TagRiver Visualization

Basak Alper

Media Arts and Technology, UC Santa Barbara

basak.alper@mat.ucsb.edu

Angus Forbes

Media Arts and Technology, UC Santa Barbara

angus.forbes@mat.ucsb.edu

ABSTRACT

TagRiver is a novel tag-cloud visualization which incorporates both multivariate and temporal data. TagRiver displays tags associated with a number of data elements simultaneously by visualizing the temporal changes within the tag information and the volume of the tags associated with each element. Tags for respective elements are scaled based on their popularity, and then are displayed in vertically adjacent polygonal regions. The area of each of these regions is proportional to the volume of the tag information of the corresponding data element. To be able to distribute various sized tags within a polygonal region we utilize a novel 2D packing algorithm which satisfies certain aesthetic criteria and runs in real-time. TagRiver updates the polygonal area and tag information for each time step and provides a summary visualization for the past time steps, where users only see the volume of the tags associated with each data element. Users are equipped with simple interactive widgets which they can use to traverse and insepct tags associated with any time step on demand.

1. INTRODUCTION

Tag cloud visualizations became highly popular on the internet over the last few years, especially after being utilized by the photo sharing website Flickr [8]. The efficacy of various approaches to tag cloud visualization has been explored in a number of studies. For example, some approaches utilize graph-drawing algorithms to cluster associated tags [3][4]. Bielenberg [5] proposed circular tag layouts in social networking navigation systems, where significant tags defining the focus of a group appear closer to the center.

Visualizing the evolution of tag information over time also has drawn some attention. Dubinko et al. [6] have proposed algorithms that sample the most salient tags within arbitrary time frames, which are then enhanced with animations and interactivity so that users can interact with the tags as they evolve over time. Russel [7] visualized evolution of individual tags in the form of line graphs.

TagRiver utilizes a layout based not on semantic relations as in [3][4][5], but rather approaches the problem with an aesthetic constraint that requires more significant (and hence larger) tags be placed in central positions given an arbitrary polygonal region.

The packing algorithm responsible from the layout runs at every time-step for each set of tags within a polygonal region. The transition from one time-step to the next is done with smooth animations. Users are able to see only the volume

information for the past time-steps in the form of stacked bar charts; however, they are able to see any time-step in detail with its tags simply by using a mouse rollover, which functions similarly to a Fish-Eye lens.

The visualization of temporal information in the form of continuously connected stacked regions is a visualization technique proposed by Havre et al. [1] in ThemeRiver. In that sense, TagRiver is a specialization of the ThemeRiver visualization for temporal tag clouds or other folksonomic datasets.

2. OVERVIEW
2.1 Visualization of Current Time Step

The TagRiver visualization divides a rectangular display area into vertically adjacent polygons. The boundaries of these polygons are determined by the relative popularity of each data element. Our current implementation uses four data elements, although the only restriction on the number of data elements is the available screen space. Relative popularity for each data element is calculated at each time step by dividing the number of tags associated with this element to the overall number of tags for that time step. This yields a normalized measure for the popularity.
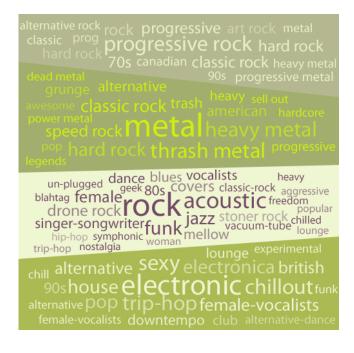
The popularity measure is then used to determine the lengths of the right and left sides of the rectangular region, respectively, for the previous and the current time step. Connecting these points with straight lines yields the vertically adjacent quadrilaterals fitting in a rectangular area. Each of these quadrilaterals is color coded and contains the tags associated with the data element they represent.

Tags within each area have the same color, and are scaled based on their number of occurrences in the current time step. Tags appearing for the first time are semi-transparent, and become more and more opaque as they appear successively in the following time steps. The tags are then positioned using a bin-packing algorithm which has been customized to take layout properties in to consideration. Details of the algorithm are discussed in section 2.3.

Since there may be a large number of tags associated with a particular category, it may not be possible to display each tag at a legible size. For this reason we set a lower bound on the size of the tag, which excludes tags which appear with only a low frequency, and thus present only a sampling of the set of tags at any given time.

2.2 Visualizing Historical Data

As time passes, the visual elements representing past time steps are modified. The tags shrink and disappear and the polygonal regions are reduced down to stacked bar charts representing normalized volume of tag information associated with each data element. Similar to the local view, each slice of time contains a set of color-coded quadrilaterals representing categories filling up a rectangle. The left side again represents the beginning of a time period, and the right side represents the end of that time period.
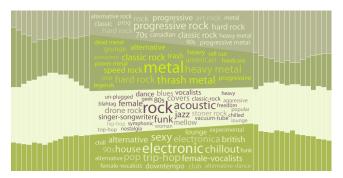

Figure 2 Summary of the historical data is provided around the current time-step of the visualization.

The horizontal axis of the visualization is the time axis, and a mouse rollover action is used to pick the current time step. The currently-selected time step has the largest area and its tags are displayed. Optionally, the width of each time slice is based on its recentness, or distance to the current time step--

The most recent past time period (previous to the current time) is largest, and each successive time period in the past become slimmer. This reflects the idea that a user would be most interested in recent local fluctuations, but overall be interested in seeing the more general trend of the activity of the categories.

The transition between consecutive time steps is smoothly animated. The polygonal regions change form with smooth animations as do the tags. If a tag is going to appear in the next time step, then the visualization animates the color, position and scale smoothly of the tag so that it is easier to follow the continuity of the tags over time. Tags that do not appear in the next time step shrink down and disappear. Those that were not present in the previous step slowly grow at their defined position and start with a semi-transparent color.

2.3 Bin-Packing Algorithm

Bin-packing algorithms are algorithms which attempt to minimize the amount of space taken up by a set of objects. Since determining the optimal minimum space is an NP-complete combinatorial problem, a variety of heuristics have been developed to find good approximate solutions within a minimum amount of time [9]. However, in general these heuristics ignore aesthetic concerns pertinent to information visualization. We present a customized bin-packing algorithm which handles these issues while displaying the descriptive tags. By providing some "slack" space to each of the tags whereby a tag may be slightly larger or smaller than indicated by its activity we can specify layout considerations, such as text justification, amount of space to fill, arrangement patterns of different size tiles, and quantization of tiles.

The bin-packing algorithm works as follows: 1) the tags are sized and then sorted by their popularity for a particular data region; 2) the first, largest tag is positioned within the centermost space which is large enough to accommodate it; 3) a set of available regions is then identified as possible positions in which to place subsequent tags; 4) a subsequent tag is placed into one of these regions; 5) if it fits then we return to step 3 using the next most popular tag, after appropriately culling the list of possible positions that this placement has invalidated; 6) otherwise we return to step 4 to examine another possible position; 7) if it does not fit anywhere then we find the largest of the possible positions that has a similar aspect ratio as the tag we trying to place and scale it to fit; 8) we then scale all subsequent tags accordingly; 9) we exit the layout algorithm once all tags are placed, or once there are no possible positions in which to place a tag, or once a tag cannot be scaled to fit within any of the possible tags without being shrunk beneath a threshold size.

In general, the results of this layout method are quite pleasing. The most relevant tag is positioned in the center with the largest size, and less relevant tags are positioned further away from the center at smaller sizes. Note that it may not be possible to position the tag in the exact center of the polygon since the top and bottom segments may be slanted. Also, because of the possible scaling of the tags during placement, the sizes of the tags will not necessarily be perfectly

representative of their frequency within the dataset. We have tried to mitigate this by making all tags with the same or smaller frequency to be similarly scaled so that the general informational aspect of the visualization still functions. Thresholds such as the maximum amount of scaling and the maximum distance of possible positions from the center are controlled by a small set of slack variables. In certain datasets too much or too little slack can adversely affect both the aesthetic display and informational content.

2.4 Software

TagRiver is written using a custom graphics framework based on the Java OpenGL bindings [10]. The framework includes a novel scene hierarchy in which animations are objects that are attached to geometric objects. Animations are traversed and updated in a separate loop, similarly to the way geometries are traversed, updated and finally rendered to the scene.

The Java platform enables TagRiver to be embedded into web pages via traditional applets or Java WebStart. Alternatively, TagRiver can be run as a desktop application.

3. DATASET

While TagRiver is meant to be a flexible visualization technique applicable to a wide variety of data, we use as an example dataset user profiles gathered from Last.fm [11]. Last.fm is a combination of a social networking site and a personalized online radio station. Last.fm builds a detailed profile of each user's musical taste by recording details of all the songs the user listens to. This information is then publicly accessible through a web services API. Specifically, it provides continually-updated information about users' online presence and also about the songs they have listened to within certain time frames. Based on this information we map a category to a particular user, and the total time spent online during a particular time frame to the activity of the category. We also can extract all the tags associated with the genres the user is currently interested in within the specified time frame. Example TagRiver visualizations utilizing this dataset can be found at **https://svn.mat.ucsb.edu/projects/TagRiver**.

4. CONCLUSION

Temporal tags are becoming common datasets in many websites utilizing folksonomic categorization. TagRiver presents a novel approach that combines temporal data visualization and tag cloud visualization techniques. It uses a novel bin-packing algorithm which organizes tags within arbitrary polygonal regions based on aesthetic criteria. The TagRiver visualization smoothly animates over time, keeping the layout as static as possible, so that both short-term changes and long-term trends draw attention.

Currently users have limited interaction with the visualization application. Extending the interactive capabilities of the application, such as by enabling users to determine the ordering and the number of categories, are possible next steps. Conducting user studies and applying the visualization to other datasets and other online sources are future steps which will help us to identify and evaluate the strengths and weaknesses of the visualization.

5. REFERENCES

[1] Havre S., Hetzler E., Whitney P., and Nowell L. 2002 ThemeRiver: visualizing thematic changes in large document collections. IEEE Transactions on Visualization and Computer Graphics, vol. 8:1, pp. 9-20

[2] Lee, K., Kim, H., Jang, C., Kim, H. 2008 Folksoviz: a subsumption-based folksonomy visualization using wikipedia texts. Proceeding of the 17th international conference on World Wide Web table of contents, Beijing, China pp. 1093-1094

[3] Kaser, O. and Lemire, D. 2007 TagCloud Drawing: Algorithms for Cloud Visualization. In proceedings of Tagging and Metadata for Social Information Organization, WWW 2007

[4] Hassan-Montero, Y. and Herrero-Solana, V. Oct 2006 Improving tag-clouds as visual information retrieval interfaces. In International Conference on Multidisciplinary Information Sciences and Technologies (InSciT2006), M´erida, Spain

[5] Bielenberg, K. 2005 Groups in social software: Utilizing tagging to integrate individual contexts for social navigation. Master's thesis, Universit¨at Bremen

[6] Dubinko, M., Kumar, R., Magnani, J., Novak, J., Raghavan, P. and Tomkins, A. 2006 Visualizing tags over time. In 15th International World Wide Web Conference, pp. 193–202 ACM Press, New York

[7] Russell, T. 2006 cloudalicious: folksonomy over time. In JCDL'06, pp. 364–366

[8] Wikipedia, 2004 Tag cloud — Wikipedia, the free encyclopedia. [Online; accessed 4-January-2009]

[9] Sleator, D. A. 1980. 2.5 Times Optimal Algorithm for Packing in Two Dimensions. Information Processing Letters, Vol. 10., No. 1

[10] JOGL API at https://jogl.dev.java.net/

[11] Last.fm at http://www.last.fm