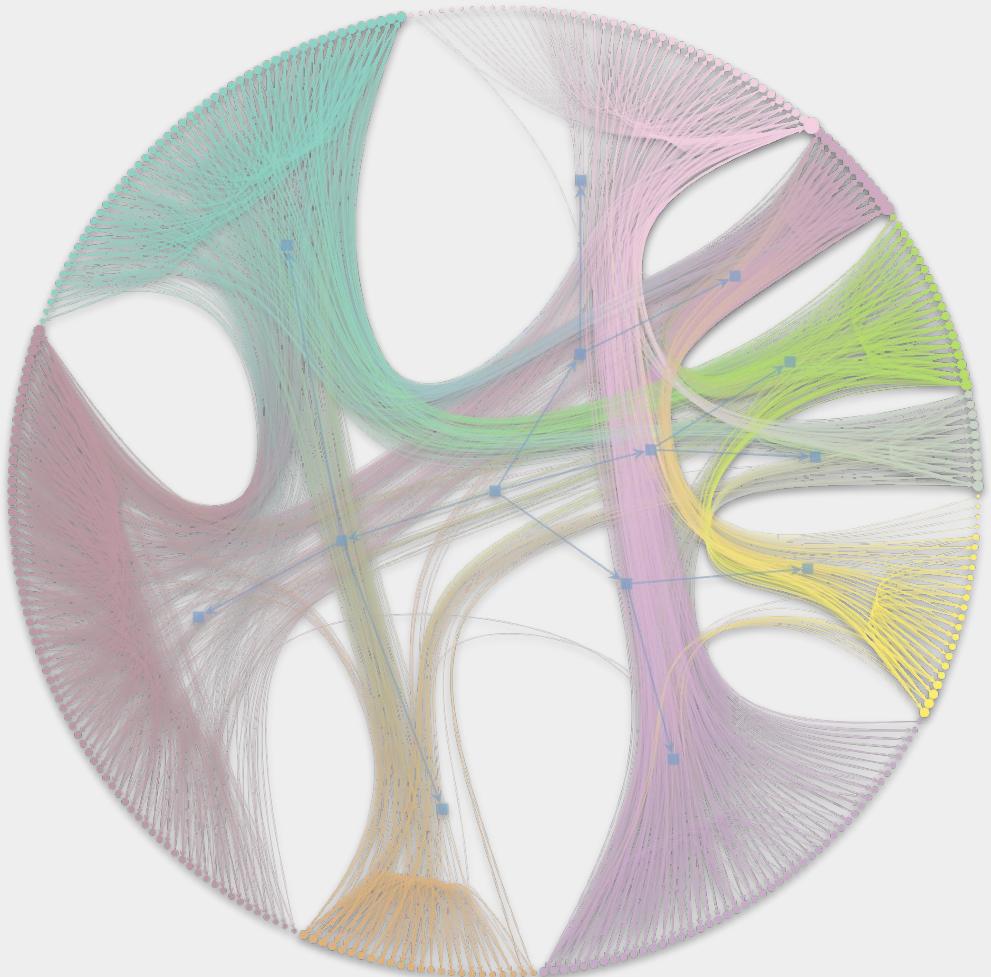


Network Elements

Part 01

ivanovitch.silva@ufrn.br
@ivanovitchm



Conhecendo

- 📚 conhecendo
- 👂 ouvindo
- 📖 lendo
- 📝 escrevendo/anotando
- 🎓 estudando

⟨conteúdo e objetos de aprendizagem⟩

Descobrindo

- 🔍 pesquisando
- 👀 observando
- 🎩 investigando
- 🧪 testando

⟨aprendizagem ativa e curadoria⟩

aqui a
aprendizagem
acontece

Partilhando

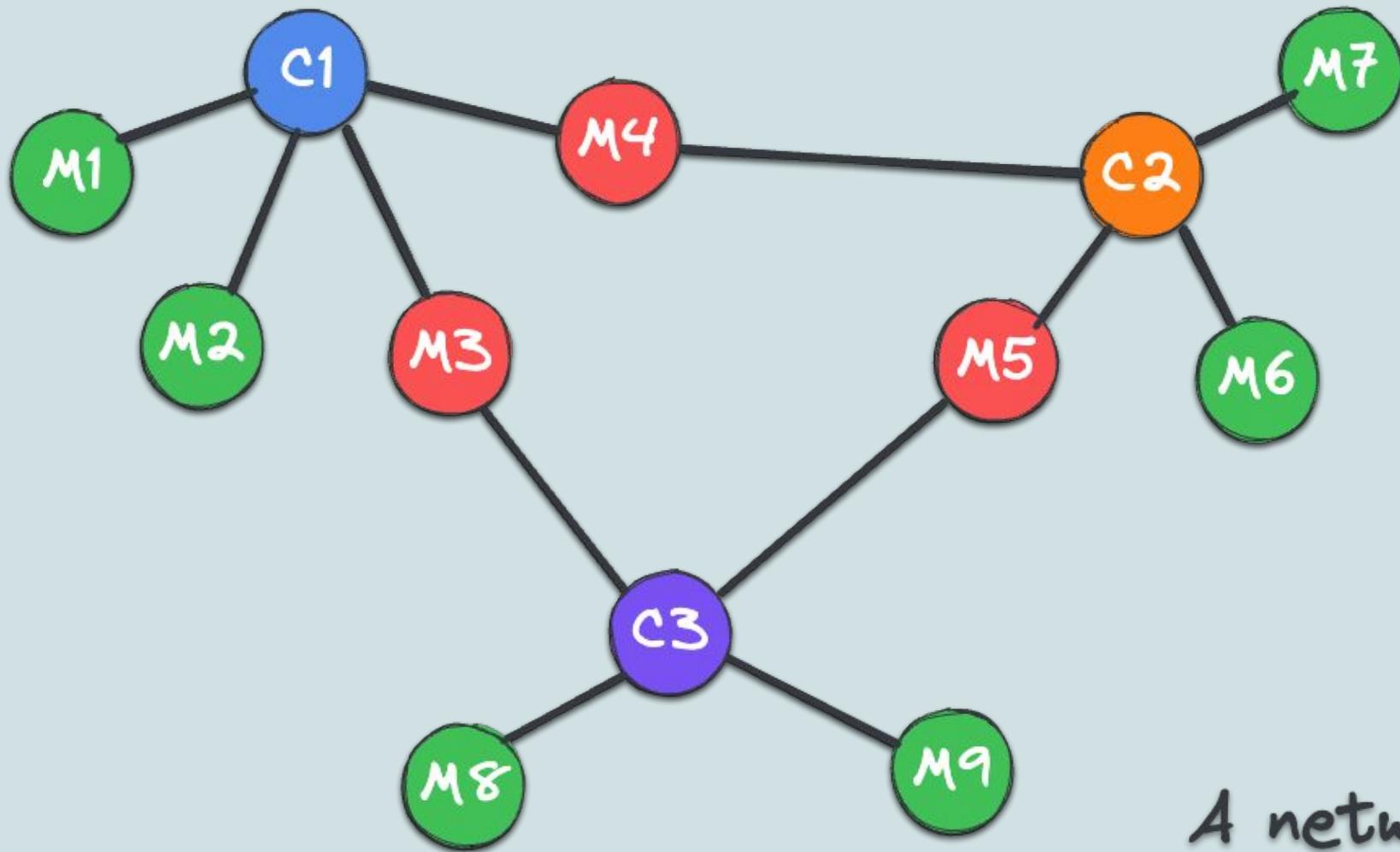
- 💬 dialogando
- ☁️ debatendo
- 💡 discutindo
- 👥 explicando
- 🎓 ensinando
- 🤝 trocando

⟨espaços, plataformas, redes e comunidades⟩

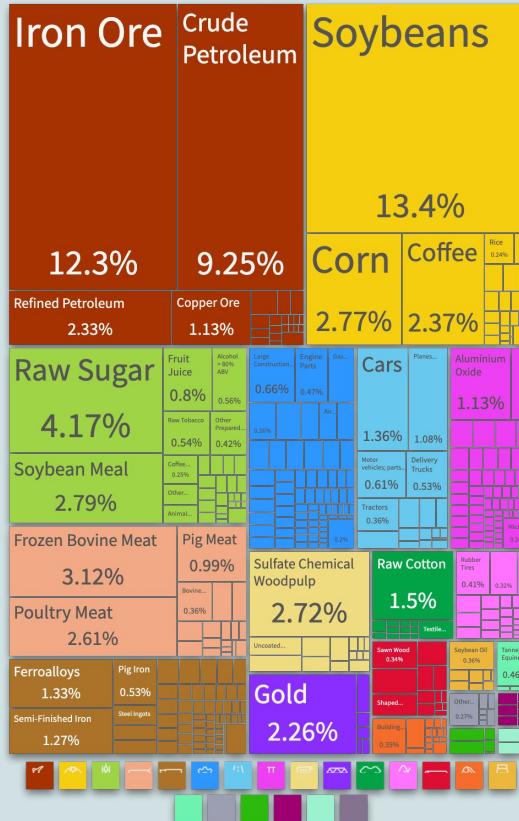
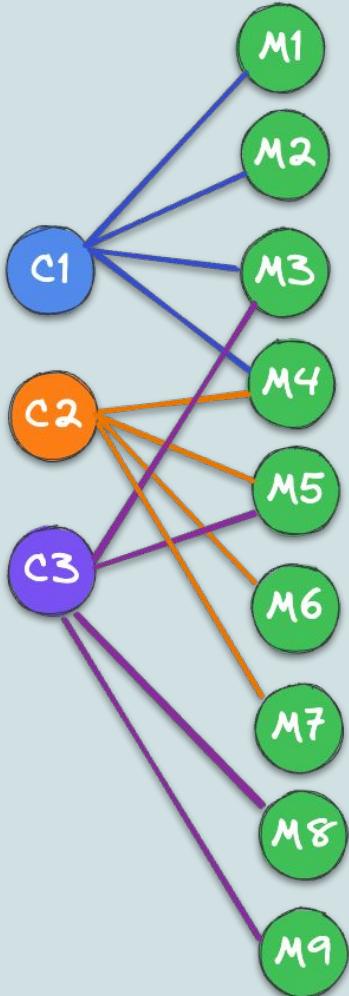
Experienciando

- 🤸 praticando
- 🤔 dando sentido
- 🌟 vivenciando
- 🎨 criando
- 😍 sentindo
- 🧩 conectando os pontos



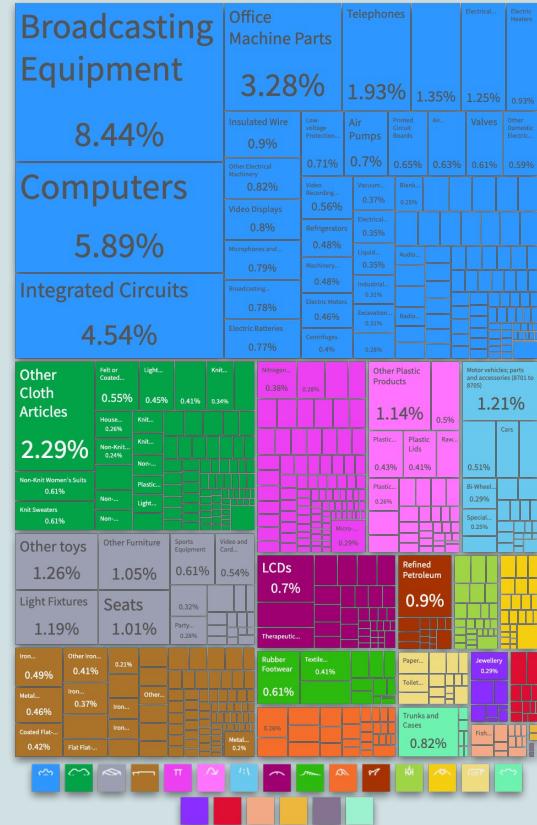


A network

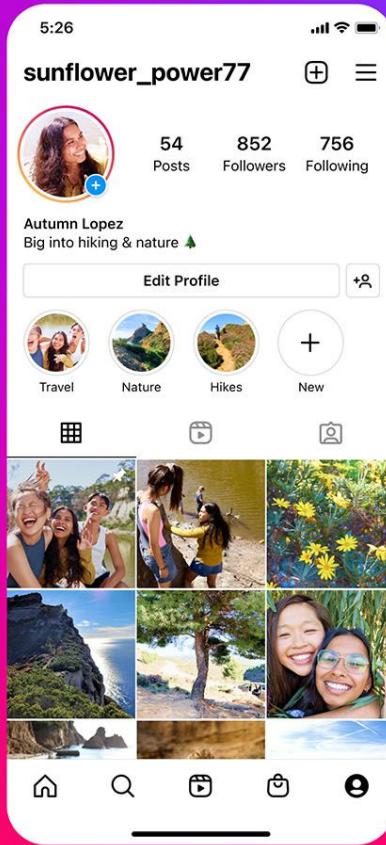
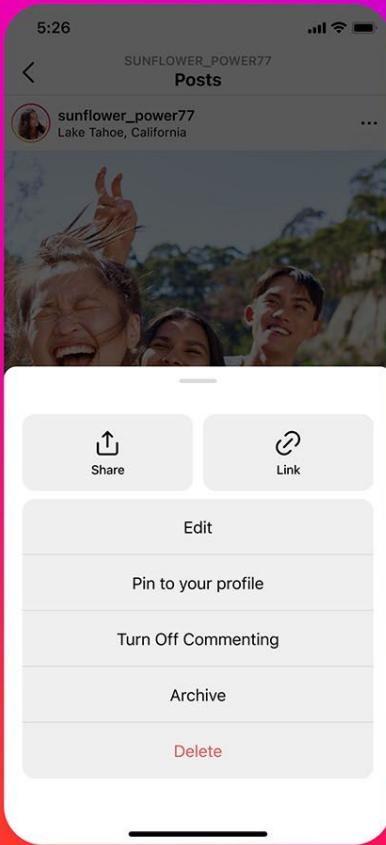


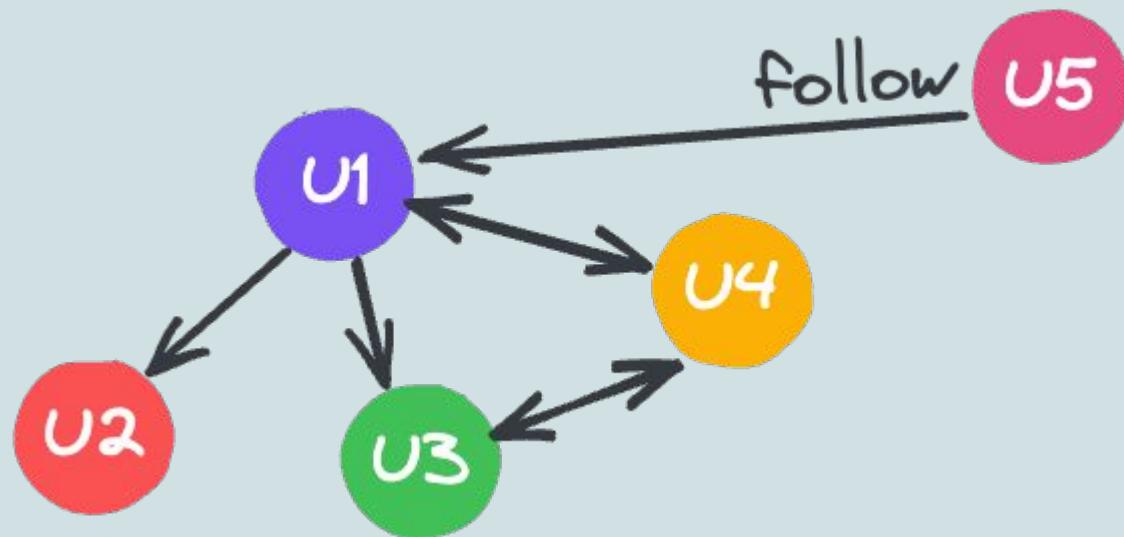
bipartite network

Exports (2020)
[Click to Select a Product]
Total: \$2.65T



<https://oec.world/en/profile/country/chn>



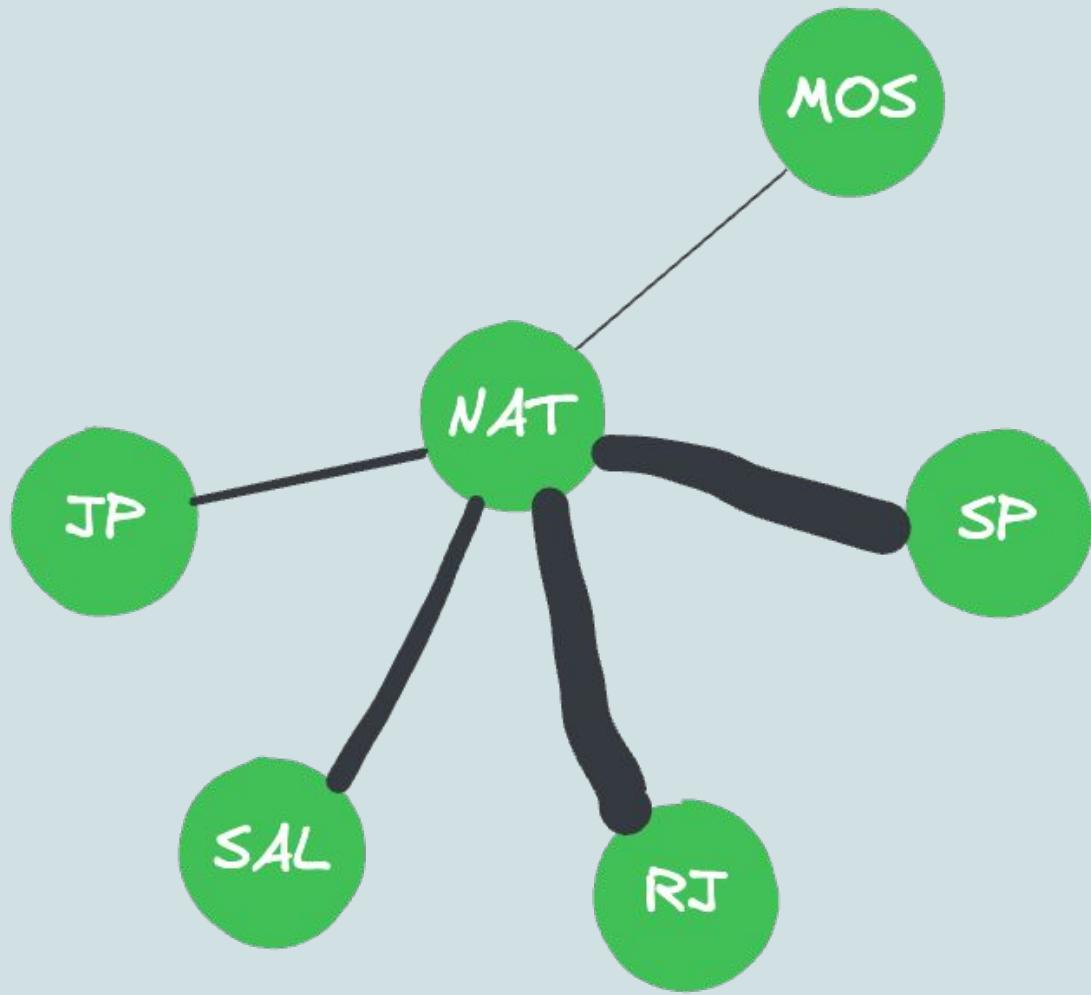


leomessi ✅ Following Message + ...

1,025 posts 457M followers 292 following

Leo Messi
Bienvenidos a la cuenta oficial de Instagram de Leo Messi / Welcome to the official Leo Messi Instagram account
themessistore.com





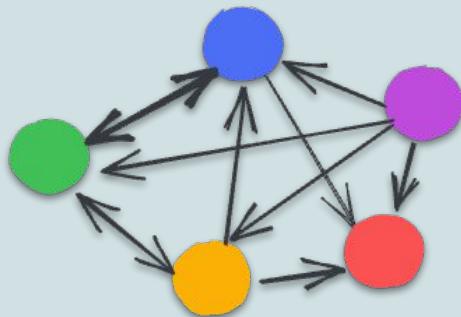


WhatsApp

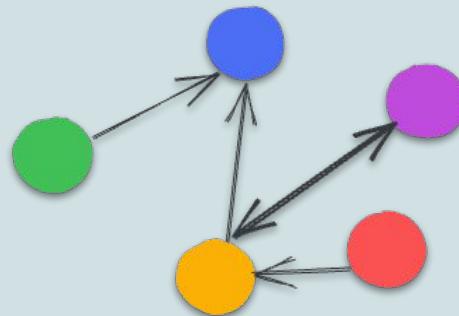


<https://github.com/kurasaitja/Whatsapp-Analysis>

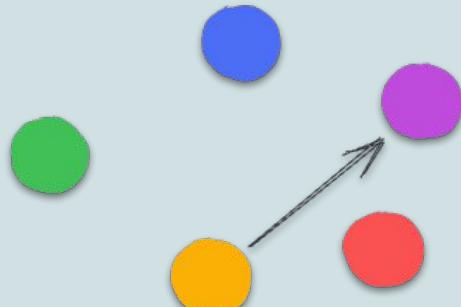
Monday



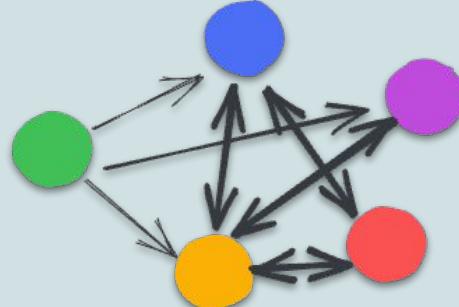
Tuesday



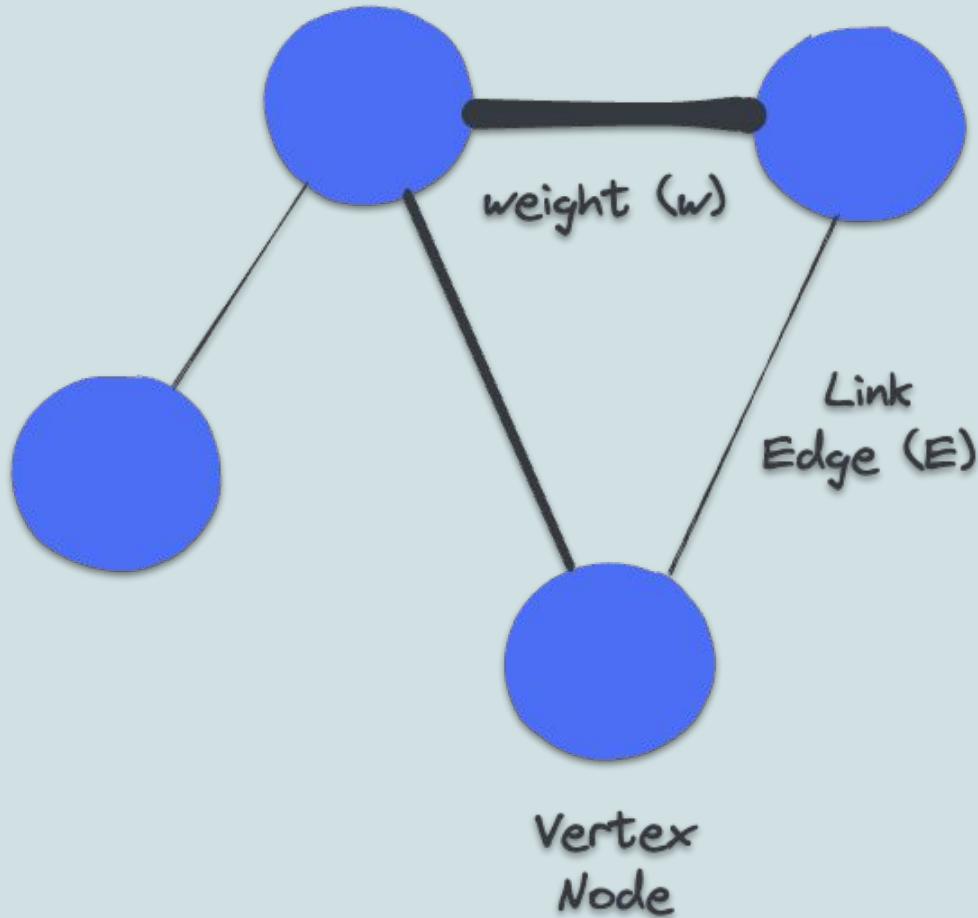
Wednesday



Thursday



Temporal network



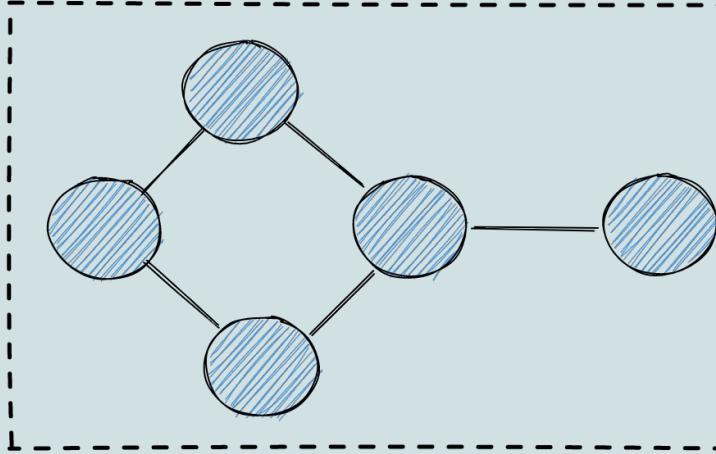
Basic Definitions

$$G = (V, E, W)$$

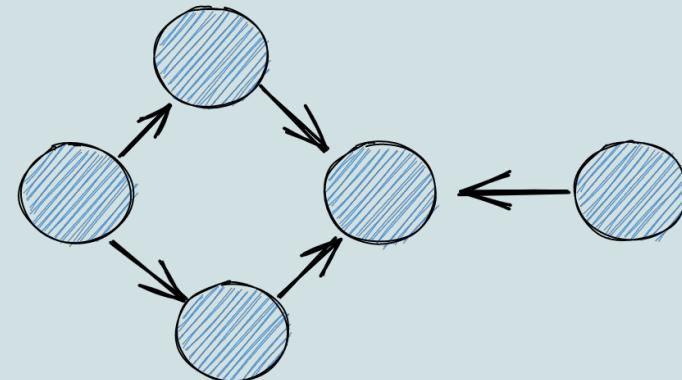
$$E \subseteq V \times V$$

$$W \subseteq R^+$$

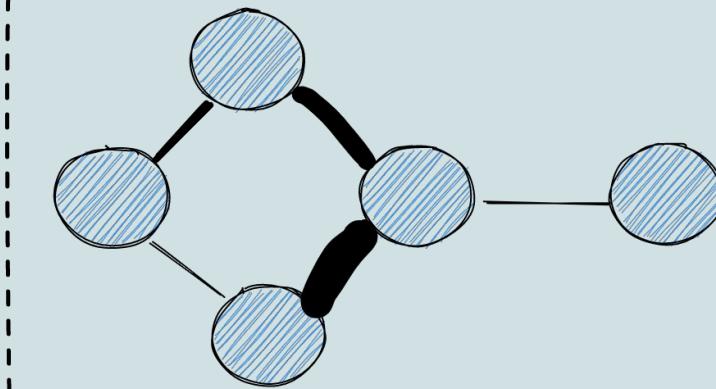
Undirected



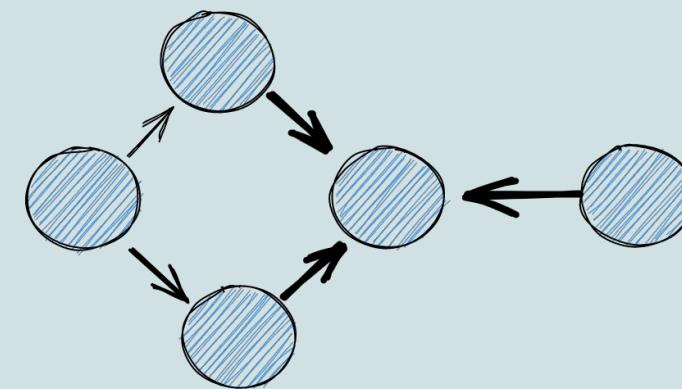
Directed



Unweighted



Weighted



Other Fundamental Properties

Density and Sparsity

Subnetwork

Degree

Network Representation



Further Reading



1 *Introduction* 9

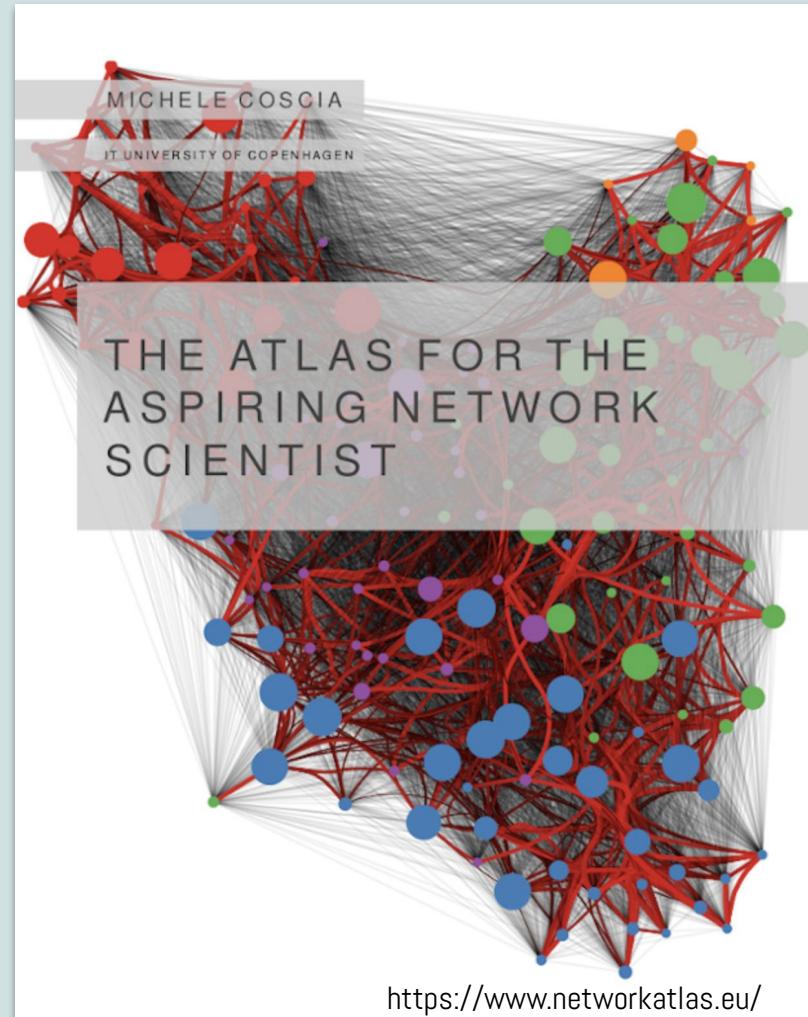
I *Basics* 21

2 *Probability Theory* 22

3 *Basic Graphs* 40

4 *Extended Graphs* 48

5 *Matrices* 68





Contact

[Mailing list](#)

[Issue tracker](#)

[Source](#)

Releases

[Stable \(notes\)](#)

3.1 — April 2023

[Documentation](#)

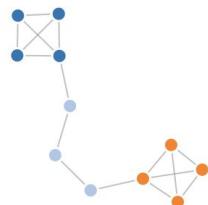
[Latest \(notes\)](#)

3.2 development

[Documentation](#)

[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



What is graph-tool?

Graph-tool is an efficient [Python](#) module for manipulation and statistical analysis of [graphs](#) (a.k.a. [networks](#)). Contrary to most other Python modules with similar functionality, the core data structures and algorithms are implemented in [C++](#), making extensive use of [template metaprogramming](#), based heavily on the [Boost Graph Library](#). This confers it a level of [performance](#) that is comparable (both in memory usage and computation time) to that of a pure C/C++ library.

► It is *Fast!*

Despite its nice, soft outer appearance of a regular Python module, the core algorithms and data structures of graph-tool are written in C++, with performance in mind. Most of the time, you can expect the algorithms to run just as fast as if graph-tool were a pure C/C++ library. See a [performance comparison](#).

OpenMP Support

Many algorithms are implemented in parallel using [OpenMP](#), which provides excellent performance on multi-core architectures, without degrading it on single-core machines.

⌚ Extensive Features

An extensive array of features is included, such as support for arbitrary vertex, edge or graph [properties](#), efficient "on the fly" [filtering](#) of vertices and edges, powerful graph I/O using the [GraphML](#), [GML](#) and [dot](#) file formats, graph [pickling](#), [graph statistics](#) (degree/property histogram, vertex correlations, average shortest distance, etc.), [centrality measures](#), standard [topological algorithms](#) (isomorphism, minimum spanning tree, connected components, dominator tree, [maximum flow](#), etc.), [generation of random graphs](#) with arbitrary degrees and correlations, [detection of modules and communities](#) via statistical inference, and much more.

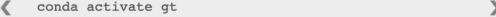


Download version 2.54

[Installation instructions](#) | [Changelog](#)

Conda installation (GNU/Linux | MacOS)

```
conda create --name gt -c conda-forge graph-tool  
conda activate gt
```



👁 Powerful Visualization

Conveniently [draw](#) your graphs, using a variety of algorithms and output formats (including to the screen). Graph-tool has its own layout algorithms and versatile, interactive drawing routines based on [cairo](#) and [GTK+](#), but it can also work as a very comfortable interface to the excellent [graphviz](#) package.

✍ Fully Documented

Every single function in the module is documented in the docstrings and in the online [documentation](#), which is full of examples.