eul.icio.us: Euler diagrams for del.icio.us

Rosario De Chiara[†], Andrew Fish[‡1]

[†]ISISLab - Dip. di Informatica ed Applicazioni “R.M. Capocelli”,
Università degli Studi di Salerno - ITALY
dechiara@dia.unisa.it

[‡]School of Computing, Mathematical and Information Sciences,
University of Brighton - UK
Andrew.Fish@brighton.ac.uk

Abstract. *EulerView incorporates a non-hierarchical classification structure to enable enhanced resource management. We apply the EulerView concept to the use of urls, constructing eli.icio.us which is a tool that realizes the EulerView concept and integrates it with del.icio.us. Currently, in del.icio.us, users can tag urls and browse for similar urls and the results can be displayed via an alphabetically sorted tag cloud. As well as facilitating easy tag management, using an EulerView display has the advantage that the user can use this representation for both tagging urls and for searching or browsing tasks, thus reducing any cognitive difficulties of users due to interacting with multiple representations.*

1 Introduction and Background

The problem of categorisation for a resource management perspective is a difficult one. On one side there is the notion of hierarchical classification structures which are very rigid and not terribly adaptable to change or accessible enough to inspire usage for information storage, retrieval and browsing in today’s ever expanding collections of resources. On the other is free-form tagging which is commonly used but is a flat space, potentially making exploration of that space more difficult that might be desired. We investigate a solution which lies somewhere in between the two extremes, and apply this to a prominent social bookmarking service, but first we recap on background information.

Metadata refers to data about data, and typical usage involves data about resources such as documents, books, articles or photos. Metadata are specific to the particular purpose they are designed for (e.g. metadata that can be associated to books). They are used to facilitate the retrieval of those specific resources. The importance of Metadata is increasing, especially in web-related activities, because many web sites allow users to add metadata to resources such as photo, URLs and blog entries. In this particular context the operation of adding metadata has been dubbed *tagging*.

Tags are an effective way for authors to categorize their resources: they facilitate future retrieval of information using tags as keywords. Similarly, tags are useful for readers who are browsing for resources. Although an author may be the best person to assign tags to his/her resources, his/her resources may mean something different to other people, or they may just perceive the tags used in a different manner. For this reason web sites like Del.icio.us allow users to tag other people’s resources. This *collaborative tagging* has been dubbed “Folksonomy” [2, 16], joining the term *folk* and the term *taxonomy*; “specifically it refers to subject indexing systems created within Internet communities”.

Delicious. del.icio.us [1] is a social bookmarking service for storing, sharing, and discovering web bookmarks. Bookmarks are stored in a flat space (which is not hierarchical): every user can attach any number of tags to their bookmarks. The social nature of bookmarking is emphasized by the fact that del.icio.us allows users to view bookmarks added by similar-minded users: users can browse through bookmarks using the tags they or others have attached. The service, at time of writing, boasts more than three million users and 100 millions bookmarks. It is worth noting that all the features available on the website are also available through an effective API.

We wish to facilitate systematic tag-management, which raises the users awareness of the relationships between categories, and so we review and discuss categorisation and classification structures.

Categorisation and Classification. We recall the following distinction between categorization and classification [13]: “Categorization divides the world of experience into groups or categories whose members share some perceptible similarity within a given context”, where category composition depends on the context and on the user of the organization; “classification involves the orderly and systematic assignment of each entity to one and only one class within a system of mutually exclusive and non-overlapping classes”, where classification usually refers to the assignment of a resource (e.g. a document, URL or photo) to a single class, among classes, often hierarchically organized making clear the complete relationships amongst classes (see [4] for a typical biological example). The process of categorisation is generally perceived as being less precise than classification: the placement of an item within a classification structure indicates precise global information about that item, whereas placement of an item within a categorisation structure may represent partial information about the item, which is to be interpreted locally, or within a given context.

We broaden the notion of classification structures: since hierarchical classification structures are often not sufficient for user-classification needs (and this is even felt to be the case in a real office setting [20]), one can consider non-hierarchical classification structures, such as polyarchies [17] or EulerView [5]. One could perform a reificia-

[1]Funded by UK EPSRC grant EP/E011160: Visualisation with Euler Diagrams.

tion of non-hierarchical classifications to squeeze them into the classification definition given above by mapping two overlapping classes to three classes: two non-overlapping classes together with another class for the objects in the intersection. Non-hierarchical classifications also fit with the notion of categorisation: items in the intersection of categories share some property. So, non-hierarchical classifications could be thought as living somewhere in the middle of this Categorisation-Classification spectrum, allowing the overlap of categories but with a formal underlying model. Strict classification system interfaces are especially useful for storage and retrieval of information, but are often thought to be too restrictive for users, for many reasons, such as: they may be time consuming to use, they may get very large and the visualisations unmanageable very quickly, and changes in the classification structure over time may require the update of all existing resources that are already classified – which may be a difficult chore. On the other hand, categorisation systems such as free-form tagging are much easier and quicker to use, but they have their own drawbacks such as: the necessity to use different visualisation techniques, such as tag clouds [3] in order to browse through existing resources.

We advocate a “best of both worlds approach”, by keeping an underlying (non-hierarchical) classification system, but presenting to the user interactive views of that data in the form of a simple categorisation structure which may be user-created and easily manipulated. This will bring the benefits of a classification structure, especially in terms of information retrieval, but with the flexibility associated to a categorisation structure.

We briefly review some diagrammatic systems which could be used as classification systems, and we introduce the non-hierarchical classification system called EulerView.

Diagrammatic classification systems. Euler diagrams are a well known, intuitive method of representing certain relationships between sets. Although the original Euler circles [6] may have not had a rigorous formal underpinning, advances since then [10, 19] have led to the development of formal diagrammatic systems based on Euler diagrams, such as Spider Diagrams [12]. Simple closed curves in the plane are used to represent sets and the spatial relationships between curves (intersection, containment and disjointness) are used to represent the corresponding set-theoretic relationships. One of the reasons that they are said to be an effective representation is that these spatial relationships are well-matched to the corresponding domain relationships they represent [9]. However, one difficulty with Euler diagram based languages is the common visual language problem of *overspecificity* [18]: if you wish to represent information about a set then you have to specify its relationship with all of the other sets which are already represented. If a large number of curves (or a large number of items densely packed together within a region) are present in a diagram then this can also lead to the diagram being cluttered [15]. One approach to try to address this visualisation problem is to introduce projections, as in [7, 14], which enable the specification of relationships within a specific context.

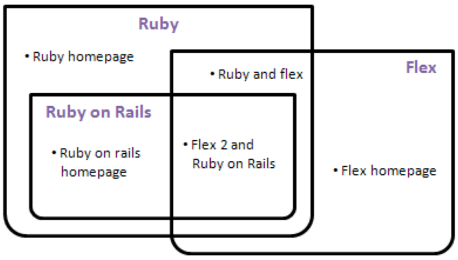


Figure 1: EulerView and Euler diagrams.

In a resource-tagging context one can consider the sets as the categories (or tags) and then one can represent sub categories, intersections between categories, and disjointness of categories. Thus they enable a non-hierarchical classification. An advantage of such non-hierarchical classifications is that, as well as enabling easier initial categorization, if a user has two separate classifications and the desire to merge them then it becomes feasible to do this easily, since categories from one classification do not need to be disjoint from categories in the other.

EulerView. In [5], EulerView was introduced to enable the common user to easily capture the naturally non hierarchical organization of file-system management. In order to avoid user difficulties due to migration from the standard Treeview, EulerView was developed as an extension of the Treeview, but with the hierarchy of a Treeviewextended into a non hierarchical structure by using the power of the Euler diagram model; Figure 1 shows Euler diagram corresponding to the EulerView in Figure 4. Developing EulerView as such a (1 + 1)-dimensional visualization of Euler diagrams also assisted the user in overcoming potential navigation difficulties (i.e. scrolling and/or zooming in 2 or 3 dimensional interfaces).

2 eul.icio.us application

We wish to enhance the user experience of tagging resources by providing them with the facility to create and easily manipulate their own categorisation structure to update tags (and in particular tags for bookmarks on del.icio.us). We have developed an information storage and display system called EulerView, which has an underlying model based on Euler diagrams. It also has a novel, flexible interface to enable users to construct multiple integrated categorisation paths, which will facilitate tagging tasks as well as be useful in enhancing ease of future navigation (or classification) tasks. This methodology will extend to multiple different users using their own (multiple) categorisation (or classification) systems, but still enabling them to share their information effectively.

Enabling the use of multiple categorisations for a single user

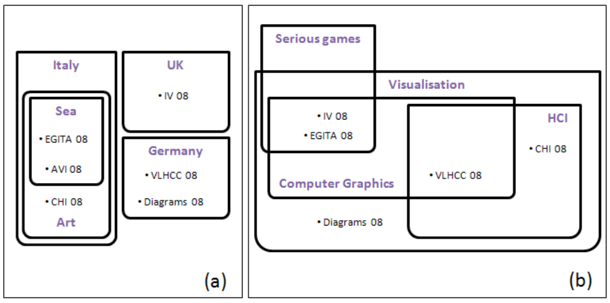


Figure 2: A personomy of bookmarks using Euler diagrams: this is a multiple (subjective) classification as perceived by one single user.

provides flexibility, as well as making it easy to extend the concept to think about multiple users. In [8] they decompose tags into extrinsic tags (i.e. tags about what or who the bookmark is about, identification of what it is, who owns it and refining categories [2]) and intrinsic categories (qualities or characteristics, self reference and task organisation). One might want to have two different categorisation structures, and to use both to classify the same bookmark: one using extrinsic tags and one using intrinsic from [8], or one might simply have different classifications in mind for the different facets of the bookmark. For example, Figure 2 shows two Euler diagrams related to topics of interest for conferences and the venues, being used to tag the same bookmarks; they could equally can be used independently to tag other bookmarks. For comparison we also show two corresponding EulerView's in Figure 3. These are examples of a single users subjective non-hierarchical multiple view classification. Placement of bookmarks within these categorisation structures, enabling partial tagging (i.e. not requiring the explicit creation of intersection categories) would provide a quick method for utilising both categorisation structures together. This would also facilitate re-use of user-created categorisation structures. Therefore, a small investment in initial creation of categorisation structures will lead to continued benefits in the future. This motivates our desire to develop a user categorisation display structure which has a high degree of flexibility.

We have integrated EulerView with del.icio.us to give rise to eul.icio.us; it facilitates user creation and management of tags attached to bookmarks in del.icio.us. The eul.icio.us tool facilitates the development of a user categorisation structure, making it easier to assign sets of tags to collections of resources. The constraints we impose are very relaxed, allowing great flexibility, although the imposition of stronger constraints could be enforced, if desired. eul.icio.us also allows the export, to del.icio.us, of the user-defined sets of tags for a collection of bookmarks. Furthermore, we enable the import of multiple bookmarks from del.icio.us into the eul.icio.us display, thereby automatically creating a categorisation structure which the user can use or manipulate.

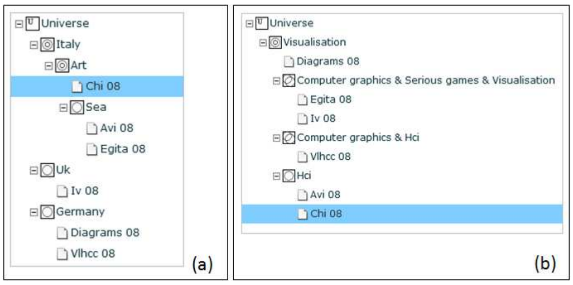


Figure 3: A personomy of bookmarks using EulerView: this is a multiple (subjective) classification as perceived by one single user.

A major advance of eul.icio.us over the existing del.icio.us bookmark management is that eul.icio.us enables the systematic management of tags. This brings many advantages such as enhancing the users' understanding of the categorisation structure (allowing user construction and manipulation of such a structure which displays relationships between resources); this can be a difficult comprehension task using "flat" tags. It has also been developed within a constraint-based framework, where the level of constraint could be varied, according the the application domain or user preferences, for instance. Tag management is an especially important issue because many social interactions within the del.icio.us community rely on tags.

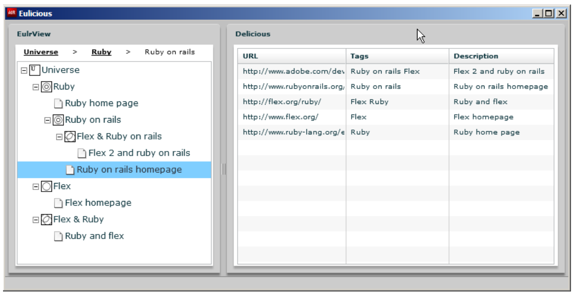
2.1 User interface

Figure 4 shows the current user interface. The left pane shows EulerView used to categorize bookmarks, whilst the right pane displays bookmarks taken from a specified del.icio.us account. A search facility is also available in del.icio.us which enables the user to search for bookmarks with a specified set of tags.

The core function of eul.icio.us is to manage bookmarks and the tags associated with them. To tag a bookmark in eul.icio.us a user has to drag it from the del.icio.us pane and drop it on the EulerView pane within a category. The semantics of the tagging operation is formally described in section 3. Clicking on a resource in eul.icio.us opens the link to that resource. Once the user is happy with the tags associated to the bookmarks dragged onto the EulerView pane, he/she can decide to export those tags to del.icio.us. Exporting to del.icio.us means that tags which are implicitly represented by the containing categories in the EulerView pane become explicit tags which are visible in del.icio.us.

The two representations, categories in EulerView and tags in del.icio.us are closely related each other: what happens in one side could be automatically reflected on the other side. We chose not to enforce automatic updates in the application but rather to let user to decide when to export the representation in del.icio.us to EulerView and vice versa; the automatic update is an alternative option that could be used if user-preference dictates.

[2] interestingly refining categories will be dealt with within the EulerView system we that develop.



Universe > Ruby > Ruby on rails

- Universe
  - Ruby
    - Ruby home page
    - Ruby on rails
      - Flex & Ruby on rails
        - Flex 2 and ruby on rails
      - Ruby on rails homepage
  - Flex
    - Flex homepage
  - Flex & Ruby
    - Ruby and flex

URL	Tags	Description
http://www.adobe.com/dev	Ruby on rails Flex	Flex 2 and ruby on rails
http://www.rubyonrails.org/	Ruby on rails	Ruby on rails homepage
http://flex.org/ruby/	Flex Ruby	Ruby and flex
http://www.flex.org/	Flex	Flex homepage
http://www.ruby-lang.org/e	Ruby	Ruby home page

Figure 4: eul.icio.us application user interface: on the left pane the EulerView control, on the right pane the del.icio.us bookmarks.

3 Formal specification

We provide a formal definition of EulerView, separating the underlying data structure, called an EulerView storage system, from the display, called and EulerView display. This enables the creation of a formal specification of our eul.icio.us tool, which is a specific instantiation of an EulerView display integrated with del.icio.us, initially to enhance the url tagging experience, with the intention of aiding in searching and browsing tasks. The underlying data structure for storage is similar to an abstract Euler (or Spider) diagram and this is how the name originated. Alternatively, one could just consider this to be an organisation of flat tag-space, by grouping together resource links whose sets of tags exactly match.

Definition 1 *Let $\mathcal{C} = \{C_1, \ldots, C_n\}$ be a set of categories and let $I = \{I_1, \ldots, I_m\}$ be a set of intersections of those categories. Let R be a set of resources (e.g. a collection of bookmarks, photos) and let $f : I \rightarrow \mathcal{P}R$ denote a function which allocates sets of resources to unique intersections of categories; that is, there are functions $f_j : I_j \rightarrow \mathcal{P}R$, for all $j \in \{1, \ldots, m\}$, each of whose images are pairwise disjoint, and $f : I \rightarrow \mathcal{P}R$ is the extension of these functions. We call $(\mathcal{C}, I, f)$ an* ***EulerView storage system.***

The display is designed, with user interaction in mind, to be more flexible than than a strict classification system using an Euler diagram-like structure and to have an appearance similar to Treeview to aid in accessibility due to its familiarity. The main idea is that each category vertex will correspond to a set of tags with which a resource can be tagged easily, but that a user may construct multiple paths to the view of the same virtual folder (which will effectively store the resources).

Definition 2 *Let $\mathcal{C} = \{C_1, \ldots, C_n\}$ be a set of categories. If $T = (V_C, E_C)$ is a labelled rooted tree, with vertex set V_C and edge set E_C, and there is a vertex labelling function $L : V_C \rightarrow (\mathcal{P}(V_C) - \emptyset) \cup \{U\}$ which assigns a reserved symbol U* [3] *to the root vertex, $*$, and a nonempty set of elements of $\mathcal{C}$ to every other vertex, then we say that T is a* ***EulerView categorisation tree*** *and the vertices of T are called* ***category vertices****.*

The following concepts are useful for discussions later in the paper.

Definition 3 *Let $T = (V_C, E_C)$ be a categorisation tree. Let the* ***rootpath*** *of a vertex $v \in V_C$ be the unique path from the root vertex $*$ to v, denoted $P(*, v)$. The* ***contextual label set*** *of v is the union of all of the labels of the internal vertices in $P(*, v)$, denoted by $L_c(v)$. The* ***total label set*** *of v is $L_c(v) \cup L(v)$, and is denoted by $L_t(v)$. Let $B(v)$ denote the* ***branch*** *of T at vertex v (that is, the tree obtained by considering v as a root vertex). Two vertices v_1 and v_2 of T are said to* ***lie in different branches*** *if $v_1 \notin B(v_2)$ and $v_1 \notin B(v_2)$; their* ***nearest common ancestor*** *is the closest vertex (to v_1 or v_2) of $P(*, v_1) \cap P(*, v_2)$.*

Of course we need to be able to add and manage resources in the categorisation structure.

Definition 4 *Let $D = (V_C \cup V_R, E_C \cup E_R)$ be a labelled rooted tree, where (V_C, E_C) is a EulerView categorisation tree, V_R is an extra set of* ***resource vertices*** *which are leaves of D, and E_R is an extra set of edges joining each resource vertex to exactly one category vertex. Let R be a collection of links to resources. If each resource vertex is labelled by an element of R then we say that D is an* ***EulerView display****.*

In an EulerView display we interpret a category vertex v as a folder in which to place a resource which is to be tagged by $L_t(v)$, but not tagged by any of v's children vertices. Therefore, when exporting, we will be adding the set of tags corresponding to the total label set of v to the resource.

Definition 5 *If two category vertices have the same total label set then we say that those vertices are* ***equivalent category vertices****.*

Equivalent category vertices represent the same underlying set of tags and any resources placed under different but equivalent category vertices will be associated with that same set of tags to be exported. In diagrammatic terminology, equivalent category vertices correspond to the same underlying zone in the Euler diagram (which is a model of the EulerView storage system). One can take the analogy further and if we have two category labels, C, which are the same and appear in different branches of the tree then we are looking at C within the context of other categories, and these could be viewed as projections [7]; extensions of the categorisation model to Euler diagrams with projections is future work.

Formal specifications of the operational behaviour of the eul.icio.us tool can be provided, but for ease of reading we describe the functionality of the core features using natural language descriptions, with occasional references using more formal language where necessary.

The eul.icio.us categorization structure can be created and manipulated by a user, and we have very few constraints enforced in order to facilitate user flexibility. The universe, or root vertex, is always present. A new category vertex can be added as child vertex of any existing category vertex. Any

[3] here U stands for the Universe of Discourse

non-root category vertex v can be moved to become a subcategory of another vertex w; note that $B(v)$, the branch from v, moves with it. Two category nodes, with distinct label sets A and B, can be intersected to create a new category vertex with label $A\&B$. One can relabel any single-label category vertex. When an intersection is created, the label of the intersection vertex is linked to the single-label category vertices involved in its label and consistently reflects the renaming of those labels. Any non-root category vertex v can be deleted and the branch from v is also deleted (including all of the resource links). Resources (which are links to bookmarks stored in del.icio.us here) can be added, deleted and moved at will (so one can place their own bookmarks and change their labels).

The export operation takes the set of resources in the EulerView display, and exports the set of tags associated to each resource (by virtue of its position) and adds all of these tags to the resources tag (i.e bookmarks on del.icio.us). The import operation allows a user to select a single photo, or a set of bookmarks, and have them imported into their EulerView display, either just as resources or together with the automatic creation of the relevant categories. This facilitates easy categorisation of bookmarks with similar tags.

Adding new categories or resources, together with renaming, is achieved by right click followed by selection via a drop down menu. Category movement is achieved via drag and drop, and category intersection is achieved using drag and drop with a modifier of holding down the control key. Exporting the eul.icio.us tags is achieved via a mouse click on a prominent button. Selection of multiple bookmarks from del.icio.us is achieved by holding down the control key. Releasing the control key allows drag and drop to a category vertex v to place the set of resources in v, whilst keeping the control key depressed enables the drag and drop to a category vertex v to place the automatic eul.icio.us structure created as a subcategory of v (which could be the root node).

User study. In [5] a user study was reported on, whose intention was to give an indication of the usefulness of EulerView for the purpose of quick single-person bookmarking. Results and user feedback suggested that EulerView is perceived as an easy to use extension of the Treeview, and that users seem to be able to perform accurate quick bookmarking, whilst storing a fairly accurate internal representation of the underlying Euler diagram model.

4 Other tasks

One of the main avenues for work with eul.icio.us to be developed further in the future is that of enhancing its functionalities to facilitate its use for user-browsing and searching.

Browsing tasks. The facility to use the same user constructed categorisation structure (i.e. the eul.icio.us display) to navigate and browse through resources as well as for tagging purposes would be clearly beneficial. Therefore, in our design we kept in mind the desire to incorporate future functionality for tasks such as: showing all resources with at least, or exactly, a given set of tags; showing similar collections of tags to a given set of tags (e.g. with a high proportion of tags in common); displaying a common, or existing, categorisation (or classification) structure to assist in tagging bookmarks of a certain type. User-browsing requirements, considered together with the ability to allow multiple user constructed paths to the same resources, could inform decisions about what resources should be displayed beneath a categorisation vertex in the eul.icio.us display structure. eul.icio.us development will inevitably move towards social browsing using tags. The "related tags" concept suggests ways of comparing EulerView categorisation structure (or indeed, actual diagrams in the future) designed by different users to facilitate the discovery of new bookmarks.

Resource Display options. For the purpose of user tagging requirements we chose to display only resources added or imported by the user; these have been placed according to user interaction (or the automatic placement due the use of the import wizard). That is, the current eul.icio.us display mechanism is "what you place and where you place it is what you get".

In order to assist with searching and browsing tasks, an alternative display option is to show the resources stored in the underling model according to the local context in the eul.icio.us display. That is, a categorisation vertex displays the resources which contain at least the set of tags corresponding to that vertex, but not any of the tags of its children vertices (else the categories will get overfilled quickly). Formally the resources displayed as children of a category vertex v would be all of those resources whose tags contains $L_t(v)$ but excludes $L(v')$ for all category vertices v' which are children of v.

For the variations of display requests we could add functionality to: display all of the resources with at least a given set of tags via an option to display all of the resource data of a given vertex or any below it (i.e. flatten from a given vertex of the tree); display all of the data with exactly a specified set of tags, easily retrievable from the EulerView storage;

For the other search problems, we have predefined (template) EulerView displays, or generate them automatically from existing tag structures (currently we perform a version of this as our import wizard); use sorting algorithms on the EulerView display (currently the facility to sort the categorization structure alphabetically is enabled) to place categories near each other if they have a high proportion of tags in common for example, either in the case of keeping the existing EulerView display or generating a new one according to user preference.

Context Constraint. An option that could be introduced, in order to ensure that the user defined categorisation structure does not become confusing (especially in the context of sharing these structures with other users) is to impose a contextual constraint on the displays. For instance, we could ensure that no vertex v of T has a label containing a category that appears in its own context (i.e. $L_c(v) \cap L(v) = \emptyset$). Currently eul.icio.us is kept as flexible as possible and so we do not impose such a constraint, but investigating the effects of such a decision of such an imposition is future work.

5 Conclusions

We have developed, and provided a formalisation of, a novel information storage and display structure called EulerView, which can be used for the systematic management of tags. It has the flexibility of a relaxed categorisation structure (the EulerView display), but also provides an underlying non-hierarchical storage model which is based on Euler diagrams. Our focus here was on user resource-tagging, but we envisage that EulerView will be of great benefit in the future as a single interface to be used for both user tagging and searching tasks, as discussed in section 4; there are many avenues that have opened up to investigate further.

We developed the eul.icio.us tool, which is an instantiation of the EulerView model, where the resources to be tagged are bookmarks which are stored on del.icio.us. Functionality has been included to enable users to build multiple non-hierarchical categorisations within which to place links to bookmarks from del.icio.us. The resource tagging information can be exported to update the tag set of the bookmarks on del.icio.us. Furthermore, imports of multiple resources from del.icio.us, with or without an associated eul.icio.us categorisation structure is enabled.

In [8], they presented data confirming that users primarily bookmark for their own benefit, even if it does benefit others as well. Therefore it seems that the use of a personalized EulerView which presents their own views of the classification without worrying too much about the views that other users have is a reasonable stance. Furthermore, in [11] they view a folksonomy as a collection of personomies (single users classifications) and the EulerView nicely captures the notion of personomy, including the subtag/supertag relation which was ignored in the paper (for simplicity purposes).

Amongst other things, EulerView is an option to bridge the gap between taxonomy and faceted systems to get the best of both worlds, as desired in [21]. On a similar theme they suggest in [11] that folksonomies are a bottom up complement to more formalised semantic web technologies as they rely on emerging semantics which result from the converging use of vocabulary. Again, EulerView may prove useful in the mid-ground between the two extremes. Investigation of EulerView in conjunction with the trend detection techniques (called FolkRank) described in [11] may prove to be another interesting avenue to pursue.

References

[1] Del.icio.us Social Bookmarking. http://del.icio.us.

[2] Wikipedia: Folksonomy. http://en.wikipedia.org/wiki/Folksonomy.

[3] Wikipedia: Tag Clouds. http://en.wikipedia.org/wiki/Tag_clouds.

[4] Wikipedia: Taxonomy. http://en.wikipedia.org/wiki/Linnaean_taxonomy.

[5] R. D. Chiara and A. Fish. Eulerview: a non-hierarchical visualization component. In *Proceedings of the IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC '07)*, pages 145–152, Washington, DC, USA, 2007. IEEE Computer Society.

[6] L. Euler. Lettres a une princesse dallemagne sur divers sujets de physique et de philosophie. *Letters*, 2:102–108, 1775. Berne, Socit Typographique.

[7] J. Gil, J. Howse, and E. Tulchinsky. Positive semantics of projections. *Journal of Visual Languages and Computing*, 13(2):197–227, April 2001.

[8] S. Golder and B. A. Huberman. The structure of collaborative tagging systems. *Journal of Information Science*, 32(2):198–208, 2006.

[9] C. Gurr. Effective diagrammatic communication: Syntactic, semantic and pragmatic issues. *Journal of Visual Languages and Computing*, 10(4):317–342, 1999.

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

[11] A. Hotho, R. Jschke, C. Schmitz, and G. Stumme. Trend detection in folksonomies. In *Proc. First Inter. Conference on Semantics And Digital Media Technology (SAMT), LNCS* 4306, pages 56–70, 2006.

[12] J. Howse, G. Stapleton, and J. Taylor. Spider diagrams. *LMS Journal of Computation and Mathematics*, 8:145–194, 2005.

[13] E. K. Jacob. Classification and categorization: a difference that makes a difference. *Library Trends*, 52(3):515–540, 2004.

[14] C. John. Reasoning with projected contours. In *Proceedings of 3rd International Conference on the Theory and Application of Diagrams, LNAI* Vol. 2980 , pages 147–150, Cambridge, UK, 2004.

[15] C. John, A. Fish, J. Howse, and J. Taylor. Exploring the notion of clutter in Euler diagrams. In *Proceedings of the 4th International Conference on the Theory and Application of Diagrams*, pages 267–282, Stanford, USA, 2006. Springer.

[16] A. Mathes. Folksonomies - cooperative classification and communication through shared metadata. Technical report, 2004. http://www.adammathes.com/academic/computer-mediated-communication/folksonomies.html.

[17] G. Robertson, K. Cameron, M. Czerwinski, and D. Robbins. Polyarchy visualization: visualizing multiple intersecting hierarchies. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 423–430. ACM Press, 2002.

[18] A. Shimojima. Inferential and expressive capacities of graphical representations: Survey and some generalizations. In *Diagrammatic Representation and Inference: proceedings of Diagrams 2004, LNCS* Vol. 2980, pages 18–21, 2004.

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

[20] Thomas W. Malone. How do people organize their desks?: Implications for the design of office information systems. *ACM Trans. Inf. Syst.*, 1(1):99–112, 1983.

[21] Z. Xu, Y. Fu, J. Mao, and D. Su. Towards the semantic web: Collaborative tag suggestions. In *Proceedings of the Collaborative Web Tagging Workshop at the WWW 2006*, Edinburgh, Scotland, May 2006.