deg(f)$.

Example. Figure 4.5(a) shows a plane graph G with its vertices and edges labeled; to reduce clutter, the G's faces are labeled separately in Fig. 4.5(b). Note how the outer face f_∞ is the unbounded plane subset exterior to all of G's bounded faces. In addition, all the vertices are outer and edges e_2 and e_6 are the only edges that are not outer.

Although in the terms of $\mathbb{R}^2$, δf_3 is a triangle, we may also refer to δf_3 in terms of a clockwise traversal of f_3's boundary edges, in which case δf_3 is any cyclic permutation of e_6, e_7, e_8. Similarly, as counter-clockwise traversal, δf_3 is any cyclic permutation of e_6, e_8, e_7. In either case, $deg(f_3) = 3$.

Finally, as an example of how a traversal may have repeated edges, the clockwise traversal δf_∞ is any cyclic permutation of e_1, e_3, e_4, e_3, e_7, e_9, e_9, e_8, e_5 and $deg(f_\infty) = 9$. □

There are also times when we are simply interested in knowing whether or not there is a plane embedding of a graph G without regard for what that particular

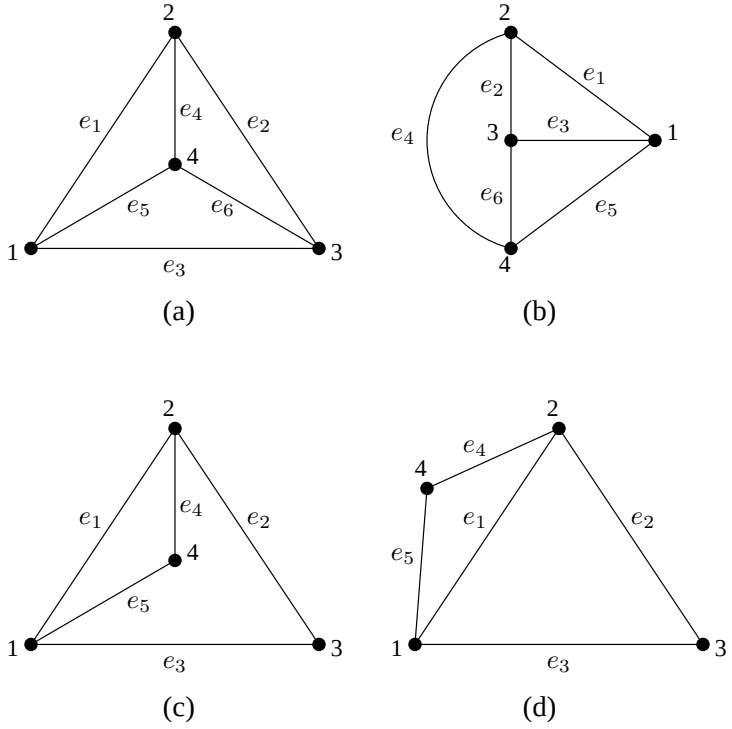


Figure 4.6: (a)(b) Two plane graphs that have the same combinatorial embedding, and (c)(d) two plane graph that have different combinatorial embeddings.

embedding is; the following definition specifies this property of G.

Definition 4.1.7. A graph is *planar* if it permits a plane embedding.

Consider the plane graph G in Fig. 4.6(a) and the plane graph G' in Fig. 4.6(b). Although G and G' are two different plane graphs, they are both plane embeddings of the same graph. It turns out that G and G' have a very similar structure: each face has degree 3 and the cyclic order of edges around each vertex is the same. For example, vertex 1 has its incident edges clockwise-ordered e_1, e_5, e_3 in both G and G'.

However, not all the plane embeddings of a graph share the same structure. For example, Fig. 4.6(c) and Fig. 4.6 show two plane graphs G and G' that are plane

embeddings of the same graph. In this case, vertex 1 has its incident edges clockwise-ordered e_1, e_5, e_3 in G and clockwise-ordered e_5, e_1, e_3 in G'. The following definition captures the notion of what defines the essential structure of a plane graph.

Definition 4.1.8. A *combinatorial embedding* of a graph G is a cyclic ordering of the incident edges (and thus adjacent vertices), about each of G's vertices and is naturally derived from a plane embedding of G. The ordering may be clockwise or counter-clockwise, as long as it is consistent over all vertices.

The term *combinatorial graph* refers to a graph and its associated combinatorial embedding. □

A combinatorial embedding encodes the faces of the plane embedding from which it was derived; a traversal of these faces' boundary edges can be recovered by traversing the combinatorial embedding. Although a combinatorial embedding encodes faces, it does not designate the outer face. As a result, two plane graphs with the same combinatorial embedding are topologically equivalent modulo the choice of outer face.

Table 4.1 shows the hierarchy of graph types according to increasing degree of specificity. For example, the greater specificity of plane graphs over combinatorial graphs means that a given plane graph contains all the information needed to construct its associated combinatorial graph.

Where the particulars can be ignored, we will refer to any one of the previous data structures simply as *graph structures*. In the case of subgraphs defined by removing vertices and/or edges, all other information is preserved including, for example, the drawing of the remaining vertices and edges in a plane graph. Lastly, we consider the concept of plane duality, which plays a prominent role in Euler diagram generation.

Definition 4.1.9 (Adapted from Diestel pg. 103 [9])**.** Let $G = (V, E)$ and (V^*, E^*)

Specificity (increasing)	Data Structure	Description
	graph	abstract vertices and edges
↓	planar graph	abstract vertices and edges that permit a plane embedding
	combinatorial graph	abstract vertices and cyclically-ordered edges derived from a plane embedding
	plane graph	concrete vertices and edges drawn in the plane

Table 4.1: Hierarchy of graph types.

be any two plane graphs. We call (V^*, E^*) a *plane dual* of G, denoted $G^* = (V^*, E^*)$, if there are bijections

$$F(G) \to V^* \quad E \to E^* \quad V \to F(G^*)$$

$$f \to v^*(f) \qquad e \to e^* \qquad v \to f^*(v)$$

satisfying the following conditions:

1. $pt(v^*(f)) \in f$ for all $f \in F(G)$,
2. $|c(e^*) \cap pts(G)| = |c(e^*) \cap c(e)| = |c(e) \cap pts(G^*)| = 1$ for all $e \in E$, and in each of e and e^* this point is an inner point of a straight line segment, and
3. $v \in f^*(v)$ for all $v \in V$.

The following proposition gives the conditions under which a plane graph has a plane dual.

Proposition 4.1.1 (Diestel pg. 104 [9])**.** *A plane graph G has a plane dual G^* if and only if G is connected.*

The proof to Prop. 4.1.1 is by the following construction, which we will call the *plane dualization* operation:

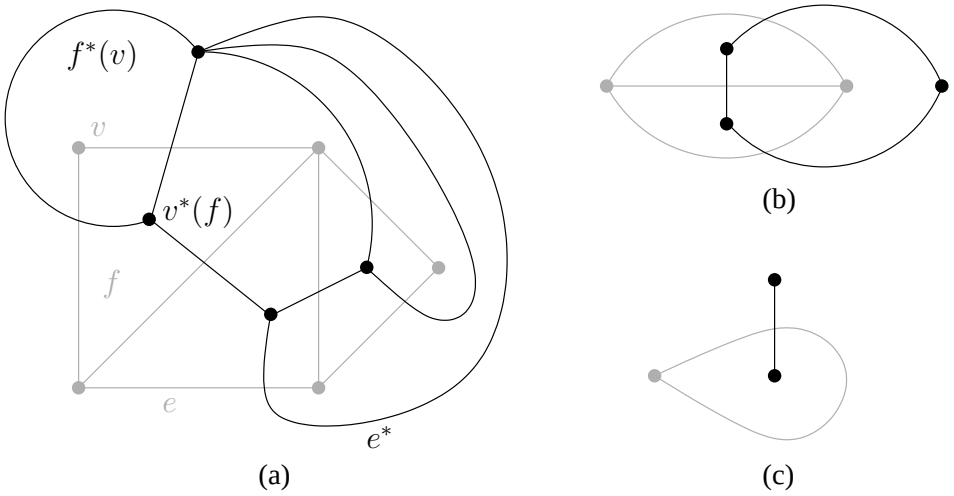


Figure 4.7: Examples of plane graphs (light gray) overlayed with their plane duals (black).

Given a plane graph G, a plane dual G^* can be created by placing a vertex $v^*(f)$ in each face $f \in F(G)$ (Def. 4.1.9(1)). An edge $e \in G$ is in the boundary of at most two faces, $f_l, f_r \in F(G)$; an edge e^* is drawn between $v^*(f_l)$ and $v^*(f_r)$ so that it crosses e at an interior point (i.e., non-endpoint) on both edges and does not cross any other edge in G or G^* (Def. 4.1.9(2)).

Also, as noted in Diestel pg. 104 [9], all plane duals of a plane graph G are topologically equivalent, so in most cases we can refer to *the* plane dual G^*.

Example. Figure 4.7 shows several examples of plane graphs (light gray) overlayed with their plane duals (black) and demonstrates how plane duality is a true dual operation in that G^{**} is isomorphic to G. Also note how the graphs in Figs. 4.7(a)(b) show that a plane graph without parallel edges may have a plane dual with parallel edges, and vice versa; Figure 4.7(c) shows a similar variation with respect to loops. □

The fact that plane dualization is a dual operation implies that the plane dual has the same connectivity as the plane graph from which it is derived; the following

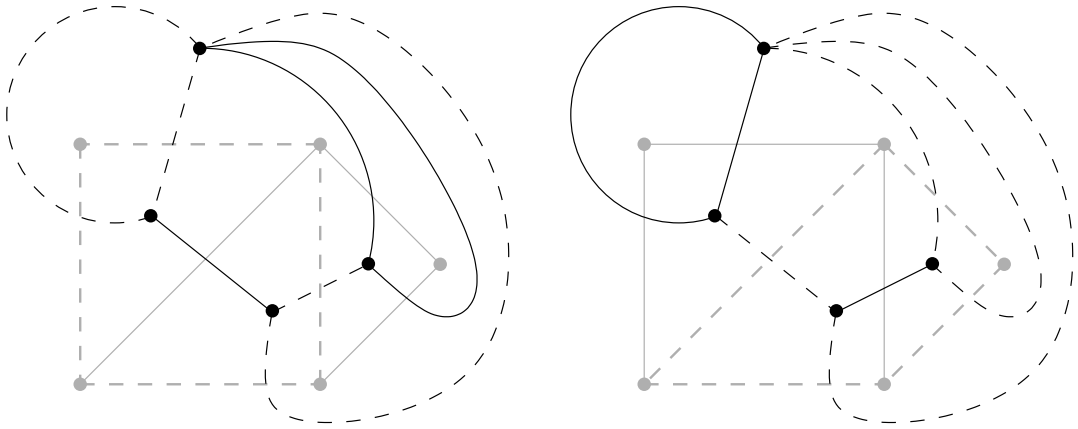


Figure 4.8: (a)(b) A plane graph G (light gray) overlayed with its plane dual G^* (black) showing how a cycle in G (dashed light gray) corresponds to a minimal edge cut in G^* (dashed black).

proposition captures this idea.

Proposition 4.1.2 (Exercise 26 from Diestel pg. 108 [9])**.** *The plane dual of a plane graph is connected.*

As we shall see in the next section, an Euler diagram's Jordan curves correspond to cycles in a plane graph; therefore, the following proposition is important because it relates these cycles to properties of the plane dual.

Proposition 4.1.3 (Proposition 4.6.1 from Diestel [9])**.** *For any connected plane graph $G = (V, E)$, a subset $E' \subseteq E$ of edges is a cycle in G if and only if the corresponding set of dual edges $E^* = \{e^* | e \in E\}$ is a minimal edge cut in G^*.*

Example. Figures 4.8(a) and (b) show examples of plane graphs (light gray) overlayed with their plane duals (black). In each case, a cycle in the plane graph is indicated by dashed light gray edges and the corresponding minimal edge cut in the

plane dual is indicated by dashed black edges. Note how each edge cut is minimal since adding one of the edges back reconnects the remaining plane dual vertices. In addition, since each edge cut is minimal, the remaining plane dual vertices are divided into exactly two connected components. □

Now that we have the basic terminology to describe graphs, and in particular plane graphs and plane duals, we can describe the association between Euler diagrams and graphs.

4.2 Euler Graphs and Duals

Consider the Euler diagram C in Fig. 4.9(a). There is a natural interpretation of C as a plane graph: place a vertex at each convergence point of C (see Def. 1.2.10 for the definition of a convergence point), and connect convergence points with edges that follow the curve segments between them. Concurrent curve segments resulting from the overlap of multiple curves are represented by a single edge, and the edge is labeled with the set of curves "containing" it. Additional vertices may also be required; for instance, c_6 is disjoint from every other curve in C, so it contains no convergence points and therefore cannot be represented by an edge unless a vertex is added at an arbitrary point in c_6. Although c_7 and c_8 are also disjoint from the other curves of C, together they contain two convergence points that suffice to act as vertices for their representative edges. Figure 4.9(b) shows the plane graph corresponding to C; this plane graph is called C's *Euler graph* and is formally specified in the following definition.

Definition 4.2.1. Let C be an Euler diagram. The *Euler graph* of C, denoted

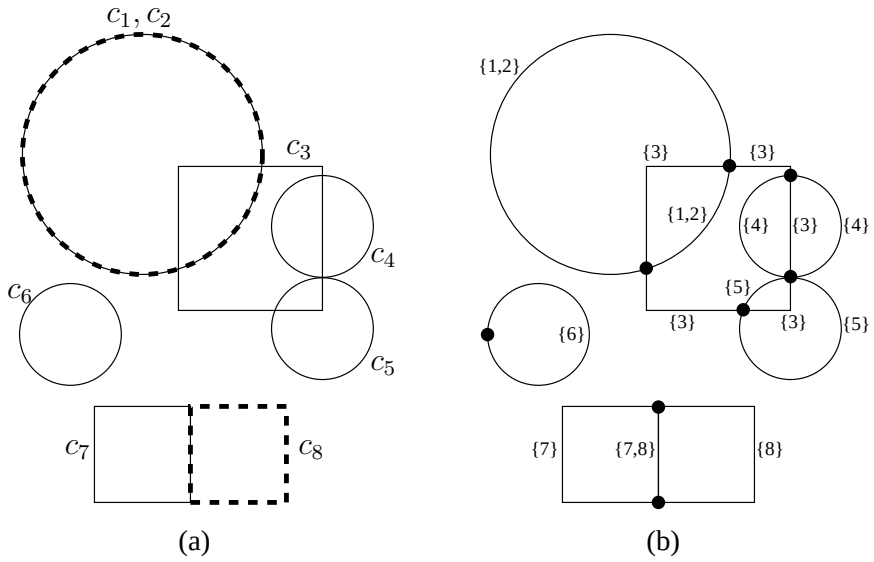


Figure 4.9: (a) An Euler diagram C and (b) its interpretation as an edge-labeled plane graph, the Euler graph $G(C)$.

$G(C) = (V, E)$, is an edge-labeled plane graph with a vertex at each convergence point of C and within (at an arbitrary point), any maximal connected subset $C' \subseteq pts(C)$ that does not contain a convergence point.

The vertex points $pts(V) = \{pt(v)|v \in V\}$ induce a partition of $pts(C)\backslash pts(V)$ into simple open curve segments, the endpoints of which coincide with the vertices. Each curve segment c between endpoints $pt(u)$ and $pt(v)$, where $u, v \in V$, is represented by an edge that connects u and v and follows c; that is, there is an edge $e = (u, v) \in E$ with $c(e) = c$.

Each edge e is labeled with the curves containing its representative curve segment; specifically,

$$l(e) = \{x|c_x \in C \text{ and } c(e) \subseteq c_x\}.$$

□

Example. Returning to the example shown in Fig. 4.9, we note that only one maximal connected subset of $pts(C)$, specifically c_6, lacks a convergence point and therefore has a non-convergence point vertex located within it. Conversely, $c_7 \cup c_8$ is also a maximal connected subset of $pts(C)$, but it contains two convergence points, so no additional vertex is needed. We also note that $G(C)$ is *not necessarily* a simple graph; in this example, $G(C)$ has several parallel edges (e.g., the three located in $c_7 \cup c_8$), as well as a loop in c_6. □

In the definition of an Euler graph, there may be a need to add non-convergence point vertices. Since the additional vertices may be located *anywhere* within their associated maximal connected subsets, an Euler diagram may have many Euler graphs; however, as was the case with plane duals, these Euler graphs are topologically equivalent, so when specific vertex locations are unimportant, we will refer to any $G(C)$ as *the* Euler graph for C. In fact, as stated in the next proposition, the conditions under which an Euler graph would contain a non-convergence point vertex are very limited.

Proposition 4.2.1. *Let C be a connected Euler diagram. The Euler graph $G(C)$ has at most one non-convergence point vertex, and such a vertex is present if and only if all of C's curves are equivalent.*

Proof. Since C is connected, the only maximal connected subset of $pts(C)$ is $pts(C)$ itself; therefore, $G(C)$ can contain at most one non-convergence point vertex. This non-convergence point vertex is added to $G(C)$ if and only if $pts(C)$ contains no convergence points. It is clear that if any two of C's curves are non-equivalent they must induce a convergence point either between each other or via an intermediate curve;

otherwise, C would be disconnected. As a result, $pts(C)$ contains no convergence points (and thus $G(C)$ contains a non-convergence point vertex), if and only if all of C's curves are equivalent, □

The next (trivial, but important) proposition relates the connectedness of an Euler diagram to the connectedness of its Euler graph.

Proposition 4.2.2. *An Euler diagram is connected if and only if its Euler graph is connected.*

Proof. Let C be an Euler diagram with Euler graph $G(C)$. It is clear that in the construction of $G(C)$, $pts(G(C)) = pts(C)$. As a result, the proposition follows since C is a connected Euler diagram if and only if $pts(C) = pts(G(C))$ is a connected set, and $pts(G(C))$ is a connected set if and only if $G(C)$ is a connected graph. □

Since a connected Euler diagram has a connected Euler graph, which is itself a connected plane graph, by Prop. 4.1.1, a connected Euler diagram has an associated plane dual; the following definition specifies the nature of this dual.

Definition 4.2.2. Let C be a connected Euler diagram. The *Euler dual* of C, denoted $G^*(C) = (V, E)$, is the edge-labeled plane dual of the Euler graph $G(C)$. Each vertex $v \in V$ is identified by the region of C containing it (i.e., $G^*(C)$ is a graph on $R(C)$), and each edge $e \in E$ is given the same label as its corresponding edge in $G(C)$. □

Example. Figure 4.10(a) and (b) shows a connected sub-diagram C of Fig. 4.9 along with its corresponding Euler graph $G(C)$. The Euler dual $G^*(C)$ is shown in Figs. 4.10(c) and (d) with the Euler graph shaded in the background for reference; to reduce clutter, the vertex and edge labels are separated. □

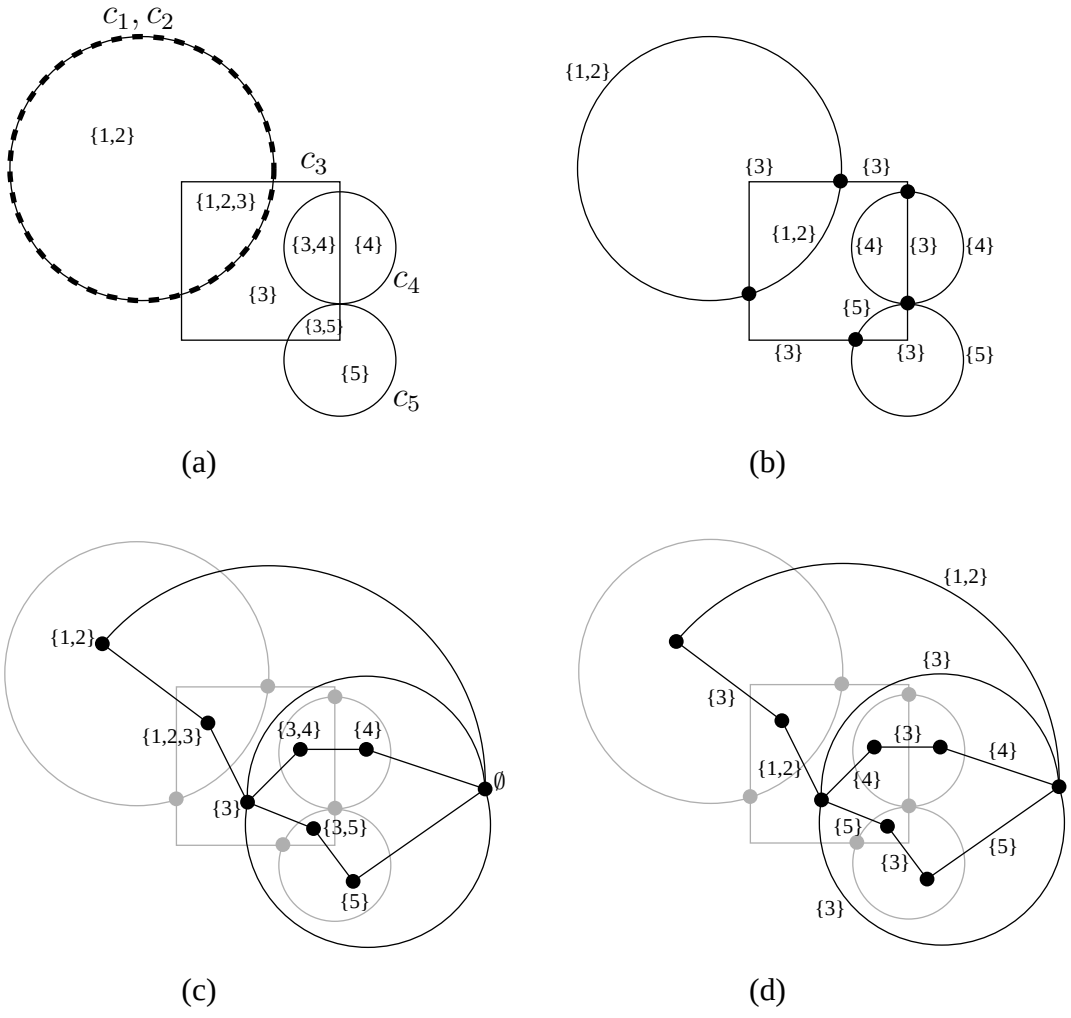


Figure 4.10: (a) A connected Euler diagram C, (b) its Euler graph $G(C)$, and (c)(d) its Euler dual $G^*(C)$ with vertices and edges labeled separately for clarity.

An important aspect of Euler duals, which is illustrated by Figs. 4.10(c) and (d), is that the edge labels can be derived from the endpoint vertices. By nature of the way an Euler diagram's regions are defined, an Euler dual edge $e = (u, v)$ is labeled

$$l(e) = (u \cup v) \backslash (u \cap v) = u \Delta v$$

where Δ is the symmetric set difference (i.e., the set of elements whose membership differs between the sets). As a result, we will generally omit the edge labels when drawing Euler duals, and instead assume they are implicit.

Definition 4.2.2 is fundamental to the remainder of this dissertation, so we wish to elaborate on some of its aspects before proceeding to the results. Of subtle importance is the requirement that the Euler diagram (and thus its Euler graph), be connected in order to have an Euler dual. Figures 4.11(a) and (b) show a disconnected Euler diagram C and its corresponding Euler graph $G(C)$. Even though the Euler graph is disconnected, the plane dualization operation can still be applied as shown in Fig. 4.11(c); this graph, which we call the "pseudo plane dual", does not satisfy the definition of a plane dual because the mapping of vertices from $G(C)$ to faces of the dual is not bijective (re: the shaded vertices with black outline). If one applies the plane dualization operation to the pseudo plane dual, as shown in Fig. 4.11(d), the resulting graph is *not* isomorphic to the original Euler graph; however, we can see by the curve labeling that it *is* the Euler graph for a connected version of C (the three non-bijective vertices of $G(C)$ get mapped to a single vertex connecting the graph). As demonstrated, in the case of disconnected Euler diagrams/graphs, the plane dualization operation is not a "true" dual operation in the sense that $G^*(G^*(C))$

is not isomorphic to $G(C)$; this is the reason why we restrict the definition of an Euler dual to connected Euler diagrams.

Fortunately, the requirement that an Euler diagram be connected in order to have an Euler dual is not overly strict (as hinted to by Fig. 4.11(d)), since it can be easily seen that a series of continuous transformations of the plane may be applied to make all the curves of an Euler diagram touch each other, and thus making the diagram connected. Figure 4.12(a) shows a disconnected Euler diagram C and its continuous transformation to a connected Euler diagram C' in Fig. 4.12(b); the resulting Euler graph and Euler dual are also shown. Although the connectedness restriction imposed by the definition of an Euler dual can be mitigated, as we shall see in Chapter 6, disconnected Euler diagram cannot be entirely ignored.

An interesting property of Euler duals is that there is no dependency between simplicity of the Euler graph and simplicity of the Euler dual. For example, Fig. 4.13(a) shows a non-simple Euler graph (light gray) whose Euler dual (black) is *simple* Conversely, Fig. 4.13(b) shows a simple Euler graph (light gray) whose Euler dual (black) is *non-simple*.

Although the independence of Euler graphs and Euler duals with respect to graph simplicity is a general feature of plane graphs and plane duals, there are some properties that are specific to Euler duals; in the next section, we consider these properties.

4.3 Euler Dual Properties

Although an Euler graph is a plane graph, its derivation from an Euler diagram results in certain specific properties not common to general plane graphs. Similarly, since

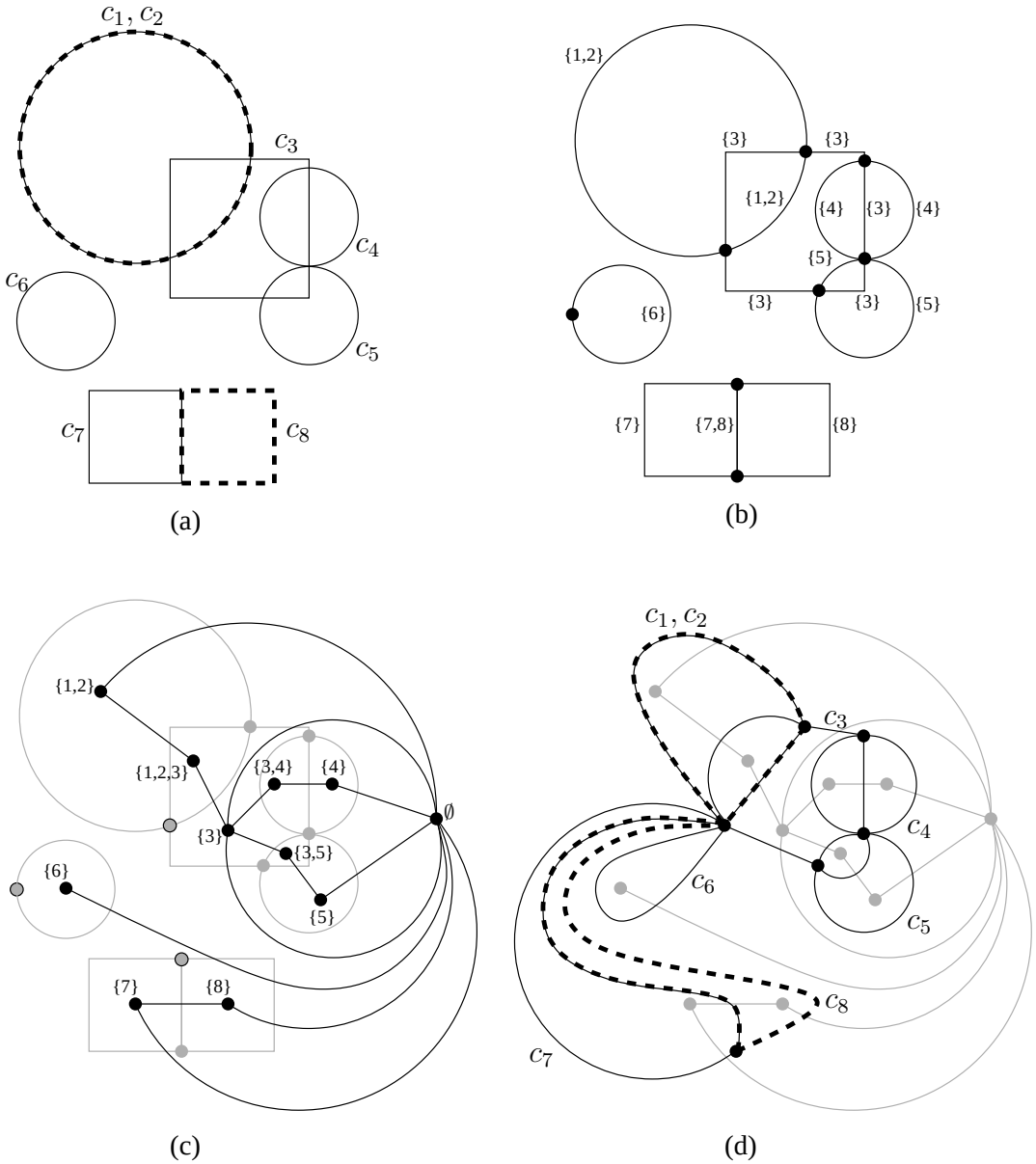


Figure 4.11: (a) A disconnected Euler diagram C, (b) its disconnected Euler graph $G(C)$, (c) the "pseudo plane dual" of $G(C)$; this is not a plane dual because multiple vertices of $G(C)$ (shaded with black outline) are mapped to a single dual face, and (d) the plane dual of (c) is not isomorphic to $G(C)$, although it is the Euler graph for an Euler diagram similar to C (as shown by the labeled curves).

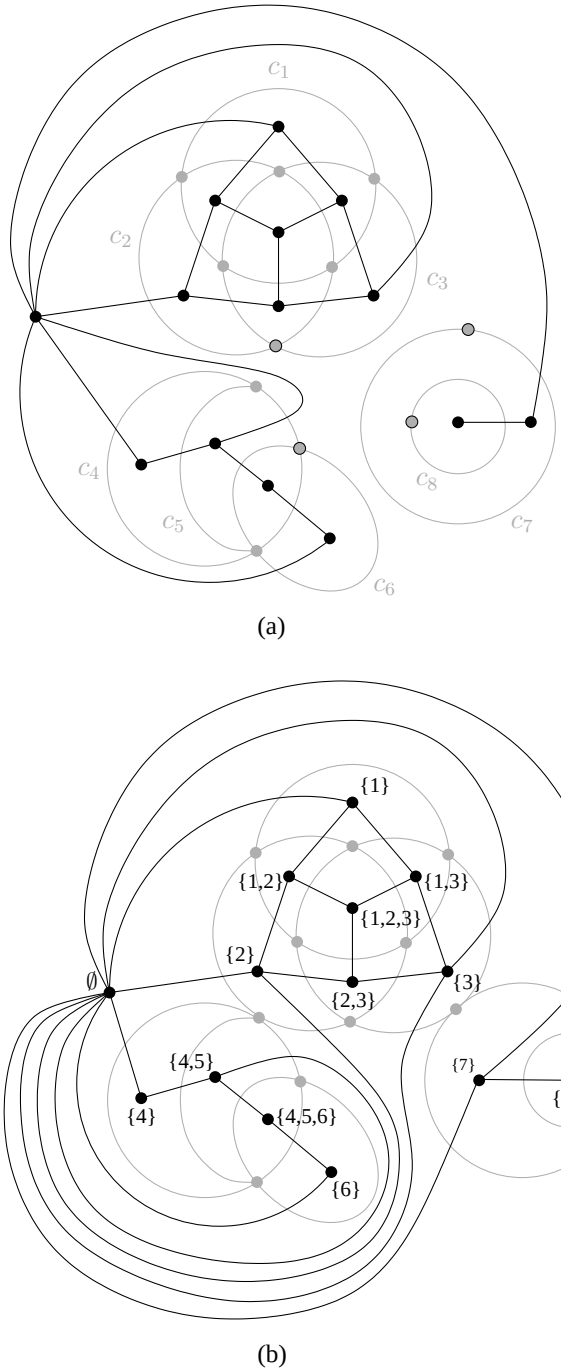


Figure 4.12: (a) A disconnected Euler diagram C with Euler graph $G(C)$ (light gray) and "pseudo plane dual" (black), and (b) a continuous transformation to a connected Euler diagram C' with Euler graph $G(C')$ (light gray), and Euler dual $G^*(C')$ (black).

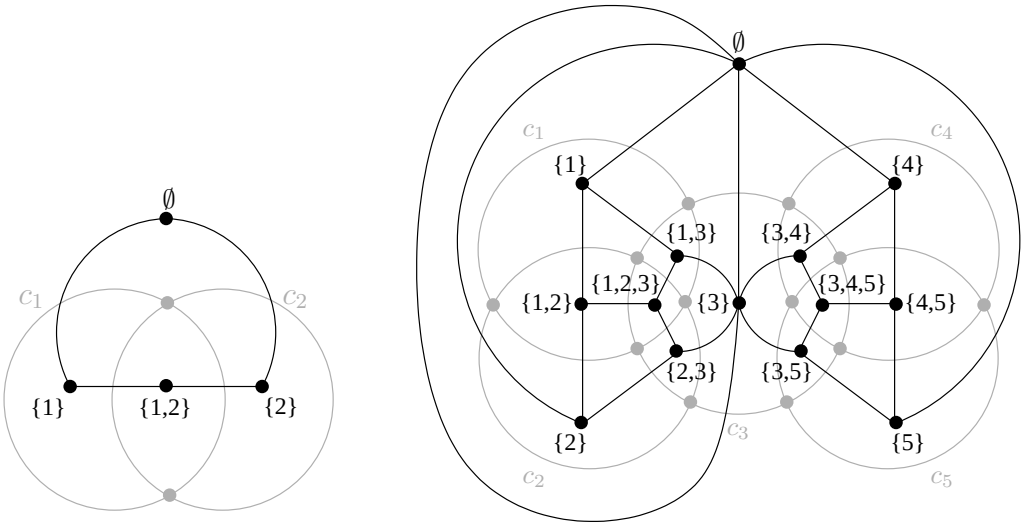


Figure 4.13: (a) A non-simple Euler graph (light gray) that has a simple Euler dual (black), and conversely, (b) a simple Euler graph that has a non-simple Euler dual (there are two $(\{3\}, \emptyset)$ edges).

Euler duals are derived from Euler graphs, they too have properties not common to general plane duals. The following propositions describe some of these specific Euler dual properties.

Proposition 4.3.1. *An Euler dual is loop-free.*

Proof. Let C be a connected Euler diagram whose Euler dual $G^*(C)$ has a loop $e^* = (v, v)$, and let e be the corresponding edge in the Euler graph $G(C)$. By definition, e represents a segment of some curve $c_i \in C$ (it may in fact represent several curve segments, but without loss of generality, we need only consider one), with region v on either side of it; however, this contradicts the Jordan Curve Theorem (Thm. 1.2.1), which states that c_i separates *distinct* interior and exterior subsets of the plane, which implies distinct regions, so $G^*(C)$ cannot have a loop. □

Proposition 4.3.2. *Let C be a connected Euler diagram with Euler dual $G^* = G^*(C)$. The Euler dual $G^* = G^*(C)$ has the property that for each curve $c_x \in C$, G^*_x is connected and $\overline{G^*_x}$ is connected, where G^*_x is the subgraph of G^* induced by vertices containing x and $\overline{G^*_x}$ is the subgraph of G^* induced by vertices* not *containing x.*

Proof. Consider the Euler graph $G(C)$ and for each curve $c_x \in C$, let E_x be the subset of $G(C)$'s edges that represent curve $c_x \in C$ and E^*_x be the corresponding dual edges in $G^*(C)$.

By the Jordan Curve Theorem (Thm. 1.2.1), the edges in E_x are exactly those that separate a face interior to c_x from a face exterior to c_x. As a result,

$$E^*_x = \{e^* = (u, v) \in G^* | x \in u \text{ and } x \notin v\}.$$

In addition, since E_x corresponds to a Jordan curve, it represents a cycle in $G(C)$, so by Prop. 4.1.3, E^*_x is a minimal edge cut of $G^*(C)$.

Since x is contained by every vertex in G^*_x and not contained by every vertex in $\overline{G^*_x}$, any path between a vertex $u \in G^*_x$ and a vertex $v \in \overline{G^*_x}$ must contain an edge from E^*_x, so E^*_x is an edge cut that divides G^*_x and $\overline{G^*_x}$. Finally, since E^*_x is a *minimal* edge cut, it divides $G^*(C)$ into two connected components, one of which must be G^*_x and the other $\overline{G^*_x}$. □

Example. Figure 4.14 shows several subgraphs of the Euler dual $G^* = G^*(C)$ from Fig. 4.10(c). In Fig. 4.14(a), the edges corresponding to E^*_1 from the proof of Prop. 4.3.2 are removed to demonstrate how each of G^*_1 and $\overline{G^*_1}$ is a connected component of the resulting graph. Also note how the edges removed are exactly those that cross edges of the Jordan curve c_1 (light gray). Since c_1 and c_2 are equivalent, Fig.

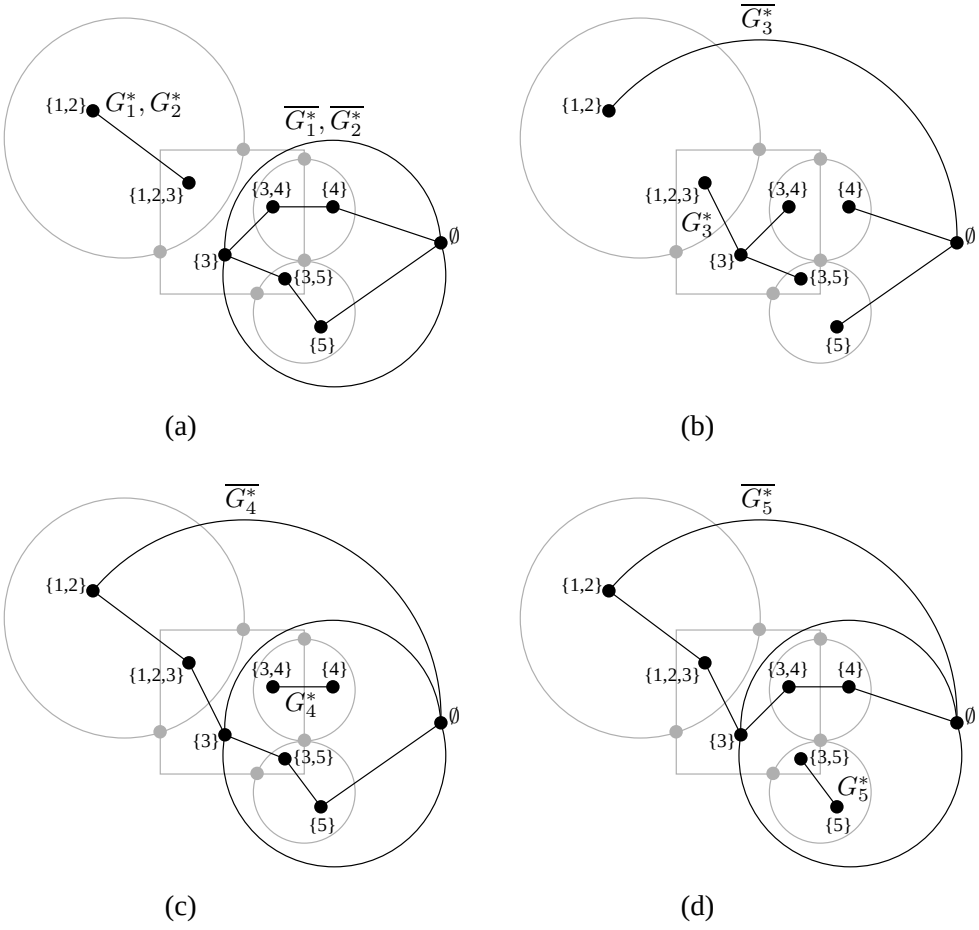


Figure 4.14: The Euler dual G^* from Fig. 4.10(c) with the edge cut E_x^* removed to show how G_x is connected and $\overline{G_x}$ is connected for (a) $x = 1$ and $x = 2$, (b) $x = 3$, (c) $x = 4$, and (d) $x = 5$.

4.14(a) also shows the layout of G_2^* and $\overline{G_2^*}$. The remaining diagrams shown in Fig. 4.14(b)–(d) show the layouts of $G_3^*/\overline{G_3^*}$, $G_4^*/\overline{G_4^*}$, and $G_5^*/\overline{G_5^*}$, respectively. □

Proposition 4.3.3. *Let C be a connected Euler diagram with Euler dual $G^*(C)$. For each face $f \in G^*(C)$ and curve $c_x \in C$, a traversal of the boundary edges δf encounters zero or two (possibly repeated) edges whose labels contain x.*

In other words, if $\delta_x f$ is the subsequence defined as

$$\delta_x f = [e \in \delta f | x \in l(e)]$$

then $|\delta_x f| = 0$ or $|\delta_x f| = 2$.

Proof. Suppose we overlay $G^*(C)$ with its corresponding Euler graph $G(C)$ as shown in Fig. 4.10(d). By the plane dual definition (Def. 4.1.9), there is a bijection between Euler graph vertices and Euler dual faces. In addition, this bijection specifies that each Euler graph vertex v is contained by *exactly* one Euler dual face $f^*(v)$, and the boundary edges $\delta f^*(v)$ are the Euler dual edges that correspond to *exactly* the Euler graph edges representing curve segments intersecting at v. As a result, if curve c_x does not pass through v, x will appear in none of $\delta f^*(v)$'s labels and $|\delta_x f^*(v)| = 0$. If curve c_x does pass through v, then x will be in at least one of $\delta f^*(v)$'s labels. In addition, since c_x is a closed curve, it must continue out of v, and since there is no other Euler graph vertex contained by $f^*(v)$, x must appear in another one of $\delta f^*(v)$'s labels. Lastly, c_x cannot re-enter $f^*(v)$ for otherwise, it would intersect itself at v and be non-simple. As a result, if c_x passes through v, x will appear in exactly two of $\delta f^*(v)$'s labels and $|\delta_x f^*(v)| = 2$. □

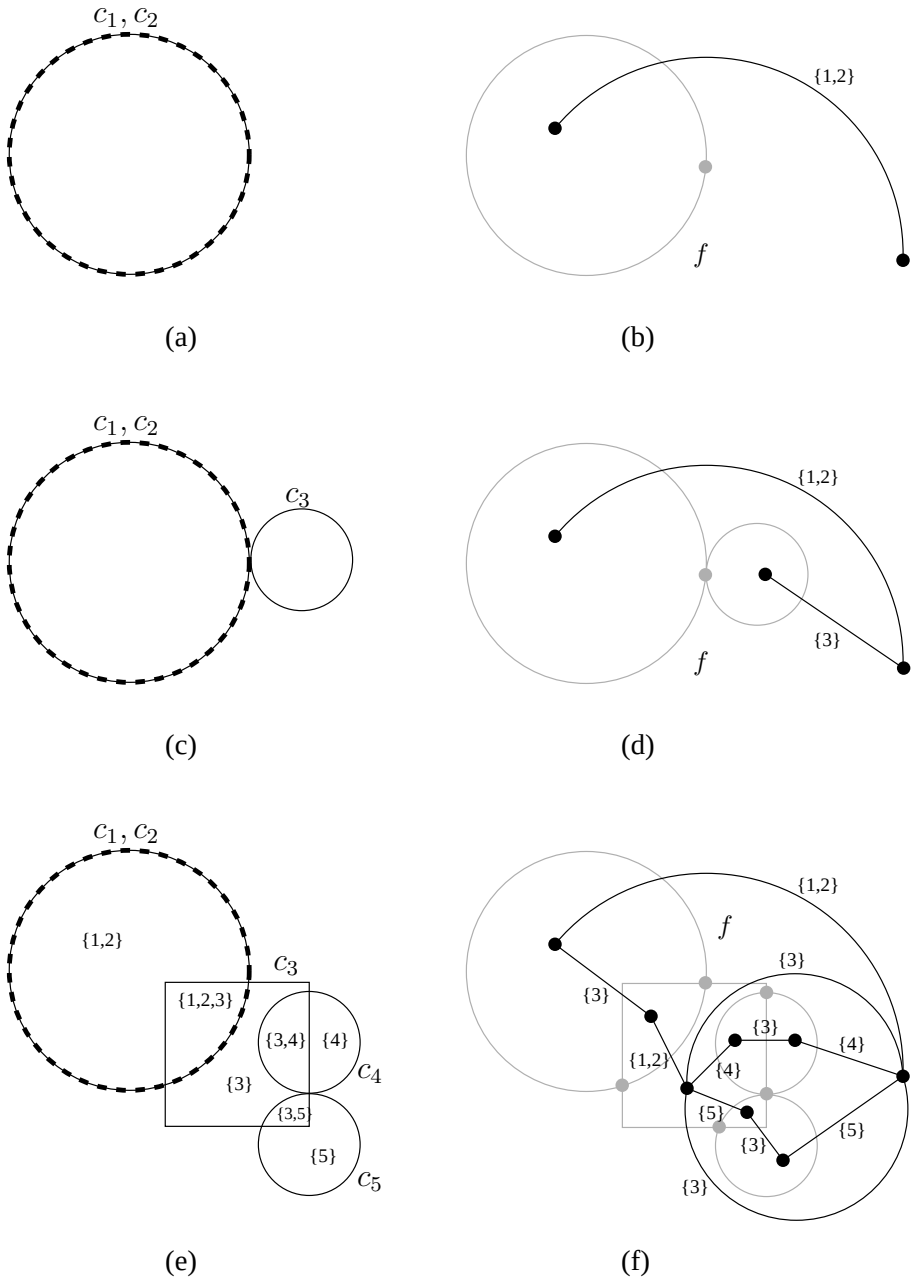


Figure 4.15: (a)(c)(e) Three examples of connected Euler diagrams along with their (b)(d)(f) corresponding Euler duals. Note that an Euler dual face f is identified in each example, and a traversal of the boundary edges δf encounters each curve label exactly zero of two times.

Example. Figure 4.15(a) shows a connected Euler diagram whose Euler graph contains a non-convergence point vertex (by Prop. 4.2.1 we know that all such Euler graphs look similar). Figure 4.15(b) shows the corresponding Euler dual with a single face f. A clockwise traversal of the boundary edges δf will encounter the only Euler dual edge twice, so in this case, $\delta_1 f = \delta_2 f = 2$.

Figure 4.15(c) shows another connected Euler diagram whose Euler dual in Fig. 4.15(d) has a single face f. As in the previous example, each Euler dual edge is encountered twice during a traversal of δf, so $\delta_1 f = \delta_2 f = \delta_3 f = 2$.

Lastly, Fig. 4.15(c) shows the connected Euler diagram from Fig. 4.10(a) along with its Euler dual in Fig. 4.15(d). One Euler dual face f is identified in Fig. 4.15. Unlike the previous examples, a clockwise traversal of δf does not encounter any repeated edges; however, the labels encountered will be a cyclic permutation of $\{1,2\}$, $\{3\}$, $\{1,2\}$, $\{3\}$. As a result, $\delta_1 f = \delta_2 f = \delta_3 f = 2$ and $\delta_4 f = \delta_5 f = 0$. □

The previous propositions describe properties that are held by all Euler duals, but not necessarily plane duals in general. In the next section, we consider how specific Euler diagram properties such as non-concurrent/concurrent, pairwise/non-pairwise, and simple/non-simple, are related to properties of the corresponding Euler dual.

4.4 Euler Diagram and Euler Dual Properties

By their definition, Euler graphs are a direct graph model of an Euler diagram's curves. Although slightly more abstract, Euler duals are in a sense more powerful in that they model both an Euler diagram's curves *and* regions; as an example, we will refer to the Euler diagram C in Fig. 4.10(a) along with its corresponding Euler graph

$G(C)$ and Euler dual $G^*(C)$.

In terms of curves, any dual edge e with $|l(e)| > 1$ represents a concurrent curve segment in C. For example, in Fig. 4.10(d), dual edges $e_1 = (\emptyset, \{1, 2\})$ and $e_2 = (\{1, 2, 3\}, \{3\})$ have $l(e_1) = l(e_2) = \{1, 2\}$ and represent curve segments in Fig. 4.10(a) that are contained by both c_1 and c_2.

Additionally, the union of boundary edge labels for a dual face f tells us which curves intersect at the vertex in $G(C)$ corresponding to f; this allows us to determine whether or not C has non-pairwise intersections. For example, the dual face in Fig. 4.10(c) with the boundary vertex traversal

$$\emptyset \to \{4\} \to \{3, 4\} \to \{3\} \to \{3, 5\} \to \{5\} \to \emptyset$$

has the corresponding edge label traversal

$$\{4\} \to \{3\} \to \{4\} \to \{5\} \to \{3\} \to \{5\}$$

whose union is $\{3, 4, 5\}$, and we can see that the vertex in $G(C)$ corresponding to f represents the intersection of c_3, c_4, and c_5 in Fig. 4.10(a).

In terms of regions, if there is a dual edge (u, v) then we know that region u and region v are adjacent in C by virtue of their boundaries intersecting along a curve segment. We can also quickly ascertain the regions of C that are adjacent to the outer region by looking at vertices adjacent to $G^*(C)$'s $\emptyset$ vertex. For example, in Fig. 4.10(c), the vertices

$$\{1, 2\}, \{3\}, \{4\}, \text{ and } \{5\}$$

are adjacent to $\emptyset$, and these are exactly the regions in Fig. 4.10(a) that share a boundary with the outer region.

Because Euler duals encode so much information about their corresponding Euler diagrams, it seems natural to consider how Euler diagram properties are related to Euler dual properties. The following propositions describe these relationships.

Proposition 4.4.1. *A connected Euler diagram C is* concurrent *if and only if its Euler dual $G^*(C)$ has an edge e with $|l(e) > 1|$.*

Proof. A trivial consequence of the construction of $G^*(C)$ described in Defs. 4.2.1 and 4.2.2. □

Example. Figure 4.16(a) shows a concurrent Euler diagram along with its Euler dual in Fig. 4.16(b). In this case, the concurrent curve segment shared by c_1 and c_2 is represented by an Euler dual edge e with $l(e) = \{1, 2\}$. As a result, by Prop. 4.4.1, the Euler diagram is correctly identified as being concurrent.

In contrast, Fig. 4.16(c) shows a non-concurrent Euler diagram along with its Euler dual in Fig. 4.16(d). In this case, every Euler dual edge has a singleton label. As a result, by Prop. 4.4.1, the Euler diagram is correctly identified as being non-concurrent. □

Proposition 4.4.2. *A connected Euler diagram C is* non-pairwise *if and only if its Euler dual $G^*(C)$ has a face f whose union of boundary edge labels contains more than two curve labels (i.e., $|\bigcup_{e \in \delta f} l(e)| > 2$).*

Proof. Sufficiency Let v be a vertex of the Euler graph $G(C)$ contained by an Euler dual face $f^*(v)$ with $|\bigcup_{e \in \delta f^*(v)} l(e)| > 2$. By the Euler dual construction, the labels

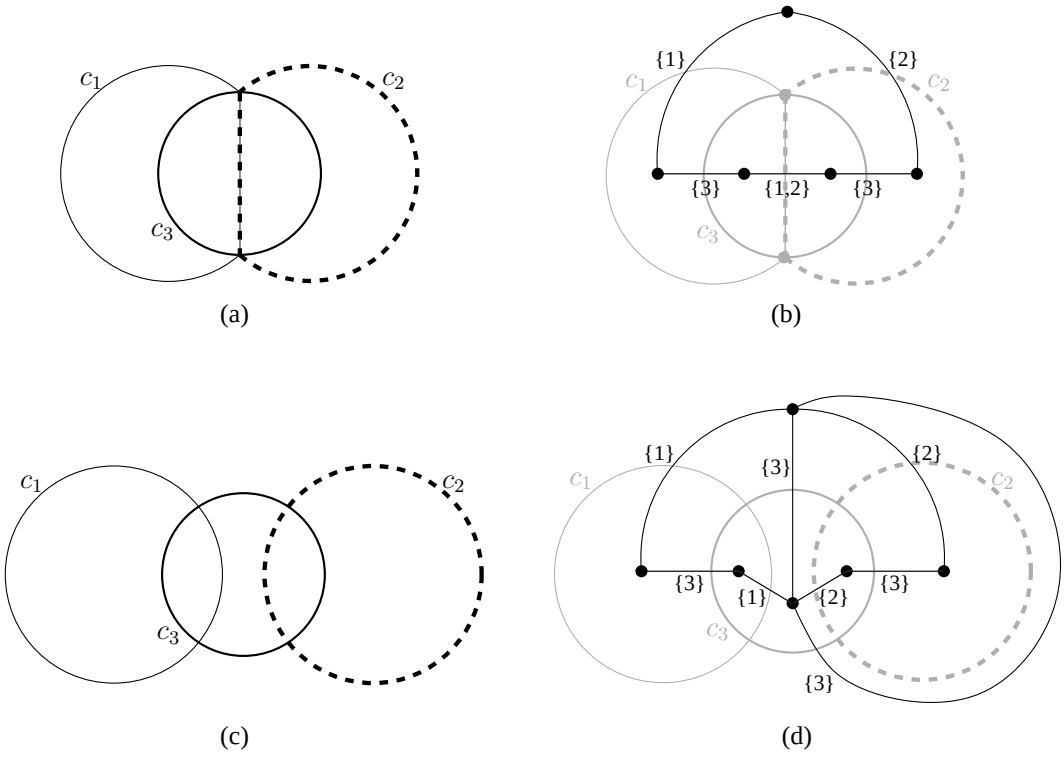


Figure 4.16: (a) A concurrent Euler diagram has (b) an Euler dual with an edge $e = \{1, 2\}$ such that $|l(e)| > 1$, and (c) a non-concurrent Euler diagram has (d) an Euler dual with $|l(e)| = 1$ for every edge e.

of $\delta f^*(v)$ identify exactly the curves that intersect at v, regardless of whether or not v is a non-convergence point vertex. Since $|\bigcup_{e\in\delta f^*(v)} l(e)| > 2$, this implies that at least three curves intersect at v, so $pt(v)$ is a non-pairwise intersection of C.

Necessity Suppose C is non-pairwise and let p be a point shared by more than two curves. If p does not correspond to an Euler graph vertex then it is the interior point of a concurrent curve segment c_s shared by more than two curves. If c_s has endpoints, then either will correspond to a convergence point, and this convergence point will be a non-pairwise intersection. If c_s has no endpoints, then it is a closed curve, so the Euler graph will contain a non-convergence point vertex located somewhere on c_s the location of which will be a non-pairwise intersection. As a result, without loss of generality, we can assume that p is the location of an Euler graph vertex v.

Let $f^*(v)$ be the unique Euler dual face containing v. By the Euler dual construction, the labels of $\delta f^*(v)$ identify exactly the curves that intersect at v. Since v is a non-pairwise intersection, at least three curves intersect at v, so $|\bigcup_{e\in\delta f^*(v)} l(e)| > 2$. □

Example. Consider the Euler diagram and Euler dual in Figs. 4.16(a) and (b). Each Euler dual face has the union of boundary edge labels

$$\{1\} \cup \{2\} \cup \{3\} \cup \{1,2\} \cup \{3\} = \{1,2,3\},$$

which contains more than two curves labels. As a result, the Euler diagram is correctly identified as being non-pairwise.

In contrast, the Euler dual in Fig. 4.16(d) has faces whose union of boundary edge labels is either $\{1,3\}$ or $\{2,3\}$. As a result, by Prop. 4.4.2, the corresponding

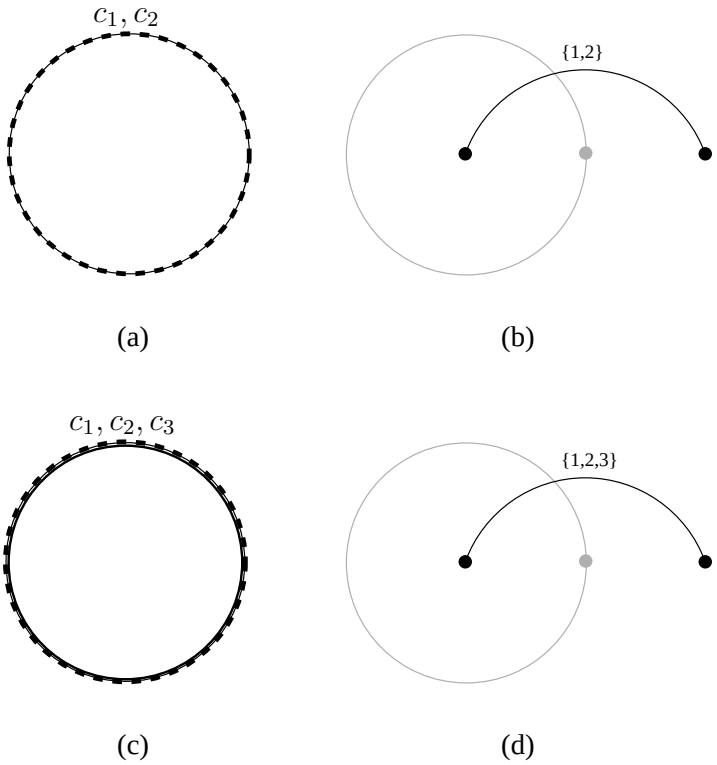


Figure 4.17: Prop. 4.4.2 holds even when the Euler graph contains a non-convergence point vertex.

Euler diagram in Fig. 4.16(c) is correctly identified as being pairwise.

Prop. 4.4.2 holds even when the Euler graph contains a non-convergence point vertex. For example, Fig. 4.17(a) shows a pairwise Euler diagram along with its Euler dual in Fig. 4.17(b). Note that the Euler graph's only vertex does not correspond to a convergence point; however, the union of boundary edge labels for the Euler dual's only face is $\{1, 2\}$. As a result, by Prop. 4.4.2, the Euler diagram is correctly identified as being pairwise.

On the other hand, Fig. 4.17(c) shows a non-pairwise Euler diagram that results from adding an additional equivalent curve to the previous Euler diagram. As shown in Fig. 4.17(d), the union of boundary edge labels for the Euler dual's only face is now $\{1, 2, 3\}$. As a result, the Euler diagram is correctly identified as being non-pairwise.

Incidentally, Fig. 4.17(a) is a representation of the *only* set system (modulo relabeling), that *must* be represented by a connected Euler that is both pairwise *and* concurrent; any additional curves, whether non-equivalent or equivalent, result in a non-pairwise intersection as exemplified by the Euler diagrams in Fig. 4.15(c) and Fig. 4.17(c). □

Proposition 4.4.3. *A connected Euler diagram C is* simple *if and only if $|C| = 1$ or its Euler dual $G^*(C)$ has the property that $|l(e)| = 1$ for every edge e and $deg(f) = 4$ for every face f.*

Proof. Clearly, when $|C| = 1$, the Euler diagram is simple, so consider the case when $|C| > 1$.

Since a simple Euler diagram is both non-concurrent and pairwise, we can combine the contrapositive implications of Props. 4.4.1 and 4.4.2 to state that C is a simple

Euler diagram if and only if for every edge e in its Euler dual, $|l(e)| \leq 1$, and for every face f in its Euler dual, $|\bigcup_{e \in \delta f} l(e)| \leq 2$.

Suppose there is an Euler dual edge $e = (u, v)$ with $|l(e)| = 1$. Since $l(e) = u\Delta v$, $|l(e)| = 1$ implies that $u = v$ and e is a loop; however, this contradicts Prop. 4.3.1. As a result, in the conditions for a simple Euler diagram to exist, it must be that the Euler dual has $|l(e)| = 1$.

Suppose there is an Euler dual face f with $|\bigcup_{e \in \delta f} l(e)| < 2$. Clearly, since $|l(e)| = 1$ and f has at least one boundary edge (since $|C| > 0$), it must be that $|\bigcup_{e \in \delta f} l(e)| = 1$. In other words, the Euler graph vertex corresponding to f is a non-convergence point vertex that bisects a single curve. By Prop. 4.2.1, the presence of a non-convergence point vertex implies that all of C's curves are equivalent; however, since $|C| > 1$, this implies the contradiction that C contains at least two concurrent curves and thus has an edge e with $|l(e) > 1$. As a result, in the conditions for a simple Euler diagram to exist, it must be that the every Euler dual face f has $|\bigcup_{e \in \delta f} l(e)| = 2$.

It remains to show that under the condition that every Euler dual edge has a singleton label, $|\bigcup_{e \in \delta f} l(e)| = 2$ implies that $deg(f) = 4$.

Consider any Euler dual face f and let x_1 and x_2 be the two curve labels appearing in δf's labels. By Prop. 4.3.3, x_1 and x_2 must appear exactly twice as the label of an edge in δf; no other curve label can appear in δf since the union of δf's labels is a 2-set. Since the edge labels are singletons, x_1 and x_2 cannot appear together as the label of an edge in δf, so δf must contain exactly 4 edges, which implies that $deg(f) = 4$. □

Example. The Euler diagram C in Fig. 4.18(a) is the special case of Prop. 4.4.3. Since C has a single curve, its Euler dual has a single face, but its degree is 2 as

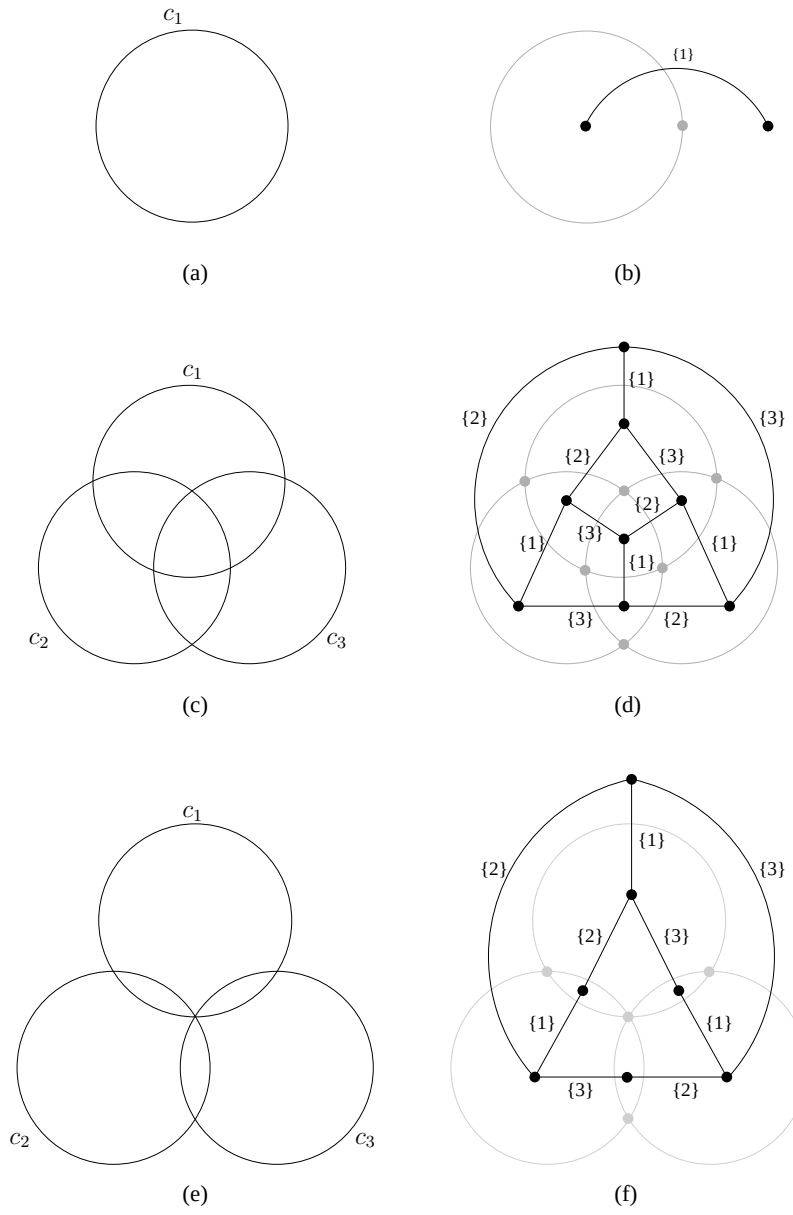


Figure 4.18: (a) A simple Euler diagram with a single curve is the special case of Prop. 4.4.3 since (b) its Euler dual (black) has a single face with degree 2. (c) A simple Euler diagram with more than one curve is the normal case of Prop. 4.4.3 since (d) its Euler dual (black) has edges with singleton labels and faces with degree 4. (e) Although this non-simple Euler diagram has (f) an Euler dual (black) whose edges have singleton labels, there is a face with degree 6, which corresponds to a non-pairwise intersection.

shown in Fig. 4.18(b).

Figure 4.18(c) shows an Euler diagram with more than one curve. In this case, the corresponding Euler dual shown in Fig. 4.18(d) has all its edges labeled with singleton sets and all its faces have degree 4. As a result, by Prop. 4.4.3, the Euler diagram is correctly identified as being simple.

Finally, Fig. 4.18(e) shows a non-simple Euler diagram. In this case, although the corresponding Euler dual shown in Fig. 4.18(f) has all its edges labeled with singleton sets, one of its faces has degree 6, and this corresponds to an intersection involving three curves. As a result, by Prop. 4.4.3, the Euler diagram is correctly identified as being non-simple. □

As we have seen, Euler duals encode many of the essential topological properties of Euler diagrams. In addition, because the Euler dual is a finite data structure, in many cases it is easier to reason with than its continuous counterpart, the Euler diagram. The utility of Euler duals is particularly evident with respect to the Euler Diagram Generation Problem (EDGP). In the next chapter, we develop necessary-and-sufficient conditions for the existence of Euler diagrams based on the Euler dual properties described in this section.

Chapter 5

Necessary-and-Sufficient Conditions for the Existence of Euler Diagrams

Suppose S is a set system on the items X and C is a connected Euler diagram representing S. As we have seen, the Euler dual $G^*(C)$ has the following properties:

1. $G^*(C)$ is a connected plane graph on S,
2. $G^*(C)$ is loop-free,
3. $G^*(C)$ may or may not have parallel edges, and
4. for each item $x \in X$, $G^*_x(C)$ is connected and $\overline{G^*_x(C)}$ is connected.

Note that with property 3, if $G^*(C)$ has no parallel edges, it is a *simple* graph because property 2 states that it is loop-free.

By definition, an Euler dual is derived from an Euler graph, which itself is derived from an Euler diagram; suppose we reversed these operations. That is, we start with

a graph G on S that has the properties of an Euler dual, compute its plane dual G^*, and then interpret G^* as an Euler diagram by grouping its edges into Jordan curves. Are we guaranteed to produce an Euler diagram that represents C? In other words, are the Euler dual properties both necessary *and* sufficient for a graph to be an Euler dual? As we shall see in this section, the answer is "Yes". We begin by defining what it means for a graph to have the Euler dual properties.

Definition 5.0.1. Let S be a set system on the items X. A graph G is said to be a *connectivity graph* for S if an only if the following properties hold for G:

1. G is a connected planar graph on S,
2. G is loop-free, and
3. for each item $x \in X$,
   (a) the subgraph G_x induced by the vertices containing x is connected, and
   (b) the subgraph $\overline{G_x}$ induced by the vertices *not* containing x is connected.

If, in addition, G is a simple graph, then we refer to G as a *simple connectivity graph.* □

In the connectivity graph definition, we relax the Euler dual property of being a *plane* graph, and only require G to be a *planar* graph. The reason for this relaxation will become evident in the next result where we show that *any* plane embedding of G suffices to define an Euler dual.

Example. Consider the set system

$$S = \{\emptyset, \{1\}, \{1,3\}, \{1,3,4\}, \{1,5\}, \{1,5,6\}, \{2\}, \{2,3\}, \{2,3,4\}\}$$

on the items $X = \{1,2,3,4,5,6\}$.

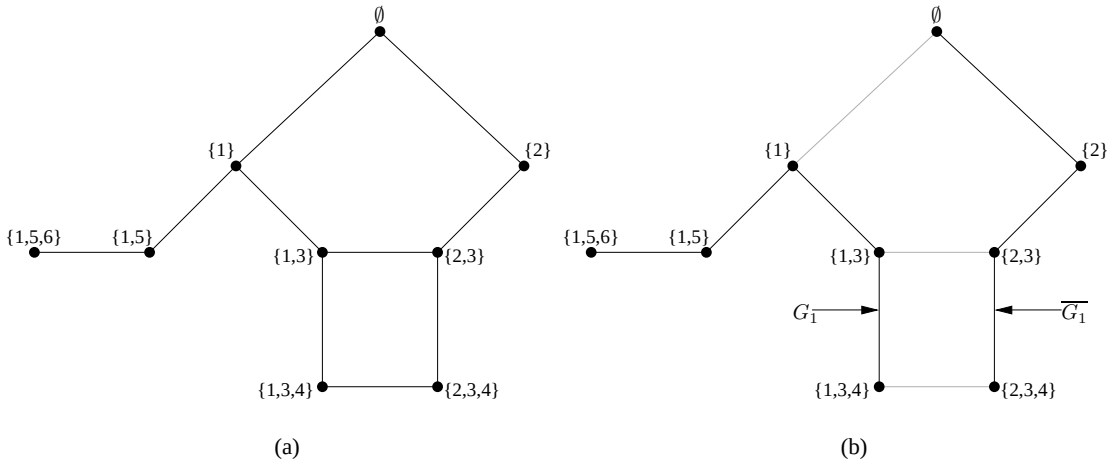


Figure 5.1: (a) A connectivity graph G for a set system S, and (b) an example of the connected subgraphs G_1 and $\overline{G_1}$ (the shaded edges are not present in either subgraph).

Figure 5.1(a) shows a connectivity graph G for S; note how each set in S is a vertex of G. Figure 5.1(b) shows the connected subgraphs G_1 and $\overline{G_1}$; for compactness, the subgraphs (black) appear concurrently, but in reality, G_1 is the vertex-induced subgraph comprising *only* the five vertices containing 1, and similarly, $\overline{G_1}$ is the vertex-induced subgraph comprising *only* the four vertices *not* containing 1. The reader is left to verify that G_x is connected and $\overline{G_x}$ is connected for all $x \in X$. An important point regarding the connectivity graph definition is that it does not specify *which* edges must be present, only that the edges be sufficient to ensure G_x and $\overline{G_x}$ connectivity while not precluding G's planarity; as a result, there may be many non-isomorphic connectivity graphs for the same set system. □

We now consider how to transform a connectivity graph for a set system S into an Euler diagram representing S. The following result is fundamental to this dissertation.

Theorem 5.0.4. *Let G be a connectivity graph for the set system S. For* every *plane embedding G_e of G, there is a connected Euler diagram C representing S whose Euler*

dual $G^*(C)$ has the same combinatorial embedding as G_e (i.e., $G^*(C)$ is topologically equivalent to G_e modulo the choice of outer face).

Proof. Step 1: stereographic projection

When an Euler dual is overlayed on its corresponding Euler graph (see, for example, Fig. 5.2(a)), its $\emptyset$ vertex is in the Euler graph's outer face. Remembering that we want to show G_e is an Euler dual and that its plane dual G_e^* is an Euler graph, if we overlay G_e and G_e^*, then G_e's $\emptyset$ vertex may not be in G_e^*'s outer face; instead, it may be inside some other face $f_\emptyset$ as shown in Fig. 5.2(a). However, suppose we place a sphere on the plane so that its "south" pole touches inside $f_\emptyset$ and then wrap the plane onto the sphere so that the outer face is now a bounded face of the "north" pole. We now rotate the sphere so that the north pole touches the plane and unwrap it by opening it up at the south pole. The resulting drawing is shown in Fig. 5.2(b); note how the $f_\emptyset$ becomes the outer face, but the combinatorial embedding of the two graphs is the same (e.g., a clockwise traversal of each vertex's adjacent vertices in Fig. 5.2(a) is the same as in Fig. 5.2(b)). This graph transformation is called a *stereographic projection*. Since the stereographic projection preserves a graph's combinatorial embedding, without loss of generality, we can assume that G_e's $\emptyset$ vertex is in G_e^*'s outer face.

Step 2: creating Jordan curves

Since for every item $x \in X$, G_x is connected and $\overline{G_x}$ is connected, the subset E_x of G's edges connecting G_x to $\overline{G_x}$ is a minimal edge cut (i.e., $G \backslash E_x$ is disconnected, but becomes connected if any edge of E_x is added back). As a result, by Prop. 4.1.3, the edges of G_e^* that correspond to E_x form a cycle in G_e^*. Figure 5.3(a) shows the edges of E_1 (dashed) that separate G_1 and $\overline{G_1}$, and Fig. 5.3(b) shows the corresponding

cycle in G_e^*. If we treat the cycle induced by item $x \in X$ as a Jordan curve c_x then

$$C = \{c_x | x \in X\}$$

is the corresponding set of Jordan curves. Since G is loop-free, each edge $e \in G$ is between distinct vertices so there is some $x \in X$ with $e \in E_x$. In other words, every edge of G has a corresponding edge in G_e^* that is part of a cycle, so the drawing of C is the same as the drawing of G_e^* as shown in Fig. 5.5(b).

Step 3: Verifying that $R(C) = S$

At this point, C is a collection of Jordan curves whose regions correspond to the faces of G_e^*, so we know that each region is a connected open subset of the plane; what we do not know is the set system that C represents. We wish to show that the unique vertex $v \in G_e$ contained by each face $f^*(v) \in G_e^*$ is in fact the element of S represented by the region of C corresponding to $f^*(v)$. If this is the case, then since $V(G_e) = S$, C is an Euler diagram representing S. The key to this step of the proof is that G_e's $\emptyset$ vertex is in the outer face of G_e^*.

Consider a path p in G_e between vertex v and $\emptyset$; Fig. 5.4(a) shows an example path (dashed) from $\{1, 3\}$ to $\emptyset$. Along each edge $e \in p$, items are added and removed. Since each item $x \in v$ is not in $\emptyset$, the number of times x can be added or removed along p must be odd. Consider some path edge $e = (s, t)$ where $x \in s$ and $x \notin t$ (i.e., x is removed). By definition, $s \in G_x$ and $t \in \overline{G_x}$; therefore, e is in the minimal edge cut E_x and its corresponding dual edge is part of c_x; in other words, p crosses c_x, and this happens as well when x is added. As a result, for each item $x \in v$, any path from v to $\emptyset$ crosses c_x an odd number of times, and since $\emptyset$ is exterior to all curves, by the

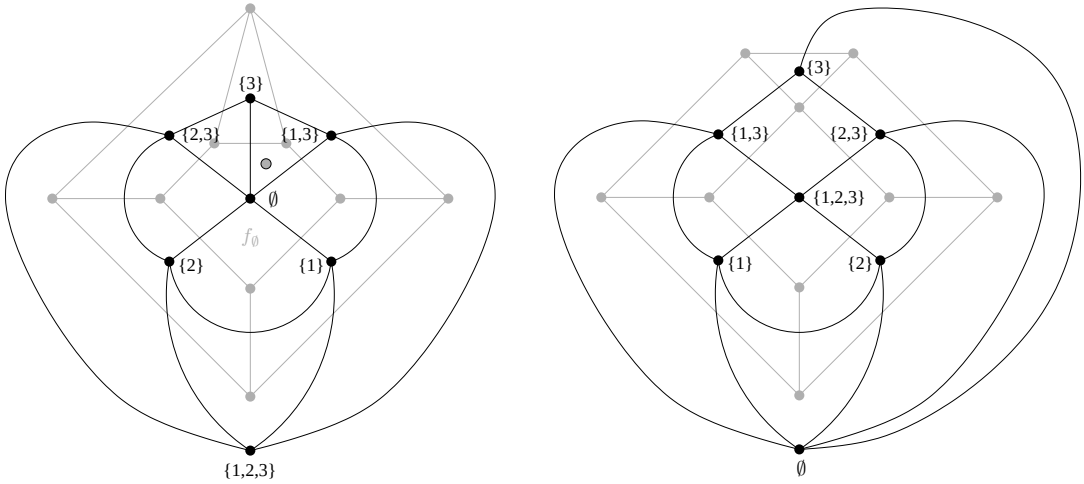


Figure 5.2: (a) An example of a plane embedding G_e of a connectivity graph (black) whose $\emptyset$ vertex is *not* in the outer face of its plane dual G_e^* (light gray), and (b) a stereographic projection of G_e and G_e^* that preserves the face structure and places the $\emptyset$ vertex in G_e^*'s outer face; the antipodal point of the projection is the black-outlined circle in (a).

odd-even rule, v is interior to c_x. Similarly, for each item $x \notin v$, the number of times x can be added or removed along p must be even, which implies that v is exterior to c_x.

The end result is that vertex $v \in G_e$ and all points in its corresponding face $f^*(v) \in G_e^*$ are contained by region v of C, so C is an Euler diagram representing S. In this constructive proof, $G_e = G^*(C)$ and $G_e^* = G(C)$. Keeping in mind that G_e may be a stereographic projection, this means that C is an Euler diagram representing S whose Euler dual has the same combinatorial embedding as G_e; therefore, the theorem's conclusion is satisfied. □

The previous theorem tells us that if we are given a set system S and can find a connectivity graph G for S, then there exists an Euler diagram C representing

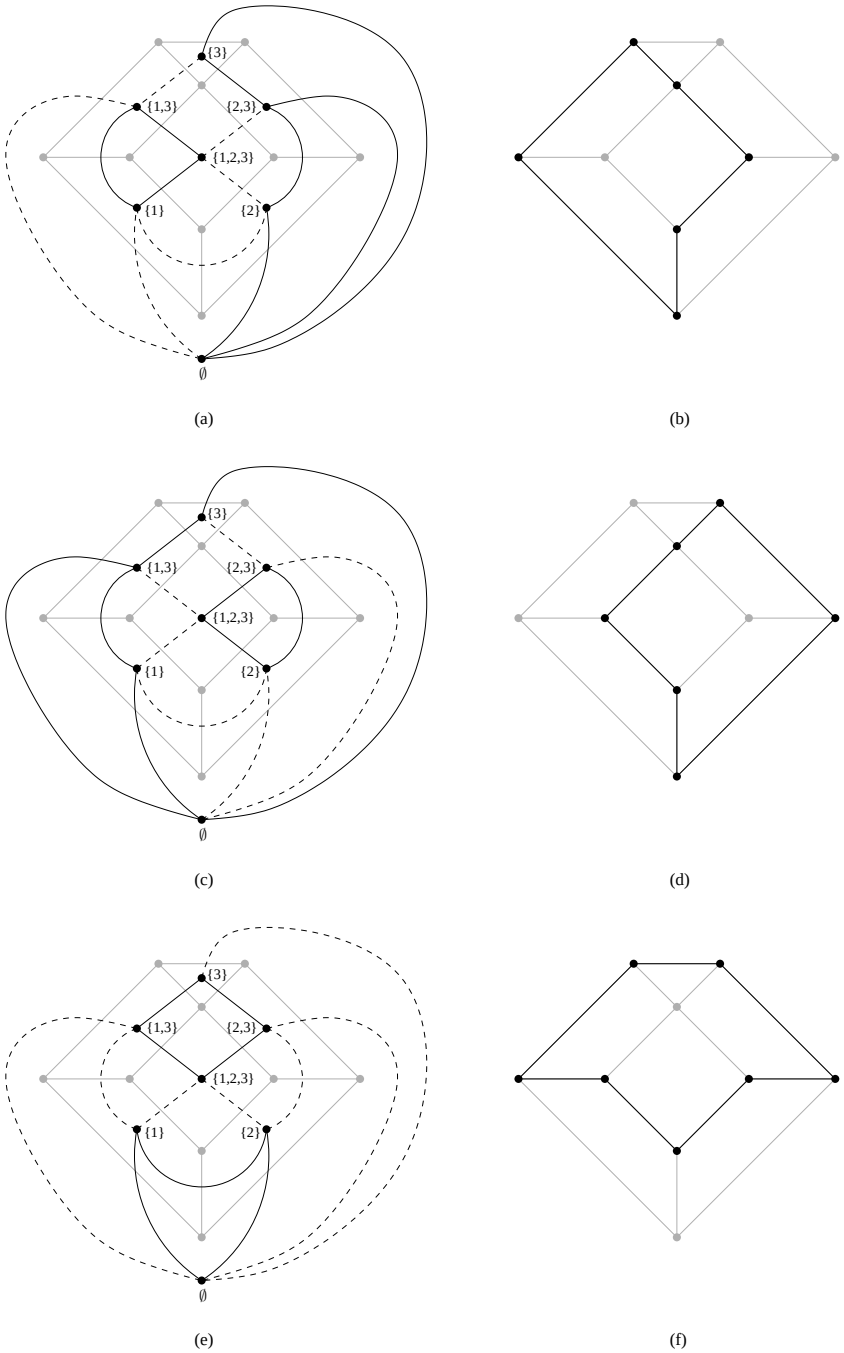


Figure 5.3: The minimal edge cuts separating G_x from $\overline{G_x}$ correspond to a cycle in G_e^*. The cut and corresponding cycle for $x = 1$, $x = 2$, and $x = 3$, are shown in (a,b), (c,d), and (e,f), respectively.

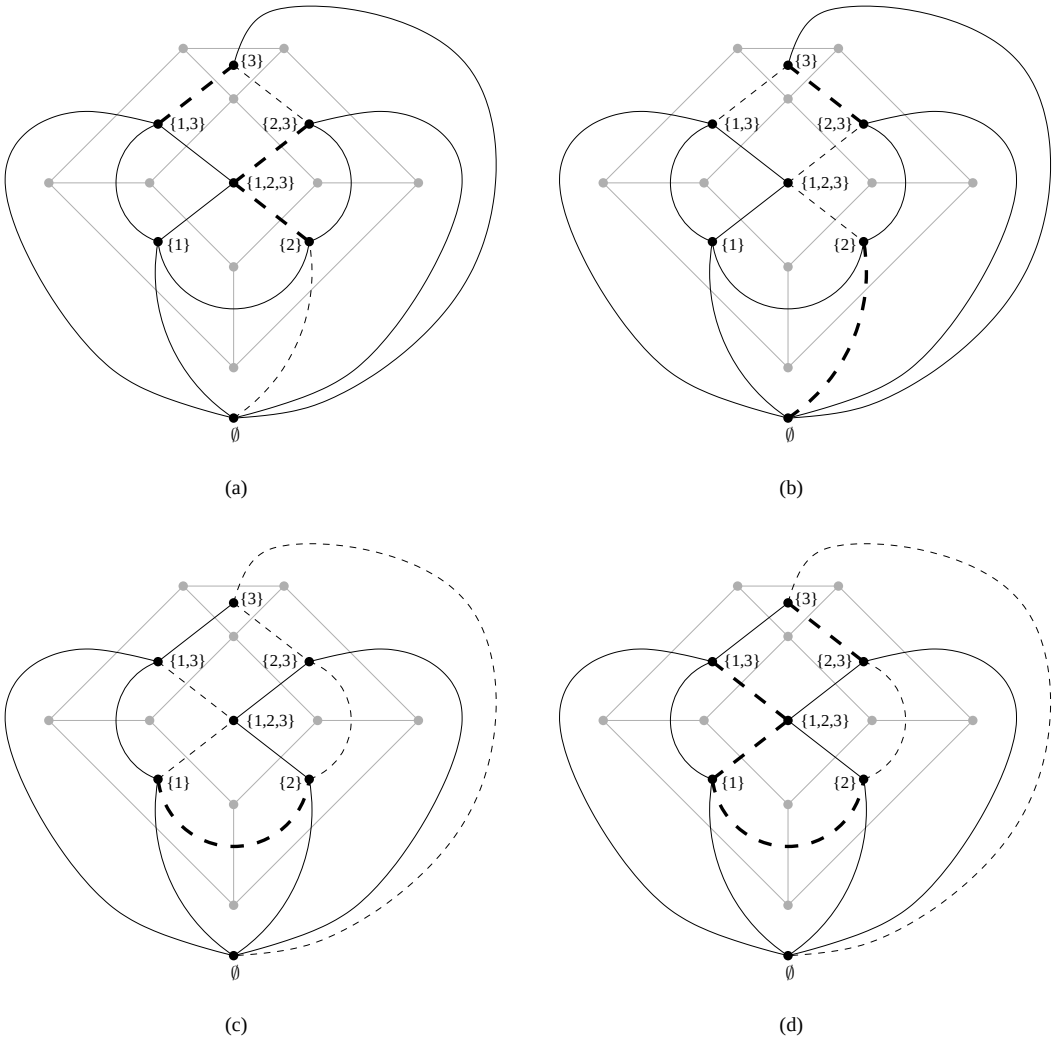


Figure 5.4: Any path (dashed) in G_e between v and $\emptyset$ must add/remove each item $x \in v$ an odd number of times and add/remove each item $x \notin v$ an even number of times. (a,b) and (c,d) show two different paths from $\{1, 3\}$ to $\emptyset$; in (a,c) the edges where item 1 is added/removed are highlighted, and in (b,d) the edges where item 2 is added/removed are highlighted.

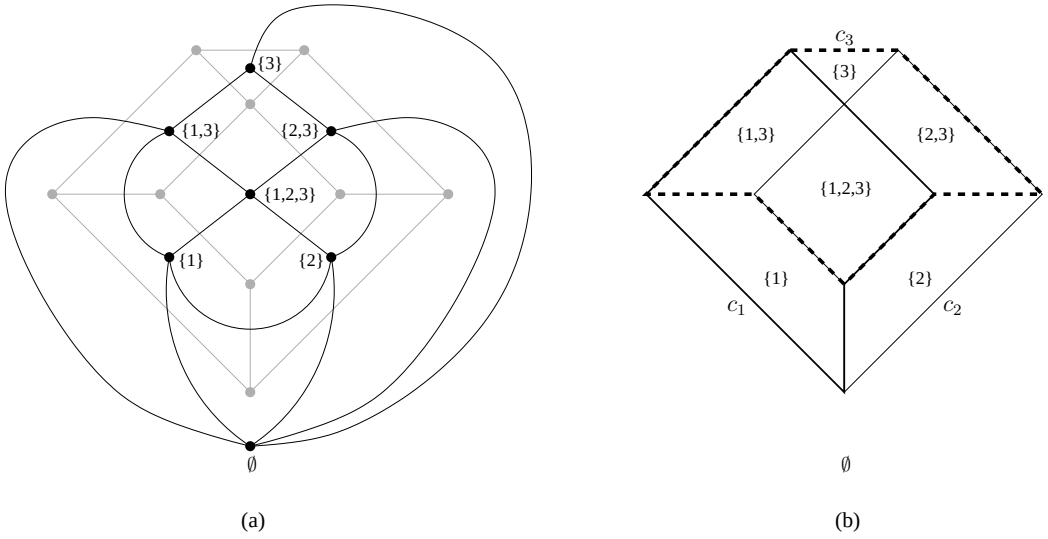


Figure 5.5: (a) A plane embedding G_e (black) of a connectivity graph for S where the $\emptyset$ vertex is in the outer face of G_e^* (light gray), and (b) the corresponding Euler diagram representing S.

S. Because the proof is constructive, in a sense, G becomes a solution to the Euler Diagram Generation Problem (EDGP).

In fact, because Thm. 5.0.4 ensures that any vertex, edge, or face properties encoded by the combinatorial embedding are shared by a plane embedding of G and $G^*(C)$, it goes beyond general Euler diagram existence and can apply to specific types of Euler diagrams (e.g., non-concurrent, pairwise, and simple). For example, if G only has edges between vertices that differ by a single element *and* has a plane embedding G_e whose faces are all degree 4, then by Thm. 5.0.4 and Prop. 4.4.3, there is a simple Euler diagram representing S.

In the following sections, we continue this exploration of the relationship between properties of a connectivity graph and properties of the resulting Euler diagram; the result is a set of theorems defining necessary-and-sufficient conditions for the existence of various types of Euler diagrams. An important aspect of the existence theorems

is that although they are based on Euler dual properties and, therefore, would seem to be limited to connected Euler diagrams, as mentioned in Section 4.2, a continuous transformation of the plane can be used to connect an Euler diagram. As a result, with due care to the implications of the disconnected to connected Euler diagram transformation, we are able to generalize the existence theorems to both connected *and* disconnected Euler diagrams.

5.1 General Euler Diagrams

Theorem 5.1.1. *Let S be a set system. There exists an Euler diagram representing S if and only if there is a* simple *connectivity graph for S.*

Proof. Sufficiency Let G be a simple connectivity graph for S. Since G is planar, it has a plane embedding G_e and by Thm. 5.0.4, there is a connected Euler diagram representing S.

Necessity Let C be an Euler diagram representing S. Without loss of generality, based on a continuous transformation of the plane, we can assume that C is *connected*. Since C is connected, it has an Euler dual $G^*(C)$. $G^*(C)$ meets all the conditions for being a connectivity graph for S; however, $G^*(C)$ may have parallel edges, which precludes it from being a *simple* connectivity graph. The being said, suppose we let $H^*(C)$ be the result of removing all but one of each set of parallel edges from $G^*(C)$. $H^*(C)$ still has the same connectivity as $G^*(C)$, and removing edges does not affect planarity, so $H^*(C)$ is a simple connectivity graph for S. □

Note that Thm. 5.1.1 applies to all Euler diagrams, regardless of whether or not they are connected. If one is trying to determine whether or not there is an

Euler diagram representing S, the need to only consider simple connectivity graphs is advantageous since it reduces the search space.

Example. Consider the set system

$$S = \{\emptyset, \{1,2,3,4\}, \{1,5,6,7\}, \{2,5,8,10\}, \{3,7,8,9\}, \{4,6,9,10\}\}$$

on the items $X = \{x | 1 \leq x \leq 10\}$.

Suppose we try to construct a connectivity graph G for S. For each item $x \in X$, there are *exactly* two sets $u, v \in S$ with $x \in u$ and $x \in v$; in other words, G_x contains exactly two vertices, and since G_x is connected, G must have an edge (u, v). As shown in Fig. 5.6(a), the edges that must be in G induce a K_5 subgraph, which precludes G from being a connectivity graph for S (since it is non-planar). So by Thm. 5.1.1, there is *no* Euler diagram representing S.

Now, suppose we remove all occurrences of item 10 from S in order to derive a new set system S'. In constructing a connectivity graph G' for S', the dashed edge in Fig. 5.6(a) is no longer needed so G' can be planar. Additional edges are also required to connect the $\emptyset$ vertex, but these do not affect the planarity of G' as shown in Fig. 5.6(b). As a result, G' is a connectivity graph for S', and the corresponding Euler diagram is shown in Fig. 5.6(c). □

5.2 Non-concurrent Euler Diagrams

The contrapositive implication of Prop. 4.4.1 along with Prop. 4.3.1 (see the proof of Prop. 4.4.3 for details), tells us that if C is a connected *non-concurrent* Euler diagram then each edge e of the Euler dual $G^*(C)$ has $|l(e)| = 1$. In other words,

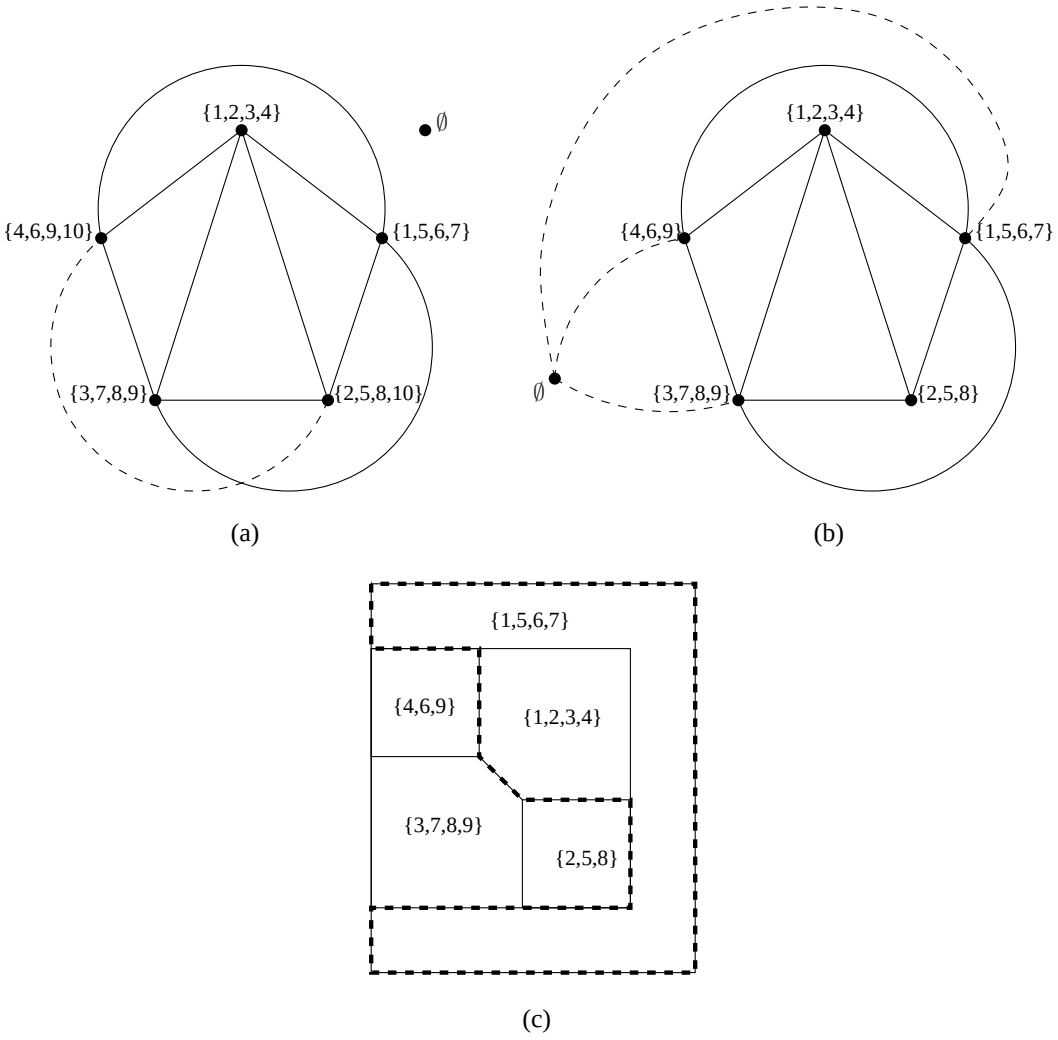


Figure 5.6: (a) A set system S that is *not* representable by an Euler diagram since G_x connectivity implies that *any* connectivity graph contains K_5 and is therefore non-planar; however, (b) by removing item 10 from S, a connectivity graph is no longer precluded from being planar, so one can be constructed resulting in (c) an Euler diagram representing the modified set system where c_1 is dashed.

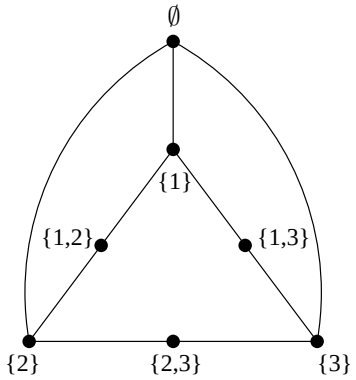


Figure 5.7: The closeness graph $G_c(S)$ for a set system S is a simple graph on S whose edges go between vertices differing by a single element.

since an Euler dual edge $e = (u, v)$ has $l(e) = u\Delta v$, $G^*(C)$'s edges can only connect vertices that differ by a single element. Since $G^*(C)$'s vertices are the elements of the set system S represented by C, we can determine directly from S the edges which *could* be in $G^*(C)$; the following definition describes a graph that represents these possible edges.

Definition 5.2.1. Let S be a set system. The *closeness graph* for S, denoted $G_c(S)$, is the simple graph on S with an edge $(u, v) \in G_c(S)$ if and only if $|u\Delta v| = 1$. □

Note that, depending on S, the closeness graph may be disconnected; in fact, it may have no edges at all.

Example. Consider the set system

$$S = \{\emptyset, \{1\}, \{1, 2\}, \{1, 3\}, \{2\}, \{2, 3\}, \{3\}\}.$$

Figure 5.7 shows the closeness graph $G_c(S)$. Note how each edge goes between vertices that differ by a single element. □

Although not used in this dissertation, the following proposition describes a potentially useful property of closeness graphs and is included for completeness.

Proposition 5.2.1. *Let S be a set system on the items X. The closeness graph $G_c(S)$ is isomorphic to a subgraph of the $|X|$-dimensional hypercube $Q_{|X|}$.*

Proof. Let $X = \{x_1, x_2, \ldots, x_n\}$, and therefore $|X| = n$. If we consider the vertices of Q_n as strings of n bits (n-bitstrings), then by definition, two vertices are adjacent in Q_n if and only if they differ by a single bit. Consider the injective function $\varphi : V(G_c(S)) \to V(Q_n)$ defined as

$$\varphi(s) = n\text{-bitstring } b \text{ where } b_i = 1 \text{ if and only if } x_i \in s.$$

φ maps $G_C(S)$ to a subset of Q_n. If $(u, v) \in G_C(S)$, then $(\varphi(u), \varphi(v)) \in Q_n$, (since $|u \Delta v| = 1$ implies that $\varphi(u)$ and $\varphi(v)$ differ by one bit); therefore, $G_C(S)$ is isomorphic via φ to a subgraph of Q_n. □

Given a set system S, we now know that if C is a connected non-concurrent Euler diagram representing S then its Euler dual $G^*(C)$ can only have edges from the closeness graph $G_c(S)$; however, since $G^*(C)$ may have parallel edges, it is not necessarily a subgraph of $G_c(S)$. As it turns out, we can ensure that $G^*(C)$ is a simple graph, and this notion is captured in the following theorem.

Theorem 5.2.2. *Let S be a set system. There exists a non-concurrent Euler diagram representing S if and only if there is a simple connectivity graph for S that is a spanning subgraph of $G_c(S)$.*

Proof. We first note that since $G_c(S)$ is a simple graph, any subgraph of $G_c(S)$ is also a simple graph. In addition, by its very nature, any connectivity graph for S is a spanning subgraph of $G_c(S)$.

Sufficiency Let $G \subseteq G_c(S)$ be a simple connectivity graph for S. Since G is planar, by Thm. 5.0.4, it is isomorphic to the Euler dual $G^*(C)$ of an Euler diagram C representing S.

Each edge of G is labeled based on its endpoints; therefore, $G^*(C)$ will have the same edge labels, and since $G \subseteq G_c(S)$, every edge $e \in G^*(C)$ has $|l(e)| = 1$. As a result, by the contrapositive of Prop. 4.4.1, C is a non-concurrent Euler diagram representing S.

Necessity Let C be a non-concurrent Euler diagram representing S. Without loss of generality, based on continuous transformation of the plane, we can assume that C is *connected*; the transformation only introduces point intersections, so C remains non-concurrent. As in the proof of Thm. 5.1.1, if we let $H^*(C)$ be the underlying simple graph represented by the Euler dual $G^*(C)$ then $H^*(C)$ is a simple connectivity graph for S. Finally, as we mentioned at the start of this section, since C is a connected non-concurrent Euler diagram, every edge $e \in G^*(C)$ has $|l(e)| = 1$, and so $H^*(C)$ is a (spanning) subgraph of $G_c(S)$. □

As with the existence of general Euler diagrams, Thm. 5.2.2 applies to both connected and disconnected non-concurrent Euler diagrams. When searching for a non-concurrent Euler diagram representing S, the need to only consider spanning subgraphs of $G_c(S)$ is particularly nice as it leads to a natural algorithm: enumerate the spanning subgraphs of $G_c(S)$ and apply the construction of Thm. 5.0.4 to any that are planar and have the G_x and $\overline{G_x}$ subgraph connectivity.

Example. Consider the set system

$$S = \{\emptyset, \{1\}, \{1,3\}, \{2\}, \{2,3\}\}.$$

Figure 5.8(a) shows the closeness graph $G_c(S)$. The subgraph of $G_c(S)$ induced by the vertices containing item 3 is *not* connected; therefore, any subgraph G of $G_c(S)$ will have its G_3 subgraph disconnected, thus precluding it from being a connectivity graph for S. As a result, by Thm. 5.2.2 there is *no* non-concurrent Euler diagram representing S. However, as shown in Fig. 5.8(b), there is a *concurrent* Euler diagram representing S.

On the other hand, consider the set system

$$S' = \{\emptyset, \{1\}, \{1,2\}, \{1,3\}, \{1,4\}, \{2\}, \{2,3\}, \{2,4\}, \{3\}, \{3,4\}, \{4\}\}.$$

Figure 5.8(c) shows the closeness graph $G_c(S')$. $G_c(S')$ does satisfy the G_x and $\overline{G_x}$ subgraph connectivity conditions for being a connectivity graph for S; however, it is non-planar since it is homeomorphic to K_5. If the dashed edge is removed, the subgraph connectivity constraints are still satisfied and the resulting graph *is* planar. As a result, by Thm. 5.2.2 there is a non-concurrent Euler diagram representing S' as shown in Fig. 5.8(d).

To verify that the Euler dual of the Euler diagram in Fig. 5.8(d) has the same combinatorial embedding as Fig. 5.8(c), just compare the adjacency of regions in the Euler diagram to the adjacency of vertices in the graph. As well, notice that the removal of the dashed edge implies that region $\{4\}$ is *not* adjacent to the outer region, which is evident in the resulting Euler diagram. □

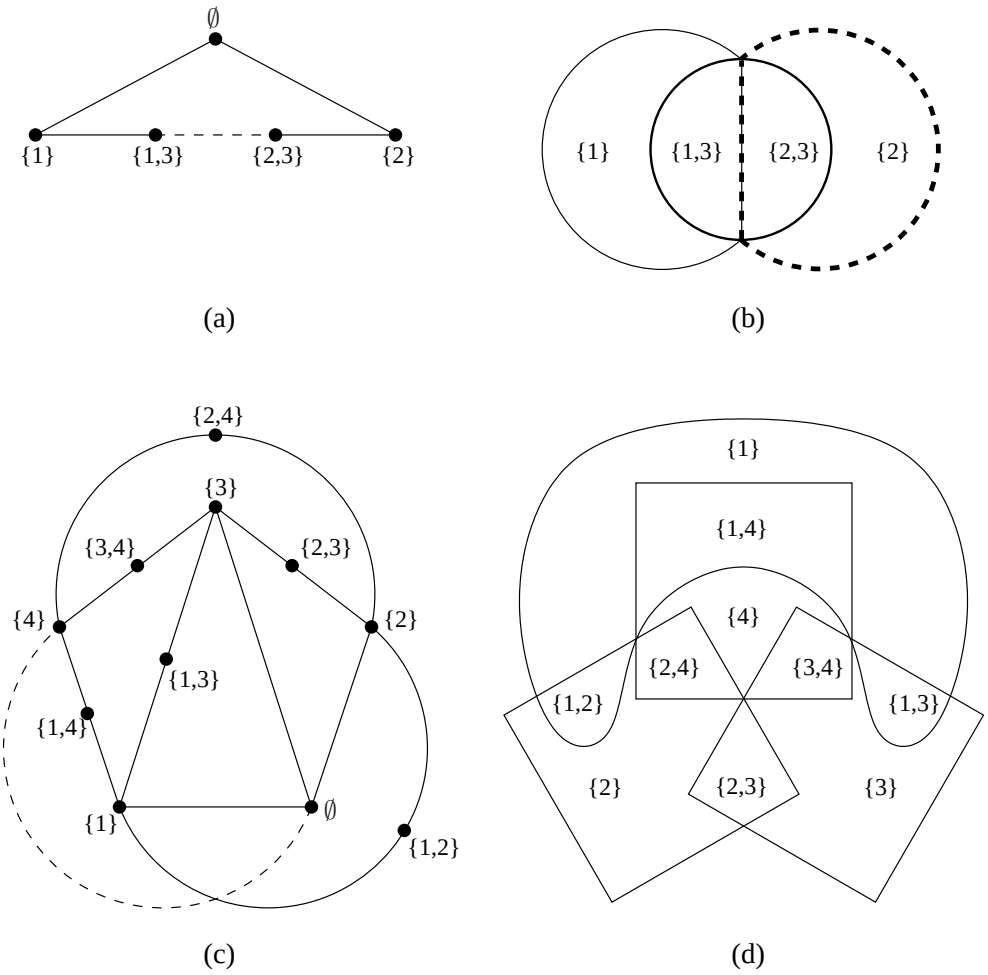


Figure 5.8: (a) A set system whose closeness graph (solid edges) does not have G_3 connectivity and is therefore *not* representable by a non-concurrent Euler diagram, (b) the concurrent Euler diagram whose Euler dual includes the dashed edge in (a), (c) a set system S whose closeness graph is not planar, but that has a spanning subgraph (solid edges) satisfying the conditions for being a connectivity graph for S, and (d) the resulting non-concurrent Euler diagram.

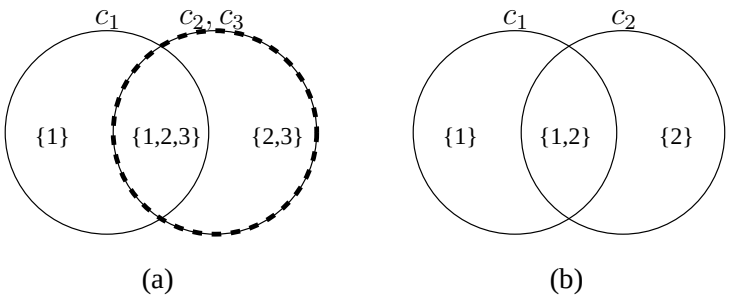


Figure 5.9: (a) An Euler diagram with equivalent curves (c_2 and c_3), is just a relabeling of (b) an Euler diagram with no equivalent curves.

5.3 Pairwise Euler Diagrams

Consider the Euler diagram C in Fig. 5.9(a), which has two equivalent curves c_2 and c_3. Because c_2 and c_3 are equivalent, the regions interior to c_2 are exactly the regions interior to c_3; in other words, with respect to the set system represented by C, items 2 and 3 always appear together. As a result, equivalent curves in an Euler diagram have an analog in terms of items within a set system; the following definition formalizes this observation.

Definition 5.3.1. Let S be a set system on the items X. Two items $x_i, x_j \in X$ are *equivalent* if for every set $s \in S$, $x_i \in s$ if and only if $x_j \in s$.

In other words, x_i and x_j are equivalent if they *always* appear together in the sets of S. □

Example. Consider the set system

$$S = \{\emptyset, \{1\}, \{1, 2, 3\}, \{2, 3\}, \{2, 3, 4\}\}.$$

Items 2 and 3 are equivalent since they always appear together in S (e.g., in

$\{1, 2, 3\}$, $\{2, 3\}$, and $\{2, 3, 4\}$). On the other hand, items 2 and 4 are not equivalent since, although 2 appears in every set containing 4 (e.g., $\{2, 3, 4\}$), 4 does not appear in every set containing 2 (e.g., $\{2, 3\}$). It is clear that in any Euler diagram representing S, curves c_2 and c_3 will be equivalent. □

Since the curves representing equivalent items are themselves equivalent, the presence of equivalent items in a set system does not fundamentally change the Euler Diagram Generation Problem (EDGP) since, as shown in Fig. 5.9, any Euler diagram with equivalent curves is just a relabeling of an Euler diagram without equivalent curves. In defining necessary-and-sufficient conditions for the existence of pairwise Euler diagrams, we consider only set systems without equivalent items; following the theorem, we will discuss the implications of this restriction.

Theorem 5.3.1. *Let S be a set system without equivalent items. There exists a pairwise Euler diagram representing S if and only if there is a simple Euler diagram representing S.*

Proof. Sufficiency Trivial since a simple Euler diagram is pairwise.
Necessity Let C be a pairwise Euler diagram. If C is non-concurrent then, by definition, it is simple and the theorem is satisfied, so suppose C is concurrent. Consider any concurrent curve segment c_s in C; because C is pairwise, c_s is part of *exactly* two curves c_i and c_j in C. If c_s is a closed curve segment (i.e., it has no endpoints), then this would imply that c_i and c_j are equivalent, which is impossible since S has no equivalent items. So c_s must have two endpoints that, by definition, occur where c_i and c_j diverge or where another curve c_k intersects c_s; however, the later is impossible since this would result in a non-pairwise intersection between c_i, c_j, and c_k. Because

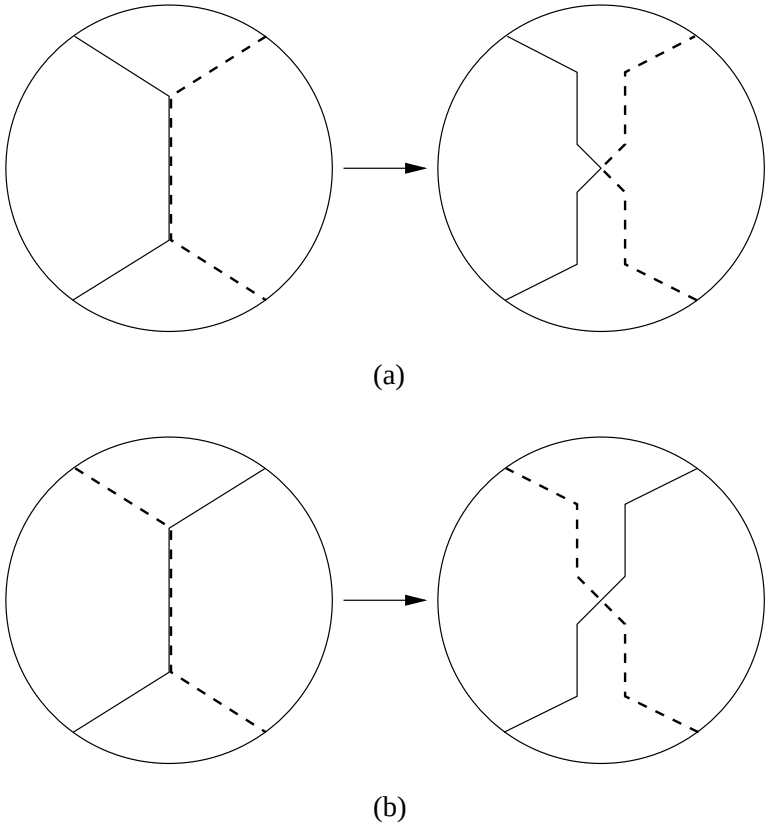


Figure 5.10: In a pairwise concurrent Euler diagram, each concurrent curve segment will be (a) tangential or (b) transverse; in either case, the transformation shown replaces the concurrent curve segment with a point intersection without adding or removing any regions.

the endpoints of c_s are only part of c_i and c_j, c_s can be "pulled apart" until all that remains is a pairwise point intersection as shown in Fig. 5.10. This transformation does not add or remove any regions, nor does it introduce any new intersections, so the result is an Euler diagram representing the same set system, but with one less concurrent curve segment. If this transformation is applied to all concurrent curve segments in C, the result is a non-concurrent pairwise (i.e., simple) Euler diagram representing S. □

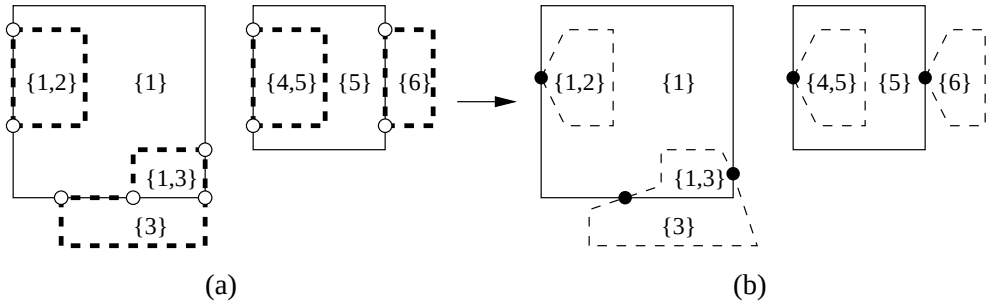


Figure 5.11: An example of how a concurrent pairwise Euler diagram can be transformed into a non-concurrent pairwise (i.e., simple) Euler diagram without altering the non-empty regions.

A theorem relating the existence of simple Euler diagrams to the existence of connectivity graphs is included in the next section; in the meantime, the following example demonstrates the concurrent to non-concurrent transformation in the proof of Thm. 5.3.1.

Example. Consider the set system

$$S = \{\emptyset, \{1\}, \{1,2\}, \{1,3\}, \{3\}, \{4,5\}, \{5\}, \{6\}\},$$

which has no equivalent items. Figure 5.11(a) shows a concurrent pairwise Euler diagram C representing S. Figure 5.11(b) shows the simple Euler diagram that results after applying the transformation in the proof of Thm. 5.3.1; note how the set system being represented remains unchanged. □

Why doesn't Thm. 5.3.1 extend to all set systems? As shown in Fig. 5.12, there are some set systems with equivalent items that are representable by pairwise Euler diagrams, but that are *not* representable by simple Euler diagrams. In addition, there are also set systems with equivalent items that are not representable by pairwise Euler

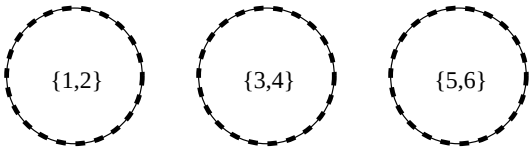


Figure 5.12: This set system, comprising disjoint pairs of items, can be represented by a disconnected pairwise Euler diagram; however, there is no connected pairwise or simple Euler diagram representing S.

diagrams (e.g., any set system with a group of three or more equivalent items). What about a set system S with only pairs of equivalent items? Suppose we transform S into a set system S' where each pair of equivalent items is represented by a single item in S'. Unfortunately, as shown in Fig. 5.9, the existence of a simple Euler diagram representing S' does not imply the existence of a pairwise Euler diagram representing S.

Clearly, there are more complicated necessary-and-sufficient conditions for pairwise Euler diagram existence when we consider set systems with equivalent curves. Since neither the presence nor absence of equivalent items and curves materially affects the remaining chapters' results, we move on and leave the following problem open (with the hint that ideas related to composite Euler diagrams in Chapter 8 can probably be applied to solve this problem relatively easily).

Open Problem 5.3.2. Let S be a set system with equivalent items. What are the necessary-and-sufficient conditions for S to be representable by a pairwise Euler diagram?

5.4 Simple Euler Diagrams

Since simple Euler diagrams are non-concurrent, they preclude the presence of equivalent curves; as a result, the concerns raised in the previous section do not apply, and the existence theorem for simple Euler diagrams applies to all set systems.

Theorem 5.4.1. *Let S be a set system on the items X. There exists a* simple *Euler diagram representing S if and only if $|X| = 1$ or there is a connectivity graph for S that is a spanning pseudo-subgraph of $G_c(S)$ with a plane embedding G_e such that each face $f \in G_e$ has $deg(f) = 4$.*

Proof. The case when $|X| = 1$ is trivial since it must be that $|C| = 1$ and $S = \{\emptyset, \{x_1\}\}$. In other words, for any Jordan curve c_1, $C = \{c_1\}$ is a simple Euler diagram representing S.

Sufficiency If $|X| > 1$, let G_e be as described in the theorem. By Thm. 5.0.4, there is a connected Euler diagram C representing S whose Euler dual $G^*(C)$ has the same combinatorial embedding as G_e. In other words, every face $f \in G^*(C)$ has $deg(f) = 4$ and, as in the proof of Thm. 5.2.2, since G_e's edges come from $G_c(S)$, each edge $e \in G^*(C)$ has $|l(e)| = 1$. As a result, by Prop. 4.4.3, C is a connected simple Euler diagram representing S.

Necessity Let C be a simple Euler diagram representing S with $|C| > 1$. Without loss of generality, based on continuous transformation of the plane, we can assume that C is *connected*; the transformation only introduces point intersections, and since C is non-concurrent, these intersections will be pairwise and C remains non-concurrent and pairwise (i.e., simple). Since C is connected, it has an Euler dual $G^*(C)$ that is a plane embedding of a graph meeting all the conditions for being a connectivity

graph for S. In addition, since C is simple, by Prop. 4.4.3, $|l(e)| = 1$ for every each $e \in G^*(C)$ and $deg(f) = 4$ for every face $f \in G^*(C)$. Finally, although $G^*(C)$ may contain parallel edges, since $|l(e)| = 1$ for every edge $e \in G^*(C)$, these edges are in $G_c(S)$, so $G^*(C)$ is a (spanning) pseudo-subgraph of $G_c(S)$. □

As with the existence of general Euler diagrams and non-concurrent Euler diagrams, Thm. 5.4.1 applies to both connected and disconnected simple Euler diagrams. When searching for a simple Euler diagram representing S, we need only consider edges in $G_c(S)$; however, unlike the case for non-concurrent Euler diagrams, since we are concerned with properties related to both subgraph connectivity *and* face structure, we cannot ignore parallel edges since their removal could alter face valency. As a result, the search space for connectivity graphs of simple Euler diagrams is not as restricted as the search space for connectivity graphs of non-concurrent Euler diagrams.

Example. Consider the set system

$$S = \{\emptyset, \{1\}, \{1,2\}, \{1,3\}, \{2\}, \{2,3\}, \{3\}\}.$$

Figure 5.13(a) shows the closeness graph $G_c(S)$; since $G_c(S)$ is homeomorphic to K_3, all its plane embeddings have the same structure as shown the figure. $G_c(S)$ has a degree 6 face, and it is evident that regardless of which parallel edges are added, the face's valency remains unchanged; therefore, any pseudo-subgraph of $G_c(S)$ is going to have a at least a degree 6 face. As a result, by Thm. 5.4.1, there is *no* simple Euler diagram representing S. However, as shown in Fig. 5.13(b), there is a non-simple Euler diagram representing S.

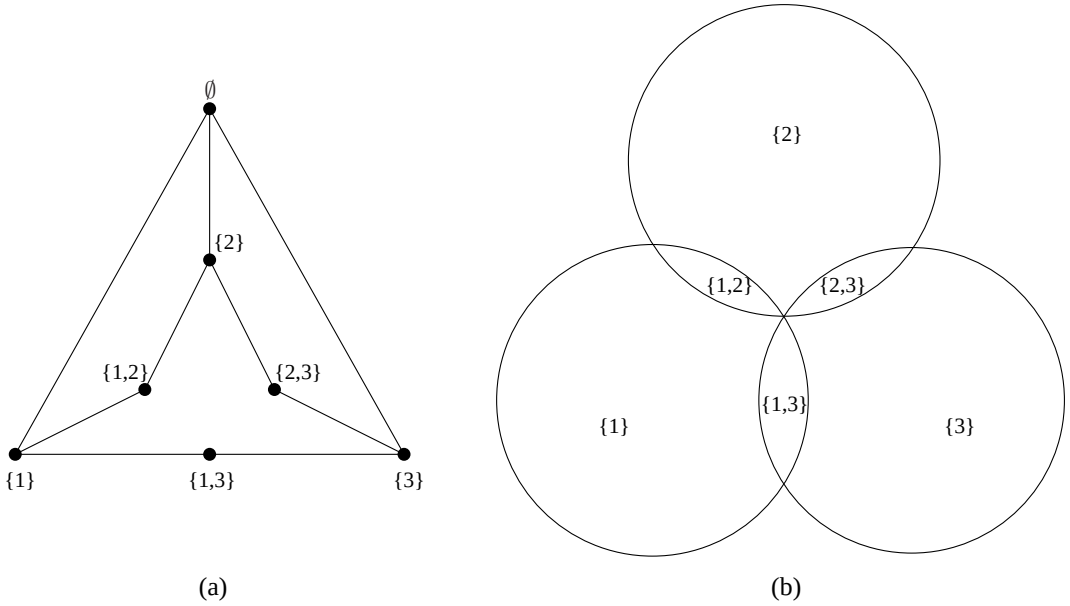


Figure 5.13: (a) The closeness graph for this set system has a degree 6 face whose size cannot be reduced by adding parallel edges, so (b) the resulting Euler diagram is non-simple.

Now, consider the set system

$$S' = \{\emptyset, \{1\}, \{1,2\}, \{1,2,3\}, \{1,3\}, \{3\}\}.$$

Figure 5.14(a) shows the closeness graph $G_c(S')$, which is a connectivity graph for S' with a degree 6 face (the outer face); therefore, its corresponding Euler diagram shown in Fig. 5.14(b) is *not* simple. However, by adding the parallel edge (dashed) shown in Fig. 5.14(c), the resulting connectivity graph for S' has all degree 4 faces. As a result, by Thm. 5.4.1, there is a simple Euler diagram representing S' as shown in Fig. 5.14(d). This is an example of why in Thm. 5.4.1 we cannot just consider subgraphs of $G_c(S)$. □

The existence theorems described in this chapter allow us to determine whether

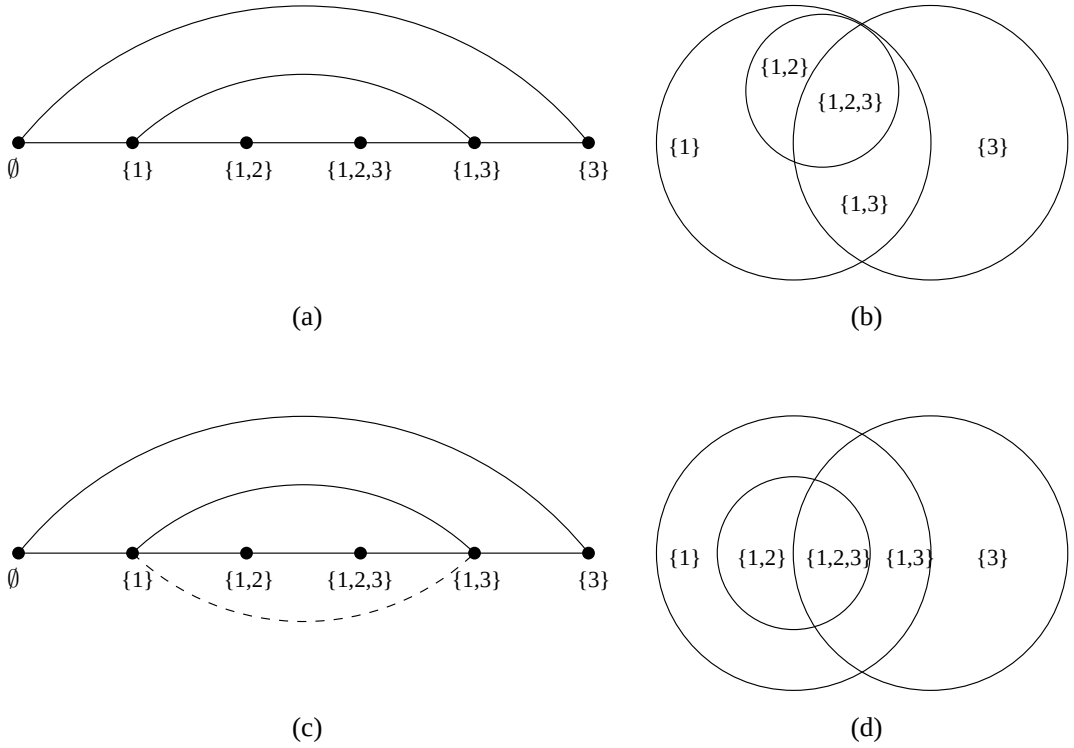


Figure 5.14: (a) The closeness graph for this set system S has a degree 6 face, so when interpreted as a connectivity graph, (b) the resulting Euler diagram is not simple; however, (c) by adding a parallel edge (dashed), the resulting graph is a connectivity graph for S with a plane embedding whose faces are all degree 4, so (d) the resulting Euler diagram is simple.

or not a particular set system is representable by a given type of Euler diagram. Because of their discrete nature, we are generally able to more easily reason about Euler diagram properties in terms of connectivity graphs than collections of Jordan curves. In the next chapter, we apply these theorems in order to study how particular Euler diagram properties affect the domain of representable set systems.

Chapter 6

Expressiveness of Diagram Types

For a given Euler diagram type (e.g., non-concurrent, pairwise, or simple), it is natural to ask which set systems are representable by Euler diagrams of the type; this *set* of set systems is referred to as the type's *representability class*. Representability classes may overlap; for example, the set system

$$S = \{\emptyset, \{1\}, \{1,2\}, \{2\}, \{2,3\}, \{3\}\}$$

can be represented by a simple Euler diagram as shown in Fig. 6.1(a) and also by a non-simple Euler diagram as shown in Fig. 6.1(b).

We say that one Euler diagram type is more (less) *expressive* than another if the former's representability class is a *proper* superset (subset) of the later's. Studying representability classes is particularly useful for understanding which restrictions (e.g., only pairwise intersections or no concurrent curves), affect expressiveness and which are merely aesthetic. In the following sections, we study the effect on expressiveness of several common Euler diagram restrictions.

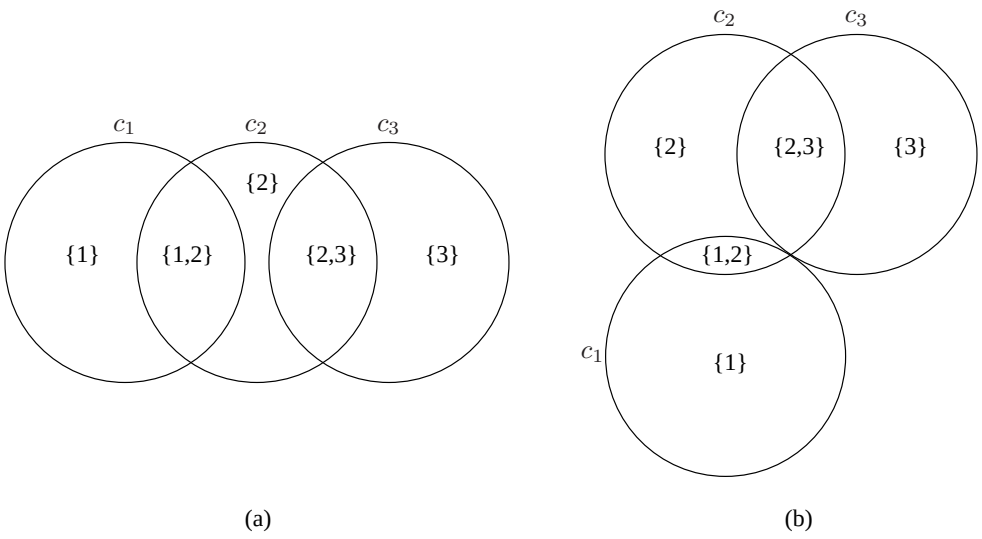


Figure 6.1: The set system $S = \{\emptyset, \{1\}, \{1,2\}, \{2\}, \{2,3\}, \{3\}\}$ can be represented by (a) a simple Euler diagram and (b) a non-simple Euler diagram.

6.1 Connected Euler Diagrams

For our first attempt at understanding Euler diagram expressiveness, we limit ourselves to the domain of connected Euler diagrams. Figure 6.2 is an Euler diagram for the universe of all set systems (the outer rectangle) where each circle represents a connected Euler diagram type (general, non-concurrent, and simple); pairwise Euler diagrams are omitted due to Thm. 5.3.1. The interior of each circle represents the associated Euler diagram type's representability class and, as is the case with Euler diagrams, the intersection of these interiors represents the intersection of the respective representability classes. The representability classes have a natural nested structure by virtue of the way the Euler diagram types are defined in Sec. 1.2. For example, since any simple Euler diagram is also, by definition, both a non-concurrent Euler diagram and a general Euler diagram, the region for set systems that are representable by simple Euler diagrams is enclosed by all three circles.

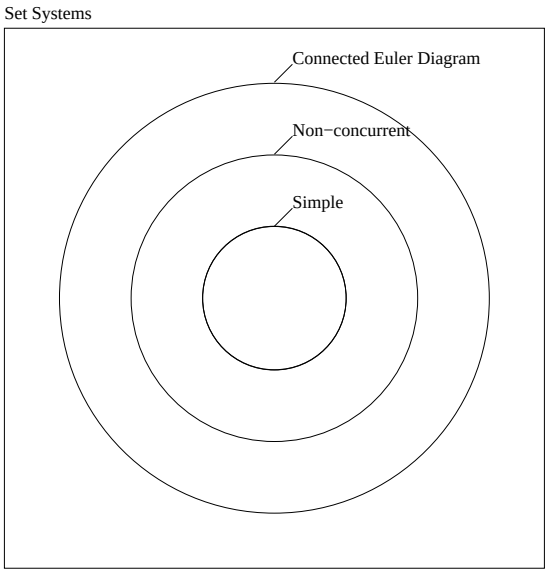


Figure 6.2: An Euler diagram representing the division of the universe of all set systems into the representability classes of various types of connected Euler diagrams. Note that since the "Non-concurrent" and "Simple" circles are nested in the "Connected" circle, they represent connected non-concurrent Euler diagrams and connected simple Euler diagrams, respectively.

In order to show that, for example, non-concurrent Euler diagrams are more expressive than simple Euler diagrams, it suffices to find a set system that is representable by a non-concurrent Euler diagram, but not by a simple Euler diagram. If we can prove that each region of Fig. 6.2 is non-empty (in terms of set systems), then Fig. 6.2 would be an accurate depiction of the connected Euler diagram types' expressiveness. In fact, this is the case as shown by the examples that followed each of Thms. 5.1.1, 5.2.2, and 5.4.1. Figure 6.3 is a version of Fig. 6.2(b) with annotations showing the Euler diagrams representing set systems in each of the regions (excluding, of course, a set system that *is not* representable by any connected Euler diagram). For example, the Euler diagram inside the "Non-concurrent" circle and outside the "Simple" circle is a connected non-concurrent Euler diagram representing a set system that *cannot* be represented by *any* connected simple Euler diagram.

6.2 Connected Euler-like Diagrams

As described in Chapter 2, a considerable amount of previous research has focused on Euler-like diagrams (see definition in Section 1.2.3). In this section, we study the differences in expressiveness between Euler and Euler-like diagrams, particularly as they pertain to common restrictions. Both Euler and Euler-like diagrams represent each item by a connected open plane subset; however, while in Euler diagrams the boundary of each item's plane subset (referred to as the "item's boundary" when the context is clear), is a Jordan curve, this may not be the case in Euler-like diagrams. Suppose we define Euler diagrams as a restricted version of Euler-like diagrams where each item's boundary is a Jordan curve. Figure 6.4(a) shows the classification of set

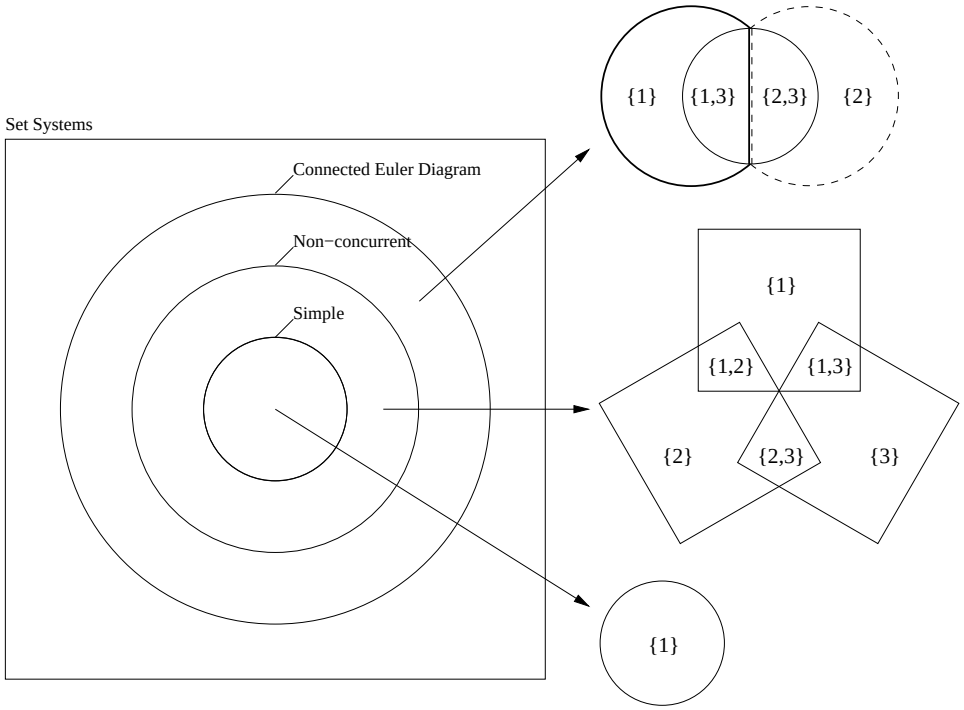


Figure 6.3: Examples of Euler diagrams representing set systems in each region; there is a set system within the region exterior to all circles that is not representable by any connected Euler diagram.

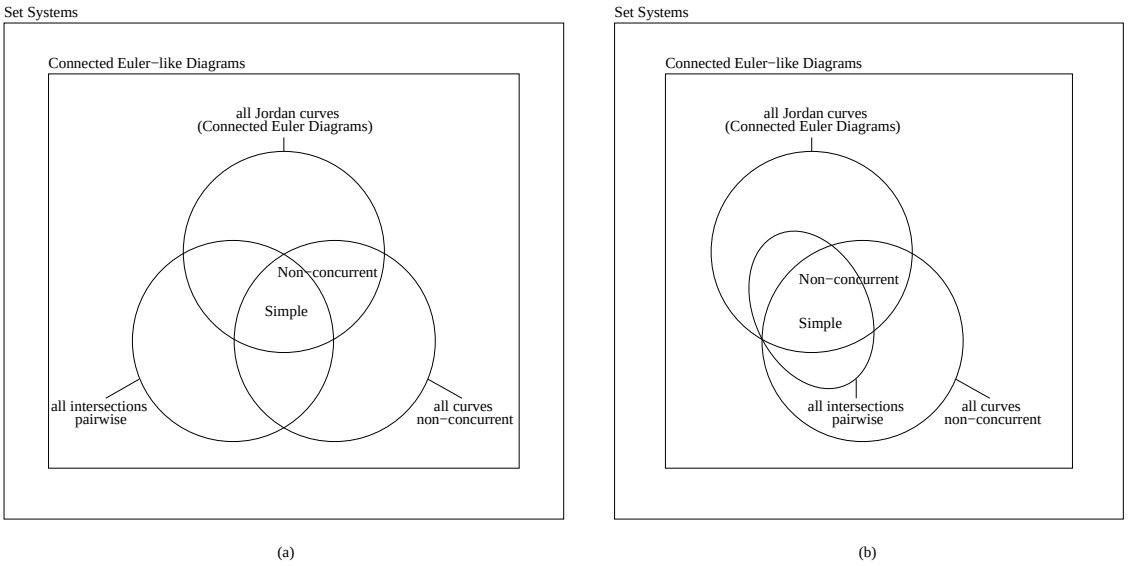


Figure 6.4: (a) The classification of set systems representable by connected Euler-like diagrams with various restrictions (the circle labels), and (b) a refined version where empty regions are removed.

systems according to whether or not they are representable by connected Euler-like diagrams with all possible combinations of the following restrictions:

- All Jordan curve item boundaries;
- All curves non-concurrent;
- All intersections pairwise.

Figure 6.4(a) differs from Fig. 6.2 in that each circle represents a restriction in the definition of an Euler diagram type rather than the type itself. The interior of each circle represents a *single* restriction, and the intersection of circle interiors represents the *conjunction* of the respective restrictions. For example, in Fig. 6.2, simple Euler diagram set systems are represented by the innermost circle, while in Fig. 6.4(a), simple Euler diagram set systems are represented by the *region* enclosed by the "Jordan", "pairwise", and "non-concurrent" circles. In this sense, the regions represent all possible types of connected Euler-like diagrams based on the three

aforementioned restrictions. Specifically, each region corresponds to the set systems that *can* be represented by connected Euler-like diagrams adhering to the containing circles' restrictions, but *cannot* be represented by *any* connected Euler-like diagram that is more restrictive. If a region is empty, then within the context of the existing restrictions, any additional restrictions are aesthetic and do not affect expressiveness. The following proposition proves that the region interior to the "pairwise" circle and exterior to the "Jordan" and "non-concurrent" circles is empty.

Proposition 6.2.1. *Any set system represented by a connected Euler-like diagram with all pairwise intersections can be represented by a connected Euler-like diagram with all pairwise intersections and either all Jordan curve item boundaries or all non-concurrent curves.*

Proof. Let S be a set system represented by a connected Euler-like diagram C with all pairwise intersections.

If C has all Jordan curve item boundaries or all non-concurrent curves then the proposition's conclusion is satisfied, so suppose C has at least one item whose boundary is *not* a Jordan curve and at least one concurrent curve segment c_s.

Since all of C's intersections are pairwise, c_s is part of *exactly* two curves.

Suppose c_s is a closed curve (i.e., it has no endpoints), as shown in Fig. 6.5(a). Since C is connected, any additional curves must intersect c_s, but this contradicts c_s having no endpoints. So C cannot have any additional curves and must be topologically equivalent to Fig. 6.5(a); however, such a diagram has only Jordan curve item boundaries, which contradicts C having at least one non-Jordan curve item boundary. As a result, c_s must have endpoints.

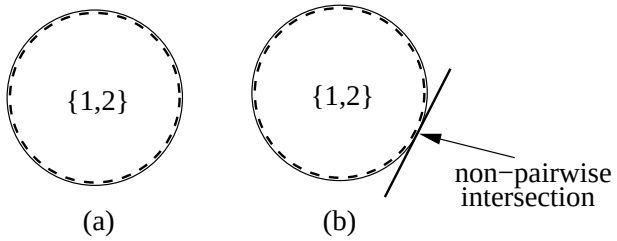


Figure 6.5: A concurrent curve segment as occurs in Prop. 6.2.1 must have at least two endpoints; otherwise, (a) the diagram is homeomorphic to one with only simply connected item subspaces, or (b) there is a non-pairwise intersection.

Suppose c_s only has one endpoint; this endpoint occurs when c_s intersects a third curve as shown in Fig. 6.5(b). Since c_s is a concurrent curve segment, this intersection will be non-pairwise and contradicts C having all pairwise intersections. As a result, c_s must have two endpoints.

Since C only has pairwise intersections, c_s cannot intersect any other curves, so the endpoints of c_s must coincide with where its two concurrent curves diverge. There are two possible configurations of c_s as shown in Fig. 6.6; in either case, the concurrent curve segment is removed while preserving the diagram's connectedness, without creating or destroying any regions, and without introducing any non-pairwise intersections. By repeating this operation, C is transformed into a connected Euler-like diagram representing the same set system, but with all pairwise intersections and all non-concurrent curves. □

Proposition 6.2.1 implies that there are *no* set systems representable by connected Euler-like diagrams with all pairwise intersections that cannot be represented by more restricted diagrams as shown in Fig. 6.4(b). The remaining regions in Fig. 6.4(b) are non-empty (in terms of set systems), as evidenced by the diagrams in Fig. 6.7; the proofs that each diagram represents a set system in its respective region are based

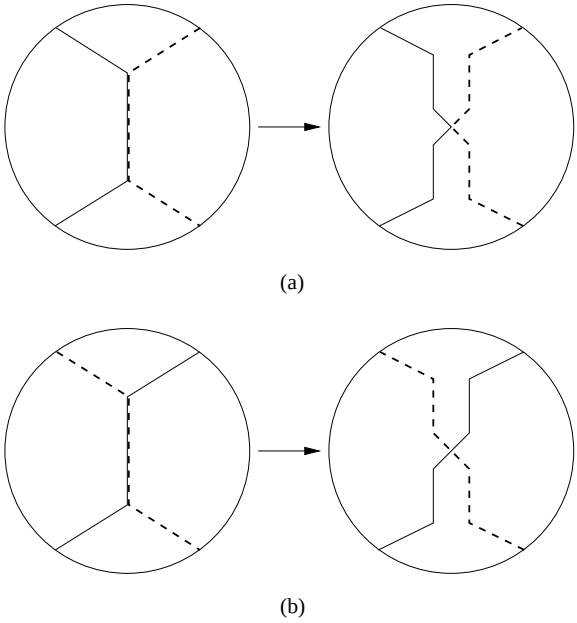


Figure 6.6: The two cases of concurrent curves with pairwise intersections being transformed into non-concurrent curves without creating or destroying regions.

on the necessary-and-sufficient conditions of the previous chapter and are included in Appendix A.6.2, pg. 274.

6.3 Disconnected Euler-like Diagrams

The previous two sections have explored the expressiveness of *connected* Euler and Euler-like diagrams; however, from previous examples (see Fig. 5.12), we know that the connected/disconnected property affects expressiveness. As we did with Euler and Euler-like diagrams in the previous section, we will introduce an explicit restriction for diagram connectedness; the resulting classification is shown in Fig. 6.8(a). The interpretation of Fig. 6.8(a) is the same as for Fig. 6.4(a); each region represents the set systems that can be represented by Euler-like diagrams adhering to the containing

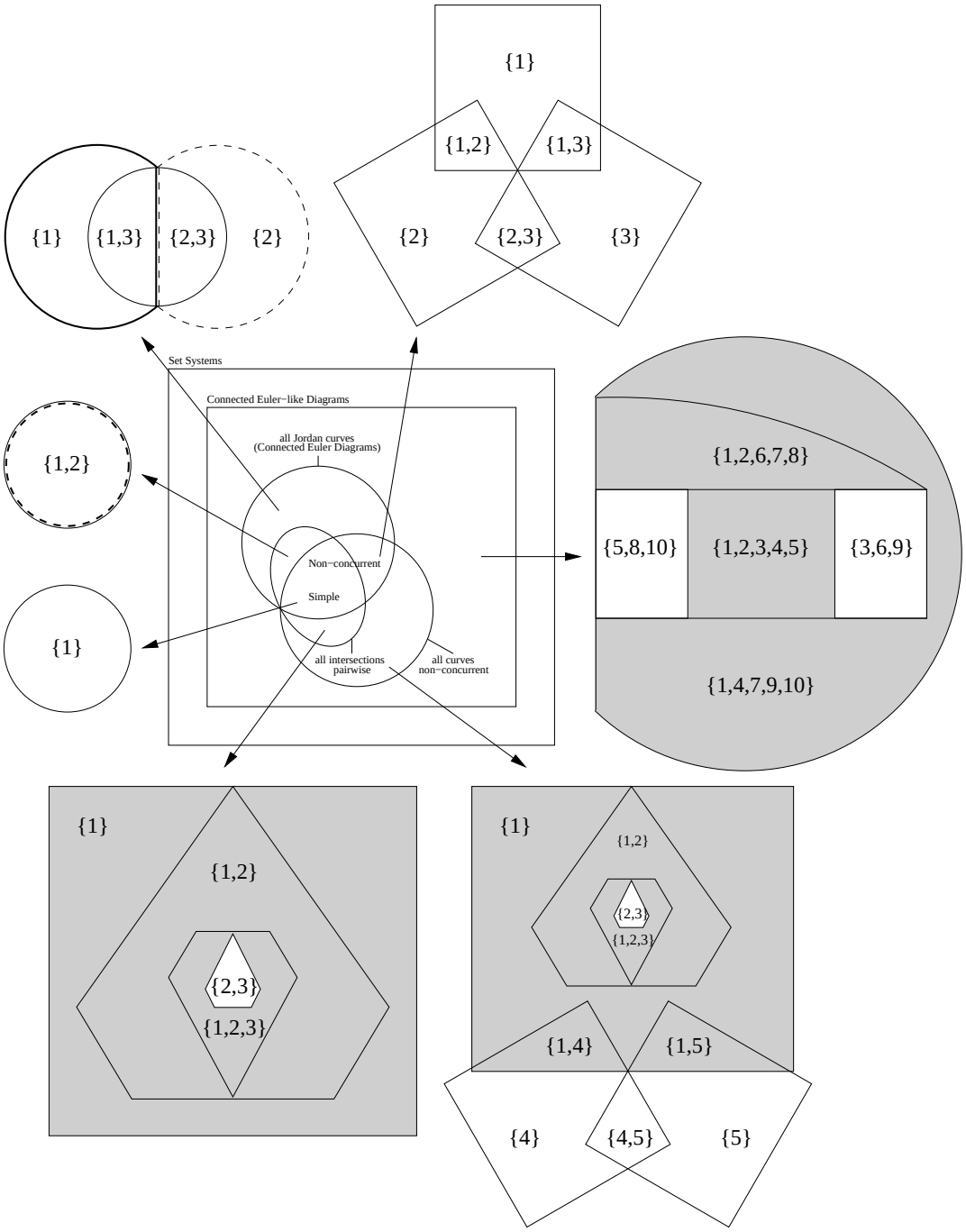


Figure 6.7: Examples of connected Euler-like diagrams representing set systems in their respective regions. When applicable, the interior of c_1 is shaded to indicate its non-Jordan curve boundary.

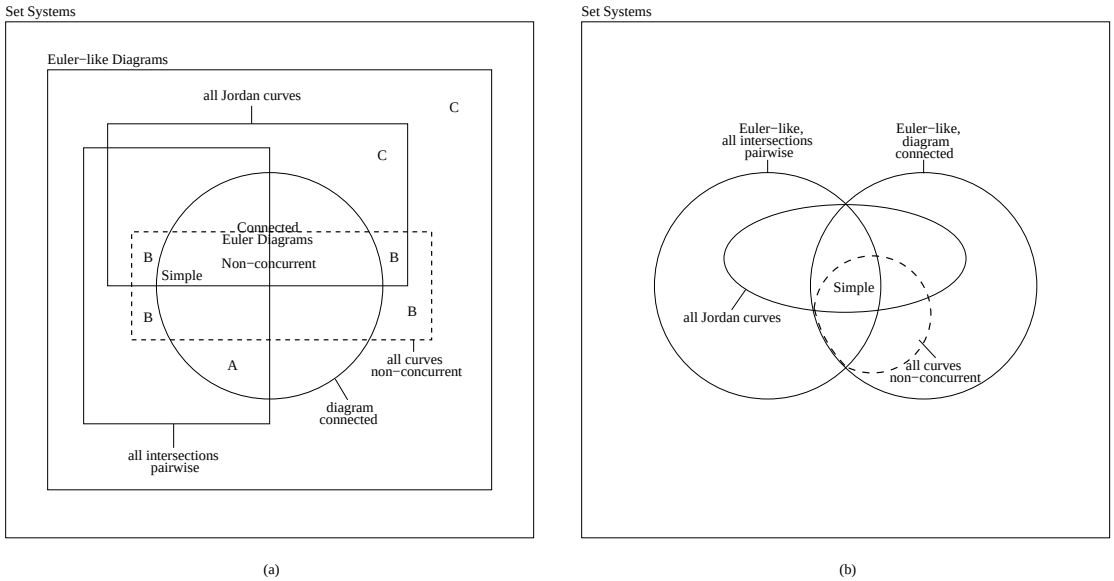


Figure 6.8: (a) The classification of set systems representable by Euler-like diagrams with various restrictions (the curve labels), and (b) a refined version where empty regions are removed.

curves' restrictions, but that *cannot* be represented by more restrictive Euler-like diagrams.

Proposition 6.2.1 can be applied to reason that the region labeled "A" in Fig. 6.8(a) is empty. The following propositions prove that there are additional regions in Fig. 6.8(a) that are also empty.

Proposition 6.3.1. *Any set system represented by a disconnected Euler-like diagram with all non-concurrent curves can be represented by a* connected *Euler-like diagram that adheres to the same restrictions (i.e., the resulting diagram does not violate any restrictions that are not* already *violated).*

Proof. Let C be a disconnected Euler-like diagram with all non-concurrent curves, and let δC be the union of C's item boundaries. Since C is disconnected, let $\delta C'$

be a maximally connected proper subset of δC. Figure 6.9(a) shows an example disconnected Euler-like diagram and subset $\delta C'$ (note that $\delta C'$ does not include the interiors of the three circles).

Since $\delta C'$ is disjoint from $\delta C \backslash \delta C'$, it must be completely contained within a connected plane subset (i.e., face) of $\delta C \backslash \delta C'$; therefore, as in the case of transforming general disconnected Euler diagrams to connected Euler diagrams, a series of continuous plane transformations can be applied within the face to make a new diagram where $\delta C'$ intersects $\delta C \backslash \delta C'$ at a point. Figure 6.9(b) shows an example transformation.

We note the following properties of the transformation:

1. the regions are preserved (i.e., the represented set system is unchanged),
2. each item's plane subset remains connected,
3. no curve self-intersections are created (i.e., Jordan curve item boundaries are preserved),
4. no concurrent curve segments are created, and
5. since C has all non-concurrent curves, the intersection between $\delta C'$ and $\delta C \backslash \delta C'$ is pairwise.

As a result, the transformed diagram does not violate any restrictions that were not already violated. By repeating this transformation until C becomes connected, the result is a *connected* Euler-like diagram that represents the same set system as C and adheres to any of C's restrictions. □

Proposition 6.3.2. *Any set system represented by a disconnected Euler-like diagram with at least one concurrent curve segment and at least one non-pairwise intersection can be represented by a* connected *Euler-like diagram that adheres to the same*

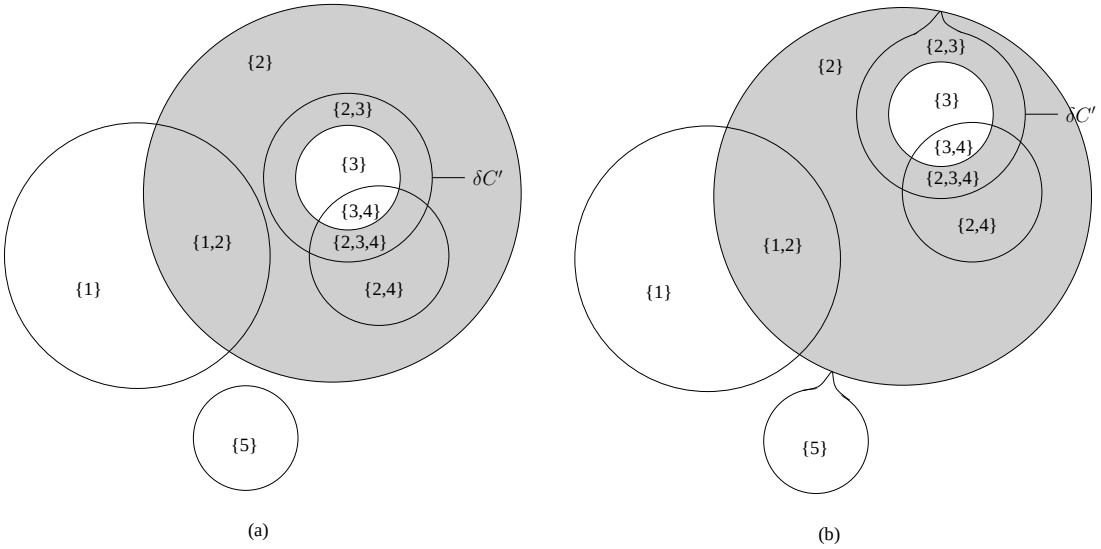


Figure 6.9: A continuous transformation as described in Prop. 6.3.1 from (a) a disconnected Euler-like diagram to (b) a connected Euler-like diagram adhering to the same restrictions.

restrictions (i.e., the resulting diagram does not violate any restrictions that are not already *violated).*

Proof. The proof is the same as the proof for Prop. 6.3.1 except that since C has a concurrent curve segment, we cannot guarantee that the intersection of $\delta C'$ and $\delta C \backslash \delta C'$ will be pairwise as shown in Fig. 6.10(b). However, since C already has a non-pairwise intersection, the addition of a non-pairwise intersection does not violate any new restrictions. □

Proposition 6.3.1 tells us that the regions of Fig. 6.8(a) that are outside the "connected" circle and inside the "non-concurrent" rectangle are empty since any set system in these regions can be represented by more restrictive Euler-like diagram types. Similarly, Prop. 6.3.1 tells us that the regions of Fig. 6.8(a) that are outside the "connected", "pairwise", and "non-concurrent" curves are empty. The regions whose

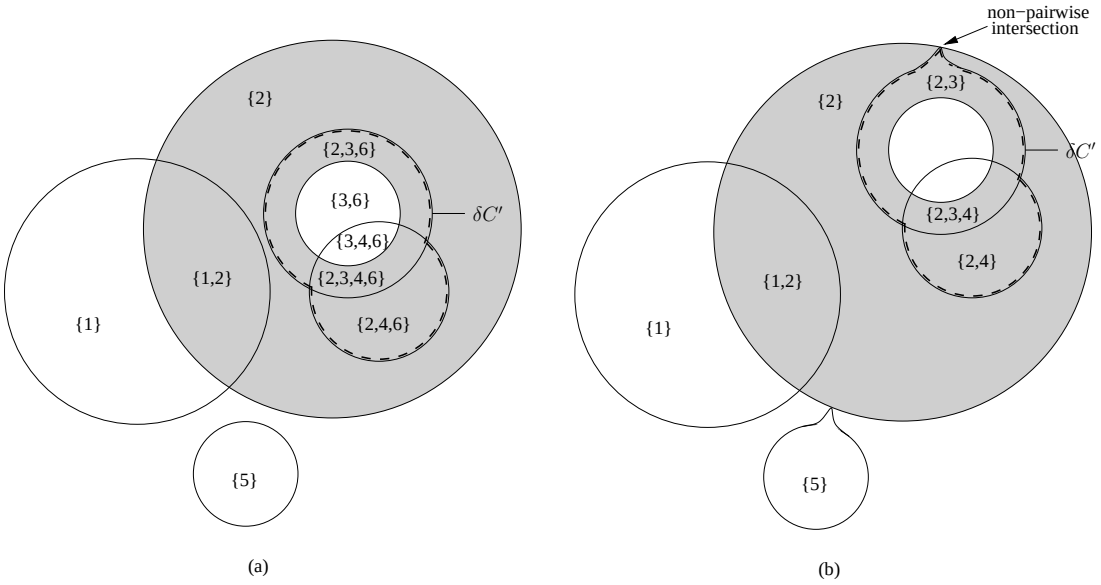


Figure 6.10: A continuous transformation as described in Prop. 6.3.2 from (a) a disconnected Euler-like diagram to (b) a connected Euler-like diagram adhering to the same restrictions.

emptiness is implied by Props. 6.3.1 and 6.3.2 are labeled "B" and "C", respectively. Notice how Props. 6.3.1 and 6.3.2 are a clarification of our assertion in Section 4.2 that any set system representable by a disconnected Euler diagram is representable by a connected Euler diagram.

Figure 6.8(b) shows the resulting classification of representability classes after removing the empty regions. The remaining regions are non-empty (in terms of set systems), as evidenced by the diagrams in Fig. 6.11; the proofs that each diagram represents a set system in its respective region are included in Appendix A.6.3, pg. 281.

Figure 6.11 is not an exhaustive analysis of all possible ways in which the restrictions can interact and affect expressiveness. For example, there may be trade-offs such as set systems representable by connected Euler-like diagrams with *either* all pair-

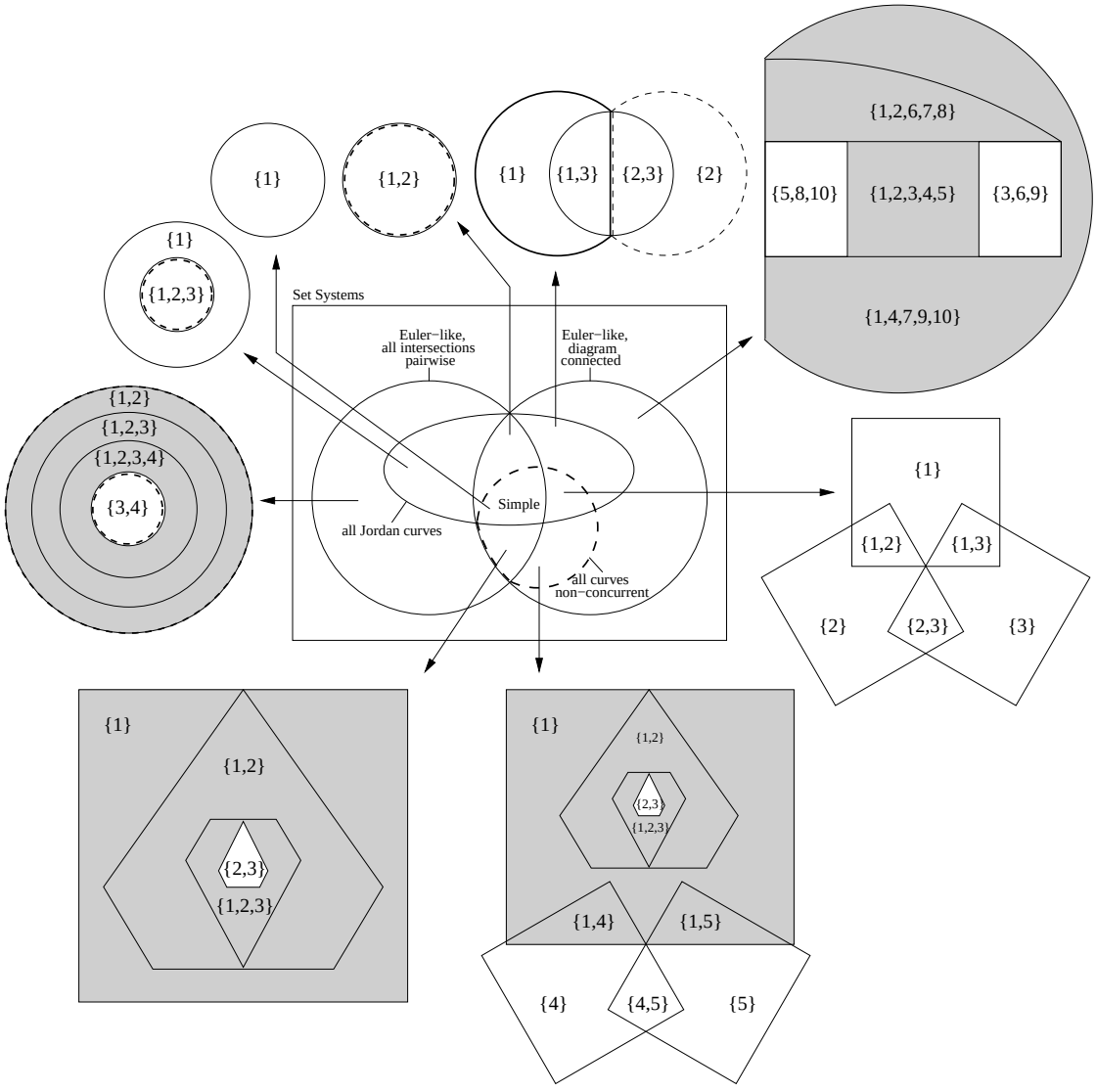


Figure 6.11: Examples of Euler-like diagrams representing set systems in their respective regions. When applicable, the interior of c_1 is shaded to indicate its non-Jordan curve boundary.

wise intersections *or* all non-concurrent curves, but not both. What we have done is consider how *increasing* restrictions affect expressiveness; this is a first step towards a better understanding of the role that restrictions play in Euler diagram generation, and demonstrates the usefulness of the necessary-and-sufficient conditions from Chapter 5. Finally, we note that several of the example diagrams in Fig. 6.11 contain equivalent curves/items. As we noted in Thm. 5.3.1, equivalent items have a meaningful effect on expressiveness; as such, we end this chapter with the following open problem.

Open Problem 6.3.3. Is it possible to find examples to demonstrate the separation of the representability classes in Fig. 6.11 that *do not* involve equivalent items, and how does a restriction of "no equivalent items" affect expressiveness?

Chapter 7

Computational Complexity of the Euler Diagram Generation Problem

In this chapter, we consider the computational complexity of the Euler Diagram Decision Problem (EDDP) and show that it is NP-complete (NPC). The EDDP is the decision version of the Euler Diagram Generation Problem (EDGP, see Def. 1.2.17); in other words, it is the question, "*Is* there an Euler diagram representing the set system S?". By Thm. 5.1.1, answering the EDDP is tantamount to determining whether or not there is a simple connectivity graph for S; as such, we formally define the EDDP as follows:

Definition 7.0.1. Euler Diagram Decision Problem (EDDP)

INPUT: A set system S.

OUTPUT: "Yes" if there is a simple connectivity graph for S; otherwise, "No".

□

As mentioned in Section 2.4, Johnson and Pollak [33] related the problem of planar representations of hypergraphs to the complexity of drawing Venn diagrams, which are similar to the Euler-like diagrams from Section 1.2.3. In particular, their paper introduced the following NPC problem:

Definition 7.0.2. Vertex-Based Venn Diagram Problem (VBVDP)

INPUT: A hypergraph $H = (V, E)$.

OUTPUT: "Yes" if there is a planar graph G on V for which each hyperedge $e \in E$ (where $e \subseteq V$), induces a connected subgraph of G; otherwise, "No".

□

A hypergraph for which there is an affirmative answer to the VBVDP is said to be *vertex-planar*. Figure 7.1(a) shows an example of a planar graph G for the vertex-planar hypergraph $H = (V, E)$ with

$$\begin{aligned} V &= \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11\} \text{ and} \\ E &= \{\{1, 2\}, \{1, 2, 3, 4\}, \{1, 3, 5\}, \{6, 7\}, \{8, 9, 10\}, \{8, 10, 11\}\}. \end{aligned}$$

Throughout the remainder of this section, we will use H as a running example.

The fact that the VBVDP involves vertex-induced connected subgraphs of a planar graph suggests that it may be related to the EDDP since connectivity graphs involve

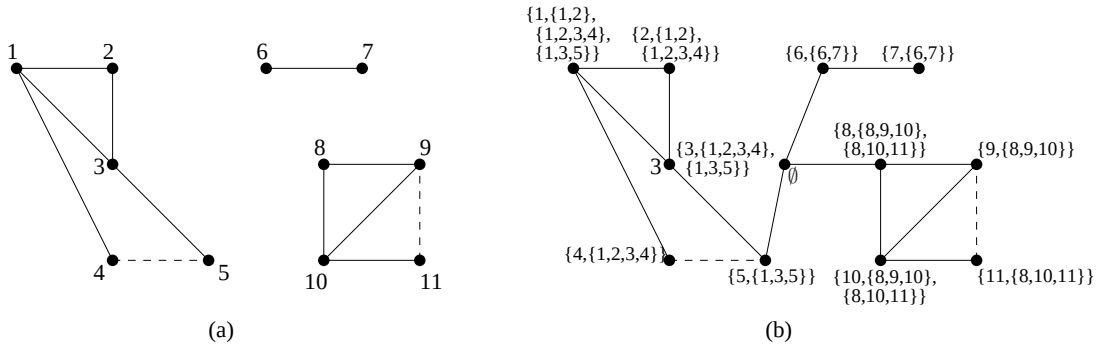


Figure 7.1: (a) The planar graph G^H that proves the hypergraph H with edge set $\{\{1,2\},\{1,2,3,4\},\{1,3,5\},\{6,7\},\{8,9,10\},\{8,10,11\}\}$ is vertex-planar, and (b) the corresponding partial connectivity graph G^S for the set system $S(H)$ (note how the $\emptyset$ vertex is used to make the graph connected). The dashed edges are unnecessary, but are included to show that the graph may have additional edges as long as it remains planar.

a similar subgraph connectivity; however, before we can study this relationship, we need to define a weaker version of the connectivity graph from Def. 5.0.1.

Definition 7.0.3. Let S be a set system on the items X. A graph G is said to be a *partial connectivity graph* for S if an only if the following properties hold for G:

1. G is a connected planar graph on S,
2. G is loop-free, and
3. for each item $x \in X$, the subgraph G_x induced by the vertices containing x is connected. □

The only difference between a partial connectivity graph and a connectivity graph is the lack of a requirement for $\overline{G_x}$. The next step in relating the VBVDP to the EDDP is to transform a hypergraph into a set system.

Definition 7.0.4. Let $H = (V, E)$ be a hypergraph. The set system $S(H)$ on the

items $V \cup E$ is defined as

$$S(H) = \{s(v)|v \in V\} \cup \{\emptyset\}$$

where $s(v) = \{v\} \cup \{e|v \in e\}$.

That is, for each hypergraph vertex $v \in V$, we define a set $s(v)$ containing v and any hyperedges that are incident to v. □

Example. Consider the hypergraph $H = (V, E)$ with

$$\begin{aligned} V &= \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11\} \text{ and} \\ E &= \{\{1, 2\}, \{1, 2, 3, 4\}, \{1, 3, 5\}, \{6, 7\}, \{8, 9, 10\}, \{8, 10, 11\}\}. \end{aligned}$$

The corresponding set system is defined as

$$\begin{aligned} S(H) \;=\; \{\;\; & \emptyset, \\ & \{1, \{1,2\}, \{1,2,3,4\}, \{1,3,5\}\}, \\ & \{2, \{1,2\}, \{1,2,3,4\}\}, \\ & \{3, \{1,2,3,4\}, \{1,3,5\}\}, \\ & \{4, \{1,2,3,4\}\}, \\ & \{5, \{1,3,5\}\}, \\ & \{6, \{6,7\}\}, \\ & \{7, \{6,7\}\}, \\ & \{8, \{8,9,10\}, \{8,10,11\}\}, \\ & \{9, \{8,9,10\}\} \\ & \{10, \{8,9,10\}, \{8,10,11\}\} \\ & \{11, \{8,10,11\}\}\}. \end{aligned}$$

Note how $S(H)$ is a set system on the items $V \cup E$ and how $\emptyset$ is explicitly added to $S(H)$. □

Through the above construction, the following proposition shows that vertex-planarity and partial connectivity graphs are interdependent.

Proposition 7.0.4. *A hypergraph H is vertex-planar if and only if there exists a partial connectivity graph for $S(H)$.*

Proof. Sufficiency Let $H = (V, E)$ be a hypergraph and let G^S be a partial connectivity graph for $S(H)$. Figure 7.1(b) shows a partial connectivity graph G^S for the set system $S(H)$ from the previous example.

For each hypergraph vertex $v \in V$, there is exactly one set $s(v) \in S(H)$ containing v. We define $G^H = (V, F)$ to be the planar graph derived from $G^S \backslash \{\emptyset\}$ by relabeling $s(v)$ to v. Figure 7.1(a) shows the planar graph G^H derived from the G^S in Fig. 7.1(b).

Each hyperedge $e \in E$ is an item of $S(H)$. By the construction of $S(H)$, a set $s(v) \in S(H)$ contains e if and only if $v \in e$. As a result, G^S_e is isomorphic to the subgraph of G^H induced by the vertex set $\{v | v \in e\}$ (i.e., the subgraph of G^H induced by e). Since G^S is a partial connectivity graph, G^S_e is connected; therefore, the subgraph of G^H induced by e is connected. So for each hyperedge $e \in E$, the subgraph of G^H induced by e is connected; therefore, G^H satisfies the VBVDP and H is vertex-planar.

Necessity Let $H = (V, E)$ be a vertex-planar hypergraph and let $G^H = (V, F)$ be a planar graph that satisfies the VBVDP for H. Figure 7.1(a) shows a planar graph G^H that corresponds to the hypergraph H from the previous example.

We define $G^S = (S(H), F')$ to be the *connected* planar graph derived from G^H by relabeling v to $s(v)$ and adding an $\emptyset$ vertex with an edge to each of G^H's connected components; this planar construction is always possible if we consider a star graph with $\emptyset$ at the center and G^H's connected components at the ray ends. Figure 7.1(b) shows the connected planar graph G^S derived from the G^H in Fig. 7.1(a).

For each hyperedge $e \in E$, the subgraph of G^H induced by e is isomorphic to the subgraph of G^S induced by the vertex set $\{s(v) | v \in e\}$, and since $e \in s(v)$ if and only if $v \in e$, this subgraph is G^S_e. Since G^H satisfies the VBVDP, the subgraph of G^H induced by e is connected; therefore, G^S_e is connected. Each vertex $v \in V$ is also an item of $S(H)$, and since G^S_v is the singleton vertex $s(v)$, G^S_v is trivially connected.

So G^S is a connected planar graph with $V(G^S) = S(H)$ such that for each item x of $S(H)$, G^S_x is connected; therefore, G^S is a partial connectivity graph for $S(H)$. □

If we weakened the EDDP so that it only required the existence of a partial connectivity graph (call this the *weak-EDDP*), then Def. 7.0.4 and Prop. 7.0.4 would provide a polynomial-time reduction from the VBVDP to the weak-EDDP that could be used to prove that the weak-EDDP is NPC.

Unfortunately, a similar reduction from the VBVDP to the EDDP is elusive due to the EDDP's requirement of $\overline{G_x}$ connectivity; however, in the proof that VBVDP is NPC, Johnson and Pollak described a reduction from Hamilton Path on cubic 3-connected planar graphs (HP-C3P, a known NPC problem [19, GT39]), which can be modified and reframed within the context of set systems to prove that the EDDP is NPC. We will now provide an overview of this proof and leave the details for Appendix A.7, pg. 284.

The first step in proving that the EDDP is NPC is to show that it is in NP. For a given set system S, we generate a random graph G on S and verify that G is planar and has the necessary subgraph connectivity; if this "guess and check" strategy can be done in polynomial time then the EDDP is NP.

Lemma 7.0.5. *The Euler Diagram Decision Problem (EDDP) is in NP.*

Proof. See Appendix A.7, pg. 284. □

We must now define a polynomial-time transformation from instances of HP-C3P to instances of the EDDP such that the EDDP answers affirmative to the transformed instance if and only if HP-C3P answers affirmative to the original instance. The next section describes this transformation.

7.1 Transformation from Hamilton Path to EDDP

For clarity, the transformation from a cubic 3-connected planar graph G to a set system $S(G'_p)$ is divided into the following steps:

1. Define a plane embedding G_p of G;
2. Define a plane graph G'_p that augments G_p with additional vertices and plane edges;
3. Define a set system $S(G'_p)$ by assigning sets to the vertices of G'_p.

The following definitions describe each of the above steps.

Definition 7.1.1. Let G be a cubic 3-connected planar graph and let $G_p = (V, E)$ be a plane embedding of G. The plane graph G'_p is derived from G_p with the following changes:

1. For each face $f \in F(G_p)$, add a vertex v_f to G'_p that is located in f and connected with a bisected edge to each vertex in the boundary of f;
2. For each edge $e \in E$, add two vertices to G'_p, v_{e_1} and v_{e_2}, that trisect e;
3. For each vertex $v \in V$, add a vertex v' to G'_p that is located in the vicinity of v (i.e., locate v' so that a plane edge *could* be drawn between v and v'). □

Example. Figure 7.2 shows an example of how a cubic 3-connected plane graph G_p is transformed to G'_p. The v_f vertices that are added in Step 1 are shown in white outline, and the edges that connect v_f to the boundary vertices of f are shown as dashed edges. The vertices that bisect and trisect edges are shown in gray. In G'_p, each of G_p's cubic vertices has degree 6 and is associated with three bisecting vertices, three trisecting vertices, and a singleton vertex. □

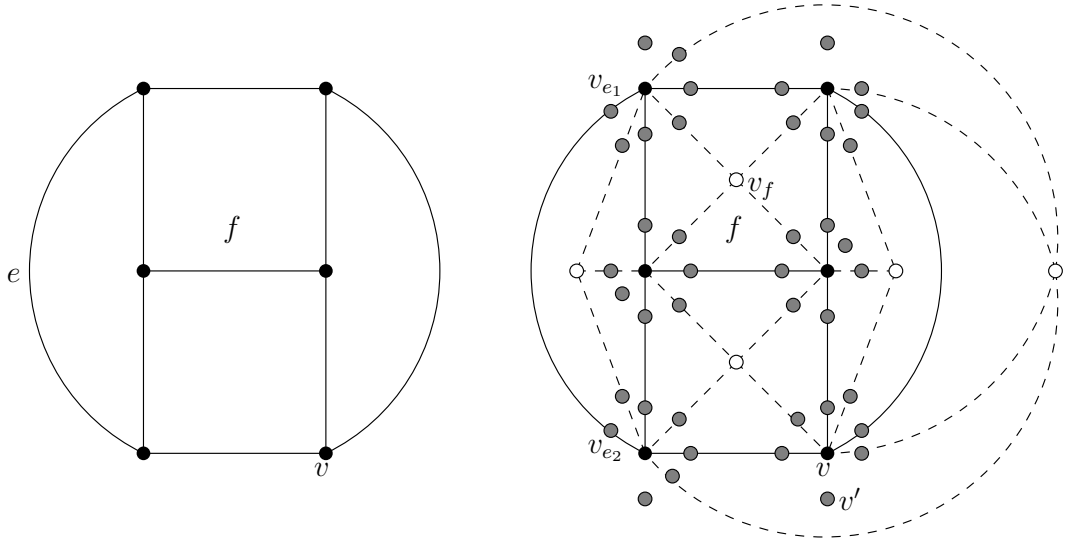


Figure 7.2: (a) A cubic 3-connected plane graph G_p and (b) its transformation to G'_p via Def. 7.1.1.

The following proposition relates to the structure of G'_p and although not needed in the set system construction, plays an important role in the proof of the EDDP's NP-completeness.

Proposition 7.1.1. *Let G_p be a cubic 3-connected plane graph, and let G'_p be the derived graph as defined by Def. 7.1.1. The plane graph $G'_p \backslash \{v' | v \in V(G)\}$ (where the v' vertices are as defined by Step 3 of Def. 7.1.1), is homeomorphic to a 3-connected graph.*

Proof. $G'_p \backslash \{v' | v \in V(G)\}$ is homeomorphic to the plane graph derived by Step 1 of Def. 7.1.1 where v_f is connected to f's boundary vertices by an edge that is *not* bisected; consider the construction of this graph face-by-face.

Since G is cubic, each of its faces must have at least three edges, and therefore three vertices, in its boundary, so $deg(v_f) \geq 3$. It is well-known that if a vertex v

is added to a k-connected graph H with at least k edges connecting v to H, then $H \cup \{v\}$ is k-connected. As a result, each time a vertex is added to one of G's faces, the resulting graph remains 3-connected. □

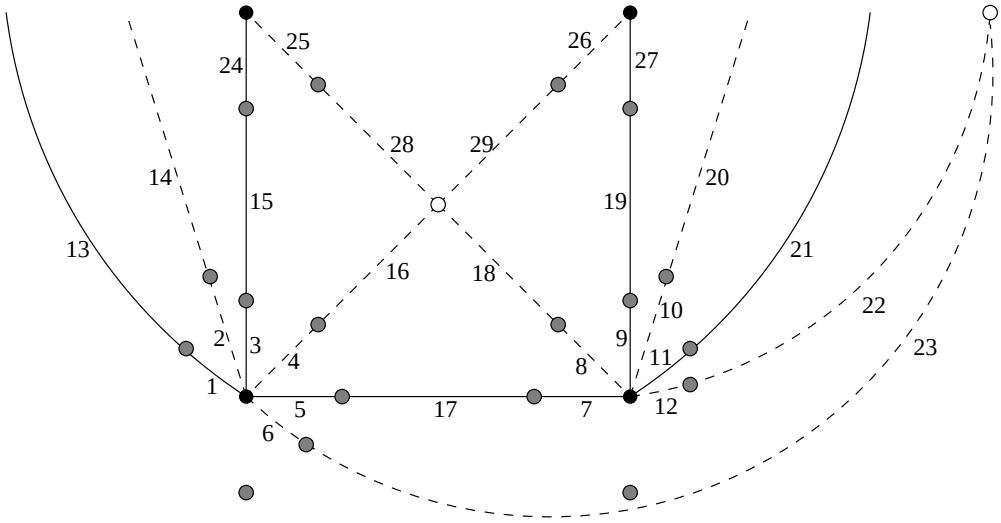
Now that we have defined G'_p, we can use it as the basis for constructing a set system.

Definition 7.1.2. Let G_p be a cubic 3-connected plane graph, and let G'_p be the derived graph as defined by Def. 7.1.1. The set system $S(G'_p) = \{s(v)|v \in V(G'_p)\} \cup \{\emptyset\}$, where $s(v)$ is initially empty, is defined by the following rules (where *unique integer* refers to an integer that has not been previously used):

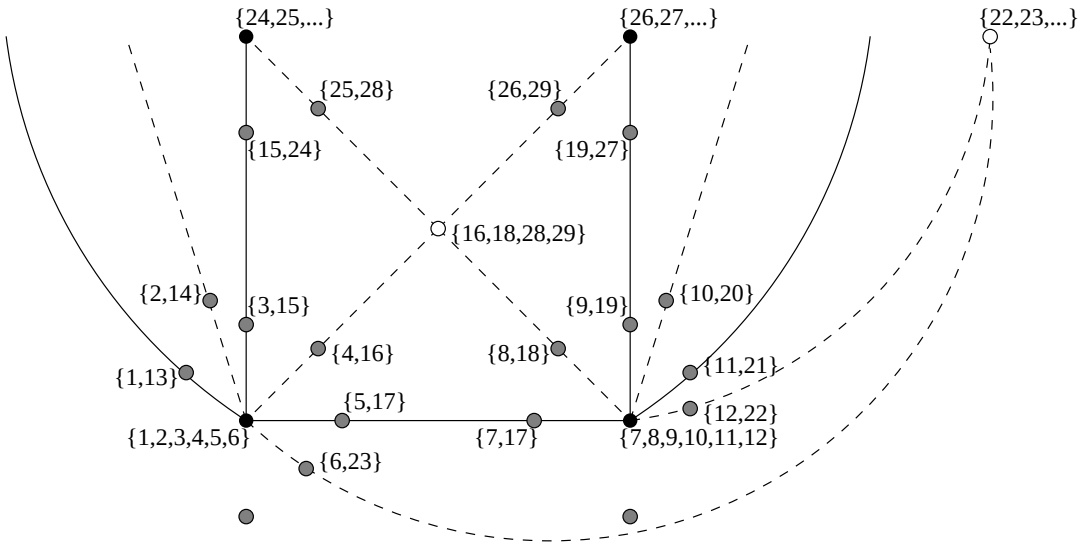
1. For each edge $(u, v) \in E(G'_p)$, choose a unique integer and add it to the sets $s(u)$ and $s(v)$; this integer is called a *pairwise* item.
2. For each vertex $v \in V(G_p)$, choose a unique integer and add it to the sets $s(v')$ (where v' is as defined in Step 3 of Def. 7.1.1), and $s(u)$ over all vertices $u \in V(G'_p)$ that are adjacent to v; this integer is called a *surrounding* item.
3. Choose a unique integer and add it to the sets $s(v)$ and $s(v')$ over all vertices $v \in V(G_p)$; this integer is called the *Hamilton* item. □

Example. The following example is based on G'_p from Fig. 7.2(b). Figure 7.3 shows the effect of Step 1 of Def. 7.1.2 on a portion of G'_p. In Fig. 7.3(a), each edge is labeled with the unique integer that was chosen for it, and in Fig. 7.3(b), each vertex v is labeled with $s(v)$ based on the labels of its incident edges.

Figure 7.4(a) shows the effect of Step 2 of Def. 7.1.2 on G'_p, assuming that we continue labeling the 81 edges incrementally as was done in Fig. 7.3(a). To reduce clutter, only the vertices that are adjacent to one vertex $v \in V(G_p)$ are labeled; the other vertices in G_p are annotated with the unique integer that would be assigned to their adjacent vertices.



(a)



(b)

Figure 7.3: The effect of Step 1 of Def. 7.1.2 on a portion of G'_p from Fig. 7.2(b) showing (a) the unique integers chosen for each edge and (b) the resulting sets.

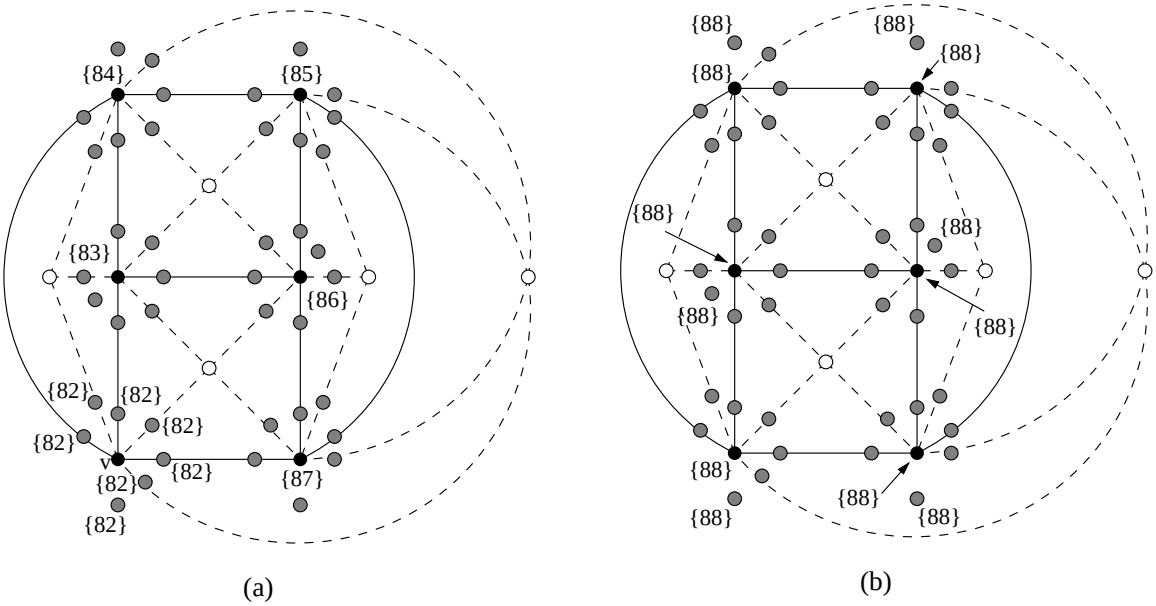


Figure 7.4: The effects of (a) Step 2, and (b) Step 3 of Def. 7.1.2 on G'_p from Fig. 7.2(b).

Finally, Fig. 7.4(b) shows the effect of Step 3 of Def. 7.1.2 on G'_p; G_p's vertices and the v' vertices from Step 3 of Def. 7.1.1 have a unique integer added to their sets.

The final set $s(v)$ represented by a vertex $v \in G'_p$ would be derived by the union of v's labels in Figs. 7.3(b) and 7.4(a)(b). For example, the vertex v in Fig. 7.4(a) would have $s(v) = \{1, 2, 3, 4, 5, 6, 88\}$ and $s(v') = \{82, 88\}$, and its adjacent vertex u at the 10 o'clock position would have $s(u) = \{1, 13, 82\}$. □

Since every non-v' vertex in G'_p has degree ≥ 2, Step 1 of Def. 7.1.2 ensures that their assigned sets are non-empty and unique. For v' vertices that have degree 0, Step 2 of Def. 7.1.2 ensures that their assigned sets are also non-empty and unique. As a result, $|S(G'_p)| = |V(G'_p)| + 1$ (the +1 accounts for the addition of the $\emptyset$ vertex).

The construction of $S(G'_p)$ relies only on a combinatorial embedding of G (since only the facial structure of G is needed to generate G'_p and the resulting sets), and

such an embedding is the usual byproduct of linear-time planarity checking [30]. In addition, since G is planar, the number of vertices, the number of edges, and the number of faces are linearly related by Euler's Formula ($V - E + F = 2$), so by choosing a data structure that efficiently supports vertex and face traversal (e.g., the Winged-Edge data structure [2]), $S(G'_p)$ can be constructed in polynomial-time with respect to G's size.

Definitions 7.1.1 and 7.1.2 define a polynomial-time transformation from instances of HP-C3P to instances of the EDDP. It now remains to show that if G is a cubic 3-connected planar graph, then G has a Hamilton path if and only if there is a connectivity graph for $S(G'_p)$.

7.2 Hamilton Path → Connectivity Graph

Let G be a cubic 3-connected plane graph with Hamilton path

$$hp = v_1, v_2, \ldots, v_{|V(G)|}$$

and Hamilton path edges

$$e_i = (v_i, v_{i+1}), 1 \leq i < |V(G_p)|.$$

Figure 7.5(a) shows an example of a planar embedding G_p of G with an annotated Hamilton path.

In Def. 7.1.2, $S(G'_p)$ is constructed by assigning unique non-empty sets to G'_p's vertices; let $G^S = (S(G'_p)\backslash\{\emptyset\}, E)$ be this vertex-labeled plane graph. Figure 7.5(b)

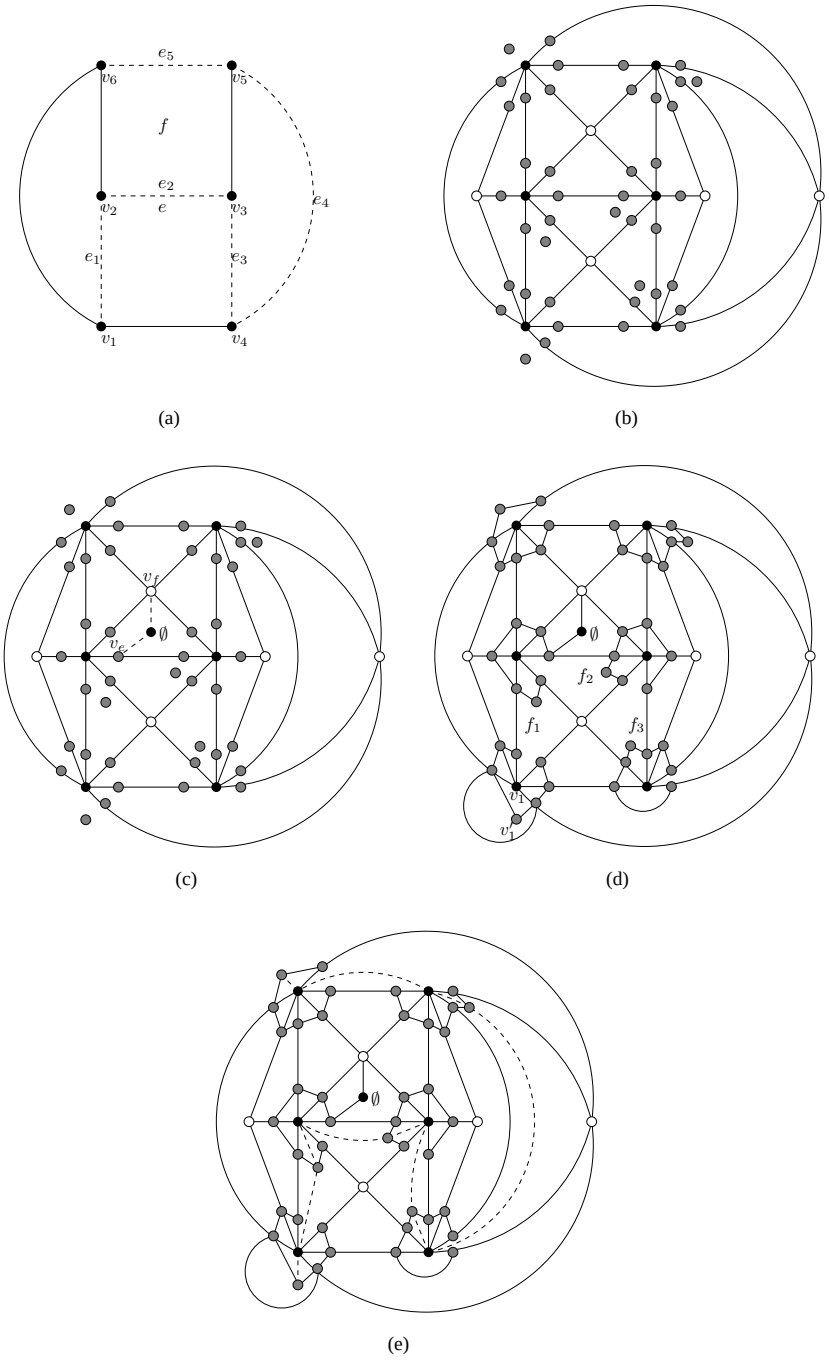


Figure 7.5: The steps, (b)–(d), for transforming (a) a cubic 3-connected plane graph G_p with Hamilton path (dashed edges) into (e) a connectivity graph for the set system $S(G'_p)$.

shows an unlabeled G^S for the example G_p from Fig. 7.5(a).

In order to make G^S a connectivity graph for $S(G'_p)$, we need to add an $\emptyset$ vertex while still maintaining planarity. Let f be any face of G_p and v_f be its corresponding vertex in G^S as defined by Step 1 of Def. 7.1.1. In addition, let e be an edge of G_p in the boundary of f, and v_e be one of the trisecting vertices in G^S as defined by Step 2 of Def. 7.1.1. Figure 7.5(a) shows an arbitrarily chosen face f and edge e, and Fig. 7.5(c) shows how an $\emptyset$ vertex and plane edges $(\emptyset, v_f), (\emptyset, v_e)$ can be added to G^S.

At this point, G^S is a connected plane graph with $V(G^S) = S(G'_p)$, but it does not have the necessary subgraph connectivity. In Step 2 of Def. 7.1.2, for each vertex $v \in G_p$, a unique *surrounding* item x_v is added to v's adjacent vertices in G^S, as well as its corresponding v' vertex (we call these v's *surrounding vertices*). In addition, Step 3 of Def. 7.1.2 adds a *Hamilton* item x_{hp} to each vertex $v \in G_p$ and its corresponding v' vertex. In order for $G^S_{x_v}$ to be connected, we need to add edges between v's surrounding vertices in such a way that plane edges can also be added to connect $G^S_{x_{hp}}$. The following construction achieves this goal.

For each Hamilton path edge e_i, let f_i be the face to the right of the edge of G^S that is homeomorphic to e_i. Step 1 of Def. 7.1.1 ensures that $f_i = f_j$ if and only if $i = j$. Figure 7.5(d) is annotated with the first three f_i faces of the Hamilton path in Fig. 7.5(a).

For the first Hamilton path vertex v_1, connect the surrounding vertices with a chain of six plane edges such that no edge is in f_1 and v'_1 is in any face about v_1 *other than* f_1. In addition, connect the two surrounding vertices that are adjacent to v'_1 with a plane edge that *does not* separate v_1 and v'_1.

For the last Hamilton path vertex $v_{|V(G_p)|}$, connect the surrounding vertices with

a chain of six plane edges such that no edge is in $f_{|V(G_p)|-1}$ and $v'_{|V(G_p)|}$ is in any face about $v_{|V(G_p)|}$ *other than* $f_{|V(G_p)|-1}$.

For each intermediate Hamilton path vertex, $v_i, 2 \leq i < |V(G_p)|$, connect the surrounding vertices with a chain of six plane edges such that no edge is in f_i and v'_i is in f_{i-1}.

Figure 7.5(d) shows the plane edges that are added to connect each set of surrounding vertices; take special note of the additional edge added to v_1's surrounding vertices.

Lastly, for $1 \leq i \leq |V(G_p)|$, connect v_i to v'_i with a plane edge, and for $1 \leq i < |V(G_p)|$, connect v_i to v'_{i+1} with a plane edge.

Figure 7.5(e) shows the final version of G^S with the Hamilton path edges (dashed) added.

It is clear that the additional edges can be added while maintaining the planarity of G^S, and by virtue of the construction, G^S_x is connected for each item x added during the construction of $S(G'_p)$ in Def. 7.1.2; what remains to prove is that $\overline{G^S_x}$ is also connected.

The details of the proof that $\overline{G^S_x}$ is connected for each item x are included in Appendix A.7.2, pg. 285, but it is fairly easy to get a sense of the proof by considering a few examples. We treat the three types of items that are in $S(G'_p)$ individually.

For a pairwise item x, $\overline{G^S_x}$ corresponds to removing the only two vertices that contain x; by the construction of $S(G'_p)$, these vertices are adjacent in G'_p. For the connectivity graph in Fig. 7.5(e), $\overline{G^S_x}$ is akin to removing two vertices that are adjacent in Fig. 7.5(b). The fact that, ignoring v' vertices, G'_p is homeomorphic to a 3-connected graph (Prop. 7.1.1) and that the $\emptyset$ vertex and the v' vertices have two

incident edges whose opposing endpoints are not adjacent ensures that $\overline{G_x^S}$ remains connected.

For a surrounding item x_v, $\overline{G_{x_v}^S}$ results from removing all of v's surrounding vertices. Figure 7.6(a) shows an example of removing v_2's surrounding vertices. Again, the 3-connectivity implied by Prop. 7.1.1 and the fact that v remains connected by the Hamilton path edges ensures that $\overline{G_{x_v}^S}$ remains connected.

Lastly, for the Hamilton item x_{hp}, $\overline{G_{x_{hp}}^S}$ results from removing all the Hamilton path vertices and their corresponding v' vertices. As shown in Fig. 7.6(d), $\overline{G_{x_{hp}}^S}$ is connected, and looks much like the result of replacing each vertex $v \in G$ with a (broken) cycle that connects its surrounding vertices, excluding v'. An inductive proof based on the sequential removal of the Hamilton path vertices is used to show that the remaining surrounding vertices form a connected subgraph, and since every other vertex (i.e., the face vertices and the $\emptyset$ vertex), is adjacent to a non-v' surrounding vertex, $\overline{G_{x_{hp}}^S}$ remains connected. The basis of the induction is that the extra edge added between the vertices on either side of v_1' keeps v_1's surrounding vertices connected after v_1 and v_1' are removed. As each subsequent Hamilton path vertex pair v_i and v_i' is removed, the two chains of v_i's surrounding vertices remain connected through v_{i-1}'s surrounding vertices. Figures 7.6(c)(d) show the first two steps of the induction.

The following lemma summarizes the result of this section.

Lemma 7.2.1. *Let G be a cubic 3-connected plane graph and let $S(G_p')$ be the derived set system as defined by Def. 7.1.2. If G has a Hamilton path then there is a connectivity graph for $S(G_p')$.*

Proof. See Appendix A.7.2, pg. 285. □

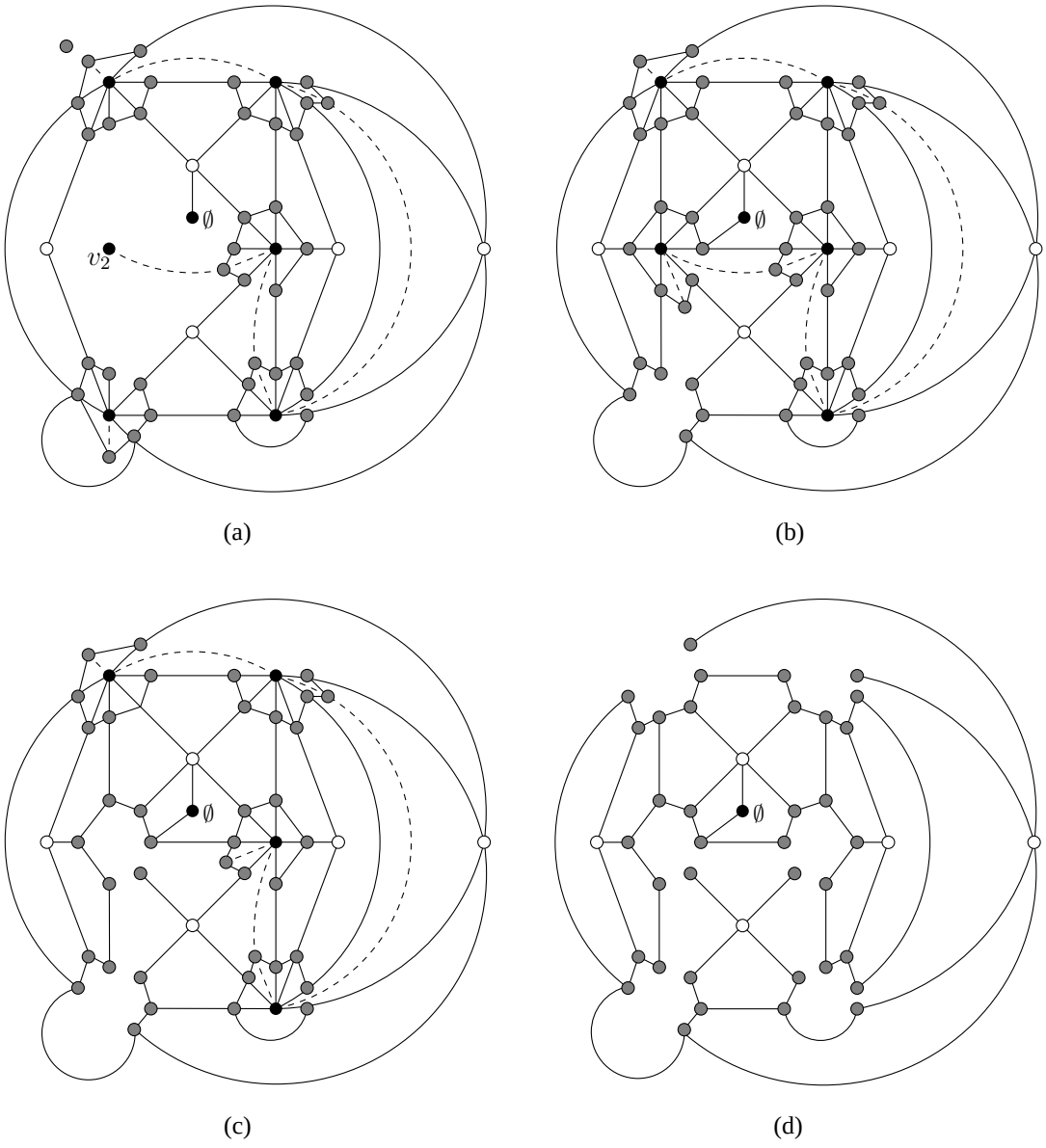


Figure 7.6: (a) Removing the surrounding vertices of v_2, (b) removing v_1 and v_1', (c) removing v_2 and v_2', and (d) the resulting connected $\overline{G^S_{x_{hp}}}$.

7.3 Connectivity Graph → Hamilton Path

Let G be a cubic 3-connected plane graph, and let G^S be a plane embedding of the connectivity graph for the set system $S(G'_p)$ on the items X.

In the construction of $S(G'_p)$, each edge $(u, v) \in G'_p$ is associated with a unique pairwise item x, which is added to the edge's endpoint sets $s(u)$ and $s(v)$. By the definition of a connectivity graph, G^S_x is connected, so there must be an edge $(s(u), s(v)) \in G^S$; therefore, G^S induces a plane embedding of G'_p (that is, a plane embedding of G'_p's underlying *planar* graph). By Prop. 7.1.1, if the singleton v' vertices are ignored, G'_p is homeomorphic to a 3-connected graph. Since all plane embeddings of a 3-connected graph are topologically equivalent [9, Whitney 1932] (i.e., their facial structures are identical except for the choice of outer face), we may consider the plane embedding of G'_p that is induced by G^S to be topologically equivalent to the plane graph that results from applying Def. 7.1.1 to G; in other words, G^S induces a plane embedding of G'_p that looks "similar" to the one in Fig. 7.2(b) in so much as the vertices and faces of G have the same type of structure.

Since G'_p is a subgraph of G^S, G^S has the same layout as G'_p, but with additional plane edges added to achieve the necessary subgraph connectivity. In G^S, the surrounding vertices of each vertex $v \in G$ must be connected since they share a unique *surrounding* item. We make the claim that by connecting vertex v's surrounding vertices, v can be connected to *at most* two other vertices that are adjacent to v in G as shown in Fig. 7.7. As a result, the Hamilton item x_{hp} induces the connected subgraph $G^S_{x_{hp}}$, which is composed of G's vertices and their corresponding v' vertices, and thus represents a *spanning* subgraph of G whose vertices have degree at most 2 (i.e., a Hamilton path in G). Figure 7.8(a) shows an example of $G^S_{x_{hp}}$ and Fig. 7.8(b)

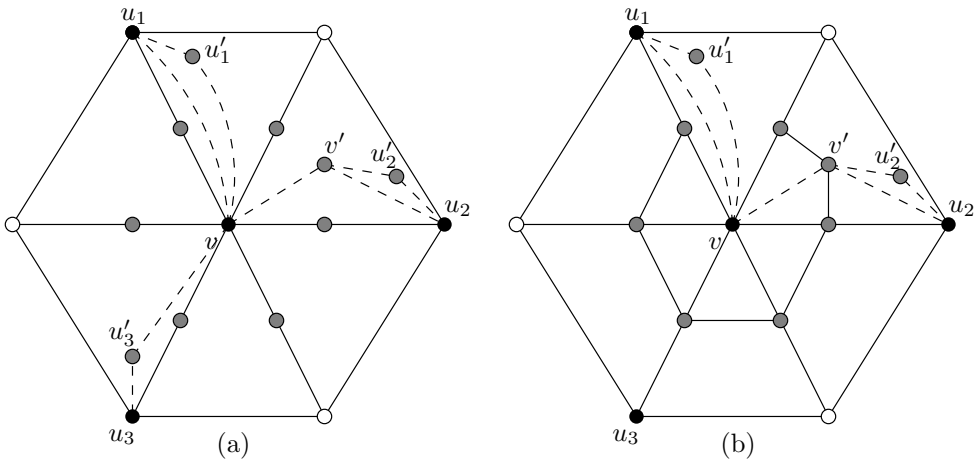


Figure 7.7: (a) In G'_p, a vertex v from G can be connected with plane edges (dashed) to at most three other vertices that are adjacent to v in G, and (b) when v's surrounding vertices are connected in G^S, additional connections can be made between v and *at most* of these adjacent vertices (note that the non-essential surrounding vertices of the u_i are omitted).

shows the induced Hamilton path in G.

The following lemma summarizes the result of this section.

Lemma 7.3.1. *Let G be a cubic 3-connected planar graph with plane embedding G_p and let $S(G'_p)$ be the derived set system as described by Def. 7.1.2. If there exists a connectivity graph for $S(G'_p)$ then G has a Hamilton path.*

Proof. See Appendix A.7.3, pg. 287. □

7.4 Results and Open Problems

We have shown that the EDDP is in NP (Lem. 7.0.5) and that, by Defs. 7.1.1 and 7.1.2, there is a polynomial-time transformation from Hamilton Path on cubic 3-connected planar graphs (HP-C3P) to the EDDP such that an instance of HP-C3P

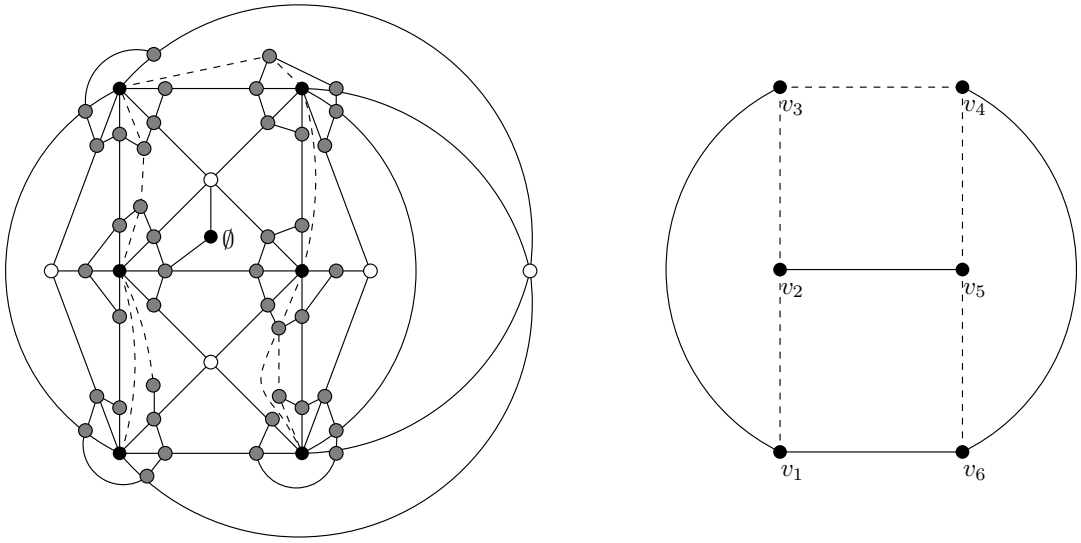


Figure 7.8: (a) The subgraph (dashed edges) of G^S that is induced by the Hamilton item, and (b) the corresponding Hamilton path induced in G.

has a Hamilton path if and only if the transformation to an instance of the EDDP has a connectivity graph (Lems. 7.2.1 and 7.3.1). Since HP-C3P is a known NPC problem, these lemmas and definitions culminate in the following theorem.

Theorem 7.4.1. *The Euler Diagram Decision Problem (EDDP) is NP-complete.* □

In light of the apparent difficulty of efficiently generating Euler diagrams, it is worth considering the corresponding decision problems for generating restricted types of Euler diagrams; in particular, *non-concurrent* Euler diagrams and *simple* Euler diagrams. We will call these EDDP versions the NC-EDDP (for non-concurrent-EDDP), and S-EDDP (for simple-EDDP).

As described by Thm. 5.2.2, given a set system S on the items X, finding a non-concurrent Euler diagram representing S is the same as searching the closeness graph $G_c(S)$ (see Def. 5.2.1), for a simple connectivity graph for S. In turn, by Prop. 5.2.1,

$G_c(S)$ is a subgraph of the $|X|$-dimensional hypercube. It would seem that $G_c(S)$ provides a significantly restricted search space for the NC-EDDP, whereas the EDDP has to potentially search all subgraphs of the complete graph $K_{|S|}$. Similarly, by Thm. 5.4.1, the S-EDDP can restrict its search to planar spanning pseudo-subgraphs of $G_c(S)$.

Even with the restricted search spaces, we have been unable to produce any polynomial time algorithms for the NC-EDDP or the S-EDDP; this difficulty lead us to consider whether these restricted versions of the EDDP could themselves be NPC.

Suppose we tried to transform HP-C3P to the NC-EDDP using a similar approach to the one used in the proof of the EDDP's NP-completeness. The key point of the transformation was to create a set system for which any connectivity graph would contain a specified subgraph. The set system constructed by Def. 7.1.2 does not work for the NC-EDDP because it results in a connectivity graph where adjacent vertices *always* differ by more than one element. That being said, a set system S that forces any connectivity graph $G \subseteq G_c(S)$ to contain a specific subgraph can be constructed using an alternative method.

Consider the plane graph G_p shown in Fig. 7.2(a) and used as an example throughout this chapter. Figure 7.9 shows a transformation of G_p to G'_p (much like that of Def. 7.1.1), where each edge of G_p is bisected. A set system S is constructed from G'_p by assigning a unique integer to each vertex of G_p and its adjacent bisecting vertices in G'_p. Consider the closeness graph $G_c(S)$ By the construction, two vertices are adjacent in $G_c(S)$ only if they are adjacent in G'_p. In addition, since every item induces a star subgraph of G'_p, each edge of G'_p must appear in $G_c(S)$ as well as in any spanning subgraph of $G_c(S)$. As a result, any connectivity graph $G \subseteq G_c(S)$

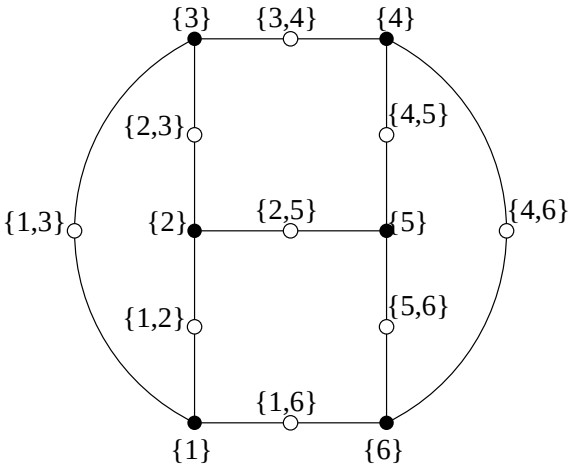


Figure 7.9: The construction of a set system whose non-concurrent connectivity graph is homeomorphic to the graph from Fig. 7.2(a).

must be isomorphic to G'_p.

The problem with the above construction is that it is not possible to specify a set of vertices that must be connected (e.g., the surrounding vertices in the EDDP construction), without specifying *exactly* how they are connected; this works fine for the "Hamilton Path → Connectivity Graph" implication, but not for the converse. A more complicated construction or a transformation from some other NPC problem is needed to prove that the NC-EDDP and the S-EDDP are NPC; at present, we leave these as open problems.

Open Problem 7.4.2. Is the Non-Concurrent Euler Diagram Decision Problem (NC-EDDP) NP-complete?

Open Problem 7.4.3. Is the Simple Euler Diagram Decision Problem (S-EDDP) NP-complete?

Chapter 8

Composite Euler Diagrams

From the previous chapter, we know that the Euler Diagram Decision/Generation Problems (EDDP/EDGP) are difficult to solve efficiently; however, it seems feasible to solve them when the set system S is "small". For example, by Thm. 5.1.1, there exists an Euler diagram representing S if and only if there exists a simple connectivity graph G for S, so a backtracking algorithm that exhaustively searched for G could be used to solve the EDDP. By Thm. 5.0.4, any combinatorial embedding of G is a combinatorial embedding of the Euler dual of an Euler diagram representing S, so the constructive proof of this theorem could be used to solve the EDGP. Unfortunately, the exponential time required to find G becomes a limiting factor as the size of S grows. In addition, any construction algorithm based on connectivity graphs is, by its very nature, restricted to producing connected Euler diagrams, and as we saw in Chapter 6, under some constraints there are set systems that are only representable by disconnected Euler diagrams.

The purpose of this chapter is to describe a heuristic algorithm for solving the

EDDP that addresses the two aforementioned shortcomings of a backtracking approach based on connectivity graphs. Given a set system S as input to the EDDP, the idea of the algorithm is to decompose S into smaller set systems; these set subsystems become the input to an exponential-time EDDP algorithm (e.g., backtracking) that solves each subproblem. If the set system decomposition can be done in polynomial time, the algorithm may allow as input to the EDDP, larger set systems than would otherwise be feasible. The algorithm is a heuristic for two reasons:

1. it still relies on an exponential-time EDDP sub-algorithm, but instead of solving the EDDP for one large set system, it solves the EDDP for (hopefully) many smaller set systems in order to reduce the exponential component of the running time (based on the fact that, for example, $2^4 > 2^2 + 2^2$), and
2. although it never says that an Euler diagram exists if one doesn't, it may say that an Euler diagram does not exist even if one does (i.e., it may give false negatives).

In the first section, we introduce some important Euler diagram properties that form the basis for the algorithm; we also describe an open problem related to why the algorithm may give false negatives. In the second section, we describe the algorithm for decomposing the input set system into subproblems and analyze its running time. Finally, in the third section, we describe how the subproblem solutions can be used as a solution to the original problem input; we also describe how a similar strategy can be used to heuristically solve the EDGP.

8.1 Mathematical Background

Consider the Euler diagram in Fig. 8.1(a); this diagram has curves whose interiors are disjoint (e.g., c_6 and c_7) as well as curves whose interiors are completely within the

interior of another curve (e.g., c_2 and c_1). As shown in Fig. 8.1(b), there is a natural recursive decomposition of this Euler diagram into nested and disjoint subdiagrams. There are a few important features of Fig. 8.1(b):

1. The $\emptyset$ vertex represents the empty plane and is added so that the decomposition into subdiagrams is a rooted tree;
2. Each parent-child edge originates in the region of the parent diagram that contains the child subdiagram;
3. When combined, all but one subdiagram (the one representing c_5), are *strictly* within the interiors of their parent regions, while c_5 *replaces* its parent region (as indicated by the '*').

We begin by formally defining the property that makes the Euler diagram in Fig. 8.1(a), and in general any set of Jordan curves, decomposable into subdiagrams.

Definition 8.1.1. A set $C = \{c_1, c_2, \ldots, c_n\}$ of Jordan curves is *composite* if there exists a non-empty proper subset $C' \subset C$ of curves such that

$$\bigcup_{c \in C'} int(c)$$

is contained by a single connected plane subset of $C \backslash C'$.

If C is not composite, then it is said to be *prime*. □

Example. There are several subsets of curves that satisfy the composite definition for the set C of Jordan curves in Fig. 8.1(a). For example, $C' = \{c_9, c_{10}, c_{11}\}$ is completely within region $\{8\}$ of $C \backslash C'$, and the same is true for $C' = \{c_3\}$, which is completely within region $\{1, 2\}$ of $C \backslash C'$. A special case is $C' = \{c_5, c_6, c_7\}$, which, although intersecting c_4, satisfies the composite definition because its *interior* does not intersect c_4. □

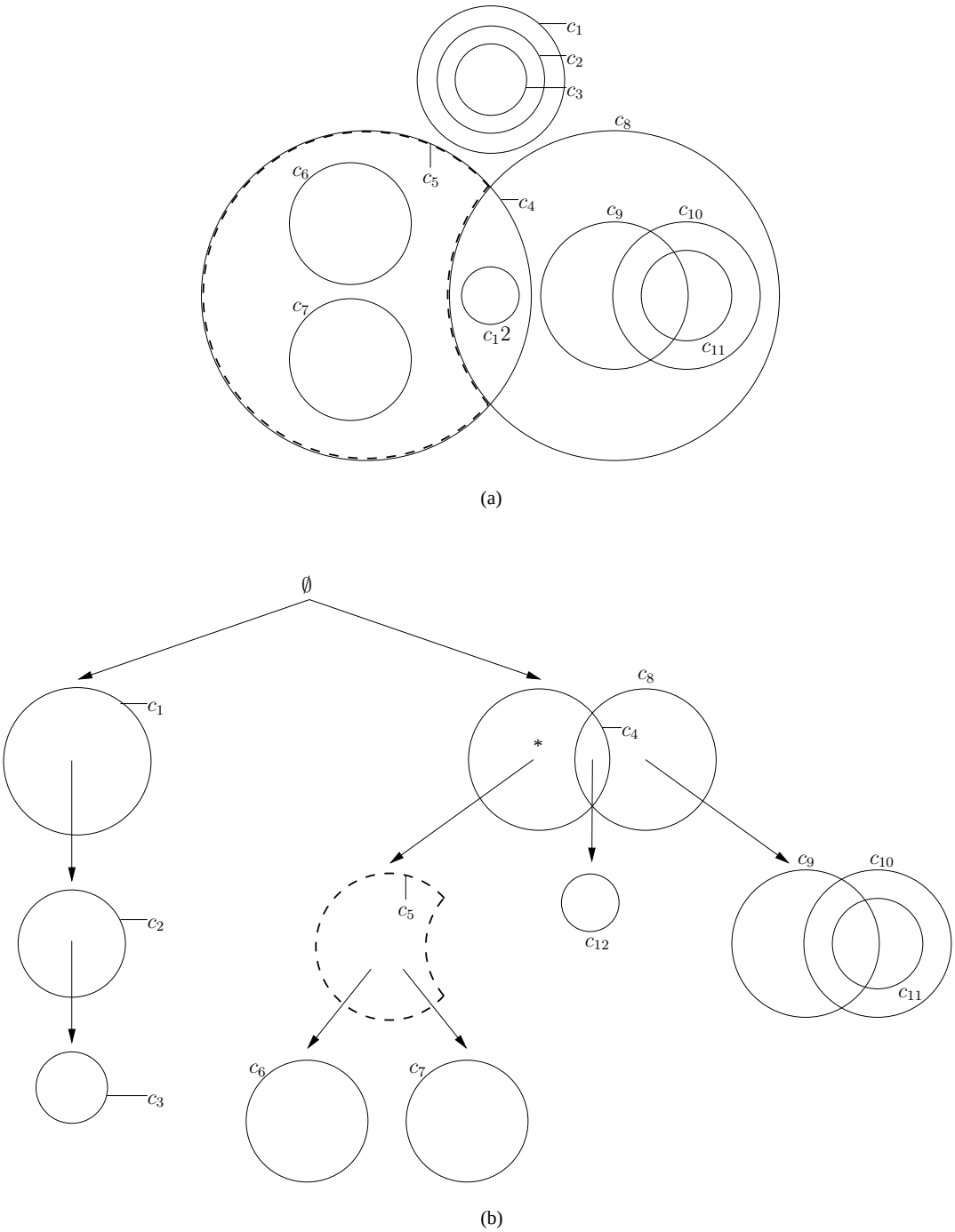


Figure 8.1: (a) A composite set of Jordan curves has (b) a recursive tree structure where each vertex is a prime set of Jordan curves. Note that the diagrams in the tree are not drawn to scale.

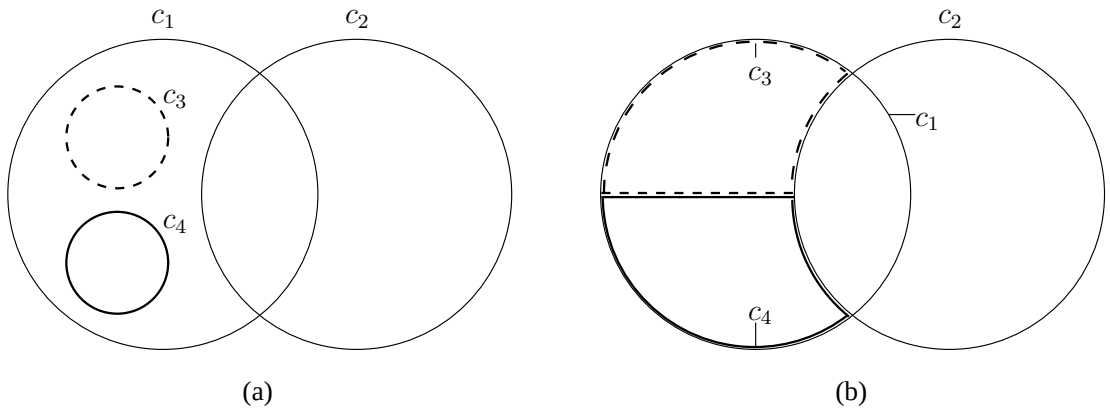


Figure 8.2: (a) Every disconnected set of Jordan curves is composite, but (b) not every connected set of Jordan curves is prime. In both cases, c_3 is a curve satisfying the composite definition.

Clearly, every disconnected set of Jordan curves is composite by considering the topological relationship of its subsets of connected curves, but the inverse is not necessarily true as shown in Fig. 8.2. In other words, every disconnected set of Jordan curves is composite, but not every connected set of Jordan curves is prime.

The decomposition tree in Fig. 8.1(b) represents a maximal decomposition in the sense that each vertex is a prime set of curves; we can think of this tree as a *prime factorization* of a set of Jordan curves in much the same way as composite integers can be factored into prime numbers. An interesting property of composite Euler diagrams is that their prime factorization may not always be composed of Euler diagrams. For example, the Euler diagram C in Fig. 8.3(a) is composite with $C' = \{c_4\}$; however, when C' is removed, $C \backslash C'$ is prime, but not an Euler diagram because region $\{1, 2\}$ is disconnected. On the other hand, Fig. 8.3(b) is another Euler diagram representing the same set system as C, but for which the removal of c_4 *does* result in a prime Euler diagram. This example leads to the following open problem.

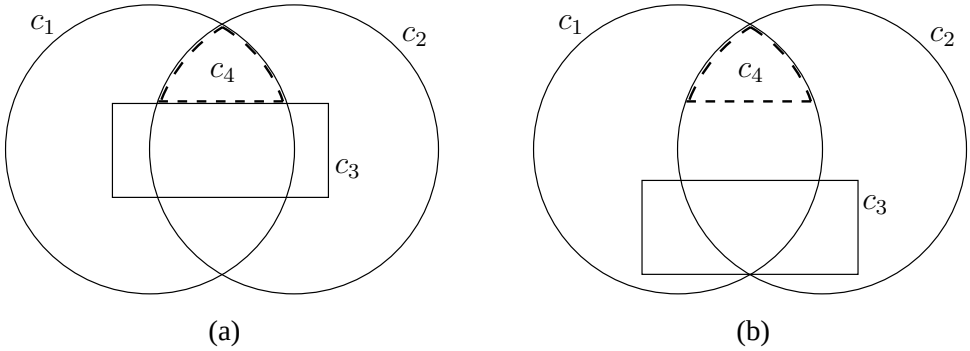


Figure 8.3: (a) A composite Euler diagram C with $C' = \{c_4\}$ satisfying the composite definition, but where $C \backslash C'$ is *not* an Euler diagram because region $r(\{1, 2\})$ is disconnected, and (b) another composite Euler diagram C representing the same system, but for which $C \backslash C'$ *is* an Euler diagram.

Open Problem 8.1.1. Is every set system represented by a composite Euler diagram representable by a composite Euler diagram whose prime factorization is composed of (prime) Euler diagrams? □

We wish to use prime factorization as the basis for decomposing the EDDP into subproblems, but up to now, our description of prime factorization has been within the context of curves (i.e., the *output* of the EDGP); in order to be useful for solving the EDDP, we need to understand how the composite and prime properties relate to set systems (i.e., the *input* of the EDDP). The following statements define the relationship between composite/prime collections of curves and their underlying set systems.

Definition 8.1.2. Let S be a set system on the items X. A pair of items $x, y \in X$ is said to *overlap*, denoted xOy, if S contains the following sets:

1. a set s_{xy} such that $x, y \in s_{xy}$,
2. a set s_x such that $x \in s_x$ and $y \notin s_x$, and

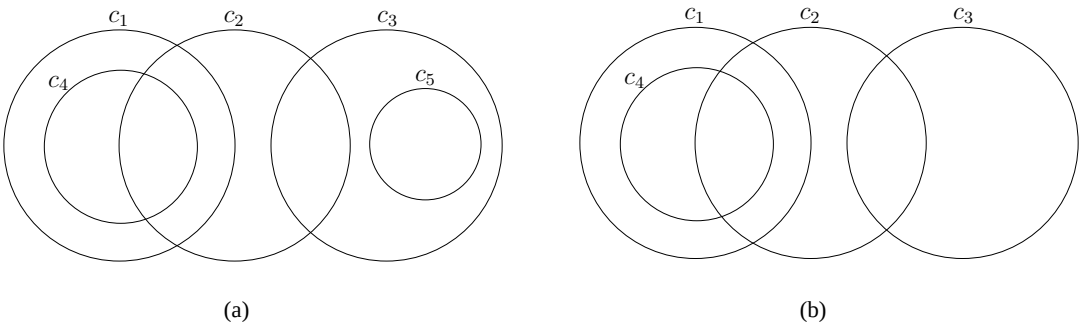


Figure 8.4: Examples of Euler diagrams representing (a) an unlinked set system, and (b) a linked set system.

3. a set s_y such that $y \in s_y$ and $x \notin s_y$. □

Example. Consider the set system

$$S = \{\emptyset, \{1\}, \{1,4\}, \{1,2\}, \{1,2,4\}, \{2\}, \{2,3\}, \{3\}, \{3,5\}\}$$

on the items $X = \{1,2,3,4,5\}$.

Items $x = 1$ and $y = 2$ overlap due to the sets $s_{xy} = \{1,2\}$, $s_x = \{1\}$, and $s_y = \{2\}$. Items $x = 1$ and $y = 4$ *do not* overlap because although there are sets $s_{xy} = \{1,4\}$ and $s_x = \{1\}$, there is no set $s_y \in S$ with $4 \in s_y$ and $1 \notin s_y$.

We can get an idea for the rationale of the "overlap" relation by considering Fig. 8.4(a), which shows an Euler diagram representing S. Because items 1 and 2 overlap, c_1 and c_2 transversely intersect; this is not the case for c_1 and c_4 because items 1 and 4 do not overlap. The diagram allows us to quickly deduce all the pairs of overlapping items: 1 and 2, 2 and 3, and 2 and 4. □

The following proposition formalizes the correspondence between items that overlap and their their respective curves.

Proposition 8.1.2. *Let S be a set system on the items X and C be an Euler diagram representing S. For any two items $x, y \in X$, the corresponding curves $c_x, c_y \in C$ transversely intersect if and only if xOy.*

Proof. See Appendix A.8.1, pg. 296. □

Consider items 1, 2, and 3 from the previous example. We determined that $1O2$ and $2O3$, but that $1\not O3$; therefore, the overlap relation is not transitive. However, as we can see from Fig. 8.4(a), there is still a "bond" between c_1 and c_3. The following definition formalizes the relationship between items 1 and 3.

Definition 8.1.3. Let S be a set system on the items X. A pair of items $x, y \in X$ is said to be *linked*, denoted xLy, if xOy or there exists an item $z \in X$ such that xOz and zLy.

In other words, xLy if there exists a sequence $x = x_1, x_2, \ldots, x_k = y$ of items such that x_iOx_{i+1} for all $1 \leq i < k$. □

Example. Referring to the set system S from the previous example, we know that $1O2$, $2O3$, and $2O4$; therefore, we can deduce the following pairs of linked items: 1 and 2, 1 and 3, 1 and 4, 2 and 3, 2 and 4.

As for the "overlap" relation, the rationale for the "linked" relation can be understood by considering Fig. 8.4(a). Two items x and y are linked if there exists a sequence of curves beginning at x and ending at y for which consecutive curves in the sequence transversely intersect. For example, 1 and 4 are linked because their exists a sequence c_1, c_2, c_4 where c_1, c_2 and c_2, c_4 are pairs of transversely intersecting curves. □

Although the linked relation applies to pairs of items, we can also think of *linked* as a set system property as stated in the following definition.

Definition 8.1.4. Let S be a set system on the items X. S is said to be *linked* if for every pair of items $x, y \in X$, xLy.

If S is not linked, then S is said to be *unlinked*.

Note that by this definition, a set system on a single item is linked. □

Example. The set system S on the items $X = \{1, 2, 3, 4, 5\}$ from the previous examples is *not* linked because, for example, $1 \not L 5$.

If we consider the set system

$$S' = \{\emptyset, \{1\}, \{1, 4\}, \{1, 2\}, \{1, 2, 4\}, \{2\}, \{2, 3\}, \{3\}\}$$

on the items $X = \{1, 2, 3, 4\}$, which is derived from S by removing any sets involving item 5, then S' is linked. Figure 8.4(b) shows an Euler diagram representing S'. Note how the unlinked set system S is represented by a composite Euler diagram in Fig. 8.4(a) while the linked set system S' is represented by a prime Euler diagram in Fig. 8.4(b). □

Finally, the following theorem relates the notion of a linked set system to a prime Euler diagram.

Theorem 8.1.3. *Let S be a set and C be an Euler diagram representing S. C is prime if and only if S is linked.*

Proof. See Appendix A.8.1, pg. 297. □

Based on the previous theorem, given a prime factorization composed of Euler diagrams, we can associate each vertex with a linked set system; in this way, the linked set systems have an explicit hierarchy. For example, the prime Euler diagram $C' = \{c_9, c_{10}, c_{11}\}$ in Fig. 8.1(b) represents the linked set system $S' = \{\{9\}, \{9, 10\}, \{9, 10, 11\}, \{10\}, \{10, 11\}\}$, which is a child of the linked set system $\{\{4\}, \{4, 8\}, \{8\}\}$. The linked set systems in a prime factorization of a composite Euler diagram representing the set system S are not quite a partition of S in the sense that, although there is a bijection between S and the union of the linked set systems, the union is not S. For example, when considered within the context of its originating composite Euler diagram, the linked set system S', which we previously described, represents the set system $\{\{8, 9\}, \{8, 9, 10\}, \{8, 9, 10, 11\}, \{8, 10\}, \{8, 10, 11\}\}$. In the following section, we describe an algorithm for decomposing a set system S into a hierarchy of linked set subsystems as would be represented by the prime factorization of an Euler diagram representing S.

8.2 Factorization Algorithm

Let S be a set system on the items X, and suppose C is a composite Euler diagram representing S whose prime factorization T is composed of Euler diagrams. For

example, the set system

$$\begin{aligned} S \;=\; \{\;\; & \emptyset, \\ & \{1\}, \{1,2\}, \{1,2,3\}, \\ & \{4,5\}, \{4,5,6\}, \{4,5,7\}, \\ & \{4,8\}, \{4,8,12\}, \\ & \{8\}, \{8,9\}, \{8,9,10\}, \{8,9,10,11\}, \{8,10\}, \{8,10,11\}\} \end{aligned}$$

on the items $X = \{i | 1 \leq i \leq 12\}$ is represented by the composite Euler diagram in Fig. 8.1(a) whose prime factorization in Fig. 8.1(b) is composed of Euler diagrams; Fig. 8.1(b) is duplicated in Fig. 8.5(a) for future reference.

Suppose we let T_S be a relabeling of T where each prime Euler diagram C' is replaced by the set system it represents (excluding the implied $\emptyset$). For example, Fig. 8.5(b) shows the T_S that corresponds to the example T of Fig. 8.5(a); note how the prime Euler diagram $C' = \{c_4, c_8\}$ in Fig. 8.5(a) is replaced by $\{\{4\}, \{4,8\}, \{8\}\}$ in Fig. 8.5(b). In addition, each edge of T between a parent diagram C'_p and a child diagram C'_c is labeled in T_S with the region of C'_p that contains C'_c. For example, in Fig. 8.5(a), $C'_c = \{c_{12}\}$ is contained by region $\{4,8\}$ of its parent diagram $C'_p = \{c_4, c_8\}$; therefore, the edge between C'_p and C'_c in T_S is labeled $\{4,8\}$. Any regions of T that are labeled with an '*' to indicate that the subdiagram *replaces* the region are also labeled with an '*' in T_S. For example, region $\{4\}$ in the example T_S of Fig. 8.5(b) is labeled $\{4\}^*$.

Lastly, we let T_X be a relabeling of T_S where each set system is replaced by its respective items. For example, Fig. 8.5(c) shows the T_X that corresponds to the example T_S of Fig. 8.5(b); note how the set system $\{\{4\}, \{4,8\}, \{8\}\}$ in Fig. 8.5(b)

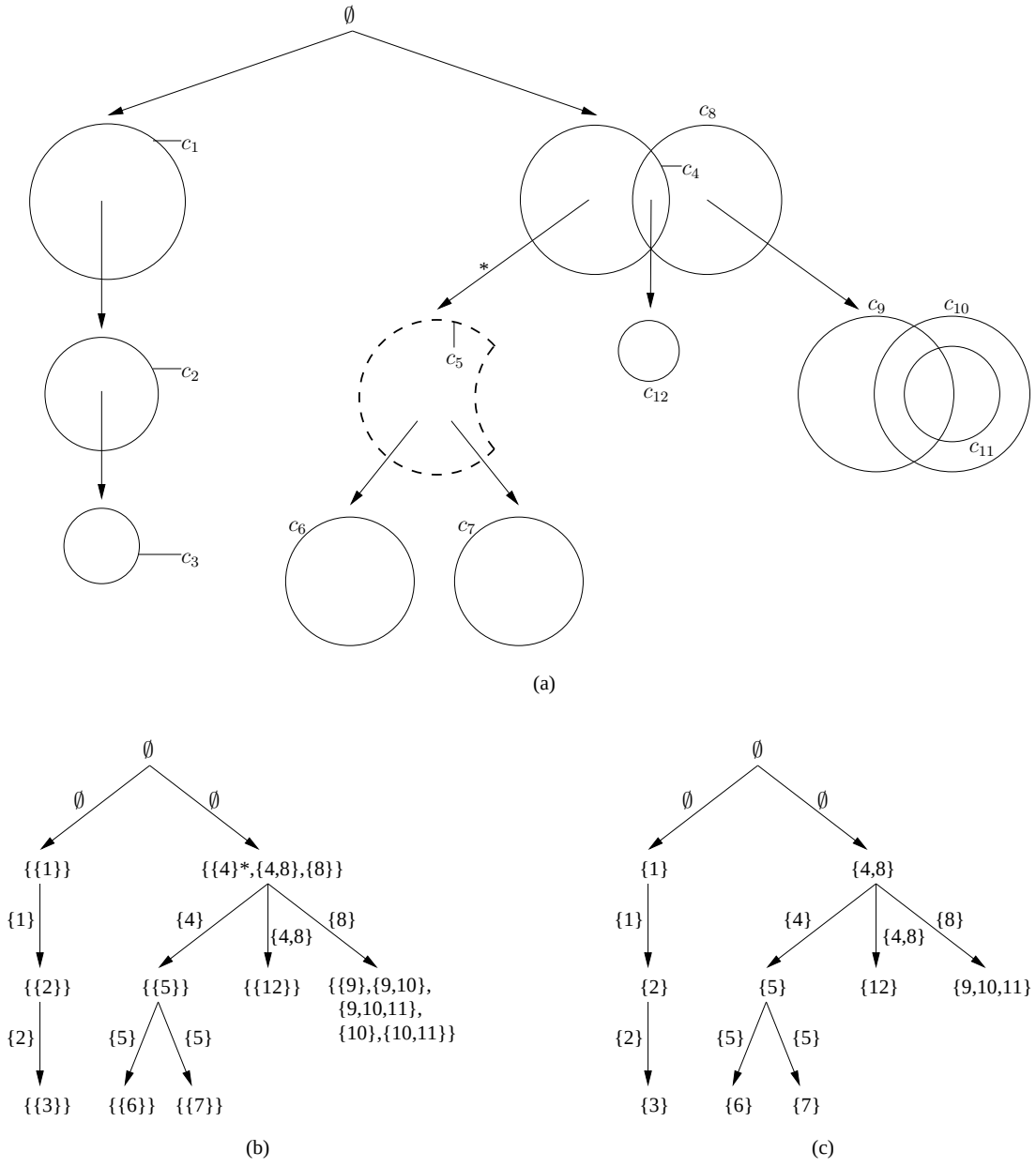


Figure 8.5: The process of transforming (a) a prime factorization T composed of Euler diagrams into (b) a relabeled tree T_S where each prime Euler diagram is replaced by the set system it represents (excluding $\emptyset$), and the edges are labeled with the parent region whence they originated, and finally into (c) a relabeled tree T_X where each set system is replaced by its respective items.

is replaced by $\{4, 8\}$ in Fig. 8.5(c).

The following subsections describe the algorithms for generating T_X and T_S in the absence of a composite Euler diagram (i.e., from the input set system alone). In the next section we describe how to use T_S to solve the EDDP/EDGP.

8.2.1 T_X Generation

We begin by describing Alg. 1, the T_X generation algorithm, with a running example using the input set system

$$\begin{aligned} S \;=\; \{\;\; & \emptyset, \\ & \{1\}, \{1,2\}, \{1,2,3\}, \\ & \{4,5\}, \{4,5,6\}, \{4,5,7\}, \\ & \{4,8\}, \{4,8,12,13,14\}, \\ & \{8\}, \{8,9\}, \{8,9,10\}, \{8,9,10,11\}, \{8,10\}, \{8,10,11\}\} \end{aligned}$$

on the items $X = \{i | 1 \leq i \leq 14\}$.

The first step of Alg. 1, lines 1–11, discovers which items of X overlap according to Def. 8.1.2 and records this information in G. G is a blend of undirected and directed graphs in the sense that an edge (x, y) may have four states: undirected, directed from x to y, directed from y to x, and bidirected. Note that there is only ever at most one edge between (x, y), even if it is bidirected. Each item is represented by a vertex in G, and the relationship between any two items is represented by edges in G as follows:

1. there is no edge between x and y if x and y *never* appear together in a set of S,

Algorithm 1 Generate T_X.

Require: A set system S on the items X.
Ensure: A T_X tree similar to Fig. 8.5(c).

$G \leftarrow$ a graph with $V(G) = X$ and $E(G) = \emptyset$ whose edges can be undirected, unidirected, or bidirected.
{Connect any vertices in G that appear together in a set of S.}
for all sets $s \in S$ **do**
    for all pairs of items $x, y \in s$ **do**
        add an undirected edge (x, y) to G
{Add directions to G's edges.}
for all edges $(x, y) \in E(G)$ **do**
    if $\exists\ s_x \in S$ such that $x \in s_x$ and $y \notin s_y$ **then**
        add a direction to (x, y) from x to y
    if $\exists\ s_y \in S$ such that $y \in s_y$ and $x \notin s_y$ **then**
        add a direction to (x, y) from y to x
{Consolidate equivalent items.}
for all undirected edges $(x, y) \in E(G)$ **do**
    remove vertex y from G
    relabel vertex x to $x = y$
{Consolidate strongly connected components (SCC).}
$G' \leftarrow$ a directed graph with $V(G') =$ the SCCs of G and $E(G') = \emptyset$
for all unidirected edges $(x, y) \in E(G)$ **do**
    $[x] \leftarrow$ the SCC containing x
    $[y] \leftarrow$ the SCC containing y
    if $([x], [y]) \notin E(G')$ **then**
        add an edge to G' directed from $[x]$ to $[y]$
        initialize $([x], [y])$'s label to $\emptyset$
    add x to $([x], [y])$'s label
{Ensure G' is connected.}
add a vertex $\{\emptyset\}$ to G'
for all source vertices $v \in V(G')$ **do**
    add an edge to G' directed from $\{\emptyset\}$ to v and labeled $\emptyset$
return $T_X \leftarrow$ the transitive reduction of G'

2. there is an undirected edge between x and y if x and y *always* appear together in a set of S,
3. there is a directed edge from x to y if x and y appear together in a set of S, x appears in a set without y, and y *never* appears in a set without x,
4. there is a directed edge from y to x if x and y appear together in a set of S, y appears in a set without x, and x *never* appears in a set without y, and
5. there is a bidirected edge between x and y if x and y appear together in a set of S, x appears in a set without y, and y appears in a set without x.

In other words, those items connected by an undirected edge are equivalent in the sense that their resulting curves in an Euler diagram will be the same and those items connected by a bidirected edge overlap; the implication of the unidirected edges will be clear later on. Figure 8.6(a) shows what G looks like after line 5 for the example set system; edges exist between items that appear together in a set. Figure 8.6(b) shows what G looks like after line 11 for the example set system; the edges are now oriented to indicate how items that appeared together relate with respect to the other sets.

The second step of Alg. 1, lines 12–15, involves discovering and removing equivalent items. Each set X_e of equivalent items is replaced by one of its members $x \in X_e$ whose label in G is updated to reflect the items in X_e. From this point on, x will take the place of any member of X_e; in any subsequent Euler diagram, curve c_x can be duplicated for each member of X_e should a representation of the equivalent items be desired. In this sense, Alg. 1 goes beyond recognizing overlapped items; it also identifies equivalent items whose presence is redundant and may contribute to unnecessary computation. At the end of this step, G only has unidirected and bidirected edges.

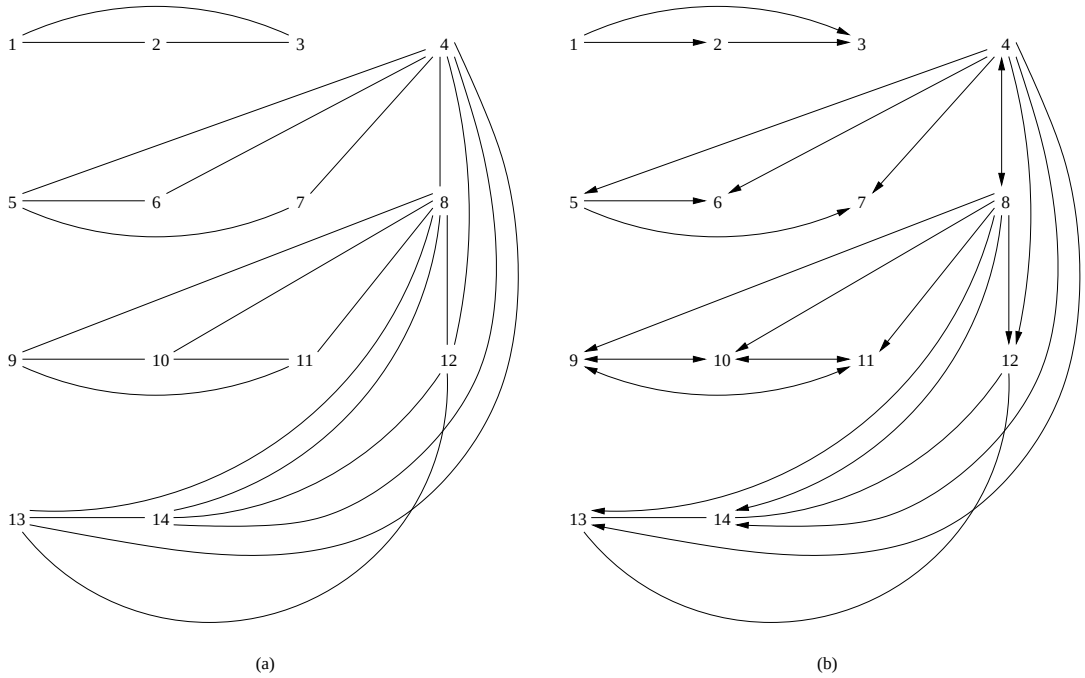


Figure 8.6: An example of Alg. 1 showing (a) what G looks like after line 5, and (b) what G looks like after line 11.

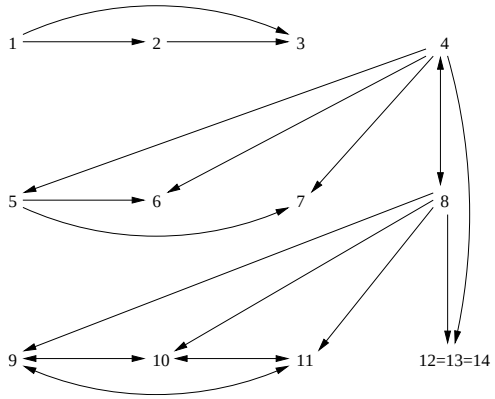


Figure 8.7: An example of Alg. 1 showing what G looks like after line 15.

Figure 8.7 shows what G looks like after line 15 for the example set system; note how items 12, 13, and 14 are now represented by a single item, but in Fig. 8.6 they are isomorphic.

The third step of Alg. 1, lines 16–23, groups together overlapped items into sets of items for which every pair is linked according to Def. 8.1.3. The strongly connected components (SCCs) of G are the connected subgraphs of G comprising *only* bidirected edges. Since within an SCC there is a path composed of bidirected edges between any two items x and y, there is a sequence of items between x and y whose consecutive pairs overlap and, by Def. 8.1.3, this implies that x and y are linked. In other words, each SCC in G represents a set of items closed under the linked relation. Any set subsystem on the items of an SCC would then, by Def. 8.1.4, be a linked set subsystem.

Algorithm 1 represents each of G's SCCs as a vertex in a new directed graph G'. Each vertex of G' represents the set of items in its corresponding SCC. For an item x, the set of items in its SCC is denoted $[x]$.

G's bidirected edges define the SCCs, but what about the undirected edges? For

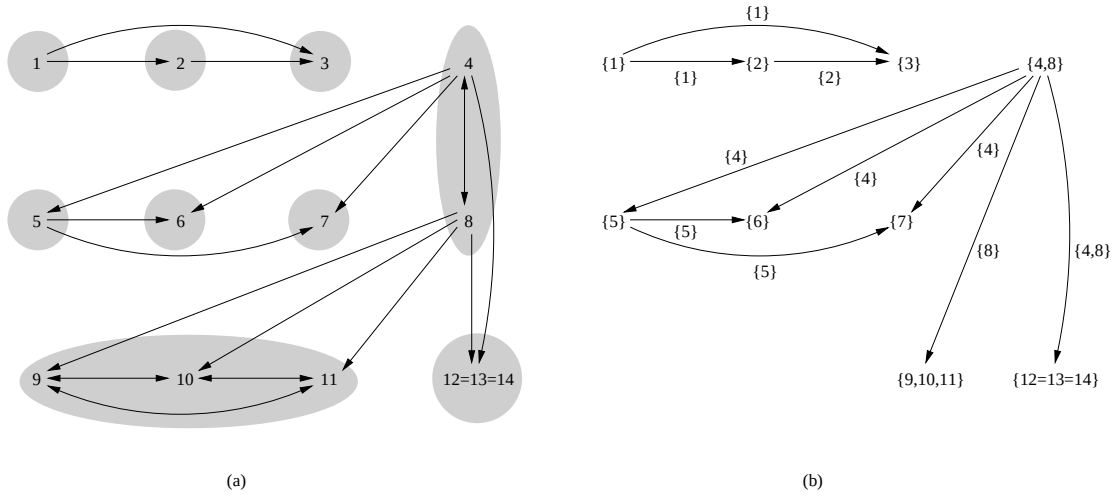


Figure 8.8: An example of Alg. 1 showing (a) what the SCCs look like in G after line 15, and (b) what G' looks like after line 24.

any two SCCs $[x]$ and $[y]$, Alg. 1 groups together the edges in G that are directed from any item in $[x]$ to any item in $[y]$ and represents them with a single edge $([x], [y])$ in G', which is labeled with the set of originating items from $[x]$. By Prop. A.8.5, pg. 293, if there is an edge in G directed from $x \in [x]$ to $y \in [y]$, then there are edges directed from x to *every* item in $[y]$; because of this fact and the labeling of $([x], [y])$, no information is lost in the transformation from G to G'. In addition, by Prop. A.8.8, pg. 295, G' is a directed acyclic graph (DAG).

Figure 8.8(a) is a version of Fig. 8.7 where the SCCs are highlighted with a shaded background. Figure 8.8(b) shows what G' looks like after line 24 for the example set system; note how the SCCs are represented by their respective sets of items and how the edges are labeled with the set of items from whence they originated in Fig. 8.8(a). Also note how in Fig. 8.8(b) an item in one SCC with an edge to an item in another SCC, for example $(8, 9)$, also has edges to every other item in the SCC.

The final step of Alg. 1, lines 24–28, ensures that G' is connected by adding an $\emptyset$

vertex and attaching it to each of G' source vertices; since G' is a DAG, the resulting graph is connected. Finally, T_X is returned as the transitive reduction of G', which by Prop. A.8.9, pg. 296, is a rooted tree. The transitive reduction of G' is the minimal directed subgraph whose transitive closure is the same as G'; in other words, edges that are implied by transitivity are removed.

Figure 8.9(a) shows what G' looks like after line 28 for the example set system; note how G' is a DAG and compare it to Fig. 8.8(b) to see how the $\emptyset$ vertex connects the graph. Figure 8.9(b) shows the transitive reduction of Fig. 8.9(a); note how the transitive reduction is a rooted tree and compare it to Fig. 8.5(c). The only difference between the trees in Fig. 8.5(c) and Fig. 8.9(b), aside from a different drawing, is the $\{12\}$ vertex in Fig. 8.5(c) is $\{12 = 13 = 14\}$ in Fig. 8.9(b) since the set system was slightly changed to provide an example of equivalent items; the similarity between the trees is noteworthy considering the former was derived from an Euler diagram while the latter was derived from a set system.

Finally, the following theorem indicates that T_X can be generated in polynomial time.

Theorem 8.2.1. *For a set system S on the items X, the running time of Alg. 1 is $O(|X|^2|S| + |X|^3)$.*

Proof. See Appendix A.8.2, pg. 300. □

8.2.2 T_S Generation

The vertices of T_X represent maximal subsets of linked items, but in order to use them as input to EDGP subproblems, we need to determine which subsets of the

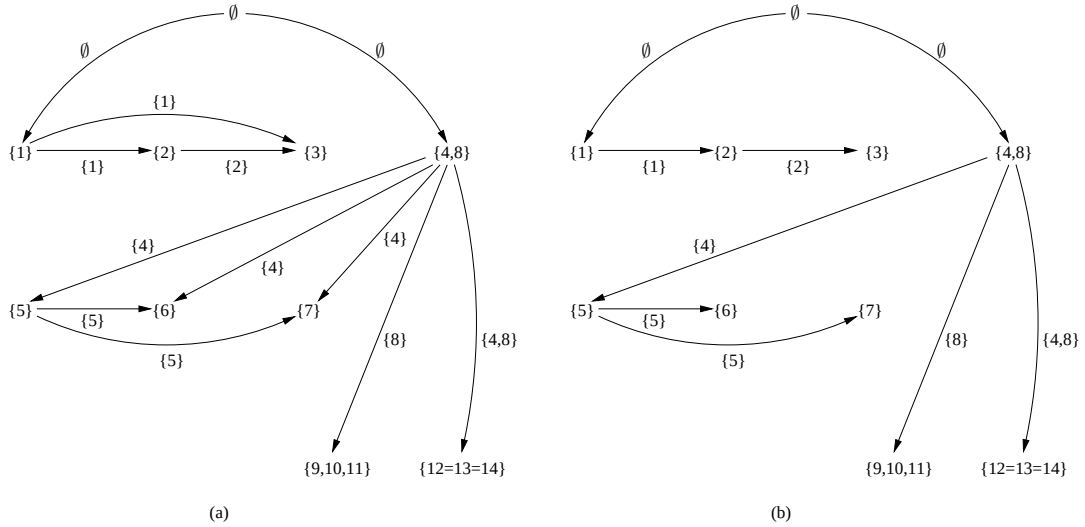


Figure 8.9: An example of Alg. 1 showing (a) what G' looks like after line 28, and (b) the final generated T_X tree.

input set system they represent. Algorithm 2 describes how to transform T_X into T_S, a form suitable for processing by EDGP subproblems.

We describe Alg. 2 by continuing with the example set system S and T_X from the previous section.

The first step of Alg. 2, lines 1–4, creates a mapping of the vertices of T_X, which are subsets of items, to set subsystems. Each set from S will be put into exactly one set subsystem (after being slightly modified).

The second step of Alg. 2, lines 5–13, takes each set $s \in S$ and descends T_X looking for the shallowest vertex v for which s is a subset of the union of the vertices from v to the root. During the descent, items that will be in ancestor diagrams of whichever child diagram s ends up being represented in are removed by line 11 for the same reason they do not appear in Fig. 8.5(b). In the following description, we let s be the original set and s' be the set modified during the descent. Line 7 initializes

Algorithm 2 Generate T_S.

Require: A set system S on the items X and T_X from Alg. 1.
Ensure: A T_S tree similar to Fig. 8.5(b).

{Initialize map of T_X's vertices to set subsystems.}
$\sigma \leftarrow$ lookup table for $V(T_X)$
for all vertices $v \in V(T_X)$ **do**
    $\sigma[v] \leftarrow \emptyset$
{Populate the set subsystems.}
for all sets $s \in S \backslash \{\emptyset\}$ **do**
    $v \leftarrow$ root of T_X
    while $s \not\subseteq v$ **do**
        $v' \leftarrow$ child of v with $v' \cap s \neq \emptyset$
        $l \leftarrow (v, v')$'s label
        $s \leftarrow s \backslash l$
        $v \leftarrow v'$
    $\sigma[v] \leftarrow \sigma[v] \cup \{s\}$
{Make T_S analog of T_X with set subsystem vertices.}
$T_S \leftarrow T_X$
for all vertices $v \in V(T_S)$ **do**
    $v \leftarrow \sigma[v]$
{Add any regions needed by subdiagrams}
for all edges $(x, y) \in E(T_S)$ **do**
    $l \leftarrow (x, y)$'s label
    if $l \notin x$ **then**
        $x \leftarrow x \cup \{l^*\}$
return T_S

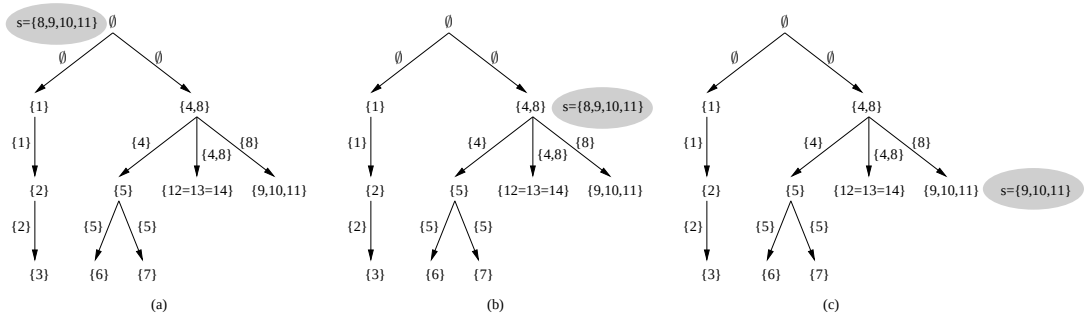


Figure 8.10: An example of how Alg. 2 finds the proper location for $s = \{8, 9, 10, 11\}$.

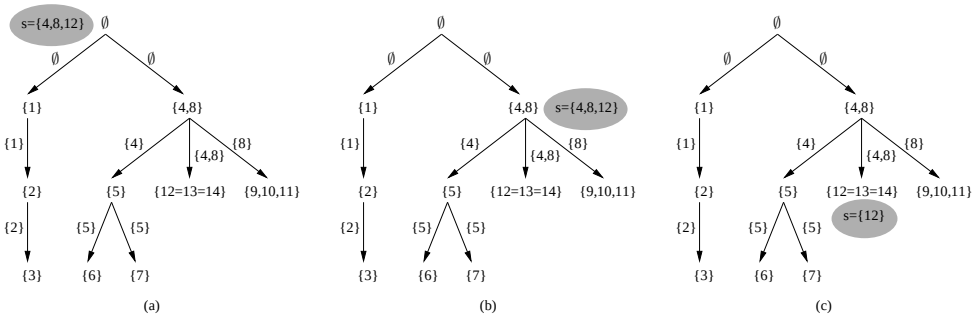


Figure 8.11: An example of how Alg. 2 finds the proper location for $s = \{4, 8, 12\}$.

v to be the root. The search stops when s' is a subset of v; at this point, we know that all items needed by s appear on the path from v to the root. When s' is not a subset of v, the child v' of v that contains an item in s' is chosen as the next vertex in the search and s' is updated to remove (v, v')'s label. This search is guaranteed to succeed because, by nature of T_X's construction, only one child of v can have an item in s' and if s' is not a subset of v, whatever items are missing must be in a descendent of v.

Figures 8.10, 8.11, and 8.12 show examples of how the second step of Alg. 2 finds the appropriate location for three different sets. Note that in Fig. 8.11, the set $s = \{4, 8, 12\}$ actually represents $\{4, 8, 12, 13, 14\}$ but is concatenated since items 12, 13, and 14 are equivalent.

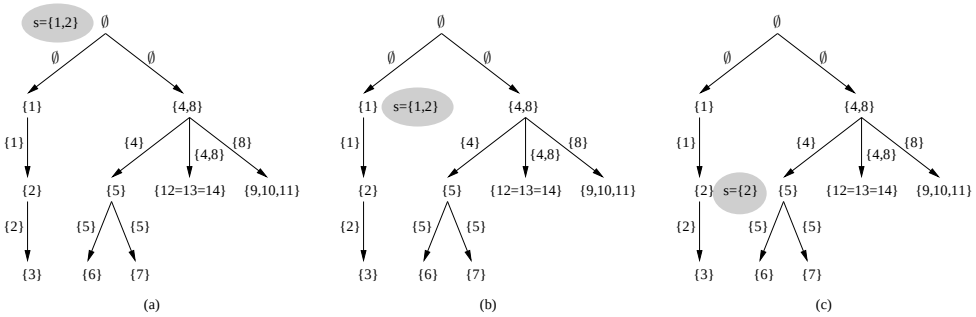


Figure 8.12: An example of how Alg. 2 finds the proper location for $s = \{1, 2\}$.

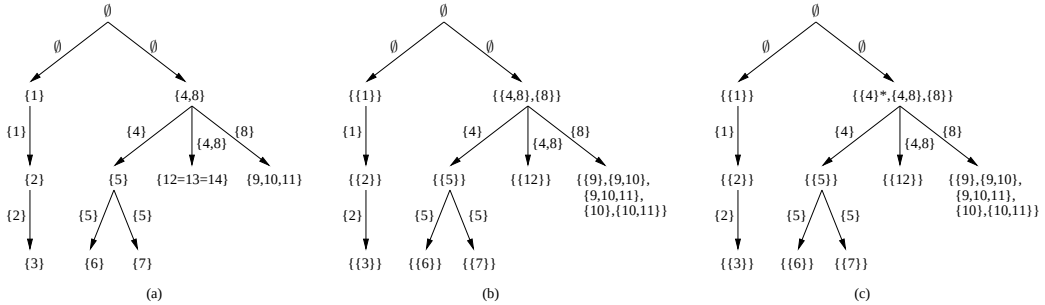


Figure 8.13: An example of Alg. 2 where (a) is the T_X tree, (b) is the T_S tree after line 19, and (c) is the T_S tree after line 23.

The third step of Alg. 2, lines 14–22, creates T_S by copying T_X and replacing each vertex v by its respective set subsystem. Afterwards, any parent regions that do not exist in S because they are *replaced* by a child diagram are added and annotated with an '*'. Figure 8.13(a) shows the example T_X, Fig. 8.13(b) shows the T_S that results when the vertices of T_X are replaced with their set subsystems, and Fig. 8.13(c) shows the final T_S after any replaced regions (e.g., {4}), are added.

Finally, the following theorem indicates that T_S can be generated in polynomial time.

Theorem 8.2.2. *For a set system S on the items X, the running time of Alg. 2 is $O(|X|^2 + |X||S|)$.*

Proof. See Appendix A.8.2, pg. 302. □

Now that T_S is constructed, we consider in the next section how to use it to solve the EDDP and EDGP.

8.3 Composition Algorithm

T_S tells us which linked set systems are represented by each subdiagram in the prime factorization of an Euler diagram representing S (if such a diagram exists). Suppose we had an algorithm `PRIME` that could construct each prime subdiagram or tell us if no such diagram existed. How could we use `PRIME` to convert T_S into a composite Euler diagram representing S?

One strategy is to traverse T_S's vertices, which can be done in polynomial time, and run `PRIME` on each linked set system. If `PRIME` indicates that any of the subdiagrams does not exist, then our algorithm stops and indicates that no composite Euler diagram exists. Depending on the answer to Open Problem 8.1.1, this negative result may be false, and so our algorithm is possibly heuristic.

Suppose `PRIME` is able to construct each subdiagram in T_S, then the tree T that results from replacing each linked set system in T_S with the corresponding prime subdiagram looks much like the prime factorization of a composite Euler diagram. Is it possible to reverse the factorization process and recover the original composite Euler diagram? Clearly, if the answer is "Yes", then the result is a composite Euler diagram representing S.

We will take an inductive bottom-up approach to the construction, beginning with the leaves of T. The leaves are the base case and `PRIME` has already constructed the

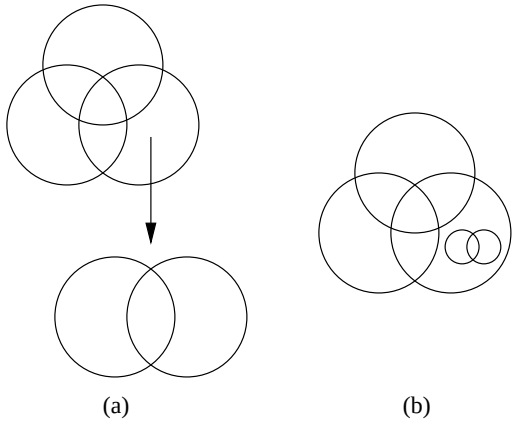


Figure 8.14: (a) A prime factorization where the parent region is *not* replaced corresponds (b) to a composite Euler diagram where the subdiagram is scaled and translated.

diagrams representing each of the leaves. For the inductive step, given a vertex v of T, we assume that the (possibly composite) Euler diagrams represented by each subtree of v have been constructed. In addition, let C_v be the prime Euler diagram constructed by `PRIME` for the linked set system represented by v. For each region r of C_v, we use the same continuous plane transformation described in Section 4.2 to connect all the subdiagrams that will be nested in r. As a result, the basic operation involves nesting an Euler diagram within a single region of a prime Euler diagram. Clearly, as shown in Fig. 8.14, if the subdiagram does *not* replace region r, then a simple scaling and translation of the subdiagram suffices to accomplish the nesting.

The case where the subdiagram *replaces* region r is more difficult to accomplish. We first note that since C_v is prime, region r has a continuous boundary (i.e., r has no holes). As a result, as shown in Fig. 8.15, some subdiagrams can be continuously deformed and translated in order to accomplish the nesting; after the translation, the subdiagram remains topologically equivalent to its original form. Unfortunately, there are some subdiagrams whose topology changes when they are continuously deformed

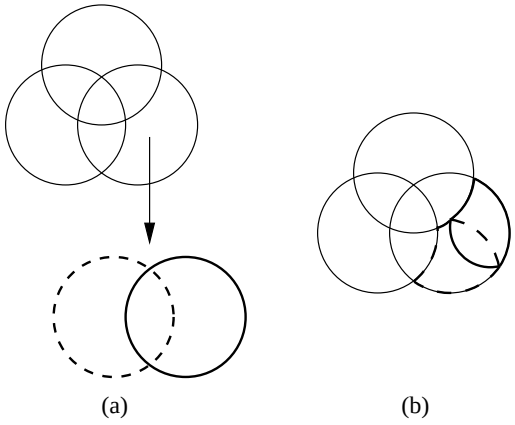


Figure 8.15: (a) A prime factorization where the parent region is replaced, and (b) the corresponding composite Euler diagram where the subdiagram's topology is preserved.

and translated. For example, the subdiagram shown in Fig. 8.16(a) has a point intersection between its two circles; as a result, the circles' interior are *not* adjacent. However, as shown in Fig. 8.16(b), after the nesting transformation, the subdiagram's curves become concurrently intersecting, and thus have adjacent interiors. That being said, the resulting composite Euler diagram still represents the proper set system.

The nesting case demonstrated by Fig. 8.16 occurs whenever the boundary of subdiagram's outer face has a cut point. In such cases, this point represents the tangential intersection of two or more curves, and "stretching" the point into a concurrent curve segment does not create or remove any regions, nor does introduce any curve self-intersections, which would violate the Jordan curve requirement. As a result, any region r of C_V can be replaced by any subdiagram, and the result will be an Euler diagram representing the correct set system.

In the end, if `PRIME` can construct prime Euler diagrams representing each of T_S's linked set systems, then a composite Euler diagram representing S can be constructed using the nesting transformations just described.

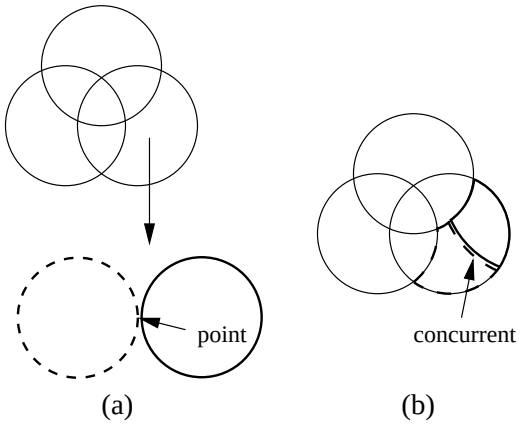


Figure 8.16: (a) A prime factorization where the parent region is replaced, and (b) the corresponding composite Euler diagram where the subdiagram's topology is altered.

If we are just interested in heuristically solving the EDDP, then it suffices to replace `PRIME` with an algorithm `EXISTS` that answers the EDDP for linked set systems: in a traversal of T_S, if `EXISTS` returns "No" at any vertex then return "No"; otherwise, return "Yes". As we mentioned, this EDDP is heuristic because it might return a false negative due to Open Problem 8.1.1; however, any "Yes" answer is correct. Since T_S can be generated and traversed in polynomial time, the complexity of the heuristic depends on the `EXISTS` algorithm. Based on Chapter 7, `EXISTS` is likely to have exponential running-time; however, the hope is that the decomposition of S into linked set subsystems will reduce `EXISTS`'s input to a feasible size.

To heuristically solve the EDGP, we not only have to run `PRIME` on each linked subsystem, but we also need to recursively construct the final composite Euler diagram; in order to be useful, it must be possible to do this construction in polynomial time. Although we leave this as an open problem, we have begun initial work that suggests that a polynomial time construction is possible. Our algorithm is based on `PRIME` returning a connectivity graph representation for each prime Euler diagram,

which is reasonable based on the results of Chapter 5, and then recursively combining the subdiagrams' connectivity graphs into a connectivity graph for the final composite Euler diagram.

Open Problem 8.3.1. Design an efficient algorithm for constructing a composite Euler diagram given its prime factorization.

Chapter 9

ω-Proportional Euler Diagrams with the Fullset

The initial results presented in this dissertation focused on small cases of area-proportional Euler and Venn diagrams with shape-constrained curves. It was obvious from this initial work that the ω-proportional Euler diagram generation problem (ω-EDGP), is not trivial even when heavily-constrained. Consequently, we proceeded to study the EDGP independently of area-proportionality and developed a graph-theoretic framework to describe its essential properties. As a final component of this dissertation, we return to the original ω-EDGP and consider it with respect to a large class of Euler diagrams, the so-called *monotone* Euler diagrams for which the number of curves is unconstrained.

For the small cases, we generated the Euler diagram's structure at the same time as drawing it area-proportionally. For monotone Euler diagrams, we take a different approach by assuming we are given a non-area-proportional diagram that needs to

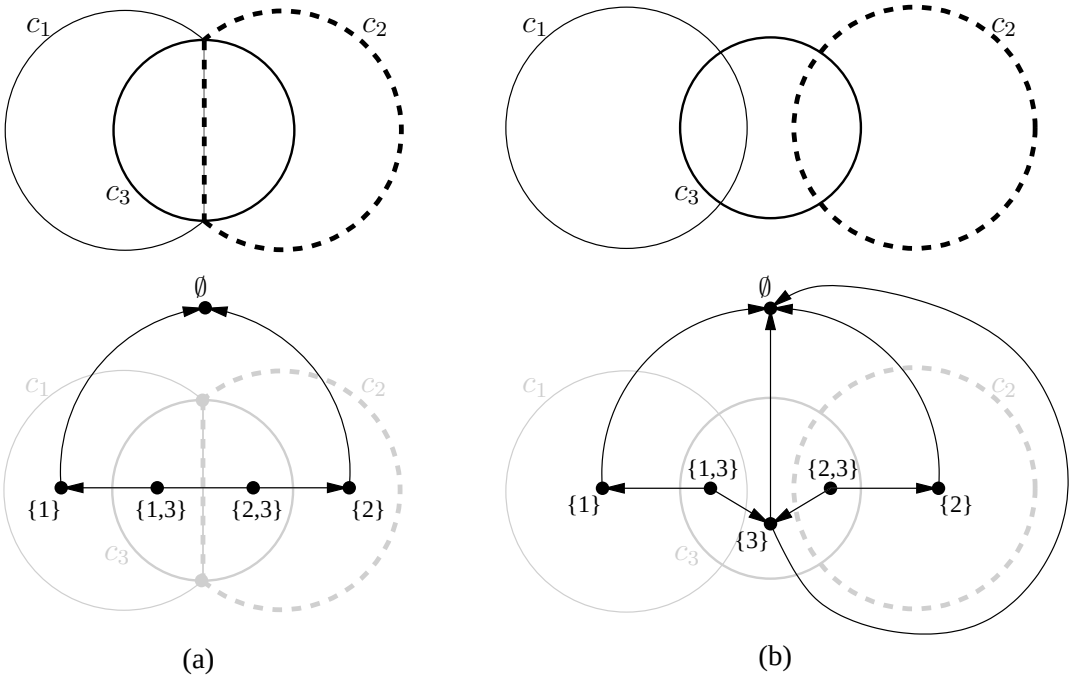


Figure 9.1: (a) A directed Euler dual where the edge $(\{1,3\},\{2,3\})$ is undirected, and (b) a directed Euler dual with two source vertices, $\{1,3\}$, and $\{2,3\}$.

be redrawn area-proportionally. After describing the redrawing algorithm, we will then show how it can actually be used to generate and draw an area-proportional Euler diagram at the same time. We begin by defining monotone Euler diagrams and describing some of their properties.

9.1 Mathematical Background

Definition 9.1.1. Let C be a connected Euler diagram. The *directed Euler dual* of C, denoted $\vec{G}^*(C)$, is a directed version of the Euler dual $G^*(C)$ where an edge $(u,v) \in \vec{G}^*(C)$ is directed from u to v if $u \supset v$; otherwise, it is undirected. □

Example. Figure 9.1(a) shows an example of a directed Euler dual. The edge

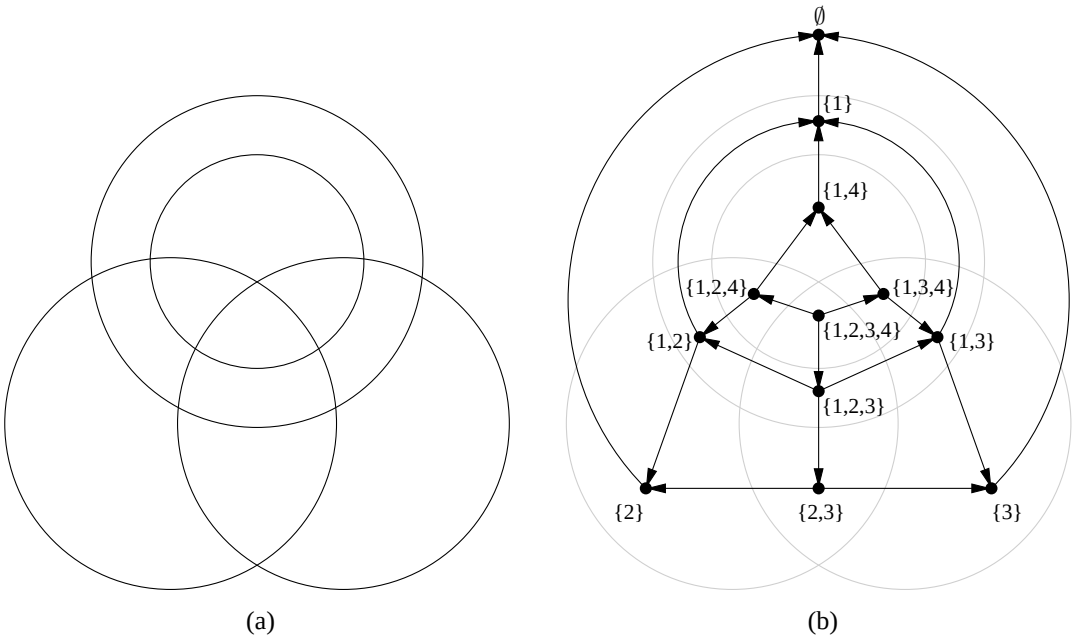


Figure 9.2: (a) A monotone Euler diagram has (b) a directed Euler dual that is a DAG with a single source vertex $\{1, 2, 3, 4\}$ and a single sink vertex $\emptyset$.

$(\{1, 3\}, \{2, 3\})$ is undirected since neither $\{1, 3\} \supset \{2, 3\}$ nor $\{2, 3\} \supset \{1, 3\}$. In contrast, Fig. 9.1(b) shows a directed Euler dual where every edge is directed and where there are two source vertices, $\{1, 3\}$ and $\{2, 3\}$. As shown in both of these examples, the $\emptyset$ vertex is always a sink vertex; however, there may also be additional sink vertices. □

Definition 9.1.2. Let C be a connected Euler diagram. C is said to be *monotone* if its directed Euler dual $\vec{G}^*(C)$ is a directed acyclic graph (DAG) with a single source vertex and a single sink vertex. □

Example. Neither of the Euler diagrams shown in Fig. 9.1 are monotone since, in the first case, the directed Euler dual is not a digraph (since it has an undirected edge), and in the second case, the directed Euler dual has two source vertices. In

contrast, the Euler diagram in Fig. 9.2(a) is monotone since its directed Euler dual, shown in Fig. 9.2(b), has all its edges directed such that it is a acyclic and has a single source vertex $\{1, 2, 3, 4\}$ and a single sink vertex $\emptyset$. □

Let C be a monotone Euler diagram representing the set system S on the items X. Since C is monotone, its directed Euler dual $\vec{G^*}(C)$ has a unique source vertex s and a unique sink vertex t, and since $\vec{G^*}(C)$ is a DAG, there is a directed path from s to every vertex $v \in \vec{G^*}(C), v \neq s$; this implies that v is a subset of s. As a result, s must be the fullset vertex X. Similarly, since there is a directed path from every vertex $v \in \vec{G^*}(C), v \neq t$ to t, t must be the $\emptyset$ vertex.

Proposition 9.1.1. *Let C be a monotone Euler diagram with directed Euler dual $\vec{G^*}(C)$. The fullset vertex is the unique source vertex of $\vec{G^*}(C)$ and the $\emptyset$ vertex is the unique sink vertex of $\vec{G^*}(C)$.*

Suppose we orient the edges of C's Euler graph $G(C)$ so that they specify a clockwise traversal of C's curves (i.e., if an edge $(u, v) \in G(C)$ represents a segment of curve c_i then it is directed so that the exterior of c_i is to the left of (u, v)). Suppose an edge $e \in G(C)$ is bidirectional; this means that neither the region to the left nor the right of e is a superset of the other. Such a case is impossible since this would imply that the corresponding directed dual edge $e^* \in \vec{G^*}(C)$ is undirected, and thus contradicting C's monotone property. The following definition is based on this observation.

Definition 9.1.3. Let C be a monotone Euler diagram. The *directed Euler graph* of C, denoted $\vec{G}(C)$, is a directed version of the Euler graph $G(C)$ where an edge

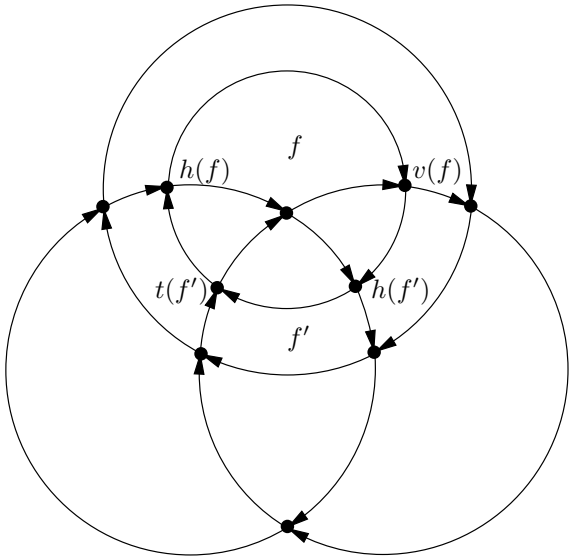


Figure 9.3: The directed Euler graph corresponding to the monotone Euler diagram in Fig. 9.2(a). Note how each face comprises two directed paths which start at the same vertex and end at the same vertex.

$(u, v) \in \vec{G}(C)$ is directed from u to v if the region to the left of (u, v) is a subset of the region to the right of (u, v); otherwise, it is directed from v to u. □

Example. Figure 9.3 shows the directed Euler graph $\vec{G}(C)$ for the monotone Euler diagram C in Fig. 9.2(a). Referring to C's directed Euler dual $\vec{G^*}(C)$, note how each directed Euler dual edge crosses from right to the left side of its corresponding directed Euler graph edge; this is a consequence of the definitions. □

The following proposition is a direct result of considering Lemma 1.1 of Bultena et al. [4] in terms of the directed Euler graph.

Proposition 9.1.2. *Let C be a monotone Euler diagram with directed Euler graph $\vec{G}(C)$. Each face $f \in \vec{G}(C)$ has a boundary δf comprising two directed paths that start at the same vertex, $h(f)$ (for head), and end at the same vertex, $v(f)$ (for tail).* □

Example. Consider face f of the directed Euler graph $\vec{G}(C)$ shown in Fig. 9.3, and note the two paths comprising its boundary. The vertex at the start of the paths is denoted $h(f)$ and the vertex at the end of the paths is denoted $t(f)$. The same property holds for f' and, in fact, all faces of $\vec{G}(C)$. □

The final property of monotone Euler diagrams that we need to consider before commencing with the drawing algorithm, relates a path in the directed Euler dual to an edge cut in the directed Euler graph that renders the resulting directed graph acyclic.

Proposition 9.1.3. *Let C be a monotone Euler diagram with directed Euler graph $\vec{G}(C)$ and directed Euler dual $\vec{G^*}(C)$. If p^* is any path in $\vec{G^*}(C)$ between the source vertex and the sink vertex, then $\vec{G}(C)\backslash p$ is a directed acyclic graph (DAG), where p is the set of $\vec{G}(C)$'s edges corresponding to p^*.*

Proof. This is an application of Prop. A.9.1, pg. 304, where $\vec{G} = G^*\vec{(}C)$ and $\vec{G^*} = \vec{G}(C)$. □

Example. Figures 9.4(a) and (b) show the directed Euler dual $\vec{G^*}(C)$ and the directed Euler graph $\vec{G}(C)$ for the monotone Euler diagram C in Fig. 9.2. A path p^* between $\vec{G^*}(C)$'s source vertex and sink vertex is indicated by the dashed lines, and the corresponding directed Euler graph edges p are grayed-out in Fig. 9.4(b). Note how the remaining directed Euler graph edges (black) represent a DAG; that is, p^* cuts each of directed cycle in $\vec{G}(C)$. □

Now that we've defined monotone Euler diagrams and the relevant properties of their directed Euler graphs/duals, we can describe how to construct an ω-proportional drawing of a monotone Euler diagram.

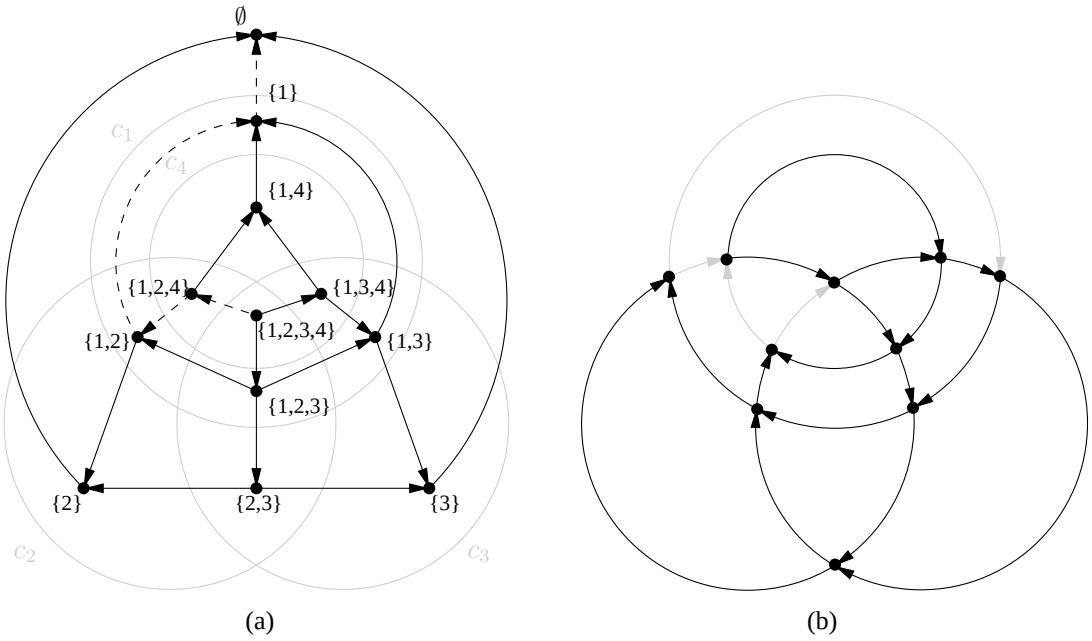


Figure 9.4: (a) Any path (dashed edges) between the source vertex and the sink vertex of a directed Euler dual corresponds to (b) an edge cut in the directed Euler graph that results in a DAG (black edges).

9.2 ω-proportional Drawings of Monotone Euler Diagrams

Before presenting a formal description and complexity analysis of the ω-proportional monotone Euler diagram drawing algorithm, we demonstrate its main steps through an example. Let C be the monotone Euler diagram shown in Fig. 9.2, and let ω be some weight function for the set system represented by C. Note that ω need not be the actual weight function represented by C; instead, it is the *desired* weight function.

The algorithm begins by choosing an arbitrary path p^* in C's directed Euler dual $\vec{G}^*(C)$ and removing the corresponding edges from C's directed Euler graph $\vec{G}(C)$. Let $\vec{G}$ be the resulting DAG per Prop. 9.1.3. For this example, suppose $\vec{G}$ is the DAG shown in Fig. 9.4(b).

Next, since $\vec{G}$ is a DAG, its vertices can be topologically ordered as shown in Fig. 9.5(a). If there are k vertices, then an arbitrary point is chosen in the plane from which emanates k equally-spaced rays. Each of $\vec{G}$'s vertices is assigned to a unique ray and the rays are ordered clockwise according to the topological order as shown in Fig. 9.5(b).

C is now redrawn region-by-region according to a topological ordering of the directed Euler dual $\vec{G}^*(C)$'s vertices, which represent C's regions. Figure 9.6(a) shows C's directed Euler graph $\vec{G}(C)$ with its vertices labeled numerically based on a topological ordering of $\vec{G}$ and its faces labeled alphabetically based on a topological ordering of $\vec{G}^*(C)$.

The fullset $\{1, 2, 3, 4\}$ is always the first region drawn, and it is represented by a regular k-gon with area $\omega(\{1, 2, 3, 4\})$ as shown in Fig. 9.6(b). Face b corresponds to

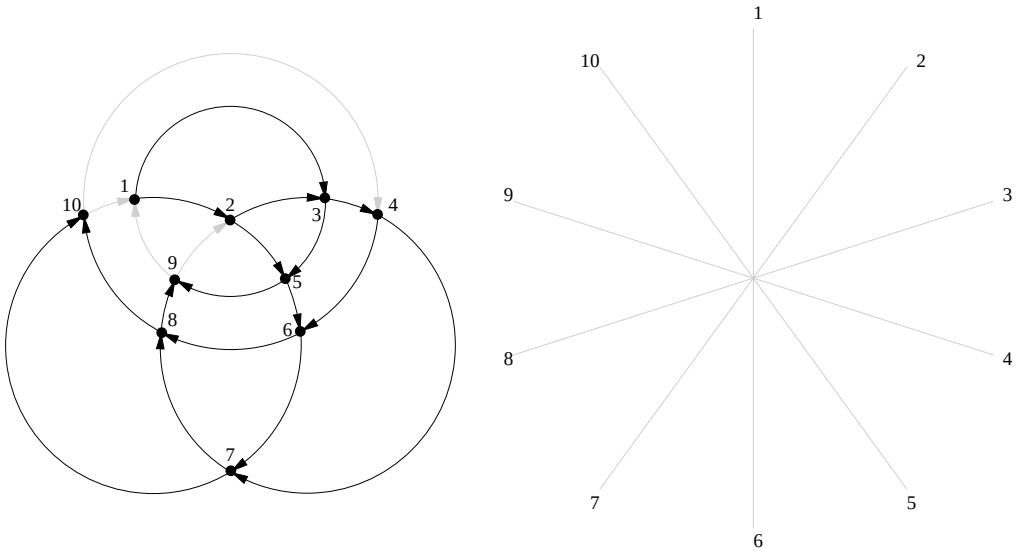


Figure 9.5: (a) Since $\vec{G}$ is a DAG, its vertices can be topologically ordered as indicated by the labels, and (b) associated with a clockwise-ordered set of rays.

region $\{1, 2, 4\}$ and is the next to be drawn. As described in Prop. 9.1.2, b's boundary comprises two directed paths: $9 \to 1 \to 2$ and $9 \to 2$. Since path $9 \to 2$ is part of the boundary of region $\{1, 2, 3, 4\}$ that has already been drawn, its position is *fixed*, while path $9 \to 1 \to 2$ has not been drawn, so its position is *free*. Path $9 \to 1 \to 2$ is drawn clockwise beginning at the ray corresponding to vertex 9 and ending at the ray corresponding to vertex 2 as shown in Fig. 9.6(c), and expands outward along the rays in order to create a face with area $\omega(\{1, 2, 4\})$. The free path may expand outward in whichever fashion is desired; in this example, it expands uniformly along the intermediate rays. In addition, as is the case with this example, the path may cross additional rays representing vertices not specified in the path; this is fine.

Figure 9.6(d) shows the next face, c, being drawn, and Fig. 9.6(e) shows the final redrawing of C with the faces alphabetically ordered according to the sequence in

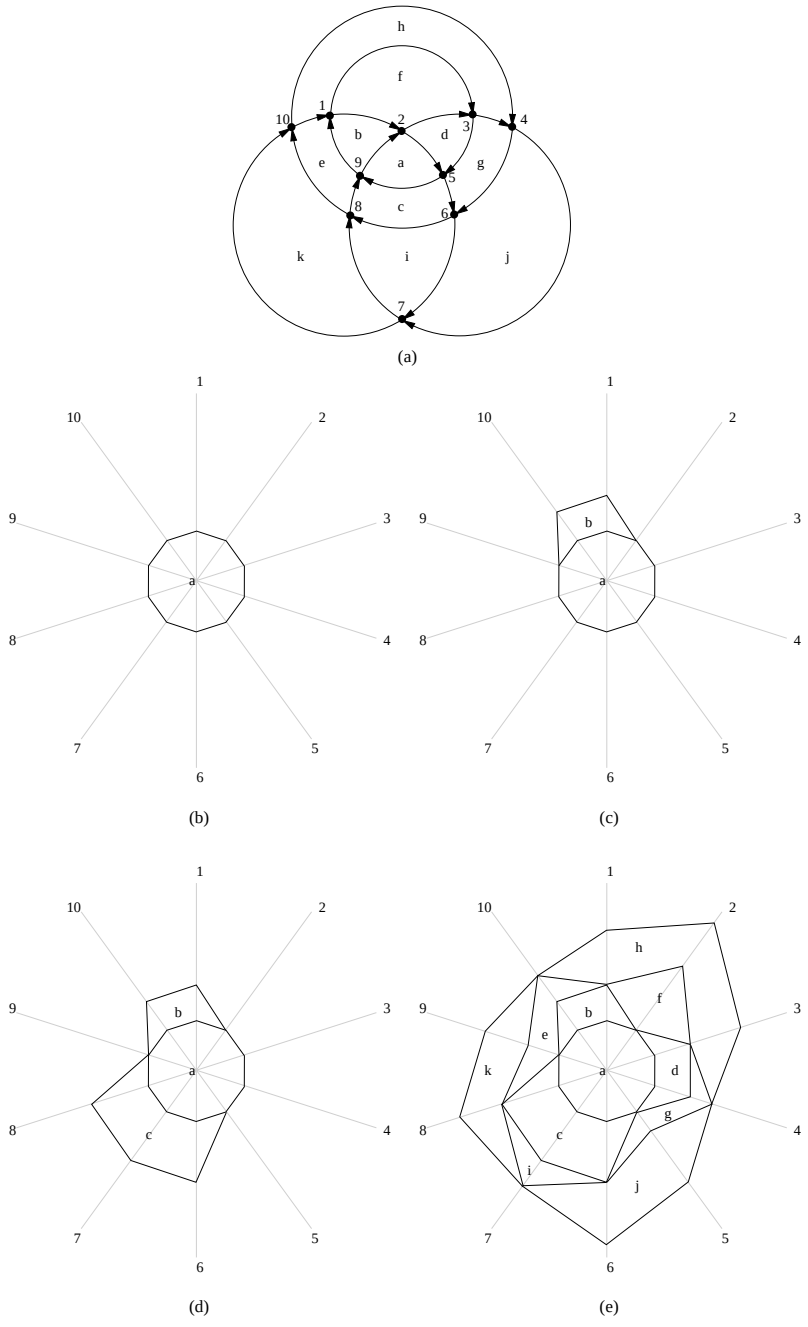


Figure 9.6: (a) The directed Euler graph $\vec{G}(C)$ with its vertices numerically labeled according to a topological ordering of $\vec{G}$ and its faces alphabetically labeled according to a topological ordering of $\vec{G}^*(C)$. (b)–(e) the successive steps of the redrawing algorithm.

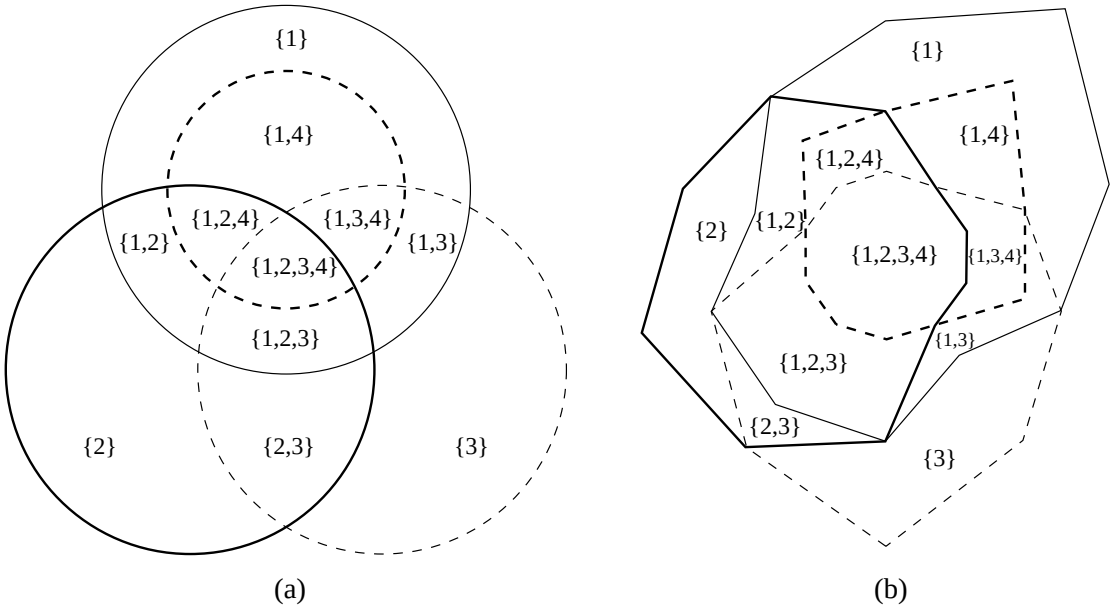


Figure 9.7: (a) The original monotone Euler diagram and (b) its redrawing as an ω-proportional Euler diagram.

which they were drawn. Figure 9.7 shows the original monotone Euler diagram C with its regions labeled along with the ω-proportional redrawing.

In general, because $\vec{G}(C)$'s faces are topologically ordered, as each face is drawn, one of its boundary paths will be fixed and the other will be free; it is the free path that allows the face's area to be set to whatever value is required by ω. The drawing algorithm can produce faces with arbitrarily large areas because the topological ordering ensures that the free path can expand outwards indefinitely; If this were not the case and a previously-drawn face with a path further out along the rays, since all paths are clockwise-oriented, this would imply the existence of a directed Euler dual edge from the free path's face to the existing face thus violating the topological ordering as shown in Fig. 9.8 (the arrows are added to emphasize the paths' orientation).

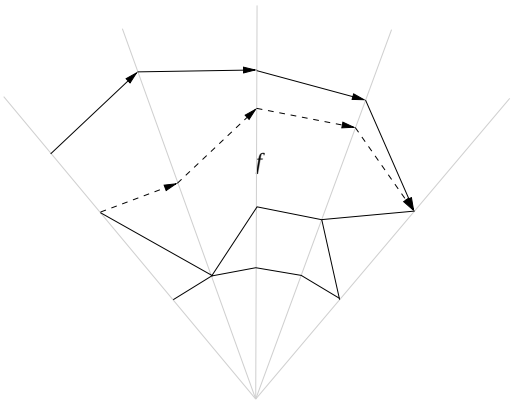


Figure 9.8: The free path (dashed arrows) of the face f currently being drawn cannot be bounded by another path (solid arrows) further out along the rays because this would imply that the faces were not drawn according to a topological ordering of the directed Euler dual.

Algorithm 3 formally specifies the area-proportional monotone Euler diagram redrawing algorithm. To make the diagram more compact, Alg. 3 layers the minimal vertices at each step of the topological ordering of $\vec{G}$ on a single ray. In addition, Alg. 3 inserts a dummy ray between each topological ray so that faces whose free path has only a single edge are able to expand outwards (since $h(f)$ and $h(t)$ are fixed). Lastly, Alg. 3 doesn't actually force the vertices of each face's polygon to intersect a ray; instead, it maintains a clockwise ordering $p_1, p_2, \ldots, p_{2k}$ of the outermost points of the current state of the drawing. The complexity of Alg. 3 is analyzed in the following theorem.

Theorem 9.2.1. *Algorithm 3 is $O(n|V|)$ for a monotone n-Euler diagram with a directed Euler graph $\vec{G}(C) = (V, E)$.*

Proof. We assume $\vec{G}(C)$ is represented using the Winged-Edge data structure [2], which provides $O(1)$ operations for inserting and removing edges, retrieving an arbitrary edge adjacent to a vertex or face, retrieving the endpoints of an edge, retrieving

Algorithm 3 Area-proportionally redraw a monotone Euler diagram.

Require: A monotone Euler diagram C and desired weight function ω.
Ensure: An ω-proportional Euler diagram C' that is topologically equivalent to C.

$\vec{G}(C) \leftarrow C$'s directed Euler graph
$\vec{G^*}(C) \leftarrow C$'s directed Euler dual
$s \leftarrow$ the unique source vertex of $\vec{G^*}(C)$
$t \leftarrow$ the unique sink vertex of $\vec{G^*}(C)$
{Compute $h(f)$ and $t(f)$ for each face f of $\vec{G}(C)$.}
for all faces $f \in \vec{G}(C)$ **do**
    $h(f), t(f) \leftarrow$ the source,sink of the two paths comprising δf
{Construct the acyclic subgraph $\vec{G} \subset \vec{G}(C)$.}
$\vec{G} \leftarrow \vec{G}(C)$
$p^* \leftarrow$ any path from s to t
for all edges $e^* \in p^*$ **do**
    remove the corresponding edge e from $\vec{G}$
{Perform a layered topological sort of $\vec{G}$.}
$k \leftarrow 0$
$L_1 \leftarrow$ source vertices of $\vec{G}$
while $|L_{k+1}| > 0$ **do**
    $k \leftarrow k + 1$
    for all vertices $v \in L_k$ **do**
        remove v and its incident edges from $\vec{G}$
    $L_{k+1} \leftarrow$ source vertices of $\vec{G}$
{Redraw the faces of $\vec{G}(C)$ ω-proportionally.}
$x \leftarrow$ an arbitrary point in the plane
$p_1, p_2, \ldots, p_{2k} \leftarrow$ the clockwise points of a regular $2k$-gon centered at x and with area $\omega(s)$
Draw polygon $p_1, p_2, \ldots, p_{2k}$
for all topologically-ordered vertices $f \in \vec{G^*}(C) \backslash \{s, t\}$ **do**
    $i, j \leftarrow$ indices where $t(f) \in L_{\lfloor (i+1)/2 \rfloor}, h(f) \in L_{\lfloor (j+1)/2 \rfloor}$
    {NOTE: points are indexed, modulo $2k$, beginning at p_1.}
    Draw polygon $p_i, p'_{i+1}, \ldots, p'_{j-1}, p_j, p_{j-1}, \ldots, p_{i+1}$ where these are the clockwise points of a polygon with area $\omega(f)$ that is contained within the sector defined by x, p_i, p_j and does not overlap any existing polygons
    $p_{i+1}, p_{i+2}, \ldots, p_{j-1} \leftarrow p'_{i+1}, p'_{i+2}, \ldots, p'_{j-1}$

the left and right faces of an edge, and retrieving the predecessor and successor edges about either of an edge's adjacent faces.

Let $\vec{G}(C) = (V, E)$ and let F be its faces.

Lines 1–4. The directed Euler dual is created by traversing the edges of each face of $\vec{G}(C)$ and inserting a corresponding dual edge about a new vertex in $\vec{G^*}C$; since each edge is visited twice, this is an $O(|F| + |E|)$ construction. s and t can be kept track of during the construction.

Lines 5–7. Similarly, the edges of each face in $\vec{G}(C)$ are traversed and the vertices where two consecutive edges are head-to-head or tail-to-tail is recorded, so this is an $O(|F| + |E|)$ operation.

Line 9. The Winged-edge data structure comprises fixed-length objects representing each vertex, edge, and face, so copying $\vec{G}(C)$ is an $O(|V| + |E| + |F|)$ operation.

Line 10. In constructing p^*, an edge directed out of the current vertex must be found by searching all the incident edges. Since $\vec{G^*}(C)$ is acyclic, each edge is visited at most once, so this is an $O(|E|)$ construction.

Lines 11–12. Since p^* is a directed path of subsets beginning at the fullset and ending at the $\emptyset$, it visits at most $n - 1$ edges. Since edge removal is $O(1)$, these steps form an $O(n)$ operation.

Lines 13–20. A topological sort is an $O(|V| + |E|)$ operation.

Lines 21–29. We assume that computing a polygon involves solving a piecewise function, the size of which depends on the number of spanned rays. Because of the curves are drawn clockwise about x, each ray intersects at most n curve segments and therefore at most n polygons. Since there are at most $2|V|$ rays, the number of equations that must be solved is at most $2|V|n$. If each equation is solved in constant

time, which is the case for the uniform polygon growth described, these steps form an $O(n|V|)$ operation.

The overall complexity is $O(|V| + |E| + |F|) + O(n) + O(n|V|)$. Euler's Formula linearly relates $|V|$,$|E|$, and $|F|$, so this simplifies to $O(n|V|)$ for $n \geq 2$. □

Given a monotone Euler diagram C represented by appropriate data structures, Alg. 3 can efficiently redraw C so that it is ω-proportional for any weight function ω. Algorithm 3 solves the problem of drawing monotone Euler diagram area-proportionally, but not the structure generation step. In the next section, we consider how, without any changes, Alg. 3 can be used to *generate* a monotone Euler diagram.

9.3 Generating Monotone Euler Diagrams

Reconsider the steps of Alg. 3 shown in Fig. 9.7. Suppose in Fig. 9.7(d), when face c is being drawn, instead of $\omega(c) > 0$ (as is normally required by a weight function), we have $\omega(c) = 0$. We'd expect face c's free path to have no expansion outwards from its fixed path as shown in Fig. 9.9(b). Suppose the algorithm continues and the same situation occurs for face h (i.e., $\omega(h) = 0$). Figure 9.9(c) shows the resulting diagram C', and the individual curves are highlighted in Fig. 9.10(b). Compare C' to the original monotone Euler diagram C in Fig. 9.10(a). Since C' is missing regions $\{1\}$ and $\{1, 2, 3\}$, it is not topologically equivalent to C; however, it is still a monotone Euler diagram. The only way for C' to *not* be an Euler diagram is if, when a face's free path collapses onto its fixed path, a curve self-intersects; however, as long as the central fullset face is not collapse, this is impossible due to the curves' clockwise orientation about the central point x.

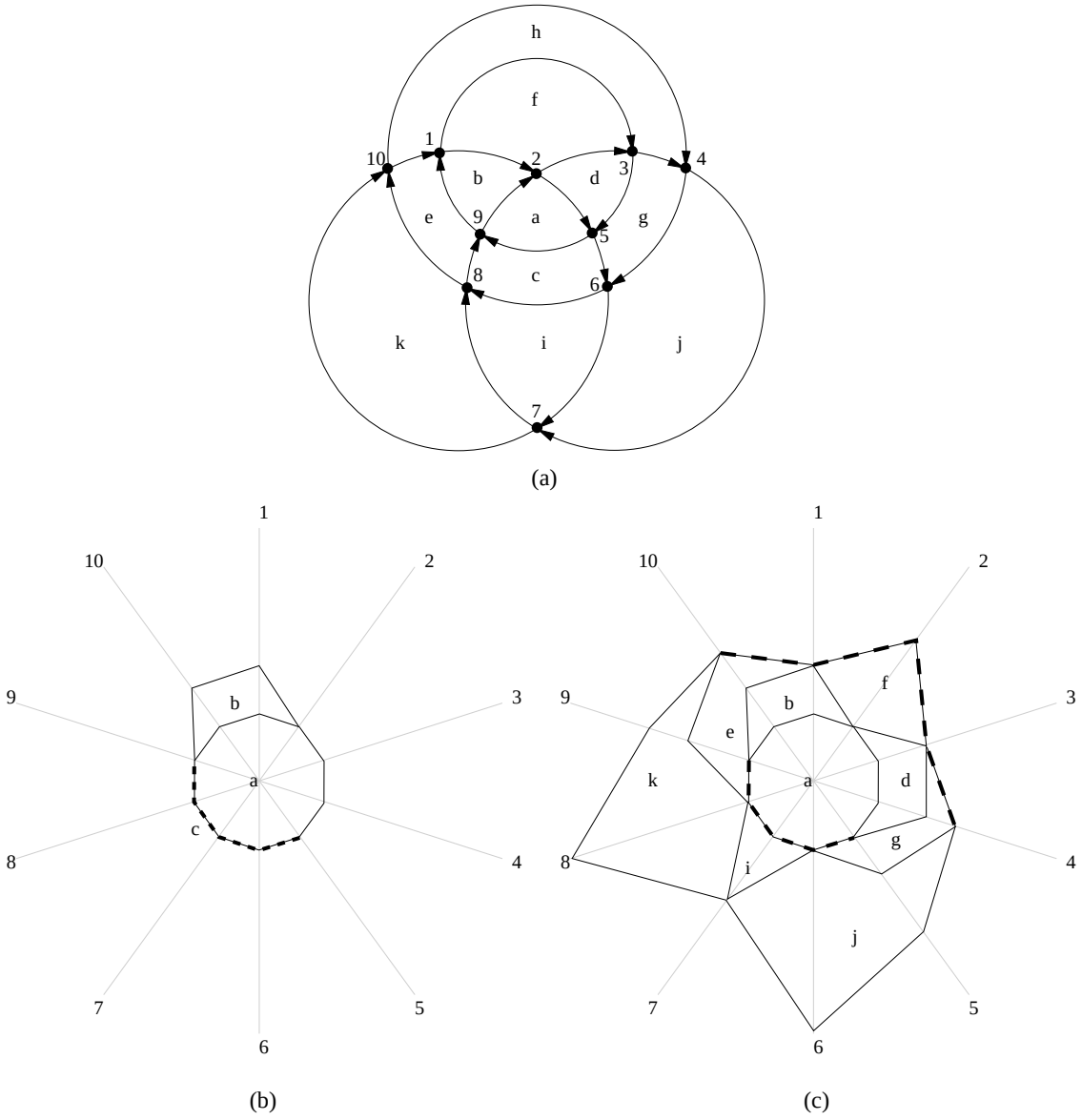


Figure 9.9: (a) The same directed Euler graph from Fig. 9.6(a). (b) When face c is drawn, the free path (dashed) follows the existing profile creating a concurrent curve segment, and (c) the final diagram where faces c and h are empty.

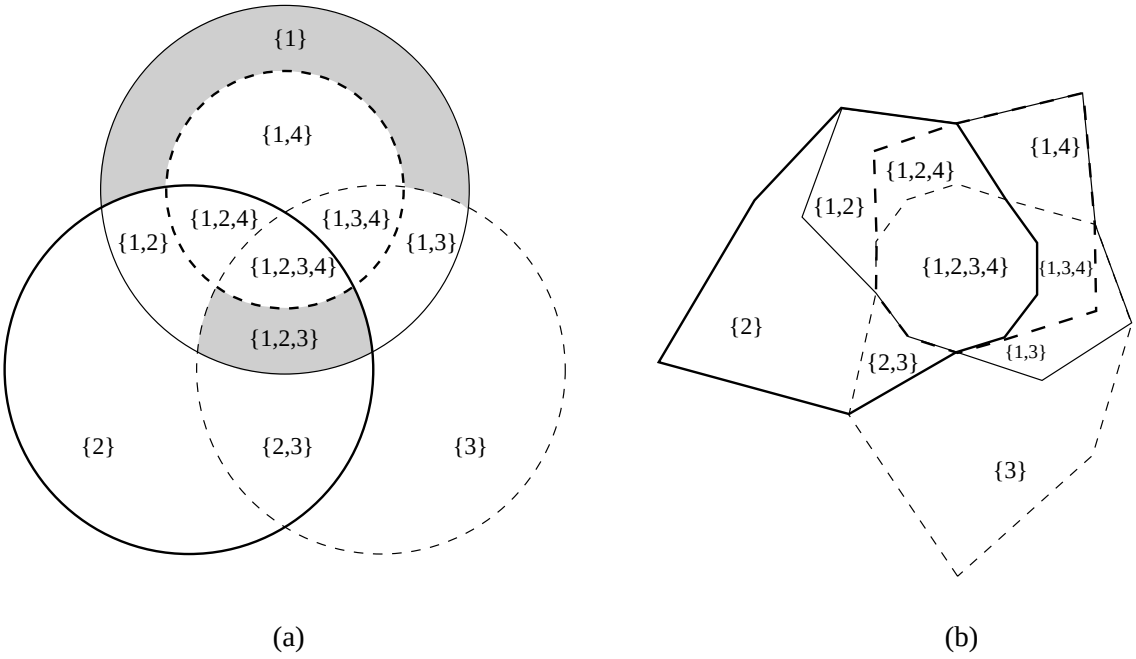


Figure 9.10: (a) The original monotone Euler diagram and (b) its redrawing as an ω-proportional Euler diagram where regions $\{1\}$ and $\{1, 2, 3\}$ are removed.

As a result of this observation, given a monotone Euler diagram C, Alg. 3 can be used to remove *any* non-fullset region r from C just be setting $\omega(r) = 0$. Of course, an initial monotone Euler diagram is still required; fortunately, this is not a problem. As described in Section 2.1, Edwards [10] developed an algorithm that can generate an n-Venn diagram for any $n > 0$. It turns out that the Venn diagrams produced by Edwards' construction are monotone.

So, given a set system S on the items X with $X \in S$ (i.e., S contains the fullset), and a weight function ω for S, we can construct an ω-proportional Euler diagram representing S via the following steps:

1. Use Edwards construction to generate a monotone $|X|$-Venn diagram C,
2. Construct a weight function $\omega' : \mathcal{PS}(X) \to \mathbb{R}$ such that $\omega'(s) = \omega(s)$ for $s \in S$ and $\omega'(s) = 0$ for $s \in \mathcal{PS}(X) \backslash S$,

3. Use Alg. 3 to redraw C so that it is ω'-proportional.

Since a set s that is in $\mathcal{PS}(X)$ but not in S has $\omega'(s) = 0$, its corresponding region in C be removed; as a result, the final Euler diagram C' will represent S and each non-empty region r will have area $\omega'(r) = \omega(r)$. If one is interested in generating an Euler diagram irrespective of area-proportionality, then ω can be arbitrarily chosen. This process for generating an Euler diagram from Edwards' Venn diagram construction leads directly to the following theorem.

Theorem 9.3.1. *Let S be a set system on the items X and ω be a weight function for S. There exists an ω-proportional Euler diagram representing S if $X \in S$.*

As we will discuss in the next chapter, although the conditions of Thm. 9.3.1 are sufficient, they are not necessary for the existence of an ω-proportional Euler diagram. There are also many parameters that can be adjusted in Alg. 3, and we will discuss some of these as well.

Chapter 10

Conclusion and Implementations

When we began research for this dissertation, our original intent was to focus on area-proportional drawings of Euler and Venn diagrams. We envisioned developing several drawing algorithms that we would then use to study the properties, both aesthetic and functional, that make a "good" drawing. As we saw in Chapter 3, even for small cases (i.e., up to three curves), generating and drawing an area-proportional Euler diagram can be challenging, particularly when the curves are shape-constrained. One of the significant results from this chapter is Thm. 3.4.2, which states that, regardless of how complex the curve shapes are allowed to become, as long as convexity is required, there are some weight functions for which there is no ω-proportional 3-Venn diagram.

Realizing that Euler diagram generation irrespective of area-proportionality is a simpler problem (and one which, in most cases, needs to be solved prior to area-proportional drawing), in Chapters 4 and 5 we developed a graph-theoretic model for Euler diagrams along with necessary-and-sufficient existence conditions. The proofs of the existence conditions are constructive, and so provide an algorithm for solving

the Euler Diagram Generation Problem (EDGP). For example, given a set system S, one can search for a non-concurrent Euler diagram representing S by considering all planar subgraphs of the closeness graph $G_c(S)$ that meet certain connectivity conditions. If an exhaustive search cannot find a compliant subgraph, then no such Euler diagram exists; otherwise, the subgraph's plane dual can be constructed (in the form of a combinatorial embedding), and then drawn using existing planar graph drawing algorithms [38] to produce the Euler diagram.

In Chapter 6, we employed the existence conditions to better understand the effect that certain constraints (e.g., non-concurrency or pairwise intersections), can have on the set systems that are representable by Euler diagrams. We discovered that in certain combinations, some constraints do not affect the diagrams' expressiveness (i.e., all set systems that are representable by diagrams conforming to constraint set A are representable by diagrams conforming to constraint set B), while in other combinations, the constraints were meaningful in terms of differentiating representable set systems. The main result of this chapter was a collection of diagrams showing canonical examples of Euler diagrams that separate the meaningful constraint combinations. Since there are exponentially many pairs of constraint sets to compare, we limited our study to pairs of constraint sets that were related by a subset structure (i.e., we only compared constraint sets A and B for which either $A \subset B$ or $B \subset A$); even within this framework, the results are interesting and demonstrate the usefulness of the existence conditions from Chapter 5. We leave further research into other combinations of constraints as an open problem.

As was evident from our exploration of small cases in Chapter 3, and suggested by previous complexity analysis of related problems (e.g., hypergraph-planarity [33] and

Euler-like diagram generation [47]), in Chapter 7 we proved that, in its most general form, the Euler Diagram Decision Problem (EDDP) is NP-complete, and thus, so is the EDGP. We leave as open problems the analysis of restricted versions of the EDGP (e.g., when limited to non-concurrent or simple Euler diagrams); however, our suspicion is that these are NP-complete as well.

Since the EDGP is NP-complete, efficient solutions will either need to be approximate (e.g., generating additional unwanted regions or missing some desired regions), or heuristic. In Chapter 8, we expanded Flower et al.'s [15] research into nested Euler diagrams, which was limited to simple Euler diagrams with transverse intersections, by considering a similar, but broader, nesting structure within the context of general Euler diagrams. These so-called composite Euler diagrams have a natural hierarchical structure, which can be used to decompose the EDGP into smaller, and hopefully tractable, subproblems. We developed an efficient algorithm for decomposing a set system into smaller set systems and showed how this strategy could be used, in combination with a deterministic EDDP algorithm, to heuristically solve the EDDP for set systems that might normally be too large to be tractable. We also described how this approach could also be applied to heuristically solve the EDGP, but left the algorithmic details for merging Euler duals as future work.

Finally, after an in-depth study of the EDGP, we returned to the ω-EDGP. In Chapter 9, we extended Bultena et al.'s [4] work into convex drawings of families of intersecting simple closed curves (FISCs) and developed Alg. 3 for redrawing *any* monotone Euler diagram so that it is ω-proportional for an arbitrary weight function ω. Some of the results of this chapter have combinatorial proofs that complement the continuous plane geometry proofs of [4]. In particular, the combination of Prop.

9.1.3 and Alg. 3 provides a discrete graph-theoretic proof to Lemma 1.3 of [4], "If a FISC is monotone, then it is isomorphic to a ray-monotone FISC."

In contrast to the general pattern of generating an Euler diagram's structure before area-proportionally drawing it, we also showed that Alg. 3 can be used to generate an Euler diagram, either area-proportionally or irrespective of area, by first generating a monotone Venn diagram (e.g., using Edwards' construction [10]), and then setting $\omega(r) = 0$ for each unwanted region r. This culminated in a constructive proof of Thm. 9.3.1, which states that for *any* set system S containing the fullset and *any* weight function ω, there is an ω-proportional monotone Euler diagram representing S. Although this theorem is not surprising since one can trivially construct such a diagram by placing n equivalent curves in the shape of a regular 2^n-gon and "pulling" out the necessary regions from each edge of the polygon, the diagrams that result from Alg. 3 are likely to have considerably fewer concurrent curve segments. In the next section, we talk about future work, which includes issues related to the drawings produced by Alg. 3.

10.1 Implementations

As part of our research, we have also implemented the area-proportional drawing algorithms described in Chapters 3 and 9 in several Java applications. Figure 10.1 shows a screen shot of the Euler Venn Diagrams (EVD) application; this program allows multiple area-proportional Euler diagrams to be created on a single page. The two circle diagram demonstrates how shade and colour are combined with labels to create a nicely-formatted diagram. The three circle diagram demonstrates how a

monochrome image can be created for print publication, as well as how individual regions can be highlighted. Figure 10.2 shows a screen shot of the EVD's implementation of the three rectangle algorithm from Section 3.2; in this example, the top-left diagram has been "fanned-out". The fanout option creates three additional diagrams, each highlighting one of the curves; the diagrams are linked to the original diagram, so a change made to one diagram is propagated to the rest. Although the present software is an experimental test-bed for our research, it has been used by other researchers to produce diagrams for presentations, posters, and articles. For example, Artes and Chauhan [1, Fig. 7] used a previous version of the program to create area-proportional Venn diagrams in order to display the results of their study on the conformance of three glaucoma tests.

We have also implemented Alg. 3 for generating and drawing area-proportional monotone Euler diagrams in a program called DrawEuler. Upon starting DrawEuler, the user enters the number n of curves to create, and an initial n-Venn diagram based on Edwards' construction is drawn so that each region has equal area. The user can click on a region and dynamically adjust its area or remove it by setting the area to 0. In addition to adjusting region areas, the user can select several polygon growth algorithms and rearrange the rays. For example, the first DrawEuler screen shot in Fig. 10.3 shows an Euler diagram representing $S = \{\emptyset, \{1\}, \{1,2\}, \{1,2,3\}\}$; note that the rays are uniformly-spaced and the curves are non-convex. In the second screen shot, the rays have been redistributed non-uniformly so that the resulting curves are now convex.

An interesting observation, which we made upon implementing Alg. 3, is shown in Fig. 10.4. In this sequence of DrawEuler screen shots, the fullset region is made

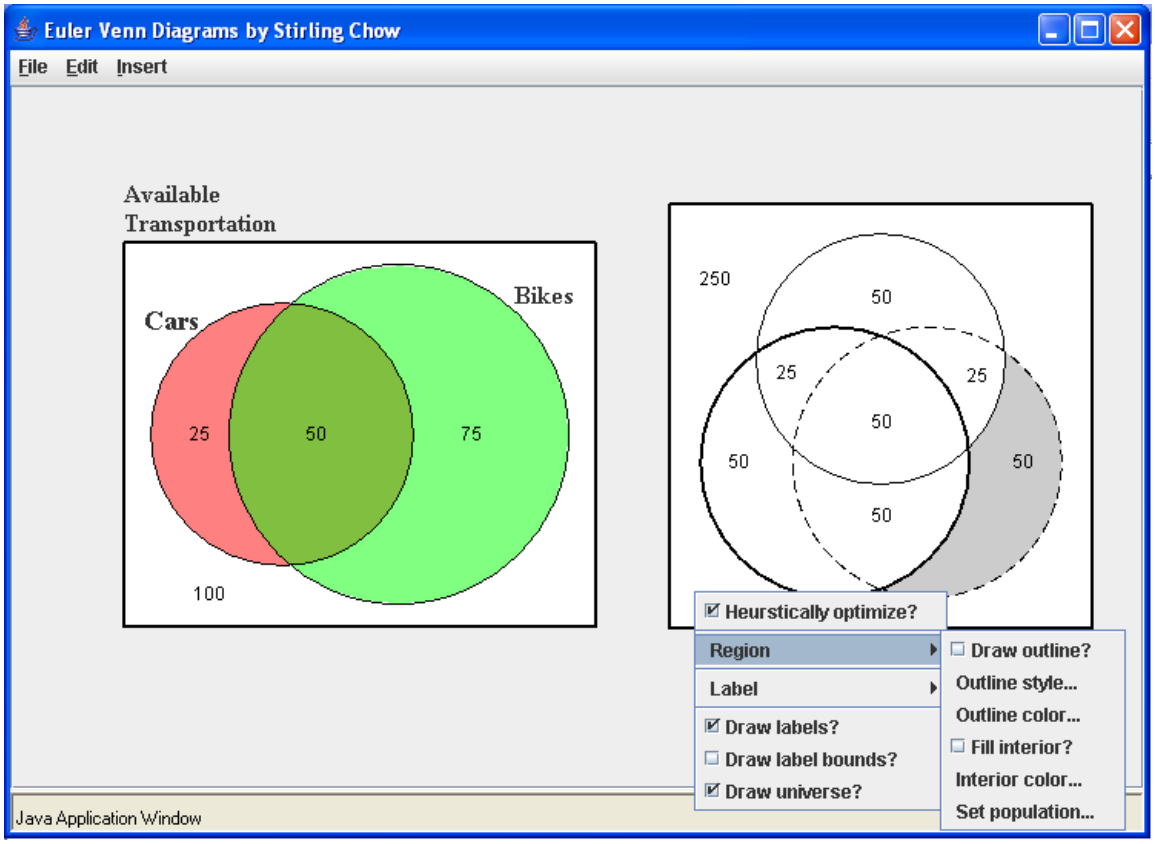


Figure 10.1: A screen shot of the implementation of the two circle and three circle ω-proportional drawing algorithms.

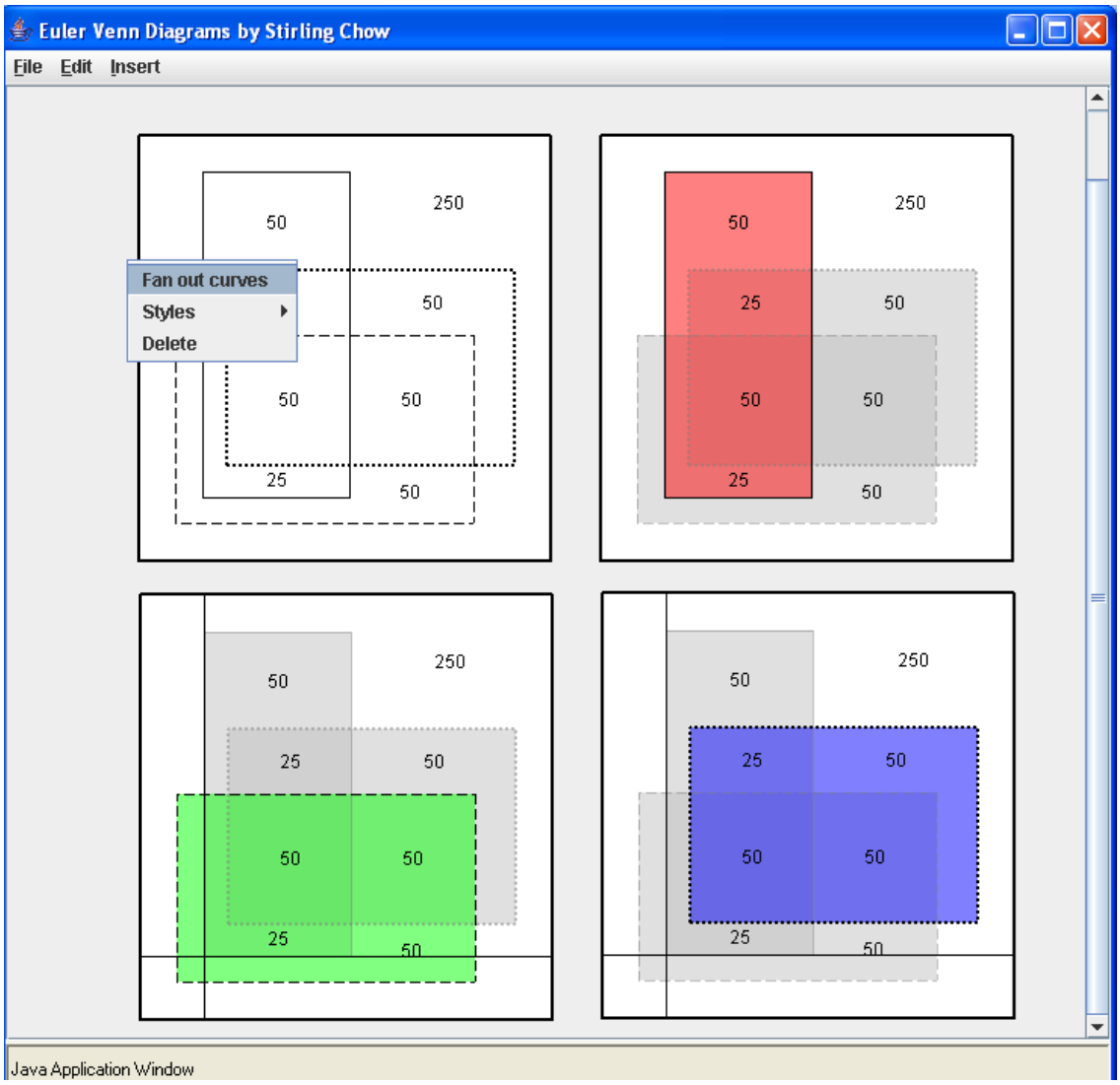


Figure 10.2: A screen shot of the implementation of the three rectangle algorithm where the "fanout" option is selected; this creates three additional linked diagrams that highlight each individual curve.

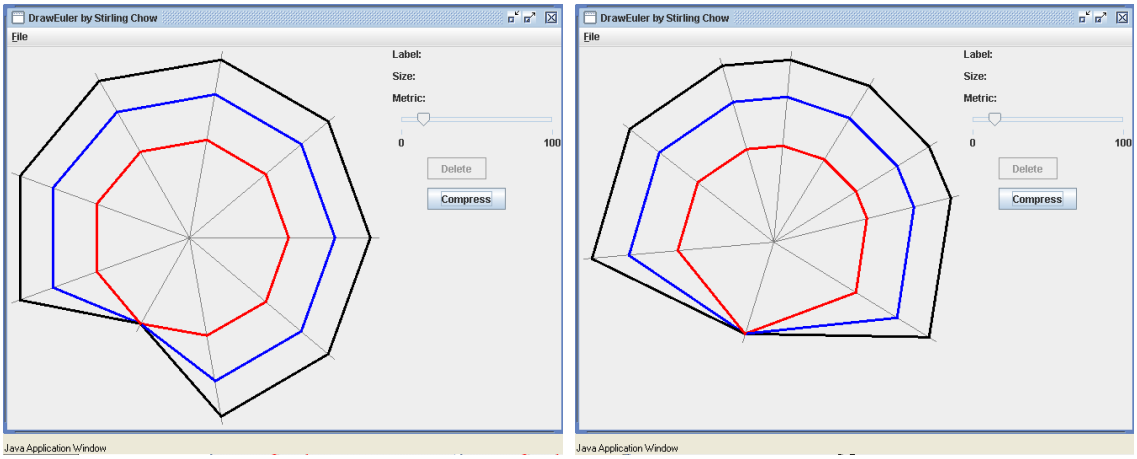


Figure 10.3: A screen shot of an Euler diagram representing $S = \{\emptyset, \{1\}, \{1,2\}, \{1,2,3\}\}$, followed by a screen shot of the same Euler diagram after the rays are adjusted to make the curves convex.

progressively smaller (i.e., $\omega(\{1,2,3\})$ is decreasing). In the final screen shot, region $\{1,2,3\}$ has been removed, yet the drawing is still a valid area-proportional Euler diagram representing $S = \{\emptyset, \{1\}, \{1,2\}, \{1,3\}, \{2\}, \{2,3\}, \{3\}\}$; however, it is non-monotone since regions $\{1,2\}$, $\{1,3\}$, and $\{2,3\}$ are source vertices in the directed Euler dual. As a result of this example and Thm. 9.3.1, we can see that the presence of the fullset in S is a sufficient, but not necessary, condition for the existence of an area-proportional Euler diagram representing S. That being said, although Alg. 3 can be used to area-proportionally draw *some* non-monotone Euler diagrams, it cannot be used to draw all such diagrams. For example, returning to the last screen shot in Fig. 10.4, any attempt to remove one of the singleton regions results in a self-intersection of the curve enclosing that region, and thus a non-Euler diagram. We leave as an open problem the determination of the conditions under which Alg. 3 can be used to area-proportionally draw *non-monotone* Euler diagrams.

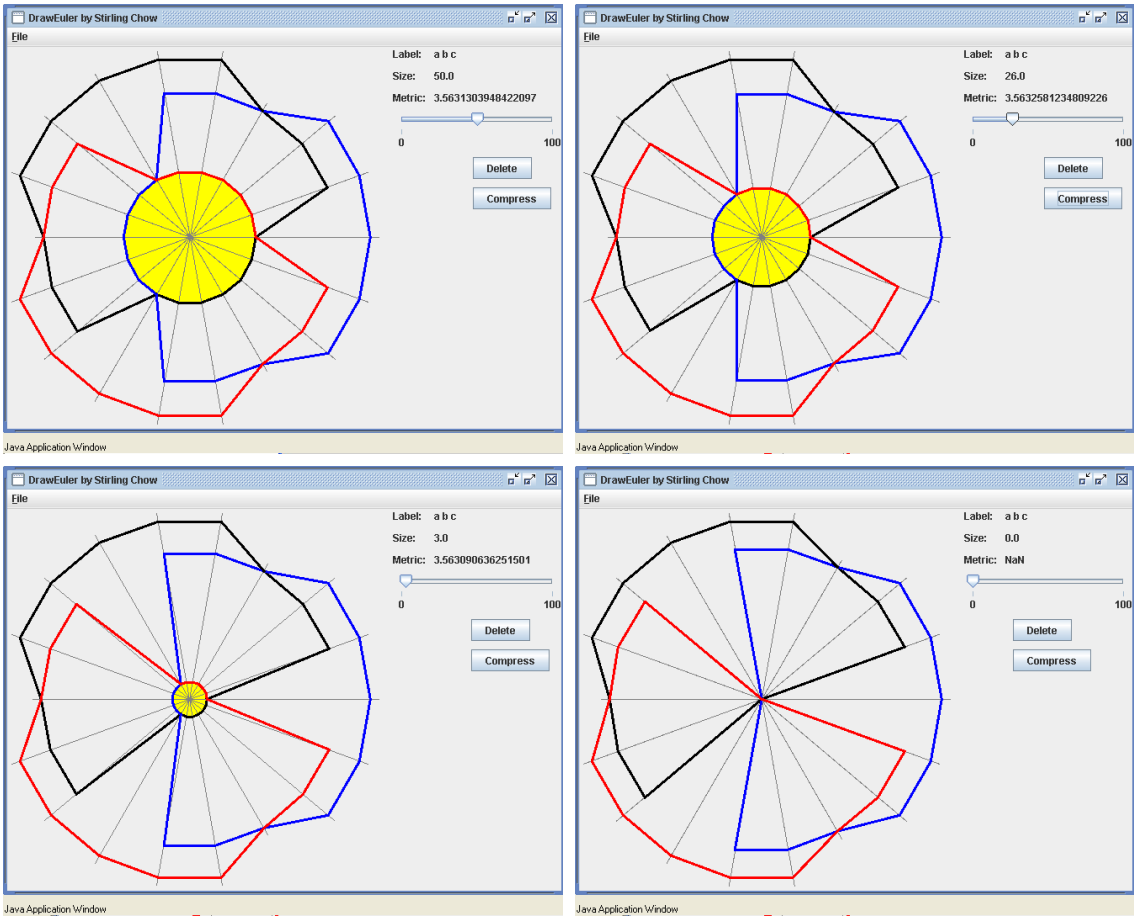


Figure 10.4:]
A sequence of screen shots (left-to-right and top-to-bottom), showing the fullset region being shrunk in a 3-Venn diagram until it disappears; this results in a non-monotone Euler diagram. In the first three screen shots, any of the singleton regions can be removed; however, once the fullset region disappears, this is no longer possible since a curve would self-intersect.

10.2 Additional Future Work

Although our research into generating and drawing area-proportional Euler and Venn diagrams has produced many useful results, there are still open problems and unexplored avenues of research. In addition to addressing the open problems described in the previous sections, we would like to implement a program that heuristically generates Euler diagrams based on the existence conditions of Chapter 5 and the properties of composite Euler diagrams from Chapter 8. Our implementation of drawing algorithms in the EVD and DrawEuler programs has also demonstrated the need for additional research into the aesthetic and functional properties that aid the comprehension of information contained in an area-proportional Euler diagram. The fanout feature of the EVD program is also an example of an area of Euler diagram research that has seen little attention: dynamic Euler diagram manipulation for the purposes of navigation and exploration. Historically, Euler and Venn diagrams were produced for a static print medium, but as electronic information systems continue to proliferate, we see more-and-more applications involving dynamic navigation of data (e.g., data mining); what kinds of interaction can be facilitated by Euler diagrams, whether area-proportional or not, to help explore and understand data? Finally, we would like to continue our development of area-proportional drawing tools, both automated and user-directed; we continue to receive requests for such software from individuals who wish to use Euler and Venn diagrams for data visualization.

As this dissertation has demonstrated, the problem of generating and drawing area-proportional Euler and Venn diagrams is far from trivial and leads to many interesting results and open problems; we hope that there will be continued interest in researching both the theoretical and practical aspects of this topic.

Bibliography

[1] Paul H. Artes and Balwantray C. Chauhan. Longitudinal changes in the visual field and optic disc in glaucoma. *Progress in Retinal and Eye Research*, 23:333–354, May 2005.

[2] B.G. Baumgart. A polyhedral representation for computer vision. *Proc. National Computer Conference*, pages 589–596, 1975.

[3] Florence Benoy and Peter Rodgers. A study into the comprehension of Euler diagrams. Technical Report 14-04, Computing Laboratory, University of Kent, July 2004.

[4] Bette Bultena, Branko Grünbaum, and Frank Ruskey. Convex drawings of intersecting families of simple closed curves. *Proceedings of the* 11^{th} *Canadian Conference on Computational Geometry*, pages 18–21, 1999.

[5] Richard L. Burden and J. Douglas Faires. *Numerical Analysis: 4th Edition*. PWS Publishing Co., 1988.

[6] ESRI Canada. Free Data: Nunavut. `http://k12.esricanada.com/metadata/nunavut.html`.

[7] Stirling Chow and Peter Rodgers. Constructing area-proportional Venn and Euler diagrams with three circles. In *Euler Diagrams Workshop 2005*, August 2005.

[8] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. The MIT Press, 2nd edition, 2001.

[9] Reinhard Diestel. *Graph Theory*, volume 173 of *Graduate Texts in Mathematics*. Springer-Verlag, Heidelberg, July 2005.

[10] Anthony W.F. Edwards. Venn diagrams for many sets. *New Scientist*, 7:51–56, January 1989.

[11] Max J. Egenhofer. Reasoning about binary topological relations. In Oliver Günther and Hans-Jörg Schek, editors, *SSD*, volume 525 of *Lecture Notes in Computer Science*, pages 143–160. Springer, 1991.

[12] G. Ehrlich, S. Even, and R.E. Tarjan. Intersection graphs of curves in the plane. *J. of Comb. Theory Ser. B*, 21:8–20, 1976.

[13] Leonard Euler. *Lettres a une Princesse d'Allemagne sur Divers Sujets de Physique et de Philosophie*. Académie Imperiale des Sciences, St. Petersburg, 1768–1772.

[14] Jean Flower and John Howse. Generating Euler diagrams. In *DIAGRAMS '02: Proceedings of the Second International Conference on Diagrammatic Representation and Inference*, pages 61–75, London, UK, 2002. Springer-Verlag.

[15] Jean Flower, John Howse, and John Taylor. Nesting in euler diagrams: syntax, semantics and construction. *Software and System Modeling*, 3(1):55–67, 2004.

[16] Jean Flower, Peter Rodgers, and Paul Mutton. Layout metrics for Euler diagrams. In *IV '03: Proceedings of the Seventh International Conference on Information Visualization*, page 272, Washington, DC, USA, 2003. IEEE Computer Society.

[17] James Foley, Andries van Dam, Steven Feiner, and John Hughes. *Computer Graphics: Principles and Practice*. Addison-Wesley, 2 edition, 1995.

[18] National Center for Geographic Information and Analysis. Cartogram central. `http://www.ncgia.ucsb.edu/projects/Cartogram_Central/`.

[19] Michael R. Garey and David S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1990.

[20] Michael T. Gastner and M.E.J. Newman. Generating population density-equalizing maps. *Proceedings of the National Academy of Sciences of the United States of America*, 101:7499–7504, 2004.

[21] R.L. Graham. Problem 1. *Open Problems at the* 5^{th} *Hungarian Colloquium on Combinatorics*, 1976.

[22] D. Gries, A. J. Martin, J. L. van de Snepscheut, and J. T. Udding. An algorithm for transitive reduction of an acyclic graph. *Sci. Comput. Program.*, 12(2):151–155, 1989.

[23] Michelangelo Grigni, Dimitris Papadias, and Christos H. Papadimitriou. Topological inference. *Proc. of the* 14^{th} *Intl. Joint Conf. on AI*, pages 901–907, 1995.

[24] Branko Grünbaum. Venn diagrams and independent families of sets. *Mathematics Magazine*, 48:12–23, 1975.

[25] Branko Grünbaum. The construction of venn diagrams. *The College Mathematics Journal*, 15(3):238–247, 1984.

[26] Branko Grünbaum. On venn diagrams and the counting of regions. *The College Mathematics Journal*, 15(5):433–435, 1984.

[27] Branko Grünbaum. Venn diagrams I. *Geombinatorics*, 1(4):5–12, 1992.

[28] Branko Grünbaum. Venn diagrams II. *Geombinatorics*, 2(4):25–32, 1992.

[29] Frank Hardisty. Cartogram Generator. `http://people.cas.sc.edu/hardistf/cartograms/`.

[30] John E. Hopcroft and Robert Endre Tarjan. Efficient planarity testing. *J. of the ACM*, 21(4):549–568, 1974.

[31] Ioan James. *Remarkable Mathematicians: From Euler to von Neumann.* Cambridge University Press, 2002.

[32] Gem Stapleton John Howse and John Taylor. Spider diagrams. *LMS Journal of Computation and Mathematics*, 8:145–194, September 2005.

[33] D.S. Johnson and H.O. Pollak. Hypergraph planarity and the complexity of drawing venn diagrams. *J. Graph Theory*, 10:309–325, 1987.

[34] Peter Hamburger Kiran B. Chilakamarri and Raymond E. Pippert. Venn diagrams and planar graphs. *Geometriae Dedicata*, (62):73–91, 1996.

[35] Jan Kratochvíl and Jiří Matoušek. String graphs requiring exponential representations. *J. Comb. Theory Ser. B*, 53(1):1–4, 1991.

[36] Bert Mendelson. *Introduction to Topology.* Dover, 1990.

[37] Edmondo Morgantini. Su di un problema di Erdös. *Rend. Sem. Mat. Univ. Padova*, (30):245–247, 1960.

[38] Takao Nishizeki and Md. Saidur Rahman. *Planar Graph Drawing.* World Scientific, 2004.

[39] University of Brighton and University of Kent. Reasoning with Diagrams project homepage. http://www.cs.kent.ac.uk/projects/rwd/.

[40] János Pach and Géza Tóth. Recognizing string graphs is decidable. In Petra Mutzel, Michael Jünger, and Sebastian Leipert, editors, *Graph Drawing*, volume 2265 of *Lecture Notes in Computer Science*, pages 247–260. Springer, 2001.

[41] Frank Ruskey and Mark Weston. A survey of venn diagrams. *Electronic Journal of Combinatorics*, Dynamic Survey, 1997 (revised 2001,2005). DS#5.

[42] Marcus Schaefer, Eric Sedgwick, and Daniel Štefankovič. Recognizing string graphs in NP. In *STOC*, pages 1–6. ACM, 2002.

[43] Marcus Schaefer and Daniel Štefankovič. Decidability of string graphs. In *STOC*, pages 241–246. ACM, 2001.

[44] F.W. Sinden. Topology of thin film RC-circuits. *Bell Systems Technological Journal*, pages 1639–1662, 1966.

[45] Waldo Tobler. Thirty-five years of computer cartograms. *Annals of the Association of American Geographers*, 94(1):58–73, March 2004.

[46] John Venn. On the diagrammatic and mechanical representation of propositions and reasonings. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 9:1–18, 1880.

[47] Anne Verroust and Marie-Luce Viaud. Ensuring the drawability of extended euler diagrams for up to 8 sets. In Alan F. Blackwell, Kim Marriott, and Atsushi Shimojima, editors, *Diagrams*, volume 2980 of *Lecture Notes in Computer Science*, pages 128–141. Springer, 2004.

[48] E. Weisstein. Mathworld: Triangle inequality. http://mathworld.wolfram.com/TriangleInequality.html/.

Appendix A

Proofs

Some main chapters, sections, and subsections (from now on simply "sections"), have theorems, lemmas, propositions, and corollaries (from now on simply "results"), whose proofs are assigned to the appendix. In such cases, the proof will be in an appendix result with the same number as the main result. The appendix result will be in an appendix section whose number matches the referencing main section's number, but with an "A." prefix. For example, Section 2.4 contains Proposition 2.4.1 whose proof is assigned to the appendix; as a result, Appendix A.2.4 also contains Proposition 2.4.1, but in this instance, the proof is fully-detailed.

A.2 Previous and Related Work

A.2.4 Hypergraph Planarity

Recall that a hypergraph $H = (V, E)$ is a set of vertices V and a set of edges E for which an edge $e \in E$ is a subset of vertices (i.e., $e \subseteq V$). Johnson and Pollak

[33] define the following graph-theoretic necessary-and-sufficient conditions for the existence of hyperedge-based and vertex-based Venn diagrams; this is similar to the relationship between connectivity graphs and Euler diagrams described in Chapter 5.

Lemma A.2.1 (Lemma 1 from Johnson and Pollak [33]). *A hypergraph $H = (V, E)$ is hyperedge-planar if and only if there exists a planar graph G on E (i.e., each vertex of G represents a hyperedge, which is a subset of V), such that for each hypergraph vertex subset $V' \subseteq V$, the subgraph induced by $\{v \in V(G) | V' \subseteq v\}$ is connected.*

Lemma A.2.2 (Lemma 8 from Johnson and Pollak [33]). *A hypergraph $H = (V, E)$ is vertex-planar if and only if there exists a planar graph G on V such that for each hyperedge $e \in E$, the subgraph induced by $\{v \in V(G) | v \in e\}$ is connected.*

Using φ from Def. 2.4.3 and Johnson and Pollak's necessary-and-sufficient conditions, the following propositions relate hyperedge-based Venn diagrams and vertex-based Venn diagrams to Euler diagrams.

Proposition 2.4.1. *There exists a set system S that is representable by an Euler diagram, but for which $\varphi(S)$ is* not *hyperedge-planar.*

Proof. Consider the set system

$$S = \{\emptyset, \{1, 2, 3, 4\}, \{1, 5, 6, 7, 11\}, \{2, 7, 8, 10, 11\}, \{3, 6, 8, 9, 10\}, \{4, 5, 9, 10, 11\}\}$$

on the items $X = \{i | 1 \leq i \leq 11\}$.

As shown in Fig. A.1(b), there is a connectivity graph for S; therefore, by Thm. 5.1.1, S is representable by the corresponding Euler diagram in Fig. A.1(c).

Let $H = (V, E) = \varphi(S)$ be the hypergraph that corresponds to S, and let G be the planar graph that, by Lem. A.2.1, must exist if H is hyperedge-planar. Then

$$\begin{aligned} V &= \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\} \text{ and} \\ E &= \{\{1, 2, 3, 4\}, \{1, 5, 6, 7, 11\}, \{2, 7, 8, 10, 11\}, \{3, 6, 8, 9, 10\}, \{4, 5, 9, 10, 11\}\}. \end{aligned}$$

For each integer $1 \leq i \leq 9$, there are exactly two hyperedges containing i; therefore, since the subgraph of G induced by i must be connected, the vertices of G that correspond to these hyperedges must be adjacent. The solid edges in Fig. A.1(a) show the subgraph of G induced by the pairwise integers.

Lastly, any subset of integers must also induce a connected subgraph of G; this means that in Fig. A.1(a), an edge (dashed) must be added to connect $\{4, 5, 9, 10, 11\}$ and $\{2, 7, 8, 10, 11\}$ since they both contain $\{10, 11\}$. The resulting subgraph of G is homeomorphic to K_5; therefore, G is not planar and cannot satisfy the conditions of Lem. A.2.1, so $\varphi(S)$ is not hyperedge-planar. □

Proposition 2.4.2. *There exists a set system S that is* not *representable by an Euler diagram, but for which $\varphi(S)$ is hyperedge-planar.*

Proof. Consider the set system

$$S = \{\emptyset, \{1, 2, 3, 4, 10\}, \{1, 5, 6, 7, 10\}, \{2, 7, 8\}, \{3, 6, 8, 9, 10\}, \{4, 5, 9\}\}$$

on the items $X = \{i | 1 \leq i \leq 10\}$.

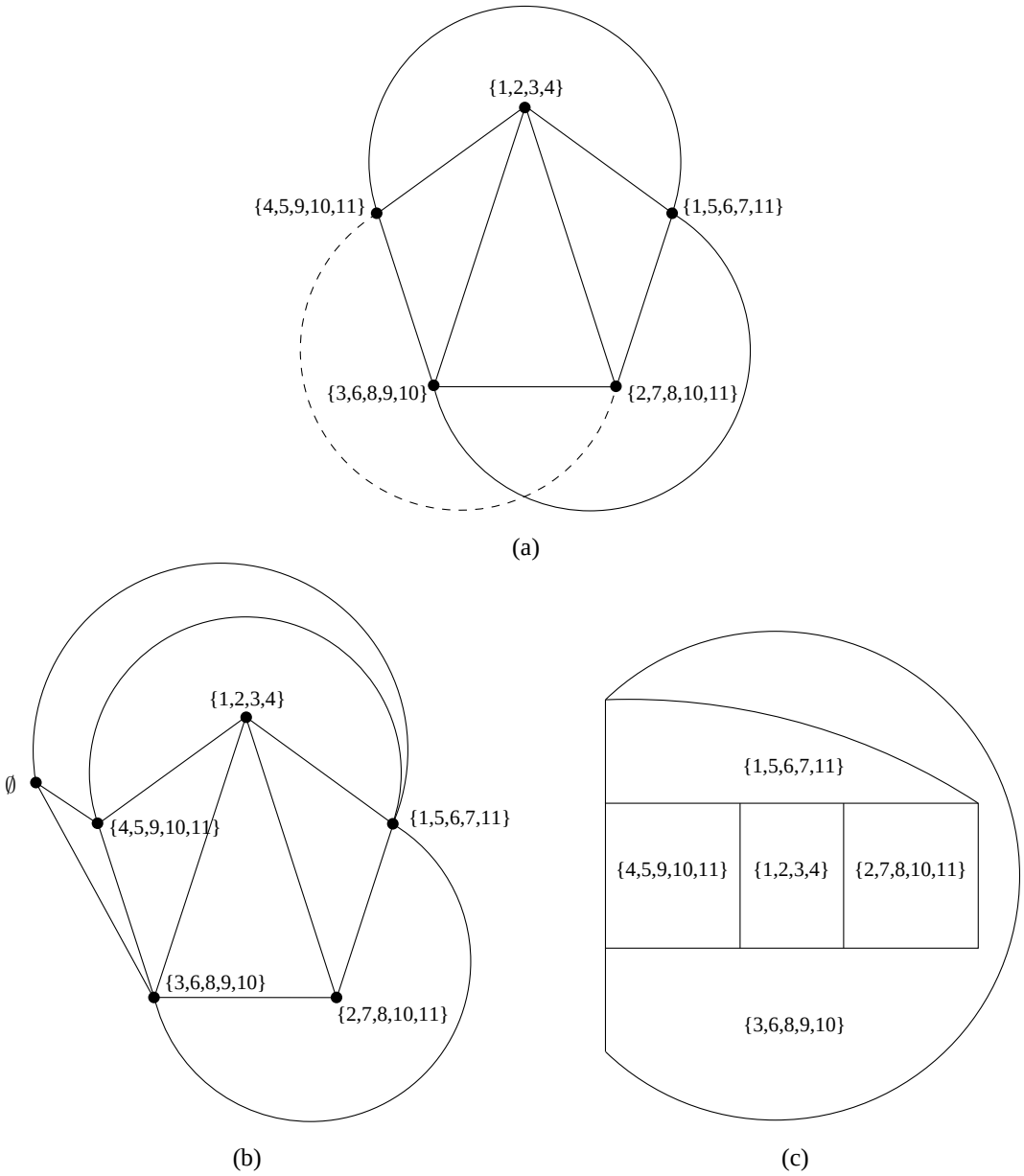


Figure A.1: (a) The non-planar graph that prevents the represented hypergraph from being hyperedge-planar, (b) a connectivity graph for the corresponding set system, and (c) the resulting Euler diagram.

Let $H = (V, E) = \varphi(S)$ be the hypergraph that corresponds to S. Then

$$\begin{aligned} V &= \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\} \text{ and} \\ E &= \{\{1, 2, 3, 4, 10\}, \{1, 5, 6, 7, 10\}, \{2, 7, 8\}, \{3, 6, 8, 9, 10\}, \{4, 5, 9\}\}. \end{aligned}$$

As shown in Fig. A.2(b), there is a planar graph that satisfies Lem. A.2.1; therefore, H is hyperedge-planar via the hyperedge-based Venn diagram in Fig. A.2(c).

Let G be the connectivity graph for S that, by Thm. 5.1.1, must exist if there is an Euler diagram representing S. For each integer $1 \leq i \leq 9$, there are exactly two sets containing i; therefore, since G_i must be connected, G must have the solid edges indicated in Fig. A.2(a).

Lastly, $\overline{G_{10}}$ must be connected; therefore, an edge (or path through $\emptyset$), must exist between $\{4, 5, 9\}$ and $\{2, 7, 8\}$. The additional edge or path (dashed), induces a subgraph of G that is homeomorphic to K_5; therefore, G is not planar and cannot be a connectivity graph for S, so S is not representable by an Euler diagram. □

Proposition 2.4.3. *There exists a set system S that is representable by an Euler diagram, but for which $\varphi(S)$ is* not *vertex-planar.*

Proof. Consider the set system

$$S = \{\emptyset, \{1, 2\}, \{1, 3\}, \{1, 4\}, \{1, 5\}, \{2, 3\}, \{2, 4\}, \{2, 5\}, \{3, 4\}, \{3, 5\}, \{4, 5\}\}$$

on the items $X = \{i | 1 \leq i \leq 5\}$.

As shown in Fig. A.3(b), there is a connectivity graph for S; therefore, by Thm. 5.1.1, S is representable by the corresponding Euler diagram in Fig. A.3(c).

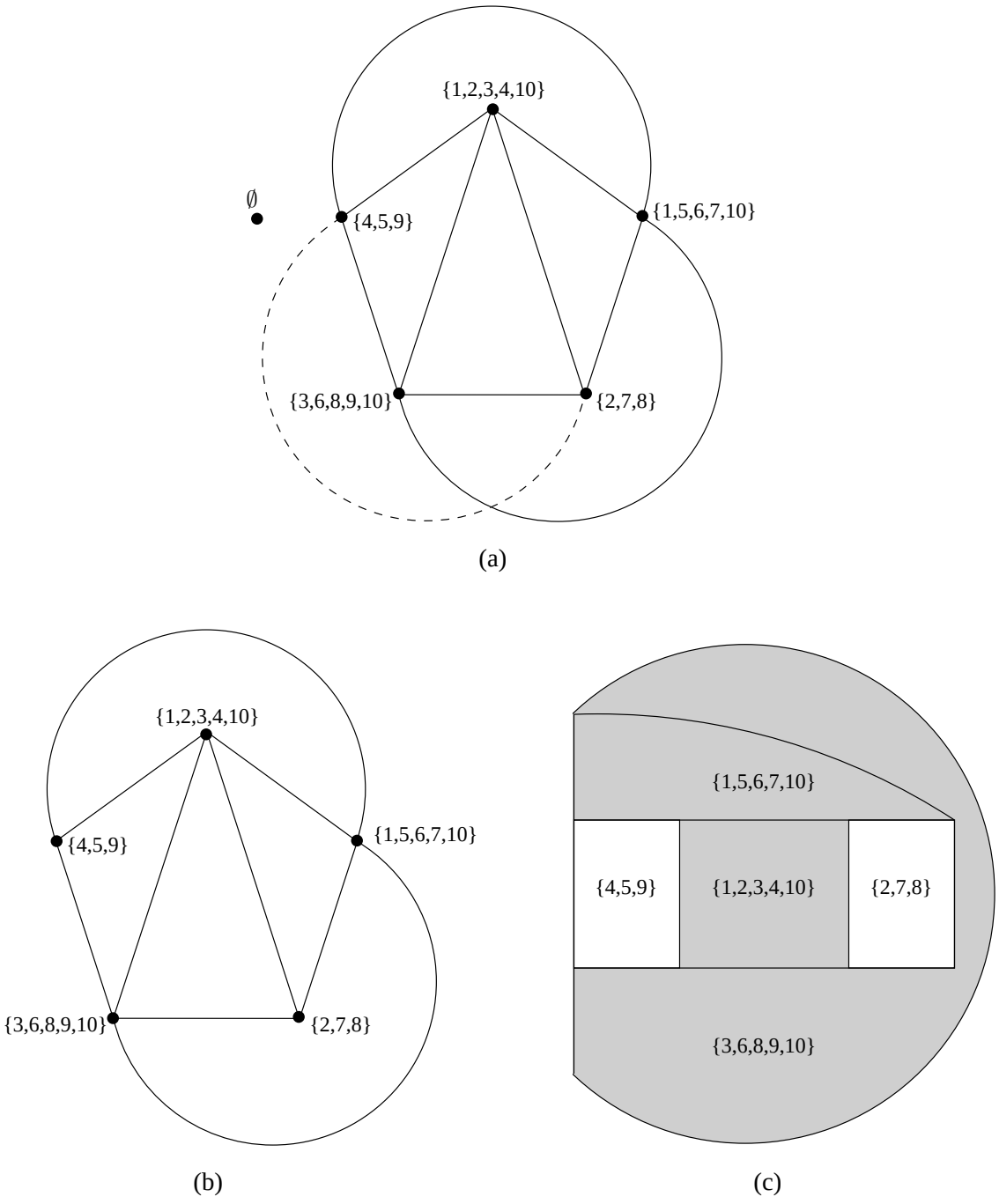


Figure A.2: (a) The non-planar graph that prevents the set system from being represented by an Euler diagram, (b) a planar graph satisfying Lem. A.2.1 for the corresponding hypergraph, and (c) the resulting hyperedge-based Venn diagram.

Let $H = (V, E) = \varphi(S)$ be the hypergraph that corresponds to S, and let G be the planar graph that, by Lem. A.2.2, must exist if H is vertex-planar. Then

$$\begin{aligned} V &= \{1, 2, 3, 4, 5\} \text{ and} \\ E &= \{\{1, 2\}, \{1, 3\}, \{1, 4\}, \{1, 5\}, \{2, 3\}, \{2, 4\}, \{2, 5\}, \{3, 4\}, \{3, 5\}, \{4, 5\}\}. \end{aligned}$$

Each hyperedge e contains exactly two vertices, and since the subgraph induced by e must be connected, these vertices must be adjacent in G. Figure A.3(a) shows the subgraph of G induced by the hyperedges; since this subgraph is isomorphic to K_5, G is not planar and cannot satisfy the conditions of Lem. A.2.2, so $\varphi(S)$ is not vertex-planar. □

Proposition 2.4.4. *There exists a set system S that is* not *representable by an Euler diagram, but for which $\varphi(S)$ is vertex-planar.*

Proof. Consider the set system

$$S = \{\emptyset, \{1, 2, 3, 4\}, \{1, 5, 6, 7\}, \{2, 7, 8, 9\}, \{3, 6, 8, 10\}, \{4, 5, 9, 10\}\}$$

on the items $X = \{i | 1 \leq i \leq 10\}$.

Let $H = (V, E) = \varphi(S)$ be the hypergraph that corresponds to S, then

$$\begin{aligned} V &= \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\} \text{ and} \\ E &= \{\{1, 2, 3, 4\}, \{1, 5, 6, 7\}, \{2, 7, 8, 9\}, \{3, 6, 8, 10\}, \{4, 5, 9, 10\}\}. \end{aligned}$$

As shown in Fig. A.4(b), there is a planar graph that satisfies Lem. A.2.2;

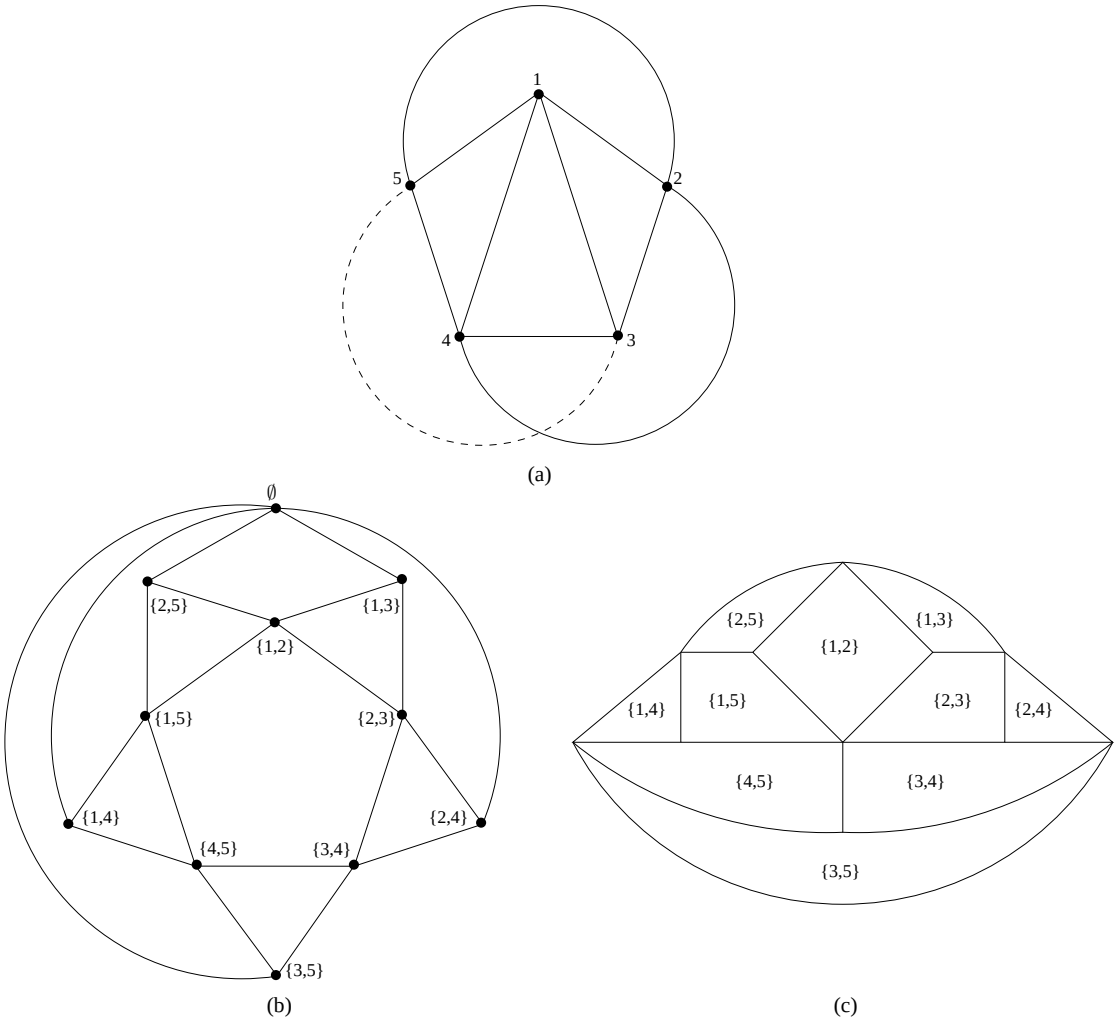


Figure A.3: (a) The non-planar graph that prevents the represented hypergraph from being vertex-planar, (b) a connectivity graph for the corresponding set system, and (c) the resulting Euler diagram.

therefore, H is vertex-planar via the vertex-based Venn diagram in Fig. A.4(c).

Let G be the connectivity graph for S that, by Thm. 5.1.1, must exist if there is an Euler diagram representing S. For each integer $1 \leq i \leq 10$, there are exactly two sets containing i, and since G_i must be connected, these sets must be adjacent in G. Figure A.4(a) shows the subgraph of G induced by the integers; since this subgraph is isomorphic to K_5, G is not planar and cannot be a connectivity graph for S, so S is not representable by an Euler diagram. □

Proposition 2.4.6. *The representability class for Euler diagrams in which the intersection of the interiors of any subset of curves is connected is a* proper subset *of the representability class for Euler diagrams.*

Proof. By definition, Euler diagrams with the subset intersection restriction are a subset of general Euler diagrams, so it remains to show that there exists a set system that is representable by an Euler diagram, but not by one with the subset intersection restriction.

The set system S in the proof of Prop. 2.4.1 is representable by an Euler diagram $C = \{c_1, c_2, \ldots, c_1 1\}$ as shown in Fig. A.1(c); however, C is not an Euler diagram that satisfies the subset intersection restriction since the intersection of $int(c_{10})$ and $int(c_{11})$ (as represented by the union of regions $\{4, 5, 9, 10, 11\}$ and $\{2, 7, 8, 10, 11\}$), is disconnected. As was the case in the argument that the hypergraph $\varphi(S)$ is not hyperedge-planar, attempting to add an edge between $\{4, 5, 9, 10, 11\}$ and $\{2, 7, 8, 10, 11\}$ in S's connectivity graph precludes planarity; therefore, these regions can never be connected, and S is not representable by an Euler diagram that satisfies the subset intersection restriction. □

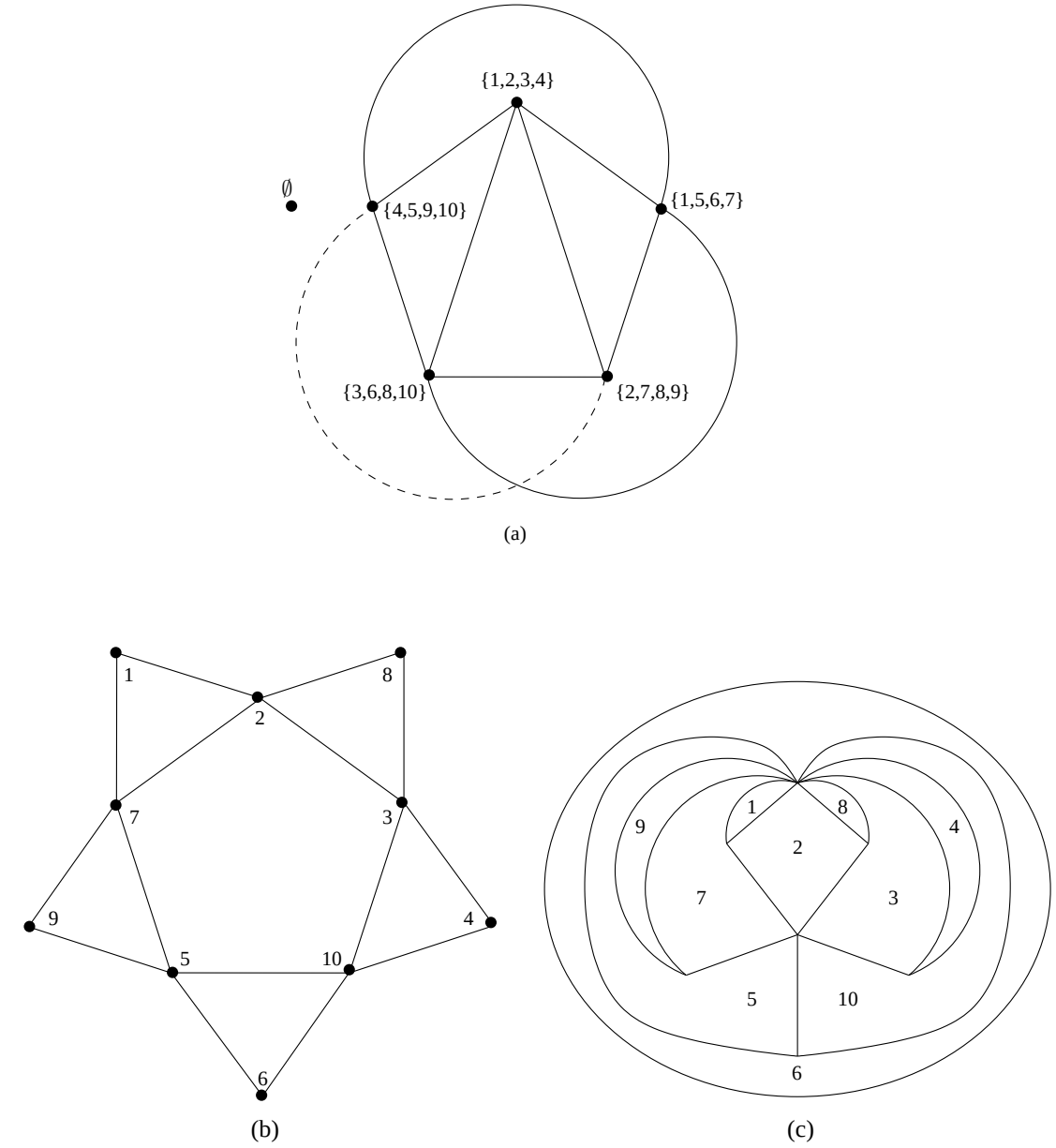


Figure A.4: (a) The non-planar graph that prevents the set system from being represented by an Euler diagram, (b) a planar graph satisfying Lem. A.2.2 for the corresponding hypergraph, and (c) the resulting vertex-based Venn diagram.

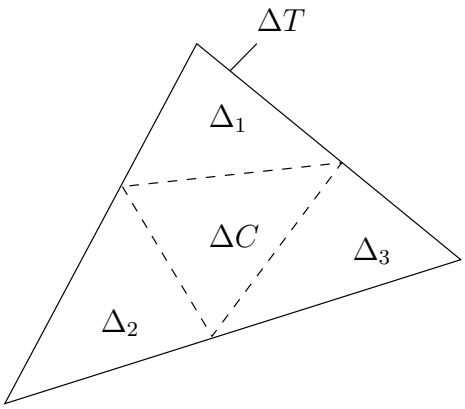


Figure A.5: If triangle ΔC is inscribed in triangle ΔT so as to induce three triangles Δ_1, Δ_2, and Δ_3, then at least one Δ_i must have an area less than or equal to the area of Δ_T.

A.3 Preliminary Results (diagrams with two or three curves)

A.3.4 Three Convex Curve Venn Diagrams

In order to prove Lem. 3.4.1, we need two additional lemmas relating to triangles inscribed in triangles and triangles inscribed in quadrilaterals.

Lemma A.3.1 (Due to Morgantini [37])**.** *If ΔC is a triangle inscribed in another triangle ΔT so that $int(\Delta T) \cap ext(\Delta C)$ is partitioned into three triangles Δ_1, Δ_2, and Δ_3 (e.g., see Fig. A.5), then at least one Δ_i must have an area less than or equal to the area of ΔC.*

Lemma A.3.2. *Let P be a quadrilateral defined by clockwise-ordered points a,b,c, and d, let e be the point on line segment $\overline{cd}$ for which $\overline{be}$ is parallel to $\overline{ad}$, and let f be an interior point of $\overline{bc}$ (e.g., see Fig. A.6(a)).*

If $length(\overline{be}) \leq length(\overline{ad})$*, then at least one of the triangles* $\Delta A = \Delta abf$ *or* $\Delta B = \Delta cdf$ *must have an area less than or equal to the area of triangle* $\Delta C = \Delta fda$ *(e.g., see Fig. A.6(b)).*

Proof. Without loss of generality, let $\overline{ad}$ be horizontal and let $height(b) \leq height(c)$ where the height is measured relative to $\overline{ad}$. As shown in Fig. A.7, there are three orientations of $\overline{ab}$ relative to vertical. We also note that since $length(\overline{be}) \leq length(\overline{ad})$, $\overline{cd}$ cannot be angled away from $\overline{ab}$; for each of the orientations of $\overline{ab}$ in Fig. A.7, the dashed line incident to d and parallel to $\overline{ab}$) indicates the limit of $\overline{cd}$'s angle.

Consider the orientation of $\overline{ad}$ and the location of f shown in Fig. A.7(a). Triangle ΔC forms the lower half of a parallelogram P' with base $\overline{ad}$ and side $\overline{df}$. Because $height(b) \leq height(c)$ and $\overline{cd}$ is never angled away from $\overline{ab}$, triangle ΔA is contained by the upper half of P'; therefore, its area is at most equal to that of ΔC. Figures A.8(a–c) show examples of P' for the range of f's.

This same argument applies for the remaining orientations of $\overline{ab}$; Fig. A.8(d–f) shows analogous examples of P' for the orientation of $\overline{ab}$ in Fig. A.7(c).

As a result, ΔA's area must be less than or equal to the area of ΔC, and the lemma is satisfied. □

With the help of the previous lemmas, we are ready to prove the lemma relating to triangles inscribed in convex curves.

Lemma 3.4.1. *Let* C *be a convex Jordan curve. If* ΔC *is a triangle inscribed in* C *so that* $int(C) \cap ext(\Delta C)$ *is partitioned into three non-empty connected subsets* R_1*,* R_2*, and* R_3*, then at least one* R_i *must have an area less than or equal to the area of* ΔC *(e.g., see Fig. 3.16(a)).*

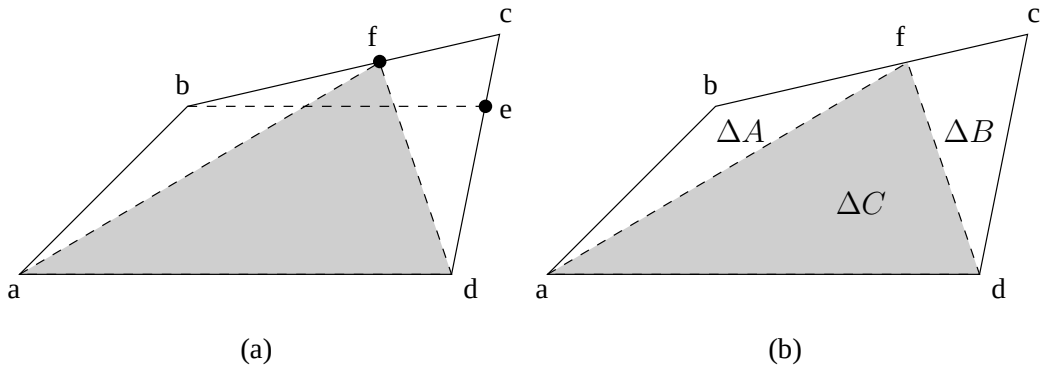


Figure A.6: (a) The labeling of quadrilateral P per Lem. A.3.2 and (b) the partitioning of P into three non-empty triangles.

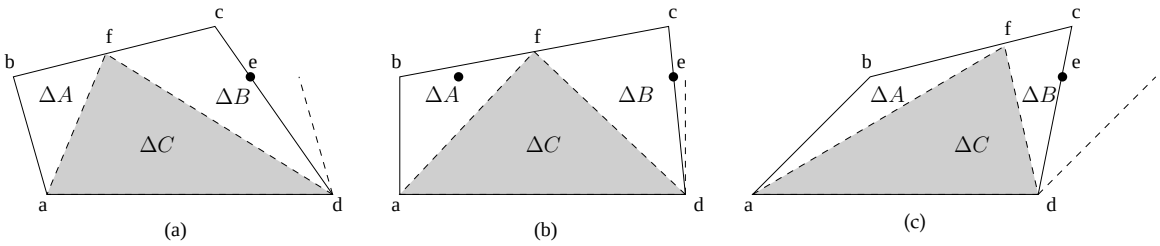


Figure A.7: In the canonical orientation of a quadrilateral for Lem. A.3.2, $\overline{ad}$ is horizontal and $height(b) \leq height(c)$; however, $\overline{ab}$ can have any angle. In addition, since $length(\overline{be}) \leq length(\overline{ad})$, $\overline{cd}$ cannot be angled away from $\overline{ab}$ as indicated by the dashed line incident to d.

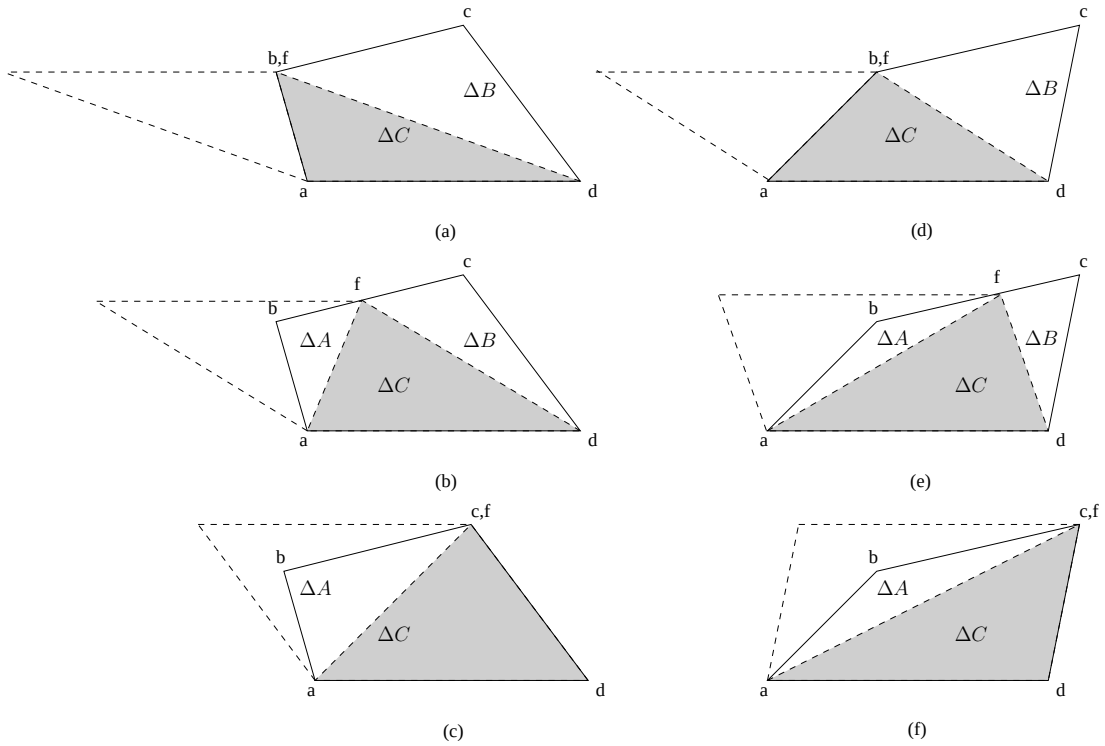


Figure A.8: Regardless of the orientation of $\overline{ab}$ and location of f, the area of ΔA is never greater than the area of ΔC; this can be seen by considering the dashed parallelogram for which ΔC is the lower half and ΔA is always contained by the top half.

Proof. Let p_1, p_2, and p_3 be the points where C and ΔC intersect, and let line L_i be the tangent of C at p_i as shown in Fig. A.10. Since C is convex, it lies completely to one side of each tangent. Suppose all tangents are parallel, then at least two must be collinear; otherwise, one of the tangents L_x is between the other two, and since C intersects each tangent, this implies that C straddles L_x contradicting C's convexity. However, two collinear tangents implies that C and ΔC share a line segment, which contradicts the lemma's preconditions. So, there is at least one tangent that is *not* parallel to the others and without loss of generality, let this tangent be L_1.

Since L_1 is not parallel to L_2 or L_3, it intersects both. As L_2 and L_3 move from p_2 and p_3, respectively, towards their intersection with L_1, they may diverge as shown in Fig. A.10(a) or be parallel to each other or converge as shown in Fig. A.10(b).

In the first case, the triangle ΔL formed by L_1, L_2, and L_3 is inscribed by ΔC and, since C is convex, completely contains C. Inscribing ΔC in ΔL partitions $int(\Delta L) \cap ext(\Delta C)$ into three triangles Δ_1, Δ_2, and Δ_3 where Δ_i completely contains R_i. As a result, if the area of each R_i is greater than the area of ΔC, then the area of each Δ_i is greater than the area of ΔC. However, by Lem. A.3.1 this is impossible.

In the second case, the points where L_2 and L_3 intersect L_1, along with p_2 and p_3, form a quadrilateral P. ΔC is inscribed in P and partitions $int(P) \cap ext(\Delta C)$ into two triangles Δ_1 and Δ_2 each of which completely contains an R_i. As a result, if the area of each R_i is greater than the area of ΔC, then the area of both Δ_1 and Δ_2 is greater than the area of ΔC. However, by Lem. A.3.2, this is impossible.

So in either arrangement of L_1, L_2, and L_3, it is impossible for the area of each R_i to be greater than the area of ΔC; therefore, at least one R_i must have an area less than or equal to the area of ΔC. □

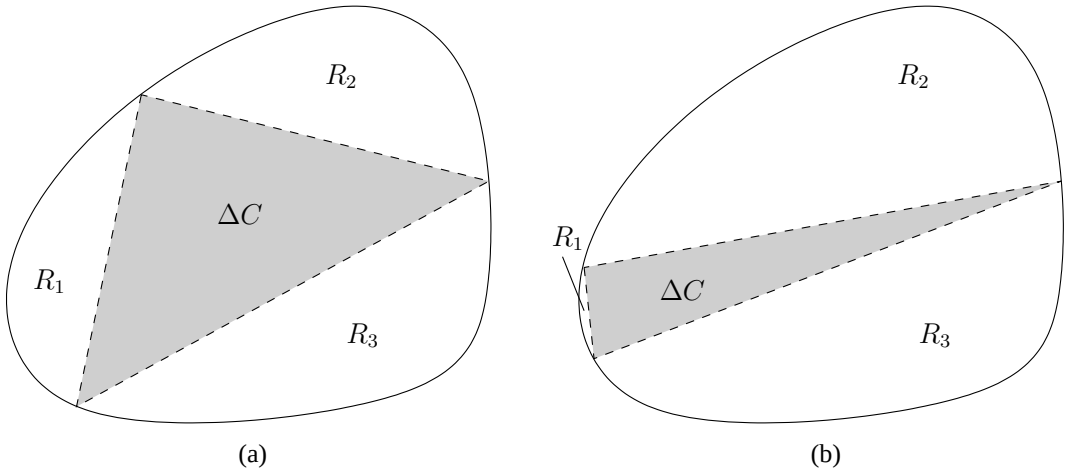


Figure A.9: If a triangle ΔC is inscribed in a closed convex curve C so that it induces connected subsets R_1, R_2, and R_3, then at least one R_i must have an area less than or equal to the area of ΔC.

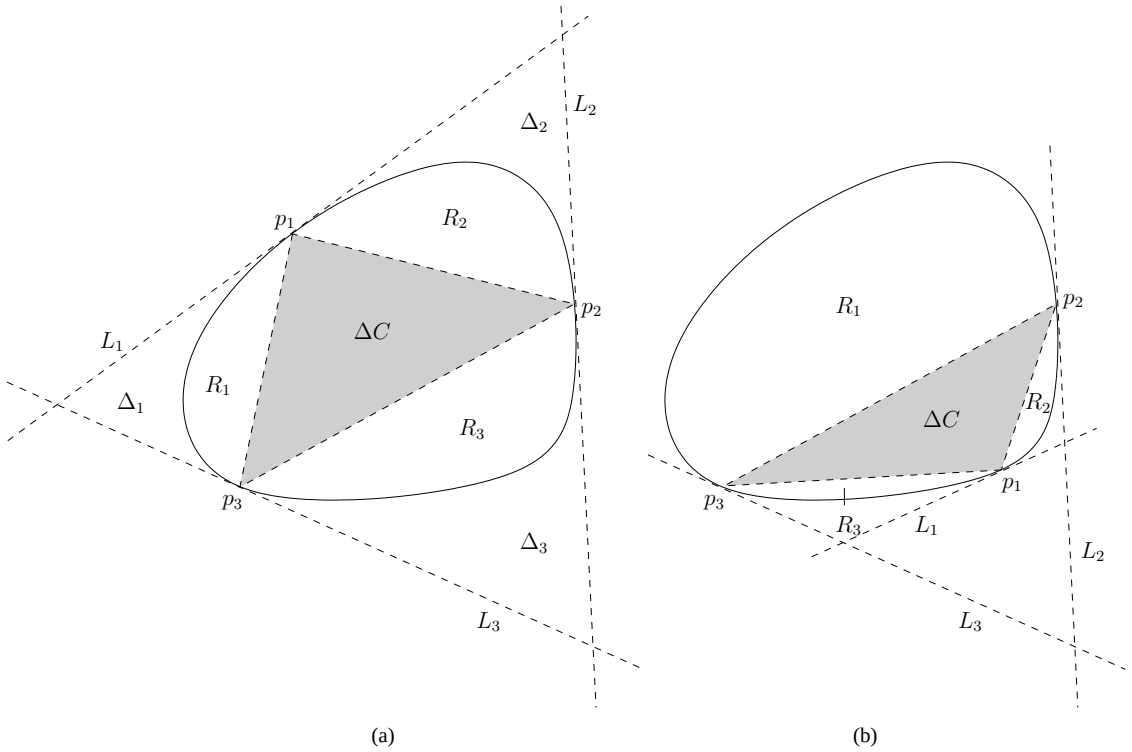


Figure A.10: There are two possible arrangements of the tangents where ΔC intersects C: (a) from p_2 and p_3 towards L_1, L_2 and L_3 diverge and (b) from p_2 and p_3 towards L_1, L_2 and L_3 are parallel or converge.

The previous lemma is the basis of the proof for the main result of Section 3.4.

Theorem 3.4.2. *There exists a weight function ω for which there is* no *ω-proportional simple* 3*-Venn diagram whose curves are convex.*

Proof. Let $\alpha \in \mathbb{R}^+$ be a constant and let ω_x be a weight function parameterized by $x \in \mathbb{R}^+$ where

1. $\omega_x(\{1\}) = \omega_x(\{2\}) = \omega_x(\{3\}) = x$,
2. $\omega_x(\{1,2\}) = \omega_x(\{1,3\}) = \omega_x(\{2,3\}) = 2\alpha$, and
3. $\omega_x(\{1,2,3\}) = \alpha$.

Suppose for all $x \in \mathbb{R}^+$ there exists an ω_x-proportional simple 3-Venn diagram $C_x = \{c_1, c_2, c_3\}$ for which c_1, c_2, and c_3 are convex. Chilakammari et al. [34] determined that all simple 3-Venn diagrams are topologically equivalent to the three circle diagram shown in Fig. A.11(a) (i.e., their regions and point intersections have the same relative positions), so for each C_x, the super-region

$$core(C_x) = r(\{1,2,3\}) \cup r_c(\{1,2\}) \cup r_c(\{1,3\}) \cup r_c(\{2,3\})$$

(where $r_c(x)$ is the *closure* of region x and thus includes its boundary), has a boundary $\delta core(C_x)$, which is a Jordan curve.

Since $core(C_x)$ may be non-convex, let $\mathcal{D}(C_x) \geq 0$ be the difference in area between $core(C_x)$ and its convex hull. By the definition of convexity, $core(C_x)$ is convex if and only $\mathcal{D}(C_x) = 0$.

Suppose, over all $x \in \mathbb{R}^+$, $\mathcal{D}(C_x)$ is never zero; that is, $\epsilon = \inf_{x \in \mathbb{R}^+} \mathcal{D}(C_x) > 0$. Consider the diagram $C = C_{\epsilon/4}$.

Since $\mathcal{D}(C) \geq \epsilon > 0$, $core(C)$ is non-convex; therefore, its boundary $\delta core(C)$ must have at least one inflection point.

Referring again to Fig. A.11(a), we can see that $\delta core(C)$ is composed of convex curve segments, so its inflection points can only occur where the segments intersect as indicated by the black circles and the white circles. Note that within a neighbourhood of each black circle, $\delta core(C)$ is a segment of the boundary of the intersection of the interiors of two of C's curves. Since C's curves are convex, the intersection of the interiors of any two of its curves is convex; therefore, a black circle which is an inflection point of $\delta core(C)$ implies the contradiction that it is also an inflection point on the boundary of a convex region.

So at least one of the white circles is an inflection point of $\delta core(C)$. Since the perimeter of C circumscribes $\delta core(C)$, and from our previous argument, the black circles are on $core(C)$'s convex hull, we must have

$$\omega(\{1\}) + \omega(\{2\}) + \omega(\{3\}) \geq \mathcal{D}(C)$$

in order for C's curves to be convex. If not, as shown in Fig. A.11(b), the perimeter of C would be inside $\delta core(C)$'s convex hull, and thus induce inflection points on its curves. However, since $\omega(\{1\}) + \omega(\{2\}) + \omega(\{3\}) = \frac{3}{4}\epsilon$, we have the contradiction $\frac{3}{4}\epsilon \geq \mathcal{D}(C) \geq \epsilon$.

So there must be an $x' \in \mathbb{R}^+$ and diagram $C' = C_{x'}$ for which $\mathcal{D}(C_{x'})$ tends to zero. In other words, C' is a diagram for which $core(C')$ is convex as shown in Fig. A.12. Consider the regions of C'. Region $\{1, 2, 3\}$ is convex since it is the intersection of the interiors of three convex curves; therefore, we can inscribe a triangle ΔC whose vertices are at the points of intersection between $\delta r(\{1, 2, 3\})$ and $\delta core(C')$. Clearly, the area of ΔC will be less than or equal to $\omega_{x'}(\{1, 2, 3\}) = \alpha$. In addition, ΔC divides

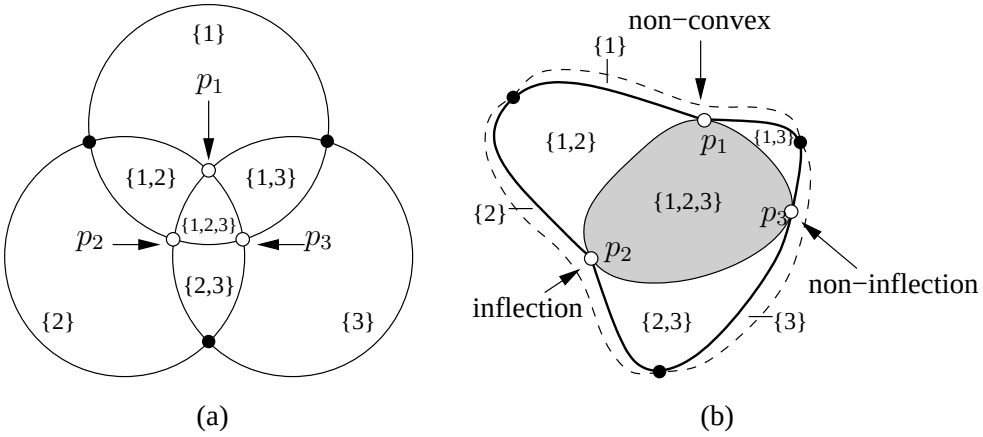


Figure A.11: (a) All simple 3-Venn diagrams are topologically equivalent to the classic three circle diagram, and (b) if C is an ω-proportional simple 3-Venn diagram for which $core(C)$ is non-convex, the inflection points of the boundary $\delta core(C)$ can only occur at the white circles; as a result, if C is to be convex and p_x is an inflection point, then $\omega(\{x\})$ must be large enough so that region $\{x\}$ does not transfer the inflection to c_x.

$core(C') \cap ext(\Delta C)$ into three connected subsets, each of which contains one of C''s 2-set regions, and thus has area at least 2α. In other words, $\delta core(C')$ is a convex Jordan curve in which there is inscribed a triangle ΔC dividing $int(\delta core(C')) \cap ext(\Delta C)$ into three connected subsets R_1, R_2, and R_3 each of which has an area greater than the area of ΔC; however, by Lem. 3.4.1, this is impossible, so C' cannot exist. Since the assumption that for every ω_x there is a C_x leads to a contradiction, there must be a weight function ω that is *not* representable by an ω-proportional simple 3-Venn diagram whose curves are all convex. □

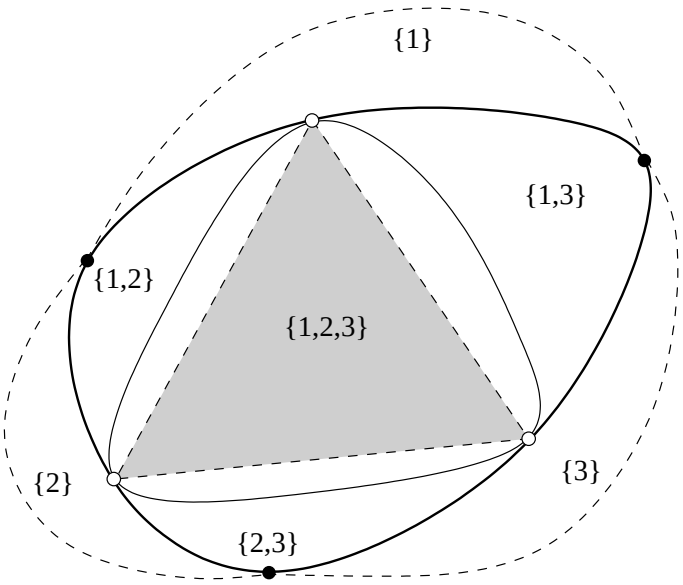


Figure A.12: If C is an ω-proportional simple 3-Venn for which $core(C)$ is convex, then a triangle inscribed in $core(C)$ so that its vertices coincide with the white circles will have area less than $\omega(\{1,2,3\})$, and the three remaining connected subsets of $core(C)$ each enclose a 2-set region, and will therefore have area greater than $\omega(\{x,y\})$, where $\{x,y\}$ is the enclosed region.

A.6 Expressiveness of Diagram Types

A.6.2 Connected Euler-like Diagrams

Figure A.13 is a version of Fig. 6.7 where each example diagram is labeled. In the following exposition, each diagram is considered and a proof is given for why it cannot be represented by *any* diagram with more restrictions (i.e., the set system is not in a region enclosed by additional curves).

(a) (connected simple Euler diagram)

$$S = \{\emptyset, \{1\}\}$$

on the items $X = \{1\}$.

Trivial since there are no possible additional restrictions. □

(b) (connected pairwise Euler diagram)

$$S = \{\emptyset, \{1, 2\}\}$$

on the items $X = \{1, 2\}$.

Suppose S can be represented by a connected simple Euler diagram. By Thm. 5.4.1, since $|X| > 1$, there is a connectivity graph G for S that is a subgraph of the closeness graph $G_c(S)$. $G_c(S)$ is a graph on S whose edges connected vertices that differ by a single element; as a result, $G_c(S)$ has *no* vertices, which implies that any subgraph is disconnected and cannot be a connectivity graph. □

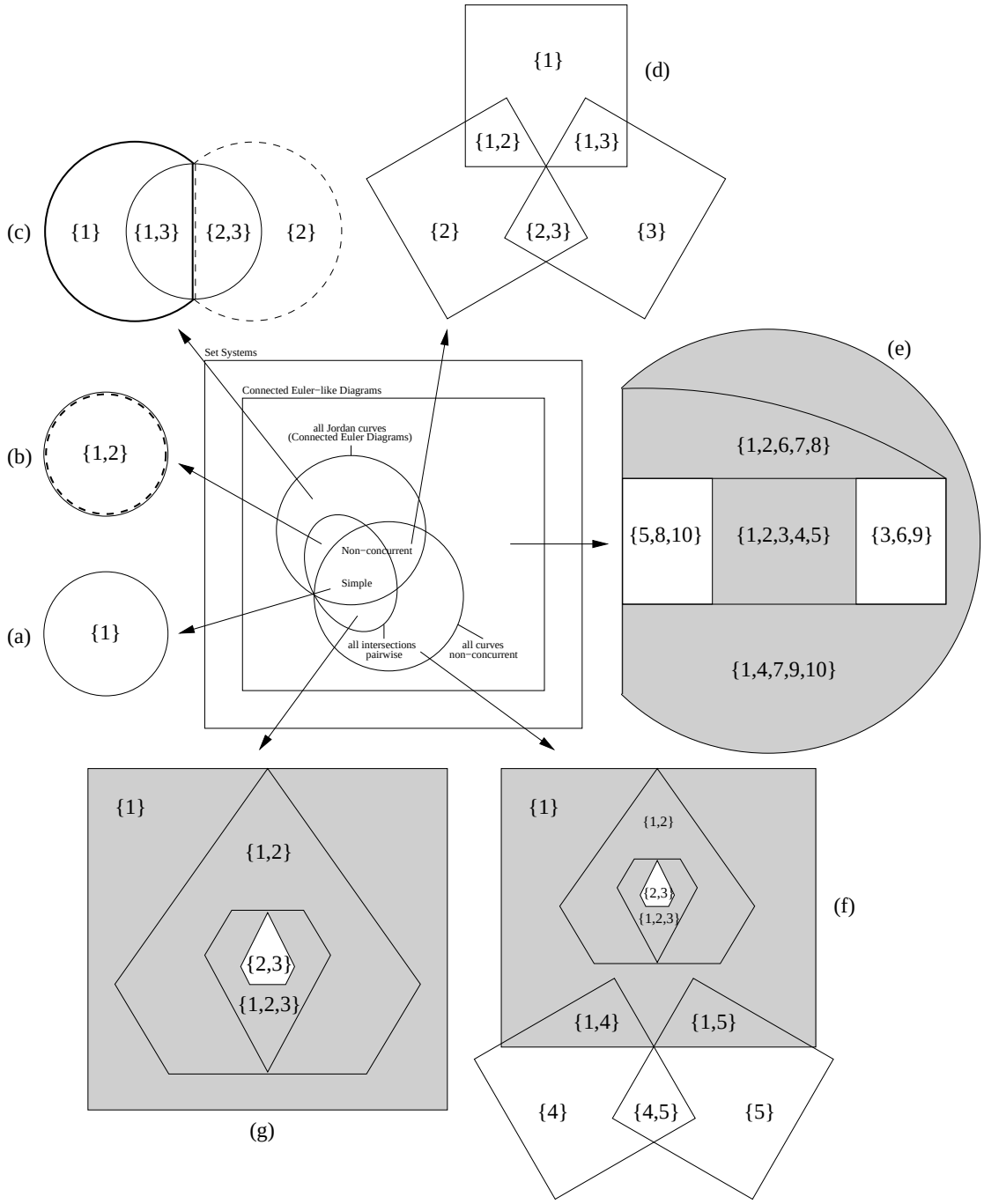


Figure A.13: Examples of connected Euler-like diagrams representing set systems in their respective regions. When applicable, the interior of c_1 is shaded to indicate its non-Jordan curve boundary.

(c) (connected Euler diagram)

$$S = \{\emptyset, \{1\}, \{1,3\}, \{2\}, \{2,3\}\}$$

on the items $X = \{1,2,3\}$.

Suppose S can be represented by a connected pairwise Euler diagram; by Thm. 5.3.1, this implies that S can be represented by a simple Euler diagram. By Thm. 5.4.1, since $|X| > 1$, there is a connectivity graph G for S that is a subgraph of the closeness graph $G_c(S)$. Figure 5.8(a) shows $G_c(S)$. Since G is a connectivity graph, the subgraph G_3 is connected; therefore, since G_3 is a graph on the vertices $\{1,3\}$ and $\{2,3\}$, there is an edge $(\{1,3\},\{2,3\})$, which contradicts that G is a subgraph of $G_c(S)$. As a result, S cannot be represented by a connected pairwise Euler diagram.

Suppose S can be represented by a connected non-concurrent Euler diagram. By Thm. 5.2.2 and the same argument as above, this yields a contradiction. As a result, S cannot be represented by a connected non-concurrent Euler diagram. □

(d) (connected non-concurrent Euler diagram)

$$S = \{\emptyset, \{1\}, \{1,2\}, \{1,3\}, \{2\}, \{2,3\}, \{3\}\}$$

on the items $X = \{1,2,3\}$.

It was proved in the example following Thm. 5.4.1 that S cannot be represented by a connected simple Euler diagram. □

(e) (connected Euler-like diagram)

$$S = \{\emptyset, \{1,2,3,4,5\}, \{1,2,6,7,8\}, \{1,4,7,9,10\}, \{3,6,9\}, \{5,8,10\}\}$$

on the items $X = \{1 \leq i \leq 10\}$.

Suppose S can be represented by a connected Euler diagram. By Thm. 5.1.1, there is a simple connectivity graph G for S. Figure A.14 shows the edges (black) that *must* be in G in order to achieve G_x connectivity for all $x \in X$; these edges must be present since they connect pairs of vertices that uniquely share x. Consider $\overline{G_1}$, which is a subgraph of G on the vertices $\{3,6,9\}$ and $\{5,8,10\}$. Since G is a connectivity graph for S, $\overline{G_1}$ is connected, which implies that G must contain the dashed edge; however, this induces a K_5 subgraph, which contradicts G's planarity. As a result, S cannot be represented by a connected Euler diagram.

Suppose S can be represented by a connected non-concurrent Euler-like diagram C. Clearly, at least one region r of C must be adjacent to the outer face, and since C is non-concurrent, r must represent a singleton set. However, S contains no singleton sets, so C cannot represent S. As a result, S cannot be represented by a connected non-concurrent Euler-like diagram. □

(f) (connected non-concurrent Euler-like diagram)

$$S = \{\emptyset, \{1\}, \{1,2\}, \{1,2,3\}, \{2,3\}, \{1,4\}, \{1,5\}, \{4\}, \{4,5\}, \{5\}\}$$

on the items $X = \{1,2,3,4,5\}$.

Suppose S can be represented by a connected simple Euler-like diagram $C = \{s_1, s_2, s_3, s_4, s_5\}$. Based on the sets in S, items 1 and 5 are the only ones that interact with item 4, so in the following exposition, without loss of generality, we can

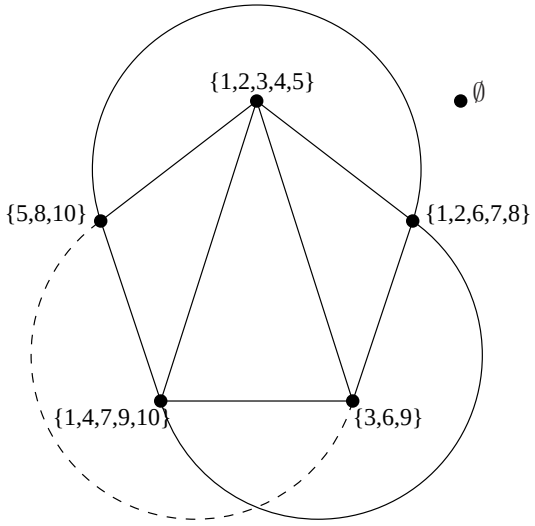


Figure A.14: In any connectivity graph representing this set system, the solid edges must be present to achieve G_x connectivity. In addition, the dashed edge must be present to achieve $\overline{G_x}$ connectivity. The resulting graph is non-planar and there can be no connectivity graph for this set system.

ignore items 2 and 3. We consider the two possible cases for δs_4: either it is a Jordan curve or it is not a Jordan curve.

Suppose δs_4 is a Jordan curve, then it is homeomorphic to the circle shown in Fig. A.15(a). Because there exist sets $\{1,4\},\{4,5\} \in S$, a portion of s_1 and s_5 must be interior to δs_4, and because there exists set $\{1,5\} \in S$, a portion of s_1 and s_5 must be exterior to δs_4. Since C is an Euler-like diagram, s_1 is a connected plane subset, so δs_1 must transversely intersect δs_4 in order for s_1 to be both interior and exterior to δs_4. Similarly, δs_5 must transversely intersect δs_4. Since C is simple, all its intersections are pairwise, so the intersections between δs_1 and δs_4 are distinct from the intersections between δs_5 and δs_4. The dashed edges in Fig. A.15(a) show the *minimum* connections between the regions; these create a disconnected $\emptyset$ region, which contradicts that C is an Euler-like diagram. As a result, δs_4 cannot be a Jordan

curve.

Suppose δs_4 is not a Jordan curve (i.e., s_4 has holes). Without loss of generality, let s_4 have a single hole, then it is homeomorphic to the shaded annulus in Fig. A.15(b). The hole must be interior to some other $s_x \in C$; otherwise, the $\emptyset$ region would be disconnected. Since C is simple, all its item boundaries are non-concurrent, so if the hole is interior to item x, there must be some set in S containing items 4 and x; $\{1, 4\}$ and $\{4, 5\}$ are the only such sets. In addition, it is clear that the hole is part of more than one region, a non-concurrent intersection must result, so without loss of generality, let the hole represent region 5. We now have the same situation as the case when δs_4 was a Jordan curve: since regions $\{1, 4\}$ and $\{4, 5\}$ are interior to s_4 and region $\{1, 5\}$ is exterior to s_4, the necessary region connections result in a disconnected $\emptyset$ region, which contradicts that C is an Euler-like diagram. As a result, δs_4 cannot be a non-Jordan curve.

Since a contradiction results regardless of whether or not δs_4 is a Jordan curve, S cannot be represented by a connected simple Euler-like diagram.

Suppose S can be represented by a connected non-concurrent Euler diagram. By Thm. 5.2.2, there is a simple connectivity graph G for S that is a subgraph of the closeness graph $G_c(S)$. Figure A.16 shows the closeness graph for S. Since G is a connectivity graph for S, $\overline{G_3}$ is connected, but the corresponding subgraph in $G_c(S)$ is disconnected, which implies the contradiction that G is not a subgraph of $G_c(S)$. As a result, S cannot be represented by a connected non-concurrent Euler diagram. □

(g) (connected simple Euler-like diagram)

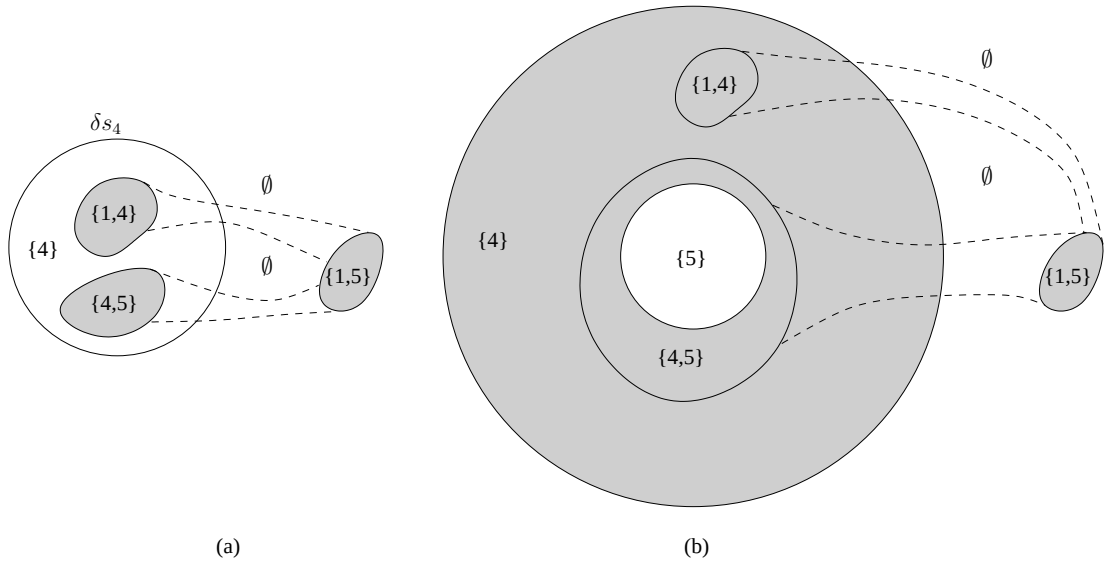


Figure A.15: (a) If δs_4 is a Jordan curve, the presence of the shaded regions implies that δs_1 and δs_5 must transversely intersect δs_4; however, if these intersections are all pairwise, the $\emptyset$ region is disconnected. (b) If δs_4 is not a Jordan curve, a similar arrangement of regions results in a disconnected $\emptyset$ region.

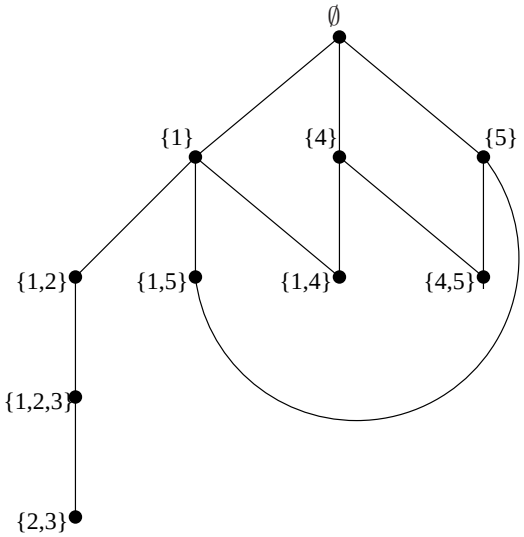


Figure A.16: The closeness graph for this set system.

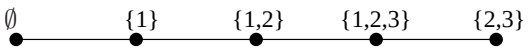


Figure A.17: The closeness graph for this set system.

$$S = \{\emptyset, \{1\}, \{1, 2\}, \{1, 2, 3\}, \{2, 3\}\}$$

on the items $X = \{1, 2, 3\}$.

Suppose S can be represented by a connected simple Euler diagram. By Thm. 5.4.1, there is a simple connectivity graph G for S that is a pseudo-subgraph of the closeness graph $G_c(S)$. Figure A.17 shows the closeness graph for S. Since G is a connectivity graph for S, $\overline{G_3}$ is connected, but the corresponding subgraph in $G_c(S)$ is disconnected, which implies the contradiction that G is not a pseudo-subgraph of $G_c(S)$ (since adding parallel edges does not improve connectivity). As a result, S cannot be represented by a connected simple Euler diagram. □

A.6.3 Disconnected Euler-like Diagrams

Figure A.18 is a version of Fig. 6.11 where each example diagram is labeled. The diagrams labeled (a)–(g) are all connected Euler-like diagrams, so the results from the previous section apply (which is why they have been labeled to match Fig. A.13). As a result, we need only consider the diagrams labeled (h) and (i).

(h) (disconnected pairwise Euler-like diagram)

$$S = \{\emptyset, \{1, 2\}, \{1, 2, 3\}, \{1, 2, 3, 4\}, \{3, 4\}\}$$

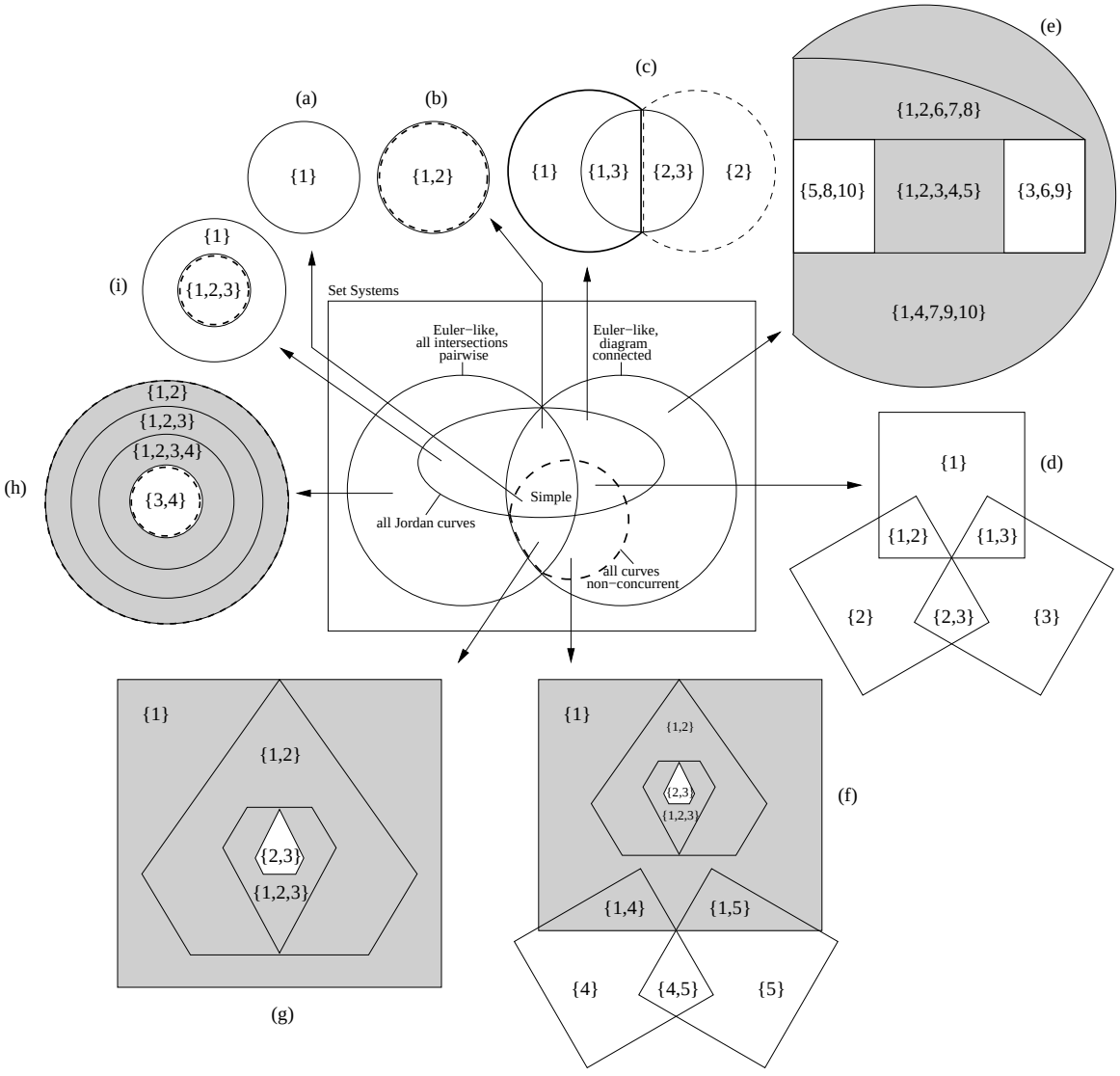


Figure A.18: Examples of Euler-like diagrams representing set systems in their respective regions. When applicable, the interior of c_1 is shaded to indicate its non-Jordan curve boundary.

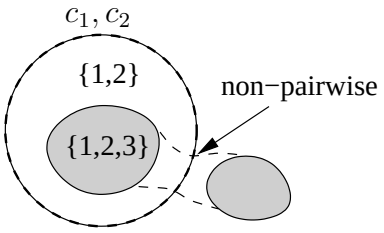


Figure A.19: Since c_1 and c_2 are equivalent Jordan curves and c_3 is a Jordan curve that appears both in the interior and exterior of c_1/c_2, it must cross c_1/c_2, and thus create a non-pairwise intersection.

on the items $X = \{1, 2, 3, 4\}$.

Suppose S can be represented by a disconnected pairwise Euler diagram C. Since items 1 and 2 are equivalent, Thm. 5.3.1 cannot be applied; however, this implies that Jordan curves c_1 and c_2 are equivalent. Because there exists the set $\{1, 2, 3, 4\} \in S$, the interior of Jordan curve c_3 must intersect with the interior of c_1/c_2. Similarly, since there exists the set $\{3, 4\} \in S$, the interior of Jordan curve c_3 must intersect the exterior of c_1/c_2. In other words, at some point c_3 must be interior to c_1/c_2 and at some other point c_3 must be exterior to c_1/c_2 as shown in Fig. A.19; however, this implies that c_3 crosses c_1/c_2, which creates a non-pairwise intersection and contradicts that C is pairwise. As a result, S cannot be represented by a disconnected pairwise Euler diagram.

Suppose S can be represented by a connected non-concurrent Euler-like diagram C. Clearly, at least one region r of C must be adjacent to the outer face, and since C is non-concurrent, r must represent a singleton set. However, S contains no singleton sets, so C cannot represent S. As a result, S cannot be represented by a connected non-concurrent Euler-liked diagram. □

(i) (disconnected pairwise Euler diagram)

$$S = \{\emptyset, \{1\}, \{1, 2, 3\}\}$$

on the items $X = \{1, 2, 3\}$.

Suppose S can be represented by a connected pairwise Euler diagram C. Since items 2 and 3 are equivalent, curves c_2 and c_3 are equivalent, and thus concurrently intersect. Since C is connected, c_1 must intersection c_2/c_3, but this results in a non-pairwise intersection, which contradicts that C is pairwise. As a result, S cannot be represented by a connected pairwise Euler diagram. □

A.7 Computational Complexity of the Euler Diagram Generation Problem

Lemma 7.0.5. *The Euler Diagram Decision Problem (EDDP) is in NP.*

Proof. Let S be the set system on the items $X = \{x_1, x_2, \ldots, x_n\}$ that is under question. We will use an adjacency matrix graph representation and treat each element $s \in S$ as a bitstring with maximum length $|X|$; this allows us to determine whether or not s contains item x_i by checking bit i in constant-time.

A random graph $G = (S, E)$ can be generated in $O(|S|^2)$ time by randomly setting each of the matrix's $|S|^2$ elements.

To determine whether G is a connectivity graph, by Def. 5.0.1, we have to check that G is planar and that for each item $x \in X$, G_x is connected and $\overline{G_x}$ is connected.

The planarity of G can be checked in linear time with respect to the number of vertices (i.e., $O(|S|)$).

The connectivity of G_x can be determined in $O(|S|^2)$ time via the following procedure:

1. Find a vertex/set $s \in S$ with $x \in s$; this can be done in $O(|S|)$ time.
2. Perform a depth-first search of G starting at s and only using edges that go to vertices/sets that contain x (i.e., only search G_x); this can be done in $O(|S|^2)$ time since G is an adjacency matrix.
3. Count the number n_x of sets that contain x; this can be done in $O(|S|)$ time.
4. G_x is connected if and only if n_x vertices are visited by the DFS in Step 2.

Similarly, the connectivity of $\overline{G_x}$ can also be determined in $O(|S|^2)$ time. Since there are $|X|$ items,checking the subgraph connectivity of G can be done in $O(|X||S|^2)$ time.

So a "guess and check" strategy can be done in $O(|X||S|^2)$ time (i.e., polynomial time with respect to the input size), which is sufficient to prove that the EDDP is in NP. □

A.7.2 Hamilton Path → Connectivity Graph

Lemma 7.2.1. *Let G be a cubic 3-connected plane graph and let $S(G'_p)$ be the derived set system as defined by Def. 7.1.2. If G has a Hamilton path then there is a connectivity graph for $S(G'_p)$.*

Proof. Let G^S be the plane graph constructed in Sec. 7.2. By the construction, G^S is connected and $V(G^S) = S(G'_p)$. In addition, the construction ensures that G^S_x is connected for each item x of $S(G'_p)$; therefore, it suffices to prove that $\overline{G^S_x}$ is connected in order to show that G^S is a connectivity graph for $S(G'_p)$.

Each pairwise item x is shared by a pair of adjacent vertices in G'_p; therefore, $\overline{G^S_x}$ results from removing these two vertices. Without the v' vertices, G'_p is homeomorphic to a 3-connected graph, so the subgraph of G'_p (excluding the v' vertices) induced by $\overline{G^S_x}$ remains connected. In addition, since each v' vertex is connected in G^S to two non-adjacent vertices in G'_p, the v' vertices remain connected to the subgraph of G'_p induced by $\overline{G^S_x}$. Lastly, the $\emptyset$ vertex is connected to two non-adjacent vertices; therefore, it also remains connected to the subgraph of G'_p induced by $\overline{G^S_x}$. So for every pairwise item x, $\overline{G^S_x}$ is connected.

For a surrounding item x_v, $\overline{G^S_{x_v}}$ results from removing all of v's surrounding vertices; this is very similar to removing v from G, and since G is 3-connected, removing v does not disconnect the graph. In addition, the Hamilton path edges ensure that v does not become isolated. Figure 7.6(b) shows an example of removing v_2's surrounding vertices; note how the Hamilton path edge between v_2 and v'_3 preserves v_2's connectivity. So for every surrounding item x_v, $\overline{G^S_{x_v}}$ is connected.

Lastly, for the Hamilton item x_{hp}, $\overline{G^S_{x_{hp}}}$ results from removing all the Hamilton path vertices and their corresponding v' vertices from G^S. We prove by induction on $i = 1, 2, \ldots, |V(G)|$ that the removal of the Hamilton path vertices v_i and v'_i leaves the remaining surrounding vertices of $v_1, v_2, \ldots, v_i$ connected.

For the base case $i = 1$, when v_1 and v'_1 are removed, v_1's remaining surrounding vertices are connected by the additional edge that was added between the vertices adjacent to v'_1. Figure 7.6(b) shows an example of removing v_1 and v'_1.

For the inductive step $2 \leq i \leq |V(G)|$, we assume that the remaining surrounding vertices of v_j, $j < i$, are connected, and show that removing v_i and v'_i leaves v_i's remaining surrounding vertices connected with the other remaining surrounding

vertices.

When v_i and v'_i are removed, the chain connecting v_i's remaining surrounding vertices is broken; however, each section stays connected to v_{i-1}'s remaining surrounding vertices via the trisected edge e between v_{i-1} and v_i and the v_f vertex corresponding to the face f to the right of e. Since, by the inductive assumption, v_{i-1}'s remaining surrounding vertices are connected with the other remaining surrounding vertices, v_i's remaining surrounding vertices are connected with the other remaining surrounding vertices. Figure. 7.6(c) shows the step of the induction where v_2 and v'_2 are removed.

So by induction, removing the Hamilton path vertices and their corresponding v' vertices from G^S leaves the remaining surrounding vertices connected. The only other vertices that remain in G^S are the vertices corresponding to the faces of G and the $\emptyset$ vertex. By the construction of G^S, each face vertex and the $\emptyset$ vertex has an edge to a non-v' surrounding vertex in G^S; therefore, $\overline{G^S_{x_{hp}}}$ is connected. □

A.7.3 Connectivity Graph → Hamilton Path

Lemma 7.3.1. *Let G be a cubic 3-connected planar graph with plane embedding G_p and let $S(G'_p)$ be the derived set system as described by Def. 7.1.2. If there exists a connectivity graph for $S(G'_p)$ then G has a Hamilton path.*

Proof. Let G^S be a plane embedding of the connectivity graph for $S(G'_p)$. As mentioned in Sec. 7.3, G^S has the same layout as G'_p, but with additional plane edges.

Consider a vertex $v \in G$ within the context of G'_p. There are six faces around v, each of which is shared with exactly one other vertex, u, that is adjacent to v in G; each face admits an additional connection between u and v, either directly with a plane edge (v, u), or indirectly through v' and/or u'. Figure 7.7(a) shows an example

of how additional plane edges may be drawn in G'_p between adjacent vertices (black circles) in G; all the possible combinations of paths involving v, v', u, and u' are shown.

In G^S, v's surrounding vertices must be connected since in the construction of $S(G'_p)$, they share a unique *surrounding* item x_v. In connecting v's surrounding vertices, the added edges must be between surrounding vertices that are the endpoints of consecutive edges incident to v, or between v' and the endpoint of an edge incident to v. Each edge drawn between consecutive surrounding vertices divides one of v's six faces and prevents an additional connection between v and the adjacent vertex in G that shares the face, unless v' bisects this edge.

At least five of the faces must be divided in order for $G^S_{x_v}$ to be connected; therefore, additional connections can be added between v and *at most* two adjacent vertices in G as shown in Fig. 7.7(b) (note that additional edges may be present, but cannot connect v to any other adjacent vertices).

The Hamilton item x_{hp} is uniquely shared by all of G's vertices and their corresponding v' vertices in G^S, and since G^S is a connectivity graph, $G^S_{x_{hp}}$ is connected. As we previously mentioned, within $G^S_{x_{hp}}$, each vertex $v \in G$ can be connected to at most two vertices that are adjacent to v in G. In addition, each v' vertex is located in a single face of G'_p; therefore, it can connect at most two adjacent vertices in G. In other words, as shown in Fig. 7.8(a), $G^S_{x_{hp}}$ induces a spanning subgraph of G's vertices, and each vertex has degree at most 2, which, by definition, is a Hamilton path in G as shown in Fig. 7.8(b). □

A.8 Composite Euler Diagrams

A.8.0 Nesting Tree Theory

This appendix section does not correspond to a dissertation section; however, it provides the mathematical justification for why Alg. 1 is correct. The first part of Alg. 1 constructs the nesting graph of Def. A.8.4 for S and then uses Prop. A.8.8 as the basis for the transitive reduction to T_X. As a result, T_X corresponds to the reduced nesting graph of Def. A.8.4.

Definition A.8.1. Let S be a set system on the items X with $x, y \in X$. We define the following relations between x and y:

1. x *overlaps* y if and only if there exists a set $s \in S$ such that $x, y \in s$;
2. x is *outside* y if and only if there exists a set $s \in S$ such that $x \in s$ and $y \notin s$;
3. x is *equivalent* to y if and only if x is not outside y and y is not outside x;
4. x *crosses* y if and only if x overlaps y, x is outside y, and y is outside x;
5. x *encloses* y if and only if x overlaps y, x is outside y, and y is not outside x.

We denote x equivalent to y by $x \equiv y$, x crosses y by xCy, and x encloses y by xEy. □

Proposition A.8.1. *Let S be a set system on the items X with $x, y \in X$. x is* not *outside y if and only if for all sets $s \in S$, $x \in s \Rightarrow y \in s$.*

Proof. If x is not outside y, there does not exist a set $s \in S$ such that $x \in s$ and $y \notin s$; in other words, for all sets $s \in S$, $x \in s \Rightarrow y \in s$.

If for all sets $s \in S$, $x \in s \Rightarrow y \in s$, there does not exist a set $s \in S$ such that $x \in s$ and $y \notin s$; therefore, x is not outside y. □

Proposition A.8.2. *Let S be a set system on the items X with $x, y \in X$. $x \equiv y$ if and only if for all sets $s \in S$, $x \in s \Leftrightarrow y \in s$.*

Proof. By definition, $x \equiv y$ if and only if x is not outside y and y is not outside x.

By Prop. A.8.1, x is not outside y if and only if for all sets $s \in S$, $x \in s \Rightarrow y \in s$. Similarly, y it not outside x if and only if for all sets $s \in S$, $y \in s \Rightarrow x \in s$.

So x is not outside y and y is not outside x if and only if for all sets $s \in S$, $x \in s \Rightarrow y \in s$ and $y \in s \Rightarrow x \in s$. In other words, $x \equiv y$ if and only if for all sets $s \in S$, $x \in s \Leftrightarrow y \in s$. □

Remark. Because of Prop. A.8.2, if x and y are equivalent items, any relations from Def. A.8.1 that hold for x hold for y, and vice versa. Essentially, equivalent items can be grouped together and treated as a single item. □

Proposition A.8.3. *The relations from Def. A.8.1 have the following properties:*

1. *The overlaps relation is reflexive and symmetric, but not transitive;*
2. *The outside relation is irreflexive, but not symmetric nor transitive;*
3. *The equivalent relation is reflexive, symmetric, and transitive;*
4. *The crosses relation is irreflexive and symmetric, but not transitive;*
5. *The encloses relation is irreflexive, antisymmetric, and transitive.*

Proof. Let S be a set system on the items X, and let $x, y, z \in X$ be three distinct items.

(1). Since $x \in X$, there exists a set $s \in S$ such that $x \in s$; therefore, x trivially overlaps itself, and the overlaps relation is reflexive.

If x overlaps y, there exists a set $s \in S$ such that $x, y \in s$. Trivially, $y, x \in s$; therefore, y overlaps x, and the overlaps relation is symmetric.

Consider the set system $S = \{\emptyset, \{x, y\}, \{y, z\}\}$ on the items $X = \{x, y\}$. Since $\{x, y\} \in S$, x overlaps y, and since $\{y, z\} \in S$, y overlaps z; however, there does not exist a set $s \in S$ such that $x \in s$ and $z \in s$, so x does not overlap z, and the overlaps relation is not transitive.

(2). Clearly, there cannot exist a set $s \in S$ such that $x \in s$ and $x \notin s$; therefore, x is not outside x, and the outside relation is irreflexive.

Consider the set system $S = \{\emptyset, \{x\}, \{x, y\}, \{x, y, z\}, \{z\}\}$ on the items $X = \{x, y, z\}$.

$\{x\} \in S$ implies that x is outside y; however, there does not exist a set $s \in S$ such that $y \in s$ and $x \notin s$, so y is not outside x, and the outside relation is not symmetric.

$\{x, y\} \in S$ implies that y is outside z, and $\{z\} \in S$ implies that z is outside x; however, as previously noted, y is not outside x, so the outside relation is not transitive.

(3). Trivially proven using Prop. A.8.2.

(4). Since the outside relation is irreflexive, x cannot be outside x; therefore, x does not cross x, and the crosses relation is irreflexive.

If xCy, x overlaps y, x is outside y, and y is outside x. Since the overlaps relation is symmetric, y overlaps x, y is outside x, and x is outside y; therefore, yCx, and the crosses relation is symmetric.

Consider the set system $S = \{\emptyset, \{x\}, \{x, y\}, \{y\}, \{y, z\}, \{z\}\}$ on the items $X = \{x, y, z\}$. $\{x, y\} \in S$ implies that x overlaps y, $\{x\} \in S$ implies that x is outside y, and $\{y\} \in S$ implies that y is outside x; therefore, x crosses y. Similarly, $\{y, z\}, \{y\}, \{z\} \in$

S implies that y crosses z.

So x crosses y and y crosses z; however, there does not exist a set $s \in S$ such that $x, z \in s$, so x does not cross z, and the crosses relation is not transitive.

(5). Since the outside relation is irreflexive, x cannot be outside x; therefore, x does not enclose x, and the encloses relation is irreflexive.

If xEy, x overlaps y, x is outside y, and y is not outside x. Since y is not outside x, y cannot enclose x, and the encloses relation is antisymmetric.

If xEy and yEz, x overlaps y, y overlaps z, x is outside y, y is outside z, y is not outside x, and z is not outside y.

Since the equivalent and outside relations are transitive, x overlaps z and x is outside z.

Lastly, by Prop. A.8.1, since z is not outside y, for all sets $s \in S$, $z \in s \Rightarrow y \in s$. Similarly, since y is not outside x, $y \in s \Rightarrow x \in s$. So for all sets $s \in S$, $z \in s \Rightarrow y \in s \Rightarrow x \in s$; therefore, z is not outside x.

Since x overlaps z, x is outside z, and z is not outside x, xEz, and the encloses relation is transitive. □

Definition A.8.2. Let S be a set system on the items X with $x, y \in X$. We say that x is *linked* to y, denoted xLy, if and only if there exists a sequence $x = x_1, x_2, \ldots, x_k = y$ of items from X such that $x_i C x_{i+1}$ for all $1 \leq i < k$. □

Proposition A.8.4. *Let S be a set system on items X. The* linked *relation is reflexive, symmetric, and transitive; therefore, for each item $x \in X$, we can define the equivalence class $[x] = \{y \in X | xLy\}$.*

Proof. Let $x, y, z \in X$ be three distinct items.

The sequence $x = x_1 = x$ satisfies the conditions of the linked relation; therefore, xLx, and the linked relation is reflexive.

If xLy, let $x = x_1, x_2, \ldots, x_k = y$ be a sequence that satisfies the conditions of the linked relation. By Prop. A.8.3(4), the crosses relation is symmetric; therefore, the reverse sequence $y = x_k, x_{k-1}, \ldots, x_1 = x$ satisfies the linked relation for yLx, and the linked relation is symmetric.

If xLy and yLz, let $x = x_1, x_2, \ldots, x_k = y$ and $y = y_1, y_2, \ldots, y_l = z$ be sequences that satisfy the conditions of the linked relation. The concatenated sequence $x = x_1, x_2, \ldots, x_k = y = y_1, y_2, \ldots, y_l = z.$ satisfies the conditions of the linked relation for xLz, and the linked relation is transitive. □

Proposition A.8.5. *Let S be a set system on the items X with $x, y \in X$ and $[x] \neq [y]$. If xEy then xEy' for all $y' \in [y]$.*

In other words, an item in one linked equivalence class encloses either all or none of the items in another linked equivalence class.

Proof. Consider an item $y' \in [y]$. Since $y' \in [y]$, yLy' and by Def. A.8.2, there exists a sequence $y = y_1, y_2, \ldots, y_k = y'$ that satisfies the conditions of the linked relation.

Suppose xEy_i for some $1 \leq i < k$, and note that the above sequence implies that yLy_i and, therefore, $[y] = [y_i]$. We will prove that xEy_i implies xEy_{i+1}.

By Def. A.8.2, y_iCy_{i+1}; therefore, y_i overlaps y_{i+1}, y_i is outside y_{i+1}, and y_{i+1} is outside y_i.

Since y_i overlaps y_{i+1}, there is a set $s \in S$ such that $y_i, y_{i+1} \in s$. Since xEy_i, y_i is not outside x, so by Prop. A.8.1, $x, y_i, y_{i+1} \in s$; therefore, x overlaps y_{i+1}.

Since y_i is outside y_{i+1}, there is a set $s \in S$ such that $y_i \in s$ and $y_{i+1} \notin s$. As before, since y_i is not outside x, $x, y_i \in s$; therefore, x is outside y_{i+1}.

Suppose y_{i+1} is outside x. Since x overlaps y_{i+1} and x is outside y_{i+1}, we have xCy_{i+1}, which implies that xLy_{i+1} and $[x] = [y_{i+1}] = [y]$; however, this contradicts the precondition $[x] \neq [y]$, so y_{i+1} is not outside x.

So x overlaps y_{i+1}, x is outside y_{i+1}, and y_{i+1} is not outside x; therefore, xEy_{i+1}.

Since $y = y_1$, xEy trivially implies xEy_1, and since xEy_i implies $xEy_{i+1}, 1 \leq i < k$, by induction, xEy_k, which implies xEy' since $y_k = y'$.

So for any item $y' \in [y]$, xEy implies xEy'. □

Definition A.8.3. Let S be a set system on the items X with $x, y \in X$ and $[x] \neq [y]$. $[y]$ is *nested* within $[x]$, denoted $[x] >_N [y]$ or $[y] <_N [x]$, if and only if there exist item $x' \in [x]$ and $y' \in [y]$ such that $x'Ey'$. □

Proposition A.8.6. *Let S be a set system on the items X. The* nested *relation is irreflexive and transitive; therefore, it is a strict partial order and defines a poset on the set of linked equivalence classes X/L.*

Proof. Let $[x], [y], [z] \in X/L$ be three distinct linked equivalence classes.

By definition, the nested relation is irreflexive.

Suppose $[x] >_N [y]$ and $[y] >_N [z]$, then there exist items $x' \in [x]$, $y', y'' \in [y]$, and $z' \in [z]$ such that $x'Ey'$, $y''Ez'$.

By Prop. A.8.5, since $[x] \neq [y]$ and $y', y'' \in [y]$, $x'Ey'$ implies $x'Ey''$. Lastly, by Prop. A.8.3(5), the encloses relation is transitive; therefore, $x'Ey''$ and $y''Ez'$ implies $x'Ez'$. Since $x' \in [x]$ and $z' \in [z]$, $[x] >_N [z]$, and the nested relation is transitive.

□

Proposition A.8.7. *Let S be a set system on the items X with $x, y, z \in X$ and $[x] \neq [y] \neq [z]$. If $[x] >_N [z]$ and $[y] >_N [z]$ then either $[x] >_N [y]$ or $[y] >_N [x]$.*

Proof. Since $[x] >_N [z]$ and $[y] >_N [z]$, there exist items $x' \in [x]$, $y' \in [y]$, and $z_x, z_y \in [z]$ such that $x'Ez_x$ and $y'Ez_y$.

By Prop. A.8.5, $x'Ez_x$ implies $x'Ez_y$. Since $x'Ez_y$, x' overlaps z_y; therefore, there exists a set $s \in S$ such that $x', z_y \in s$, and since $y'Ez_y$, z_y is not outside y', so by Prop. A.8.1, $x', z_y, y' \in s$, and x' overlaps y'.

Suppose x' is not outside y' and y' is not outside x', then, x' is equivalent to y', which implies $[x'] = [y']$; however, since $x' \in [x]$ and $y' \in [y]$, $[x] = [x'] = [y'] = [y]$, which contradicts the precondition $[x] \neq [y]$. So x' is outside y' and/or y' is outside x'.

Suppose x' is outside y' and y' is outside x', then since x' overlaps y', we have $x'Cy'$, which implies $x'Ly'$ and $[x'] = [y']$; however, as before, this contradicts the precondition $[x] \neq [y]$. So either x' is outside y' and y' is not outside x', or x' is not outside y' and y' is outside x'. In either case, since x' and y' overlap, this implies either $x'Ey'$ or $y'Ex'$, and since $x' \in [x]$, $y' \in [x]$, and $[x] \neq [y]$, we have either $[x] >_N [y]$ or $[y] >_N [x]$. □

Definition A.8.4. Let S be a set system on the items X and let $G_{>_N} = (X/L, E)$ be a directed graph whose vertices are the linked equivalence classes of S and whose edges are $E = \{([x], [y]) | [x] >_N [y]\}$ where $(u, v) \in E$ is directed from u to v. $G_{>_N}$ is referred to as the *nesting graph* of S, and its transitive reduction $G^R_{>_N}$ is referred to as the *reduced nesting graph* of S. □

Proposition A.8.8. *The nesting graph of a set system is a directed acyclic graph (i.e., $G_{>_N}$ is a DAG).*

Proof. Let S be a set system on the items X.

By Prop. A.8.6, the nested relation is a strict partial order on the elements of X/L; therefore, its corresponding poset graph $G_{>_N}$ is a DAG. □

Proposition A.8.9. *The reduced nesting graph of a set system is a forest of rooted trees where the each tree's edges are directed away from its root.*

Proof. Let S be a set system on the items X and let $G^R_{>_N}$ be its reduced nesting graph.

Suppose $G^R_{>_N}$ has a vertex with more than one edge directed towards it; that is, there exist vertices $[x], [y], [z] \in X/L$ such that $[x] >_N [z]$ and $[y] >_N [z]$. By Prop. A.8.7, either $[x] >_N [y]$ or $[y] >_N [z]$. Without loss of generality, let $[x] >_N [y]$. As a result, there exist directed paths $[x] \to [y] \to [z]$ and $[x] \to [z]$, which contradicts the fact that $G^R_{>_N}$ is a transitive reduction; therefore, each vertex of $G^R_{>_N}$ has at most one edge directed towards it. □

A.8.1 Mathematical Background

Proposition 8.1.2. *Let S be a set system on the items X and C be an Euler diagram representing S. For any two items $x, y \in X$, the corresponding curves $c_x, c_y \in C$ transversely intersect if and only if xOy.*

Proof. Sufficiency Since xOy, let s_{xy}, s_x, and s_y be sets that satisfy Def. 8.1.2. Since $x \in s_x$ and $y \notin s_x$, we must have $int(c_x) \nsubseteq int(c_y)$. Similarly, due to s_y we must have $int(c_y) \nsubseteq int(c_x)$. In addition, since $x, y \in s_{xy}$, we must have $int(c_x) \cap int(c_y) \neq \emptyset$. As a result, since $int(c_x)$ is connected and $int(c_y)$ is connected, they must overlap somewhat like what is shown in shown in Fig. A.21. Clearly, there must be a segment of c_x that is exterior to c_y and a segment of c_x that is interior to c_y, and since c_x

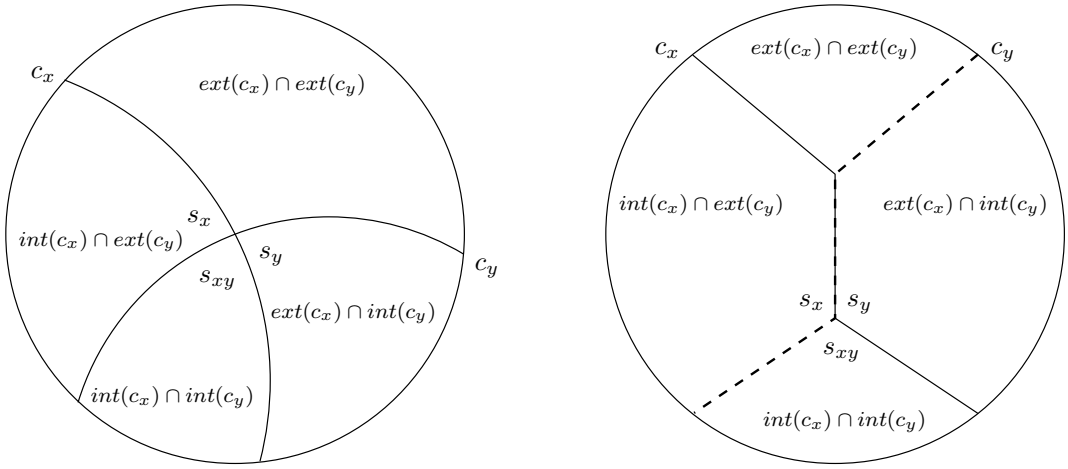


Figure A.20: In both transverse (a) point and (b) concurrent intersections of two curves c_x and c_y, there are regions representing sets s_{xy}, s_x, and s_y that satisfy Def. 8.1.2.

is continuous, there is either a point intersection or a concurrent intersection that satisfies the transverse property of Def. 1.2.8.

Necessity Figure A.20 shows the two possible types of transverse intersections, point and concurrent, between curves c_x and c_y; note that there may be additional concurrent curves involved in the intersections, but we focus on the interaction of c_x and c_y. As shown by the labels, it is clear that within the immediate neighbourhood of the point intersection or one of the endpoints of the concurrent intersection, the interiors/exteriors of c_x and c_y combine to create the regions representing sets s_{xy}, s_x, and s_y that satisfy the definition of xOy. □

Theorem 8.1.3. *Let S be a set and C be an Euler diagram representing S. C is prime if and only if S is linked.*

Proof. Sufficiency (by contrapositive) Let C be composite and let C' be a subset of

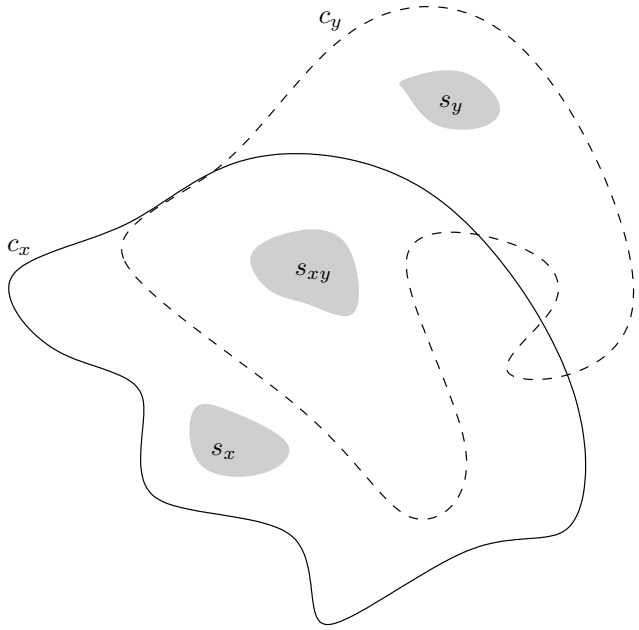


Figure A.21: If items x and y overlap, their corresponding curves, c_x and c_y must have overlapping interiors; as a result, c_x and c_y have at least two transverse intersections (either point or concurrent, or both).

C's curves as defined in Def. 8.1.1. Since C' is a non-empty proper subset of C, there exist curves $c_x \in C'$ and $c_y \in C \backslash C'$ representing items x and y, respectively. Suppose xLy, then by Def. 8.1.3, there exists a sequence $x = x_1, x_2, \ldots, x_k = y$ of items such that $x_i O x_{i+1}$ for all $1 \leq i < k$. Since $c_x \in C'$ and $c_y \in C \backslash C'$, there must be some i such that $c_{x_i} \in C'$ and $c_{x_{i+1}} \in C \backslash C'$; therefore, since $x_i O x_{i+1}$, by Prop. 8.1.2, c_{x_i} transversely intersects $c_{x_{i+1}}$. However, by Def. 8.1.1, C' is contained within a single connected plane subset of $C \backslash C'$, so there can be no transverse intersections between curves in C' and curves in $C \backslash C'$, thus contradicting the transverse intersection of c_{x_i} and $c_{x_{i+1}}$. So $x \not L y$ and S is unlinked.

Necessity (by contradiction) Let S be a set system on the items X, and let C be a prime Euler diagram representing S. Suppose S is unlinked; that is, there are items $x, y \in X$ such that $x \not L y$. Let $[x]$ be the subset of items that are linked to x, including x (since the linked relation is irreflexive). Since $x \in [x]$ and $y \notin [x]$, $[x]$ is a non-empty proper subset of X.

Let $[c_x]$ be the curves corresponding to $[x]$, then $[c_x]$ is a non-empty proper subset of C. Since the linked relation is transitive, no item in $[x]$ is linked to an item in $X \backslash [x]$, so there is no item $x' \in [x]$ and $y' \in X \backslash [x]$ such that $x' O y'$. As a result, by Prop. 8.1.2, no curve in $[c_x]$ transversely intersects a curve in $C \backslash [c_x]$; therefore, $[c_x]$ must be contained in a single plane subset of $C \backslash [c_x]$ implying the contradiction that C is composite, so S must be linked. □

A.8.2 Factorization Algorithm

T_X Generation

Theorem 8.2.1. *For a set system S on the items X, the running time of Alg. 1 is $O(|X|^2|S| + |X|^3)$.*

Proof. We represent S by a linked list of integers; each integer is a bitwise representation of a subset of $|X|$. For example, the binary integer 01001 represents the subset $\{1, 4\}$ and 11101 represents the subset $\{1, 3, 4, 5\}$. Using this representation, we can determine whether a subset contains a particular item in constant time via bitwise operations. The following is a line-by-line analysis of the running time of Alg. 1.

Line 1. We use an adjacency matrix representation for G. An entry for an edge (x, y) is an integer representing bitwise flags whose combination indicates the edge's orientation. Since G has $|X|$ vertices, the running time of this step is $O(|X|^2)$.

Lines 2–5. The outer loop is performed $|S|$ times. The inner loop can be implemented by a double loop iterating over the $|X|$ bits of each set. Since adding an edge to G is a constant-time operation, the running time of these steps is $O(|S||X|^2)$.

Lines 6–11. The outer loop is performed $|X|^2$ times by checking in constant time each entry in G's adjacency matrix to determine whether it represents an edge. The body involves a scan of S, which requires $O(|S|)$ time since S is represented by a linked list. Since determining whether or not a set contains an item and updating an edge's direction are constant-time bitwise operations, the running time of these steps is $O(|X|^2|S|)$.

Lines 12–15. Like in the previous steps, the outer loop is performed $|X|^2$ times. Removing a vertex is an $O(|X|)$ operation since its respective row and column in G's

adjacency matrix need to be cleared. Since at most $|X|$ vertices are removed, and since relabeling is a constant-time operation using a lookup table, the running time of these steps is $O(|X|^2)$.

Lines 16–17. The running time for computing the strongly connected components of a graph with an adjacency matrix representation is $O(V^2) \equiv O(|X|^2)$ [8]. We are interested in knowing which items are in each SCC and which SCC contains a given item; for the former, we treat each SCC as a set and use a bitwise representation, and for the later, we use a lookup table. Since there are at most $|X|$ SCCs, and each SCC is represented by $|X|$ bits, the running time for initializing the SCC representations and lookup tables is $O(|X|^2)$. Lastly, since G' can have up to $|X|$ vertices, the running time for its initialization using an adjacency matrix is $O(|X|^2)$; therefore, the running time of these steps is $O(|X|^2)$.

Lines 18–23. The outer loop is performed $|X|^2$ times. Determining the SCC for x and y is a constant-time operation using a lookup table. Determining whether an edge is in G''s adjacency matrix is a constant-time operation, as is adding an edge. Since a bitwise representation is used for edge labels, adding an item is a constant-time bitwise operation; therefore, the running time of these steps is $O(|X|^2)$.

Lines 24–27. When G' is initialized, an extra $\emptyset$ vertex can be added; this does not change the running time of the previous operations. Determining whether a vertex is a source vertex requires an $O(|X|)$ scan of that vertex's row or column in the adjacency matrix. Since there are at most $|X|$ source vertices, and adding an edge to G' is a constant-time operation, the running time of these steps is $O(|X|^2)$.

Line 28. Since G' is a directed acyclic graph (DAG), computing its transitive reduction is an $O(V^3)$ operation [22]; therefore, the running time of this step is

$O(|X|^3)$.

From the above analysis, the running time of the algorithm is $O(|X|^2|S| + |X|^3)$.

□

T_S Generation

Theorem 8.2.2. *For a set system S on the items X, the running time of Alg. 2 is $O(|X|^2 + |X||S|)$.*

Proof. We represent S and T_X as in the proof of Thm. 1. The following is a line-by-line analysis of the running time of Alg. 2.

Line 1. T_X is represented by at most an $|X|x|X|$ adjacency matrix, so initializing T_S's adjacency matrix and copying T_X is an $O(|X|^2)$ operation. Since T_S's vertices are initially labeled with bitwise integers representing subsets of X, copying these into a lookup table is an $O(|X|)$ operation; therefore, the running time for these steps is $O(|X|^2)$.

Lines 2–7. We represent S' as a linked list of bitwise integers, so copying S to S' and omitting $\emptyset$ is an $O(|S|$ operation. Since T_S has at most $|X|$ vertices and for each vertex we need to examine its $|X|$ potential edges, determining the vertex postorder is an $O(|X|^2)$ operation. Once the ordering is determined, the outer loop is performed $|X|$ times. We represent l as a bitwise integer with $|X|$ bits to representing the union of the edges labels, which are subsets of X; l is initialized to 0 to indicate $\emptyset$. As the traversal progresses, l can be updated in constant time by a bitwise OR (on descent) or AND (on ascent) with the current edge's label. The subset of S' comprising sets involving only items from v and l can be determined by an $O(|S|)$ scan of S' and constant-time bitwise operations. If we represent the new label for v with a linked

list, then each set can be removed from S' and added to v's label in constant time. So the running time for the loop body is $O(|S|)$; therefore, the running time for these steps is $O(|X|^2 + |X||S|)$.

Lines 8–13. The outer loop is performed $O(|X|^2)$ times. Since x's set system is represented by a linked list, determining whether $l \in x$ is an $O(|S|)$ operation; however, since the vertices of T_S partition S, over *all* vertices, the scan is $O(|S|)$. Since updating the edge's label is a constant-time operation using a lookup table and adding l to x's set system is a constant-time operation, the running time for these steps is $O(|X|^2 + |S|)$.

From the above analysis, the running time of the algorithm is $O(|X|^2+|X||S|)$. □

A.9 ω-Proportional Euler Diagrams with the Fullset

A.9.1 Mathematical Background

Rather than developing our results in terms of directed Euler graphs and directed Euler duals, we take a more general strategy and develop the results in terms of directed plane graphs and their directed plane duals; directed Euler graphs and directed Euler duals then become a special case.

Definition A.9.1. Let $\vec{G}$ be a connected directed plane graph. The directed plane dual $\vec{G}^*$ is the plane dual of $\vec{G}$'s intrinsic undirected plane graph where for each edge $e \in \vec{G}$, the corresponding dual edge $e^* \in \vec{G}^*$ is oriented from left-to-right across e (i.e., if $e^* = (u, v)$ then u and v are the vertices representing the left and right faces

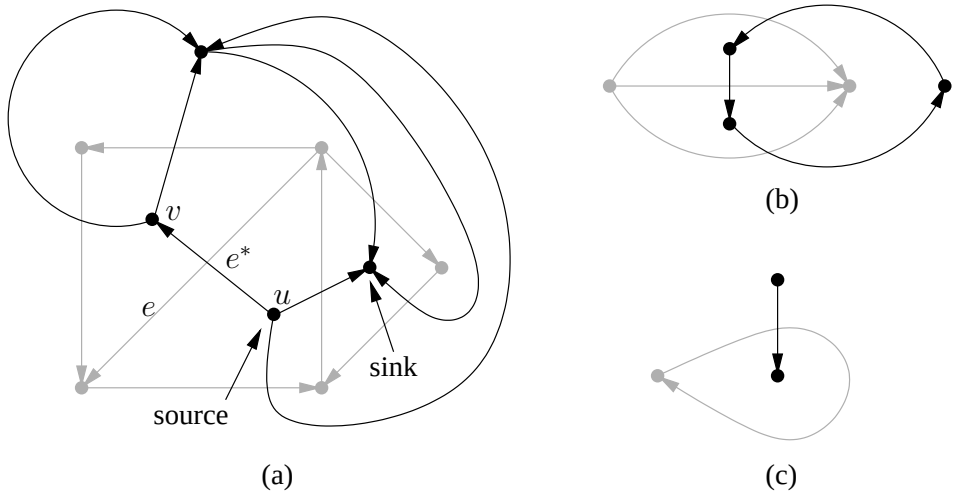


Figure A.22: Examples of directed plane graphs (light gray) with their corresponding directed plane duals (black); note how in (a) the dual edge e^* is directed from the left to right faces of its corresponding edge e.

of e, respectively). □

Example. Figure A.22 shows some examples of directed plane graphs (light gray) and their corresponding directed plane duals (black). Notice how in Fig. A.22(a), a plane graph face whose boundary edges form a directed cycle corresponds to a dual vertex that is either a source or sink depending on the cycle's orientation; this is a general feature of directed plane duals. □

Proposition A.9.1. *Let $\vec{G}$ be a connected directed plane graph with a single source vertex s and a single sink vertex t. If p is a path from s to t and p^* is the set of dual edges in $\vec{G}^*$ corresponding to p, then $\vec{G}^* \backslash p^*$ is a directed acyclic graph (DAG).*

Proof. Clearly, if $\vec{G}^*$ is acyclic then $\vec{G}^* \backslash p^*$ is also acyclic, and thus a DAG.

So suppose $\vec{G}^*$ has a directed cycle $\vec{C}$. Since $\vec{G}^*$ is a plane graph, $\vec{C}$ can have one of two possible orientations as shown in Fig. A.23 (this is a schematic overlay of $\vec{G}$ and $\vec{G}^*$ much like Fig. A.22(a) except that unnecessary detail is grayed out).

Consider the orientation of $\vec{C}$ in Fig. A.23(a). The edges of $\vec{G}$ whose corresponding dual edges form $\vec{C}$ are directed as shown by the dashed edge; consider such an edge $e = (u, v) \in \vec{G}$. Since $\vec{G}$ has a single source s, there is a path from s to u, and since u is interior to $\vec{C}$, s must also be interior since no edges of $\vec{G}$ lead from vertices that are exterior to $\vec{C}$ to vertices that are interior to $\vec{C}$. Similarly, since $\vec{G}$ has a single sink t, there is a path from v to t, and since v is exterior to $\vec{G}$, t must also be exterior to $\vec{G}$. Figure A.23(a) shows the locations of s and t relative to $\vec{C}$ (and remember that $s, t \in \vec{G}$ while $\vec{C} \in \vec{G^*}$).

The same argument applies to the orientation of $\vec{C}$ in Fig. A.23(a), except that the relative locations of s and t are reversed. As a result, any path p in $\vec{G}$ from s to t must cross an edge of $\vec{C}$; therefore, the corresponding dual edges p^* contain an edge of $\vec{C}$, and thus $\vec{C}$ is cut in $\vec{G^*} \backslash p^*$.

Since this argument applies to all cycles in $\vec{G^*}$, $\vec{G^*} \backslash p^*$ is acyclic, and thus a DAG.

□

Example. Figure A.24(a) shows a directed plane graph $\vec{G}$ (light gray) and its corresponding directed plane dual $\vec{G^*}$ (black). $\vec{G}$ has a single source s and a single sink t. Note how the two dashed cycles in $\vec{G^*}$ are examples of the orientations from Fig. A.23 (there are also several additional cycles), and how s and t are on opposing sides of every cycle.

Figure A.24(b) shows a path in $\vec{G}$ from s to t (dashed edges), and the result of removing its corresponding dual edges from $\vec{G^*}$. Note how every cycle in $\vec{G^*}$ is cut; therefore, what remains of $\vec{G^*}$ is a DAG. □

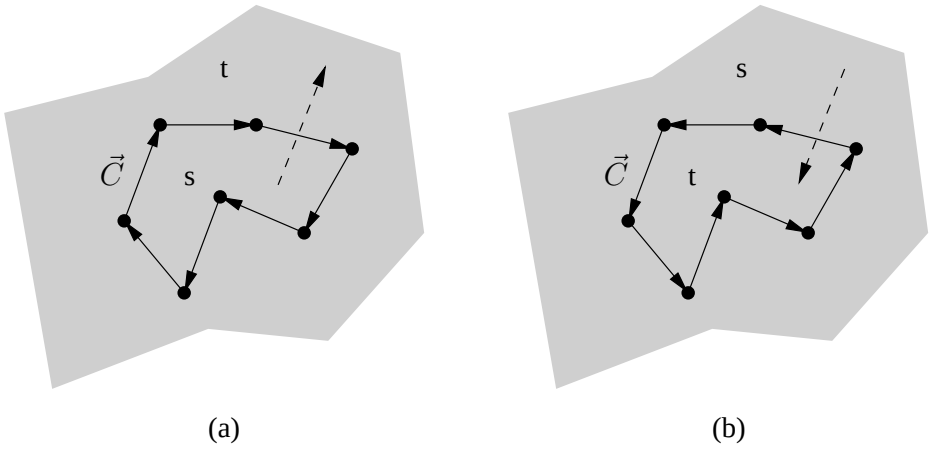


Figure A.23: (a) One possible orientation of a directed cycle $\vec{C}$ in a directed plane dual implies that the corresponding graph's source s must be inside the cycle and the sink t outside, while (b) the other possible orientation implies the opposite. In either case, a path from s to t crosses $\vec{C}$ exactly once.

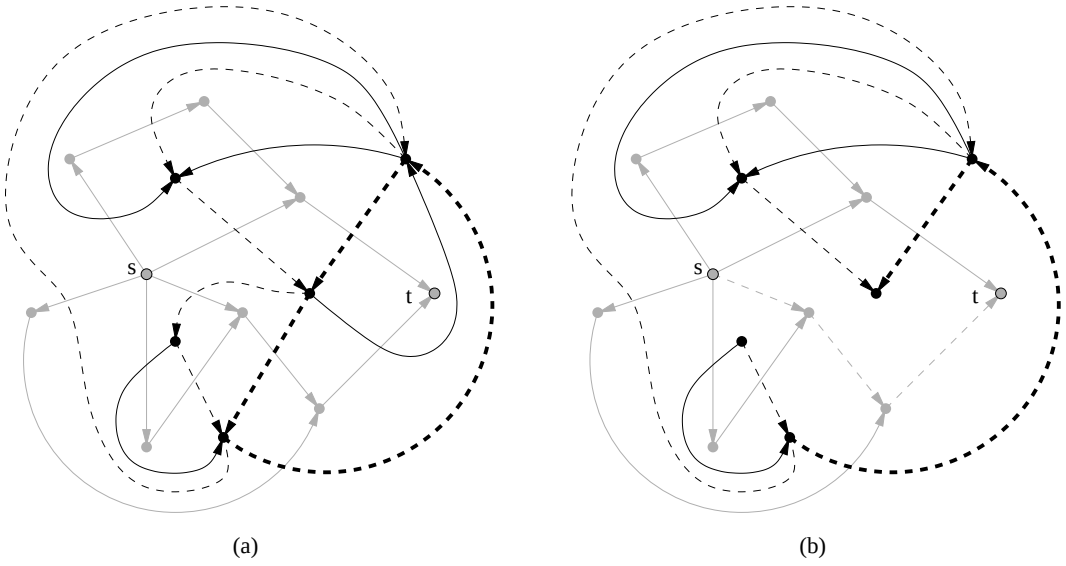


Figure A.24: (a) A directed plane graph $\vec{G}$ (light gray) with a single source s and a single sink t and its corresponding directed plane dual $\vec{G^*}$ (black); two of the (several) cycles in $\vec{G^*}$ are shown with dashed black edges. (b) Removing the edges of $\vec{G^*}$ that correspond to a path (dashed light gray) in $\vec{G}$ from s to t results in a DAG (black edges).