CS519 – Final Project Report

Pratik Krishnamurthy and Ajinkya Patil, School of EECS, Oregon State University

1. PROBLEM DESCRIPTION

Open-Source projects usually have a large number of contributions in the form of commits. The data related to commits in such projects is of interest to software evolution researchers who would want to find interesting patterns in the project from such data. Manually analyzing such data to answer research questions can be cumbersome. This visualization system aims to help them in quickly answering their research questions.

2. TARGET AUDIENCE/USER

The target audience/users we have identified for this visualization are *Software Evolution Researchers*. Researchers may want to understand patterns by looking at many similar datasets and try to draw conclusions. Based on our interaction with these researchers, the questions that they want to have answered are like: "Do developers commit more often through GIT or SVN?", "Is there a relation between date and commit size and commit frequency?" etc. These are some of the questions that we are hoping to address with the visualizations of the dataset.

3. HIGH & LOW LEVEL QUESTIONS FROM USERS

The high-level questions that the researchers have are:

Q1. Is there a relation between the date and the commit frequency (number of commits) and commit size (LOC)?

Q2. Do developers commit more often in Git or SVN?

Q3. Is there a relation between LOC and SLOC over time?

For each of these high level questions, the low level questions are as follows:

Q1. Is there a relation between the date and the commit frequency (number of commits) and commit size (LOC)?

Q1.1: How does the commit size and frequency vary with time throughout the project?

Q2: Do developers commit more often through GIT or SVN?

Q2.1: How many commits were made each day?

Q2.2: On what days did developers commit through GIT and SVN?

Q3. Is there a relation between LOC and SLOC over time?

Q3.1: How does LOC and SLOC vary with date throughout the project?

4. ABSTRACT OPERATIONS

One thing to note is that Lines of Code (LOC) is equal to the sum of Source Lines of Code (SLOC – raw code), comments and whitespace. Based on the research questions, the abstract operations that are necessary to answer each question are as listed below:

Is there a relation between the date and the commit frequency (number of commits) and commit size (LOC)?→ Retrieve the date, type of VCS and ratio (LOC / number of commits). Compare the ratios (LOC with the number of commits) for all the dates.

Do developers commit more often through GIT or SVN? → Retrieve the number of commits and type of VCS for each date. Compare the number of commits on each date.

Is there a relation between LOC and SLOC over time? → Retrieve the VCS type, ratio (LOC / SLOC) and date. Compare the ratios for all the dates.

5. DATA TYPES

Variable	Type	Expected Value
Commit ID	Nominal	Hash value (SHA1)
Commit Date	Interval	UNIX timestamp
Author	Nominal	John, Alex,…
LOC	Quantitative	100,5500,…
SLOC	Quantitative	5,11,…
Version Control Name	Nominal	Git or SVN

6. ENCODING STANDARDS

Since each of our questions are independent, we planned for different views for each question.

The first view will address question 1, in which we have the following abstract data types:

Ratio (LOC / Number of Commits)	Quantitative	Color Density
Date	Quantitative	Position
Type of VCS	Categorical	Color

We represent the relationship between the number of commits and the Lines of Code (LOC) over each date as a ratio (LOC / number of commits) and this is the most important data for this question, followed by date and type of VCS. We chose ratio as the encoding variable because the question asks for the relation between the commit frequency and commit size and ratio helps us get that relation. Ratio and Date are both quantitative variables while type of VCS is categorical. Using position, length, area, volume to encode a ratio doesn't make sense because the small variations in decimal values would not be accurately represented. Thus, we decided to represent ratio with 'Color Density' as it will help us compare the small changes in ratios over time and it is a way to encode quantitative data according to Mackinlay [1]. We chose 'Position' to encode Date because it is first on Mackinlay's ranking [1] for

Quantitative data. We chose ‘Color’ for type of VCS as it is second on Mackinlay’s ranking for categorical data [1].

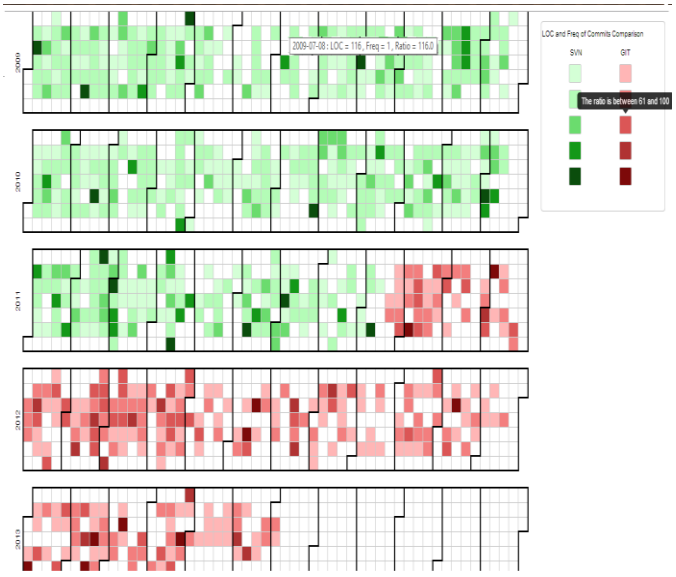

Figure 1: This is an example of the first view of our visualization that would answer the first question. There are two types of interactions in this example: a. When the user hovers over a date, the Lines of Code (LOC) committed on that date, the number of commits and the ratio of the two are displayed. b. When the user hovers over the legend, a tool-tip pops up to help the user interpret the visualization.

The visualization shows the variation of commit size and frequency for each date. The user can find out the variation looking at the color densities at the ‘Overview’ level. Next, the user can get the ‘abstract’ detail looking at each cluster (looking at color or color densities) and then the specific details for each date or month.

The second view will address question 2, in which we have the following abstract data types:

Number of commits	Quantitative	Color Density
Date	Quantitative	Position
Type of VCS	Categorical	Color

Number of Commits is the most important variables in this question followed by Date and type of VCS. Number of commits is quantitative and it makes sense to choose Density for comparing this data over time to accommodate slight differences in the count [1]. Date is Quantitative data and we represent it using position. Type of VCS is categorical and we represent it using ‘Color’ [1].

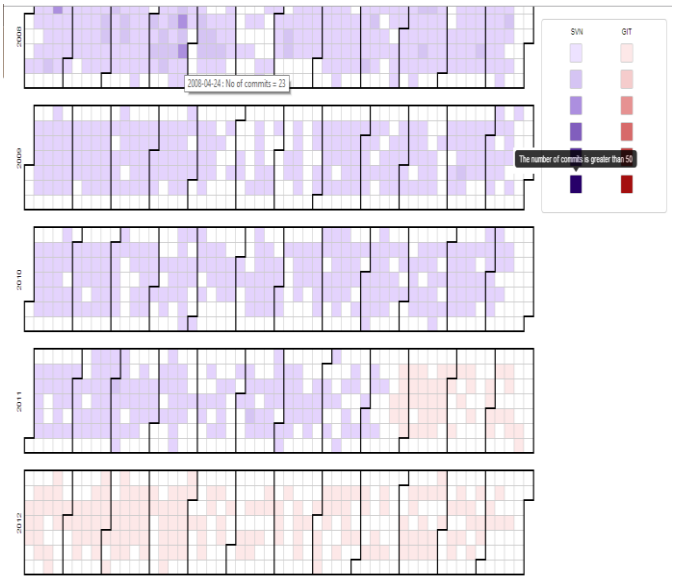

Figure 2: This is an example of the second view of our visualization that would answer the second research question. It shows the number of commits on each day and the VCS used to make the commit. It also has the same interactions as the previous one with difference of the ‘Number of commits’ that comes up on hovering over each date.

The third view will address question 3, in which we have the following abstract data types:

Ratio (LOC / SLOC)	Quantitative	Color Density
Date	Quantitative	Position
Type of VCS	Categorical	Color

We express the relationship between LOC and SLOC using a Ratio (LOC / SLOC). Ratio is the most important variable followed by Date and type of VCS. We chose ratio as the encoding variable because the question asks for the relation between the LOC and SLOC and ratio helps us get that relation. Ratio is quantitative and we choose ‘Density’ to encode it since we want to compare the ratios over time with slight differences in the number. Type of VCS is categorical and we choose ‘Color’ to encode it [1]. Date is quantitative and we choose to encode it using ‘Position’ [1].

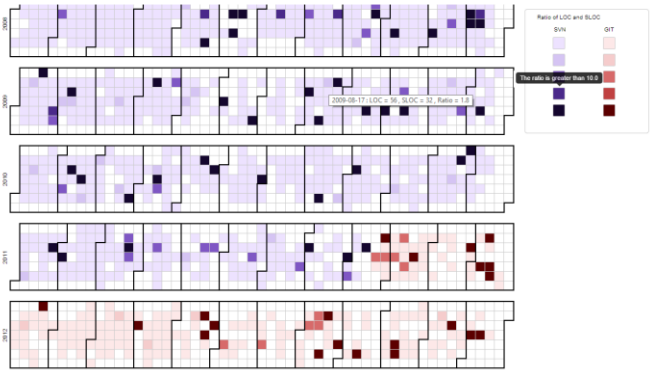

Figure 3: This is an example of the third view of our visualization that would answer the third research question with the same interactions

as the other except that 'LOC, SLOC and Ratio' are displayed on hovering over a date.

7. JUSTIFICATIONS

Clustering methods provide a basis for data aggregation. Clustering relates to partitioning a data set into subsets exhibiting a certain similarity. The clustering process also provides an abstraction of the data. Concentrating on the clusters, rather than on individual data values, allows for an analysis of data sets with a much larger number of tuples.

A technique specifically designed for the analysis of clustered time-oriented data is the Cluster Calendar View.

The Cluster Calendar View facilitates the comparison of cluster representatives (overview), exploration of the values of a single cluster representative (abstract detail), and exploration of daily and monthly values of interest (specific details). Cluster affiliation is presented indirectly by color coding.

Hence, this visualization is perfect to answer the research questions since our data is spread across time spanning many years. It accommodates our encoding decisions and has a compact and extensive structure thereby allowing the user to differentiate and spot differences in the densities (ratio) across time (date with position encoding). Also, our research questions require the users to get an overview of the data and detail on demand and the Calendar view helps us achieve this.

8. IMPLEMENTATION

We used D3 to implement the visualizations. The data had to be prepared for the visualizations. We used D3 features like nesting to compute the sum and count of the commits for each date. The calendar view implementation had functions to come up with the months and days (as rectangles). We computed values for the most important variable for each visualization and assigned colors to the rectangles accordingly. We used Bootstrap for tooltips over the legends and jQuery for other functionality.

9. USAGE SCENARIO

Depending on the question the researcher wants answered he will open the appropriate visualization for Q1, Q2 or Q3. Say that he wants to view the variation of the LOC with the number of commits over the project duration. For this, he will open the visualization meant to answer Q1. The calendar view that comes up will have all the dates encoded at their positions, ratios (LOC / number of commits) encoded with color density and the type of VCS encoded with color.

The user will look at the legends to understand what each color density mean in the visualization. The lighter shades mean low ratio while the darker ones indicate a high ratio. At a high level, the user will see that there are two different colors which the user will understand looking at the legend. Also, the user will see different shades of the colors throughout. Scanning through the visualization, the user will observe that the initial few days have darker shades indicating a high ratio (the lines of code committed were high compared to the number of commits on that day). As the user looks towards the end of the visualization where the data for SVN ends, the shades are lighter indicating that ratio of LOC and number of commits is low (the lines of code committed are almost equal to the number of commits). At a lower level, a user can look for particular months when the ratio is very high and try to find patterns. For details, a user could also hover over individual dates to get the actual numbers. An insight or hypothesis that a user could derive from this visualization is as follows:

Initially, the color densities for the ratio are darker which indicates developers might have committed more LOC in just a few commits indicating that the project might have been in the beginning stages of development. In the middle, there are equal number of commits being made and LOC being submitted indicating that the project could be nearing its completion. In the end part, looking at the color darkness, it seems like the developers are again committing a lot of LOC indicating that a major feature was being rolled out or the project was being completely revamped.

A similar use case could be applied to the visualizations for the other research questions.

10. PROBLEMS

There is no separation of dates where the data for SVN ends and that for GIT starts. There might be too many color densities for the user to analyze and understand the visualization. Also, the system doesn't provide the user the flexibility to filter the variables for visualization. The legends don't have labels over them because of which a user has to hover over each of them to know what the color density means.

11. IF WE HAD MORE TIME

The users could be given the freedom to choose the colors they want for the visualization. Also, there could be a feature where the user could choose a subset of the actual data to be displayed. Each month could have the month name at its top for easy search of date. Also, on hovering over each date, there could be a tooltip visualization that shows the details of each commit on that date. There could be a feature where the user could filter the data according to the data he/she desired to visualize.

12. CONCLUSION

The calendar view manages to answer all the research questions of our users without having to look at other visualizations. We hope to improve the visualization further based on feedback from our users.

REFERENCES

[1] S.K. Card and Jay. Mackinlay, "The Structure of the information visualization design space".

[2] https://github.com/mbostock/d3/wiki

[3] http://bost.ocks.org/mike/

[4] Aigner W et. Al, "Visual Methods for analyzing time-oriented data".