Kaden, Laiba, and Ibrahim

Abhishek Viswanathan

INFSCI 0610

13 December 2023

# Final Project Report

For our final project we chose to analyze and create networks for the film *Avengers: Infinity War*. In the film, the character Thanos goes on a mission to collect the 6 Infinity Stones to wipe out half of the universe’s population. In order to stop him, the Avengers and groups of other superheroes from prior Marvel movies team up to take him down. The goal of our project was to look at the character interactions and dynamics in the film and show not only how characters interacted with each other but also to find out and show the importance all of them had to the film’s narrative. This movie is the culmination of all twenty-one prior movies in Marvel’s cinematic universe; it had been building up to this moment, and we wanted to see how all the characters in the film converged. Despite being named as an Avengers movie, Thanos is actually the protagonist and the networks we created also reflect that.

In order to get data for our networks, we first had to come up with what we wanted to represent with our networks. Our initial plan was to make one large graph that highlighted all of the interactions each character had with each other, but we felt as though the scope ended up being too large to capture in one graph. After thinking about possible solutions, we decided to split our data down into two smaller graphs. The first graph would be a weighted undirected graph, with a focus on the narrative elements of the film and the Infinity Stones, while the second graph would be a weighted directed graph that would highlight how the characters interacted with each other during the movie.

After deciding what we wanted to focus on, we then began the data collection process. For the narrative graph, we initially used Word to take down notes on the major and minor interactions of the film. We classified a narrative interaction as one that contributed to the progression of the plot, revealed crucial information about the Infinity Stones, character motivations, or pivotal events that propelled the storyline forward. We decided that the data we wanted to use for the interaction graph would be based on the dialogue in the film, and we would take note of who characters talked to throughout the film. Initially, we wanted to just use the script as a basis for the dialogue, but we found out that scripts only show who is speaking for a particular line and not who it's supposed to be directed to. So, we made another Word doc to list out the information for each line of dialogue, the speaker, and the intended recipients. This process ended up being the longest part by far, and was the most time exhausting task of the project.

Once our note taking was complete, we needed to figure out how to transcribe the notes into data for use in our graphs. We decided to use Excel to create adjacency matrices for both of our graphs and used the data from these matrices as the base for the nodes and edges of each graph. Both matrices had each character that was represented in the graphs as a respective row and a column. For the first graph, everytime we thought a narrative interaction had occurred, we added a value of one to the respective character’s row and column with who the interaction was with. We only added that +1 to either the row or the column, not both because the total between the cross sections of the rows and columns then became the overall edge weight between the two nodes. For the second graph, we also added a value of one to either the respective character’s row and column but we did so based on who was speaking and who was being spoken to. Because the dialogue graph was directed, we had to use a different approach; the rows represented the out degree (who was doing the speaking) and the columns represented the in degree (who was being spoken to).

At first, we planned to use NetworkX in Python to visualize our 2 graphs. About halfway into the data collection process, we had some rough drafts of our narrative and dialogue graphs. They looked decent, but we soon realized that as we continued to add in more nodes and edges, the graphs became congested and hard to read. It might have been due to the fact that our code was being run and executed within a JupyterLab notebook, but no matter how much we tried to space out the nodes and enlarge the size of the graph, it still remained too crowded. We then decided to swap over to Gephi, as it's supposed to be better for larger graphs. With the narrative graph, we didn’t write any code to put in, instead we imported the .csv file straight into Gephi. The dialogue graph still used the code we initially wrote, and after that code was complete, we used the “nx.write\_gexf()” command to export that into a .gexf file that could then be imported into Gephi. Once we had our graphs in Gephi, we then experimented with the different layouts that Gephi offers for visualization. Our biggest hurdle with Gephi’s graph creation was finding a visualization that had enough space and gravity to where each node had space around it, the edges were clear and visible, and the graph remained readable. We experimented with the ForceAtlas2, Expansion, OpenOrd, and Yifan Hu layouts before deciding to use the Fruchterman Reingold layout for both graphs. From there we experimented more with changing the size and opacity of the nodes and edges, before finally coming up with a couple of visualizations that we felt were usable.

While we switched over to Gephi for the visualization part of our project, we still used NetworkX for the analysis of our graphs. We calculated measurements for the degree centrality, betweenness centrality, embeddedness, and the degree assortativity coefficient for both graphs. With our dialogue graph, we also did additional calculations for the nodes with the highest in and out degree, to see who had the most lines and who was spoken to the most often. After running and creating our code, we found that Thanos had the highest degree centrality and betweenness centrality in both graphs. For the narrative graph, he also had the highest embeddedness and other characters such as Thor and Captain America were also high up in the lists of our calculations. For the dialogue graph, we found that Iron Man and Starlord were the 2 characters with the highest in and out degrees which lines up well with their character’s personalities; they are both more humorous characters known for their witty remarks and quick comebacks. Throughout the film, different groups of heroes and villains appear and interact with one another and all these groups are reflected on our graphs by showing up as subnetworks. These subnetworks visually displayed the relationships between characters because each one consists of characters that share similar views and are often allied with each other; Thanos’ followers of the Black Order supported him in every way they could to acquire the Infinity Stones, the undercover Avengers worked together under the radar to combat injustice, the sanctioned Avengers publicly represented the just motivation of protecting the Earth from global and extraterrestrial threats. Characters that connected subnetworks to each other corresponded with the characters that had the highest betweenness centrality measures in both graphs; these characters acted as bridges between subnetworks like Iron Man connecting the Sorcerers and the undercover Avengers with each other.

Overall, the graphs we have created highlight the differing importance that certain characters have to the film. Even though the film was an Avengers’ film, Thanos is the protagonist which disrupts the familiar narrative structure of hero-centric movies where protagonists are good guys and antagonists are bad guys. The entire story revolves around Thanos trying to acquire the 6 Infinity Stones; the superheroes are just the defenders. This explains why Thanos would have the highest degree centrality, largest betweenness centrality, and highest embeddedness. While the overall flow of both graphs was somewhat muddled due to our changes to make it readable, Thanos is very much prominent in both graphs, as a large amount of edges from all of the nodes in both connect to him. Other characters such as Thor, Iron Man, and Captain America also have a high importance to the narrative and events in the film, but ultimately it all centers around Thanos. Another takeaway we got from our graphs are the relationships between all of the characters within the film. During the film, groups such as the Guardians of the Galaxy, Avengers, and individual characters from standalone films such as Spider-Man and Black Panther all converged with one another and got a chance to interact with each other for the first time. Our graphs got to show the dynamics not only within each group, but also how they all interacted with each other, and how the dynamics of these characters evolved past their individual narratives. We did have a couple of limitations with our final graphs that were mainly due to our unfamiliarity with Gephi and our concerns with how the graphs would look when presenting them to the class, but overall we felt as though our final graphs did a good job of maintaining the balance between readability and practicality.

# Appendix

## Contribution Statements:

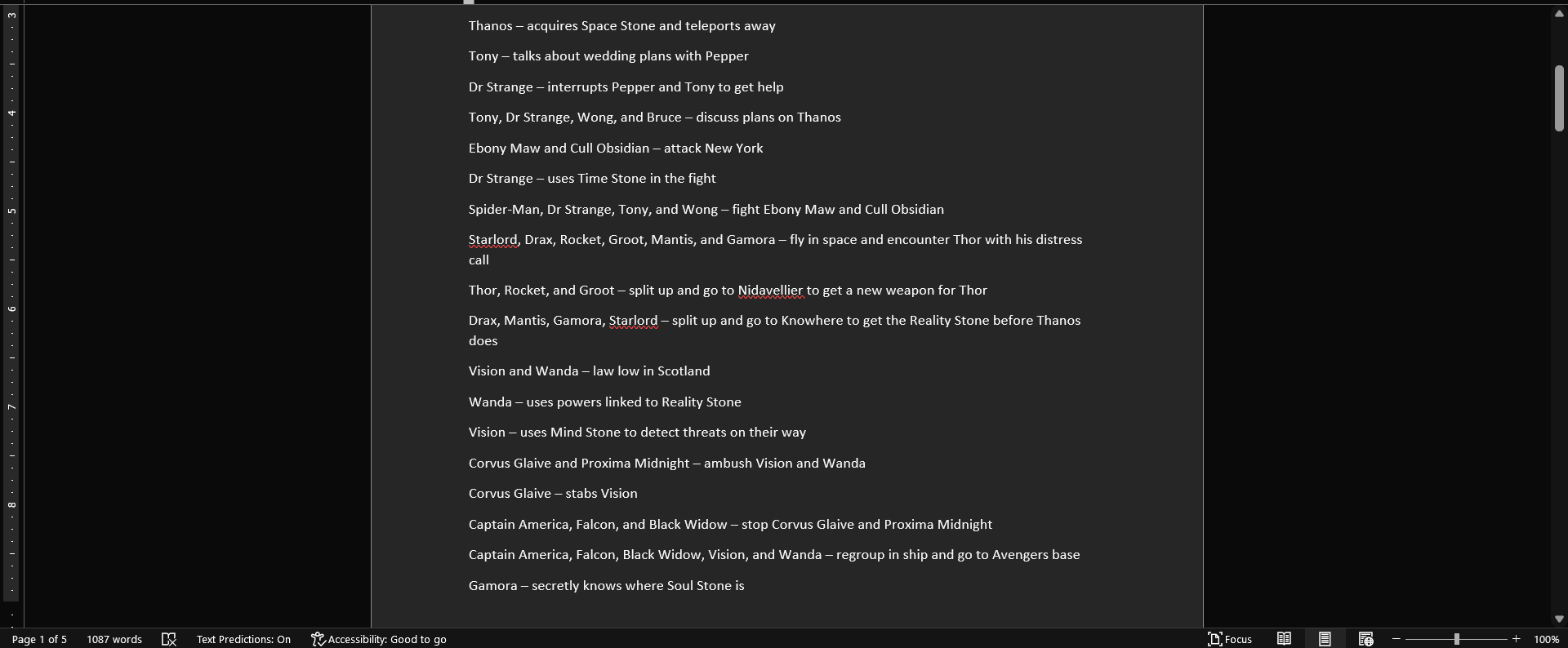
Kaden Jackson – created the initial notes documents for the narrative and dialogue graphs, transcribed the data from the notes documents into Excel, created initial graphs in NetworkX, created the code for the network metrics and analysis, created slideshow presentation, created graphs in Gephi and wrote final report.

Ibrahim Qutyan – collaborated on creating slideshow presentation and collaborated with writing final report.

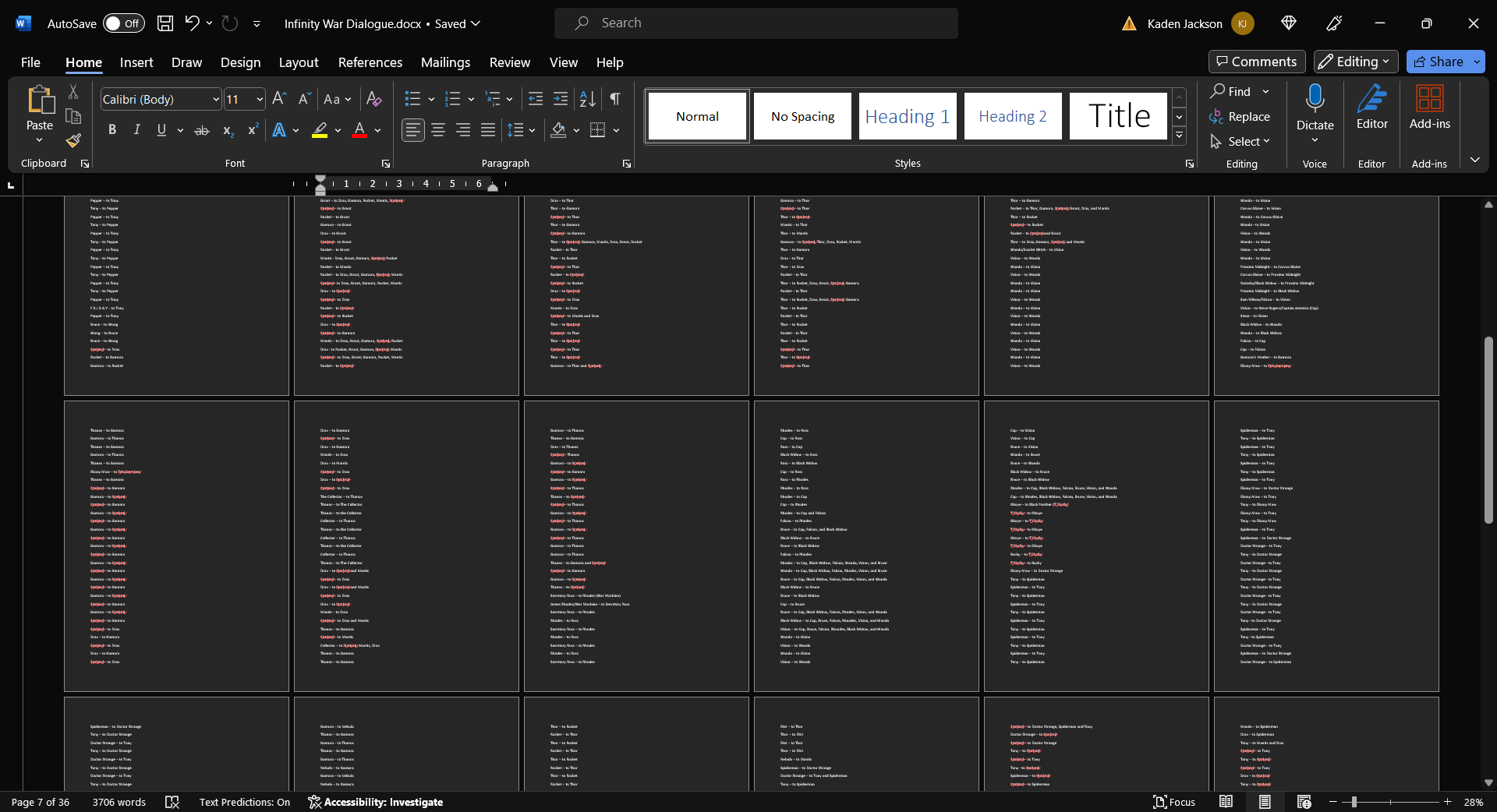
Laiba Awan – collaborated on creating slideshow presentation and collaborated with writing final report.

## Notes (just going to add a singular screenshot of both documents):

***Figure 1:*** *A picture from the document of the narrative notes taken for the 1st graph.*

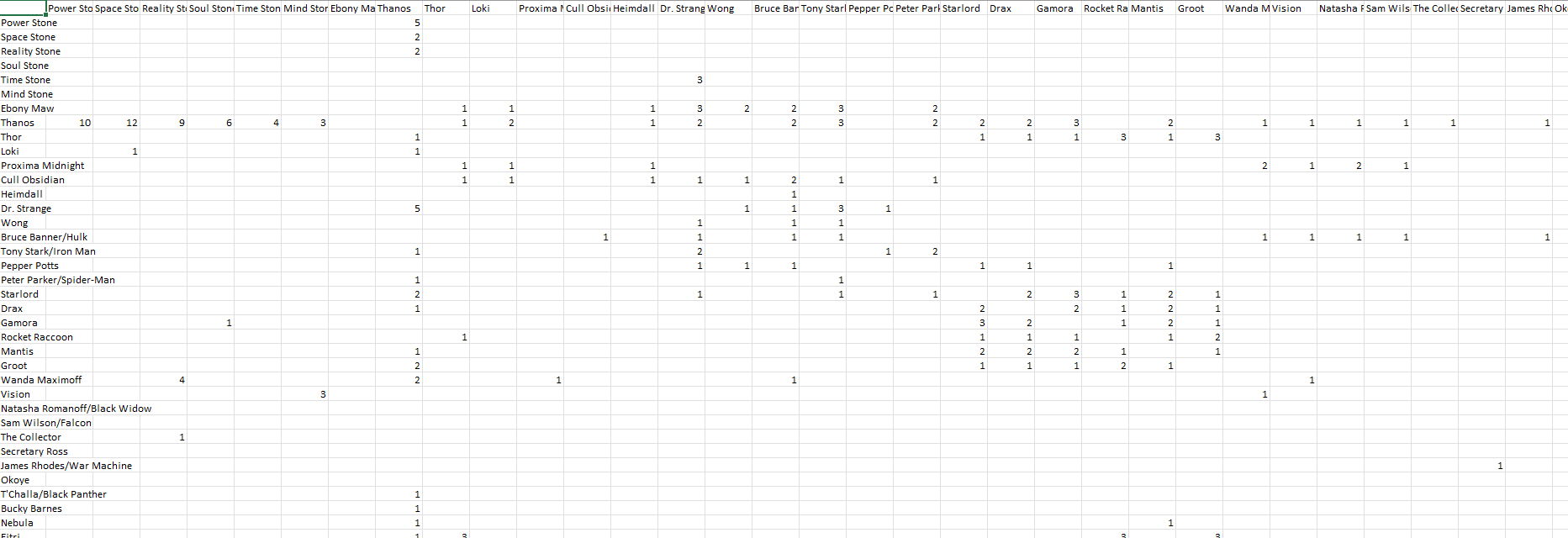
****

***Figure 2:*** *A photo from the 36 page notes document created for the dialogue graph.*

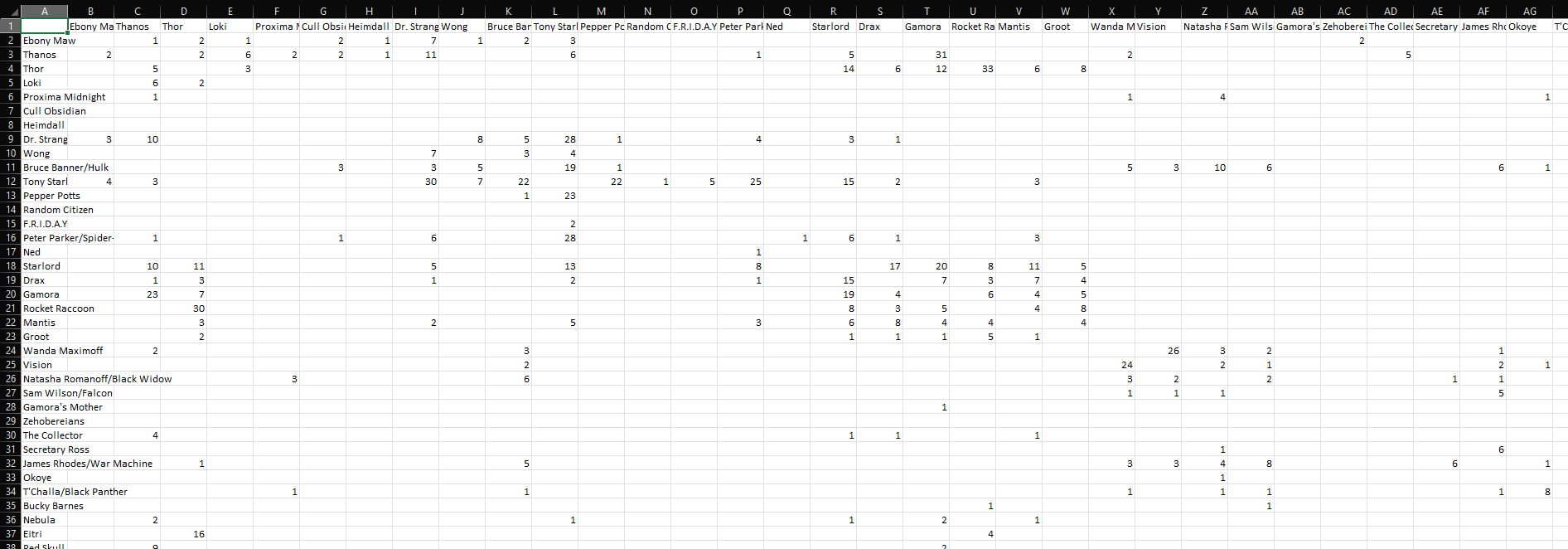
****

## Matrices:

***Figure 3:*** *A snippet of the matrix created from our notes for the narrative graph*

****

***Figure 4:*** *A snippet of the matrix created from our notes for the dialogue graph*

****

## 

## Code:

***Figure 5:*** *The code we initially used for the narrative graph. Once we swapped over to Gephi we instead just imported the .csv file and did away with this code*

|  |
| --- |
| import pandas as pd import networkx as nx import matplotlib.pyplot as plt  file\_path = 'Plot Narrative Spreadsheet.csv'  data = pd.read\_csv(file\_path, index\_col=0)  character\_groups = {  'Black Order': ['Thanos','Ebony Maw', 'Proxima Midnight', 'Cull Obsidian', 'Corvus Glaive'],  'Asgardian Ship Survivors': ['Thor', 'Loki', 'Hulk', 'Heimdall'],  'Sorcerers': ['Wong', 'Dr. Strange'],  'Undercover Avengers': ['Steve Rogers/Captain America', 'Wanda Maximoff', 'Vision', 'Natasha Romanoff/Black Widow', 'Sam Wilson/Falcon', 'Bucky Barnes'],  'Sanctioned Avengers': ['Tony Stark/Iron Man', 'James Rhodes/War Machine'],  'Guardians of the Galaxy': ['Starlord', 'Gamora', 'Drax', 'Rocket Raccoon', 'Groot', 'Mantis', 'Nebula'],  'Wakandans': ['Okoye', "T'Challa/Black Panther", 'Shuri', "M'Baku"],  'Other Characters':['Pepper Potts', 'The Collector', 'Secretary Ross', 'Eitri', 'Red Skull', 'Maria Hill', 'Nick Fury'], }  stone\_colors = {  'Space Stone': 'blue',  'Reality Stone': 'red',  'Power Stone': 'purple',  'Mind Stone': 'yellow',  'Time Stone': 'green',  'Soul Stone': 'orange', } G = nx.Graph()  for group, characters in character\_groups.items():  color = plt.cm.tab20(hash(group) % 20)   for character in characters:  G.add\_node(character, group=group, color=color)  for stone, color in stone\_colors.items():  G.add\_node(stone, group='Infinity Stone', color=color)  for i, source in enumerate(data.index):  for j, target in enumerate(data.columns):  weight = data.iloc[i, j]  if weight > 0:  if source in G.nodes() and target in G.nodes():  if G.nodes[source]['group'] != 'Infinity Stone' and G.nodes[target]['group'] != 'Infinity Stone':  if G.nodes[source]['group'] == G.nodes[target]['group']:  G.add\_edge(source, target, weight=weight \* 2)   else:  G.add\_edge(source, target, weight=weight)  node\_colors = [G.nodes[node]['color'] for node in G.nodes()]  pos = nx.spring\_layout(G) plt.figure(figsize=(12, 12)) nx.draw(  G,  pos,  with\_labels=True,  node\_size=500,  node\_color=node\_colors,  font\_weight='bold',  font\_color='black',  edge\_color='gray',  width=2,  ) plt.title('Character and Infinity Stone Interaction Network') plt.show() |

***Figure 6:*** *The code we used for the character dialogue graph*

|  |
| --- |
| import pandas as pd import networkx as nx import matplotlib.pyplot as plt import matplotlib.colors as mcolors  file\_path = 'Character Dialogue Spreadsheet.csv' data = pd.read\_csv(file\_path, index\_col=0) G = nx.DiGraph() G.add\_nodes\_from(data.index)  for i, source in enumerate(data.index):  for j, target in enumerate(data.columns):  weight = data.iloc[i, j]  if weight > 0:   G.add\_edge(source, target, weight=weight)  node\_colors = list(mcolors.CSS4\_COLORS.keys())[:46]   pos = nx.spring\_layout(G)  plt.figure(figsize=(12, 12))  nx.draw(  G,  pos,  with\_labels=True,  node\_size=2000,  node\_color=node\_colors,  font\_weight='bold',  arrowsize=15, ) edge\_labels = nx.get\_edge\_attributes(G, 'weight') nx.draw\_networkx\_edge\_labels(G, pos, edge\_labels=edge\_labels) plt.title('Character Dialogue Network') plt.show() nx.write\_gexf(G, 'dialogue\_graph.gexf') |

***Figure 7:*** *The updated code we used for the network metrics of the narrative graph. Results are in the image directly below (might be hard to read)*

|  |
| --- |
| import pandas as pd import networkx as nx import matplotlib.pyplot as plt  data = pd.read\_csv('Plot Narrative Spreadsheet.csv', index\_col=0)  character\_groups = {  'Black Order': ['Thanos', 'Ebony Maw', 'Proxima Midnight', 'Cull Obsidian', 'Corvus Glaive'],  'Asgardian Ship Survivors': ['Thor', 'Loki', 'Hulk', 'Heimdall'],  'Sorcerers': ['Wong', 'Dr. Strange'],  'Undercover Avengers': ['Steve Rogers/Captain America', 'Wanda Maximoff', 'Vision', 'Natasha Romanoff/Black Widow', 'Sam Wilson/Falcon', 'Bucky Barnes'],  'Sanctioned Avengers': ['Tony Stark/Iron Man', 'James Rhodes/War Machine'],  'Guardians of the Galaxy': ['Starlord', 'Gamora', 'Drax', 'Rocket Raccoon', 'Groot', 'Mantis', 'Nebula'],  'Wakandans': ['Okoye', "T'Challa/Black Panther", 'Shuri'],  'Other Characters': ['Pepper Potts', 'The Collector', 'Secretary Ross', 'Eitri', 'Red Skull', 'Maria Hill', 'Nick Fury'],  'Infinity Stones': ['Space Stone', 'Reality Stone', 'Power Stone', 'Mind Stone', 'Time Stone', 'Soul Stone'], }  G = nx.Graph()  for group, characters in character\_groups.items():  for character in characters:  G.add\_node(character, group=group)  for i, source in enumerate(data.index):  for j, target in enumerate(data.columns):  weight = data.iloc[i, j]  if weight > 0:  if source in G.nodes() and target in G.nodes():  if G.nodes[source]['group'] != 'Infinity Stones' and G.nodes[target]['group'] != 'Infinity Stones':  if G.nodes[source]['group'] == G.nodes[target]['group']:  G.add\_edge(source, target, weight=weight \* 2)  else:  G.add\_edge(source, target, weight=weight)  pos = nx.spring\_layout(G) plt.figure(figsize=(12, 12))  nx.draw(  G,  pos,  with\_labels=True,  node\_size=500,  font\_weight='bold',  font\_color='black',  edge\_color='gray',  width=2, )  plt.title('Character and Infinity Stone Interaction Network')  degree\_centrality = nx.degree\_centrality(G) betweenness\_centrality = nx.betweenness\_centrality(G) clustering\_coefficient = nx.clustering(G) embeddedness = {edge: len(list(nx.common\_neighbors(G, edge[0], edge[1]))) for edge in G.edges()} degree\_assortativity = nx.degree\_assortativity\_coefficient(G)  sorted\_degree\_centrality = sorted(degree\_centrality.items(), key=lambda x: x[1], reverse=True) sorted\_betweenness\_centrality = sorted(betweenness\_centrality.items(), key=lambda x: x[1], reverse=True) sorted\_clustering\_coefficient = sorted(clustering\_coefficient.items(), key=lambda x: x[1], reverse=True) sorted\_embeddedness = sorted(embeddedness.items(), key=lambda x: x[1], reverse=True)  print("Degree Centrality (from largest to smallest):") print(sorted\_degree\_centrality) print("\nBetweenness Centrality (from largest to smallest):") print(sorted\_betweenness\_centrality) print("\nClustering Coefficient (from largest to smallest):") print(sorted\_clustering\_coefficient) print("\nEmbeddedness (from largest to smallest):") print(sorted\_embeddedness) print("\nDegree Assortativity Coefficient:") print(degree\_assortativity) |

## 

## *Figure 8: The code we used for the network metrics of our dialogue graph. Results are in the image directly below (might be hard to read)*

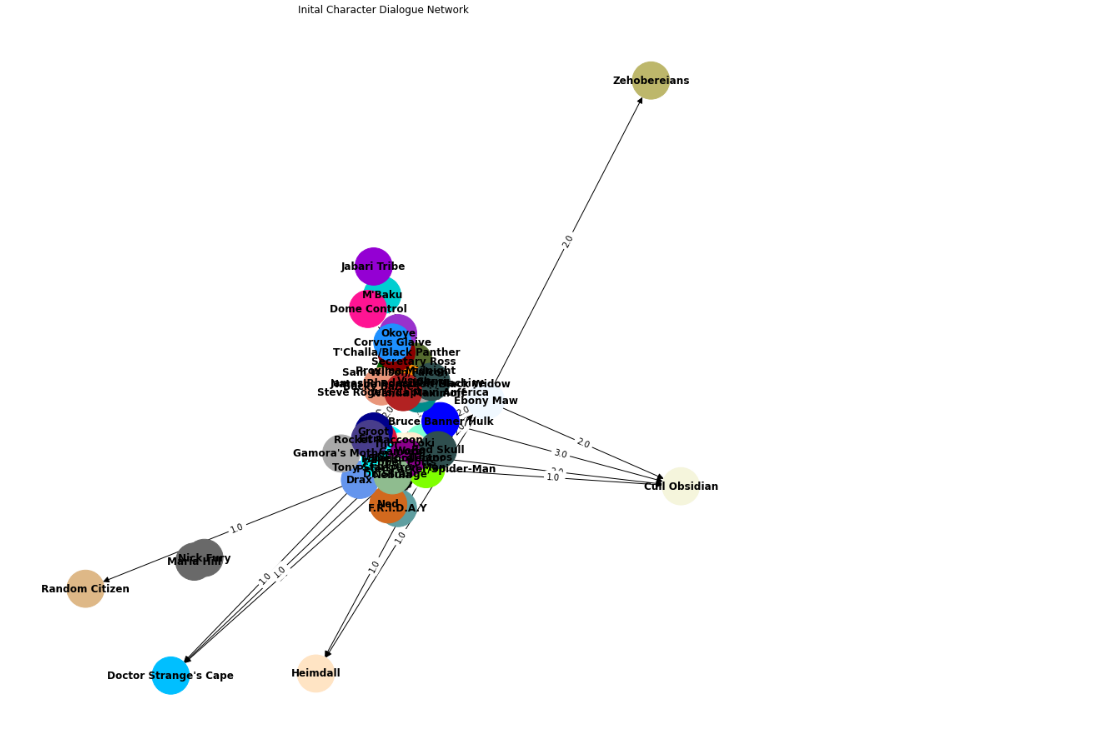
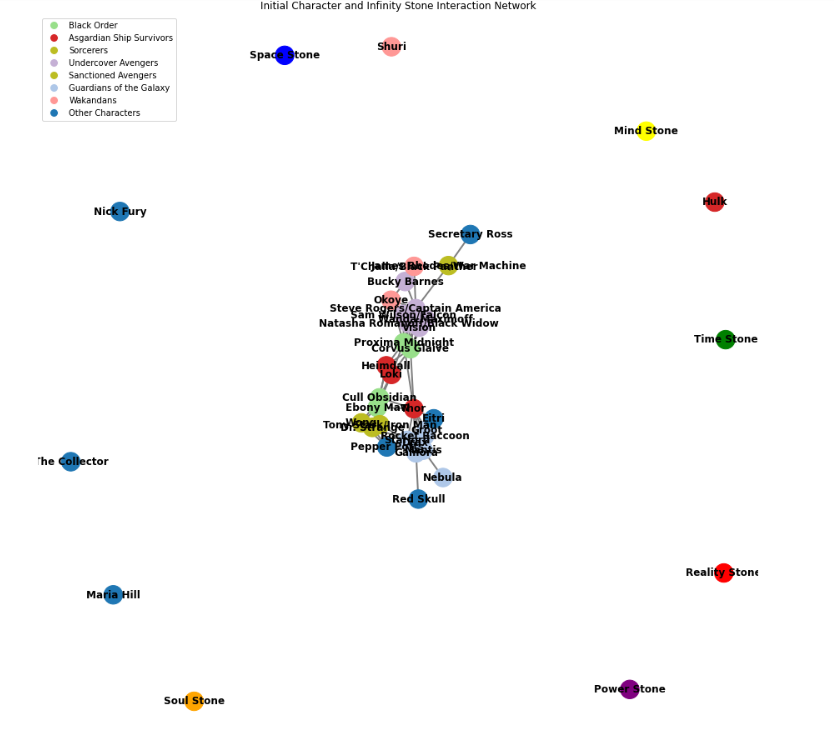
|  |
| --- |
| import pandas as pd import networkx as nx  file\_path = 'Character Dialogue Spreadsheet.csv' data = pd.read\_csv(file\_path, index\_col=0)  G = nx.DiGraph() G.add\_nodes\_from(data.index)  for i, source in enumerate(data.index):  for j, target in enumerate(data.columns):  weight = data.iloc[i, j]  if weight > 0:  G.add\_edge(source, target, weight=weight)   in\_degree = G.in\_degree(weight='weight') out\_degree = G.out\_degree(weight='weight') degree\_assortativity = nx.degree\_assortativity\_coefficient(G) betweenness\_centrality = nx.betweenness\_centrality(G, weight='weight') degree\_centrality = nx.degree\_centrality(G)   embeddedness = {} for u, v in G.edges():  preds\_u = set(G.predecessors(u))  preds\_v = set(G.predecessors(v))  common\_preds = preds\_u.intersection(preds\_v)  embeddedness[(u, v)] = len(common\_preds)   sorted\_in\_degree = sorted(in\_degree, key=lambda x: x[1], reverse=True) sorted\_out\_degree = sorted(out\_degree, key=lambda x: x[1], reverse=True) sorted\_embeddedness = sorted(embeddedness.items(), key=lambda x: x[1], reverse=True) sorted\_betweenness\_centrality = sorted(betweenness\_centrality.items(), key=lambda x: x[1], reverse=True) sorted\_degree\_centrality = sorted(degree\_centrality.items(), key=lambda x: x[1], reverse=True)   print("\nIn-Degree (from largest to smallest):") print(sorted\_in\_degree) print("\nOut-Degree (from largest to smallest):") print(sorted\_out\_degree) print("\nDegree Assortativity Coefficient:") print(degree\_assortativity) print("\nEmbeddedness (from largest to smallest):") print(sorted\_embeddedness) print("\nBetweenness Centrality (from largest to smallest):") print(sorted\_betweenness\_centrality) print("\nDegree Centrality (from largest to smallest):") print(sorted\_degree\_centrality) |



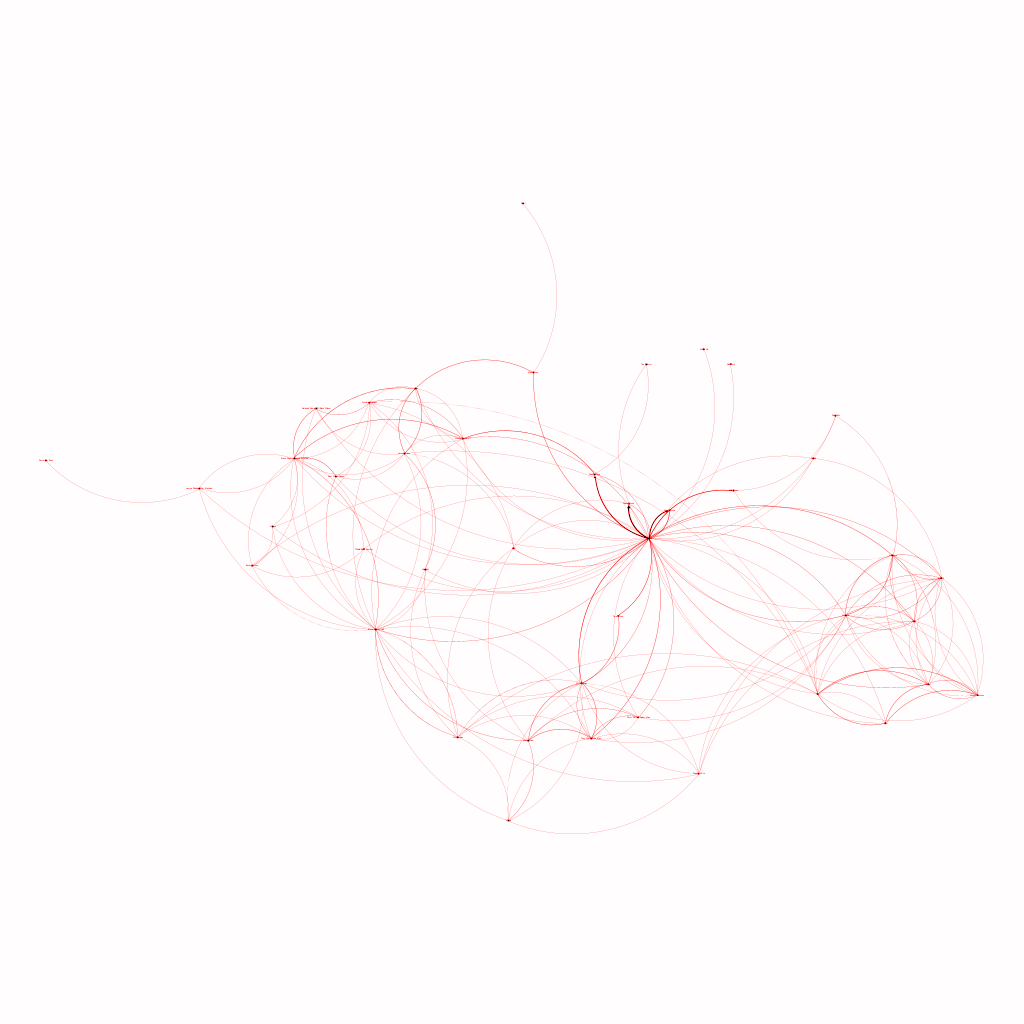
## 

## Graphs:

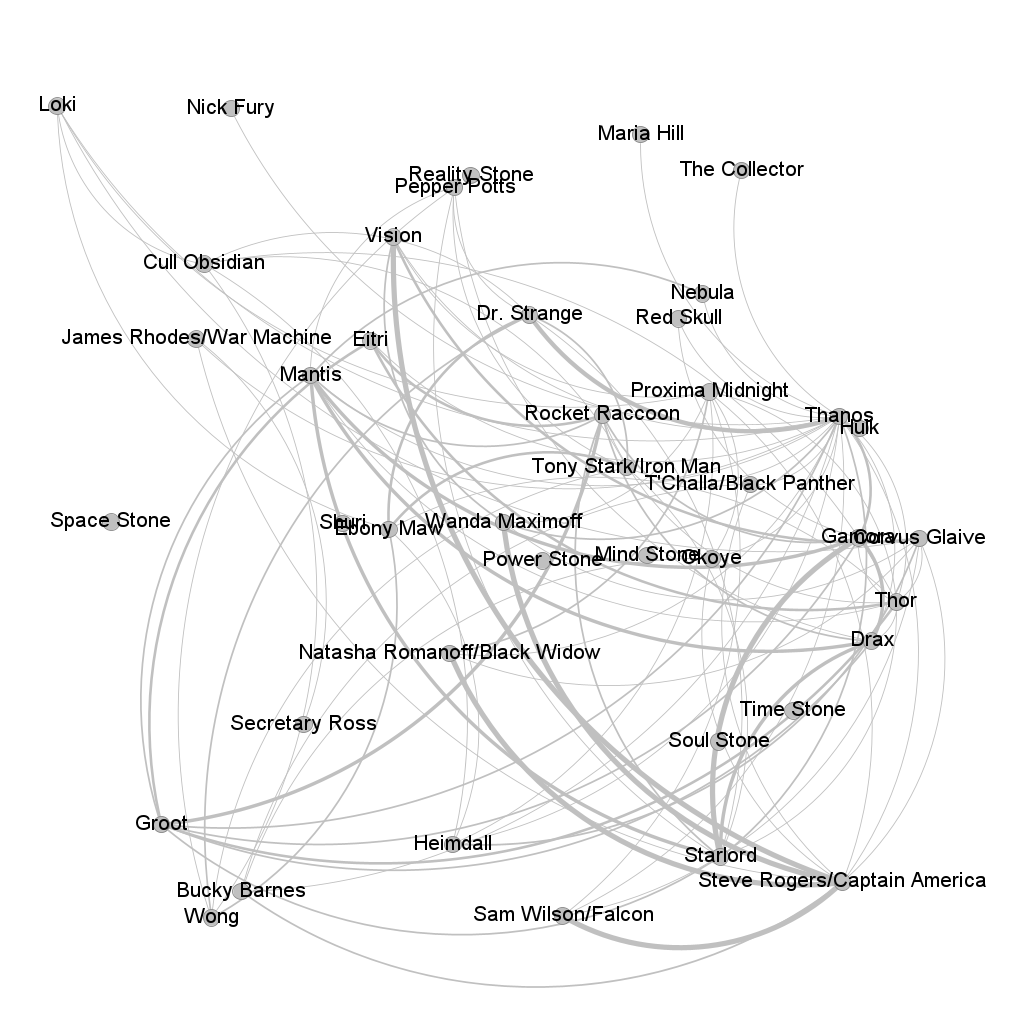
***Figures 9 and 10:*** *The initial rough draft for the narrative and dialogue network. Even though we hadn’t fully finished the graphs yet, it became clear that both of these graphs had a scope that was too big to visualize in NetworkX without readability issues.*

**

***Figure 11:*** *One of our initial Gephi graphs for the narrative graph. While it was quite spacious, there wasn’t a good way to have this be presented and readable to the class. This is an example of the types of problems we ran into when trying to come up with the graphs from Gephi.*

****

***Figure 12:*** *The Final Narrative Graph*



***Figure 13:*** *The final Dialogue Graph*

