Personalised Resource Categorisation using Euler Diagrams

Paolo Bottoni[1], Gennaro Cordasco[2], Rosario De Chiara[2], Andrew Fish[3], and Vittorio Scarano[2]

[1] Department of Computer Science, "Sapienza" University of Rome - ITALY
[2] ISISLab - Dipartimento di Informatica, Università di Salerno - ITALY
[3] School of Computing, Engineering and Mathematics, University of Brighton - UK

Abstract. The categorisation of information is a very common practice. Often, the user may need to use multiple hierarchies or require multiple characterisations to be active at the same time, or they may wish to define cross-cutting groups for special purposes. The work developed in this paper is aimed at providing a flexible, seamless management of various ways of organising the required information. The concept of a set is adopted as the fundamental notion of information organisation, and, in particular, the familiar visual representation of sets and their relationships in terms of Euler Diagrams is used. We instrument the visualisation of sets to enable the uniform application of functions to items presented in selected diagram regions corresponding to set, or category, intersections. We present a system that realises this novel concept, together with rationale for the choices made in its development, as well as a simple scenario of use.

1 Introduction

Users continually make categorisations in their daily practices and exploit them to perform personal or organisational tasks. They often need multiple hierarchies or multiple characterisations to be active at the same time, or to define cross-cutting groups for special purposes. In such situations, current tools either require the use of complex procedures, or do not allow such usages at all.

Typically, the main categorisation provided by current file systems is an inflexible one, and hierarchical-based representations have well-known limitations in supporting user categorisation. On the other hand, the explicit construction of virtual folders is usually restricted to specific cases (e.g. emails), that are typically used to save the results of some search. As a result of this, users may have to deal with two different types of categorisation, which potentially may cause cognitive difficulties for non-sophisticated users in particular. Therefore, helping users to efficiently define and maintain the desired categorisations in the context of resource-based tasks which are routinely performed, or even for special purposes, would be of benefit in such situations.

The tagging of documents, or the use of metadata in general, is an emerging methodology for handling resources, but tag-based methodologies have the downside that documents with missing or mistyped tags may be omitted, and they can run into trouble if one also wishes to utilise methods based on file types. Although modern operating systems support both tag and file-type based save, search and retrieval methods, they do not explicitly exploit spatial features, leveraging users' spatial abilities and memory.

The work developed in this paper is aimed at providing a flexible and seamless management of different ways of organising information, whilst exploiting visual representations to facilitate user understanding of the structure of the information. Through these representations, users should be able to dynamically define the categories they are interested in, to manage multiple characterisations at the same time, and to access or define actions relevant to the different categories. In particular, we propose to adopt

sets, rather than folders, as the fundamental notion of information organisation, and to exploit their familiar visual representation in terms of Euler Diagrams (EDs).

The main contribution of the paper is the realisation as a flexible framework, and together with an implementation of a user interface for the construction and modification of Euler Diagrams as a non-hierarchical categorisation structure. We also provide a palette of operations, permitting a user to select from a collection of predefined operations and to apply the operations to the zones of an Euler Diagram.

2 Related Work

The categorization process can be viewed as the process of assigning tags to items. Hierarchies have been so widespread since the advent of personal computing that they are often automatically considered the natural way to organize data. There are many different techniques to visualize hierarchies (cf. [13,14,14]). Since hierarchical classification structures are often not sufficient for user classification needs, one can consider non-hierarchical classification structures, such as polyarchies [16] or EulerView [4].

Venn diagrams were used to represent non-hierarchical directories, replacing the traditional structure of file systems in [3], where diagrams could be drawn with curves representing categories (or tags) and files could be placed within more than one directory by utilising curve overlapping. In [6], an accessible Euler diagram interface was developed which enabled more general resource management, together with efficient interpretation algorithms to detect the underlying meaning of the regions of the diagram. A reification of an ED-based categorisation structure was integrated into Flickr in [5], utilising the non-hierarchical categorisation structure.

There has been relatively little user testing of the ED concept, with the idea generally being taken for granted as being beneficial. In [2], the comprehension of basic EDs (without items) with the same zone sets, but different visual properties e.g. curve jaggedness, was examined. In [8] the effects on user comprehension and preference of varying well-formedness conditions were investigated.

3 Background on Euler diagrams

An *Euler diagram* is a pair $d = \langle \mathcal{C}, \mathcal{Z} \rangle$ where $\mathcal{C} = \mathcal{C}(d)$ is a set of labelled *contours* (closed curves) in the plane and $\mathcal{Z} = \mathcal{Z}(d)$ is the collection of *zones*, z, which are specified as being inside a set of contours, $X_z \subseteq \mathcal{C}(d)$, and outside the rest of the contours of the diagram; X_z is called a *zone descriptor* (for details see [9].

Figure 1 shows an Euler diagram with four labelled curves representing the categories: $MUSIC$, $TECHNOLOGY$, $WORK$, $GADGET$. This set of curves decomposes the plane into a set of seven zones (the region outside all curves is itself a zone). As can be seen from the notion of a zone descriptor, we describe zones by specifying the labels of the complete set of curves that the zone is inside, with the outside set of curves being derivable from this information. For instance, in Figure 1 we can see that there are two music files (making use of the icons), one belonging to the zone $\{MUSIC\}$ and the other to the zone $\{MUSIC, TECHNOLOGY\}$.

We view the zones of an Euler diagram as a repository in which to place resources, such as `files` and `urls`, and so the basic notion of Euler diagrams is extended to capture the placement of items in the diagram, in a similar manner to that of unitary alpha Spider diagrams in the diagrammatic logic context [10]. However, in terms of the semantics assigned within this application area, we assign to each zone, z, the set of

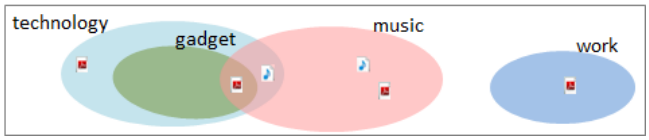

Fig. 1. An example of an ED used for resource categorisation: the two music files and the four PDF documents are categorised according their placement in the ED.

tags in its zone descriptor, X_z, and extend this assignation to any items placed in that zone. Thus an ED provides a means to build a non-hierarchical categorisation structure, and can be used to categorise resources, as in [6].

4 Categorising items with FunEuler

We describe the FunEuler system, together with the rationale for the design decisions adopted. The application is available at [1]. **Architecture.** FunEuler is conceived as a simple application running beside the windows that a user uses to manipulate their files and documents. FunEuler is implemented in Java on top of the EulerVC library described in [7]. This library provides a number of set manipulation functions and it enables the interactive creation and modification (e.g. curve addition, removal, or transformation) of an ED. It also allows the manipulation of items, which are simple elements that associate a resource to a two dimensional co-ordinate.

FunEuler is filetype agnostic, handling the placement (via drag and drop) of resources from any repository of files as well as URLs. An example of the flexibility of the paradigm is that it is permissible to categorise files (e.g. .doc, .pdf) and URLs (e.g. email addresses and bookmarks) using the same ED. We highlight that data in FunEuler is not duplicated, with FunEuler keeping track of both the resource location in the repository and the position in which the user placed the files on the ED.

Figure 2 shows the functional blocks of FunEuler: 1) repositories from which the items to be categorised can be retrieved (left); 2) a library of functions, see [7], for diagram manipulation and query operations for zones and items (centre); 3) processes for invoking operations (right), where arrows indicate the parameters to be passed.

Figure 3 shows a screenshot of the current user interface, which is comprised of three panes: the palette of operations, the Euler Diagram and the Results Euler Diagram. The left-hand pane contains a list of icons for the possible operations that the user may apply to the regions of the ED shown in the main pane. The main portion of the application window contains the ED that is used for categorisation and this is where the majority of the user-interaction is likely to occur. The rightmost pane shows a diagram that depicts the output from the application of the user-selected operations. At the top of the interface, a standard menu bar enables the user to select different interaction modes: modify a curve (Arrow icon), draw a new curve (Pencil icon), query the diagram (Eyedropper icon), reset the zoom level (Magnifying glass icon), and save the diagram (Camera icon).

Diagram construction and interaction. We allow users to construct a diagram by adding curves, in the form of ellipses, one at a time by a simple mechanism of left click and drag to specify one axis, whilst using the mouse scroll wheel to specify the other axis. The restriction of curves to be ellipses in the EDs does not permit the representation of arbitrary relations between curves, so a user might not be able to draw a diagram containing all and only regions from a specific selection. However, it is

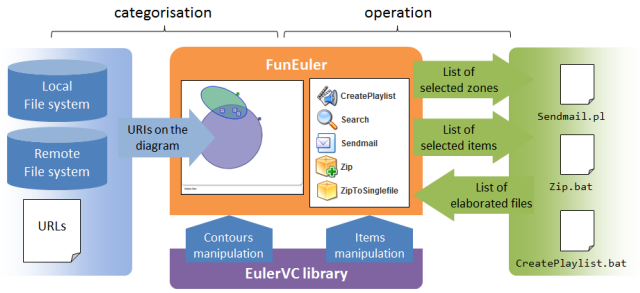

Fig. 2. FunEuler architecture.

important to realise that the creation of a diagram with the exact set of relations is not necessary in this context.

When a user creates a curve c, FunEuler assigns a random colour to c, as well as an automatic choice of set name, according to a pre-determined sequence. These set names, as well as the colours, can then be edited by the user. This enables users to rapidly construct a diagram, via ellipse additions, without having to explicitly name and set a colour for each new curve constructed since this could interrupt the user's reasoning flow. Moreover, every ellipse present in the diagram can be quickly modified by translation, rotation or by shape alteration (i.e. modifying the size of its axes).

In order to try to enhance the ease of comprehension of a diagram, and promote interactivity, FunEuler allows the user to query the diagram: when a user selects a position p in the diagram, exploiting the eyedropper, the entire zone z containing p is highlighted and the zone descriptor X_z is displayed in the status bar. The intent is to simplify item categorization and to help a reader understand an existing categorisation.

To enhance user navigation within larger diagrams, FunEuler provides horizontal and vertical scrollbars together with spatial zooming, via a select/zoom option and scroll in and out, in order to provide a larger working area when needed.

We utilise a drag and drop approach for resource classification. That is, items are placed within the relevant region of the diagram in order to categorise them. Item properties are visualized through tool tip text. The status bar, at the bottom of the interface, shows the current categorization for the selected item. Double click on an item allows to access the item using the default program associated with its extension/protocol. Each item can be interactively repositioned and consequently re-classified. Operations applied to single or multiple zones effect all items within those zones. However, in order to not restrict the selection of items (and hence the application of operations) to be all the items within the selected zones, we also allow the selection of multiple individual items that are displayed via a common box-selection.

Operational functionality. At the core of the FunEuler functionalities is the ability to select and apply one, or more, *operations* to regions of the diagram. This is performed by using the familiar drag and drop: the user first selects (by using the eyedropper) all of the zones to which an operation is to be applied, and then drags the operation icon onto the diagram, placing it on one of the selected zones. Placing the operation icon on a zone which is not already selected applies the operation to that zone, disregarding the selection.

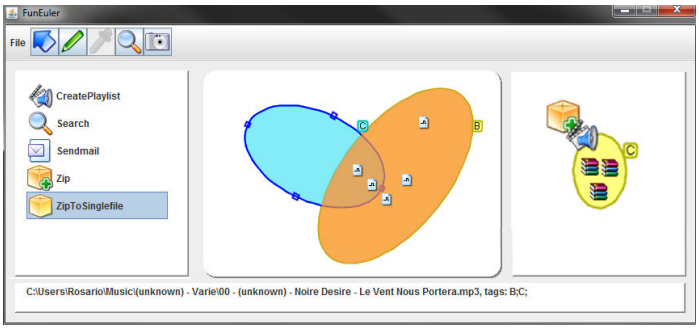

Fig. 3. A screenshot of the FunEuler interface.

For resources produced as the result of operations performed, we adopt an automatic naming convention of assigning the zone name followed by the type extension. For example, if we were to apply the zip function to the region in the intersection of the "Rock" and "Pop" curves, we would automatically name it `Pop&Rock.zip`.

Developing functionalities. FunEuler supports two different approaches to the development of functionalities, depending on the user confidence in software development: (a) write arbitrarily complex operations in any language capable of handling files (e.g. perl, shell script); (b) combine pre-existing operations in sequences that modify files in successive steps (e.g. encode some videos, create a zip containing the encoded videos, send it by email).

To allow the end user to design a new operation, FunEuler provides access to two lists for the items and the zones that the operation is to be applied to: one list containing the pathnames of the items and the zones they belong to, and another list reporting the zones and the items that each zone contains.

Each operation can produce zero, one or several files as result. As an example, the `zip file` operation produces a zipped file for each pathname in the input list, whilst the `zip to single file` operation produces one single `zip` file containing all of the files of the input list. On the other hand, the `send by email` operation does not produce any result files. Independent of the number of files produced the Results pane supports their representation in a special single-curve ED, where file icons are automatically placed within the curve representing the results set.

Since the result set is itself an ED, users can apply operations to it. This facilitates the sequential combination of operations, which produces a new Results diagram. The operation icons are added consecutively, on top of the previous ones. At a higher level, we can view the class of diagrams in FunEuler as being *closed under the application of operations*, since the application of an operation to a region of a diagram causes the generation of a new diagram in the Results pane of the application window. Given this closure property, FunEuler allows users to drag the Results diagram onto the main diagram area. Hence, users can categorise items which are the result of one of more operations alongside the original items that were already displayed in their diagram.

An example: `mp3` files management. FunEuler is a flexible framework on which to host services based upon an Euler Diagram As an example we developed a prototypical `mp3` files management system, considering the interest in it from the student population, in which users can tag `mp3` files and search for them by genre. By combining these

operations it is possible to categorise large archives of `mp3` files using few mouse clicks. Users can undergo *search-categorise-tag* cycles: search for untagged files, categorise them by dragging and dropping their icons onto the zone representing the correct genre, and tag files by applying the tagging operation. Untagged `mp3` files can be looked for by searching the zone outside all of the curves. Users can search multiple regions simultaneously, by using the eyedropper to select the set of zones to be queried.

5 Conclusion and Future works

We have brought together concepts of visual classification, spatial arrangements and functional application in a novel Euler diagram based interface called FunEuler.

There are many potential applications for the developed concept and interface. Firstly, in the workflow application area (e.g. see Alfresco), we could allow the drag and drop of a TeX document into a particular region, automate the running of macros such as PDFTeX and observe the results appear automatically elsewhere as an item in the PDF category. Users could then understand and track workflow involving multiple people on different systems. Secondly, the approach can be extended to full-fledged Euler Diagram Programming, by assigning behaviour to curves instead of zones.

References

1. Funeuler homepage. `http://www.isislab.it/projects/funeuler`.
2. F. Benoy and P. Rodgers. Evaluating the comprehension of Euler diagrams. In *Proc. IV 2007*, 771–780, 2007.
3. R. De Chiara, U. Erra, and V. Scarano. VennFS: a Venn Diagram File Manager. In *Proc. IV 2003*, 120–126. IEEE CS Press, 2003.
4. R. De Chiara and A. Fish. Eulerview: a non-hierarchical visualization component. In *: Proc. VLHCC 2007*, 145–152, IEEE CS Press.
5. R. De Chiara, A. Fish, and S. Ruocco. Eulr: a novel resource tagging facility integrated with Flickr. In *Proc. AVI 2008*, 326–330. ACM Press, 2008.
6. G. Cordasco, R. De Chiara, and A. Fish. Interactive visual classification with Euler diagrams. In *Proc. VL/HCC 2009*, 185–192. IEEE CS Press, 2009.
7. G. Cordasco, R. De Chiara, and A. Fish. Efficient on-line algorithms for Euler diagram region computation. *Computational Geometry*, 44(1):52 – 68, 2011.
8. A Fish, B. Khazaei, and C. Roast. User Comprehension of Euler Diagrams. To appear in *Journal of Visual Languages and Computing,* 2011.
9. J. Flower, A. Fish, and J. Howse. Euler diagram generation. *JVLC*, 19:675–694, 2008.
10. J. Howse, G. Stapleton, and J. Taylor. Spider diagrams. *LMS Journal of Computation and Mathematics*, 8:145–194, 2005.
11. A. Verroust-Blondet. J. Thièvre, M. Viaud. Using Euler diagrams in traditional library environments. In *Proc. ED 2004*, 189–202. ENTCS, 2005.
12. C. John, A. Fish, J. Howse, and J. Taylor. Exploring the notion of clutter in Euler diagrams. In *Proc. Diagrams 2006*, LNAI 4045, 267–282, 2006.
13. B. Johnson. TreeViz: treemap visualization of hierarchically structured information. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 369–370. ACM Press, 1992.
14. D. E. Knuth. *Fundamental Algorithms*, volume 1 of *The Art of Computer Programming*, pages 308–316. Addison-Wesley, third edition, 1997.
15. J. Lamping and R. Rao. Laying out and visualizing large trees using a hyperbolic space. In *ACM Symposium on User Interface Software and Technology*, pages 13–14, 1994.
16. G. Robertson, K. Cameron, M. Czerwinski, and D. Robbins. Polyarchy visualization: visualizing multiple intersecting hierarchies. In *Proc. HFCS 2002*, 423–430. ACM Press, 2002.