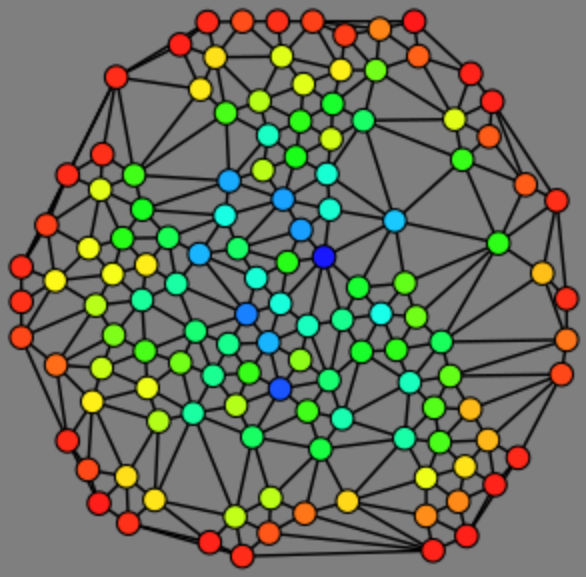
ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS



# Social Network Analysis

Γεωργίου Αλέξιος-Λάζαρος (3180027)

A card's game competitive deck network

# Table of Contents

# Introduction:

## What is Yu-Gi-Oh! card game?

It's a card game between 2 players, there are a lot of rules in the game that makes it complicated, and every card has a different effect and type. Very briefly, you can win by make your opponent lose all his life points or wait for him not to run out of cards from his deck.
The rules and the card types are not important for the purpose of this document, so you don't have to know the game to understand the goal and the outcome of this network analysis.
What we need to know for our case?
The main thing we must understand is that the game contains a big pool of more than 11,000 different unique cards and each player must choose a deck of cards consisted of 40-60 and 0-15 for his extra deck. So, in total he should have 40-75 cards in his deck from the pool of cards the game offers.
Finding a good deck that maximizes your chances of winning is very important part in every card game. In our case, due to the game's cards pool expanding in addition with a banning list (by KONAMI) at regular intervals, people are trying every day to find the best deck to win in tournaments and the game is trying control the balance between decks by editing the banning lists or adding new cards.
We also need to know the idea of a "meta" deck. Meta decks are usually popular decks that tend to be powerful in between updates of the game (pool or banning list). People tend to play mostly meta-decks competitively due to a lot of reasons; they probably can't think of a better deck or by default the deck is very powerful, so everyone is playing the same small number of decks with small changes or choosing a deck that is very good on countering those decks (an anti-meta deck).
The game itself got more popular recently with the release of a video-game "Yu-Gi-Oh! Master Duel" at 18-Jan-2022. I used the last 1 month's (release date to 18-Feb-2022) meta decks to create a network of cards.


## What is our Network consisted of?:

Our Nodes are the cards, we only use the Name and the ID of the card.
Our Edges are the relationship between the cards, if a card is in the same deck on a popular meta deck then they are related, this relationship isn't weighted with multiple relationships on various decks and it's an undirected relationship.

## Dataset description:

Nodes.csv (Two columns)
ID: The ID of the card, each card with the same name has the same unique number.
Can be found in the bottom left of the card, is a number.
(Of course, there are multiple copies of the same card but the id between these cards is the same)
Name: The name of the Yu-Gi-Oh! card. Can be seen in the top of the example card "Time Wizard"

Edges.csv (Two columns)
Source: The source card node
Target: The target card node
The relationship is explained above.

# Purpose of the analysis:

This can be used for various purposes.
1) We can find out which cards are important for the network. Some cards tend to be used in a lot of decks since they are more generally purposed. KONAMI can see if affecting a card's effect or adding it to the banning list/ limiting list, how will this affect the network of meta decks. Will it ruin the system of meta-decks and affect a lot of players? Maybe we want to find a small number of cards that will dismantle all the meta decks.
2) We can find which cards are close to each other, in the game some cards are from the same "family" of cards, usually they are released together, and they even have a similar name and can only be used with each other as combos. This could be a useful to deck suggestions to players on a deck builder application.
3) We can expand the analysis by containing more information on the Nodes, cards also include two types and an attribute. We can find patterns on the connection between the cards with the same attribute (Homophily) or see how the network structure looks like with colored communities.

# How I collected the data?

I made some Python scripts.

## Fetching the Nodes:

getNodes.py

```
import json
import pandas as pd

#File data is found on a GET request in the below link
#Provided by API of db.ygoprodeck.com (Fetched @February 2022)
#Simply download the data using a browser and the link (data must be placed in same directory)
#https://db.ygoprodeck.com/api/v7/cardinfo.php


#Read File of all yugioh cards data
f = open("cardinfo.php", "r")
stringfile = f.read()
f.close()

#String to JSON
jsonfile = json.loads(stringfile)


data = []

#Appending each card's data into data list
for carddata in jsonfile['data']:
    cardId = carddata['id']
    cardName = carddata['name']
    data.append([cardId, cardName])


#Making a pandas dataframe using the data
df = pd.DataFrame(data, columns = ['ID', 'Name'])

#Save file into csv
df.to_csv('Nodes.csv', index = False)
```

Getting the Nodes was quite simple after small research, it turns out there is an API that provides all the card's information. I made a GET request and filtered the id and the name into a csv file.

```
#https://db.ygoprodeck.com/api/v7/cardinfo.php
```

Following the above link, we can GET the cardinfo.php which is a json formatted file with all the game's information.

## Fetching the edges:

I made a complicated script using automated libraries (Selenium and Chrome webdriver).
I couldn't use traditional data scraping since the page didn't have the information on GET request, the information is added dynamically probably from a hidden API into the pages, so the HTML page isn't fully loaded on GET, I had to use Selenium to make the page load the decks dynamically.

```
#https://www.masterduelmeta.com/top-decks#dateRange=Last%204%20weeks&tournamentsOnly&page=1
```

Following the above link, you can find the master duel meta decks between 18-Jan and 18-Feb.
Each deck can be clicked and there is a .ydk file inside which is basically a .txt file with the Ids of the cards in the deck.
After downloading all the deck files from every page, I ended up with around 284 meta decks.

Now we create the edges by iterating between the decks. For every deck we add the corresponding edges, for every card in the deck we add the edge with the other cards of the deck if we haven't added that edge already. (getEdges.py)

```
import os
import pandas as pd


#Creating edges by checking meta deck files


decks = []
path = "decks/"
#Reading the deck files
for file_name in os.listdir(path):
    file_extension = file_name.split(".",1)[1]
    if file_extension == "ydk":
        try:
            with open(path + file_name, encoding="utf8") as file:
                filelines = file.readlines()
            file.close()


            current_deck = []
            for line in filelines:
                line = line.strip()
                if line.isnumeric():
                    current_deck.append(line)

            decks.append(current_deck)

        except:
            print("There was an error on ydk file: {}".format(file_name))

print("Reading files ended..")

#Adding all edges between cards
#If a card is in the same deck with other card there should be connected
data = []
count = 0

for deck in decks:
    count += 1
    for card1 in deck:
        for card2 in deck:
            #Cards should not connect with itself
            #We also don't have to add same edges multiple times
            if card1 != card2:
                if [card1, card2] not in data and [card2, card1] not in data:
            #Checking if the edge is already part of the data is increasing the algorithm's complexity
                    data.append([card1, card2])
    print("Finished: Deck {}".format(count))



print("Edges:", len(data))

#Making a pandas dataframe using the data
df = pd.DataFrame(data, columns = ['Source', 'Target'])
```

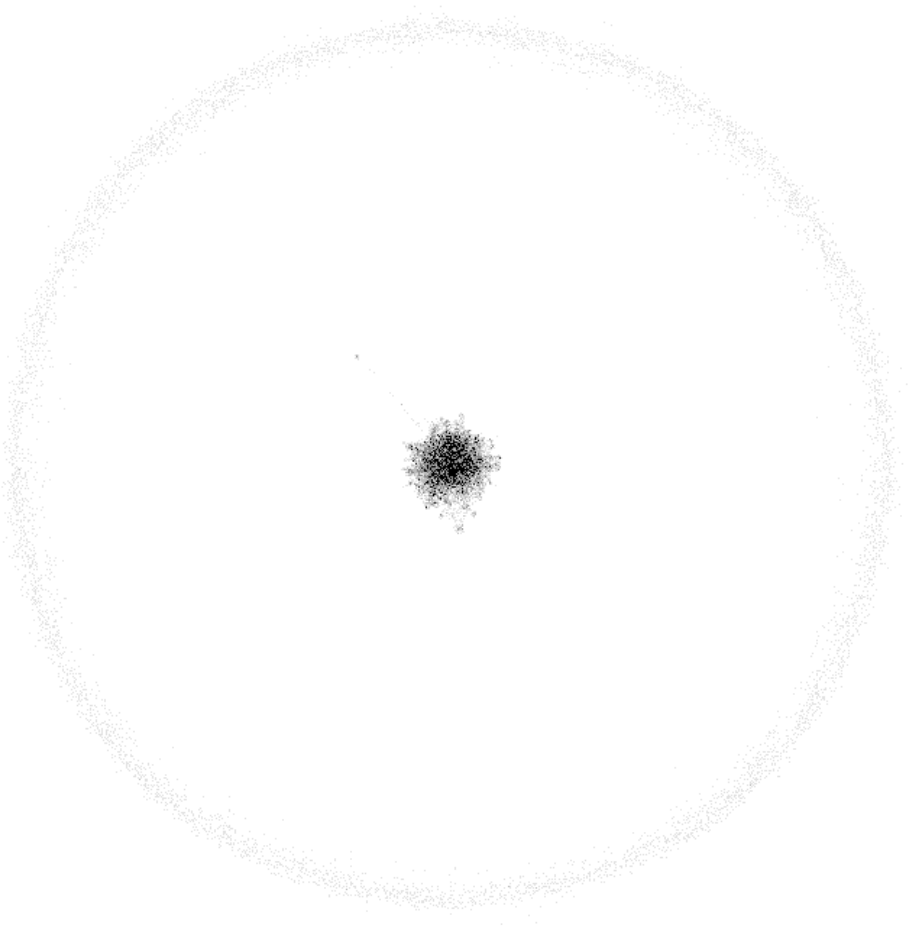
```
#Save file into csv
df.to_csv('Edges2.csv', index = False)
```

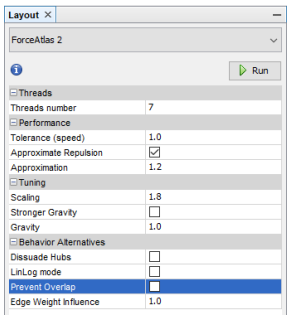
## Gephi:

Gephi is a very strong tool for Network Analysis, and it is used through all the representations, measures of this document. Now that the data is ready, we can import them in Gephi and analyze the network.

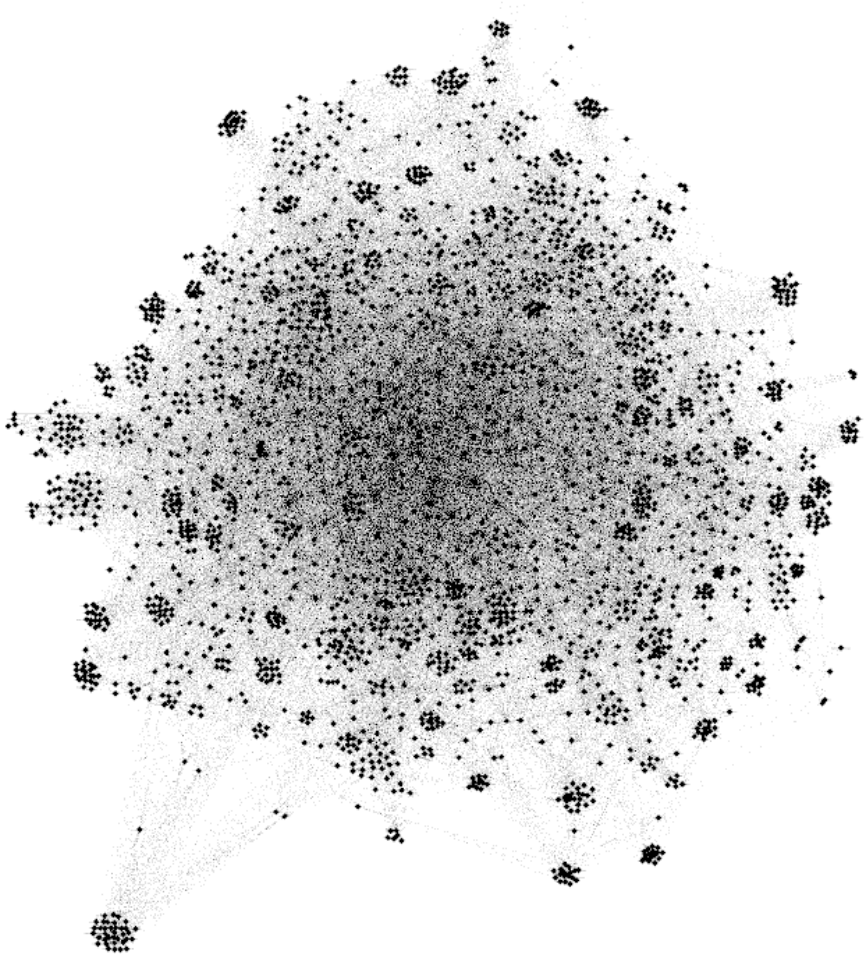
# Network Analysis:

## Graphical Representation (a first glance):





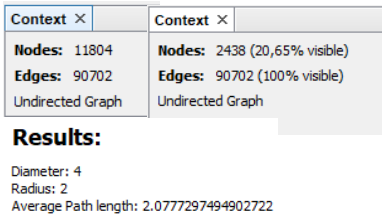

By running a layout algorithm (ForceAtlas2) with default settings, we can see there are a lot of “useless” cards in the pool. These cards are usually old and simply worse that have a very rare use on decks. Competitive decks tend to include newer cards in general. Also, these decks cannot be in a large number on any moment so even with ~280 decks we still cannot include every unique card on the pool.
We can filter out the 0-degree nodes and focus on the inner graph, which seems quite dense compared to other social traditional networks. From now we will ignore these nodes on every statistic on our analysis like they were not part of the data.



This is the actual network (excluding the 0-degree nodes), we will improve the visualization based on more metrics later.

## Basic topological properties:

Nodes: 11804 (2438 with 1 or greater degree)
Edges: 90702




Network diameter: 4
Radius: 2
Average path length: 2.0777
Density: 0,031

Network's diameter is 4 nodes which is a small size comparing to number of Nodes we have (2438). There are a lot of edges in the graph and that makes the longest shortest path between two connected nodes quite small. There are also nodes with a very high degree in the graph which helps a lot on making most of paths smaller.

The graph is very dense (small number of nodes compared to edges), this is due to the nature of the relationship I defined.
Every card/node must have in theory at least 12 connected cards since it's a part of a deck.

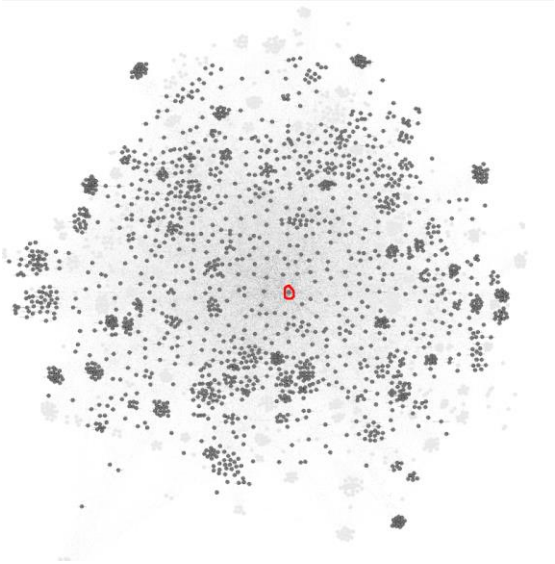
*Decks can only have the same copy of a card 3 times so the minimum unique number of cards we can meet on a deck is* $\left\lfloor \frac{minimum\ cards\ on\ a\ deck\ -2\ extra\ copies\ of\ the\ card}{3\ copies\ for\ each\ other\ card} \right\rfloor = \left\lfloor \frac{40-2}{3} \right\rfloor = 12$

In practice, we have some very popular cards that are very powerful for various reasons and can be used in many decks.

| name | Degree |
|---|---|
| Ash Blossom & Joyous Spring | 1690 |
| Maxx "C" | 1540 |
| Called by the Grave | 1365 |
| Monster Reborn | 1273 |
| Accesscode Talker | 1124 |
| Lightning Storm | 1082 |
| Infinite Impermanence | 952 |
| Knightmare Phoenix | 938 |
| Divine Arsenal AA-ZEUS - Sky Thu... | 934 |
| Raigeki | 924 |
| Crystron Halqifibrax | 897 |
| Nibiru, the Primal Being | 897 |
| Knightmare Unicorn | 883 |
| Harpie's Feather Duster | 760 |
| Forbidden Droplet | 708 |

*They often have a powerful general-purpose effect in the game or have a new mechanic introduced which is more powerful like hand trap cards like Ash Blossom and Maxx "C".*
*You can think of these cards like wild cards on a traditional game of cards.*

The average path length is quite smaller than the diameter for the above reasons.



Gray nodes relate to Ash Blossom, the most degree card in the game, compared to transparent nodes that the card is not related to. More than half of useful cards have been seen in the same deck with this card.

## Component measures:

Number of connected components: 1

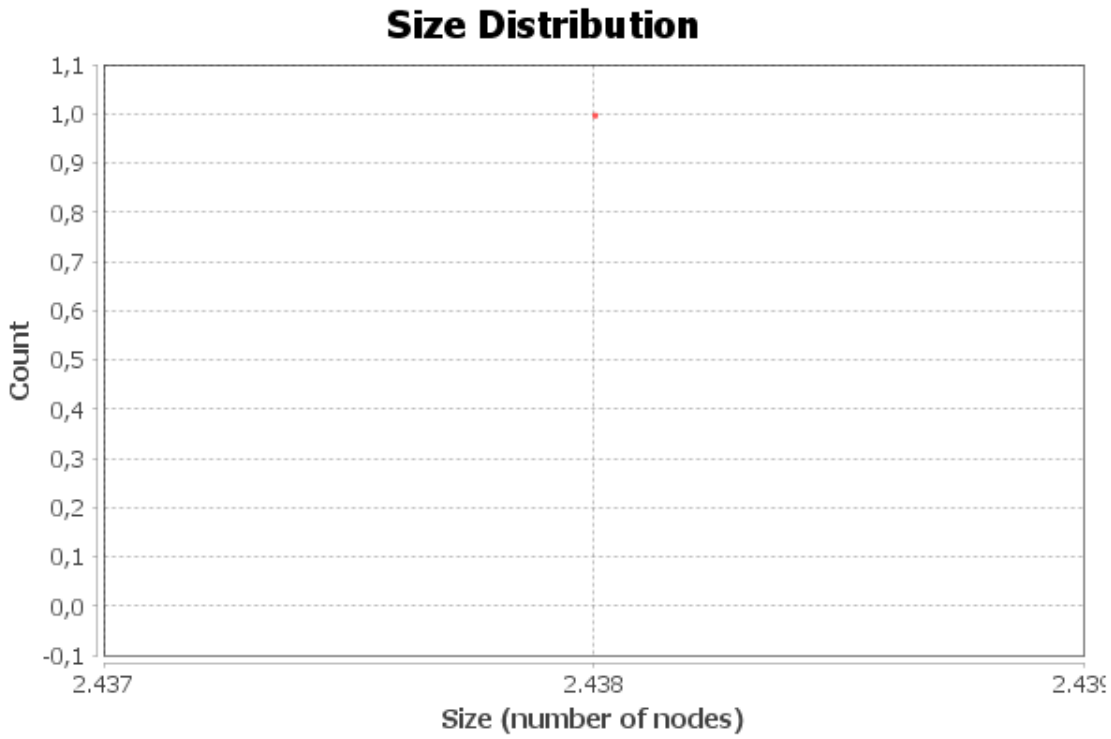
# Connected Components Report

## Parameters:

Network Interpretation: undirected

## Results:

Number of Weakly Connected Components: 1




As expected we only have one connected component consisted of all the nodes on our undirected graph. Our graph is quite dense considering the nodes-edges ratio that makes every node connected with at least one path.

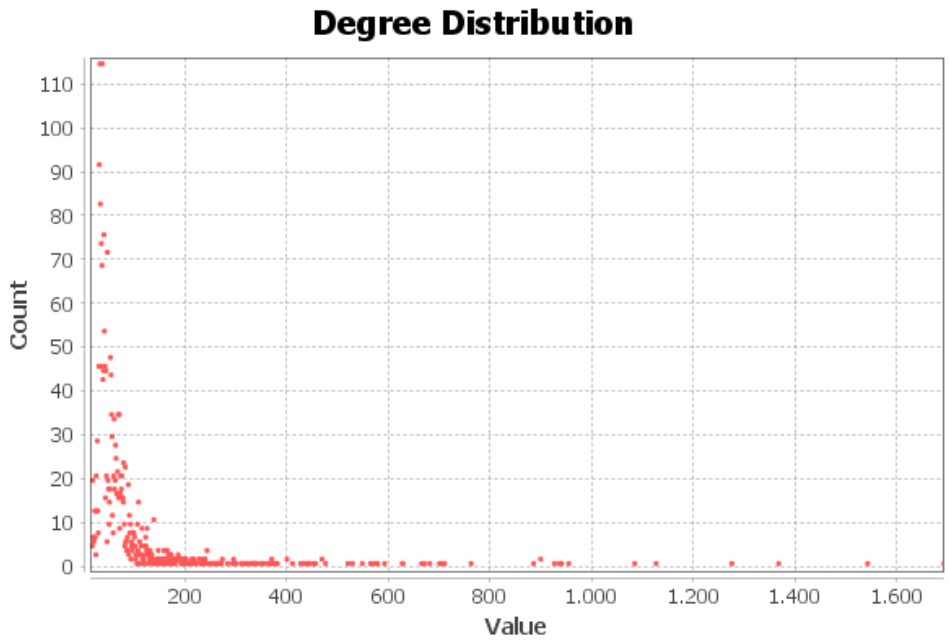
## Degree measures:

Every node has a degree, which is the number of nodes he is connected to. Since we have an undirected graph, we do not have indegree or outdegrees since there is no direction on edges. We also do not have weights on edges, so the average degree is simply the sum of all the nodes' degrees divided by the number of nodes (remember we filtered out the 0 degrees nodes).

# Degree Report

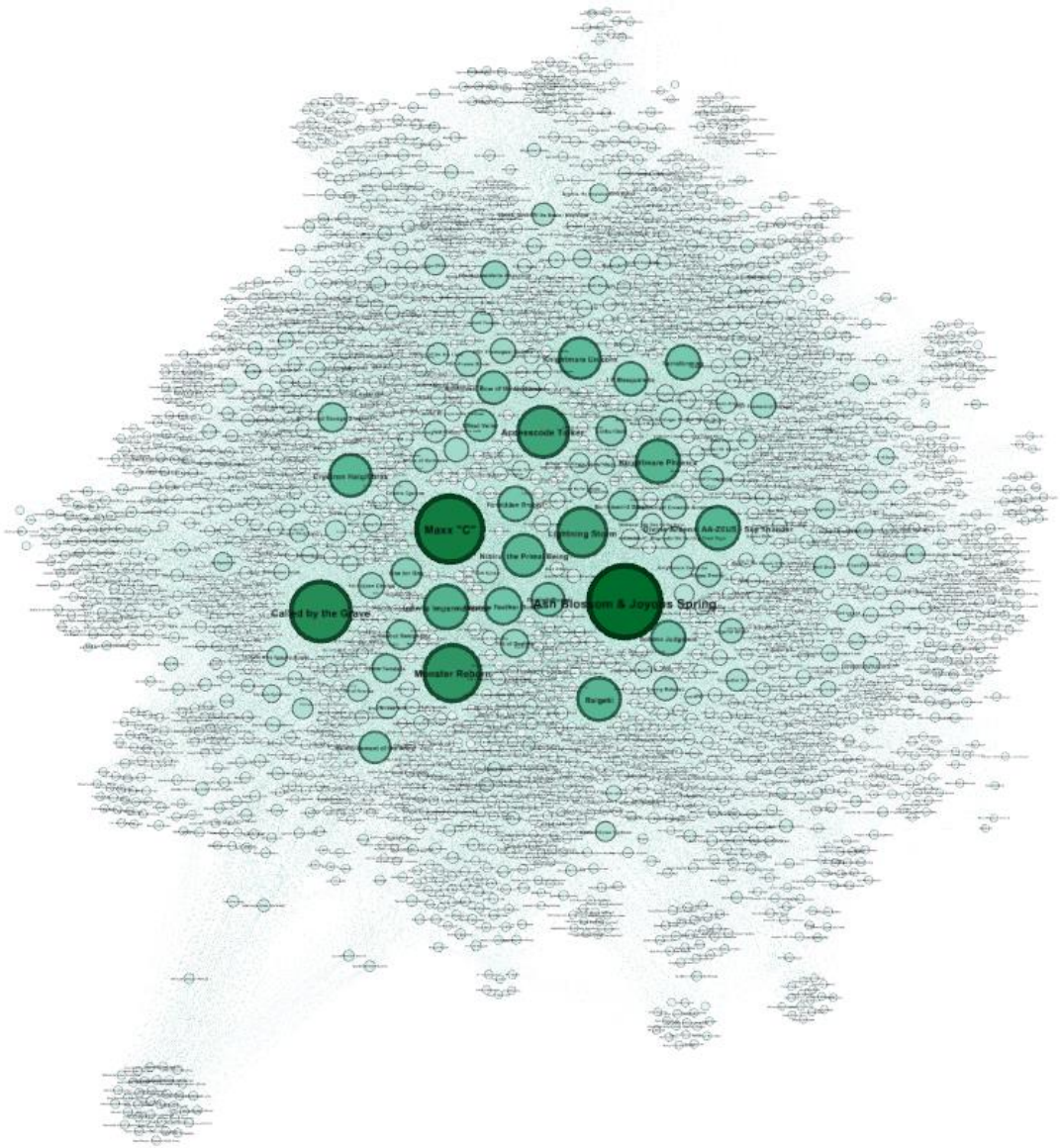
### Results:

Average Degree: 74,407




| name | Degree |
|---|---|
| Ash Blossom & Joyous Spring | 1690 |
| Maxx "C" | 1540 |
| Called by the Grave | 1365 |
| Monster Reborn | 1273 |
| Accesscode Talker | 1124 |
| Lightning Storm | 1082 |
| Infinite Impermanence | 952 |
| Knightmare Phoenix | 938 |
| Divine Arsenal AA-ZEUS - Sky Thu... | 934 |
| Raigeki | 924 |
| Crystron Halqifibrax | 897 |
| Nibiru, the Primal Being | 897 |
| Knightmare Unicorn | 883 |
| Harpie's Feather Duster | 760 |
| Forbidden Droplet | 708 |
| Terraforming | 702 |
| Solemn Judgment | 698 |
| I:P Masquerena | 679 |
| Foolish Burial | 668 |
| Apollousa, Bow of the Goddess | 663 |
| Reinforcement of the Army | 626 |
| Effect Veiler | 625 |
| One for One | 590 |
| Borrelsword Dragon | 576 |
| Linkuriboh | 575 |
| Crossout Designator | 567 |
| Borreload Savage Dragon | 562 |
| Twin Twisters | 546 |
| Pot of Desires | 527 |
| Predaplant Verte Anaconda | 518 |
| Forbidden Chalice | 474 |
| PSY-Frame Driver | 467 |
| PSY-Framegear Gamma | 467 |
| PSY-Framelord Omega | 454 |
| Abyss Dweller | 450 |
| Mekk-Knight Crusadia Avramax | 442 |
| | 440 |
| Allure of Darkness | 439 |
| Imperial Order | 435 |
| Pot of Prosperity | 430 |
| Evenly Matched | 425 |

Average degree is ~74. Maximum degree is 1690.
There are a lot of nodes with a degree around 35. (Top left red dot)
There are also a lot of nodes with a degree more than 200.
We can see some popular cards on the graph like Ash-Blossom with 1690 at the bottom right side of the graph.
The degree of a card is a good metric of the synergy the card has between all the cards of the game.




A visualization of degree analysis, node size and label size has been adjusted to degree.
Also, color of node has been adjusted to its degree, small degree nodes are whiter, higher degree nodes are greener.
ForceAtlas2 was readjusted to prevent overlaps between nodes. We can see the most important nodes of the graph.

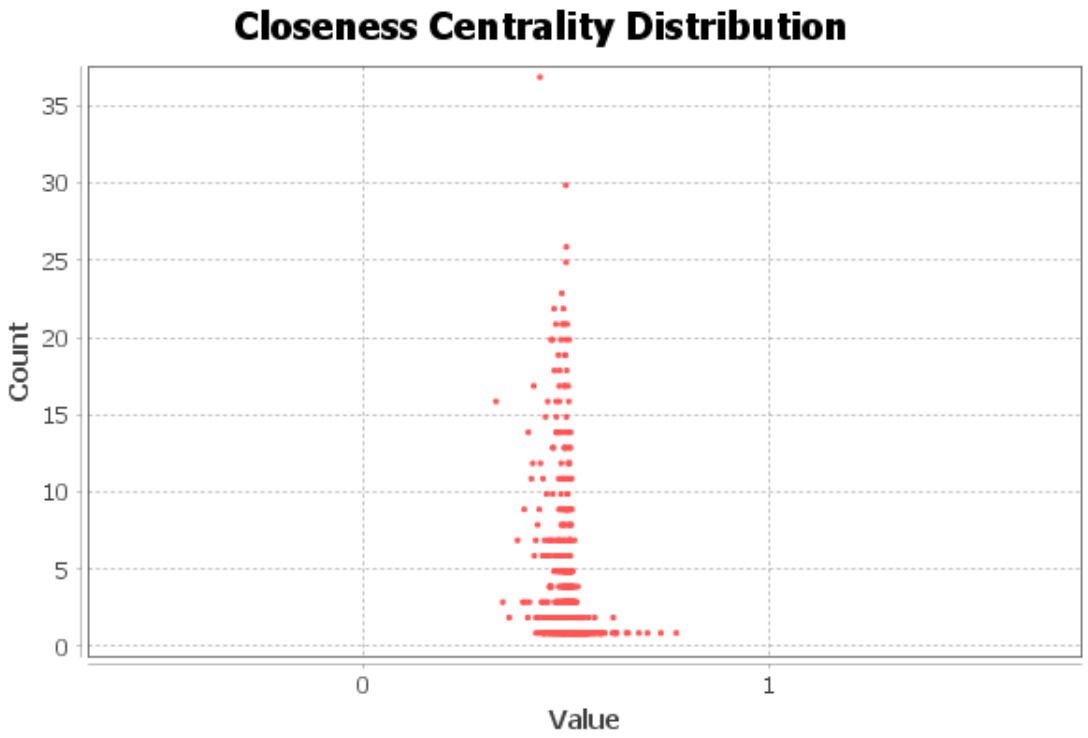
## Centrality measures:

### Degree centrality:

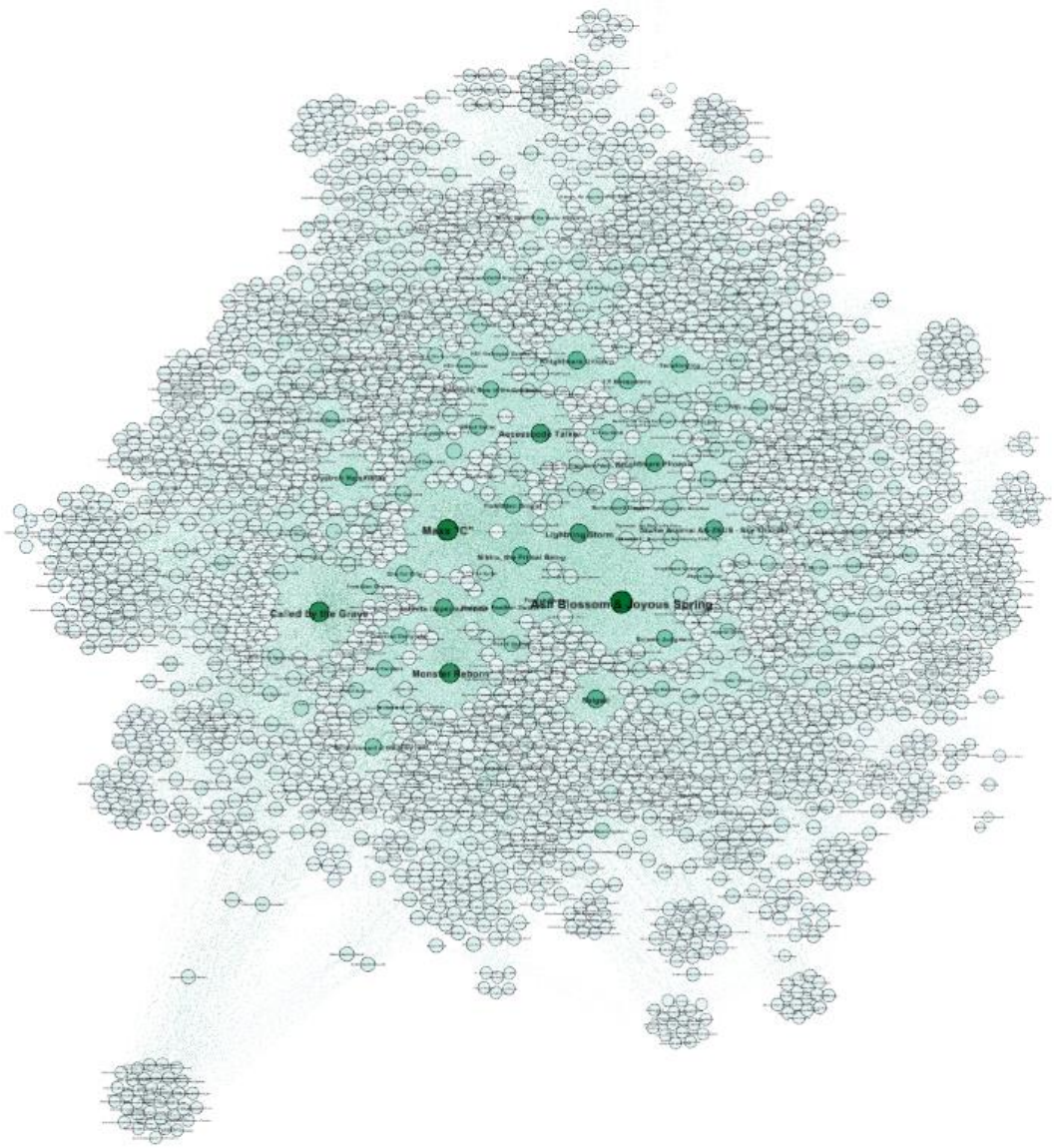
Degree centrality is based on the degree of its node.
The above graph shows the centrality ratios of each node with the color, white nodes are not that centered and green nodes are well centered.

### Closeness centrality:

This is a metric for each node, it indicates how much the node is part of the center of the network. It is calculated via the sum of the length of the shortest path of the node for each other node in the graph.



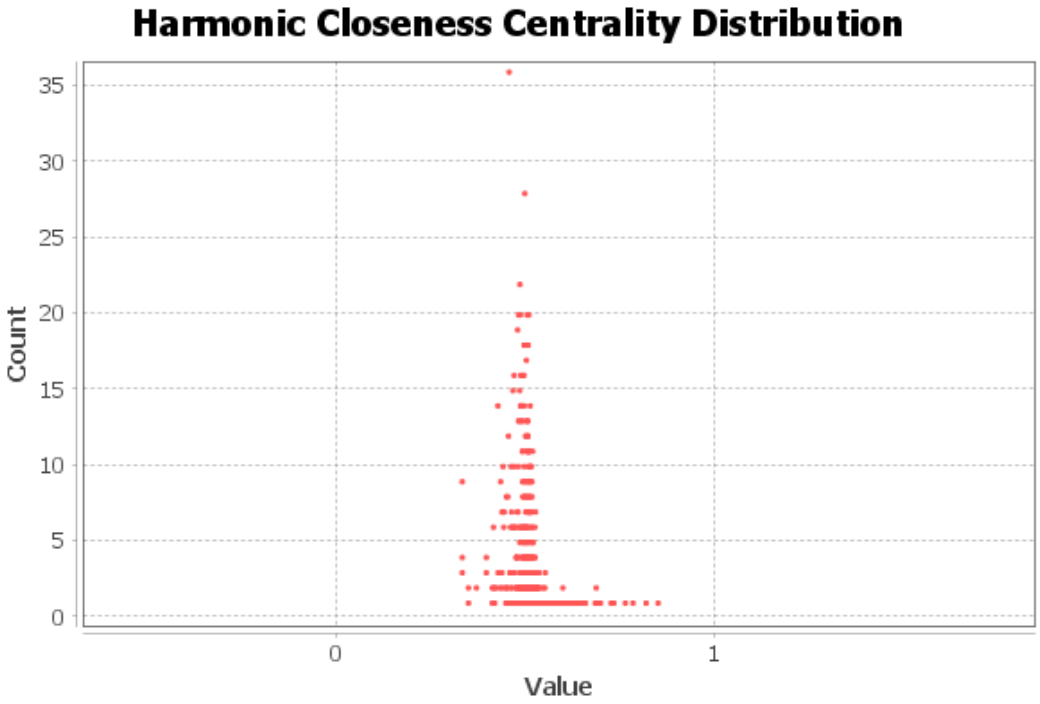

All the nodes have a value of closeness centrality, the smaller node is 0.32 and highest 0.84. Most nodes have a value of 0.5 which is an indication of the density of the network.
*All the nodes are referring to the non-0-degree nodes.*



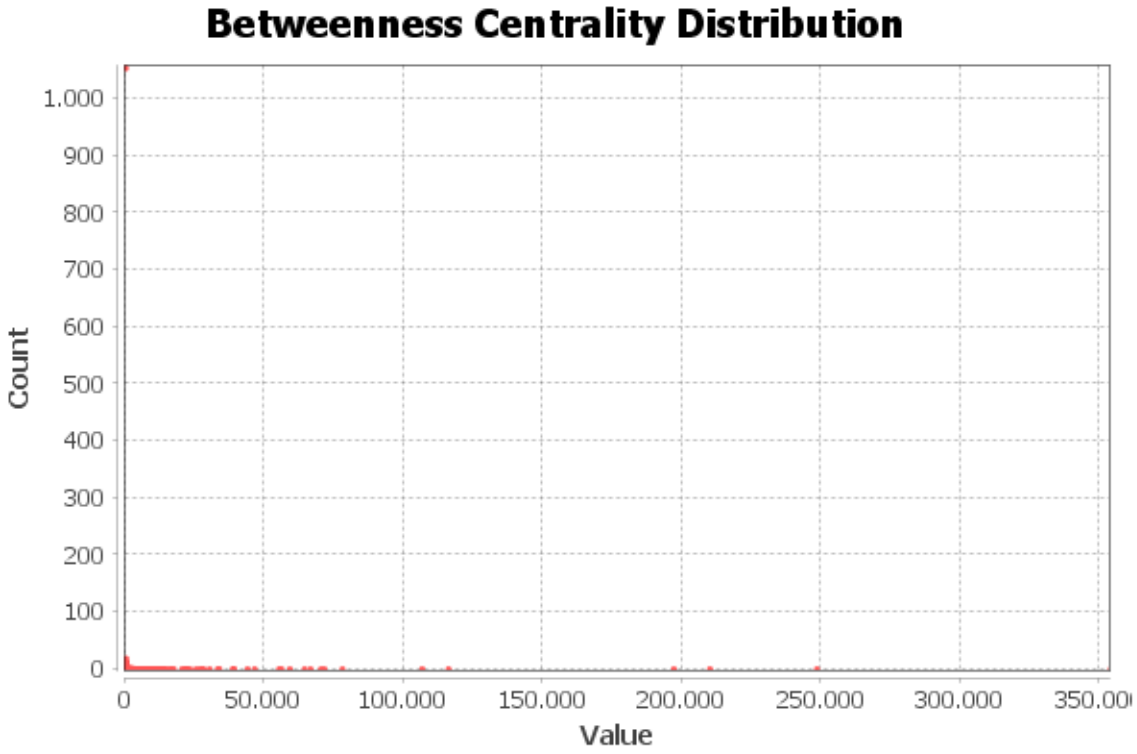

Node colors are “Ranking” of their degree.
Node sizes are based on the closeness centrality.
We can see that all the nodes have a quite large size. The network is quite dense, the average path is quite small, and all the nodes are well centered.
This is reasonable because every card in the game can affect all the other decks and gameplay. So, in a sense even if two decks do not use a card, if the game makes a change on a card then the players will not use the same deck on a tournament.

## Harmonic Closeness centrality:

HCC is a variation of Closeness centrality; the metric distribution of each node is quite like the closeness centrality with the exception that the mean is a little higher (each node has a higher harmonic than closeness value). Our graph is connected so there is not much value in this metric.



## Betweenness centrality:

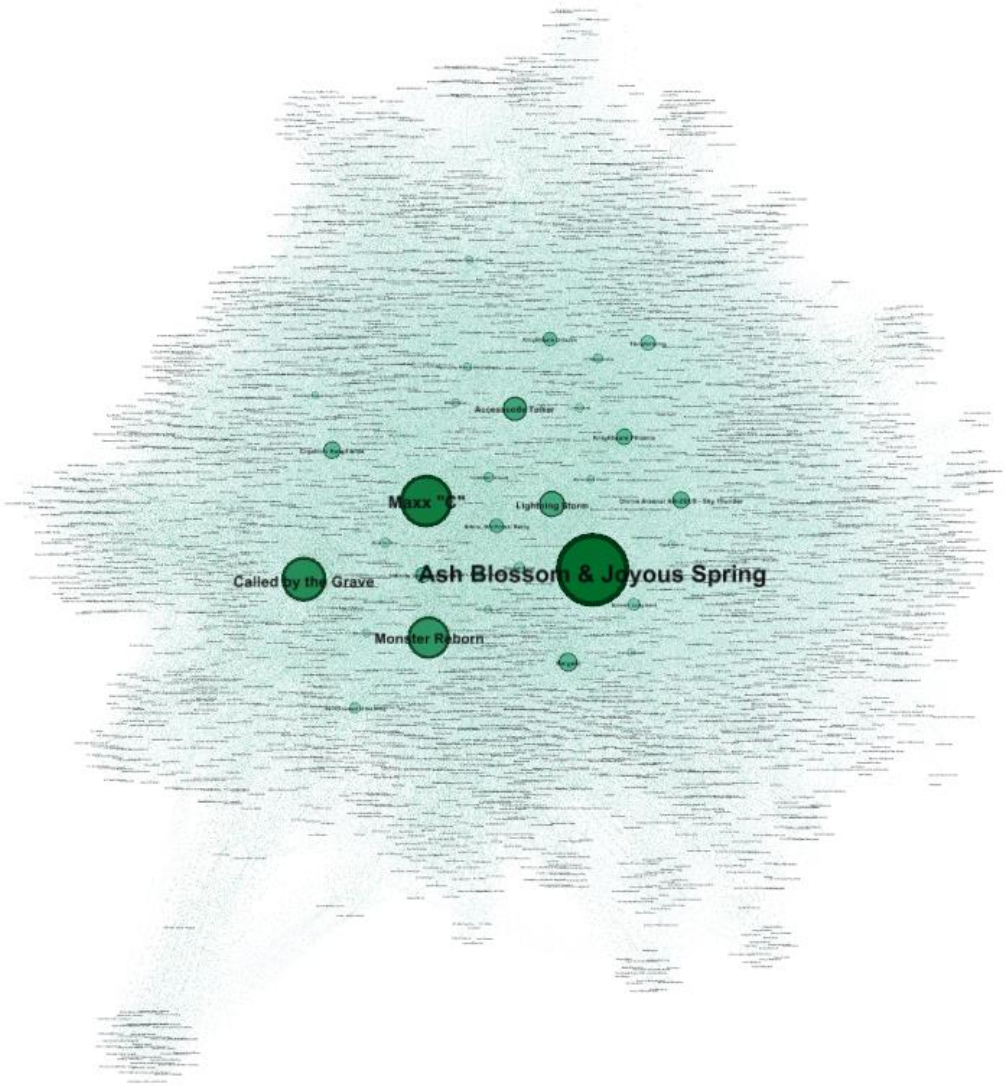



Metrics of the most centered nodes by betweenness centrality

| Id | Label | Interval | name | Degree | Weighted Deg... | Eccentricity | Closeness Centr... | Harmonic Closeness Cent... | Betweenness Centra... | Authority | Hub | Modularity Class | PageRank | Component ID | Clustering Coeffic... | Number of triangles | Eigenvector Centr... |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 14558127 | Ash Blossom & Joyous Spring | | Ash Blossom ... | 1690 | 1690.0 | 2.0 | 0.765389 | 0.846738 | 353750.547297 | 0.163469 | 0.163457 | 0 | 0.008442 | 0 | 0.045946 | 65575 | 1.0 |
| 23434538 | Maxx "C" | | Maxx "C" | 1540 | 1540.0 | 3.0 | 0.727463 | 0.814868 | 248246.893499 | 0.155243 | 0.155241 | 4 | 0.007596 | 0 | 0.050759 | 60151 | 0.945951 |
| 24224830 | Called by the Grave | | Called by the... | 1365 | 1365.0 | 2.0 | 0.6945 | 0.780057 | 209795.94466 | 0.144695 | 0.144693 | 2 | 0.006647 | 0 | 0.059073 | 54993 | 0.878087 |
| 83764719 | Monster Reborn | | Monster Reb... | 1273 | 1273.0 | 3.0 | 0.673204 | 0.759882 | 196840.920406 | 0.131506 | 0.131504 | 1 | 0.006325 | 0 | 0.058747 | 47563 | 0.802247 |
| 86066372 | Accesscode Talker | | Accesscode ... | 1124 | 1124.0 | 3.0 | 0.647106 | 0.729517 | 106447.37711 | 0.129374 | 0.129373 | 6 | 0.005429 | 0 | 0.072534 | 45778 | 0.776878 |
| 14532163 | Lightning Storm | | Lightning Storm | 1082 | 1082.0 | 2.0 | 0.642669 | 0.721994 | 115899.007672 | 0.122825 | 0.122824 | 11 | 0.005304 | 0 | 0.072448 | 42369 | 0.742685 |
| 10045474 | Infinite Impermanence | | Infinite Impe... | 952 | 952.0 | 3.0 | 0.618842 | 0.694228 | 64255.283034 | 0.117555 | 0.117554 | 5 | 0.004519 | 0 | 0.08588 | 38876 | 0.703365 |
| 2857636 | Knightmare Phoenix | | Knightmare P... | 938 | 938.0 | 3.0 | 0.61665 | 0.691355 | 66352.976546 | 0.117156 | 0.117155 | 0 | 0.00448 | 0 | 0.087365 | 38393 | 0.698267 |
| 90448279 | Divine Arsenal AA-ZEUS - Sky Thunder | | Divine Arsen... | 934 | 934.0 | 3.0 | 0.616026 | 0.690535 | 70131.093154 | 0.113574 | 0.113573 | 7 | 0.004494 | 0 | 0.08661 | 37737 | 0.679678 |
| 12580477 | Raigeki | | Raigeki | 924 | 924.0 | 3.0 | 0.614473 | 0.688483 | 77810.775567 | 0.107808 | 0.107807 | 13 | 0.004518 | 0 | 0.082059 | 34992 | 0.651077 |
| 50588353 | Crystron Halqiflbrax | | Crystron Hal... | 897 | 897.0 | 3.0 | 0.610318 | 0.682944 | 71305.839367 | 0.105535 | 0.105534 | 4 | 0.004356 | 0 | 0.086817 | 34888 | 0.638435 |
| 27204311 | Nibiru, the Primal Being | | Nibiru, the Pr... | 897 | 897.0 | 3.0 | 0.610318 | 0.682944 | 55910.523696 | 0.112262 | 0.112261 | 2 | 0.004302 | 0 | 0.088698 | 35644 | 0.669891 |
| 38342335 | Knightmare Unicorn | | Knightmare ... | 883 | 883.0 | 3.0 | 0.608186 | 0.680071 | 55059.210999 | 0.115799 | 0.115798 | 1 | 0.004147 | 0 | 0.097264 | 37875 | 0.688027 |
| 18144507 | Harpie's Feather Duster | | Harpie's Feat... | 760 | 760.0 | 3.0 | 0.589644 | 0.65463 | 43737.204977 | 0.098924 | 0.098923 | 9 | 0.00363 | 0 | 0.102361 | 29523 | 0.590763 |
| 24299458 | Forbidden Droplet | | Forbidden Dr... | 708 | 708.0 | 2.0 | 0.584974 | 0.645261 | 33637.734892 | 0.097869 | 0.097868 | 7 | 0.003304 | 0 | 0.115763 | 28973 | 0.580586 |
| 73628505 | Terraforming | | Terraforming | 702 | 702.0 | 3.0 | 0.581901 | 0.642935 | 38921.109193 | 0.088665 | 0.088665 | 0 | 0.003486 | 0 | 0.106344 | 26166 | 0.532958 |
| 41420027 | Solemn Judgment | | Solemn Judg... | 698 | 698.0 | 3.0 | 0.58093 | 0.641909 | 46257.256348 | 0.08976 | 0.08976 | 0 | 0.003419 | 0 | 0.10892 | 26495 | 0.539072 |

Notice that a card with only 38 degree is in the top of the metrics. Each metric can change the order of the “who is most centered node” by a lot of positions, so we need to understand what each metric counts and how this is related to each network.

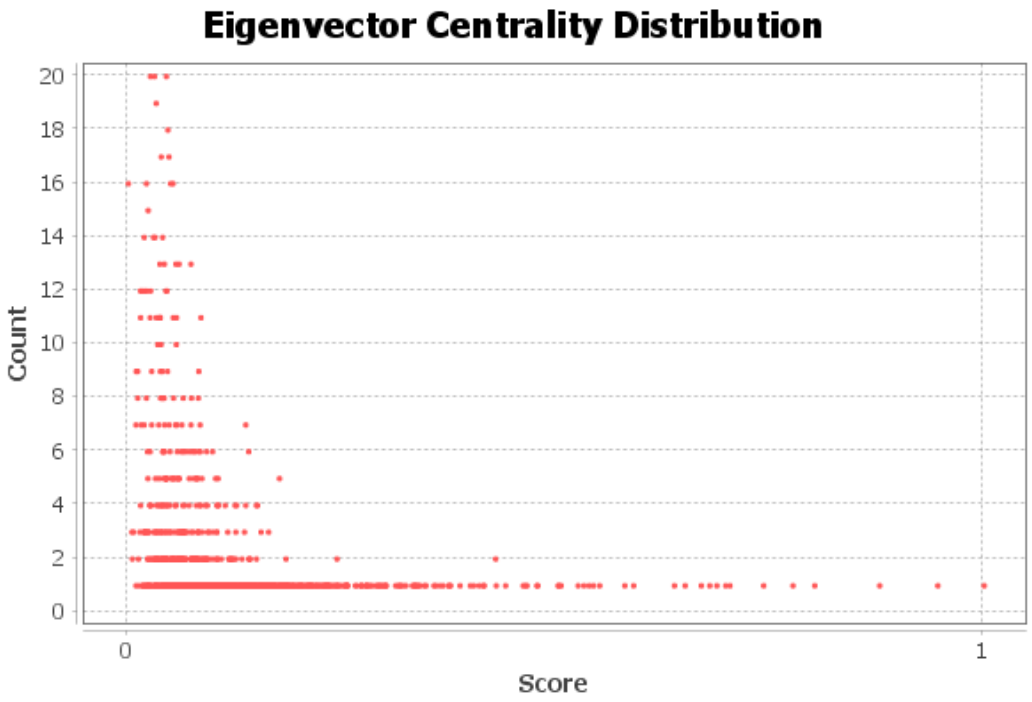
Betweenness centrality is a metric that is based on how many time the node acts like a bridge to a shortest path between two other nodes. It’s a centrality metric.
Below we see a graph that the size of the nodes and labels is corresponding to the betweenness centrality metric.


Maxx "C"
Lightning Storm
Called by the Grave
Ash Blossom & Joyous Spring
Monster Reborn

## Eigenvector centrality:

Network Interpretation: undirected
Number of iterations: 100
Sum change: 0.0590935462135583

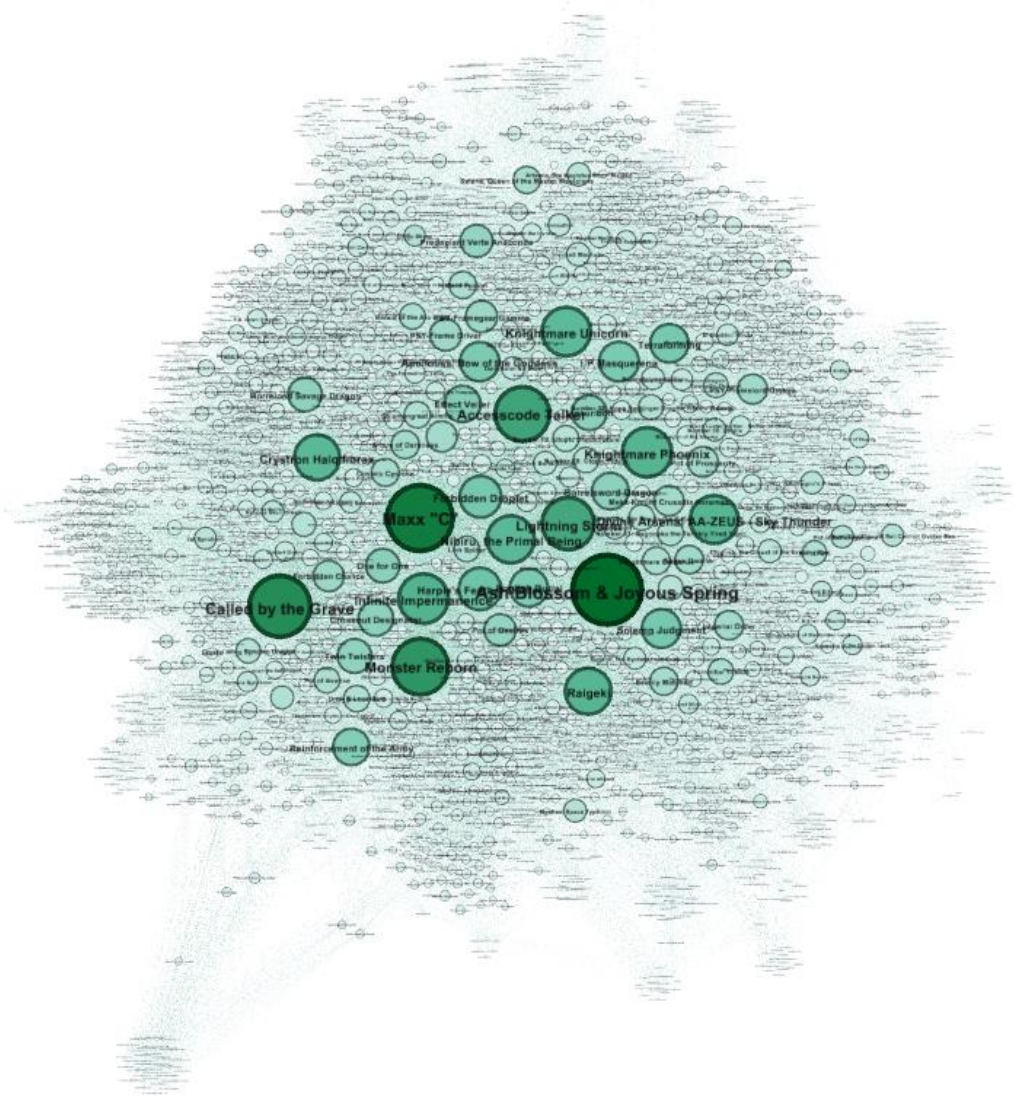
**Results:**




Distribution: very little cards are above 0.6-1, most cards are around 0.13.

| Id | Label | Degree | Closeness Centra... | Harmonic Closeness Cent... | Betweenness Centra... | Eigenvector Centr... |
|---|---|---|---|---|---|---|
| 14558127 | Ash Blossom & Joyous Spring | 1690 | 0.765389 | 0.846738 | 353750.547297 | 1.0 |
| 23434538 | Maxx "C" | 1540 | 0.727463 | 0.814868 | 248246.893499 | 0.945951 |
| 24224830 | Called by the Grave | 1365 | 0.6945 | 0.780057 | 209795.94466 | 0.878087 |
| 83764719 | Monster Reborn | 1273 | 0.673204 | 0.759882 | 196840.920406 | 0.802247 |
| 86066372 | Accesscode Talker | 1124 | 0.647106 | 0.729517 | 106447.37711 | 0.776878 |
| 14532163 | Lightning Storm | 1082 | 0.642669 | 0.721994 | 115899.007672 | 0.742685 |
| 10045474 | Infinite Impermanence | 952 | 0.618842 | 0.694228 | 64255.283034 | 0.703365 |
| 2857636 | Knightmare Phoenix | 938 | 0.61665 | 0.691355 | 66352.976546 | 0.698267 |
| 38342335 | Knightmare Unicorn | 883 | 0.608186 | 0.680071 | 55059.210999 | 0.688027 |
| 90448279 | Divine Arsenal AA-ZEUS - Sky Thunder | 934 | 0.616026 | 0.690535 | 70131.093154 | 0.679678 |
| 27204311 | Nibiru, the Primal Being | 897 | 0.610318 | 0.682944 | 55910.523696 | 0.669891 |
| 12580477 | Raigeki | 924 | 0.614473 | 0.688483 | 77810.775567 | 0.651077 |
| 50588353 | Crystron Halqifibrax | 897 | 0.610318 | 0.682944 | 71305.839367 | 0.638435 |
| 18144507 | Harpie's Feather Duster | 760 | 0.589644 | 0.65463 | 43737.204977 | 0.590763 |
| 24299458 | Forbidden Droplet | 708 | 0.584974 | 0.645261 | 33637.734892 | 0.580586 |
| 65741786 | I:P Masquerena | 679 | 0.578722 | 0.638216 | 30067.398061 | 0.551163 |
| 4280258 | Apollousa, Bow of the Goddess | 663 | 0.576532 | 0.634934 | 27899.63278 | 0.544149 |
| 41420027 | Solemn Judgment | 698 | 0.58093 | 0.641909 | 46257.256348 | 0.539072 |
| 73628505 | Terraforming | 702 | 0.581901 | 0.642935 | 58921.109193 | 0.532958 |
| 81439173 | Foolish Burial | 668 | 0.577215 | 0.63596 | 38511.841682 | 0.526041 |
| 85289965 | Borrelsword Dragon | 576 | 0.564512 | 0.616879 | 21893.632365 | 0.506207 |
| 32807846 | Reinforcement of the Army | 626 | 0.571127 | 0.627137 | 38796.907662 | 0.503916 |

Top nodes based on Eigenvector metric.




Node size is adjusted on Eigenvector metric.

Eigenvector counts the influence of a node in the network, if a node is connected to an important node it gets more score on their influence of the network. It calculates how important is the node you are connected to not only to how many nodes you are.

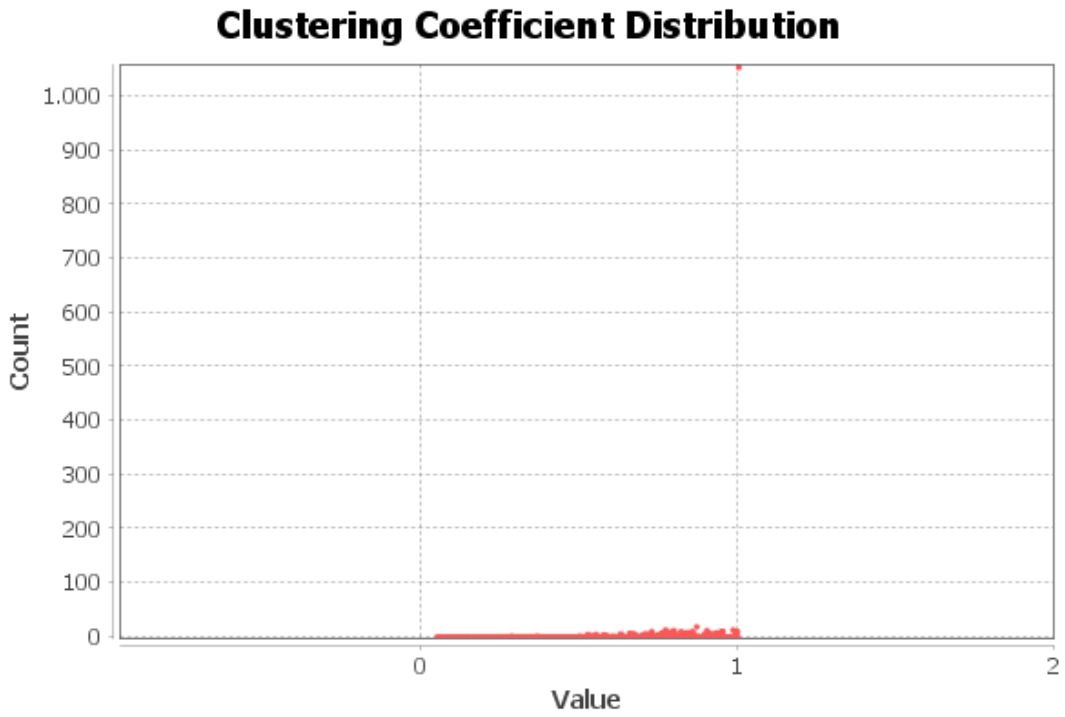
There is not much difference between centrality measures in our network, there are some exceptions but in general we know which nodes are well centered in our graph.

## Clustering effects:

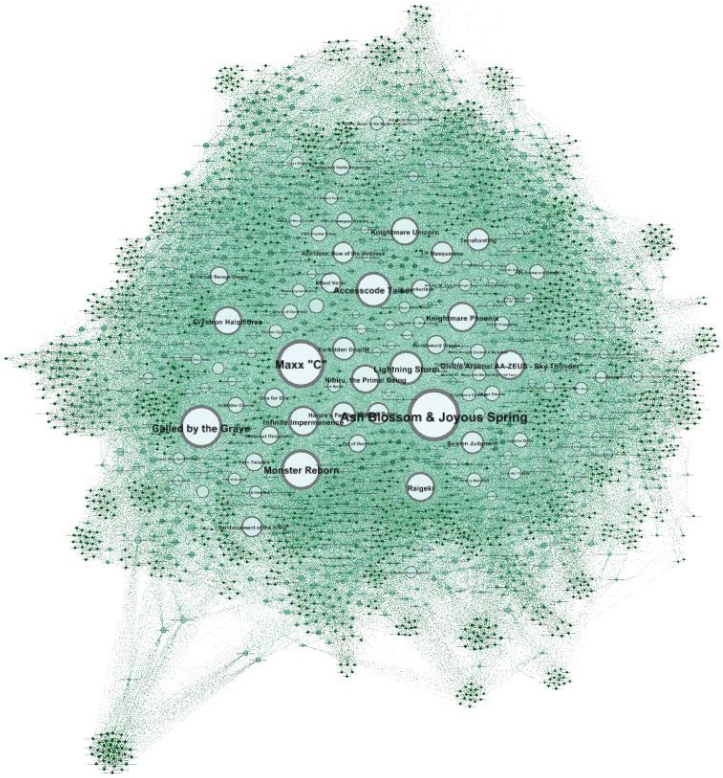
Average clustering coefficient: 0.8
Number of triangles: 1,595,904

**Results:**

Average Clustering Coefficient: 0,800
Total triangles: 1595904
The Average Clustering Coefficient is the mean value of individual coefficients.




Nodes with big degree tend to have low clustering coefficient, and nodes with lower degree have a high coefficient.




We can see that green nodes have high coefficient, it's more likely for them to form new edges (triadic closure).
If there are 3 nodes connected with A – B, B – C, then its more likely to form a new edge A - C in our next version of a graph.
In practice, we can think for 3 cards where A – B, B – C, have a lot of synergy and let's say A – C do not have, then in a next month's version of meta decks is more possible to see those 3 cards on the same deck, A – C combination will still not be a good thing but these cards would not end up on the same deck without an undirect reason (B).
The average of the coefficient indicates our network is well connected which is confirmed by previous statistics such as number of edges and density.

## Bridges and local bridges:

A bridge is an edge that unifies two parts of the graph, if the bridge would be deleted then we would have two separated components. Since our network is dense we do not have any edge that is a global bridge, all our nodes are connected.
Local bridges are edges that when deleted they increase the shortest path between two nodes. I am going to use another python code to check for bridges, although it seems unlikely we have one, it's a great chance to get my hand into another library. So, in order for a local bridge to exist we need to find two connected nodes that do not have a common friend, which is fairly difficult in our dense network with average path of 2.

bridges.py

```
import pandas as pd
import networkx as nx

#Read edges file
df = pd.read_csv("Edges.csv")

#Turn it into a graph
G = nx.from_pandas_edgelist(df, source='Source', target='Target',
create_using=nx.DiGraph())

#We need to transform our graph into an undirected graph
G2 = G.to_undirected()

#Prints the list of bridges
print(G2)
print(list(nx.bridges(G2)))
print(list(nx.local_bridges(G2)))

============== RESTART: C:\Users\Alexis\Desktop\yugioh\network.py
==============
Graph with 2429 nodes and 90433 edges
[]
[]
>>>
```

Turns out there are neither bridges nor local bridges on our network.

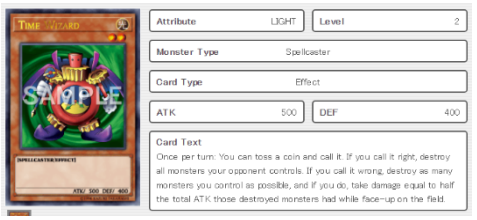
## Homophily:

With our data we do not have any node attributes that we can make a hypothesis that nodes with the same attribute are more likely to be connected.
Although we can check with networkX if the nodes with similar degree have the same connections.
homophily.py

```
print(nx.degree_assortativity_coefficient(G2))
-0.15649205292940063
>>>
```

The assortativity_coefficient is negative which means there is no indication that this would happen.

It would be interested if we checked if there is Homophily in other card/node attributes imported by the API in a less succinct analysis.
Each card has a card type: Monster Card/ Trap Card/ Spell Card and more
Each Monster card has a type: Aqua/Beast/Dinosaur and much more..
Each card has an attribute: DARK,DIVINE,EARTH,FIRE and more




We can also get a price attribute, the price that a card costs to buy online, categorize them by price range and see if expensive cards tend to connect with each other.
We can also add the set name the card was released, since cards that are released on the same set tend to work together (high synergy).

There are plenty of node attributes we could check the homophily of the network other than the degree which would make more sense.


```
{
  "id": 6983839,
  "name": "Tornado Dragon",
  "type": "XYZ Monster",
  "desc": "2 Level 4 monsters\nOnce per turn (Quick Effect): You can detach 1 material from this card, then target 1 Spell/Trap on the field; destroy it.",
  "atk": 2100,
  "def": 2000,
  "level": 4,
  "race": "Wyrm",
  "attribute": "WIND",
  "card_sets": [
    {
      "set_name": "Battles of Legend: Relentless Revenge",
      "set_code": "BLRR-EN084",
      "set_rarity": "Secret Rare",
      "set_rarity_code": "(ScR)",
      "set_price": "4.08"
    },
    {
      "set_name": "Duel Devastator",
      "set_code": "DUDE-EN019",
      "set_rarity": "Ultra Rare",
      "set_rarity_code": "(UR)",
      "set_price": "1.4"
    },
    {
      "set_name": "Maximum Crisis",
      "set_code": "MACR-EN081",
      "set_rarity": "Secret Rare",
      "set_rarity_code": "(ScR)",
      "set_price": "4.32"
    }
  ],
  "card_images": [
    {
      "id": 6983839,
      "image_url": "https://storage.googleapis.com/ygoprodeck.com/pics/6983839.jpg",
      "image_url_small": "https://storage.googleapis.com/ygoprodeck.com/pics_small/6983839.jpg"
    }
  ],
  "card_prices": [
    {
      "cardmarket_price": "0.42",
      "tcgplayer_price": "0.48",
      "ebay_price": "2.99",
      "amazon_price": "0.77",
      "coolstuffinc_price": "0.99"
    }
  ]
}
]
```

## Graph density:

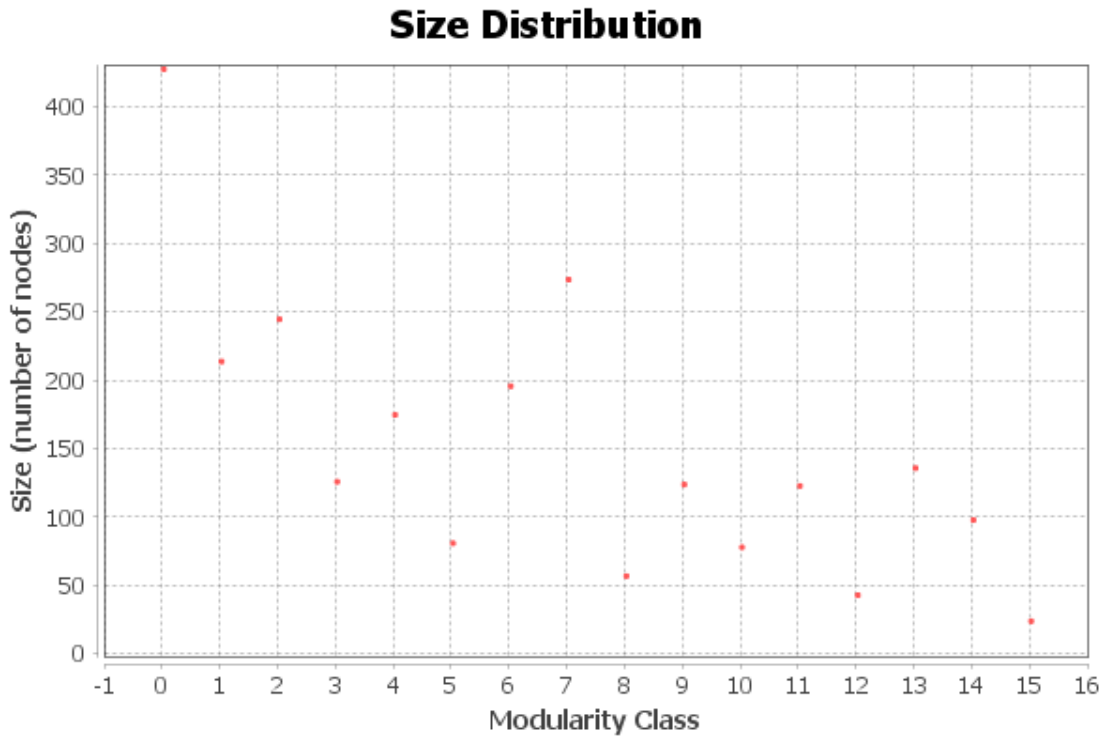
Graph density is a simple metric that shows how well connected the nodes are. We simple calculate the maximum possible number of edges the graph can have in a ratio with how many edges we have.
Density is 0.031 in our network which is very high compared to other common networks like a social network. So, we can say our graph is dense. The reason why we have a lot of edges on our graph is explained in the topology section.

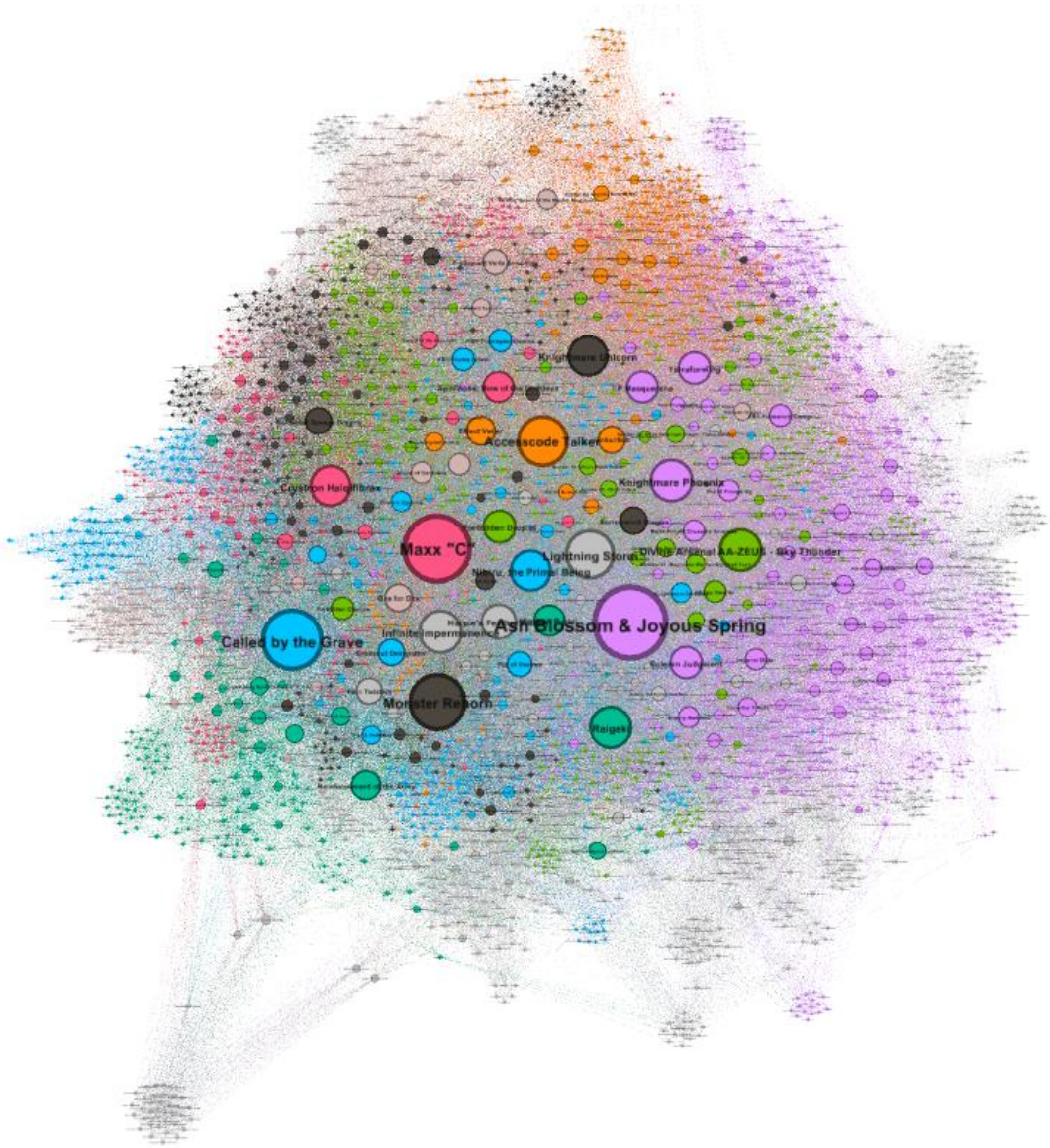
## Community structure (modularity), cliques:

Modularity is a measure that shows how much diverse there is in the network, how much the network can be split in groups. Our network is quite dense so there is high modularity (0.379). The algorithm splits the network into 16 groups. (With resolution parameter = 1)

## Results:

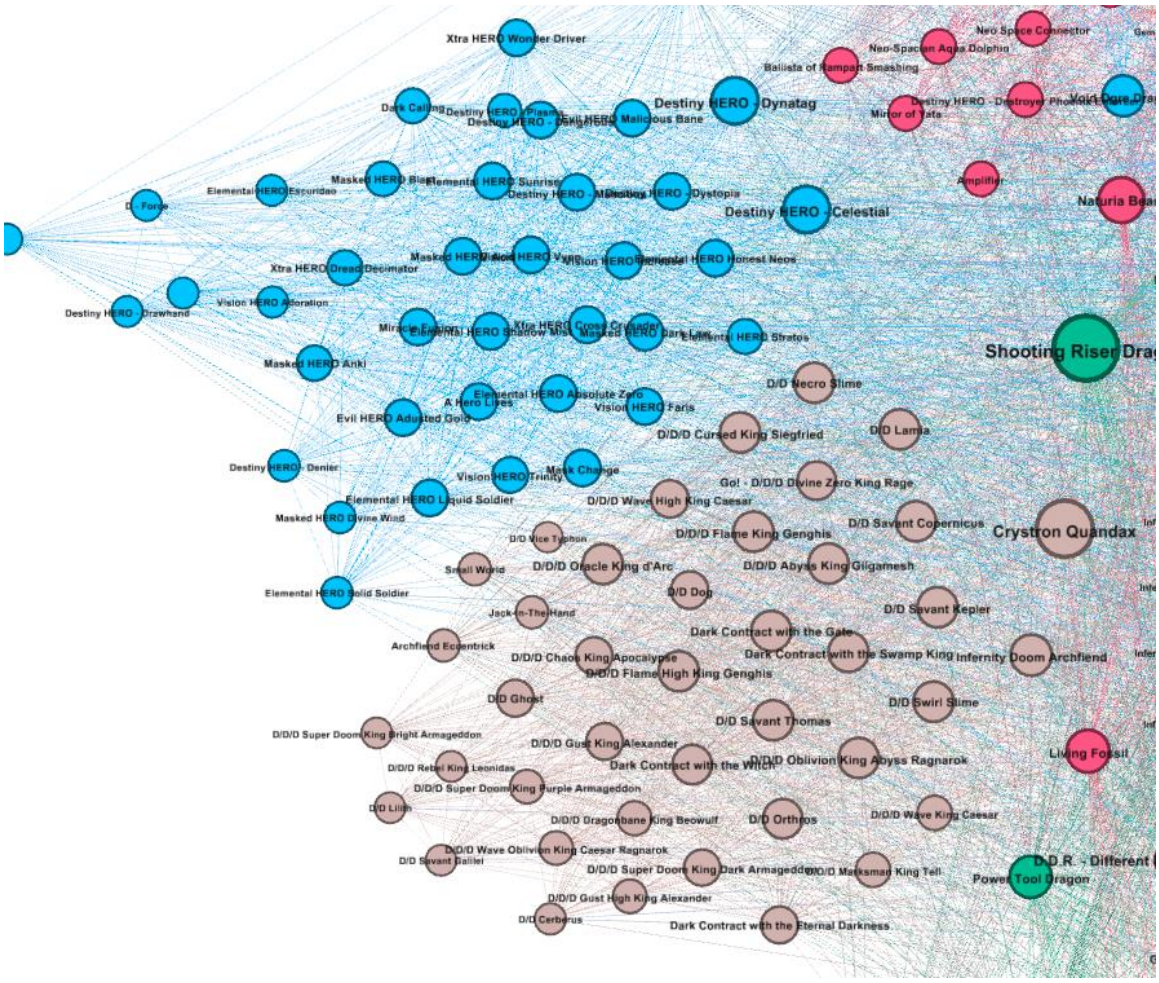
Modularity: 0,379
Modularity with resolution: 0,379
Number of Communities: 16



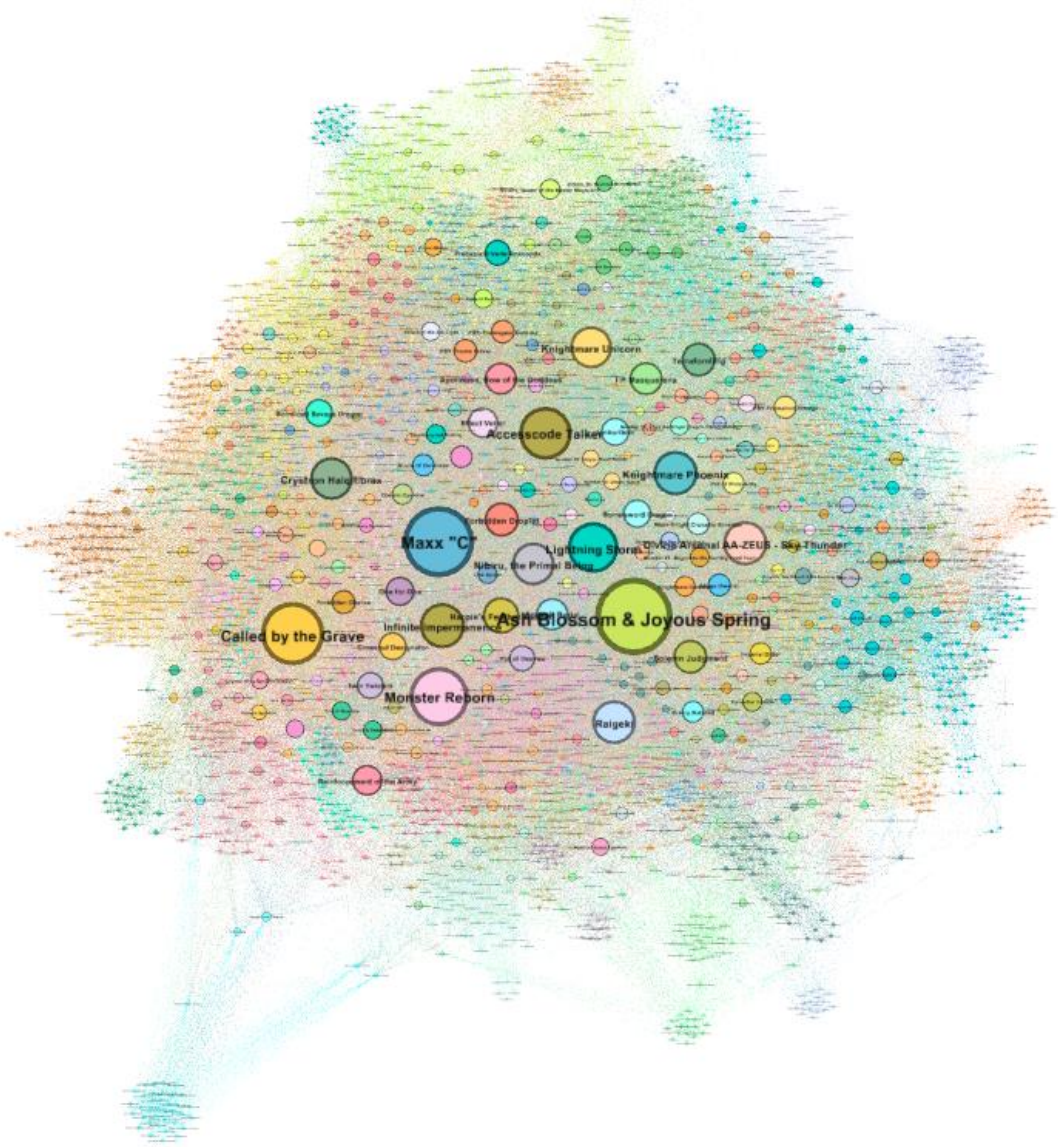



This is very interesting to see since there are a lot of communities in our network. Some cards are part of series, and they usually have similar names and can work with other cards of the same series.

For example, we can notice the left part of the graph below, the cards from HERO series are in different group than the D/D/D series.




The algorithm has split all the series like that in the network with different colors. In general, each group has cards from series including more general-purpose cards in the center.




Increasing number of groups (79) make the distinction clearer. (Resolution = 0.2)

We can find the maximal cliques on our graph. A clique is a complete graph that contains a node, a complete graph is a group of nodes that are all connected with each other, so for each node on the graph we can find the maximal clique he belongs to, meaning the clique that has the most nodes and he is included.
The maximum clique is the biggest clique from all the maximal cliques (the biggest clique from all the nodes' biggest cliques).

This is easy with networkX, we already have imported our data to make a network, now we can iterate in nx.find_cliques(G2), and get all the maximal cliques in the graph.
In the code below we try to find the longest maximal clique, aka the maximum clique.

cliques.py

```
count = 0
maximumlength = 0
for clique in nx.find_cliques(G2):
    if len(clique) > maximumlength:
        maximumlength = len(clique)
        print(maximumlength)
        print(clique)
    count +=1
print(count)
>>>
============== RESTART: C:\Users\Alexis\Desktop\yugioh\cliques.py
==============
34
[40392714, 58577036, 7614732, 31437713, 90448279, 20137754, 35058588, 48608796,
423585, 2857636, 1218214, 35726888, 23434538, 72664875, 2434862, 34325937,
30270176, 41114306, 43040603, 86066372, 3775068, 15610297, 70369116, 39317553,
72336818, 38342335, 41999284, 54366836, 84013237, 51735257, 4998619, 9659580,
12580477, 90590303]
36
[31893528, 94772232, 58577036, 48068378, 67598234, 51993760, 82732705, 2857636,
595626, 23434538, 67616300, 40771118, 66547759, 72336818, 41999284, 19665973,
49238328, 88000953, 31829185, 50588353, 86066372, 30170981, 1966438, 58616392,
48934760, 102380, 67287533, 14386013, 52615248, 87988305, 14509651, 94212438,
77637979, 70369116, 1861629, 38342335]
46
[78876707, 59057152, 7452945, 95772051, 94730900, 58346901, 95953557, 81570454,
5361816, 51670553, 19680539, 74029853, 59934749, 43898403, 7811875, 32807846,
21351206, 23562407, 56824871, 51412776, 40251688, 75719089, 41359411, 88923963,
22091647, 24224830, 10613952, 14745409, 42472002, 37478723, 63789924, 24027078,
19748583, 55742055, 83438826, 83519853, 21223277, 15693423, 57690191, 90861137,
73359475, 42052439, 46008667, 77656797, 64867422, 42199039]
52
[71645242, 85840608, 34959756, 67922702, 2295440, 96363153, 9622164, 8487449,
44097050, 63362460, 59934749, 99674361, 32807846, 27548199, 48686504, 303660,
33420078, 33158448, 62706865, 35272499, 41999284, 79606837, 17955766, 50954680,
1118137, 2403771, 9742784, 65993085, 50588353, 67696066, 4280258, 81439174,
1200843, 61677004, 60071928, 94693857, 242146, 52947044, 60461804, 3113836,
83764718, 37780349, 7394770, 44935634, 28985331, 99645428, 75944053, 74997493,
51405049, 70369116, 39964797, 11066358]
217931
>>>
```

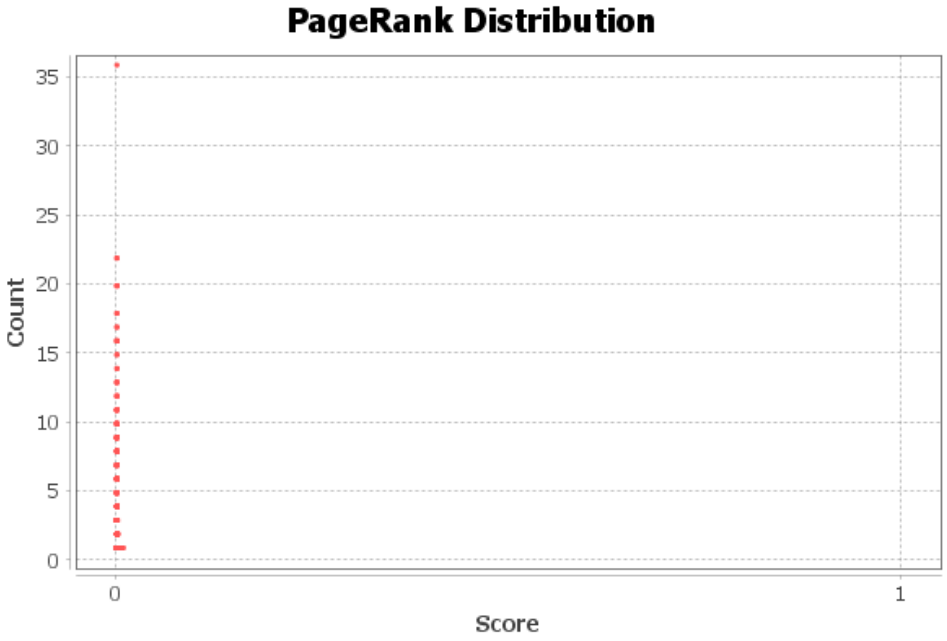
We iterated between 217k maximal cliques (cliques that are the biggest cliques for a node), this number is quite bigger than the nodes because a node can have multiple maximal cliques.
We found out that the last printed clique is the biggest in the graph with 52 members. (The numbers are the IDs of the cards).

## PageRank:

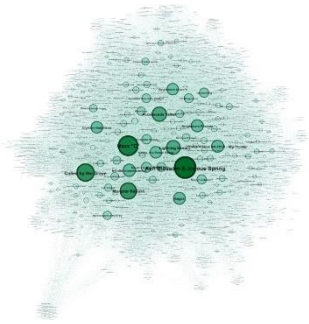
### Parameters:

Epsilon = 0.001
Probability = 0.85

### Results:




| Label | PageRank |
|---|---|
| Ash Blossom & Joyous Spring | 0.008442 |
| Maxx "C" | 0.007596 |
| Called by the Grave | 0.006647 |
| Monster Reborn | 0.006325 |
| Accesscode Talker | 0.005429 |
| Lightning Storm | 0.005304 |
| Infinite Impermanence | 0.004519 |
| Raigeki | 0.004518 |
| Divine Arsenal AA-ZEUS - Sky Thunder | 0.004494 |
| Knightmare Phoenix | 0.00448 |
| Crystron Halqifibrax | 0.004356 |
| Nibiru, the Primal Being | 0.004302 |
| Knightmare Unicorn | 0.004147 |
| Terraforming | 0.003486 |
| Solemn Judgment | 0.003419 |
| Foolish Burial | 0.003202 |
| I:P Masquerena | 0.003185 |
| Apollousa, Bow of the Goddess | 0.003093 |
| Reinforcement of the Army | 0.002996 |
| Effect Veiler | 0.002983 |
| One for One | 0.002906 |



Each node has a PageRank metric from almost 0 to 0.008.
The nodes order is like centrality measures which is a good thing for the network.
Similar graph with PageRank as color.
PageRank value is based on the number of edges each node has, and it works better with directed graphs. It's a repeating algorithm that calculates the PageRank metric based on the PageRank metrics of the nodes that are pointing the current node.
In our case it is showing the important of the nodes which returns the centrality of the node. It's used on web pages, that is why it works better with directed graphs (web pages have links pointing to other pages).

## Sources (citations):

Yu-Gi-Oh! Cards information:
Provided by API of db.ygoprodeck.com
Link: https://db.ygoprodeck.com/api/v7/cardinfo.php

Yu-Gi-Oh! Decks information:
Fetched from https://ygoprodeck.com/
Link: https://ygoprodeck.com/deck-search/?&_sf_s=master%20duel&sort_order=views&post_date=2022-01-18%202022-02-18&offset=0

Gephi:
All the visualizations and statistic graphs, as also the whole analysis of the network is being completed with the help of the Gephi tool.
Link: https://gephi.org/

Python:
Used to fetch and create the dataset, libraries that were used are pandas, os, BeatifulSoup, requests, json, time, Selenium, networkX and respective documentations

Lectures: Social Network Analysis
Link: https://edu.dmst.aueb.gr/course/view.php?id=24