Kent Academic Repository

Full text document (pdf)

Citation for published version

Dib, Fadi and Rodgers, Peter (2014) A Tabu Search Based Approach for Graph Layout. Journal of Visual Languages and Computing, 25 (6). pp. 912-923. ISSN 1045-926X.

DOI

https://doi.org/10.1016/j.jvlc.2014.10.019

Link to record in KAR

http://kar.kent.ac.uk/43502/

Document Version

Author's Accepted Manuscript

Copyright & reuse
Content in the Kent Academic Repository is made available for research purposes. Unless otherwise stated all content is protected by copyright and in the absence of an open licence (eg Creative Commons), permissions for further reuse of content should be sought from the publisher, author or other copyright holder.

Versions of research
The version in the Kent Academic Repository may differ from the final published version.
Users are advised to check **http://kar.kent.ac.uk** for the status of the paper. **Users should always cite the published version of record.**

Enquiries
For any further enquiries regarding the licence status of this document, please contact:
researchsupport@kent.ac.uk

If you believe this document infringes copyright then please contact the KAR admin team with the take-down information provided at **http://kar.kent.ac.uk/contact.html**

KAR Kent Academic Repository

University of Kent

Journal of Visual Languages and Computing ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

Journal of Visual Languages and Computing

journal homepage: www.elsevier.com/locate/jvlc

euler*Force*: Force-directed layout for Euler diagrams ☆

Luana Micallef [a,b,*], Peter Rodgers [a]

[a] *School of Computing, University of Kent, UK*
[b] *Helsinki Institute for Information Technology HIIT, Aalto University, Finland*

ARTICLE INFO

Article history:
Received 25 August 2014
Accepted 15 September 2014

Keywords:
Euler diagram
Venn diagram
Force-directed

ABSTRACT

Euler diagrams use closed curves to represent sets and their relationships. They facilitate set analysis, as humans tend to perceive distinct regions when closed curves are drawn on a plane. However, current automatic methods often produce diagrams with irregular, non-smooth curves that are not easily distinguishable. Other methods restrict the shape of the curve to for instance a circle, but such methods cannot draw an Euler diagram with exactly the required curve intersections for any set relations. In this paper, we present euler*Force*, as the first method to adopt a force-directed approach to improve the layout and the curves of Euler diagrams generated by current methods. The layouts are improved in quick time. Our evaluation of euler*Force* indicates the benefits of a force-directed approach to generate comprehensible Euler diagrams for any set relations in relatively fast time.

© 2014 Elsevier Ltd. All rights reserved.

1. Introduction

Euler diagrams can represent containment, exclusion and intersection among data sets using closed curves [10]. They are widely used in various areas (e.g., genetics [20]; ontologies [15]), and automatic diagram drawing techniques have been devised (e.g., [27,30]). A number of visual languages use Euler diagrams as a basis (e.g., Euler/Venn diagrams [32]; Venn-II diagrams [28]; constraint diagrams [18]; see survey [29]).

The closed curves facilitate reasoning about sets as they have a strong perceptual organizational effect on humans in dividing the space into regions and communicating memberships [23]. However, the curves have to be smooth and not too close to one another [2], highly symmetrical, and when possible, circles [3]. An Euler diagram should be *well-matched* [4], such that the regions in the diagram correspond exactly to the required set relations. If possible, an Euler diagram should also be *well-formed* [26], such that: each set is depicted by exactly one curve; each set relation is depicted by exactly one region; the curves are simple, non-concurrent and cross when they meet; and no point is on more than two curves. Nonetheless, generating an Euler diagram that satisfies all of these criteria is not always possible [24].

The well-matched diagrams produced by current methods (e.g., [27]) often have non-smooth, non-symmetric curves that are not easily distinguishable, as in Fig. 1. Other methods use circles to ensure curve smoothness and symmetry (e.g., [30]), but the generated diagrams are not well-matched and some of the regions might not correspond to any of the required set relations. Alternatively, some methods draw only well-formed Euler diagrams (e.g., [11]), but the curves are often non-smooth and a diagram cannot be drawn for all data. Also, the importance of different aesthetic criteria varies by context and data.

Using a layout method, the diagram is transformed into another that depicts the same set relations, but optimizes

☆ This paper has been recommended for acceptance by Shi Kho Chang.

* Corresponding author at: Helsinki Institute for Information Technology HIIT, Aalto University, Finland.
E-mail addresses: L.Micallef@kent.ac.uk (L. Micallef), P.J.Rodgers@kent.ac.uk (P. Rodgers).

http://dx.doi.org/10.1016/j.jvlc.2014.09.002
1045-926X/© 2014 Elsevier Ltd. All rights reserved.

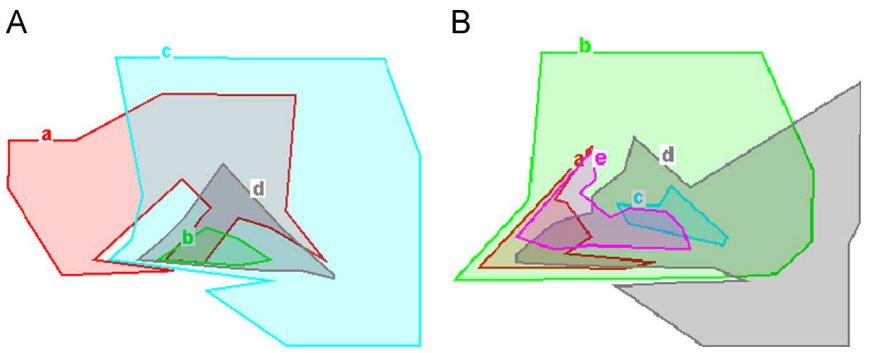

Fig. 1. Well-matched Euler diagrams generated by a drawing method [27].

specific aesthetic criteria. Two such methods, one by Rodgers et al. [25] and another by Flower et al. [14], have been proposed, but both are computationally expensive.

Rodgers et al. defined (but did not implement) a method that uses graph transformations to generate a layout that satisfies a particular well-formedness property [25]. However, this method does not take into account important curve aesthetics such as regularity, smoothness and symmetry and so, it cannot improve the layout of diagrams like those in Fig. 1, which are already well-formed. Graph transformations could also be computationally expensive [9].

Flower et al. implemented a method that uses a multi-criteria optimization technique to improve curve aesthetics [14]. They defined metrics to handle curve roundness, smoothness, closeness and size uniformity, and combined them in a fitness function. Thus, this method could improve the layout of diagrams like Fig. 1A, but not Fig. 1B as their method handles diagrams with up to four curves. The effectiveness and correctness of these aesthetic metrics were not evaluated, and it is still unclear how the different metrics interact. The method uses a hill-climbing heuristic and thus, it is likely to encounter local minima and provide a local rather than a global best-optimized solution. The method is slow, as multi-criteria optimizations are more computationally expensive than single-criteria ones [21]. Assigning appropriate weights to the various criteria is difficult [21] and expecting users to assign these weights makes the method unusable.

In graph drawing, force-directed methods have been widely used and evaluated to produce layouts with desired aesthetic features with relatively good performance [5,19]. The physical analogy used by such methods is that of a system of physical structures (the vertices of the graph) that exert a force over others in the system, such that these structures move according to the force applied to them. The system is brought to a halt when the algorithm positions the structures appropriately so that the forces are in equilibrium. One of the simplest force-directed methods is the spring embedder [6]. In such methods, the forces result from electrically charged particles (the vertices) that repel one another, so that the vertices are not too close to each other, and springs (the edges between vertices) that attract connected particles, so that the length of the edges is approximately uniform.

A closed curve represented as a polygon is like a graph with a set of vertices and edges, so the repulsive and attractive forces used in a spring embedder for graph drawing would transform a closed curve into a smooth regular circle. Thus, if such forces are applied to all the curves in a diagram and other new forces are applied to ensure that the required curve intersections are maintained, the diagrams in Fig. 1 would be converted to those in Fig. 2, so the curves are smooth, more regular and evenly distributed. The diagram layouts in Fig. 2 were generated by our method euler*Force*, which is the first to use a force-directed approach to improve the curve aesthetics and layout of Euler diagrams.

In this paper, we describe euler*Force*, the force model and algorithm it uses to improve the diagram layouts, and our evaluation of the method. The implementation of euler*Force* is available at http://www.eulerdiagrams.org/eulerForce.

2. The force model and algorithm

The main challenge was to devise an appropriate force model that acts on the vertices, edges and curves in the diagram to improve the layout of Euler diagrams while still depicting the same set relations. This is the first force model for Euler diagrams, so we opted for a simple algorithm to equilibrate the forces. This facilitates understanding of the different forces and how they interact with one another to allow for further refinement of the force model.

2.1. Force model

Our physical system is similar to that of the simple spring embedder (Section 1), in that the vertices act like electrically charged particles and the edges like springs. The force model consists of *repulsive* and *attractive* forces between different structures in the layout, including (i) vertices, (ii) edges and (iii) entire polygons. Thus, the forces in our system differ from those used in simple graph drawing methods by systematically moving any of these structures rather than just the vertices.

Similar to the typical spring embedder in graph drawing, our repulsive forces follow the inverse square law and our attractive forces follow Hooke's law [5]. Thus,

Please cite this article as: L. Micallef, P. Rodgers, euler*Force*: Force-directed layout for Euler diagrams, Journal of Visual Languages and Computing (2014), http://dx.doi.org/10.1016/j.jvlc.2014.09.002

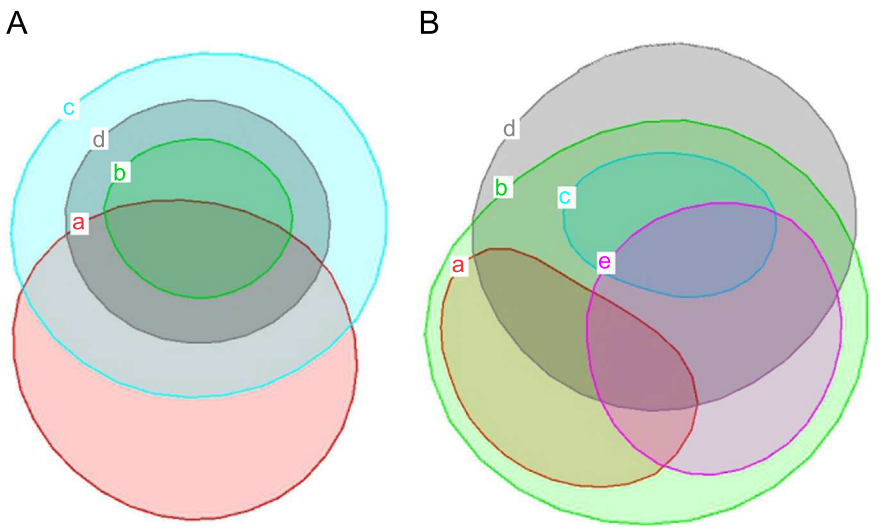

Fig. 2. The improved layouts generated by our force-directed method, euler*Force*, for the Euler diagrams in Fig. 1.

given d is the Euclidean distance between two structures s_1 and s_2 in the diagram, these forces are defined as follows: *repulsive forces* – inversely proportional to the squared distance between structures s_1 and s_2, so the repulsive force between s_1 and s_2, that is the repulsive force exerted on s_2 by s_1 and on s_1 by s_2, is $f_r = c_r/d^2$ where c_r is a constant that determines the strength of the force; *attractive forces* – directly proportional to the distance between structures s_1 and s_2 so the attractive force exerted between s_1 and s_2, that is the attractive force exerted on s_1 and s_2 by the spring between s_1 and s_2, is $f_a = c_a d$ where c_a is the stiffness of the spring that determines the strength of the force and the natural length of the spring is zero. The constants c_r and c_a vary depending on the objective and the required strength of the force. In specific cases, the definition of the repulsive or attractive force could defer from those above, yet the direction remains unchanged.

Our repulsive forces are the same as those used in Eades' spring embedder [6]. Our attractive forces are different from those of Eades, as Eades uses logarithmic rather than linear (Hooke's law) springs stating that the latter could be too strong. However, Di Battista et al. argue that, "it is difficult to justify the extra computational effort by the quality of the resulting drawings" [5]. Since our attractive forces assume linear, Hooke's law springs with natural length zero, they are the same as those used in Tutte's force-directed barycentre method [33]. We opted for such attractive forces as these forces are namely used to smooth the curves and to regain regions that are lost during the layout improvement process. Thus, while in the former the edges should be as short as possible to produce smooth curves, in the latter the force of the spring should be strong enough to attract structures and regain the lost regions.

We now discuss how such repulsive and attractive forces between vertices, edges and polygons are used in our force model to generate layouts that meet our objectives (in bold).

2.1.1. Obtaining regular, smooth, similarly shaped convex curves

We use typical forces for a simple spring embedder [5].

(**F1**) Repulsion for vertices not to be too close to one another: for every polygon p in the current layout and for every pair of distinct vertices v_1 and v_2 of p, a repulsive force is exerted between v_1 and v_2, so v_1 and v_2 move away from one another.

(**F2**) Attraction for approximately uniform edge lengths: for every polygon p in the current layout and for every pair of distinct vertices v_1 and v_2 of p that are connected by an edge, an attractive force is exerted between v_1 and v_2, so v_1 and v_2 move closer to one another.

2.1.2. Maintaining the same set of regions as that in the initial diagram layout

We devised a set of forces for each different type of curve relation to ensure that: (a) the current improved layout maintains the regions in the initial layout; (b) if the current layout has new regions or is missing any of the regions in the initial layout, forces correct the layout accordingly. We opted to use forces to correct layouts that depict the incorrect set of regions rather than to disallow such layouts altogether, to avoid local minima. So for every pair of distinct polygons in the initial layout, the following forces are applied.

(**F3**) If the two polygons in the initial layout *do not intersect*, and in the current layout they still *do not intersect*, if p_1 and p_2 are these two polygons in the current layout, for every vertex v_1 of p_1 and for every vertex v_2 of p_2, a repulsive force is exerted between v_1 and v_2, so these vertices move accordingly and the required disjointness of p_1 and p_2 is reinforced.

(**F4**) If the two polygons in the initial layout *do not intersect*, but in the current layout they *do intersect*, if p_1 and p_2 are these two polygons in the current layout, for every vertex v_1 of p_1 and vertex v_2 of p_2: if v_1 is inside or on an edge of p_2 and v_2 is inside or on an edge of p_1, an attractive force is exerted between v_1 and v_2; if v_2 is not

Please cite this article as: L. Micallef, P. Rodgers, euler*Force*: Force-directed layout for Euler diagrams, Journal of Visual Languages and Computing (2014), http://dx.doi.org/10.1016/j.jvlc.2014.09.002

inside or on an edge of p_1, a *repulsive force* is exerted on v_1 by v_2; if v_1 is not inside or on an edge of p_2, a *repulsive force* is exerted on v_2 by v_1. As these vertices move accordingly, the required disjointness of p_1 and p_2 is regained.

(**F5**) If the two polygons in the initial layout *intersect*, and in the current layout they still *intersect*, if p_1 and p_2 are these two polygons in the current layout, for every vertex v_1 of p_1 and for every vertex v_2 of p_2: if both v_1 and v_2 are on the boundary of the overlapping region, that is v_1 is inside p_2 and v_2 is inside p_1, a repulsive force is exerted between v_2 and v_1, so these vertices move accordingly and the required intersection of p_1 and p_2 is reinforced; if v_1 is not inside p_2 and v_2 is inside or on an edge of p_1, a repulsive force is exerted on v_1 by v_2, so these vertices move accordingly and p_1 and p_2 are not too close to one another; if v_2 is not inside p_1 and v_1 is inside or on an edge of p_2, a repulsive force is exerted on v_2 by v_1, so these vertices move and p_1 and p_2 are not too close to one another.

(**F6**) If the two polygons in the initial layout *intersect,* but in the current layout they *do not intersect*, if p_1 and p_2 are these two polygons in the current layout, for every vertex v_1 of p_1 and vertex v_2 of p_2, a special attractive force defined as $f=c/d^2$ where c is a constant determining the strength of the force and d is the Euclidean distance between v_1 and v_2 is exerted between v_1 and v_2, so these vertices move accordingly and the required intersection of p_1 and p_2 is regained.

(**F7**) If in the initial layout one of the polygons *contains* the other and in the current layout the polygons still *depict* the required *containment*: if p_1 and p_2 are these two polygons in the current layout and p_2 is contained in p_1, for every vertex v_1 of p_1 and for every vertex v_2 of p_2, a repulsive *force* is exerted between v_1 and v_2, so these vertices move accordingly and the required containment of p_2 in p_1 is reinforced.

(**F8**) If, in the initial layout, one of the polygons *contains* the other, but in the current layout, the polygons *do not depict* the required *containment*, if p_1 and p_2 are these two polygons in the current layout and according to the initial layout, p_2 should be contained in p_1, for every vertex v_1 of p_1 and vertex v_2 of p_2: if v_1 is inside or on an edge of p_2 and v_2 is not inside or on an edge of p_1, an attractive force is exerted between v_2 and v_1; if v_2 is inside or on an edge of p_1, a repulsive force is exerted on v_1 by v_2; if v_1 is not inside or on an edge of p_2, an attractive force is exerted on v_2 from v_1. As these vertices move accordingly, the required containment of p_2 in p_1 is regained.

F3–**F8** are applied between vertices of polygons to (a) maintain the regions of the initial layout and (b) correct layouts that are not depicting the same set of regions as that of the initial layout. However, to ensure (a) and reduce the need for (b), if a vertex v_1 of polygon p_1 is closer to a point x on an edge $e=(v_2, v_{2b})$ of a polygon p_2 than vertex v_2 of p_2, **F3**–**F8** are also applied between v_1 and e, such that e is moved based on the forces exerted on it about x.

2.1.3. Depicting each set relation by exactly one region

As the vertices are moved during the layout improvement process, a region depicting a set relation could be split up into more than one component, making the diagram difficult to comprehend as one of the most important well-formedness properties is not met [26]. Thus, for every pair of distinct polygons, p_1 and p_2, in the current layout and for every region r in any or both of p_1 and p_2: (**F9**) while r is made up of more than one component, if k is the smallest component of r, for every vertex v_1 of p_1 and vertex v_2 of p_2, if v_1 is inside or on an edge of k and v_2 is not inside or on an edge of k, an attractive force is exerted between v_1 and v_2, so these vertices move accordingly and a component of r is discarded.

2.1.4. Ensuring the curves are not close to one another

Layouts with curves close to one another are difficult to comprehend [2] and could break the important wellformedness property of non-concurrent curves [26]. The repulsive forces in our model keep the vertices apart and thus aid to achieve this objective.

2.1.5. Centring contained curves in their containing curve or region

Sometimes a curve is contained in another curve or a region. The repulsive forces in the model would ensure that this contained polygon remains inside the containing polygon or region. However, centring this contained polygon in its containing polygon or region, so that its boundary is equidistant from that of the containing structure, could improve the layout and its symmetry. Thus, (**F10**) when a polygon is contained in another polygon or region, if c_1 is the centroid of the contained polygon and c_2 is the centroid of the containing polygon or region, an attractive force is exerted on c_1 from c_2, so that the entire contained polygon is moved closer to c_2 and centred in its containing polygon or region.

2.1.6. Attaining adequately sized curves and regions

If the size of the regions is inadequate, the layout could be difficult to understand, particularly when regions are not easily visible and their area is disproportional to that of other regions [2]. Thus, a set of forces is required to adjust the size of the polygons and to move these polygons closer or further away from one another, so the required adequate region areas are obtained.

An adequate region area could be one that is similar to the area of other regions in the layout, so that the total area of the diagram is evenly distributed among its regions [2]. However, to facilitate the identification of the number of curves in which a region is located, an adequate region area could be one that is inversely proportional to the number of curves in which it resides, in that the greater the number of curves it is located in, the smaller the region area. So, if a k-curve region is a region located in k curves in a diagram with n curves, the area of the region is assigned a weight $w=n/k$. Thus, if for instance a diagram has three curves ($n=3$), a 1-curve region ($k=1$, $w=3$) will be twice as large as a 2-curve region ($k=2$, $w=3/2$) and three times as large as a 3-curve region ($k=3$, $w=1$).

The size of the polygons are adjusted accordingly by progressively increasing or decreasing the strength of the repulsive force **F1** that ensures that the vertices of

Please cite this article as: L. Micallef, P. Rodgers, euler*Force*: Force-directed layout for Euler diagrams, Journal of Visual Languages and Computing (2014), http://dx.doi.org/10.1016/j.jvlc.2014.09.002

polygons are not too close to one another. The greater the repulsive force, the further away neighbouring vertices of a polygon are from one another, thus enlarging the size of the polygon. The polygons are then moved using the following forces to adjust the region areas. (**F11**) To increase a region area: if r is the region whose area should be increased and c_1 is the centroid of r, for every polygon p that contains r, if c_2 is the centroid of p, an attractive force is exerted on c_2 from c_1, so that the entire polygon p is moved closer to c_1, thus increasing its size. (**F12**) To decrease a region area: if r is the region whose area should be decreased and c_1 is the centroid of r, for every polygon p that contains r, if c_2 is the centroid of p, a repulsive force is exerted on c_2 from c_1, so that the entire polygon p is moved further away from c_1, thus decreasing the size of r.

Similar to **F3**–**F8**, other forces have been included to correct any generated layouts whose regions differs from those in the initial layout, either because new regions are displayed or required regions are missing. We could have disallowed these incorrect layouts from the layout improvement process altogether, but we opted to accept them and correct them using the following forces, to reduce the chances of reaching a local minimum. Thus, if while increasing or decreasing region area, (**F13**) the current layout has a region that is not depicted in the initial layout: if r is the region that is in the current but not the initial layout and c_1 is the centroid of r, for every polygon p that contains r in the current but not in the initial layout, if c_2 is the centroid of p, a repulsive force is exerted on c_2 from c_1, so the entire polygon p is moved further away from c_1, thus reducing the size of r and its appearance in the layout until it is no longer visible. If alternatively (**F14**) the current layout does not have a region that is depicted in the initial layout: if r is the region that is in the initial but not the current layout, for every pair of distinct polygons p_1 and p_2 that should contain r, if c_1 is the centroid of p_1 and c_2 is the centroid of p_2, an attractive force is exerted between c_1 and c_2, so the polygons that should contain r get closer and the missing region is regained.

2.2. Algorithm

Our algorithm is similar to that used by Eades [6] to balance out the forces in the system. Given some set relations, an Euler diagram is generated by a current automatic drawing method and used as the initial layout. The algorithm then goes through the system in discrete time steps, so that at every step, the resultant force exerted on each of the vertices, edges and entire polygons in the layout is calculated and the vertices, edges and entire polygons are moved accordingly based on the magnitude and the direction of the resultant force. This new layout is then used as the starting layout for the next discrete time step. After a number of steps, the magnitude of the resultant force exerted on each of the vertices, edges and entire polygons is reduced to zero and the algorithm stops as the forces in the system equilibrate and no further changes in the layout are carried out.

Since most of the forces in the system are exerted on and relocate the vertices of the polygons in the layout, polygons with fewer vertices are subject to fewer changes than those with more vertices. Thus, before the algorithm goes through the system in discrete time steps, the number of vertices on each of the polygons in the layout is equalized. For instance, if a layout has two polygons p_1 and p_2, and p_1 has 10 vertices and and p_2 has 12 vertices, two vertices are added to p_1. This is done by first adding a vertex x between two vertices v_1 and v_2 of the polygon that are connected by an edge (v_1, v_2) and then, removing (v_1, v_2) and adding two new edges (v_1, x) and (x, v_2) between v_1 and x and x and v_2 respectively. Since the forces in the system can enlarge the size of the polygons, at the end of every discrete time step, the length of the edges of each polygon is checked and vertices are added to make the edges smaller and the polygons smoother.

Due to the various forces in the system, a limit is set on the magnitude of the resultant force exerted on a structure. This limit is inversely proportional to the number of discrete time steps the algorithm has already gone through in the system, so major changes are only carried out at the initial steps when a more extensive search for an appropriate layout is required. During the final steps, minor changes are carried out to refine the layout and ensure the algorithm converges to a solution.

The transition from the initial to the final layout is animated, thus facilitating understanding of how the forces in the system aid in improving the layout and how they interact with one another [5]. This method was thus helpful to understand and appropriately define the required forces to lay out Euler diagrams and to devise the first force model to improve the layout of such diagrams. Moreover, such a simple algorithm could possibly aid in preserving the mental map of the layout [7] from the initial to the final improved layout.

Eades's simple spring embedder [6] was aimed for non-dense graphs with few vertices. Poor layouts by this embedder are reported for graphs with hundreds of vertices [19], as in such cases a local minimum is more likely to be reached. As discussed earlier, we mitigate this issue by using specific forces that correct generated layouts that depict different regions than those in the initial layout. Even so, Euler diagram layouts typically have fewer than hundreds of vertices as often these diagrams have few curves. Later on, further sophisticated techniques can be adopted to handle more specific aesthetic criteria and to improve the efficiency and performance of our force-directed algorithm.

3. Evaluation

To evaluate our method euler*Force*, we used its software implementation to improve the layouts of Euler diagrams generated by a current drawing method [27], and we compared euler*Force*'s layouts with those generated by the only other implemented layout method for Euler diagrams [14]. All the experiments were run on an Intel Core 2 Duo CPU E7200 @2.53 GHz with 3.23GB RAM, 32-bit Microsoft Windows XP Professional SP1, SP2 and SP3 and Java Platform 1.6.0.14.

Please cite this article as: L. Micallef, P. Rodgers, euler*Force*: Force-directed layout for Euler diagrams, Journal of Visual Languages and Computing (2014), http://dx.doi.org/10.1016/j.jvlc.2014.09.002

3.1. Accuracy, time and aesthetics

We tested euler*Force* on diagrams automatically generated by Rodgers et al.'s method [27], to evaluate its effectiveness in generating improved layouts that satisfy our objectives. Rodgers et al.'s method was chosen, as it is the only method that draws a diagram for set relations for which a well-matched, well-formed Euler diagram can be drawn. Thus, if an improved layout generated by euler*Force* did not satisfy our objective of depicting each set relation by exactly one region or our objective of ensuring the curves are not close to one another, the diagram layout was not well-formed and a limitation in our method was evident, as a well-formed diagram for those set relations is known to exist (i.e., the initial diagram generated by Rodgers et al.'s method).

A library of Euler diagrams generated by Rodgers et al.'s method for all the set relations for which a well-formed Euler diagram with three, four and five curves can be drawn was assembled. This library included: nine Euler diagrams with three curves, 114 Euler diagrams with four curves, and 342 Euler diagrams with five curves.

Our method euler*Force* was then used to improve the layout of the diagrams in this library. Figs. 3–5 illustrate a few of (i) the diagrams in the library (also Fig. 1), and (ii) the corresponding layout generated by euler*Force* (also Fig. 2). The layouts (ii) in Fig. 3 and Fig. 4 depict precisely the same set of regions as those in the initial library layout (i) (also Fig. 1 and 2), but those in Fig. 5 do not and are thus examples of cases where euler*Force* fails to produce an appropriate layout. We now discuss these layouts and the results obtained.

3.1.1. Accuracy

The improved layouts for all the nine and 114 diagrams with respectively three and four curves had the same regions as those of the initial incomprehensible layouts, and thus satisfied our objective of maintaining the same set of regions as that in the initial diagram layout. For the 342 diagrams with five curves, only 209 of the improved layouts (61%) satisfied our objective of maintaining the same set of regions as that in the initial diagram layout. The latter result could be due to the increased number of vertices that are unmanageable with a simple spring embedder [6,19], particularly when the diagram has various regions.

Fig. 5A(ii) generated by euler*Force* for the diagram and initial layout Fig. 5A(i) has two new unwanted regions, *abcd* and *abcde*, that are not depicted in the initial layout. Thus, curves *a* and *b* should be disjoint. Curve *a* in the final layout generated by euler*Force* in Fig. 5A(ii) is not completed smooth as the forces that were specifically devised to correct layouts depicting regions that are different from the initial are striving to regain the disjointness between curves *a* and *b*. However, these forces seem to be weaker than other interacting forces in the system and thus, an inappropriate layout is generated. This also indicates the limitations of a simple spring embedder in managing various interacting forces in the system.

Fig. 5B(ii) generated by euler*Force* for the diagram and initial layout Fig. 5B(i) has two missing required regions, *ad* and *be*, that are depicted in the initial layout and one new unwanted regions, *abcde*, that is not depicted in the initial layout. All the curves in the final layout generated by euler*Force* in Fig. 5B(ii) are smooth and regular. However, the layout is not well-formed as there is a point on the three curves *a*, *b* and *e*. This example indicates the limitations of a simple spring embedder when a diagram has various regions. For various curve overlaps to be displayed, the curve will likely have to attain a less regular shape and thus, the strength of the forces, particularly those that aim at generating regular, smooth and similarly shaped convex curves, might have to be dynamically tuned using more sophisticated techniques. In fact, for region *abcde* not to be depicted in the diagram and for the diagram to be well-formed in that no point is on more than two curves, curves *b*, *c* and *e* should attain a more elongated shape rather than a circular shape, as in Fig. 5B(ii).

Thus, more sophisticated force-directed techniques such as those used for laying out large graphs (e.g., [16]) should be adopted for the algorithm to overcome local minima and to handle Euler diagrams with thousands of vertices and with various curves and regions.

3.1.2. Time

On average, the final improved layout for diagrams with three curves was generated by euler*Force* in 7 s, those with four curves in 26 s, and others with five curves in 77 s. Thus, though our current method uses a simple algorithm, which is not as efficient as other more sophisticated alternatives, improved layouts are still generated in relatively fast time. This is comparable to force-directed approaches for graphs, which typically produce layouts in around a minute [19]. Also, a response time of 10 s or less ensures the users' attention is maintained [22]. However, better-optimized algorithms should be considered in future force-directed approaches for Euler diagram layouts.

3.1.3. Aesthetics

As illustrated in the examples in Figs. 2–4, the curves of all the generated layouts depicting the correct set of the regions were smooth. Also, whenever possible, the curves were regular, similarly shaped and convex, all of which facilitate understanding [2]. So euler*Force* satisfies our objective of obtaining regular, smooth, similarly shaped convex curve. Similarly, the curves of all the generated layouts depicting the correct set of the regions were well-formed and satisfied the most important well-formedness properties of regions made up of at most one component and non-concurrent curves, as in Figs. 2–4. Even in diagrams with various curves contained in other curves or regions, as in Figs. 2, 3A–C and 4A and B, none of the curves are too close to one another. This could have been further facilitated by the forces that centre contained curves in their containing curve or region.

Layouts generated by a spring embedder are likely to be symmetric [8], as shown by most layouts in Figs. 2–4. However, besides the basic forces that are typical for a spring embedder in graph drawing, other forces that we devised for Euler diagrams are likely to aid in generating

Please cite this article as: L. Micallef, P. Rodgers, euler*Force*: Force-directed layout for Euler diagrams, Journal of Visual Languages and Computing (2014), http://dx.doi.org/10.1016/j.jvlc.2014.09.002

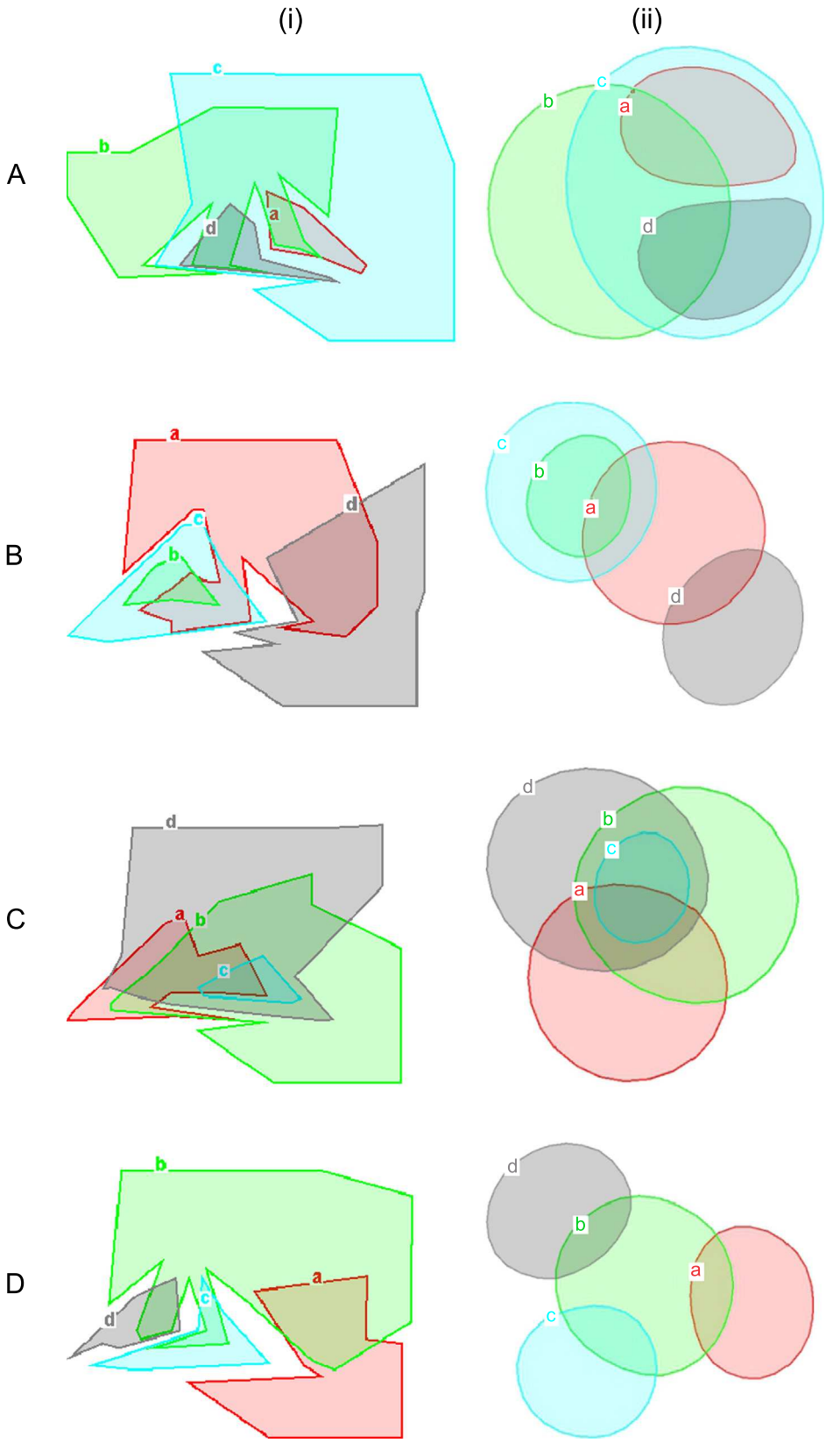

Fig. 3. Examples of (i) diagrams with four curves by Rodger et al.'s method [27] in our library and (ii) the correct layouts by euler*Force*.

Please cite this article as: L. Micallef, P. Rodgers, euler*Force*: Force-directed layout for Euler diagrams, Journal of Visual Languages and Computing (2014), http://dx.doi.org/10.1016/j.jvlc.2014.09.002

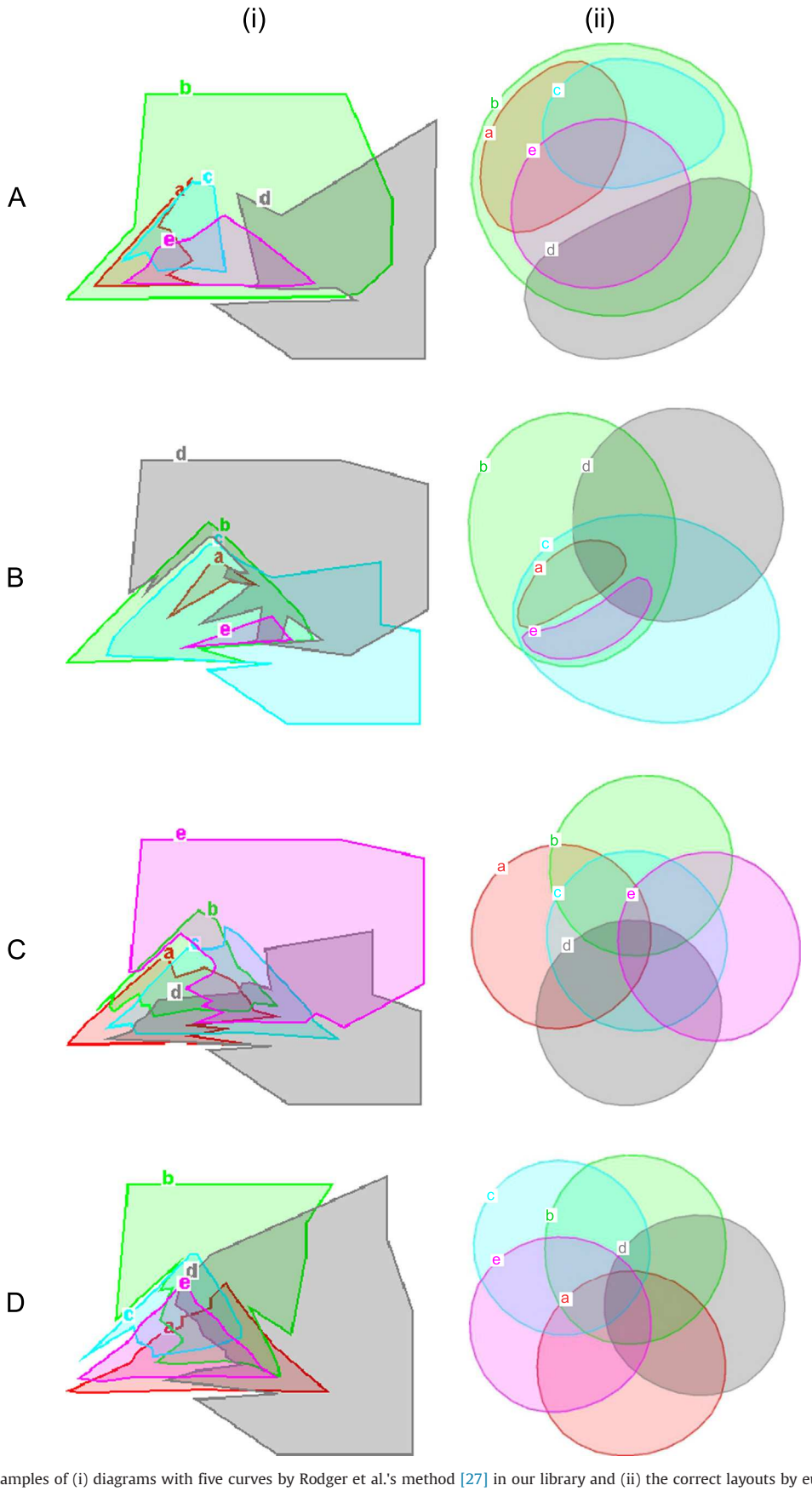

Fig. 4. Examples of (i) diagrams with five curves by Rodger et al.'s method [27] in our library and (ii) the correct layouts by euler*Force*.

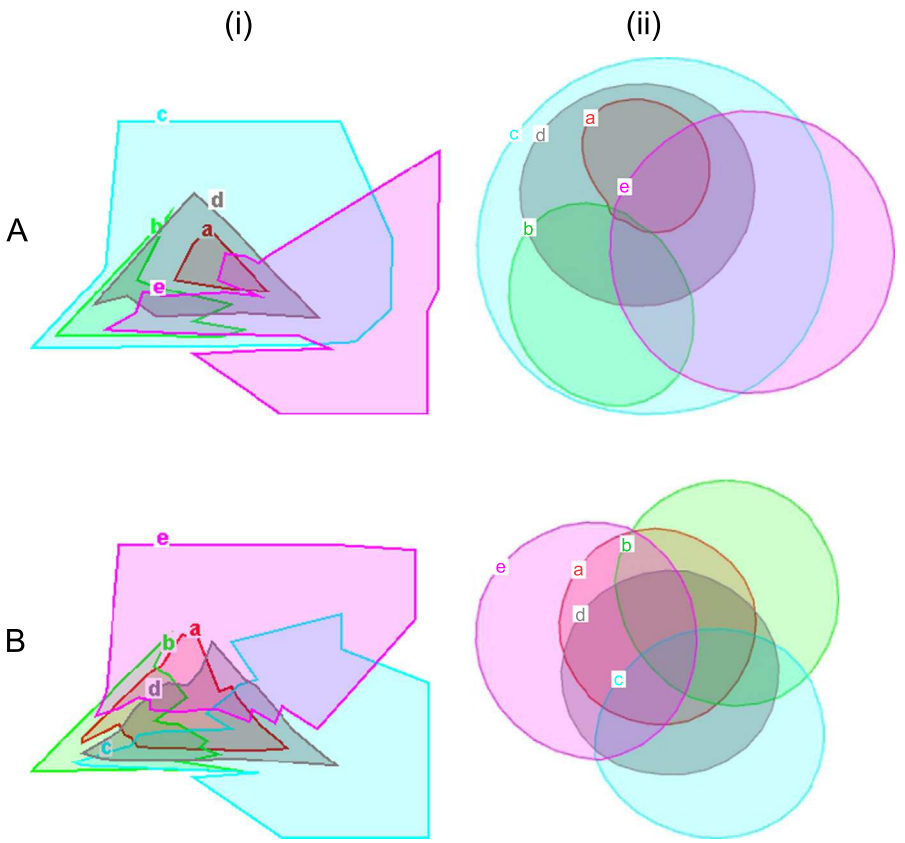

Fig. 5. Examples of (i) diagrams with five curves by Rodger et al.'s method [27] in our library and (ii) the incorrect layouts by euler*Force*.

symmetric layouts. In particular, the forces that centre contained curves in their containing curve or region aid in generating highly symmetric layouts, as Figs. 2, 3A–C and 4A and B.

Having adequately sized regions and curves also aid diagram comprehend [2]. The area of the diagram could be evenly distributed among its regions, but in our case we opted for an adequate region area that is inversely proportional to the number of curves in which it resides. The generated layouts including Figs. 2–4 indicate that this approach is effective as it ensures that: curves contained in other curves or regions are not too large for them to fit appropriately in the containing curve or region with possibly other regions, as in Figs. 2, 3A–C and 4A and B, and without breaking well-formedness; the number of curves in which a region is located is easier to identify.

For the layouts to be effectively evaluated, formalized aesthetic metrics and cognitive measures are required. Very few studies have investigated the aesthetics of such diagrams (Section 1), but no criteria have been formalized.

3.2. *eulerForce versus previous methods*

The only previous layout method that has been implemented is Flower et al.'s multi-criteria optimization method [14]. We compared the diagram layouts generated by Flower et al.'s method with those generated by euler*Force*.

As initial layouts, Flower et al. used diagrams generated by techniques [12,13] available at the time. Fig. 6A(i) and B(i) illustrate diagrams generated by these techniques. The technique we used to generate the initial layouts for euler*Force* [27] is more recent, but yet a variant of those used by Flower et al. for their method.

Given sets *a*, *b*, *c*, *d* and the set relations {ø, *a*, *c*, *ac*, *cd*, *acd*, *bcd*, *abcd*}, Flower et al.'s initial layout is Fig. 6A(i) and the generated improved layout is Fig. 6A(ii), while euler*Force*'s initial layout is Fig. 1A and the generated improved layout is Fig. 2A. Flower et al.'s initial and final layout look similar as the position and the orientation of the curves is barely changed, indicating that the method is limited to a minimal local search leading to a layout whose aesthetics could be improved further. For instance, the layout generated by euler*Force* has regular, similarly shaped, circular curves. The containing and contained curves *c*, *d* and *b* are centre aligned and the distance between curve *c* and *d* is the same as the distance between curve *d* and *b*. All of these features further aid in indicating subsets in the data depicted by the diagram, thus facilitating data analysis. So, in contrast to Flower et al.'s layout, euler*Force*'s layout is symmetric, compact, easy to understand and remember.

Similar observations are evident for the layouts depicting set relations {ø, *a*, *c*, *d*, *ac*, *ad*, *bc*, *abc*} where Flower et al.'s initial layout is Fig. 6B(i) and the generated improved layout is Fig. 6B(ii), while euler*Force*'s initial layout is Fig. 3B(i) and the generated improved layout is Fig. 3B

Please cite this article as: L. Micallef, P. Rodgers, euler*Force*: Force-directed layout for Euler diagrams, Journal of Visual Languages and Computing (2014), http://dx.doi.org/10.1016/j.jvlc.2014.09.002

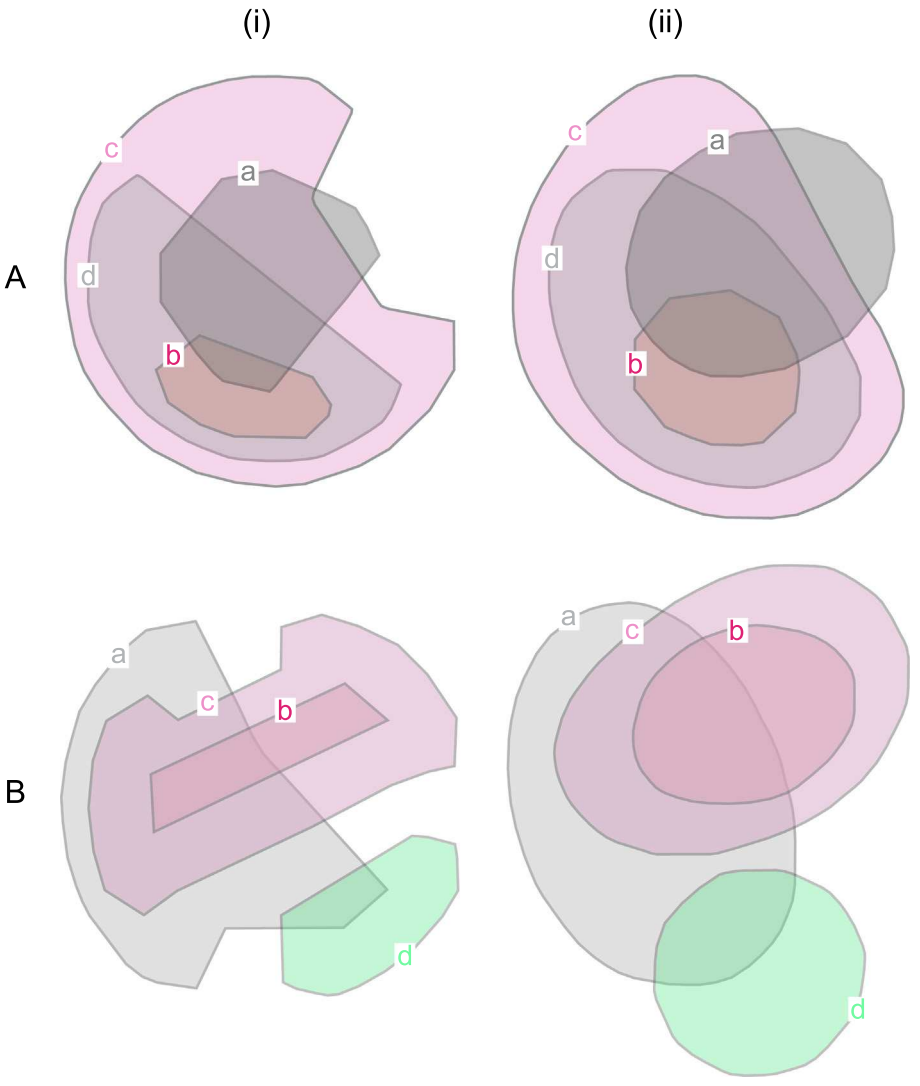

Fig. 6. The improved layouts (ii) generated by Flower et al.'s method [14] for the diagrams and initial layouts (i).

(ii). Flower et al.'s final layout, Fig. 6B(ii), was generated after 80 iterations and after the line segments in the diagram were converted to Bézier curves. The final layout of euler*Force*, Fig. 6B(ii), was generated in 17 s. So a layout improvement method using a force-directed approach as euler*Force* could be faster than ones using multi-criteria optimization like Flower et al.'s method. After all, multi-criteria optimization is known to be computationally expensive [21]. In contrast to euler*Force*, Flower et al.'s method is limited to diagrams with up to four curves and thus, no layouts with more than four curves could be included in our comparative analysis.

Though the initial layouts used by euler*Force* in our evaluation are less comprehensible than those used by Flower et al.'s method, the final improved layouts generated by euler*Force* are more aesthetically desirable and easier to use than those generated by Flower et al.'s method. The effectiveness of the layouts should be evaluated using formalized aesthetic metrics and cognitive measures. However, none are available for Euler diagrams and so, our comparative analysis and evaluation of the layouts was limited to a visual comparison of the layouts and based on the findings of the very few studies on Euler diagram aesthetics [2,3,26]. Even though Flower et al. defined a few aesthetic metrics to devise their layout method [14], these metrics were not evaluated.

4. Conclusion

In this paper, we have described our layout method, euler*Force*, the first method that uses a force-directed approach to improve the layout of Euler diagrams. Our evaluation indicates great potential for using force-directed techniques to improve Euler diagram layouts in quick time and to generate comprehensible diagrams given the required set relations.

It would be interesting to evaluate the layouts generated by euler*Force* for initial layouts that are not well-formed and for set relations for which a well-formed Euler diagram cannot be drawn. Until now, euler*Force* has been

Please cite this article as: L. Micallef, P. Rodgers, euler*Force*: Force-directed layout for Euler diagrams, Journal of Visual Languages and Computing (2014), http://dx.doi.org/10.1016/j.jvlc.2014.09.002

evaluated for initial layouts that are well-formed and for set relations for which a well-formed diagram can be drawn. This was intentional to evaluate the effectiveness of the forces that we specifically devised to ensure that there is only one region for each set relation and that the curves are not too close to one another. However, the effectiveness of these forces in handling not well-formed diagrams should be evaluated, so that if necessary, the force model is adapted to handle such diagrams.

We adopted a simple spring embedder algorithm to facilitate understanding and evaluation of our force model, which is the first for Euler diagrams. However, this algorithm is not as efficient as other force-directed algorithms and is unable to handle hundreds of vertices [19]. Such limitations are evident in our euler*Force* evaluation for Euler diagram layout with five curves, as discussed in Section 3. Until now, our focus was on the force model rather than the algorithm. In the future, sophisticated force-directed algorithms such as those used for laying out large graphs [17] can be adopted and investigated in the context of Euler diagrams.

For instance, a multilevel approach such as that used in graph drawing [34] can be adopted to overcome local minima and to efficiently handle layouts with thousands of vertices and thus, with various curves and regions like those in Fig. 5. As an example, Hu's method [16] uses this approach to lay out graphs with over 10,000 vertices in less than a minute.

The Barnes–Hut algorithm [1] can be used to efficiently and dynamically compute the appropriate forces at every step of the layout improvement process. This method has already been successfully used in graph drawing (e.g., [16]) and could aid in cases such as those in Fig. 5. Force-directed techniques in graph drawing have also demonstrated that adding magnetic fields to the system and its springs could aid in satisfying various aesthetic criteria [31] and should thus be considered for Euler diagram layouts.

Other future work includes gathering more empirical evidence to assess Euler diagram aesthetics and to formalize metrics that evaluate the effectiveness of Euler diagram layouts.

References

[1] J. Barnes, P. Hut, A hierarchical O ($N \log N$) force-calculation algorithm, Nature 324 (1986) 446–449. (Nature Publishing Group).
[2] F. Benoy, P. Rodgers, Evaluating the comprehension of Euler diagrams, in: Proceedings of the 11th International Conference on Information Visualization (IV), IEEE Computer Society, 2007, pp. 771–780.
[3] A. Blake, et al., The impact of shape on the perception of Euler diagrams, in: Proceedings of the 8th International Conference on the Diagrammatic Representation and Inference (Diagrams), Lecture Notes in Artificial Intelligence, vol. 8578, Springer, 2014, pp. 123-137.
[4] P. Chapman, et al., Visualizing sets: an empirical comparison of diagram types, in: Proceedings of the 8th International Conference on the Diagrammatic Representation and Inference (Diagrams), Lecture Notes in Artificial Intelligence, vol. 8578, Springer, 2014, pp. 146-160.
[5] G. Di Battista, et al., Force-directed methods, Graph Drawing: Algorithms for the Visualization of Graphs, Prentice-Hall, Upper Saddle River, NJ, USA, 1999, 303–325.
[6] P. Eades, A heuristic for graph drawing, Congr. Numer. 42 (1984) 149–160.
[7] P. Eades, et al., Preserving the Mental Map of a Diagram, International Institute for Advanced Study of Social Information Science, Fujitsu Limited, Numazu-shi, Shizuoka, Japan, 1991.
[8] P. Eades, X. Lin, Spring algorithms and symmetry, Theor. Comput. Sci. 240 (2000) 379–405. (Elsevier).
[9] H. Ehrig, et al., Handbook of Graph Grammars and Computing by Graph Transformation: Applications, Languages and Tools, 2, World Scientific Publishing Co., River Edge, NJ, USA, 1999.
[10] L. Euler, Lettres à une Princesse d'Allemagne sur divers sujets de physique et de philosophie, Lettres, 2, L'Académie Impériale des Sciences de Saint-Pétersbourg, St. Petersburg, Russia, 1768, 102–108.
[11] J. Flower, A. Fish, J. Howse, Euler diagram generation, J. Vis. Lang. Comput. 19 (2008) 675–694. (Elsevier).
[12] J. Flower, J. Howse, Generating Euler diagrams, in: Proceedings of the 2nd International Conference on the Diagrammatic Representation and Inference (Diagrams), Lecture Notes in Artificial Intelligence, vol. 2317, Springer, 2002, pp. 61-75.
[13] J. Flower, J. Howse, J. Taylor, Nesting in Euler diagrams, in: Proceedings of the 1st International Workshop on Graph Transformation and Visual Modeling Techniques (GT-VMT 2002), Electronic Notes in Theoretical Computer Science, Elsevier, vol. 72, 3, 2003, pp. 93–102.
[14] J. Flower, P. Rodgers, P. Mutton, Layout metrics for Euler diagrams, in: Proceedings of the 7th International Conference on Information Visualization (IV), IEEE Computer Society, 2003, pp. 272–280.
[15] J. Howse, et al., Visualizing ontologies: a case study, in: Proceedings of the 10th International Semantic Web Conference (ISWC), Springer, 2011, pp. 257–272.
[16] Y. Hu, Efficient and high-quality force-directed graph drawing, Math. J. 10 (2005) 37–71.
[17] Y. Hu, Algorithms for visualizing large networks, Combinatorial Scientific Computing, 2011.
[18] S. Kent, Constraint diagrams: visualizing invariants in object-oriented models, in: Proceedings of the 12th ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA), ACM, vol. 32, 10, 1997, pp. 327–341.
[19] S.G. Kobourov, Spring embedders and force directed graph drawing algorithms, Comput. Res. Repos. (CoRR) (2012). (abs/1201.3011).
[20] L.P. Lim, et al., Microarray analysis shows that some microRNAs downregulate large numbers of target mRNAs, Nature 433 (2005) 769–773. (Nature Publishing Group).
[21] R.T. Marler, J.S. Arora, Survey of multi-objective optimization methods for engineering, Struct. Multidiscip. Optim., 26, , 2004, 369–395 (Springer).
[22] R.B. Miller, Response time in man-computer conversational transactions, in: Proceedings of the Fall Joint Computer Conference, Part I, (AFIPS), ACM, December 9–11, 1968, pp. 267–277.
[23] S.E. Palmer, Common region: a new principle of perceptual grouping, Cogn. Psychol. 24 (1992) 436–447. (Elsevier).
[24] P. Rodgers, A survey of Euler diagrams, Special Issue of the Journal of Visual Languages and Computing on Visualization and Reasoning using Euler Diagrams, vol. 25, Elsevier, 2014.
[25] P. Rodgers, et al., Euler graph transformations for Euler diagram layout, in: Proceedings of the 27th IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC), 2010, pp. 111–118.
[26] P. Rodgers, L. Zhang, H. Purchase, Wellformedness properties in Euler diagrams: which should be used?, in: Proceedings of the IEEE Transactions on Visualization and Computer Graphics, vol. 18, 2012, pp. 1089–1100.
[27] P. Rodgers, et al., Embedding wellformed Euler diagrams, in: Proceedings of the 12th International Conference on Information Visualization (IV), IEEE Computer Society, 2008, pp. 585–593.
[28] S.-J. Shin, The Logical Status of Diagrams, Cambridge University Press, New York, NY, USA, 1994.
[29] G. Stapleton, A survey of reasoning systems based on Euler diagrams, Electron. Notes Theor. Comput. Sci. 134 (2005) 127–151. (Elsevier).
[30] G. Stapleton, et al., Automatically drawing Euler diagrams with circles, J. Vis. Lang. Comput. 23 (2012) 163–193. (Elsevier).
[31] K. Sugiyama, K. Misue, Graph drawing by the magnetic spring model, J. Vis. Lang. Comput. 6 (1995) 217–231. (Elsevier).
[32] N. Swoboda, Implementing Euler/Venn reasoning systems, Diagrammatic Representation and Reasoning, Springer, 2002, 371–386, (http://dx.doi.org/10.1007/978-1-4471-0109-3_21. Available at http://link.springer.com/chapter/10.1007/978-1-4471-0109-3_21).
[33] W.T. Tutte, How to draw a graph, in: Proceedings of the London Mathematical Society, Citeseer, vol. 13, 1963, pp. 743–768.
[34] C. Walshaw, A multilevel algorithm for force-directed graph drawing, in: Proceedings of the 8th International Symposium on Graph Drawing (GD 2000), Lecture Notes in Computer Science, Springer, vol. 1984, 2001, pp. 171–182.

Please cite this article as: L. Micallef, P. Rodgers, euler*Force*: Force-directed layout for Euler diagrams, Journal of Visual Languages and Computing (2014), http://dx.doi.org/10.1016/j.jvlc.2014.09.002