Author: Jackie Cho
Mentor: Justin Gosses
Title: Graph Visualization of Wikis and AutoML for Network Cyber Security

During an internship with the Office of Chief Information Officer Data Analytics team, a prototype of a large data visualization project, which visualizes the NASA Wiki pages and the links between them as a series of large graphs using a JavaScript visualization library, D3.js, has been delivered and transitioned to its deployment stage. The goal of this project is to allow the hundreds of people at Johnson Space Center working in flight operations (FOD), extravehicular activities (EVA) and many other organizations that use wikis to see the larger structure of the wikis, sometimes 10,000s of pages, in ways not previously possible. Traditionally, users experience the wiki on a page by page basis and navigate to a new page via either the search box or hyperlink on one page that takes them to another page. As the wiki develops over time organically as a result of non-centralized efforts by many people, there is no central table of contents. By representing pages and links between them as graphs as well as splitting and grouping graphs based on categories, we can show end-users the larger structure for the first time.

We suspect being able to see what pages are two or five links away in a single image will speed navigation relative to the current situation where pages must be navigated one link at a time with visibility of only what is one link away. By showing users the structure of the wiki, they can interact with the wiki in a different manner and ask new questions based on the visualized information. Having such visualization also incentivizes admins to link their pages to allow for better navigation and flow of information. Some questions answered by the project include: “What are the most common categories?”, “How do categories link to one another and how are they grouped?”, “What are the different structural shapes of individual categories?” and etc. The main challenges to architecting the project were that we wanted the end product to exist entirely inside of a wiki with no extra server or maintenance requirements while visualizing a large amount of information that was both too big to load quickly and too complex for a human to understand quickly.

The early stage of the project involved experimenting with existing MediaWiki extensions to explore the data through a Docker container and its implementation to the backend server. It now serves as an API to scrape the wiki data in JSON format, which is then preprocessed and prepared with Python scripts for visualization. Due to the high volume of data which significantly slowed down the loading speed and reduced the readability of the graphs, I partitioned the data into categories and applied to multiple graph models, one being a 3D-force-directed-graph with UI capabilities which is implemented with dat.GUI, a lightweight controller library for JavaScript. Some UI features include altering link opacity, strength, visibility, and node size, highlighting node/link on hover, and focusing on node and its neighbors by click, which allow users to control and manipulate the graph for an interactive visualization and enhanced readability. A treemap is also implemented to visualize the overall structure of the Wiki and improve navigation through the different categories, which are filtered by “descriptive” vs “content” using a string-matching algorithm with Python.

As an alternative to the treemap, we applied our data to a 2D-force-graph which visualizes individual categories by their size, the number of links between the categories, and average number of links within each category, giving an in depth information of the categories and their relationships. To structure our nodes and links in a more readable format, a clustering

algorithm called “force-in-a-box” is implemented on the 2D-force-graph to group the nodes in a treemap cell by keyword, which is found by creating a dictionary of recurring strings in the category names. The data is also applied to an adjacency matrix which also visualizes the overall structure of the wiki categories and their links to each other, but with various filtering options such as by name, crossing reduction, and bandwidth reduction.

Our current findings show that each graph model has different strengths and weaknesses and thus a single graph model does not meet all of our requirements. Currently, we are experimenting with merging the individual graphs to extend the widest selection of features to the users and deploying the graph application as a “Special Page” extension on the JSC Wiki site. Some immediate improvement to the graph include automating the data preprocessing steps to a single Python script and applied as a Cron job to update the dataset periodically, and translating it to JavaScript to run on the front-end instead.

Future work involved in Phase II of the project include implementing a “page-centric” representation of the graph, which I have created a demonstration by isolating a dataset for a single page and displaying only its 1st and 2nd neighbors which are color coded. The purpose is to allow users to visualize the steps it would take to reach their target page from one of its neighbor pages, which is a feature that potential has the most usability for the average Wiki user. Other ideas include implementing a recommendation system based on the semantic “property” information of the wiki pages or use natural language processing to predict the STI topic with highest probability on each page and use that as as a “pseudo” wiki category based on content. Evidently, there are many ways this project can evolve to create new features and thus currently serves as a foundation for various future projects for the Data Analytics team.

With the conclusion of this project, I will transition to a machine learning project on detection of network traffic associated with malware servers. An initial attempt on a small subset of network traffic has already been completed. I will use at least two seperate automated machine learning (AutoML) frameworks to test out a wider variety of algorithms and auto-generated features for improved scores. The goal for this project will be to both improve prediction metrics on the malicious network traffic prediction project and document pros and cons of the different AutoML frameworks to help the team decide which to apply on future projects.

This internship did not only educate me in the field of data science and analytics, but also introduce a newfound interest in pursuing this field in my further studies and career. It has also influenced me in considering research in machine learning, artificial intelligence, or natural language processing in my undergraduate studies and possibly graduate school. With a new appreciation and respect for data science and its extensive application to our daily lives, I hope to continue learning beyond this internship.

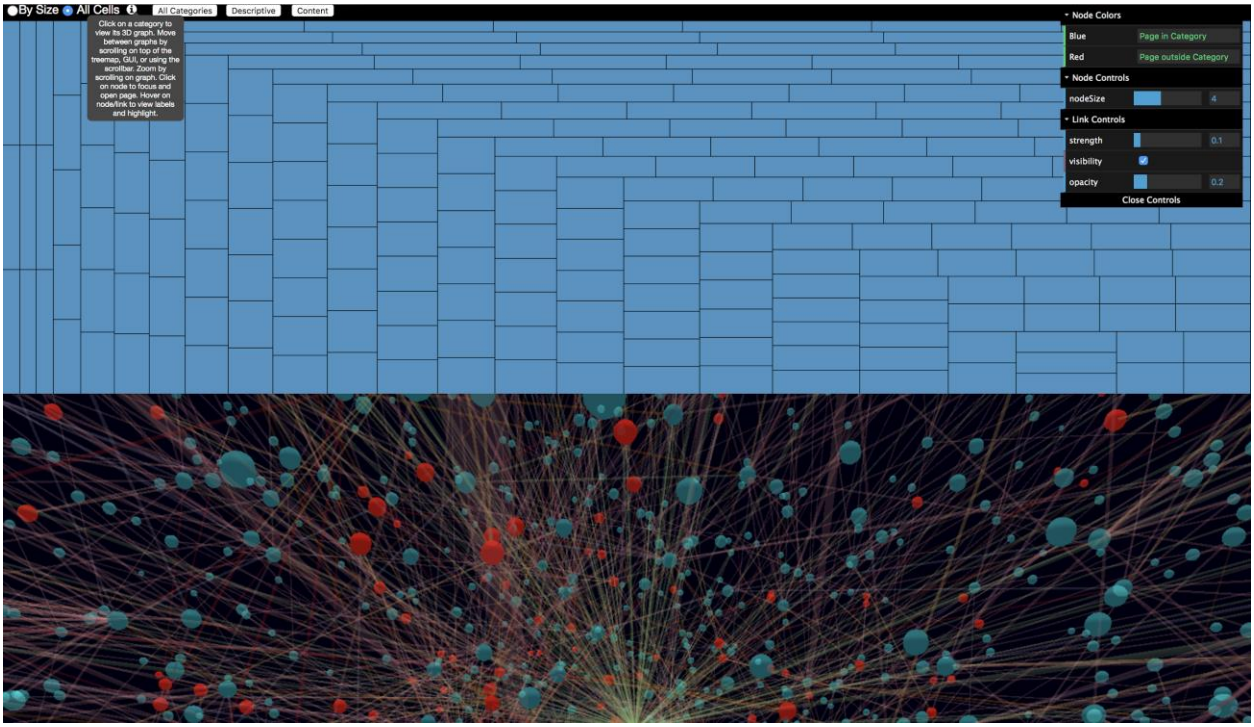

Figure 1: Treemap of the ISS Wiki categories (redacted category names) and a 3D-force-graph of one of the categories

Figure 2: Full view of the 3D-force-graph of a single category, showing pages within the category (blue nodes), and linked pages outside the category one neighbor away (red nodes) with links colored by type. UI control panel using dat.GUI library. This image demonstrates how complicated the hyperlink relationships can become.

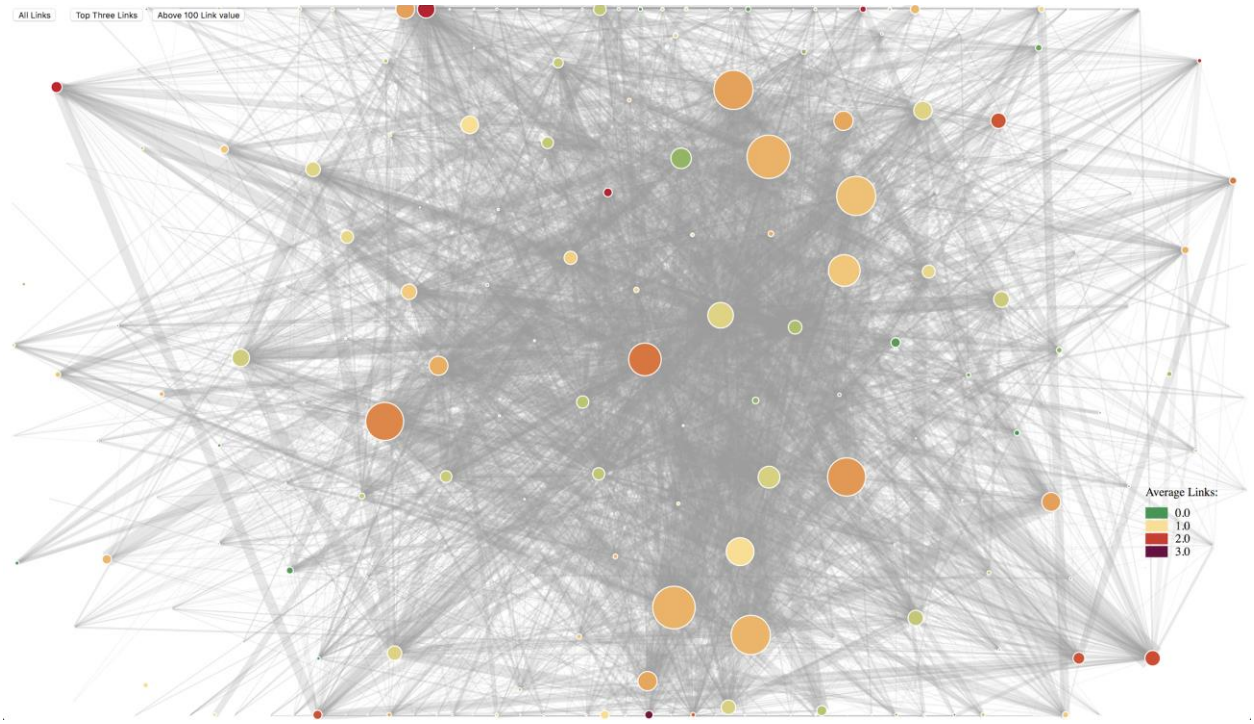

Figure 3: 2D-force-graph of the categories and their links to each other, showing number of links between categories (link width), category size (node size), and average number of links within a category (node color).

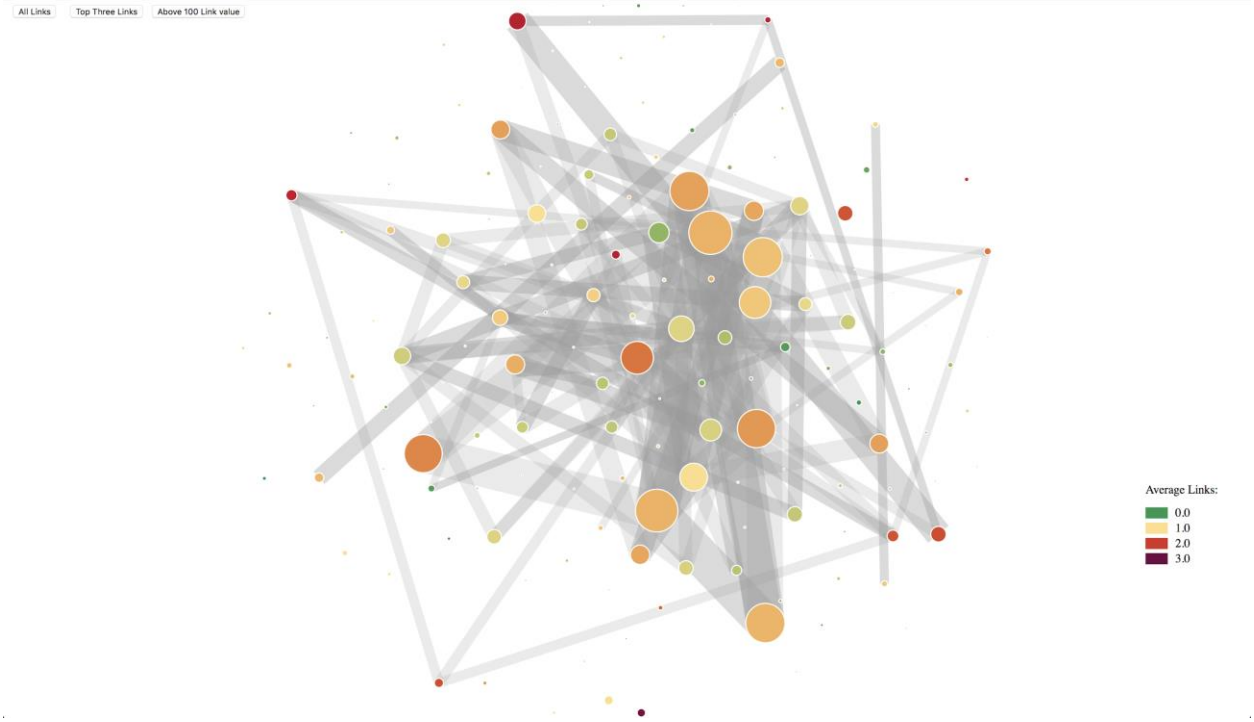

Figure 4: 2D-force-graph of the categories showing the top three links above 100 in strength for each category. Same as figure 3 but links are limited to the top three.

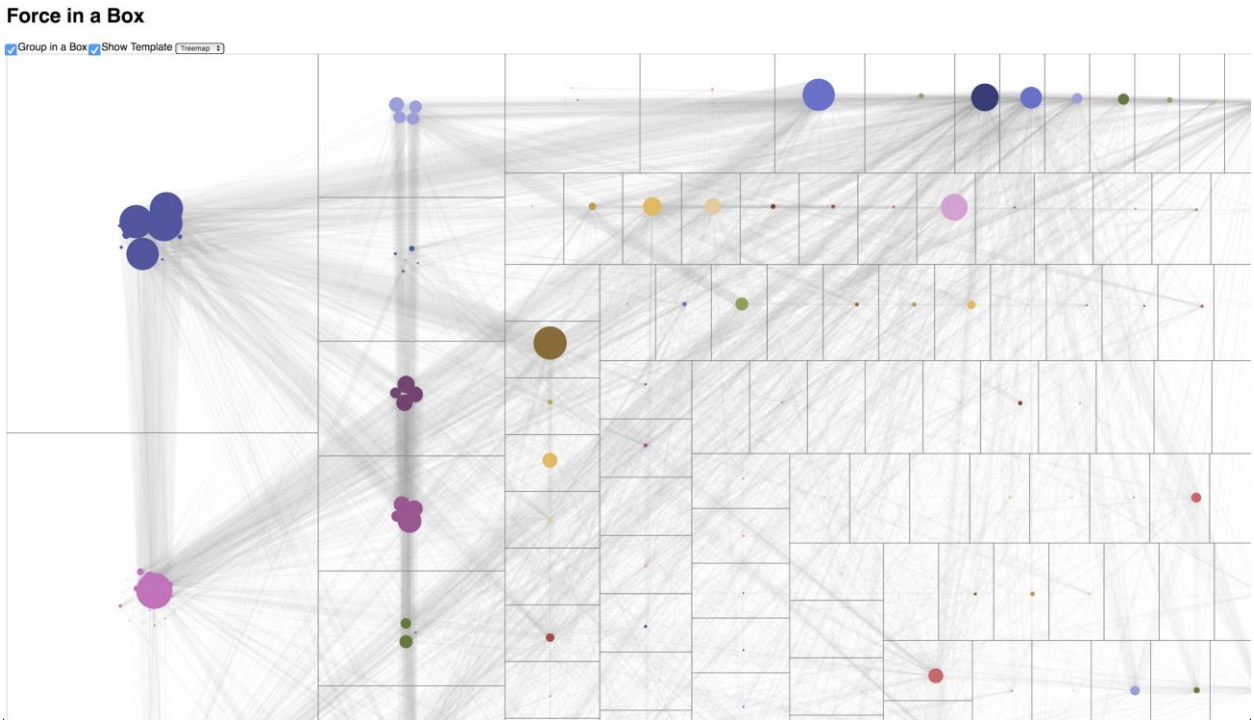

Figure 5: Force-in-a-box algorithm on the category treemap grouped by keywords. All category and keywords redacted.

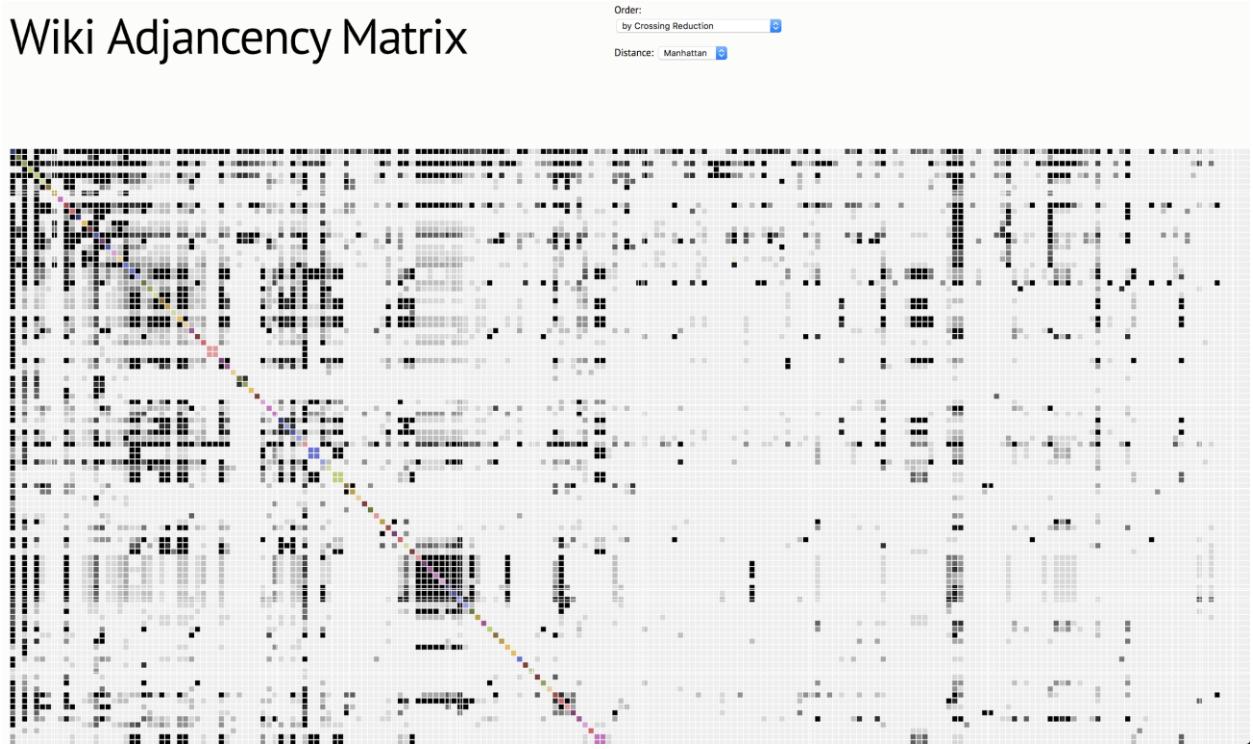

Figure 6: Adjacency matrix of the wiki categories and their links. Filtered by crossing reduction. (Redacted category names would appear as column and row names)

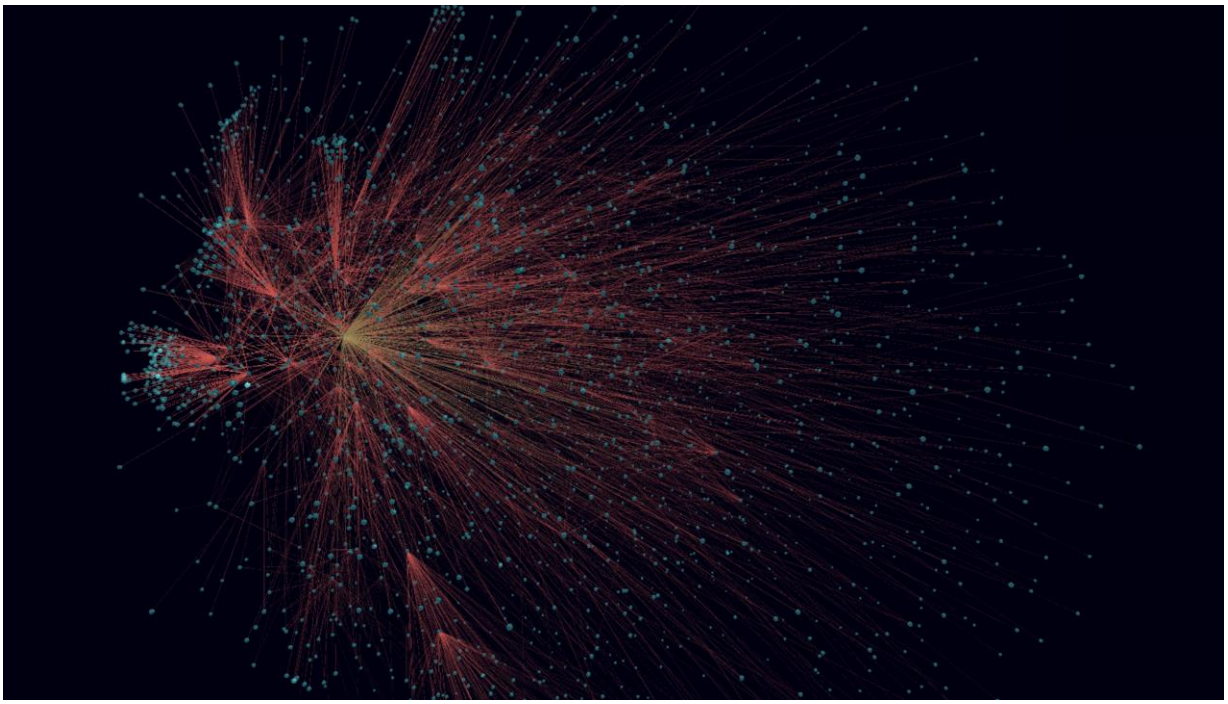

Figure 7: Page-centric view of the 3D-force-graph showing 1st (yellow links) and 2nd (red links) neighbors of a specific page.

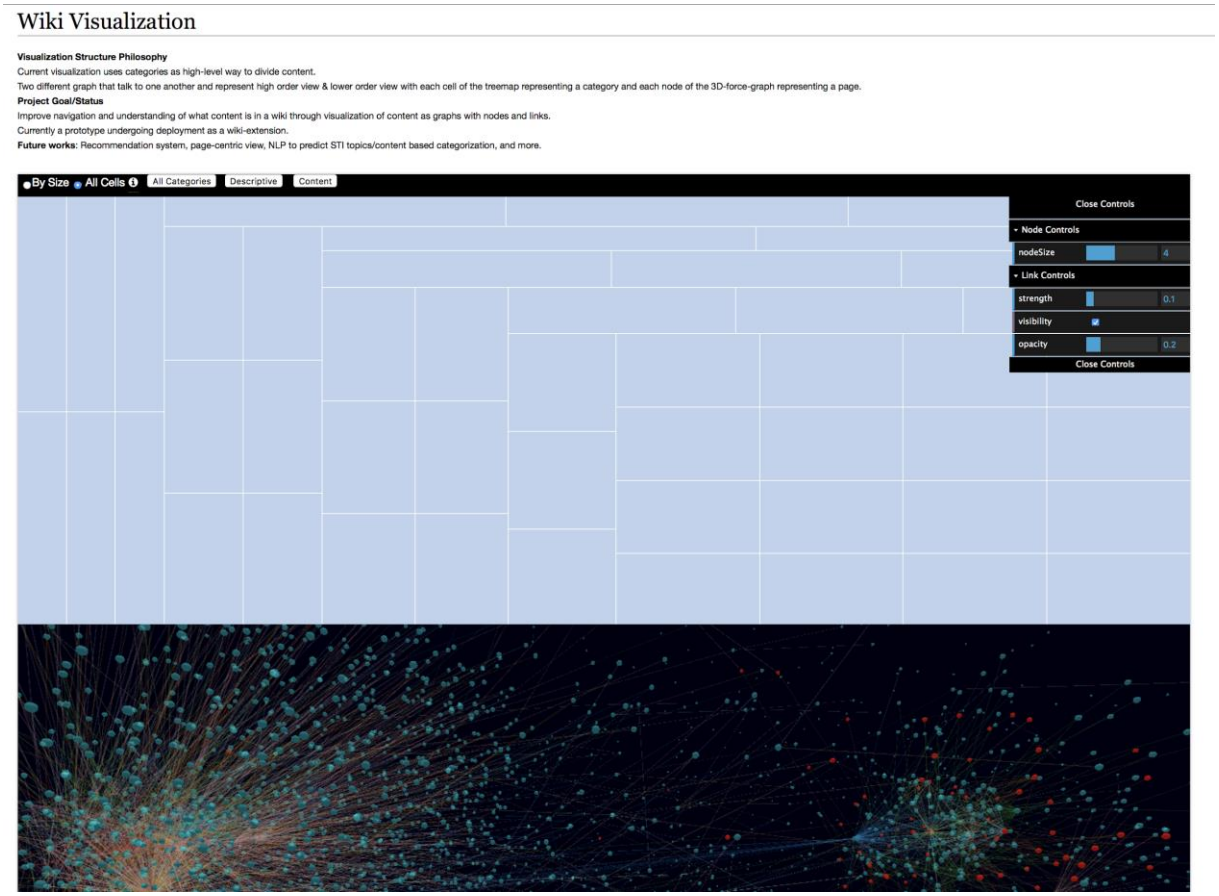

Figure 8: Treemap and 3D-force-graph deployed as a Special Page extension on a local wiki Docker container. This is how the application is seen when run inside the wiki. Content-based category treemap (redacted category names).