An Incremental Construction Algorithm for Venn Diagrams

J Eloff L van Zijl

Department of Computer Science, University of Stellenbosch, Stellenbosch, South Africa,
{eloff, lynette}@cs.sun.ac.za

Abstract

The drawing of Venn diagrams have traditionally served as an aid to analyze the characteristics exhibited by the diagram and its underlying Venn graph. Little attention has hitherto been given to the presentation, construction and way in which a Venn diagram conveys information. We present a new algorithm for the construction and layout of a Venn diagram based on an incremental technique. The algorithm constructs the Venn diagram by incrementally deriving new intersections from previously placed ones on a grid. We also propose criteria which can be applied to analyze a Venn diagram's drawing and layout. Our algorithm aims to generate an aesthetically pleasing layout by satisfying the proposed criteria.
Keywords: *Venn Diagrams, Graph Drawing, Layout*
Computing Review Categories: *F.2.2, G.2.2*

1 Introduction

Many computerized design and modeling systems are based on Venn diagrams [6]. Software engineering applications, such as STATEMATE [5], make use of higraphs to model statecharts. These higraphs are extended from Venn diagrams [7] by introducing a hierarchical structure. Figure 1 is an illustration of a simple higraph

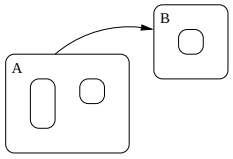

Figure 1: A simple Higraph

It is important that efficient algorithms to generate Venn diagrams are found given the application of visual formalisms based on them. Research has for the most part focussed on the graph theoretic properties exhibited by Venn diagrams and their associated Venn graphs [3, 6] or their application to model other graph structures such as hypergraphs [8].

We propose an algorithm for the construction and layout of Venn diagrams. The majority of criteria used to evaluate graph drawings cannot be applied to Venn diagrams in general. We formulate and propose a set of criteria that can be applied to analyze the drawing of a Venn diagram. Our algorithm aims at satisfying the proposed criteria to generate an aesthetically pleasing layout.

Section 2 presents an overview of different techniques that can be applied in the construction of Venn diagrams. Section 3.1 discusses some preliminary definitions before the algorithm is presented in Section 3.2. Section 4 presents a proof outline for our algorithm and discusses the new criteria for evaluating Venn diagrams. We then apply the criteria in analyzing the drawings and layouts produced by our algorithm. We conclude our discussion in Section 5 by presenting a summary.

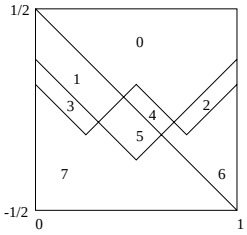

Figure 2: A 3-Venn diagram constructed with the *zigzag* function containing 3 curves and 8 regions

2 Techniques for Constructing Venn Diagrams

Algorithms such as the *zigzag* function [4] can be used to construct a general Venn diagram. Figure 2 is an example of a 3-Venn diagram constructed with this function.

The regions in the diagram are labelled by using a binary representation to describe the inclusion of sets for a given region. Each set is assigned a unique binary digit (bit) and the regions are labelled in decimal for simplicity. For example, the region labelled as 5 in Figure 2 describes the intersection of sets 1 and 4.

The *zigzag* function confines the Venn diagram within a unit square making it difficult to interpret the diagram, especially when higher order diagrams are constructed. Figure 3 shows the individual regions for each set of the 3-Venn diagram after they were extracted from the drawing in Figure 2. The construction of the individual regions for higher order diagrams will require many computations

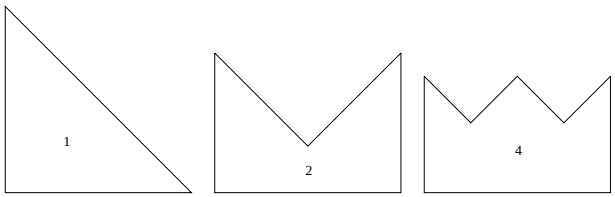

Figure 3: Individual regions extracted from a 3-Venn diagram constructed with the *zigzag* function

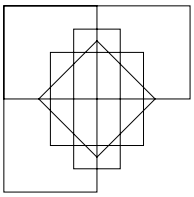

Figure 4: Boyd's construction for a 5-Venn diagram using rectangles

to determine the various points at which regions intersect with one another.

Various other techniques exist for the construction of Venn diagrams, often imposing a limit on the size of the Venn diagram. Boyd's construction for a 5-Venn diagram using rectangles is illustrated in Figure 4 [2]. Confining Venn diagrams to specific geometric shapes effectively limits the possible constructions and requires that other means are found to extend the drawing to a higher order diagram. These extensions often break the symmetry introduced by the geometric shapes. Figure 5 shows a construction known as a Clown Venn diagram. The drawing is primarily composed of ellipses [6]. The fifth set is constructed from an irregular shape suggesting that a complex algorithm was used to generate the diagram. The rectangular shape of the fifth set breaks the symmetry originally introduced by the ellipses.

3 The Incremental Construction Algorithm

The Incremental Construction algorithm constructs a Venn diagram on a two- dimensional grid representing the plane. A label on the grid either describes a region containing a number of intersections of the Venn diagram or the exterior plane.

A construction which we call a Tri is initially placed on the grid and forms the basis for the layout of the Venn diagram. The Tri contains $\frac{n(n+1)}{2}$ intersections of the *n*-Venn diagram. The remaining intersections are then derived from the Tri and placed on the grid. Various functions are used throughout the construction algorithm. An overview of these functions will be presented before examining the construction algorithm itself. Various other algorithms are used during and after the construction of the Venn diagram. These algorithms are presented after the main algorithm.

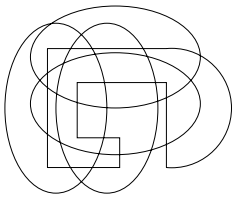

Figure 5: The Clown Venn diagram construction

3.1 Preliminary Definitions

A two-dimensional, infinite square grid G is used for the construction and layout of the Venn diagram and represents the plane. A position on the grid is given by $G[j,i]$ such that $0 \leq i, j \leq G_s - 1$ where G_s is the maximum number of rows and columns on the grid and j and i denote the row and column respectively. Positions on the grid are numbered from 0 to $G_s - 1$ row-wise, starting at the top left corner. The label contained at a specific position on the grid is given by $l = G[j,i]$ such that $0 \leq l \leq 2^n - 1$. If $G[j,i] = 0$, the label is a reference to the exterior plane, otherwise it describes a region of the Venn diagram containing one or more intersections. Initially all positions in G reference the exterior plane so that the following condition holds after the initialization of G:

$$G[j,i] = 0 \text{ for } 0 \leq j,i \leq G_s - 1.$$

Regions are labelled with integer values and each set in the Venn diagram is associated with a unique binary digit (bit). Each bit in the binary representation of a label that is set to 1 describes the inclusion of its associated set in that region. For example, the label $9(1001_2)$ describes the intersection of two sets since bit 0 and 3 are set to 1. For simplicity we will map each bit to an alphabetical character so that $1(1_2) \rightarrow A$, $2(10_2) \rightarrow B$, $4(100_2) \rightarrow C$, etc. A region labelled as 3 has a binary representation of 11_2 and will be referred to as AB.

The number of sets describing an intersection is given by the composition function $C(x)$ where x is the label of the region on the grid and $0 \leq x \leq 2^n - 1$. We define $C(x)$ as the number of 1s in the binary representation of x. For example, if $G[j,i] = 5$ then $C(G[j,i]) = 2$ since $5_{10} = 101_2$.

The label set L is used to keep track of labels that must still be placed on the grid. Initially $L = \{l \in \mathbb{N} \mid 1 \leq l \leq 2^n - 1\}$ for an *n*-Venn diagram. Whenever a label is placed on the grid it is removed from L . The construction algorithm terminates when $L = \emptyset$.

We define

$$\text{Max}(L) = \begin{cases} 0 & \text{if } L = \emptyset \\ l & \text{if } L \neq \emptyset \end{cases}$$

where l is the least element of L such that $C(l)$ is maximal on L. For example, if $L = \{3,5,7,9\}$, then Max $= 7$.

Finally, we define $\wedge$ and $\vee$ to be the bitwise AND and OR operators respectively. For example, $13 \wedge 12 = 12$ and $7 \vee 9 = 15$.

3.2 Constructing the Venn Diagram

We now present the main algorithm for constructing the Venn diagram. An example is presented in Section 3.3 to illustrate how the algorithm works.

Algorithm 1

Algorithm Construct
Input n, the size of the Venn diagram
Output A grid G containing the Venn diagram

1. Initialize L and the grid.
2. Construct the initial layout:
   (a) Let $i = n$
   (b) For $j = 1 \ldots n$ Do:
      i. $G[j,i] = 2^{j-1}$
      ii. $L = L - \{2^{j-1}\}$
      iii. $i = i - 1$
3. Construct the Tri:
   For $i = 1 \ldots G_s - 2$ and $j = 1 \ldots G_s - 2$ Do:
   If $G[j-1,i] \neq 0$ and $G[j,i-1] \neq 0$ Then
   (a) $G[j,i] = G[j-1,i] \vee G[j,i-1]$
   (b) $L = L - \{G[j,i]\}$
4. While $L \neq \emptyset$ Do
   (a) Let $l = Max$ and $c = C(l)$
   (b) For $j = 1 \ldots G_s - 2$ and $i = 1 \ldots G_s - 2$ and $l \neq 0$ Do:
   If IsValid(j,i,l) Then
      i. $L = L - \{l\}$
      ii. $G[j,i] = l$
      iii. $l = 0$
   (c) For $j = 1 \ldots G_s - 2$ and $i = 1 \ldots G_s - 2$ Do:
   If $G[j,i] = 0$ and $G[j-1,i] \neq 0$ and $G[j,i-1] \neq 0$ Then
      i. $G[j,i] = G[j-1,i] \vee G[j,i-1]$
      ii. If $G[j,i] \in L$ Then $L = L - \{G[j,i]\}$

Algorithm 2

Algorithm IsValid
Input (j,i) is a coordinate pair on the grid. l is the label that must be placed.
Output The function returns TRUE if the position is valid for placing l or FALSE otherwise.

1. Let Valid = FALSE
2. Let $v = G[j,i-1] \vee G[j,i+1] \vee G[j-1,i] \vee G[j+1,i]$
3. If $v \wedge l = l$ and $G[j,i] = 0$ Then Valid = TRUE
4. Return Valid

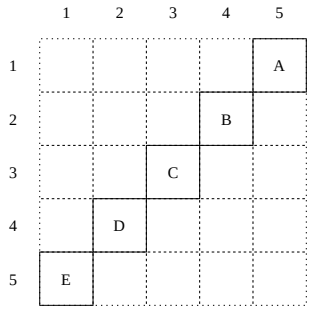

Figure 6: Initial construction for a 5-Venn diagram

Algorithm 3

Algorithm Optimize
Input A grid G containing a layout of a Venn-Diagram
Output A grid G' containing an optimized layout of G
For $j = G_s - 2 \ldots 2$ and $i = G_s - 2 \ldots 2$ Do
If $G[j,i] = 2^n - 1$ Then

1. Let $G[j,i] = 0$
2. If not (IsValid $(j-1,i,G[j-1,i])$ and
   IsValid $(j+1,i,G[j+1,i])$ and
   IsValid $(j,i-1,G[j,i-1])$ and
   IsValid $(j,i+1,G[j,i+1])$)
   then $G[j,i] = 2^n - 1$

3.3 Illustrative Example: A 5-Venn Diagram

We now present an example to illustrate how the algorithm works. A 5-Venn diagram will be constructed on a grid with $G_s = 25$. Both G and L are initialized in step 1 so that G contains only 0s and $L = \{1,2,\ldots,31\}$.

The initial layout is made in step 2b to form a diagonal construction on the grid. The first label will be placed at $G[1,5]$, the second at $G[2,4]$ and so on until $j = 5$. After the completion of step 2b the grid contains the diagonal construction illustrated in Figure 6.

Step 3 now proceeds by constructing the Tri from the diagonal. The Tri is constructed by deriving new labels from previously placed labels. A new label at $G[j,i]$ is derived by examining the labels at $G[j-1,i]$ and $G[j,i-1]$. The first position in our example where a new label can be derived is located at $G[2,5]$. The algorithm examines the labels at $G[1,5]$ and $G[2,4]$ (containing the values 1 and 2 respectively) and combines them to form a new label at $G[2,5]$ by computing their bitwise OR. The process is repeated for each position on the grid where $G[j-1,i] \neq 0$ and $G[j,i-1] \neq 0$ and is illustrated in Figure 7. Figure 8 shows the completed Tri.

We will now examine the first few iterations of step 4 which will place the remaining labels in L on the grid to complete the Venn diagram. When step 4 is reached, $L = \{5,9,10,11,13,17,18,19,20,21,22,23,24,25,26,27,29\}$. After step 4a has been executed, $l = 23$ and $c = 4$. Step 4b now searches through the grid to locate a valid position for placing l. When a valid position is found for l it is removed from L, $G[j,i] = l$ and $l = 0$. The first position

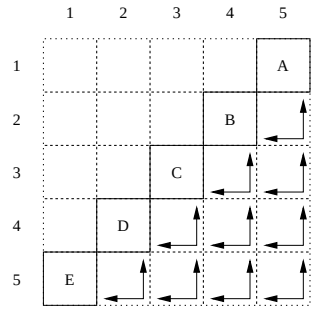

Figure 7: Construction of the Tri. New labels are derived by examining the labels located to the top and left of the grid position under investigation

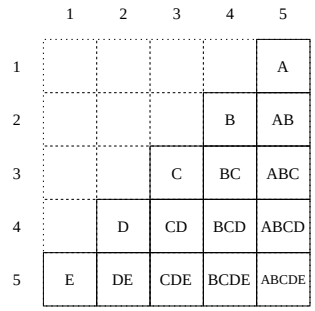

Figure 8: Completed Tri for a 5-Venn diagram

that will be located during step 4b in our example is $G[5,6]$. Figure 9 shows the grid after placing l. Since $l = 0$, step 4b now terminates and the algorithm proceeds to step 4c.

Step 4c searches the grid and attempts to derive new intersections by applying the same method used for the construction of the Tri in step 3. No new labels will be generated at this stage and the algorithm returns to step 4a. Upon the second iteration, $l = 27$, $c = 4$ and the first valid position is located at $G[4,6]$. Figures 10 and 11 show the grid after the completion of step 4b during the second and fifth iteration of step 4 respectively.

During the fifth iteration step 4c will generate two labels ($G[4,7] = 31$ and $G[5,7] = 31$). Step 4c will in most cases generate a label equal to $2^n - 1$. This is attributed to the fact that when step 4c is executed the labels at $G[j-1,i]$ and $G[j,i-1]$ have a high composition and by combining them, a label with an even higher composition will be formed. It is acceptable for step 4c to generate labels that have already been placed. However, a special test is included to ensure that the new label is removed from L should $G[j,i] \in L$.

Step 4 continues until $L = \emptyset$. The completed 5-Venn diagram is illustrated in Figure 12. The final diagram can now be optimized to reduce its area. Algorithm 3 attempts to remove redundant occurrences of labels equal to $2^n - 1$. Since the Venn diagram is constructed from the top left corner of the grid and advances towards the bottom right corner, the optimization algorithm starts at the bottom right corner and moves toward the top left corner.

When a grid position $G[j,i] = 2^n - 1$ is encountered it

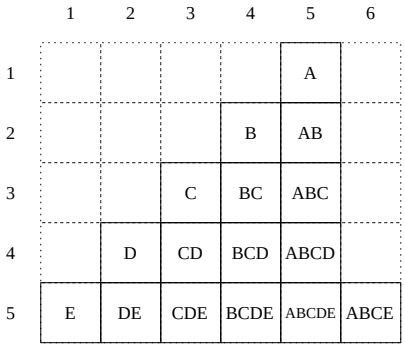

Figure 9: The grid after the first iteration of step 4b

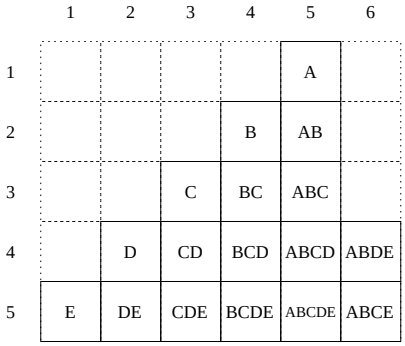

Figure 10: The grid after the second iteration of step 4b

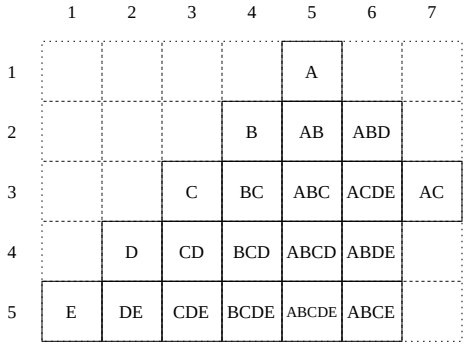

Figure 11: The grid after the fifth iteration of step 4b

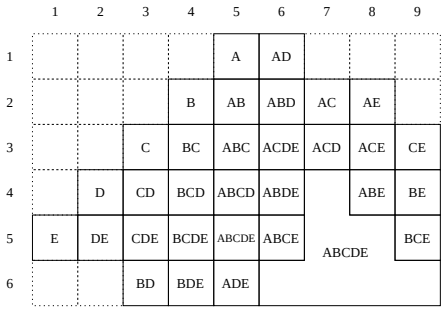

Figure 12: The completed 5-Venn diagram

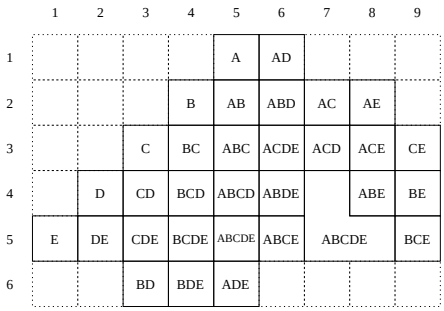

Figure 13: Optimized 5-Venn diagram after applying Algorithm 3 to the Venn diagram in Figure 12

is set to 0 in step 1. The positions around $G[j,i]$ are then tested in step 2 by using Algorithm 2 to determine if they are still valid given the removal of the label at $G[j,i]$. If the surrounding labels are no longer valid then $G[j,i]$ is set to 2^n-1 again. Figure 13 shows the 5-Venn diagram after applying Algorithm 3.

It is important to note that there are two disjoint areas labelled ABCDE in Figure 12. Although this is perhaps an undesirable layout, it is acceptable. Disjoint regions are allowed for composite labels as long as the region for each set in the composite label is continuous. Figure 14 shows an unacceptable layout where there are two disjoint regions for A.

4 Analysis

We begin by presenting an argument for the correctness of our algorithm and sketching an outline for a formal proof. Criteria for evaluating the drawing of Venn diagrams and their construction algorithms are presented and applied in the analysis of our algorithm.

4.1 Correctness of the Algorithm

Algorithm 1 is essentially a graph layout algorithm. We cannot compare different layouts for an *n*-Venn diagram produced by the same algorithm and state that one is correct while the other is incorrect. Figure 15 illustrates two layouts for a 5-Venn diagram. The top diagram was constructed on a grid with $G_s = 11$ while the diagram at the bottom was constructed on a grid with $G_s = 25$. Both layouts are acceptable and valid. The different layouts can be directly attributed to the different values of G_s that were used.

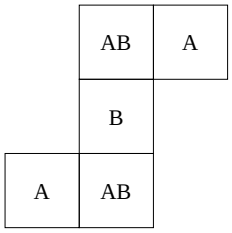

Figure 14: An example of an unacceptable layout forming two disjoint regions labelled as A

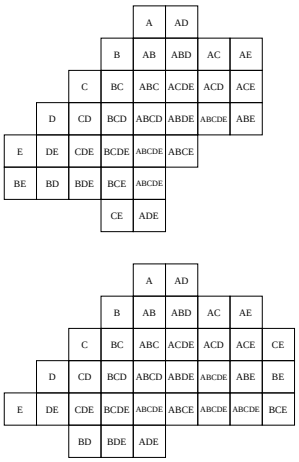

Figure 15: Two 5-Venn diagrams for different values of G_s

We can define certain conditions that must be satisfied to produce a valid layout. These conditions are:

1. A label may only be placed if its placement ensures that a continuous area for each individual set described by the label is formed. An example of an unacceptable layout which breaks this condition is illustrated in Figure 14 where the individual set A is described by two disjoint regions.

2. A layout is valid if, and only if, all the labels initially in L have been placed on the grid and their placement satisfies condition 1.

Condition 2 is the termination condition for the algorithm. Assuming that condition 1 is satisfied, the algorithm will terminate and the layout will be acceptable since condition 1 ensures that labels are placed at valid positions.

The Tri contains exactly $\frac{n(n+1)}{2}$ labels. In order to prove that the algorithm will terminate we must show that there are enough valid positions so that the remaining $2^n-1-(\frac{n(n+1)}{2})$ labels can be placed. We can now divide L into k subsets where $1 \leq k \leq n$. Initially $L_k = \{m \in \mathbb{N} \mid C(m) = k, 1 \leq m \leq 2^n-1\}$.

After the construction of the Tri, L_k will contain exactly t_k elements that must still be placed on the grid where $t_k = \binom{n}{k} - (n-k+1)$. For $n = 1$ the Tri will contain $\frac{1(1+1)}{2}$ labels which are equal to the total number of labels given

by $2^1 - 1$. This is also true for $n = 2$ and consequently the algorithm will terminate for $n = 1$ and $n = 2$.

We are currently focusing on the construction of a proof to show that for $n \geq 1$ and $n \in \mathbb{N}$ there are enough valid positions to place the remaining labels after the Tri has been constructed. To complete this proof we must also determine to what extend the value of G_s influences the number and location of valid positions. Our experimental results listed in Table 1 empirically shows that the algorithm will terminate for $1 \leq n \leq 14$, thus satisfying condition 2.

4.2 Criteria for Evaluating Venn Diagrams

We feel that the general criteria used to evaluate a graph drawing and its layout [1] cannot be directly applied to a Venn diagram. We propose the following criteria to be used for evaluating a Venn diagram's layout and its construction algorithm:

Intersection Visibility: Ideally one would like to be able to quickly and easily locate and interpret intersections. Having too many small intersections located around one another reduces visibility. Visibility can be further increased if one can manipulate the individual sets in the diagram.

Efficiency: Efficiency must address issues such as runtime and minimization of area to be considered for practical application in software.

Symmetry: The use of symmetrical shapes will lead to better intersection visibility although it may impact on the order of Venn diagrams that may be drawn.

4.3 Evaluation of the Incremental Construction Algorithm

Algorithm 1 is based on a mixture of grid and orthogonal drawing techniques as it contains vertical and horizontal segments placed on a grid. These segments influence the visibility of intersections since segments spanning a number of grid positions are far easier to interpret and locate than a segment contained within in a single grid position. The binary labelling mechanism employed by the algorithm makes it easy to manipulate groups of sets in the Venn diagram by applying a mask on the grid to isolate certain sets, thus increasing visibility. Figure 19 shows the individual sets for the 5-Venn diagram in Figure 12. Figure 16 shows a Venn diagram obtained from Figure 13 after applying a mask to isolate sets A and E.

We consider the algorithm to be efficient, although not optimal in terms of area. Table 1 shows the optimal area in grid positions for an n-Venn diagram compared to the actual area used by Algorithm 1. A grid of 1000x1000 was used to obtain the data in Table 1. The graph in Figure 17 clearly illustrates the exponential characteristics of the algorithm.

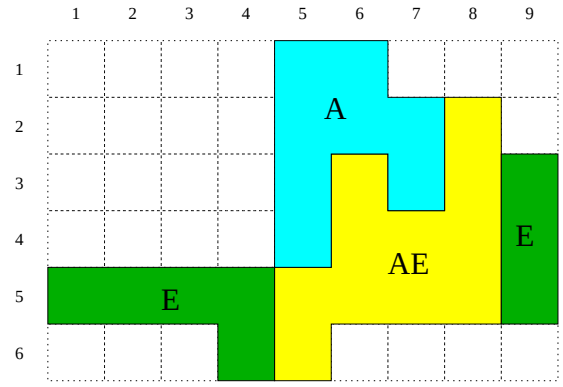

Figure 16: Venn diagram showing sets A and E together with the area where they intersect with one another

n	Optimal Area	Total Area	Optimized
1	1	1	1
2	3	3	3
3	7	7	7
4	15	16	16
5	31	38	34
6	63	84	73
7	127	172	156
8	255	411	366
9	511	1105	848
10	1023	1417	1289
11	2047	3648	2749
12	4095	5350	5314
13	8191	16178	12532

Table 1: Algorithm Performance

Our experiments have indicated that the area of the Venn diagram can be reduced by limiting the grid size. Reducing G_s can influence the total area used for constructing the Venn diagram. The size of the grid also impacts on the number of searches that must be carried out to locate a valid position. Table 2 shows the number of grid positions that were examined during the construction.

The shapes generated for the individual sets by our algorithm are not symmetrical at all as can be seen in Figure 19. We chose this approach to generate higher order Venn diagrams.

5 Conclusion

We presented a new algorithm for constructing Venn diagrams. We also proposed new criteria which can be applied in the analysis of Venn diagram construction algorithms and the layout produced by these algorithms. In doing so we have joined two fields of study by applying graph drawing concepts to construct and analyze Venn diagrams.

Empirical evidence seems to indicate that our algorithm is general and that the construction of a Venn diagram is limited only by the size of the grid which is used

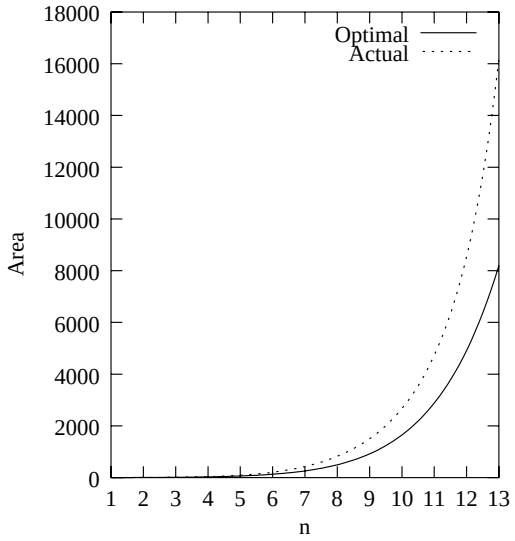

Figure 17: Optimal vs. actual area in terms of grid positions for different *n*-Venn diagrams

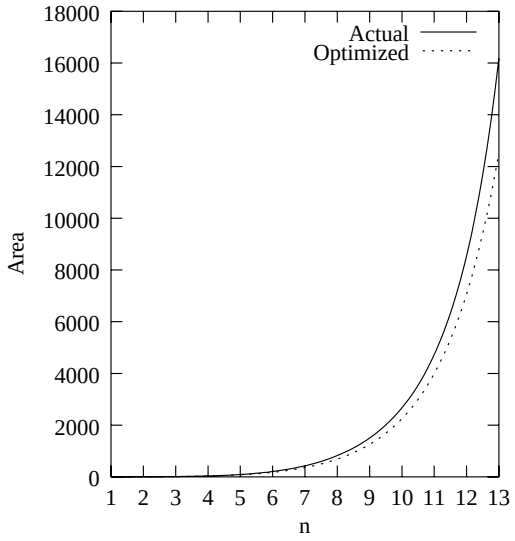

Figure 18: The actual area compared to the area after applying Algorithm 3 to remove redundant labels for different *n*-Venn diagrams

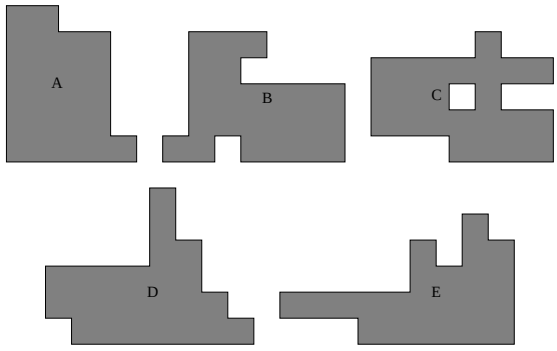

Figure 19: The shapes of the individual sets for the initial 5-Venn diagram in Figure 12

n	G_s	Number of Searches
1	1000	996004
2	1000	996004
3	1000	3992008
4	1000	15980044
5	1000	48976108
6	1000	127060312
7	1000	298986828

Table 2: Grid Positions Searched

during the implementation of the algorithm. Our analysis have also shown that the constructed diagram can be manipulated to extract certain information making it of practical use in applications. We are currently working on a proof to show that the algorithm is general and that it will terminate for all values of *n*.

References

[1] Giuseppe Di Battista et al. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall, 1999.

[2] A.V. Boyd. Venn diagram of rectangles. *Mathematics Magazine*, 58(4):251, 1985.

[3] Kiran B. Chilakamarri, Peter Hamburger, and Raymond E. Pippert. Venn Diagrams and Planar Graphs. *Geometriae Dedicata*, (62):73–91, 1996.

[4] J. Chris Fisher and Branko Grünbaum. Diagrams Venn and How. *Mathematics Magazine*, 61(1):36–40, 1988.

[5] Ornit Grossman and David Harel. On the algorithmics of higraphs. Technical Report CS97-15, Department of Applied Mathematics and Computer Science, The Weizmann Institute of Science, Rehovot, Israel, 1997.

[6] Peter Hamburger and Raymond E. Pippert. Simple, Reducible Venn Diagrams on Five Curves and Hamiltonian Cycles. *Geometriae Dedicata*, (68):245–262, 1997.

[7] David Harel. On visual formalisms. *Communications of the ACM*, 31(5):514–530, May 1988.

[8] D.S. Johnson and H.O. Pollak. Hypergraph planarity and the complexity of drawing venn diagrams. *Journal of Graph Theory*, 11(3):309–325, 1987.