Storyflow - Tracking the Evolution of Stories

Andreea Muscalagiu 1456893
Sorin Adrian Robert Davidoi 1456867

June 17, 2015

Chapter 1

Problem definition

A narrative or story is any report of connected events, actual or imaginary, presented in a sequence of written or spoken words, or still or moving images [2]. Every story has characters who interact with each other and locations where events take place. The interaction of multiple characters between two adjacent time frames is a session. In a storyline visualization, these characters are represented as lines and relationships between them are proportional to the distance between their corresponding lines. Understanding how relationships between characters evolve over the course of the story is an interesting subject for various applications. Such applications could be analyzing tweets related to a particular story or the plot of a movie.

The paper Storyflow - Tracking the Evolution of Stories by Liu et al. deals with creating visualizations that help users better analyze a story, particularly the connections between characters and locations. By considering algorithms and techniques included in the paper, our task was to provide a visualization that facilitates this analysis. The result should be aesthetically pleasing, easy to follow, interactive and it should support hierarchical locations and the rendering of a large number of characters. This can be achieved by minimizing 4 metrics: line crossings, line wiggles, wiggle distance and white space. Line crossings create visual clutter and may lead to occlusion. Line wiggles represent how straight the lines are. The reason to minimize them is that wiggled lines are harder to follow and increase the visual clutter. Wiggle distance is related to line wiggles. Minimizing it leads to a more compact layout. The last metric, white space, is minimized in order not to waste screen space or have an unbalanced layout.

The optimization problem is split into two parts: discrete optimization to minimize the first 2 metrics and continuous optimization to minimize the last 2 metrics. Minimizing the number of line crossings affects the visualization mostly, so it is the first step in the algorithm and it consists of ordering lines, sessions and locations such that there is minimum occlusion. The second step deals with aligning as many sessions and lines as possible between adjacent time frames. Finally, to generate a compact and pleasing storyline visualization, the

wiggle distance and white space must be optimized. A set of interactions are also provided, according to the paper, such as adding, deleting, repositioning and straightening of lines, as well as bundling of sessions. To sum up, the user's tasks were the following:

- choose a data set to visualize
- minimize line crossings
- minimize line wiggles
- minimize white space and wiggle distance
- add real-time interaction

Chapter 2

Our Approach

We have decided to implement at least a part of the algorithms for optimizing the storyline visualization and also add some user interaction, such as highlighting one or more characters. We have used the data sets provided on Tanahashi's website [3] and transformed them into the JSON format. The algorithm we have chosen to implement is the ordering algorithm, which involves minimizing line wiggles in 2 steps: sorting locations and ordering sessions and lines. We would like to obtain a good initial layout. We have decided to follow the rules regarding how to represent characters and their relationships. Additionally, we have added a representation for locations as colored closed contours around the characters in that location.

Chapter 3

Related Work

Such a storyline visualization was first introduced by R. Munroe through his movie narrative chart [1]. The lines of the characters run from left to right and their relationships are mapped to the distance between their lines at every time frame. Lines are adjacent when characters are in the same location. This visualization was drawn by hand and has lead to research on automatically generating a storyline layout.

An existing visualization technique, designed by Tanahashi and Ma [4] produces a pleasant-looking storyline visualization. Unfortunately, it takes too much time, when having a large number of characters, because it is based on a genetic algorithm. Therefore, no user interaction is provided. The implementation in Python and the data sets are available on Tanahashi's website [3].

Chapter 4

Implementation

Chapter 5

Conclusion

In conclusion, we consider that creating visualizations using D3 is a satisfying activity, given that it takes little effort to obtain considerable results, while the algorithmic part is rather frustrating because of the difficulty with manipulating matrices in JavaScript. Still, we would like to try to implement the rest of the optimization techniques in order to see how they improve the layout of our visualization. We would like to add that we found it difficult to understand a scientific paper, and especially reproduce the steps described in it, as the authors are not always specific in their explanations. It would've been very useful to have an open source version of the algorithm in some programming language in order to compare the solutions and make sure that we understood the steps to be followed.

Bibliography

[1] Movie narrative chart. http://xkcd.com/657/.

[2] Narrative. https://en.wikipedia.org/wiki/Narrative.

[3] Tanahashi's website. http://vis.cs.ucdavis.edu/~tanahashi/storylines/.

[4] Y. Tanahashi and Kwan-Liu Ma. Design considerations for optimizing storyline visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2679–2688, 2012.