Visualizing Laws over Time

CPSC 547: Project Update

Knomos Knowledge Management Inc.

Monday, November 23, 2015

K. Dextras-Romagnino, Y. Pushak
kdextras@cs.ubc.ca, ypushak@cs.ubc.ca

Contents

1 Previous Work

1.1 Overview Tools

1.1.1 Comparing Two Years

Tarantula, by Jones et al. [1], uses a dense display to source code documents where each line of code has two quantitative attributes. Each line of code is given one pixel in height and a bivariate colour map. A similar approach could be used to encode the changes between two documents, where instead of using colour to encode quantitative attributes, colours were selected to encode the categorical data of section added, deleted, or changed. One of the problems we see with adapting Tarantula to our task, is that it will not clearly illustrate the hierarchical structure of the law such as showing the relationship between sections, subsections, paragraphs, and sub-paragraphs.

Schulz et al. [2] discuss several methods for representing the changes between hierarchical documents, including a sunburst, polar Treemap, or InterRing, or a Treemap layout. However, out data set is not well suited to a circular layout as in the first three as the top layer of the tree typically would have many children requiring a large center node which would take up unnecessary space. Tu et al. [3] explored the use of a Treemap to encode changes within a hierarchical document. However, we have opted not to use a Treemap (or a radial layout) as we believe they would be less intuitive to the average user for understanding the structure of a law. Instead this structure could be easily displayed in a much more familiar context using a zoom-able icicle layout also mentioned by Schulz et al.[2], which can then be scented with colours to encode the additional information about added, deleted, and modified sections.

1.1.2 Comparing Multiple Years

While all of the methods discussed in Section 1.1.1 are potential options for providing an overview of changes between two years, they would struggle to show differences between several years, particularly when the same line has changed multiple times in different years. While we believe that it is very important to provide a salient summary of the differences between two years, we also see it necessary to provide an overview of all of the changes to a document that have occurred in the entire timeline.

In contrast to the previous methods discussed, ThemeRiver [4] takes as input a collection of documents and identifies themes over time. These themes are represented stacked area marks whose height encodes the weight of a topic which varies over the x-axis to encode the changes over time. This vis technique was extended by the History Flow Vis idiom presented by Viegas et al. [5],to visualize edits made to Wikipedia articles over time. The horizontal axis was again used to represent time with the vertical axis encoding the height of the document. Each line was then colour coded to reflect the author. This view allowed users to see how the document structure grew, shrank, and shifted over time.

TreeVersity2 [6] is a tool that visualizes changes in large hierarchical data sets over time similar to what we are trying to do. It contains an overview line graph showing the aggregated changes in a given time interval as well as another view showing stacked bar graphs (centered in the middle) representing the aggregated changes between any two given years. There is also the option of breaking the view into smaller components to gather more detailed information. Though we won't be using the same idioms due to differences in structure of our data set, we plan to use the same idea of combining general overview of the changes in law over time as well the comparison between two dates and finally, having the ability to drill down to see further details.

1.2 File Difference Tools

File difference tools are used to compare changes between two text files. In some cases [7, 8], two versions of a file are given as input to the tool, which then displays each side-by-side using colour-coded mark-ups such as green highlights in one file, to indicate an added section of text, and red highlights in the other, to indicate a deleted section of text. Other tools, such as the "Track Changes" feature in Microsoft Word 2013 [9],

only show the document in one view, and embed the changes made to it, i.e., inserted and deleted text, within the single view. In this case, changes in text are encoded with a font colour change, with deleted text represented using a strike-out mark and added text underlined. One clear advantage of this representation is that reduces the amount of screen space needed for the vis idiom. On the other hand, when extensive edits are made to a document, this technique may become more cluttered, and the user may have more difficulty in piecing together the individual edits to build a mental model of each document version. When two side-by-side views are used to display each version of the document, it is easy to see what is contained in each version of the document, however unnecessary additional screen space is needed to encode regions of the documents that are identical in each document.

2 Project Progress and Updates

We are currently, and not surprisingly, running behind our initial target project deadlines. Mostly, this is because we have found that some tools we expected to be able to simply find and use, did not appear to exist and some of the tasks turned out to be more difficult than originally planned.

Primarily, parsing the XML files into a format that we wanted proved to be harder than expected. This was partially due to the fact that it took awhile to get accustomed to using HTML and JavaScript, languages that we had little previous knowledge about. Also, it took time to determine in which way we wanted the data formatted as well as what information we wanted to extract in order to make the process of building our visualizations easier. There are two main XML files that need to be parsed for any given law, the law XML and the changes XML. Currently we have found a way to properly parse the law XML and are still working on parsing the changes XML.

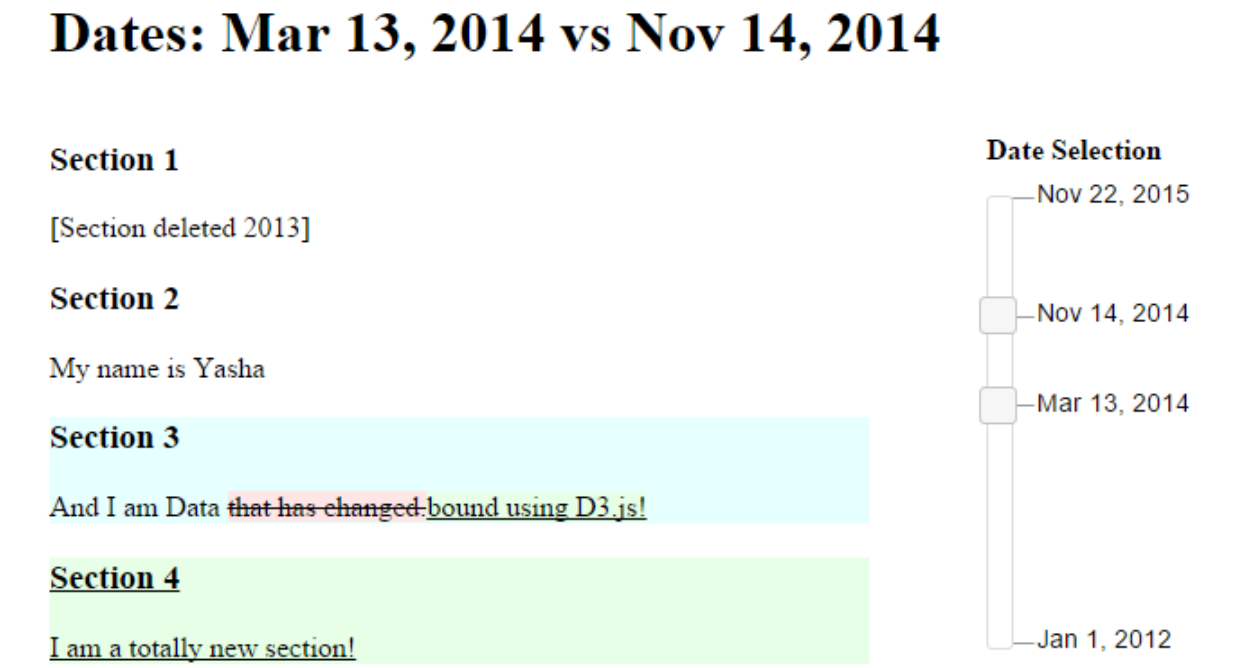

Figure 1: A screen shot of our current prototype for comparing two documents.

Finding a difference tool that adequately compared two strings proved a more challenging task than initially anticipated. We finally settled on a JavaScript difference tool by N. Fraser [10]. This then required minor modifications to alter how the output was formatted to suit the needs of our project.

We found building our timeline slider took much more effort than expected. To build it timeline

slider we are extending a slider from [11], which is an extended form of the slider from jQuery-ui [12]. Their slider allows us to draw labels on the slider, which were then required to extend this to allow the labels to be non-linearly spaced so that they can approximate the time differences between each of the dates where a change occurred. We plan to further modify this slider by adding a scented bar chart to help the user identify where to drill down for more details. The current implementation we have will detect when two labels are too close together and would cause occlusion. Instead of drawing them on top of each other, the layout is adjusted to separate them. While this allows each label to be seen individually, it does mean that comparing the length between two labels no longer preserves the amount of time between them. While there were a few existing implementations that looked promising for this task, they all proved to be buggy. Having done this, we have completed task 3 in our initial proposed timeline.

Working towards task 2, we have combined the file difference tool with d3.js, and the timeline to allow us to compare a mock-version of the data across multiple years. We have not yet completed task 2, however, as we have been iteratively modifying the implementation to support the true format that our data will be in. It is currently using a simplified representation that does not support the nested structure of the document. We anticipate that supporting the true document structure will require roughly one more day of work. In Figure 1 we present an example of our working prototype with the mock data that we have created.

We are updating task 3, as we have realized that a table of contents does not make sense with the data set we are using. Only the top-level section are given titles, thus any additional sections would not have meaningful labels. We have decided modify this display to use an icicle plot to display the changes made to each section. Since the law documents are always at most four levels deep, we do not need to worry about this display taking up too much room. However the user may be required to zoom in to a section to be able to see the individual colours of each of the sections if there are many. We will support zooming in and out with a double click, and navigation to a section of the document in the main diff view with a single click. Nodes in the icicle plot will be coloured based on the type of changes made within the corresponding section.

2.1 Timeline

Our updated timeline is presented in 1, with a short description of each task. We have made several changes to our original timeline. Task 3 is the first one that has been completed, so it has been moved to the top of our list. Tasks 1 and 2 we hope to have completed soon within one week, since we were unable to meet their original deadlines. We have also updated our perceived difficulty of task 5, and moved it forward in our timeline. This task pertains to adding a bar chart to the timeline slider, we believe this will now be easier than we once thought since we have already spent quite a bit of time working with the timeline slider code for task 3, and are now more familiar with what we will need to do.

Task 4 has been updated since we are now planning to use an icicle plot instead of a table of contents, we have also moved it's deadline back one week. Tasks 6, 7 and 8 remain optional tasks. We have our minds set on being able to adapt the Stack Layout [13] code to at least show the sections added and deleted over time, and we hope to be able to modify it to also show sections changed, however any of this will only be time permitting. We have decreased the difficult of each of tasks 6 and 7, however, because we now believe that modifying the data format will take minimal effort, and we have found an existing open source library that we can use as a base for task 7.

Number	Description	Difficulty	Date
3	Create time line slider to choose which two dates to compare	3	2015-11-23
1	Parse data so we can see the law at a given date	5	2015-11-30
2	Encode changes between two dates in main diff view	3	2015-11-30
5	Add bar chart to time line slider to scent widget	2	2015-11-30
4	Icicle plot to summarize changes across the document	3	2015-12-07
6*	Derive data for history flow view	1	2015-12-07
7*	Modify Stack Layout [13] to summarize all changes in law over time	5	2015-12-07
8*	Create view for summary of a given section over time	3	2015-12-07
9	Prepare final presentation	3	2015-12-15
10	Final written report	5	2015-12-15

Table 1: Our updated timeline. Tasks marked with * will only be completed time permitting. We will only begin work on tasks 6 and 7 if everything else is proceeding ahead of schedule, in which case, we would likely drop task 8. If we are behind schedule, then we will skip tasks 6-8 entirely.

References

[1] J. Jones and M. Harrold. Empirical evaluation of the tarantula automatic fault-localization technique. In *Proceedings of the 20th IEEE/ACM International Conference on Automated Software Engineering*, ASE '05, 2005.

[2] H. Schulz, S. Hadlak, and H. Schumann. The design space of implicit hierarchy visualization: A survey. *IEEE Transactions on Visualization and Computer Graphics*, 17(4), 2011.

[3] Y. Tu and H. Shen. Visualizing changes of hierarchical data using treemaps. *IEEE Transactions on Visualization and Computer Graphics*, 13(6), 2007.

[4] S. Havre, P. Whitney, and L. Nowell. Themeriver: Visualizing thematic changes in large document collections. *IEEE Transactions on Visualization and Computer Graphics*, 8:9–20, 2002.

[5] Fernanda B. Viégas, Martin Wattenberg, and Kushal Dave. Studying cooperation and conflict between authors with history flow visualizations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2004.

[6] Pack M.L. Plaisant C. Guerra-Gómez, J.A. and B. Schneiderman. Visualizing changes over time in datasets using dynamic hierarchies. *IEEE Transactions on Visualization and Computer Graphics*, 19(12), 2013.

[7] Diff Checker. https://www.diffchecker.com/. Accessed: 2015-11-07.

[8] CanLii. http://www.canlii.org/en/bc/laws. Accessed: 2015-11-08.

[9] Microsoft Office Word 2013. https://products.office.com/en-ca/word. Accessed: 2015-11-07.

[10] Neil Fraser. Diff.js. https://neil.fraser.name/news/2006/03/19/. Accessed: 2015-11-22.

[11] jQuery-ui Labeled Slider. http://bseth99.github.io/projects/jquery-ui/7-jquery-ui-labeled-slider.html. Accessed: 2015-11-22.

[12] jQuery-ui. https://jqueryui.com/. Accessed: 2015-11-22.

[13] Stack Layout. https://github.com/mbostock/d3/wiki/Stack-Layout. Accessed: 2015-11-22.