Design 2 - Github Commit Graph - Group A2

In this design workshop you will sketch multiple solutions to visualize the commit graph and the branches of a git repository. Submit one document as a group which contains the analysis, sketches and the reflection with the final visualisation.

PART 1 - ANALYSIS

Take a look at various Github Network Graphs. Here are two examples 1, 2.

Think about how these networks are different. Analyze the "dimensions" of these networks. What are the relevant attributes (e.g., commits, users, branches, commit size, etc.) of these representations?

First example:

- The first example emphasizes on the different development trees. A clear distinction is visible between the Master branch and the feature development branches by means of colour.
- The commits are represented by dots and display easily who commits and what is committed.
- It's hard to derive how many contributors are participating to the repository.

Second example

- The black line represents the system which gets commits from the Continuous Integration system(jenkins). The coloured lines represent direct patches on the master branch by different contributors.
- It is hard to view the different contributors which are pushed by the continuous integration system.
- The pull requests are confusing and do not faithfully show how the request is merged.

In both examples it is not possible to view the commit size simply by looking at the graph.

What other attributes could be relevant in this graph? Write a list of all the attributes your visualization could show.

- Tooltip could show more information, for example: which files are changed / added or removed.
- Branch names are not displayed on all branches.

Are there different roles, i.e., different types of users who might want to achieve different things? Write a list of user roles.

- Contributing
- Reviewer
- Information
- Progress

Think about which tasks a user of your visualization might want to achieve. Write down a list of tasks.

- See different branches
- See different contributors
- See merges
- Read commit messages
- See added/removed/changed lines in each commit (progress)

Identify one role that you want to design your visualization for. Prioritize your task and attribute lists based on this role's needs.

Contributing:

1. See difference in branches
2. See different merges
3. See contributors
4. Reading of commit messages
5. See added/removed/changed lines in each commit (progress)

PART 2 - SKETCHING

Design two alternative visual representations for the Github Network. You should design for an interactive system, i.e., you should not assume that you have to fit all content onto paper. You can refer to Heer's article Reading 3 for the examples of network visualisations, browse D3 website, and feel free to use any other sources.
Here are some questions to consider:

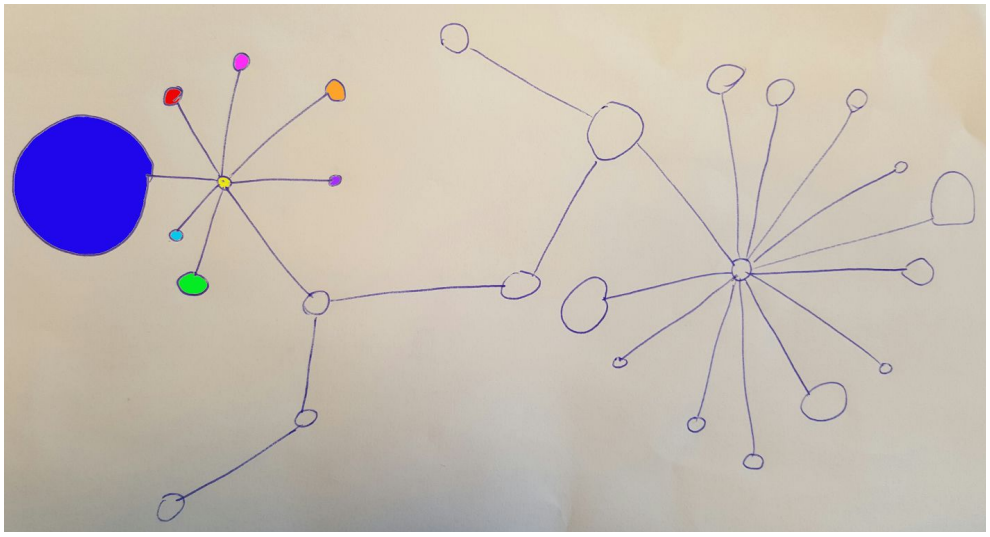
Decide on which visual variable to use for which attributes of the visualizations. Consider the strengths and weaknesses of visual variables that were mentioned in Carpendale's article Reading 1 (also briefly discussed in this week's lecture: Process). Use the strongest visual variable for the most important attributes of the data.

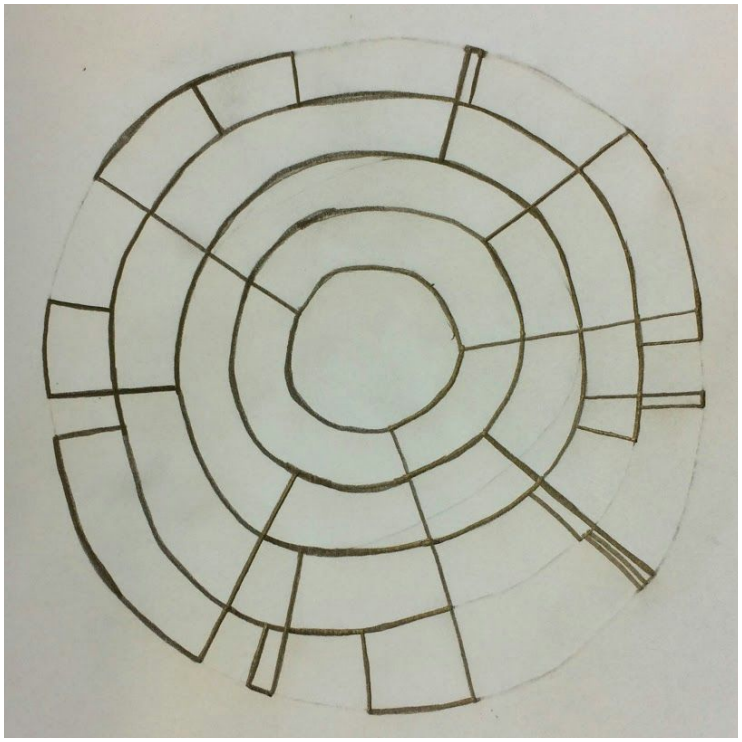
The strongest visual variable is definately how the branches correlate to the master, eg how the commits are formed back into the master.

Do you think it is necessary to represent every single commit as a separate node? Could you think of ways to aggregate this?

No it is not. The commit nodes can be aggregated to features for example. By representing them as features, it is possible to display a better flow of development. Below, in the first sketch, we have an example of such a representation, a sunburst. Every branch/feature has its own part in the circle, sub-branches are stacked on top of it. The more commits a branch has, the larger it is.

The second sketch is a force-directed layout. Each node represents a branch/feature, and the size represents the amount of commits.





Do you think that every contributor needs a “row”, as on the default network view on github? Could you think of a smarter way to summarize those?

We don’t think every contributor needs his/her own row. The primary objective for viewing this graph, the network tab, is to view the branches and the commits made in this repository. If the user would have wanted to view the contributors there’s a different part of the website dedicated to that purpose.

A smarter way to summarize those is by having a separate page with steam/curve diagrams which is already applied in the contributors tab on Github.

Is a node-link diagram the appropriate representation? Or should you consider alternative graph representations?

A node-link diagram could work, however not for the tasks we have defined, because it does not accurately show progression.

PART 3 - Group Reflection

Time: 40 minutes, or as long as you like.
Take your analysis and ideas and discuss it again. Can you find a consensus? Come up with one visualization that you agree is ideal. Reflect on the process that you went through to come up with your final design (refer to this week's lecture: Process for inspiration on design process).

For our purposes the sunburst appears to be the better option. To define this we first prioritized the different aspects of a git view that a contributor needs to see at first sight. In our opinion, the most important thing a contributor can see is the size/code difference of branches and the insight in the commits. We started by looking at different graphs which could realize this priority. The sunburst (zoomable[1]) graph, which we sketched above, looks like the best match because of the following:

Pro's:

- The graph supports a large amount of branches in a compact view.
- The graph does not get bigger as the amount of branches or commits increase.
- It provides an overview of new feature branches at first glance.
- Any problems due to scaling could be solved by adding a zoomable effect.

Con's:

- Visualizing the branches/commits chronologically is difficult.
- Large systems are still hard to visualize.
- Relative size is more difficult to determine when using a circle.

[1] http://bl.ocks.org/kerryrodden/477c1bfb081b783f80ad