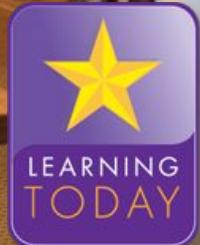


Lesson #07

Network Analysis in Python: introduction



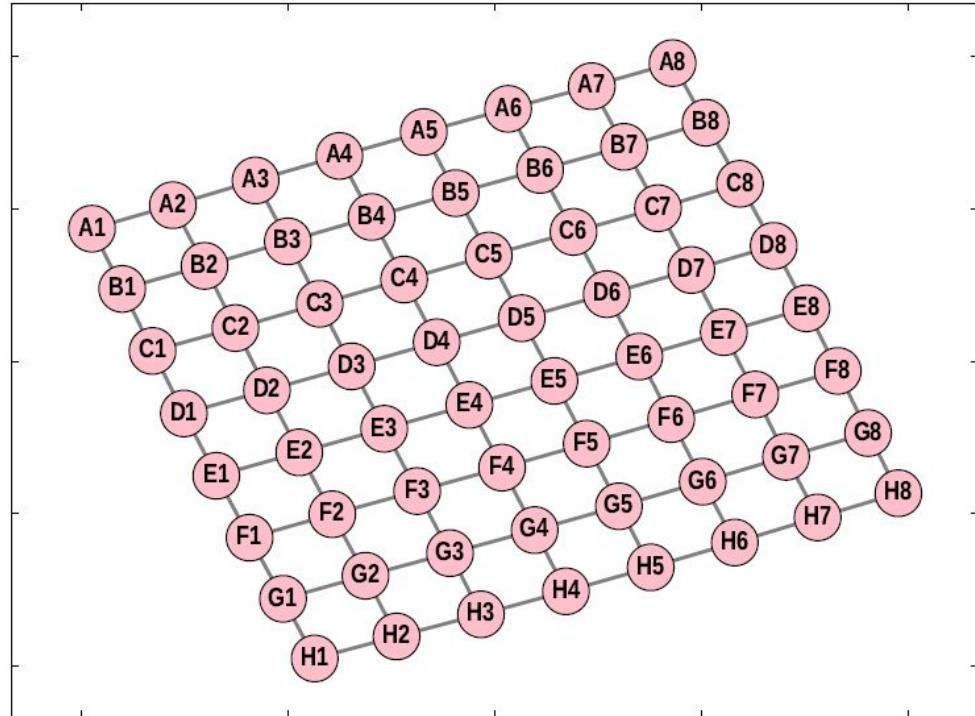
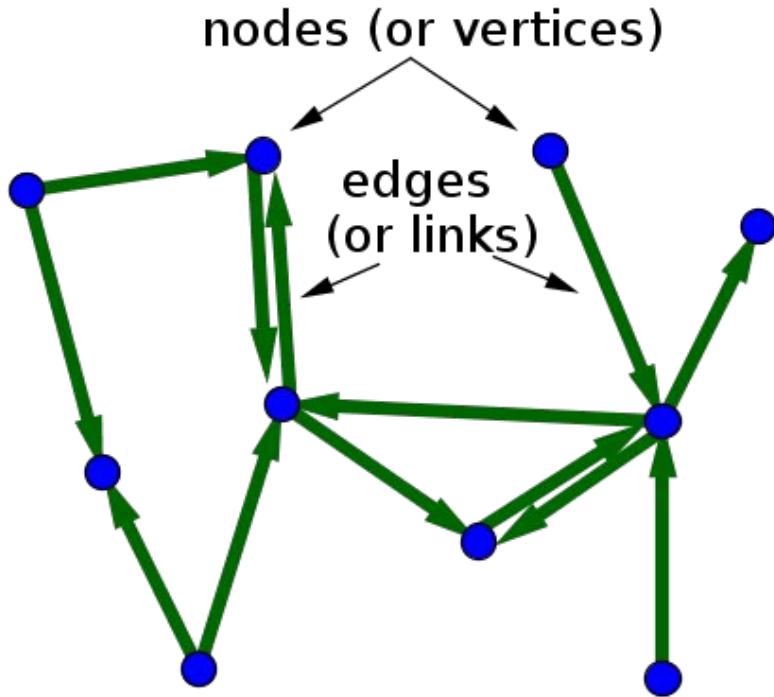
Introduction to NetworkX
Construct a simple network with NetworkX
Add attributes
Visualize a network with matplotlib and nxviz
Share and preserve networks

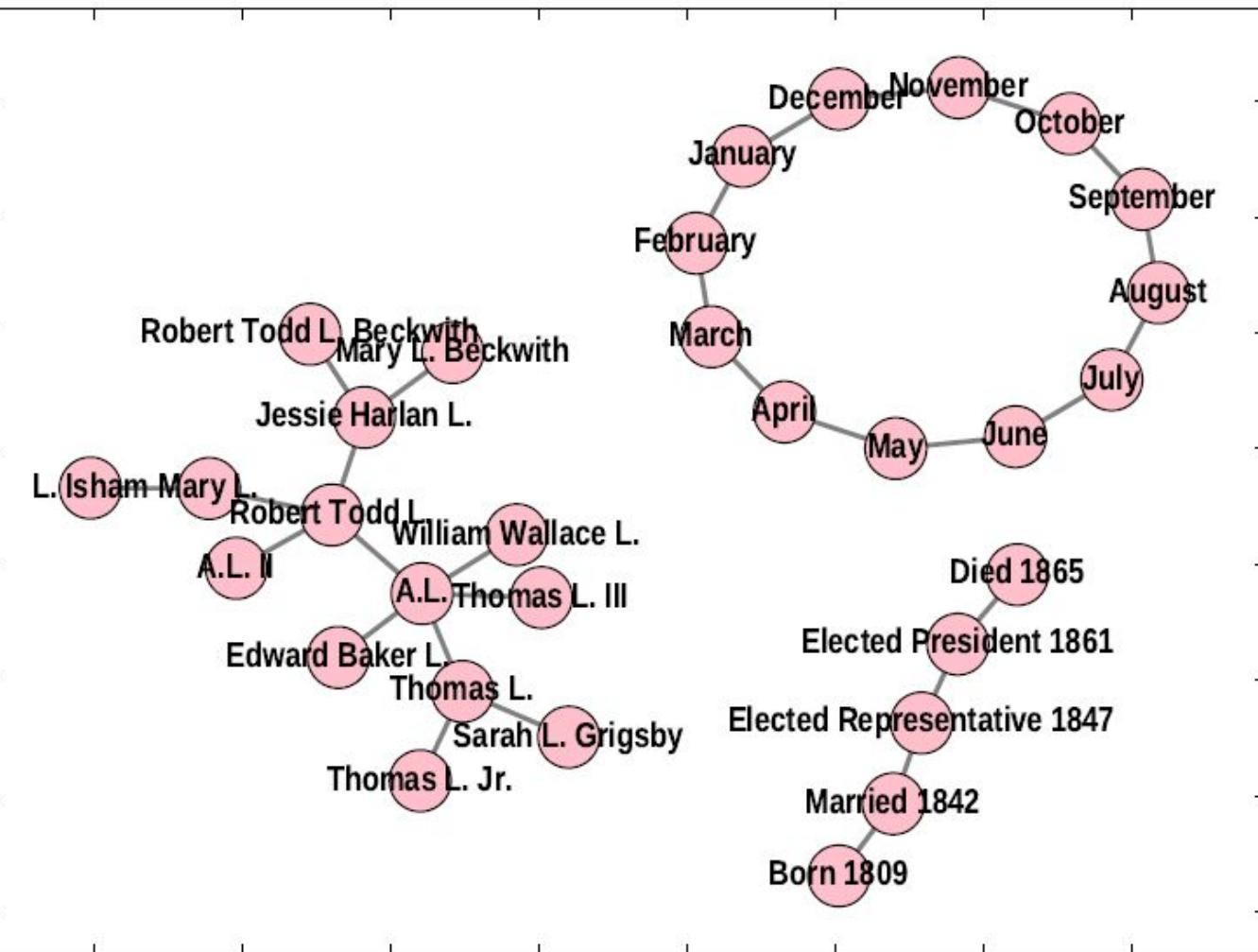


What is a network?



Know your networks

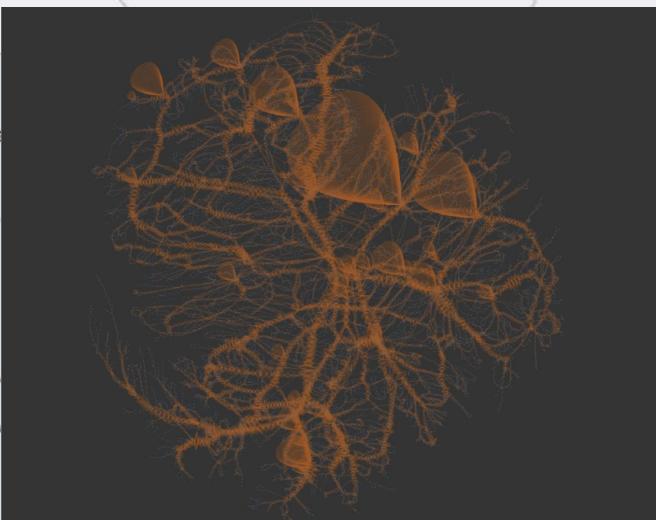
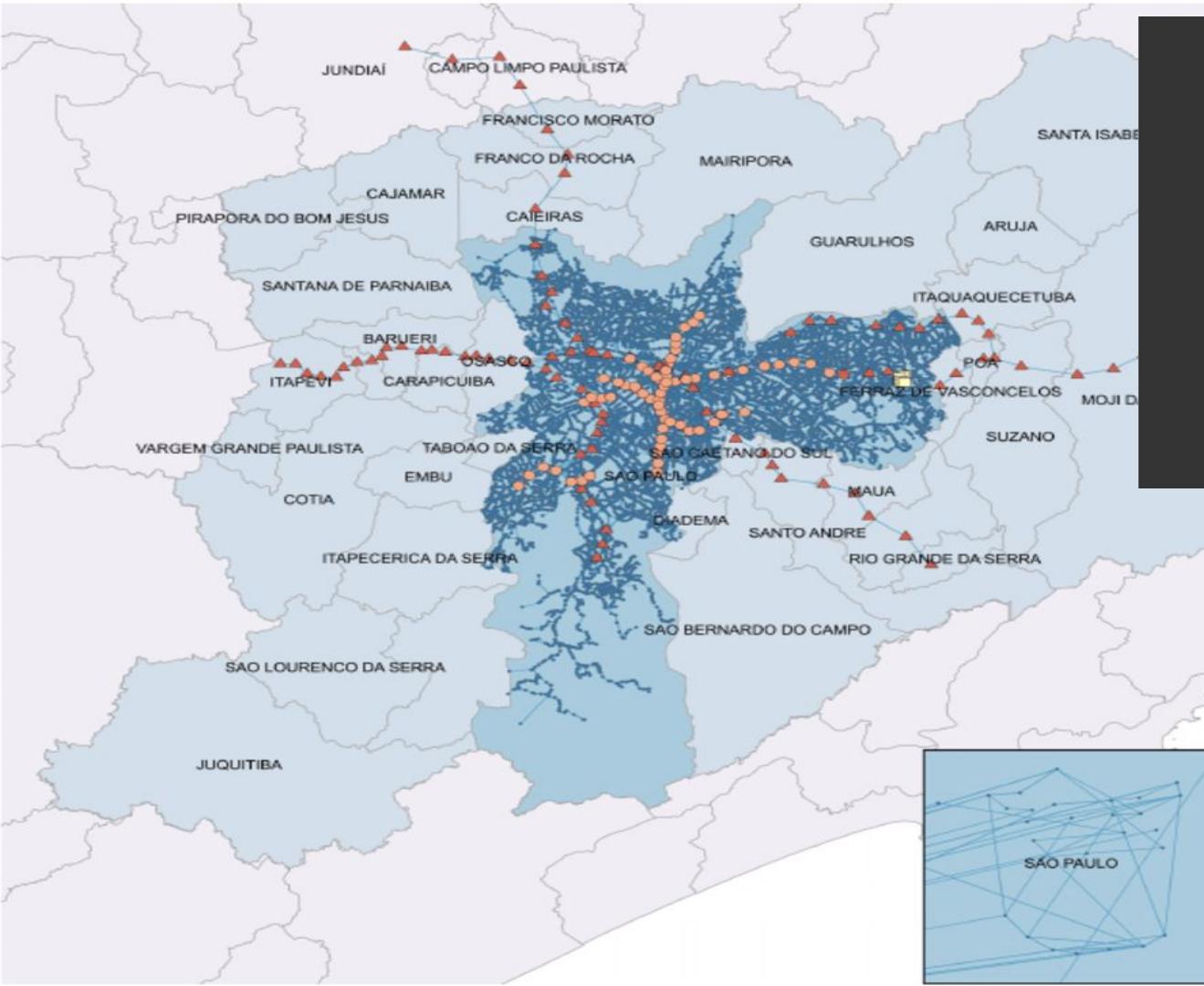




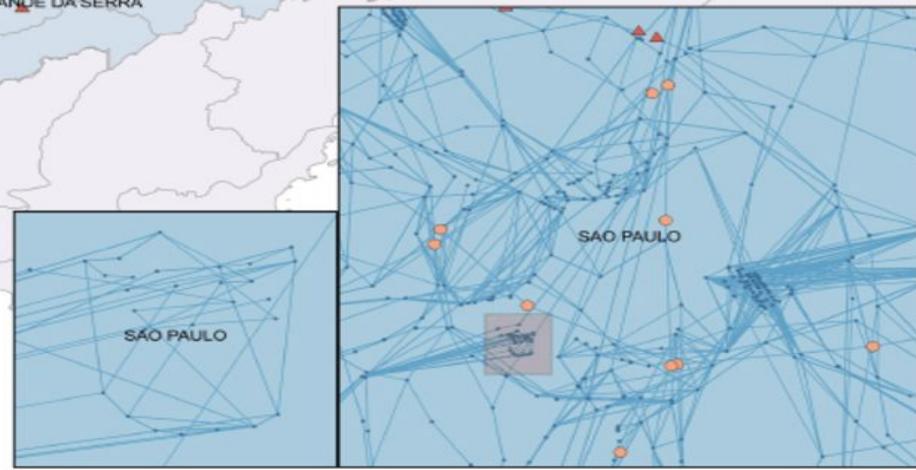
All the networks listed previously are simple because they have a regular or almost regular structure.

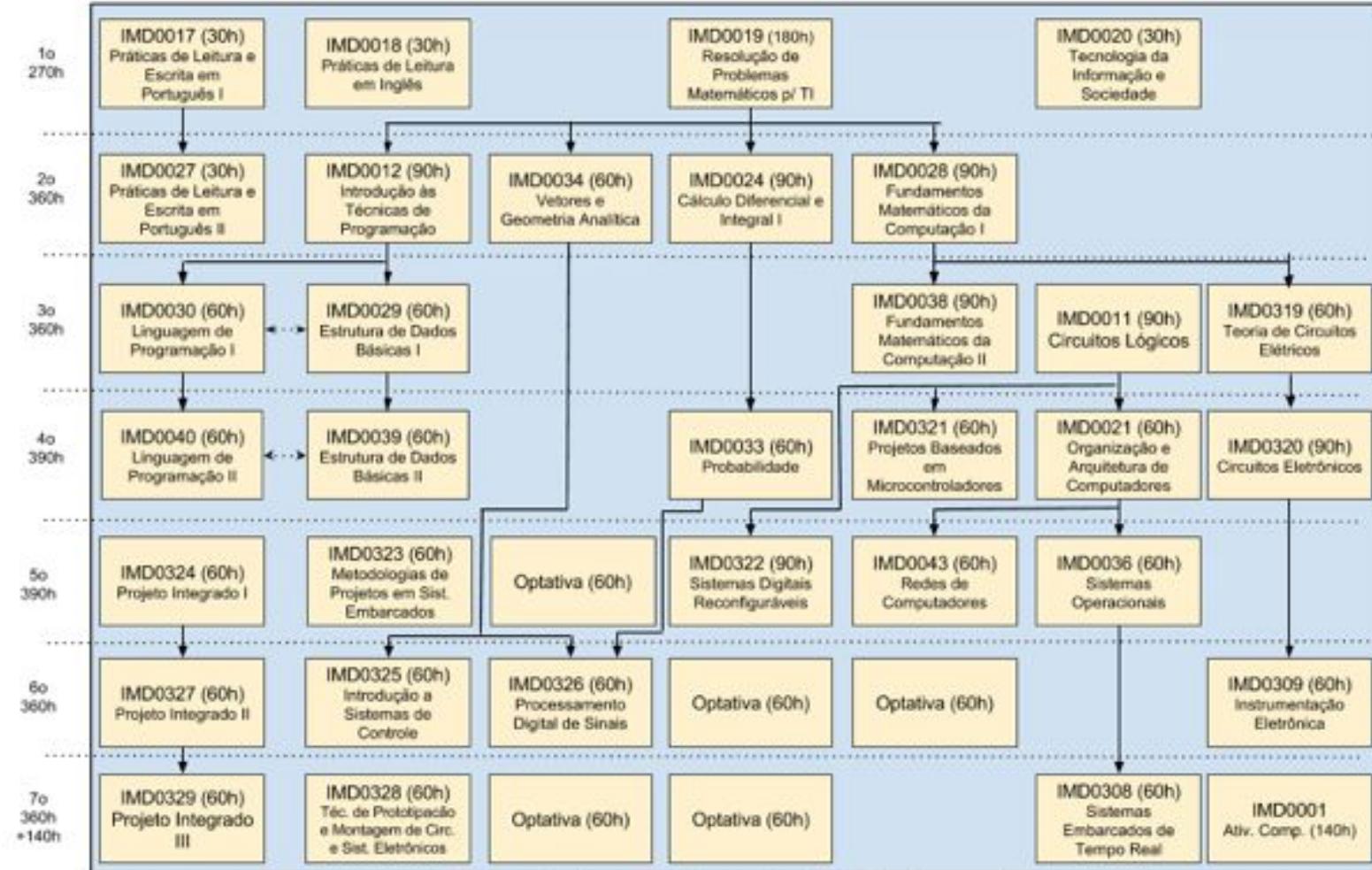
A simple network is simple not because it is small, but because it is regular.

Complex Network Analysis (CNA)



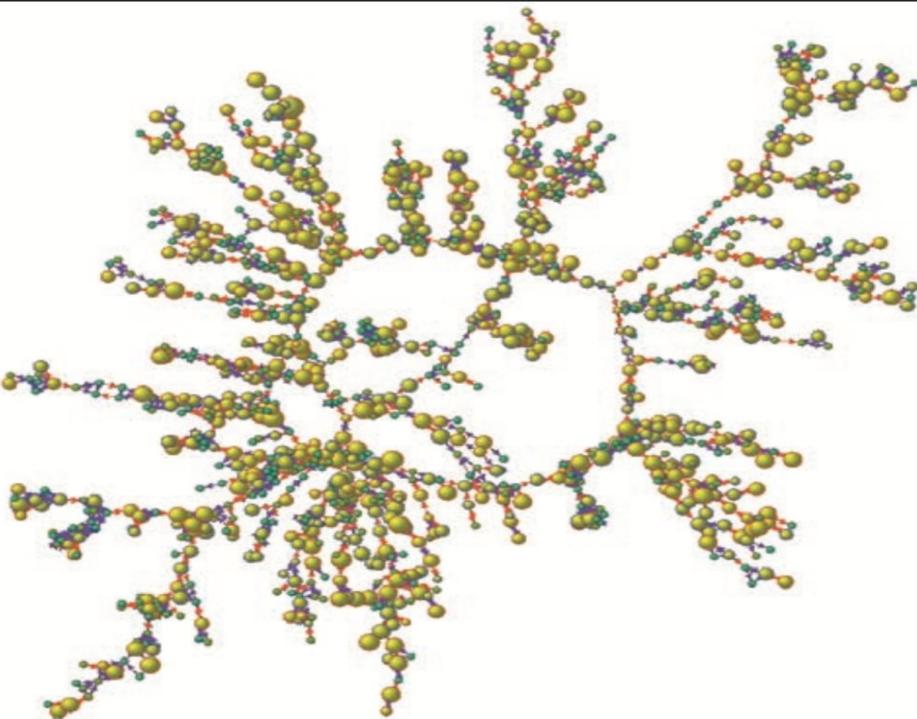
<https://goo.gl/I8SS2X>



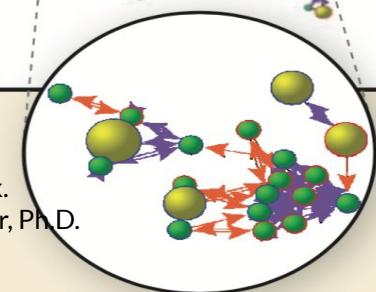
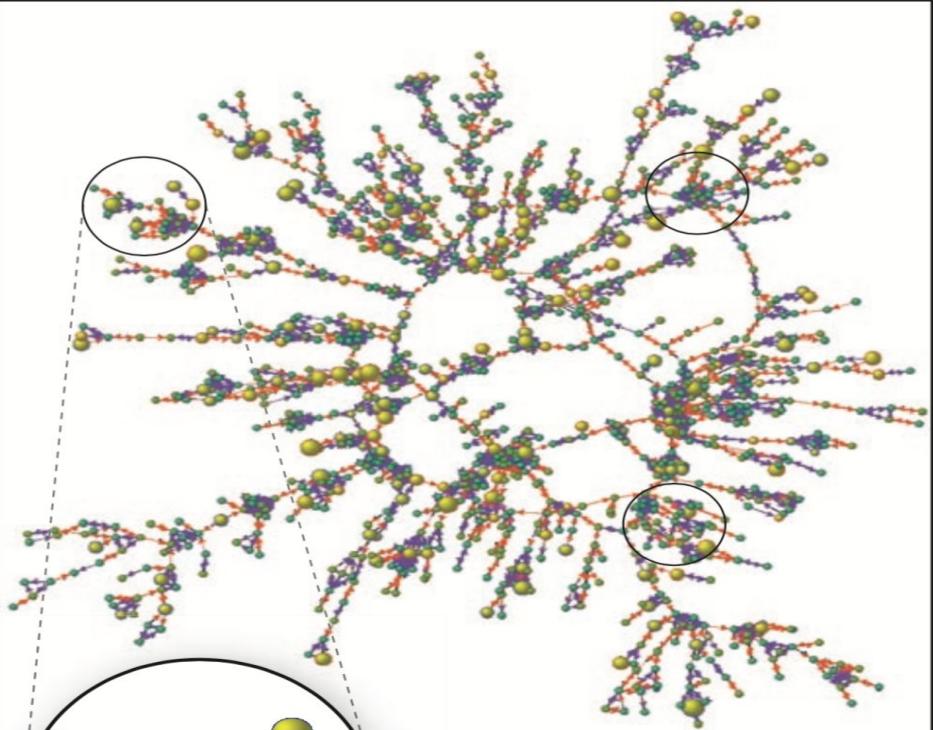


Bacharelado em Tecnologia da Informação Ênfase em Sistemas Embarcados (MT)

1971



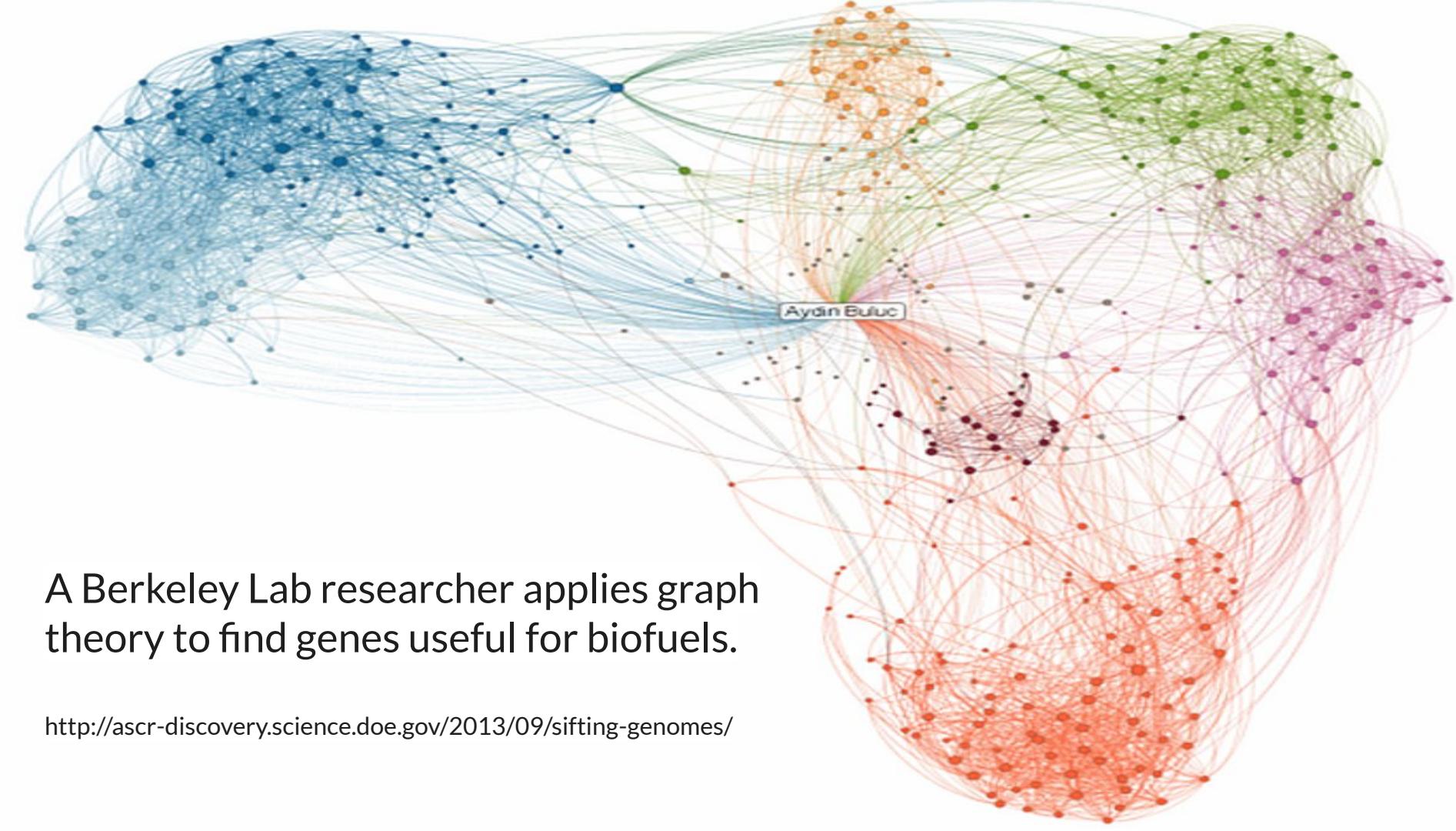
2000



Red borders - women
Blue border - men
Interior color

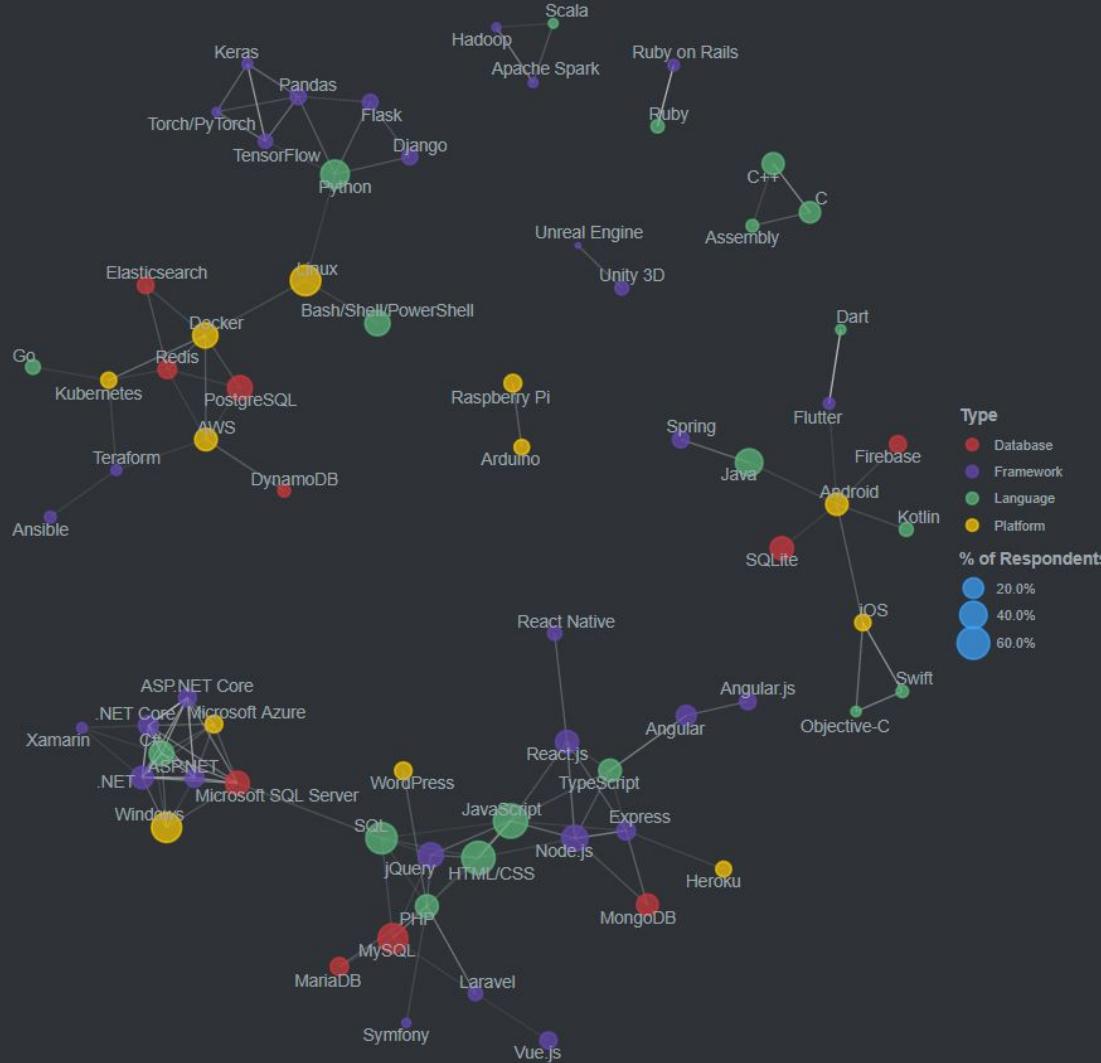
Yellow ≥ 1 cigarette
Green => no cigarette

The Collective Dynamics of Smoking in a Large Social Network.
Nicholas A. Christakis, M.D., Ph.D., M.P.H., and James H. Fowler, Ph.D.
N Engl J Med 2008; 358:2249-2258 May 22, 2008



A Berkeley Lab researcher applies graph theory to find genes useful for biofuels.

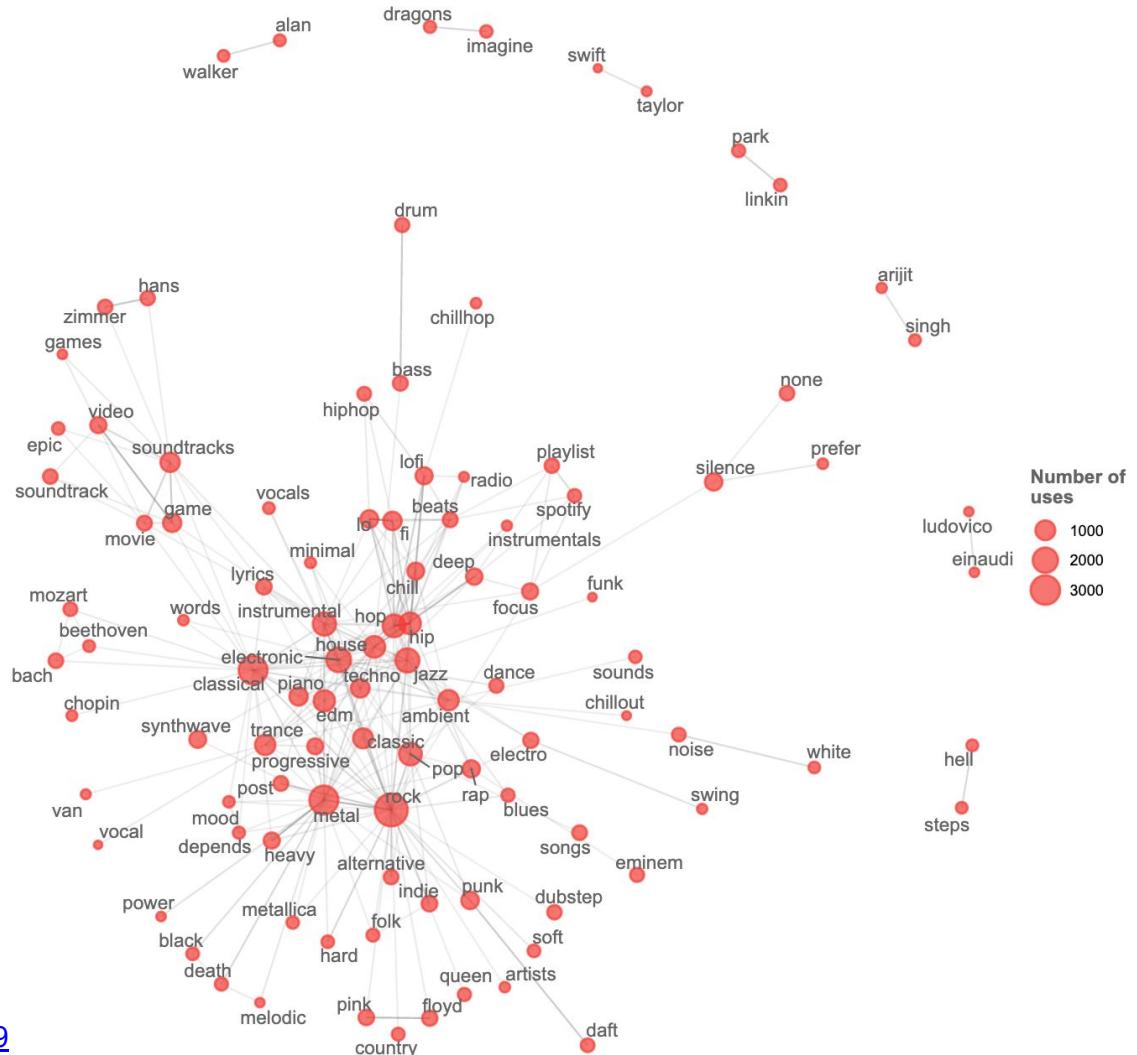
<http://ascr-discovery.science.doe.gov/2013/09/sifting-genomes/>



How the technologies are connected

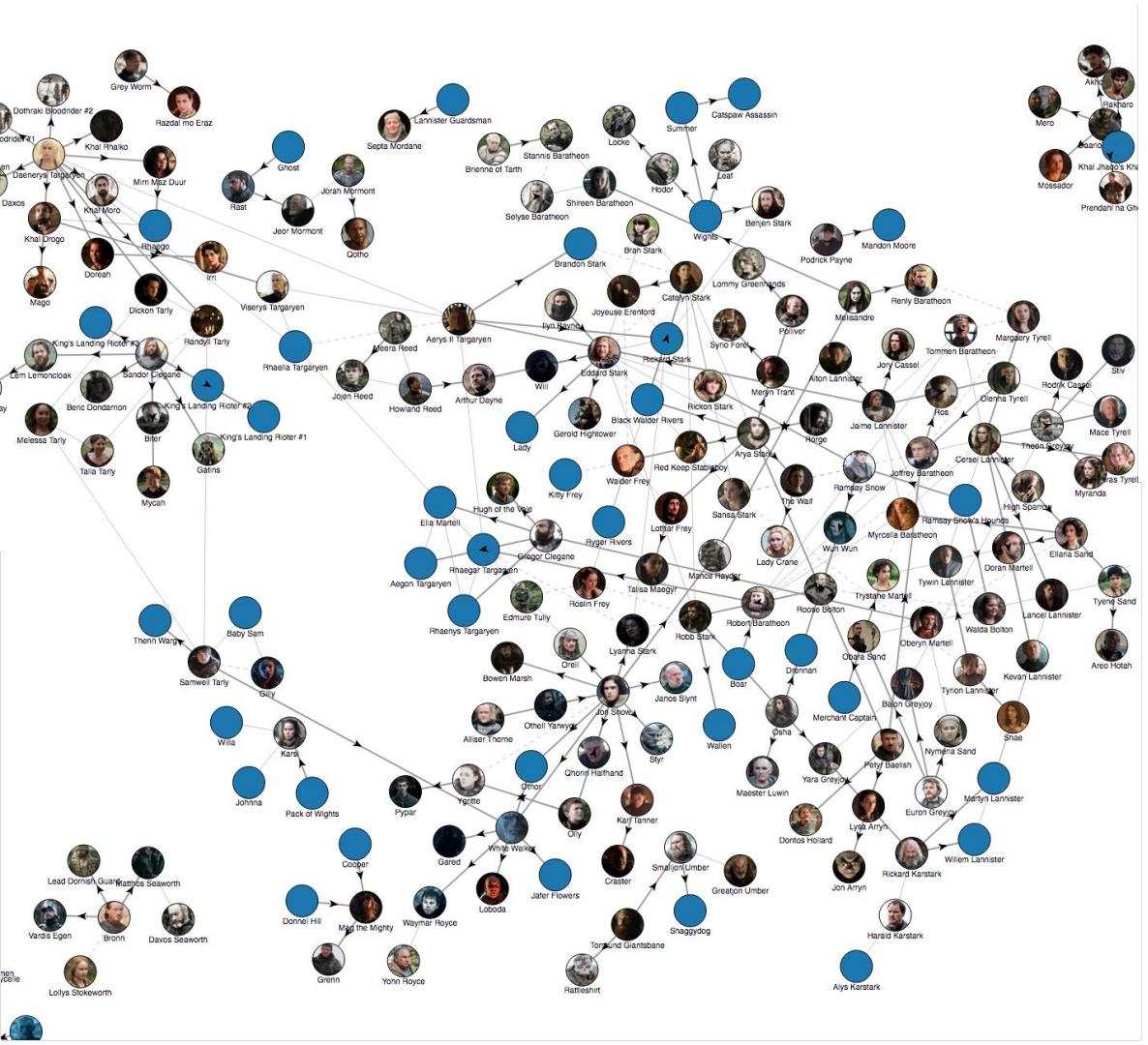


Music for focus while coding

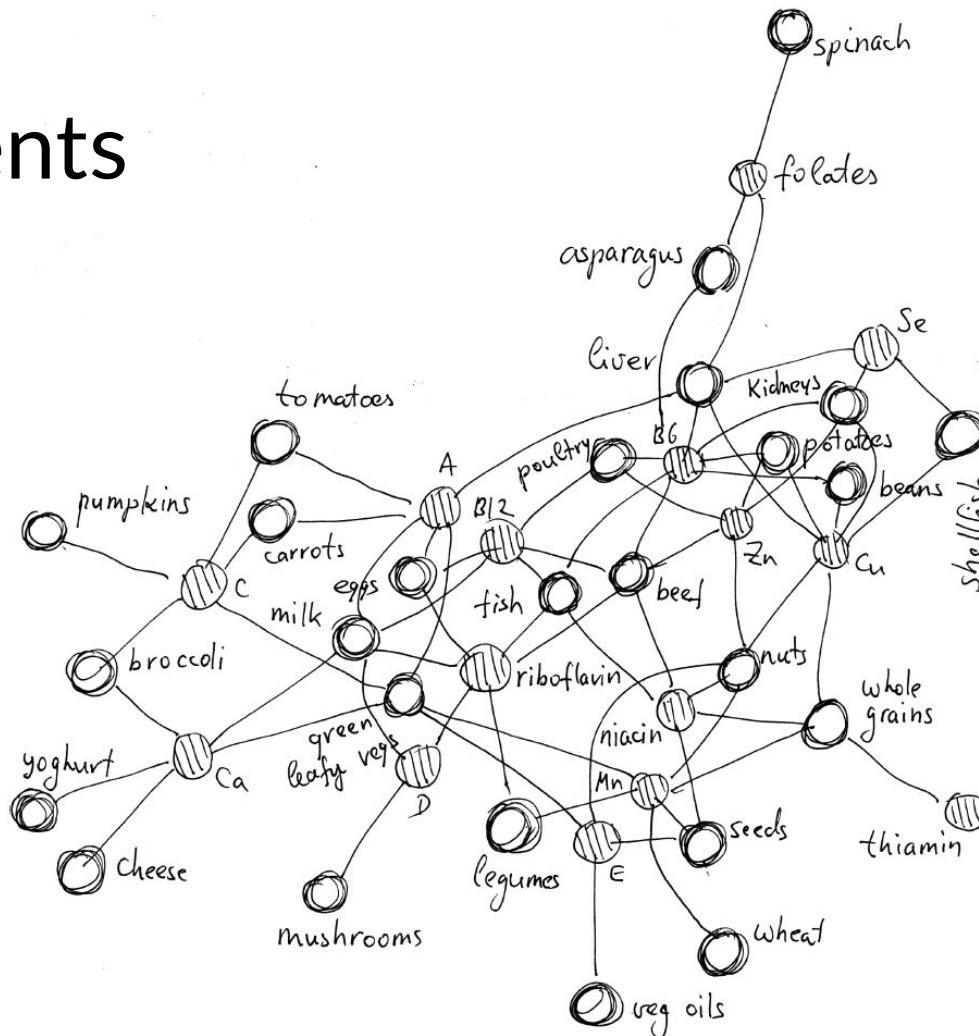


Game of Thrones character relationship

<http://bit.do/gotchar>



Food & Nutrients Network



Contact

[Mailing list](#)

[Issue tracker](#)

[Source](#)



NetworkX

Network Analysis in Python

<https://networkx.github.io/>

Releases

Stable [\(notes\)](#)

2.5 — August 2020

[download](#) | [doc](#) | [pdf](#)

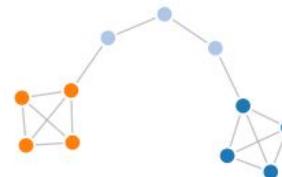
Latest [\(notes\)](#)

2.6 development

[github](#) | [doc](#) | [pdf](#)

[Archive](#)

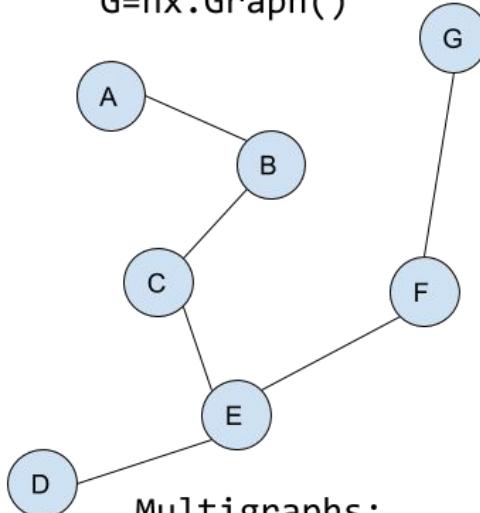
NetworkX is a Python package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks.



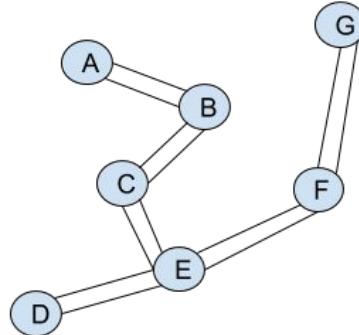
Software for complex networks

- Data structures for graphs, digraphs, and multigraphs
- Many standard graph algorithms
- Network structure and analysis measures
- Generators for classic graphs, random graphs, and synthetic networks
- Nodes can be "anything" (e.g., text, images, XML records)
- Edges can hold arbitrary data (e.g., weights, time-series)
- Open source [3-clause BSD license](#)
- Well tested with over 90% code coverage
- Additional benefits from Python include fast prototyping, easy to teach, and multi-platform

Undirected:
G=nx.Graph()

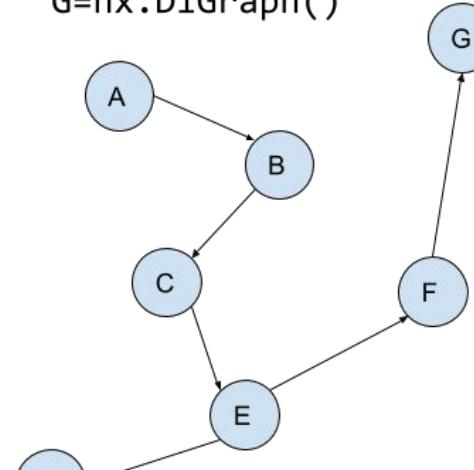


Multigraphs:
G=nx.MultiGraph()

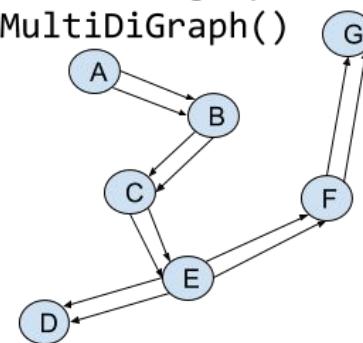


```
import networkx as nx  
nx.__version__  
2.5
```

Directed:
G=nx.DiGraph()



Directed Multigraphs:
G=nx.MultiDiGraph()



Add nodes and edges

```
G = nx.Graph([("A", "eggs")])  
  
G.add_node("spinach") # Add a single node  
G.add_node("Hg") # Add a single node by mistake  
G.add_nodes_from(["folates", "asparagus", "liver"]) # Add a list of nodes  
  
G.add_edge("spinach", "folates") # Add one edge, both ends exist  
G.add_edge("spinach", "heating oil") # Add one edge by mistake  
G.add_edge("liver", "Se") # Add one edge, one end does not exist  
G.add_edges_from([('folates', 'liver'), ('folates', 'asparagus')])
```

Remove nodes and edges

```
G.remove_node("Hg")
G.remove_nodes_from(["Hg"]) # Safe to remove a missing node using a list
G.remove_edge("spinach", "heating oil")
G.remove_edges_from([('spinach', "heating oil")]) # See above
G.remove_node("heating oil") # Not removed yet
```

Look at Nodes list

```
G.nodes
```

```
NodeView(( 'A' , 'eggs' , 'spinach' , 'folates' , 'asparagus' , 'liver' , 'Se' ))
```

```
G.nodes(data=True)
```

```
NodeDataView({ 'A': {} , 'eggs': {} , 'spinach': {} , 'folates': {} ,  
             'asparagus': {} , 'liver': {} , 'Se': {} })
```

```
G.nodes[ "A" ]
```

```
{}
```

Look at Edges list

```
G.edges  
EdgeView([('A', 'eggs'), ('spinach', 'folates'), ('folates', 'liver'),  
         ('folates', 'asparagus'), ('liver', 'Se')))  
  
G.edges(data=True)  
EdgeDataView([('A', 'eggs', {}), ('spinach', 'folates', {}),  
             ('folates', 'liver', {}), ('folates', 'asparagus', {}),  
             ('liver', 'Se', {})])  
  
G.edges["A", "eggs"]  
{}  
  
G["folates"]  
AtlasView({'spinach': {}, 'liver': {}, 'asparagus': {}})
```



Up to section 2.2



Add attributes

```
G.add_node("Honey", edible=True)
G.add_nodes_from([("Steel", {"edible" : False})])

G.add_edge("Honey", "Steel", weight=0.0)
G.add_edges_from([("Honey", "Zn")], related=False)
```

```
G.nodes["Honey"]
{'edible': True}

G.nodes["Steel"]
{'edible': False}

G.edges["Honey","Steel"]
{'weight': 0.0}

G.edges["Honey","Zn"]
{'related': False}
```

G.add_edges_from() vs G.add_weighted_edges_from()

```
G.add_edges_from([("Honey", "Zn")], related=False)
```

```
G.add_weighted_edges_from([( "Honey", "Zn", 0.01),  
                           ( "Honey", "Sugar", 0.99)])
```

```
G.edges["Honey", "Zn"]  
{'related': False, 'weight': 0.01}
```

```
G.edges["Honey", "Sugar"]  
{'weight': 0.99}
```

Update and remove attributes

```
G.nodes["Zn"]["nutrient"] = True # Zinc is a nutrient
G.nodes["Zn"]
{'nutrient': True}

G.edges["Zn","Beef"]["weight"] = 0.95 # Zinc and beef are well connected
G.edges["Zn","Beef"]
{'weight': 0.95}

del G.nodes["Zn"]["nutrient"]
G.nodes["Zn"]
{}

del G.edges["Zn","Beef"]["weight"]
G.edges["Zn","Beef"]
{}
```

Define or change an attribute of existing nodes and edges by calling `nx.set_node_attributes()` or `nx.set_edge_attributes()`:

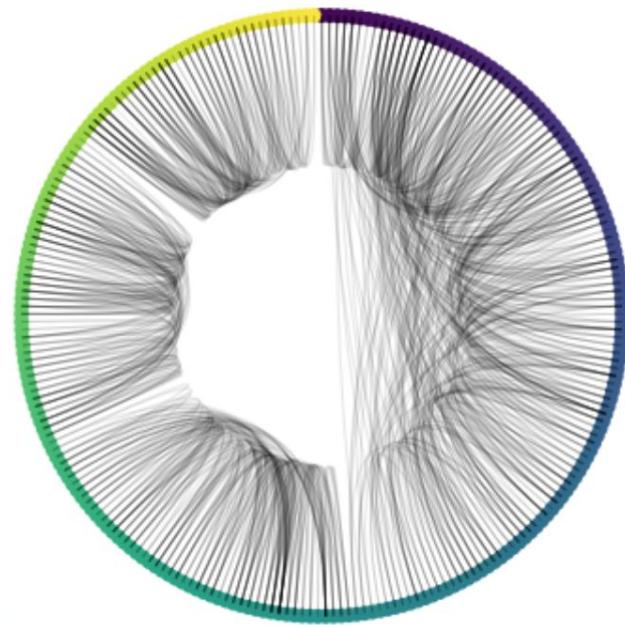
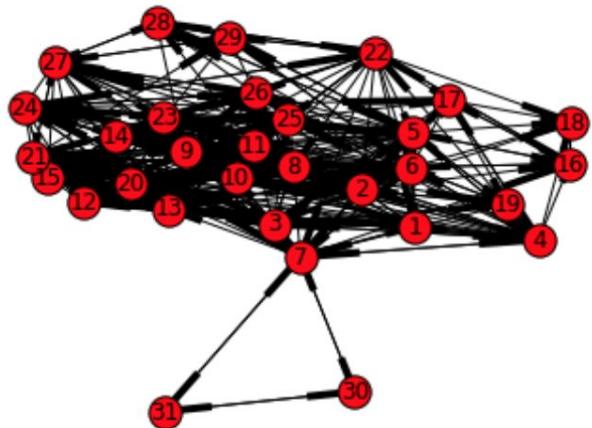
```
nutrients = set(("B12", "Zn", "D", "B6",
                  "A", "Se", "Cu", "Folates",
                  "Ca", "Mn", "Thiamin", "Riboflavin",
                  "C", "E", "Niacin", "Starch", "Mg", "Na"))

nutrient_dict = {node: (node in nutrients) for node in G}
nx.set_node_attributes(G, nutrient_dict, "nutrient")

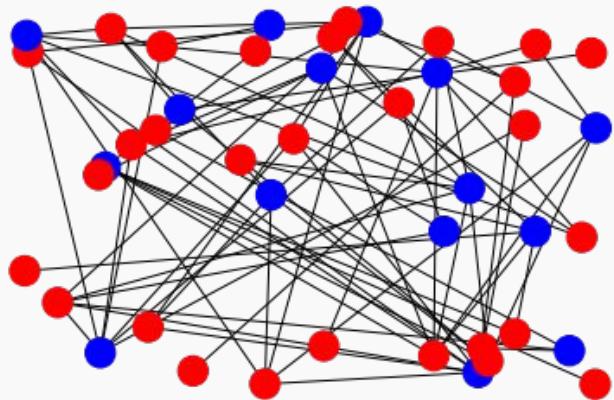
G.nodes(data=True)
NodeDataView({'A': {'nutrient': True}, 'B12': {'nutrient': True},
             'B6': {'nutrient': True}, 'C': {'nutrient': True}, ...})
```



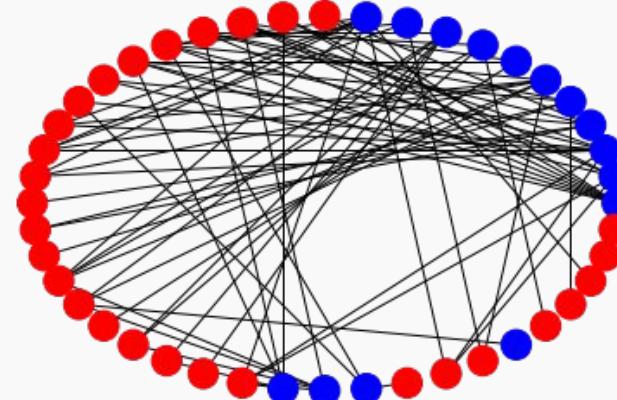
Irrational vs Rational Visualization



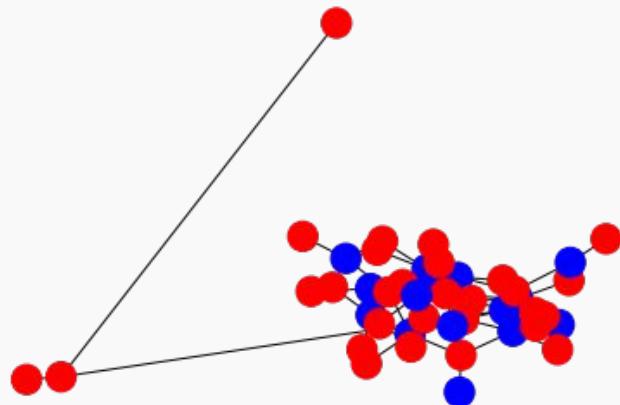
Random



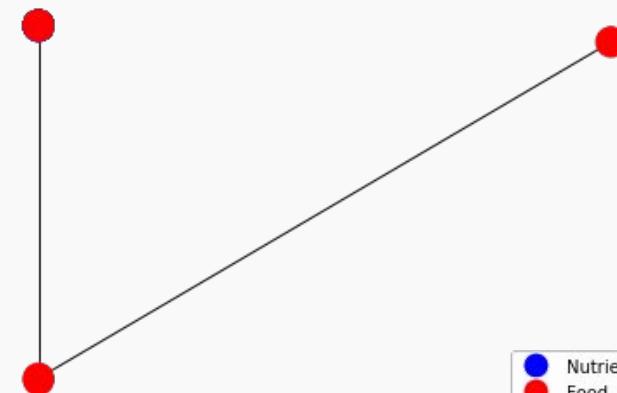
Circular



Spring



Spectral



Nutrient
Food

```
import matplotlib.pyplot as plt

# Prepare for drawing
colors = [ "blue" if n[1]["nutrient"] else "red" for n in G.nodes(data=True) ]

# Create 4 subplots
fig, plot = plt.subplots(2, 2, figsize=(15,10))
subplots = plot.reshape(1, 4)[0]

# Configure the layouts and title of subplots
layouts = (nx.random_layout, nx.circular_layout,
            nx.spring_layout, nx.spectral_layout)
titles = ("Random", "Circular", "Spring", "Spectral")
```

```
# Main loop
for plot, layout, title in zip(subplots, layouts, titles):
    pos = layout(G)
    nx.draw_networkx(G, pos=pos, ax=plot, with_labels=False, node_color=colors)
    plot.set_title(title)

plt.show()
```

Code

Issues 22

Pull requests 3

Projects 2

Wiki

Insights

Visualization Package for NetworkX <http://nxviz.readthedocs.io/>

[network-visualization](#)[networkx](#)[network](#)[visualization](#)

1,461 commits

9 branches

3 releases

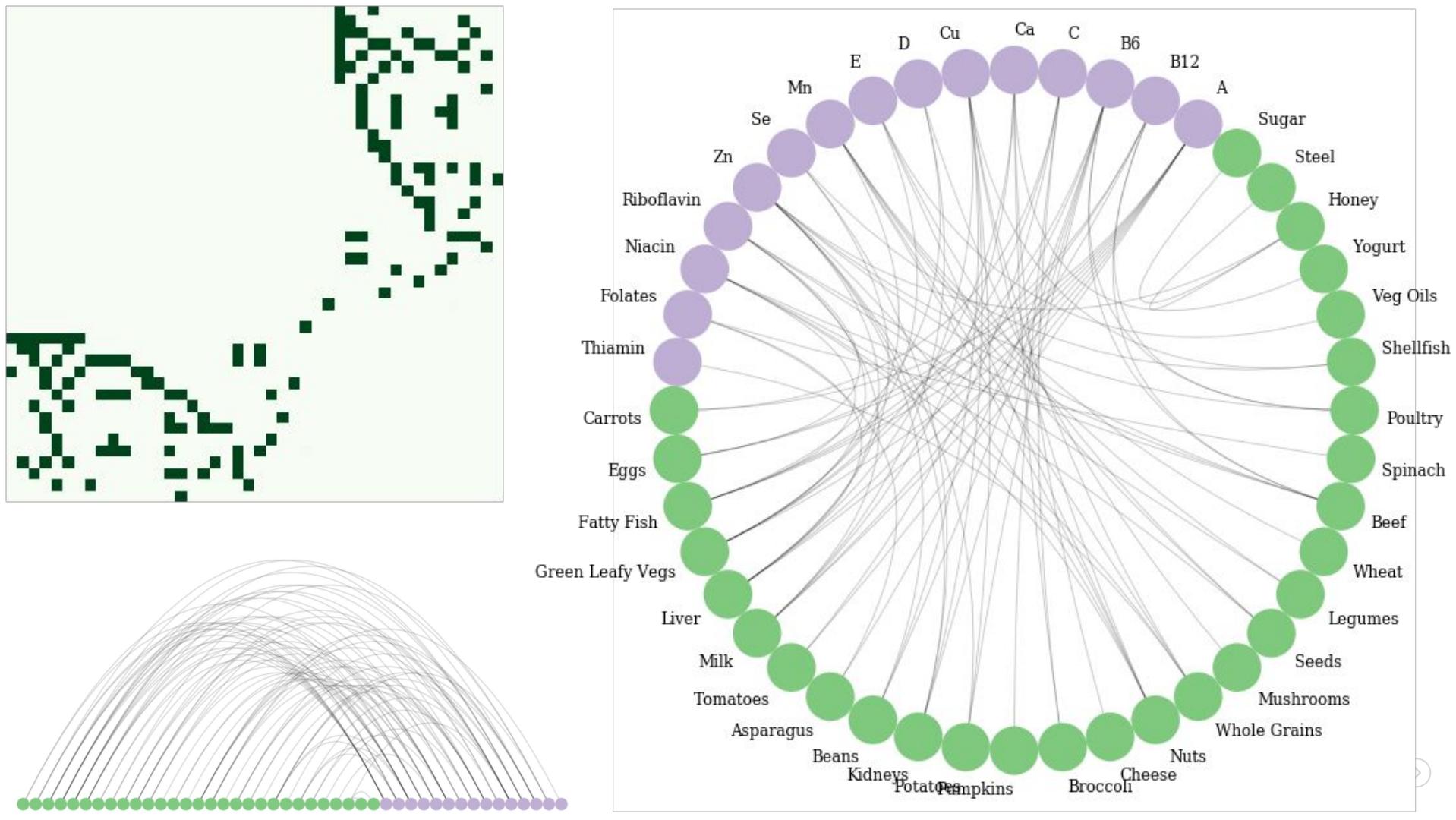
14 contributors

MIT

Branch: master ▾

[New pull request](#)[Create new file](#)[Upload files](#)[Find File](#)[Clone or download ▾](#)

ericmjl	Merge pull request #547 from ericmjl/test-health-check-fixes	...	Latest commit 6ea5823 8 days ago
.github	Create pull_request_template.md		5 months ago
docs	docfixes		3 months ago
examples	applied black		8 days ago
nxviz	fix pycodestyle issue		8 days ago
tests	black formatting		8 days ago
.bumpversion.cfg	bumpversion 0.6.1		11 days ago
.editorconfig	Initial skeleton.		3 years ago
.gitignore	Include .DS_Store		3 years ago



```

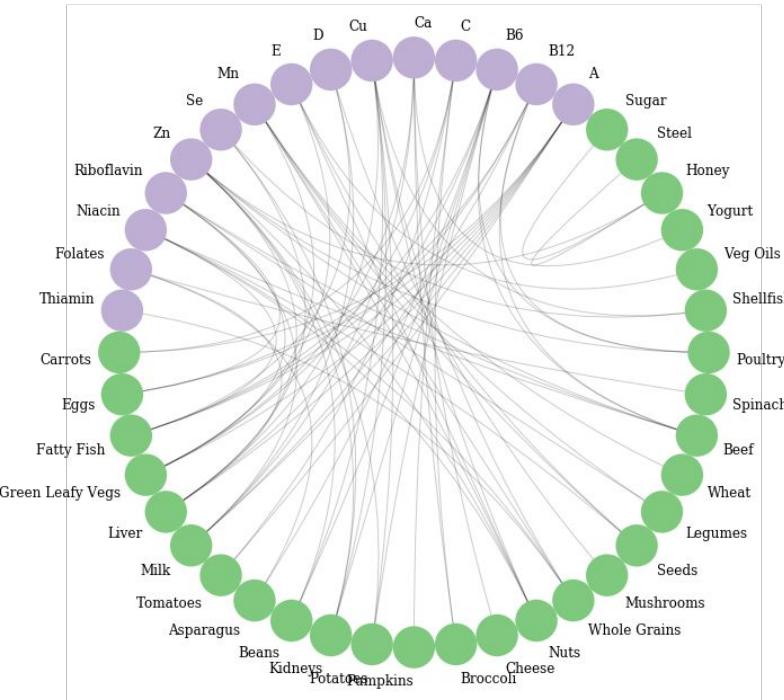
# Import necessary modules to use Circos plot
import matplotlib.pyplot as plt
from nxviz import CircosPlot

# Create the CircosPlot object: c
c = CircosPlot(G, node_color='nutrient',
                node_grouping='nutrient',
                node_order="nutrient",
                node_labels=True,
                figsize = (10,10),
                nodeprops={"radius": 0.5},
                #node_label_layout="rotation",
                #node_label_layout="numbers",
                fontsize=12
            )

# Draw c to the screen
c.draw()
c.figure.tight_layout()

# Display the plot
plt.show()

```



```

NodeDataView({'A': {'nutrient': True},
             'B12': {'nutrient': True},
             'B6': {'nutrient': True},
             'C': {'nutrient': True},
             ....
            })

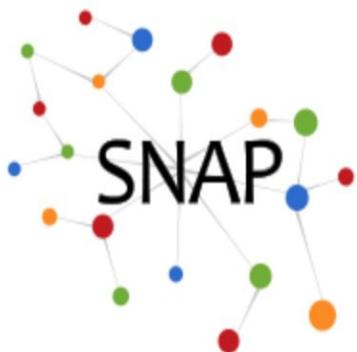
```

Update and remove attributes

Format	Attributes	Reader	Writer	Supported by Gephi?
Adjacency list	Not stored	nx.read_adjlist()	nx.write_adjlist()	Yes
Edge list	Not stored	nx.read_edgelist()	nx_write_edgelist()	Yes
Graph exchange XML format	Stored	nx.read_gexf()	nx.write_gexf()	Yes
Graph modeling language	Stored	nx.read_gml()	nx.write_gml()	w/o attributes
GraphML	Stored	nx.read_graphml()	nx.write_graphml()	Yes
Pajek NET	Not stored	nx.read_pajek()	nx.write_pajek()	Yes
Pickle	Stored	nx.read_gpickle()	nx.write_gpickle()	No
YAML	Stored	nx.read_yaml()	nx.write_yaml()	No

`nx.write_graphml(G, "nutrients.graphml")`





Stanford Large Network Dataset Collection

- [Social networks](#) : online social networks, edges represent interactions between people
- [Networks with ground-truth communities](#) : ground-truth network communities in social and information networks
- [Communication networks](#) : email communication networks with edges representing communication
- [Citation networks](#) : nodes represent papers, edges represent citations
- [Collaboration networks](#) : nodes represent scientists, edges represent collaborations (co-authoring a paper)
- [Web graphs](#) : nodes represent webpages and edges are hyperlinks
- [Amazon networks](#) : nodes represent products and edges link commonly co-purchased products
- [Internet networks](#) : nodes represent computers and edges communication
- [Road networks](#) : nodes represent intersections and edges roads connecting the intersections
- [Autonomous systems](#) : graphs of the internet
- [Signed networks](#) : networks with positive and negative edges (friend/foe, trust/distrust)
- [Location-based online social networks](#) : Social networks with geographic check-ins
- [Wikipedia networks, articles, and metadata](#) : Talk, editing, voting, and article data from Wikipedia
- [Temporal networks](#) : networks where edges have timestamps
- [Twitter and Memetracker](#) : Memetracker phrases, links and 467 million Tweets
- [Online communities](#) : Data from online communities such as Reddit and Flickr
- [Online reviews](#) : Data from online review systems such as BeerAdvocate and Amazon

SNAP networks are also available from [SuiteSparse Matrix Collection](#) by [Tim Davis](#).

SNAP for C++ ▶
SNAP for Python ▶
SNAP Datasets ▶
BIOSNAP Datasets
What's new
People
Papers
Projects ▶
Citing SNAP
Links
About
Contact us

 **Tiago Peixoto**
@tiagopeixoto

Physics & Networks.

 social.skewed.de/@tiago

 Wien, Österreich  skewed.de/tiago  Joined December 2008

5,001 Following 3,362 Followers

 Followed by Yan Holtz, Kirk Borne, and 2 others you follow

[Tweets](#) [Tweets & replies](#) [Media](#) [Likes](#)

 Pinned Tweet

 **Tiago Peixoto** @tiagopeixoto · Sep 17
I'm happy to announce a new project, called Netzschleuder:
networks.skewed.de ✓

How do I download a copy of my information on Facebook?

<https://www.facebook.com/help/212802592074644>

Lesson 07 - Complex Network Analysis in Python.ipynb