Data and Info Visualization - Stackoverflow Network

Andrew Struthers, Gihane Ndjeuha, Kollin Trujillo, Nathan Chapman, Nick Haviland

March 2023

Introduction

In this project, we will be discussing network systems and their graph visualizations. Specifically, we will be investigating the relationship between how users ask questions and receive answers on stackoverflow.com. We will be working with data from the Stanford Network Analysis Project (SNAP), with our specific goal of investigating emergent phenomena such as the existence of relatively small groups of users that are all active on each other's posts (i.e. finding cliques), and the time-dependency of user activity on the platform. We will be using Python and Mathematica to generate two different visualizations of the same subset of data so that we can compare and contrast the visualization methods of both programming languages. By comparing multiple methods, we hope to see the same emergent phenomena in both visualizations, while experimenting with different libraries that help make clear and high quality visualizations.

Community Structure

Since the main pattern we are looking for is a clique, we first need to look at how exactly a clique can appear inside of a network. This concept of grouping multiple nodes together is known as "community structure". Community structure refers to the tendency for groups of nodes in a network to become more interconnected than the rest of the network. Examples of community structures can be seen without needing to be drawn in a graph: in a social network based around students, those at the same school would be further interconnected. Community structures give insight into the function of the network and provide further insight that can be difficult to see with just a network.

We can use various different algorithms and classifications to clearly pull out the underlying community structures in a graph. Those algorithms include Hierarchical Clustering, Girvan-Newman Algorithm, and Modularity Maximization. Hierarchical Clustering is when pairs of nodes are evaluated based on some measure of similarity, then grouped as a community if they are similar enough to each other or dissimilar enough to other communities. The Girvan-Newman Algorithm finds edges between nodes that connect communities and delete them, leaving communities separate within the network. We can see an example of this in the below figure. We can see that edge 9 was deleted, seperating the community ABC from the community DEF. Finally, Modularity Maximization will find a measure of modularity from searching over different possible clusters of nodes within a network. Typically, this fails to detect small communities, but is good at highlighting large communities within a graph.

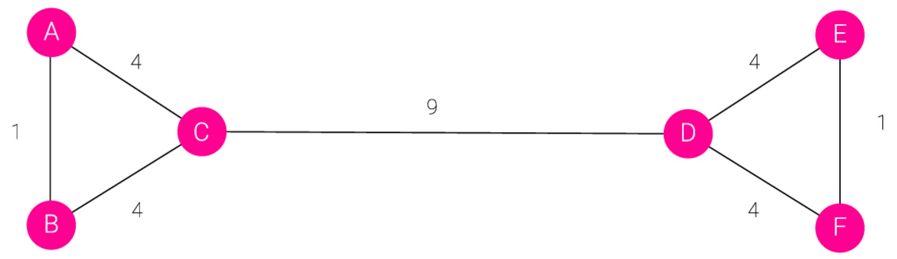

Figure 1: Original graph before Girvan-Newman Algorithm

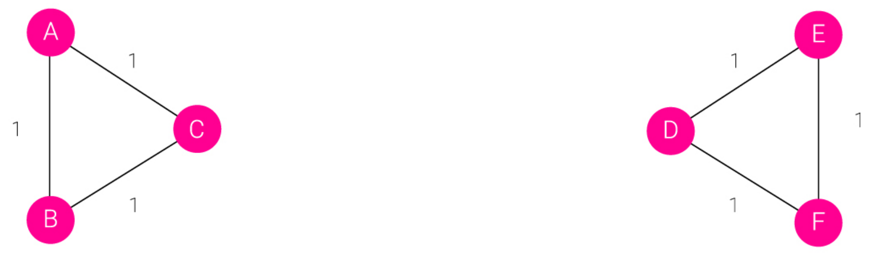

Figure 2: Graph after edge 9 was deleted as per the Girvan-Newman algorithm

Data Selection

Our specific dataset, sourced from SNAP, is a temporal network of interactions of users on Stack Overflow. There are three different subsections of the data, each with a slightly different meaning. The three types of interactions are each represented by a directed edge (u, v, t), where:

- User u answered user v's question at time t in the dataset **sx-stackoverflow-a2q**
- User u commented on user v's question at time t in the dataset **sx-stackoverflow-c2q**
- User u commented on user v's answer at time t in the dataset **sx-stackoverflow-c2a**

There is also a dataset that contains the union of all three graphs, labeled **sx-stackoverflow**. The union of all three graphs contains a total of $2,601,977$ nodes, and a grand total of $63,497,050$ temporal edges. We are just going to be examining the dataset **sx-stackoverflow-a2q**, and we will only visualize a very small sliver of the full $17,823,525$ temporal edges. Since our goal is to identify some emergent phenomena such as cliques within the data, it isn't necessary for us to plot all $2,464,606$ nodes. That would simply be too much data for a clean visualization. With this in mind, we will only be working with the first 5 days of data, roughly 1000 nodes. This should be plenty of data to see the community behavior we are looking for.

Python Visualization

One of the first things we can do for visualizing this data is to plot the data using Plotly. Using some careful coloring, we can get the following visualization.

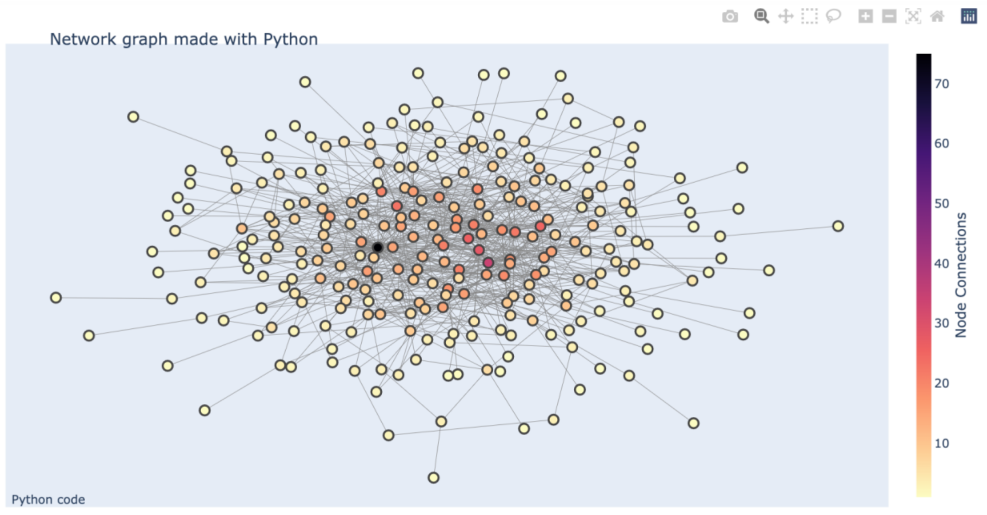

Figure 3: Clear, color coordinated network graph

As we can see, many users have fewer than 10 edges leading out from their node. This represents the notion that, of the users that answered questions, most answered very few times. We can see that a handful of users answered in the 30-40 range of questions, but one user clearly stands out from all the rest, having answered more than 70 questions in the 5 day time frame this data is displaying. We will talk more about this user later.

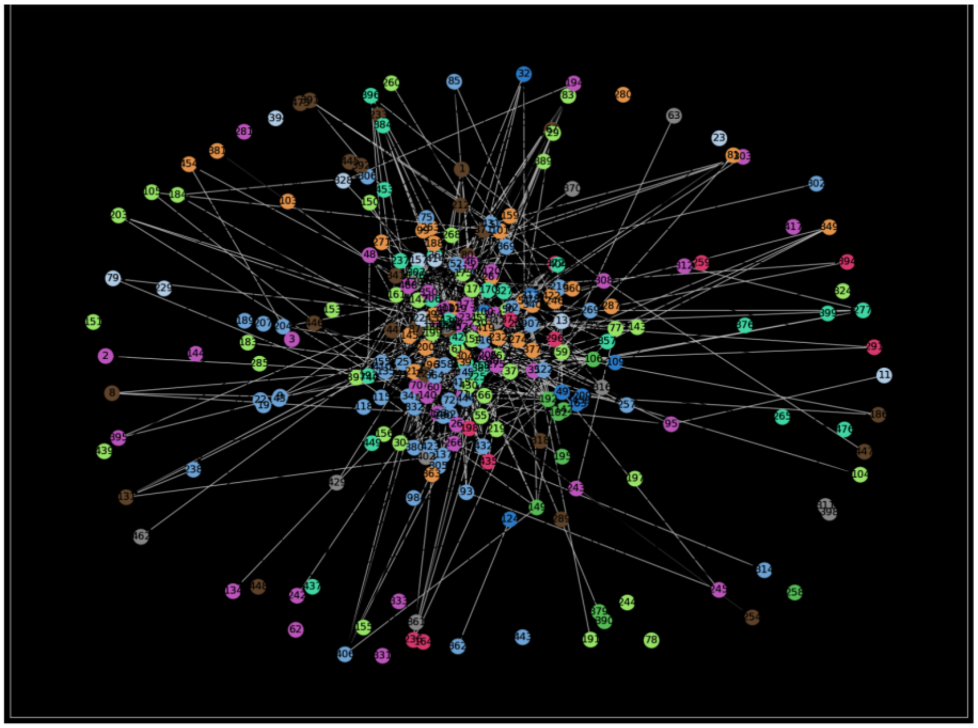

Figure 4: Simple community graph using the Gervin-Newman algorithm

In the figure above, we can see the same graph represented with much different coloring and labeling. As opposed to the figure above, which simply displays each of the nodes and their respective directed edges, the visualization below highlights each node based off of their assigned community, as determined by the Gervin-Newman algorithm. The Python package "networkX" provides the Gervin-Newman algorithm as a built in algorithm for analyzing network data. We can see here that one of the largest communities is represented by nodes that are light, almost pastel, green. This starts to demonstrate the community effect we were hoping to see, since there are very few unique colors in this community graph relative to the number of nodes being analyzed.

Mathematica Visualization

In addition to many visualization tools, Mathematica has a built in function for implementing community structures on a network, CommunityGraphPlot, which can generate community structures or take in user given communities. Looking at the data in some simpler views first, however, can provide some interesting results. Below, we can see a histogram of the user response counts. We can see that, of the users that answer questions, most users answer very few questions. The average user that does provide some number of answers seems to answer only about 1 or 2 questions. Very few users from our subset of data answered more than 10 questions.

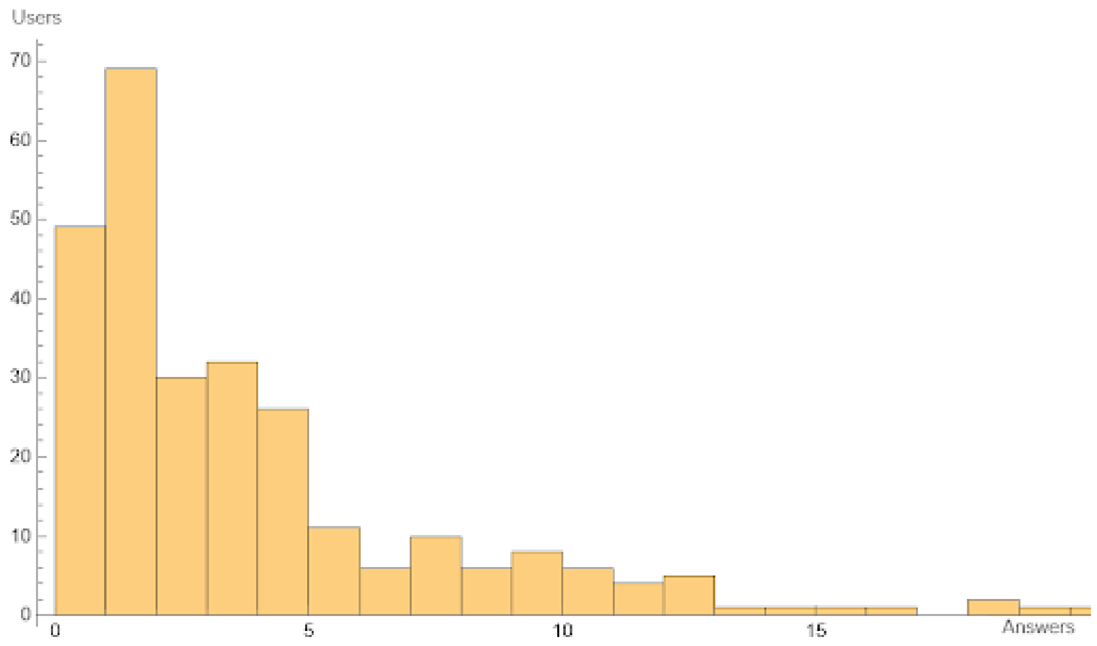

Figure 5: Histogram of user response numbers

Below, we can see a histogram of the time series data plotted, showing us the frequency of when users were actively answering questions on Stack Overflow. In the time series histogram, the most interactions happen late at night, but we must keep in mind that the data is in Pacific Standard Time, and doesn't contain information on the timezone the active user was in.

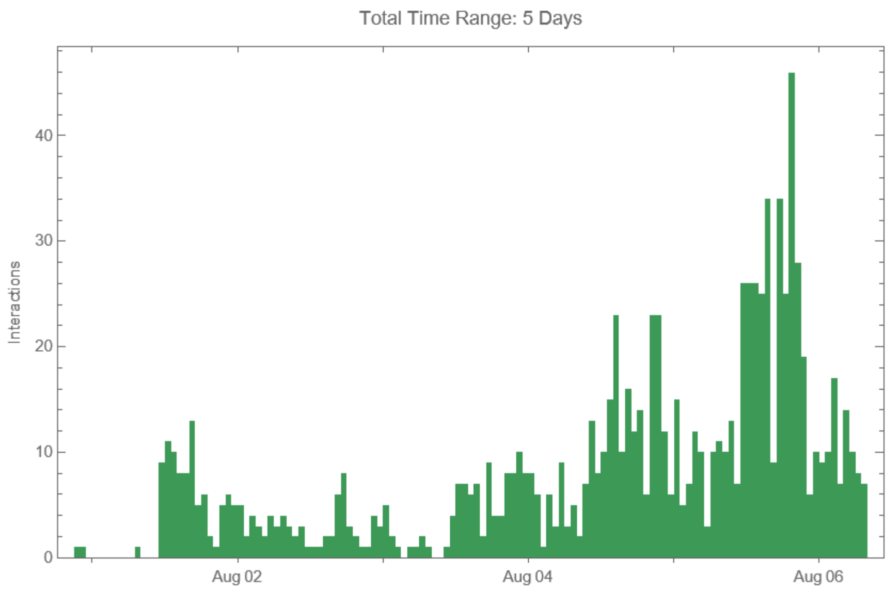

Figure 6: Histogram of user response times

Here in Figure 7 we can see the only cyclic relationship in the data. A cyclic relationship is when two nodes in a directed graph both point to one another. This forms a closed loop where user A answered user B's question, and user B also answered user A's question.

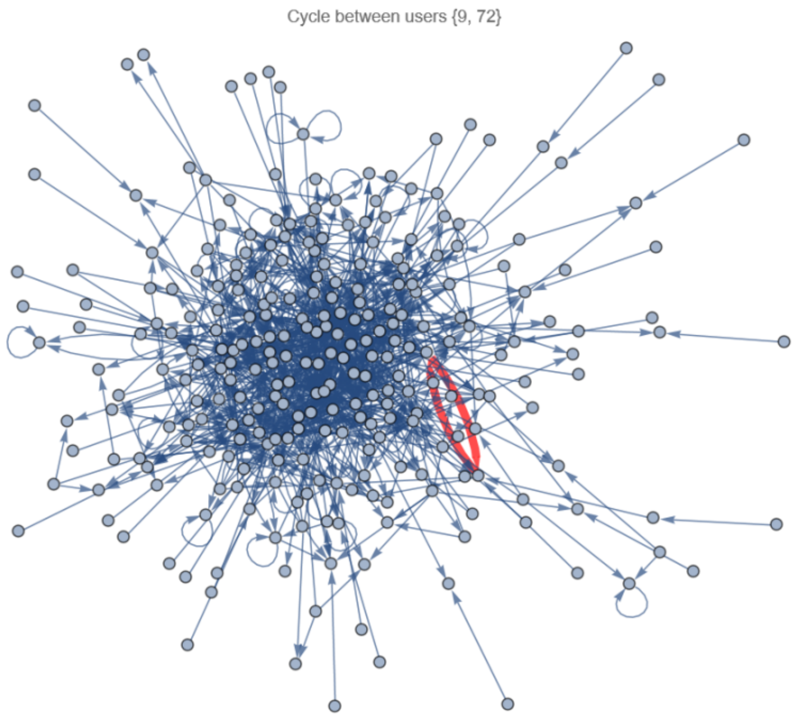

Figure 7: Only cyclic relationship in the data

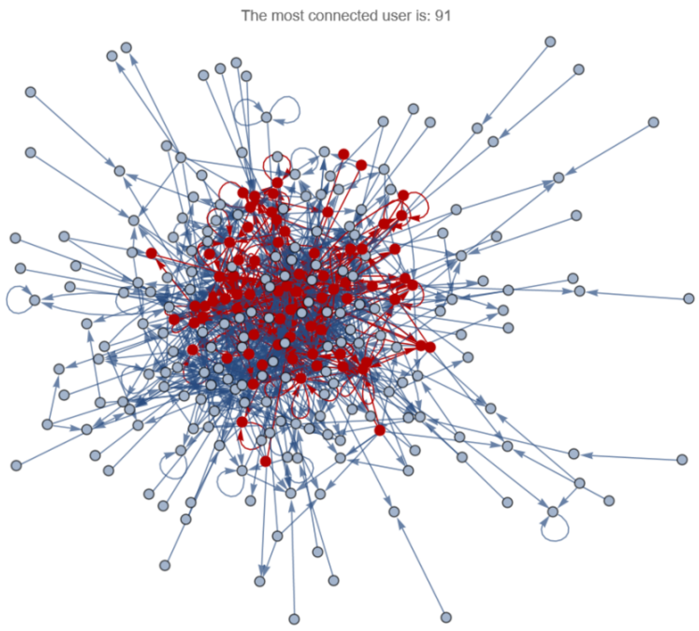

Figure 8: All 91 connections related to the most connected user

Above, we can see the most connected user. This is the user very clearly highlighted in Figure 3, the one user that had more than 70 directed edges. Each one of the red nodes represent users that have interacted with this one user at the center of the subgraph. Each edge represents an answer that either the most connected user provided to one of these highlighted users, or an answer from one of the users to one of the most connected user's questions. This subgraph details all relationships with this user, and as we can see from the graph, there are a total of 91 connections.

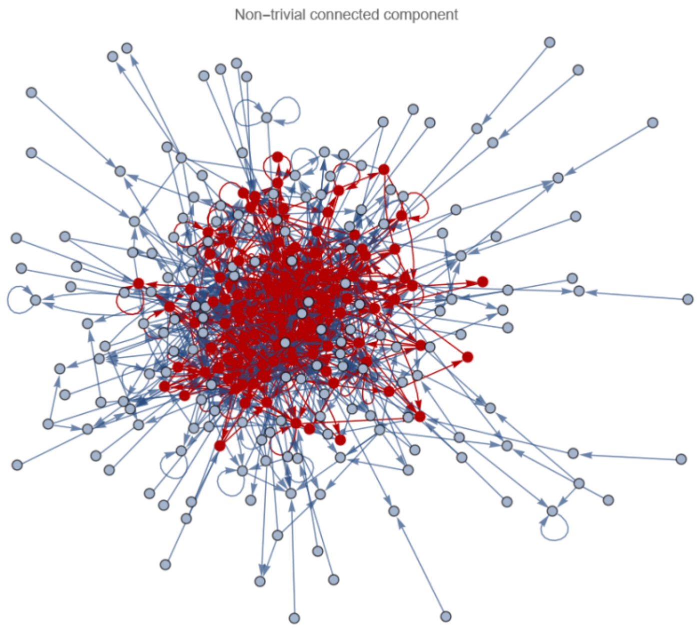

Figure 9: Non-Trivial Connected Component subgraph

In Figure 9, we see the non-trivial connected component subgraph. This is a subgraph where, regardless of the node you start at, you can successfully navigate to each other node in this subgraph through a directed edge. This, as well as the most connected user graph, are communities in their own sense.

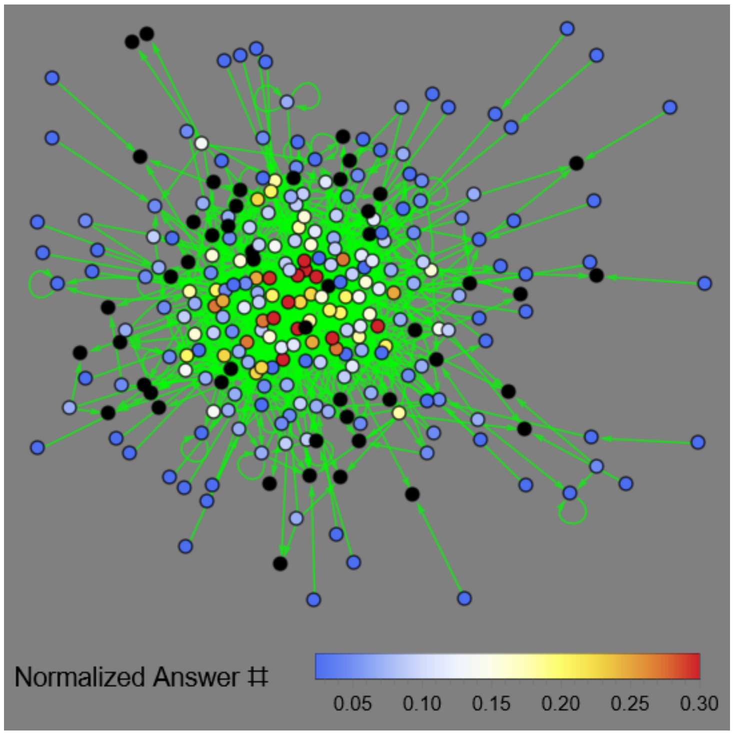

Figure 10: High contrast normalized graph for higher resolution of the average user

For Figure 10, we have made a graph that attempts to increase the resolution of the visualization. This graph is showing the network with the user nodes colored according to the number of answers they have posted. Black corresponds to no answers, and the rest are colored with blue being the lowest and red being the highest, with white in between. The absolute values on the color legend correspond to the normalized number of answers, or rather the number of answers of each user, scaled by the maximum number of answers any user has provided. We tried to accomplish this by normalizing the number of connections of each user and carefully choosing the color scheme so that the large majority of users who only answered one or two questions, had more distinction between them, and the color scale wasn't thrown off by a few very prominent users. Comparing this visualization with Figure 3, we can see that the high contrasting, bright green, edge color mixed with the colored normalized nodes, the users that only answered a few questions have much more distinction now. These high contrast colors, although typically not used for clean and appealing visualizations, serve to highlight the subtle differences between the large majority of users in this data. We can see that the super active users are still highlighted in bright red, but the distinction between them is much less noticeable than the distinction between the different blue-shaded nodes. This visualization is a good tool for showing how we can manipulate data and change the color scheme to highlight specific patterns and behavior that would otherwise be lost using more "typical" visualization strategies.

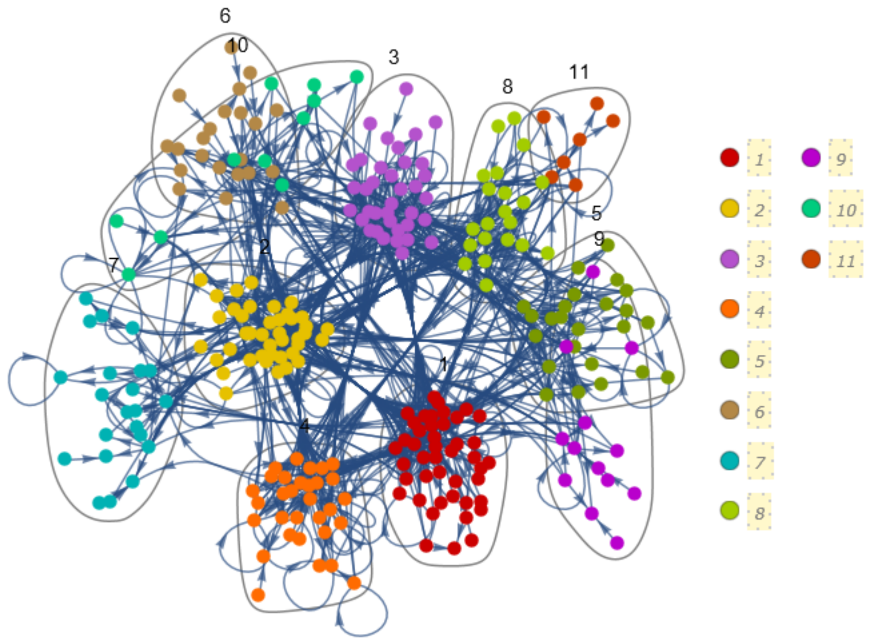

Figure 11: Distinct community visualization

Finally, in Figure 11, we can see a demonstration of the CommunityGraphPlot function in Mathematica highlighting each of the 11 distinct communities in our network. Each different color assigned to the nodes represents a different community, as determined by the algorithms discussed earlier. This clearly shows the clique behavior we were suspecting we would find inside of this data. One way these communities could form, we hypothesize, is because of the different programming languages one can receive help about on Stack Overflow. Many different programming languages and related bugs or questions can be found on Stack Overflow, and it seems to make sense that users knowledgeable in C++, for example, would frequently be posting questions and supplying answers to others working with C++. We did not have the data to delve in to this theory more, but each one of these communities could potentially represent a group of users forming a clique due to their programming language of choice.

Interesting Findings

The first interesting finding we notices is that there is only one instance of two people asking and answering each others questions, meaning that person A asked a question that person B answered, then person B asked a question that person A answered. In graph theory, this is known as a “cycle”, where two nodes in a directed graph have edges going to each other.

We also looked at a histogram of the number of interactions throughout the time window, and in the time series histogram, we noticed that the most interactions happen late at night. This is pretty typical of programmers to be up in the middle of the night bug fixing, but since the timezone of the data is Pacific Standard Time, the more likely option is people around the world interacting with this site during normal business operating hours in their timezone.

Like mentioned above, we theorized that the different programming languages one can receive help about on Stack Overflow were responsible for creating these communities. Since we did not have the data to delve in to this theory, an interesting follow up on this research could be figuring out the language, or general topic, each one of these communities focused on in their interactions

One last interesting finding was that most people have a number of interactions between $[0, 10]$ answers to questions, but the one user in this time frame had more than 70 interactions in the 5 days of data we looked at. This is pretty remarkable seen as how that averages to about one interaction every two hours if this person didn’t sleep at all, or about one interaction every hour if every user they interacted with got 8 hours of sleep each night.

Hardships and Lessons Learned

The first hardship we dealt with when working with this data is that the provided dataset off of the SNAP contained *every* interaction on Stack Overflow, since its conception in August of 2008. As one could imagine, there are quite a few answers to questions. Uncompressed, the text file ended up being 436 MB in size. This was also just the size of the answers to questions data. The union of all three datasets provided, uncompressed, was 1.52 GB in size. Text editors refused to open this file regardless of the machine we tried opening it on. Opening the answers to questions text file in a basic text editor didn’t work out very well either, and loading the file in to Mathematica or Python was also difficult. Besides for not wanting, or needing, to visualize 2.5 million nodes and nearly 18 million edges, this was the other reason we trimmed the data down to only around 1000 entries. The 1000 edges we graphed were more than sufficient to show the patterns in the data we were interested in finding.

Using Python, one of the hardships was to find the right layout to display the data so we can clearly see the different nodes. We experimented with so many layouts including spiral layout display, the random layout, etc but the spring layout ended up being the most appropriate.

In Mathematica, the first hurdle that came about was trying to parse the data as it came in such a big file. The solution to this was to read in the data line-by-line via a read-only file stream, stopping after the first 1,000 rows/graph edges/interactions as the graph was already complex enough. The other notable hurdle was in creating the video for the evolution animation, sadly not shown in this report, but embedded in the presentation. Firstly, parallelization was used to convert the manipulatable graph objects into rasterized sets of pixels that the exporting algorithm could more easily handle. Then, in order to more efficiently render the frames of the animation, codecs from a comparatively extended version of ffmpeg needed to be found, downloaded, and installed to use the HEVC-NVENC GPU accelerated video encoding. Finally, the labels associated with the legend of the "degree graph" were unable to be scaled to reflect the true values of the data; leaving the labels to correspond to the normalized vertex-degree data used for color scaling.

Conclusion

We had theorized that, by analyzing the network data of various users on Stack Overflow, that we would have been able to see emergent phenomena such as cliques. Upon closer analysis using various different algorithms for analyzing network data, we found that these communities were clearly evident and in the case of most users, were easily separable and well defined clusters. An interesting area of study that furthers the work done in this report could be figuring out why exactly these communities form. The working theory we came up with, that unfortunately doesn't have data to back it up currently, is that these communities formed as a result of different programming languages providing their own central point in which people all over the world can ask and receive information about.