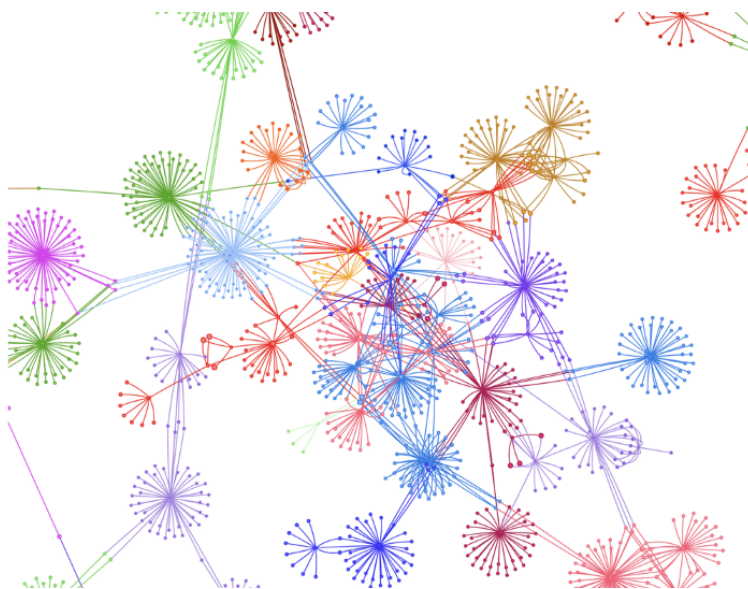




Valentina Alto

Follow

Apr 12, 2020 · 4 min read · Member-only

Image source: <http://carrefax.com/articles-blog/2018/10/14/case-law-network-graph>

## Introduction to Network science with NetworkX

Part 3: Visualizing the mathematical structure behind social networks' communities

Here we come to the third part of this series about graph theory and network science. Let's briefly recap what we've been learning so far.

In [Part 1](#), we saw the math behind networks and how to build them from scratch. Furthermore, we went through a set of metrics that characterize networks and saw how those latter can be clustered into three categories (random, small-world and preferential attachment networks).

[Part 2](#) was about [NetworkX](#), the Python package used to build networks from scratch, visualize them and simulate the dynamics of phenomena.

In this last article of the series, I will provide a concrete example of modeling phenomena. More specifically, I'm going to provide a visual representation of Facebook communities, using the social network dataset from Stanford University, available [here](#). It contains the aggregated networks of 10 individuals' Facebook friends list. Within this framework, individuals will be represented by nodes, while edges will be the "Facebook friendship" among those individuals.

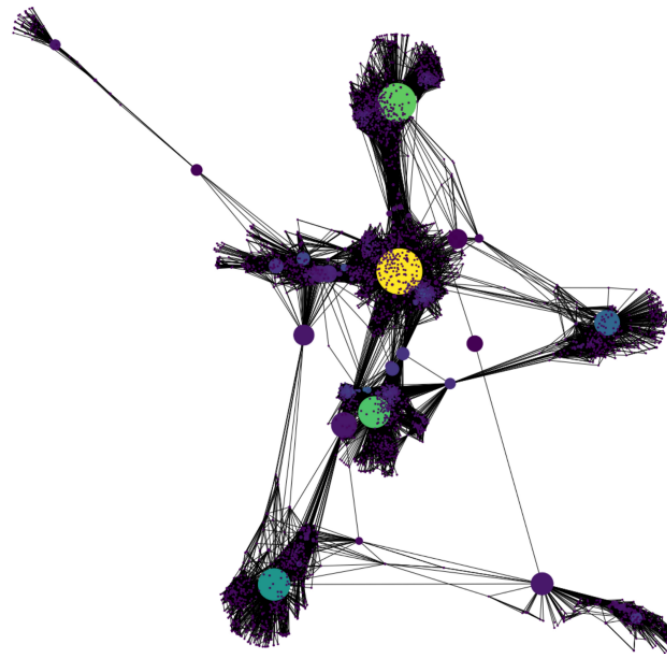
In this analysis, I will use the community detector algorithm module, which can be installed via *pip* following those [instructions](#). Furthermore, for a nicer visualization, I will also use an amazing package called Netwulf (I recommend you to have a look at the official documentation [here](#), it is really interesting!).

So let's first start by importing the necessary packages and building our network from the Facebook dataset:

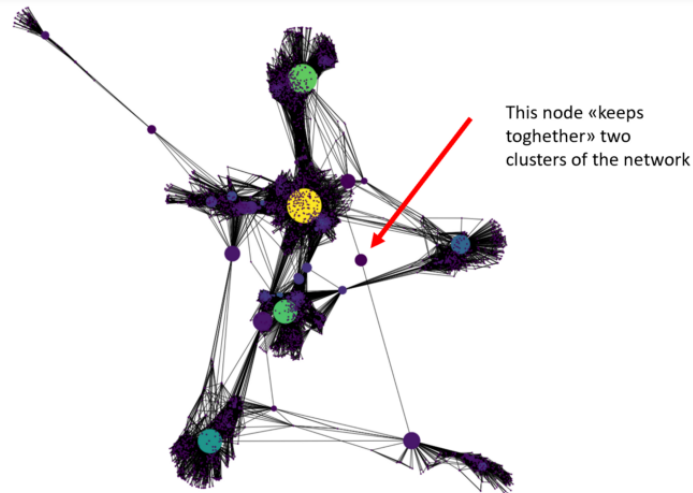
```
G_fb = nx.read_edgelist("facebook_combined.txt.gz", create_using =
nx.Graph(), nodetype=int)
```

Now I'm going to set up my visualization environment. Since we will have a huge amount of nodes and edges, we want to make sure to have a clear visualization. Hence, I'm going to set the nodes' size proportional to nodes' betweenness centrality, while their color will be equal to their degrees (if you are not familiar with these concepts, make sure to read [Part 1!](#)).

```
pos = nx.spring_layout(G_fb) #setting layout
betCent = nx.betweenness_centrality(G_fb, normalized=True,
endpoints=True)
node_color = [20000.0 * G_fb.degree(v) for v in G_fb]
node_size = [v * 10000 for v in betCent.values()]
plt.figure(figsize=(20,20))
nx.draw_networkx(G_fb, pos=pos, with_labels=False,
node_color=node_color,
node_size=node_size )
plt.axis('off')
```



Let's focus on this picture and make some considerations. First of all, you can well appreciate the importance of betweenness centrality here (remember that it is indicated by the sizes of nodes). Indeed, besides those nodes for which it is evident their importance (namely, the yellow one) there are some which have a very low degree, yet are extremely important. Namely, consider the following node:



It has degree=2, yet its betweenness centrality is pretty high.

Furthermore, you can notice how, even those the 10 individuals might not be friends among each other, they are connected anyway via other individuals. There is no “isolated” cluster of nodes: at least one “bridge node” exists in between.

Now let's move on and try to detect communities among those individuals. The algorithm behind community detection is that of a clustering task. The criterion of clustering is that good communities should have a high number of intra-community edges, while that of inter-community should be low.

As mentioned above, the algorithm can be easily implemented by importing the module *community*.

```
import community as c

parts = c.best_partition(G_fb)
values = [parts.get(node) for node in G_fb.nodes()]

pos = nx.spring_layout(G_fb)
betCent = nx.betweenness_centrality(G_fb, normalized=True,
                                     endpoints=True)
node_size = [v * 10000 for v in betCent.values()]
plt.figure(figsize=(20,20))

nx.draw_networkx(G_fb, pos=pos, with_labels=False,
                 node_color=values,
                 node_size=node_size )

plt.axis('off')
```



Now each community is represented with a different color. As you can see, the majority of communities are clustered around the individual's network. However, it is interesting to notice how “outlier” individuals (those who belong to a community that is far away from them) are those who create connections among clusters.

Finally, as promised, let's have an interactive visualization of our network:

```
import netwulf as nw
nw.visualize(G_fb)
```

With Netwulf, you have the possibility to customize the visualization of your network, without previously configuring it on Python.

### Conclusions

Network science is an open research field and it is incredible how new complex systems, arising every day, can find mathematical representations thanks to Networks. Plus, besides their macro analysis of one system, Networks also provide a closer insight into agent roles within that systems: thanks to

[Sign In](#)[Get started](#)

I hope you enjoyed this series about Network science and, if you are curious to learn more, in the references there are some interesting articles worth readings!

### References

- <https://snap.stanford.edu/data/egonets-Facebook.html>
- <https://python-louvain.readthedocs.io/en/latest/>
- <https://netwulf.readthedocs.io/en/latest/>
- <http://www.network-science.org/>
- <http://ryancompton.net/2014/06/16/community-detection-and-colored-plotting-in-networkx/>



--




### More from Python in Plain English

[Follow](#)

New Python content every day. Follow to join 500k+ monthly readers.

[Read more from Python in Plain English](#)


### Recommended from Medium

 Vukašin Višković

**Data Science during Holidays—  
an examination of Google  
Trends Data**

 Santiago Gonçalves Moreira Dian...


**Accelerating COVID-19 qPCR  
analysis with machine learning.**

 Jenny Hung

**Market Randomness and  
Market Direction**

 Andrew Benesh

**GA COVID-19 Report March 1,  
2021**

 Rishav Kumar Roy

**K-Means Clustering**

 Sunny Srinidhi in DataSeries

**Here's how you can improve  
your Netflix recommendations**

 Ali Mahmudan

**The Comparison of Spatial  
Models**

 DEANDREE

**Can you profit in Crypto  
Markets with not-so-basic  
strategies [Part2] ?**

[About](#) [Help](#) [Terms](#) [Privacy](#)