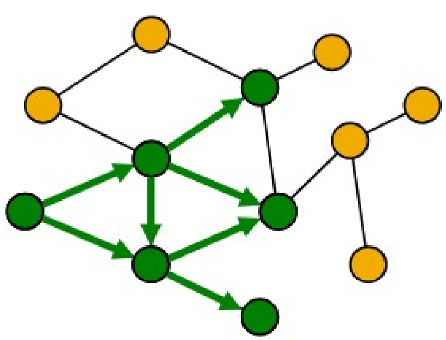
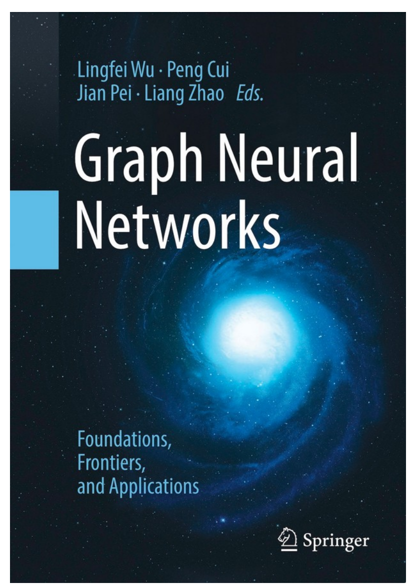
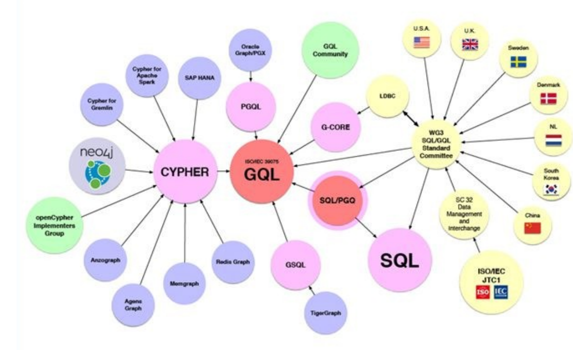
Other Topics
(wrapping up the NetSci course)

Pedro Ribeiro
(DCC/FCUP & CRACS/INESC-TEC)







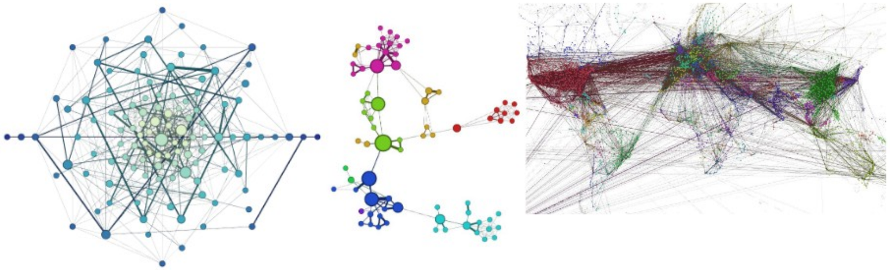
Remembering Our Classes

An Introduction To Network Science

U.PORTO
FC FACULDADE DE CIÊNCIAS UNIVERSIDADE DO PORTO

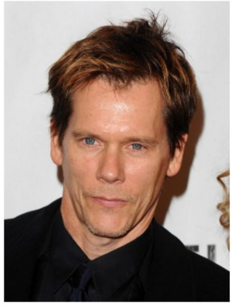
Pedro Ribeiro
(DCC/FCUP & CRACS/INESC-TEC)

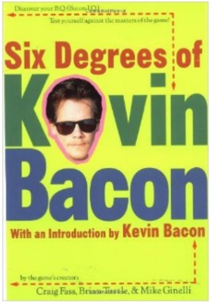
CRACS INESCTEC



More examples of "Small World"

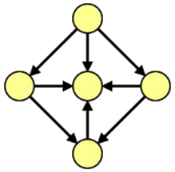
- The six degrees of Kevin Bacon
  - How many connections to link Kevin Bacon to any other actor, director, producer...
  - "Game" initiated in 1994



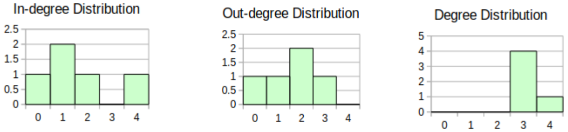


Node Properties

- **Degree related metrics:**
  - **Degree sequence**
    an ordered list of the (in,out) degree of each node
    - In-degree sequence: [4, 2, 1 , 1, 0]
    - Out-degree sequence: [3, 2, 2, 1, 0]
    - Degree sequence: [4, 3, 3, 3, 3]

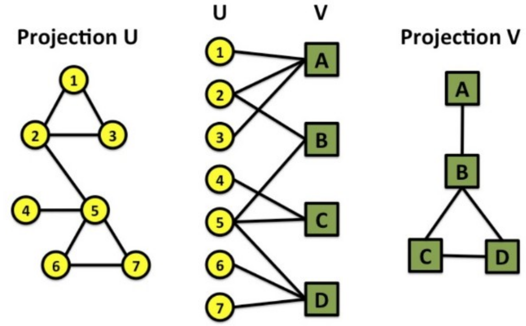


  - **Degree Distribution**
    a frequency count of the occurrences of each degree
    [usually plotted as probability → normalization]



Bipartite

- **A bipartite graph is a graph whose nodes can be divided into two disjoint sets U and V such that every edge connects a node in U to one in V.**



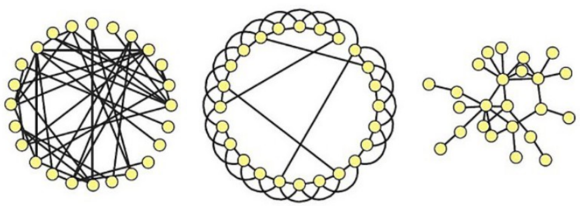
Example:
- Actor Network
  . U = Actor
  . V = Movies

Image: Adapted from Leskovec, 2015

2 - Measuring Networks and Random Graph Models

Measuring Networks and Random Graph Models

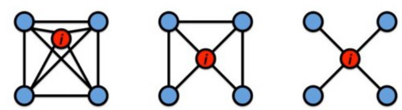
Pedro Ribeiro
(DCC/FCUP & CRACS/INESC-TEC)



(Heavily based on slides from Jure Leskovec and Lada Adamic@ Stanford University - CS224W)

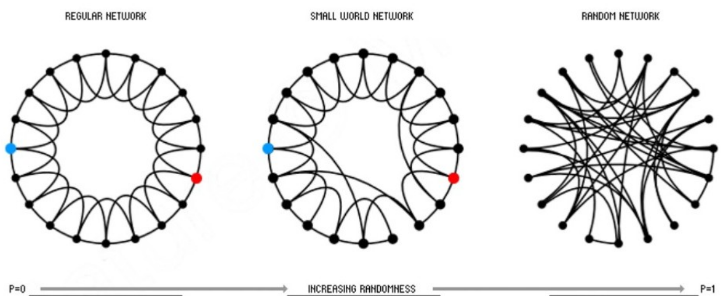
(3) Clustering Coefficient

- **Clustering coefficient**:
  - What portion of ***i***'s neighbors are connected?
  - Node ***i*** with degree $\boldsymbol{k_i}$
  - $C_i \in [0,1]$
  - $C_i = \frac{2e_i}{k_i(k_i-1)}$ where e_i is the number of edges between the neighbors of node i



- **Average clustering coefficient:** $C = \frac{1}{N}\sum_i^n C_i$

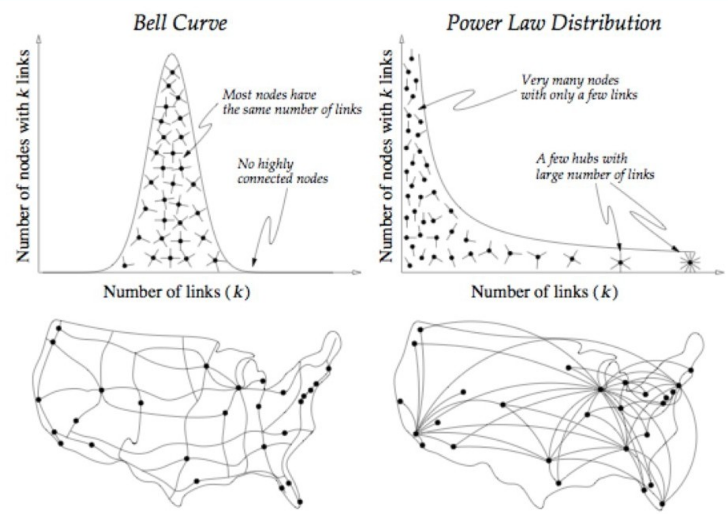
The Small World Model



High clustering High diameter	High clustering Low diameter	Low clustering Low diameter
$h = \frac{N}{2\bar{k}}$ $C = \frac{1}{2}$		$h = \frac{\log N}{\log \alpha}$ $C = \frac{\bar{k}}{N}$

Rewiring allows us to "interpolate" between a regular lattice and a random graph

Interpreting Power-Laws

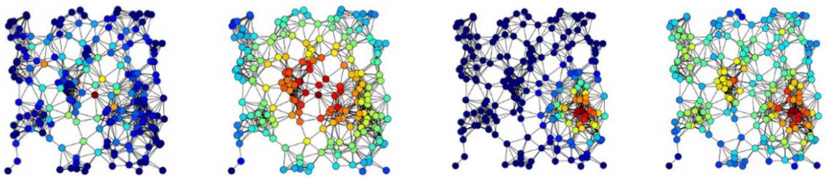


3 – Node Centrality

Node Centrality

Pedro Ribeiro
(DCC/FCUP & CRACS/INESC-TEC)

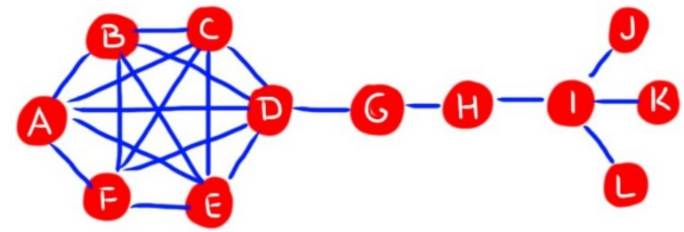
U.PORTO
FC FACULDADE DE CIÊNCIAS UNIVERSIDADE DO PORTO



(Heavily based on slides from Jure Leskovec and Lada Adamic @ Stanford University)

Betweenness: Question

- Find a node that has **high betweenness** but **low degree**



Closeness: Definition

- **Closeness** is based on the **length of the average shortest path** between a node and all other nodes in the network

Closeness Centrality:

$$C_C(i) = \frac{1}{\sum_{j=1}^{N} d(i,j)}$$

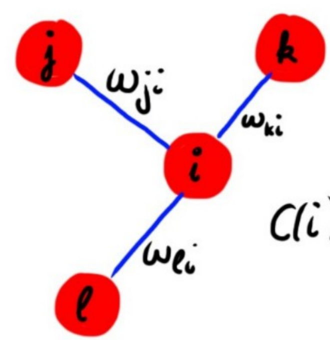
Normalized Closeness Centrality:

$$C'_C(i) = C_C(i) \times (n-1)$$

When graphs are big, the -1 can be discarded and we multiply by *n*

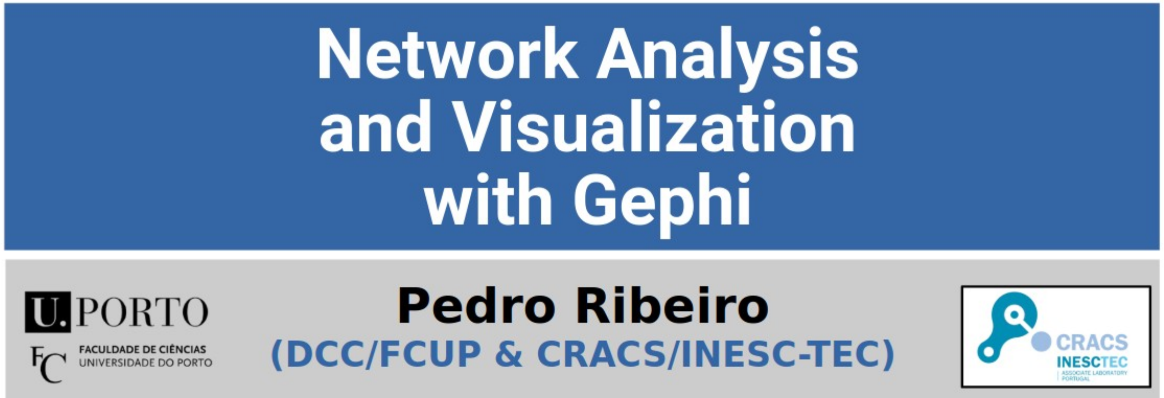
Eigenvector Centrality

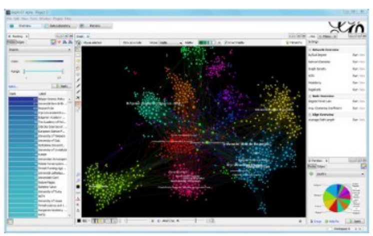
- How "central" you are depends on how "central" your neighbors are

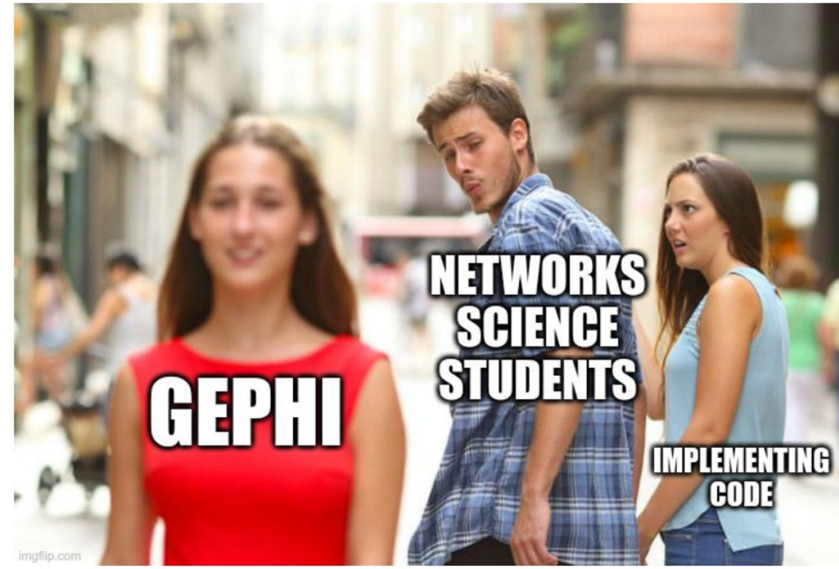


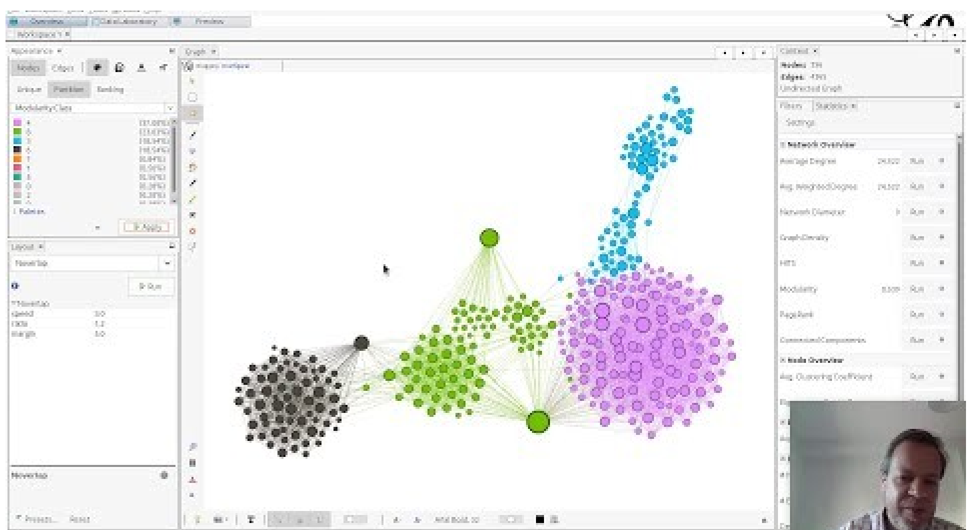
$$C(i) = \omega_{ji} \cdot C(j) + \omega_{ki} \cdot C(k) + \omega_{li} \cdot C(l)$$

4 - Network Analysis and Visualization with Gephi



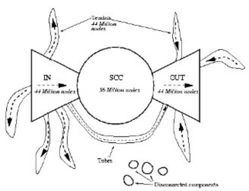


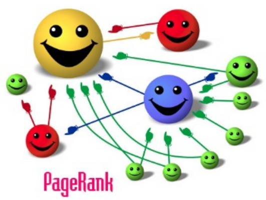




Link Analysis: PageRank

Pedro Ribeiro
(DCC/FCUP & CRACS/INESC-TEC)





(Heavily based on slides from Jure Leskovec and Lada Adamic @ Stanford University)

Structure of the Web

- **Broder et al.**: Altavista web crawl (Oct '99)
  - Web crawl is based on a large set of starting points accumulated over time from various sources, including voluntary submissions.
  - 203 million URLS and 1.5 billion links

Goal: **Take a large snapshot of the Web and try to understand how its SCCs "fit together" as a DAG**

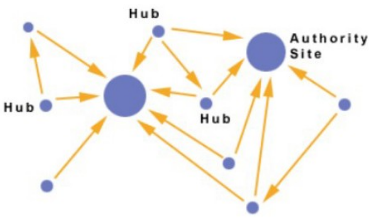
Tomkins, Broder, and Kumar

Hubs and Authorities

Interesting pages fall into two classes:

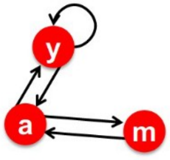
1) Authorities are pages containing useful information
- Newspaper home pages
- Course home pages
- Home pages of auto manufacturers

2) Hubs are pages that link to authorities
- List of newspapers
- Course bulletin
- List of auto manufacturers



PageRank: How to Solve?

- **Power Iteration:**
  - Set $r_j \leftarrow 1/N$
  - **1:** $r'_j \leftarrow \sum_{i \to j} \frac{r_i}{d_i}$
  - **2:** $r \leftarrow r'$
  - If $|r - r'| > \varepsilon$: goto **1**
- **Example:**



	y	a	m
y	½	½	0
a	½	0	1
m	0	½	0

$r_y = r_y/2 + r_a/2$
$r_a = r_y/2 + r_m$
$r_m = r_a/2$

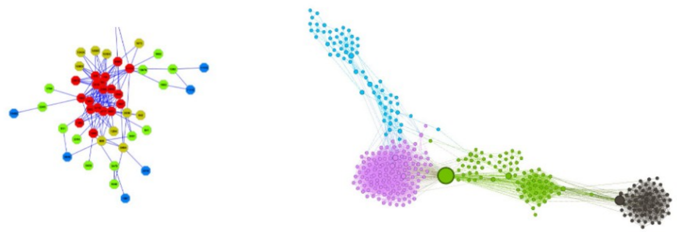
$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{matrix} 1/3 & 1/3 & 5/12 & 9/24 & & 6/15 \\ 1/3 & 3/6 & 1/3 & 11/24 & \dots & 6/15 \\ 1/3 & 1/6 & 3/12 & 1/6 & & 3/15 \end{matrix}$$

Iteration 0, 1, 2, …

6 - Roles and Community Structure in Networks

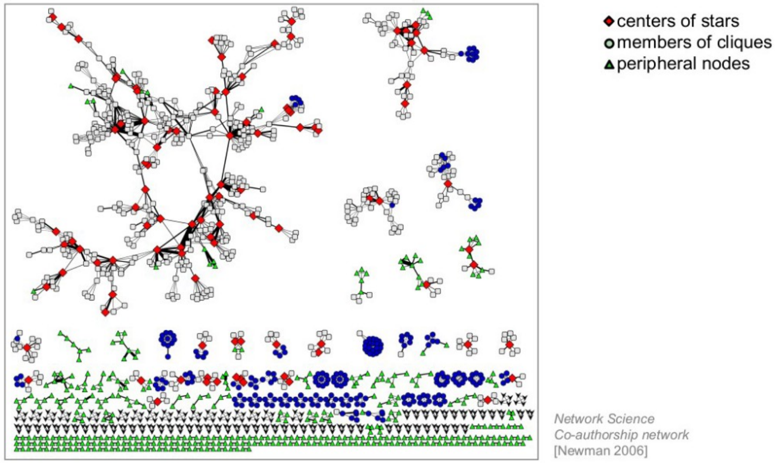
Community Structure in Networks

Pedro Ribeiro
(DCC/FCUP & CRACS/INESC-TEC)



(Mainly selected slides from Jure Leskovec and Gonzalo Mateos)

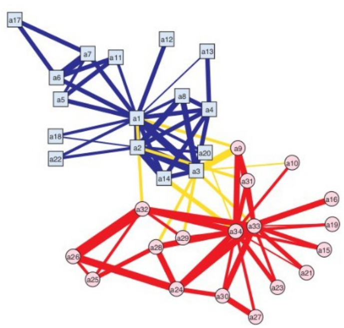
Examples of Roles



Network Science Co-authorship network
[Newman 2006]

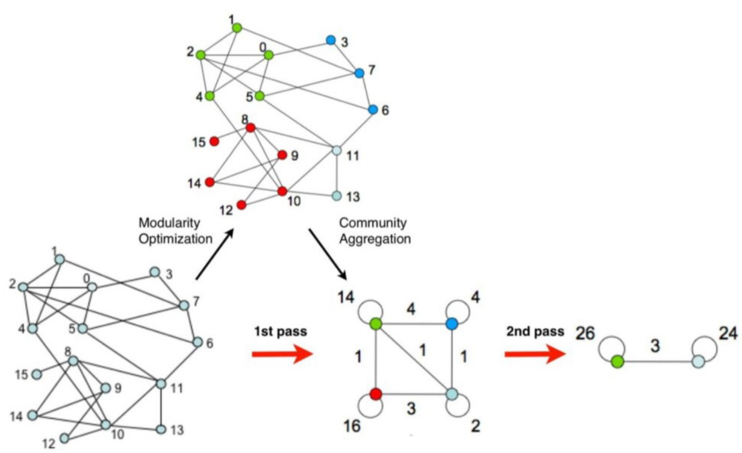
Zachary's karate club

- Social interactions among members of a karate club in the 70s

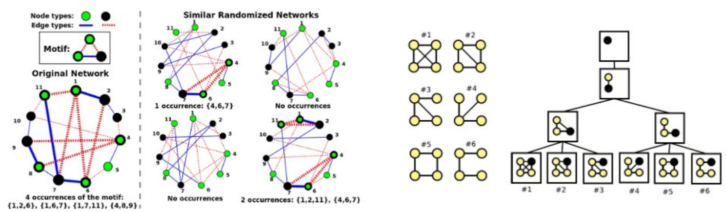


- Zachary witnessed the club split in two during his study
  - ⇒ Toy network, yet canonical for community detection algorithms
  - ⇒ Offers "ground truth" community membership (a rare luxury)

Louvain Algorithm Overview



7 - Subgraph Patterns



Subgraphs as Fundamental Ingredients of Complex Networks

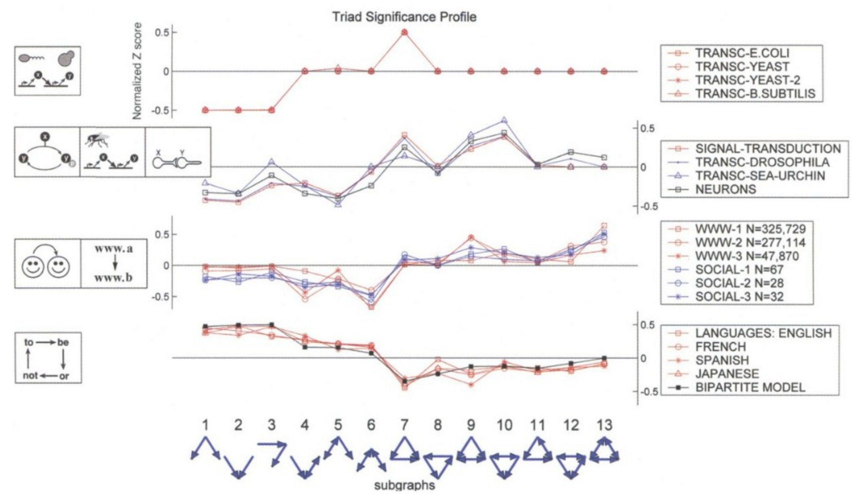
Concepts, Methods and Applications

Pedro Ribeiro

DCC/FCUP & CRACS/INESC-TEC

Network Science (DCC/FCUP)

Example Application



Different networks have similar fingerprints!

Image: (Milo et al., 2004)

Subgraph concepts – Significance

Traditional Null Model – keep **Degree Sequence**

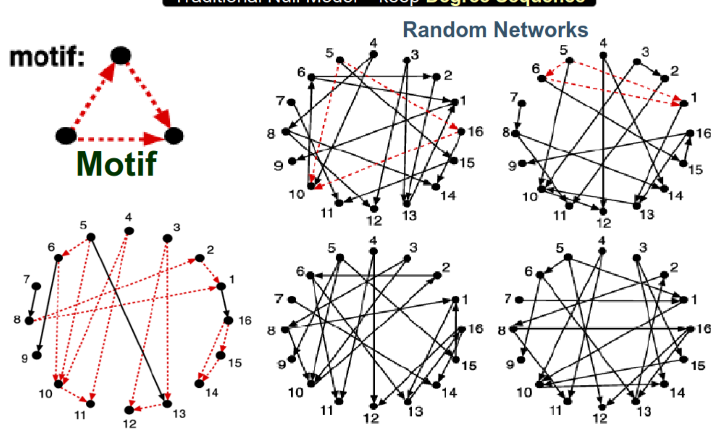
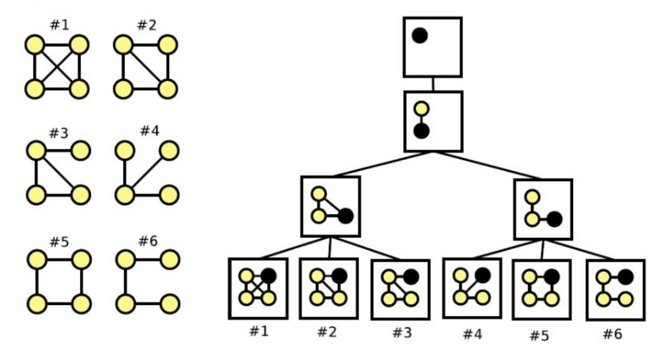


Image: Adapted from (Milo et al., 2002)

The G-Trie data structure

- **G-Tries:** (customized) collections of subgraphs
  - **Common substructures** are identified
  - Information is **"compressed"**



[Ribeiro & Silva, DMKD,2014]

8 - Analyzing networks with NetworkX

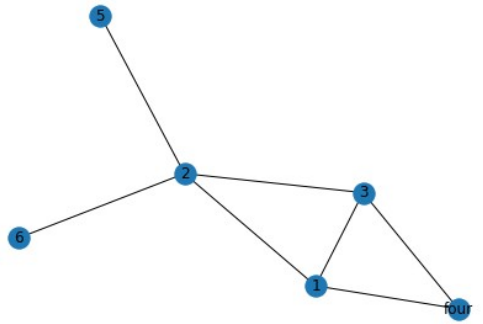
NetworkX
Network Analysis in Python

Adding edges and nodes explicitly

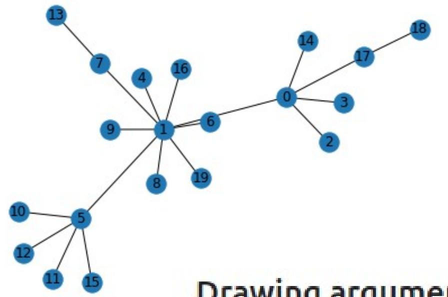
Here is our first example of an undirected graph:

```
# Create an empty undirected graph
G = nx.Graph()

# Create some nodes
G.add_node(1)              # create a single node
G.add_nodes_from([2,3])    # create nodes from a list
G.add_node("four")         # node labels can be of different types (anything hashable)

# Create some edges
G.add_edge(1,2)
G.add_edges_from([(2,3),(3,"four"), ("four", 1), (1,3)])
G.add_edges_from([(2,5),(2,6)]) # if a node does not exist, it is created

# Show nodes and edges
print(G.nodes)
print(G.edges)

# Draw the graph (more on this later)
nx.draw(G, with_labels = True)
```

```
[1, 2, 3, 'four', 5, 6]
[(1, 2), (1, 'four'), (1, 3), (2, 3), (2, 5), (2, 6), (3, 'four')]
```



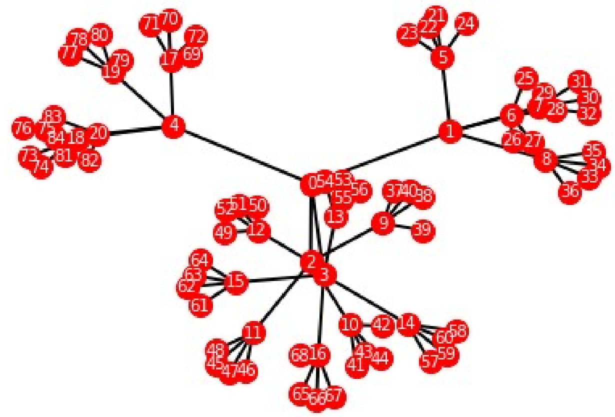

```
# Returns a random Barabasi-Albert graph with n=20 nodes and each new node connects to m=1 nodes
G = nx.barabasi_albert_graph(20,1)
nx.draw(G, with_labels=True)
```



Drawing arguments

It accepts many possible arguments:

```
G = nx.balanced_tree(4,3)

nx.draw(G, with_labels=True,
        node_size=200,      # size of nodes
        font_size=10,       # text label size
        node_color="red",   # color of nodes
        font_color="white", # color of nodes
        width=2,            # width of edges
        pos=nx.spring_layout(G, iterations=500, k=1.5)
        )
```



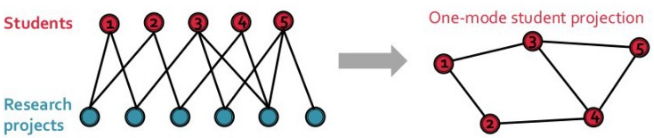
9 - Network Construction

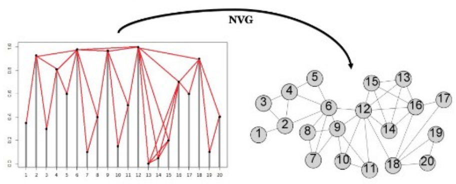
Network Construction

Pedro Ribeiro
(DCC/FCUP & CRACS/INESC-TEC)

U.PORTO FACULDADE DE CIÊNCIAS UNIVERSIDADE DO PORTO

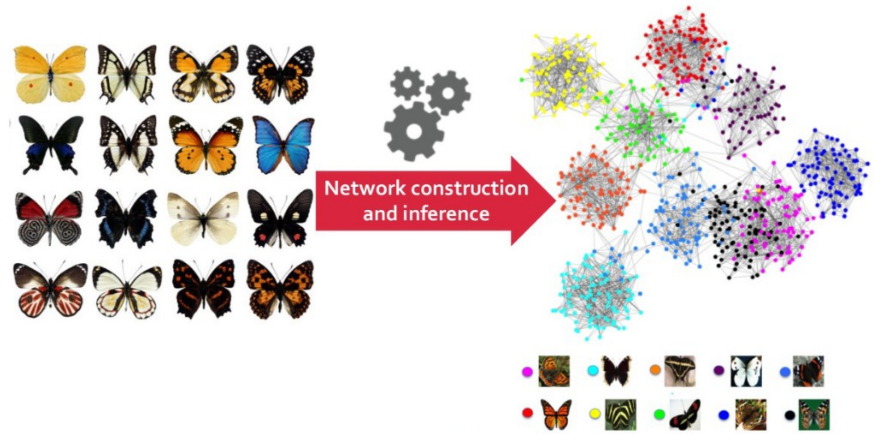
CRACS INESCTEC





(Mainly selected slides from Jure Leskovec, Lucas Lacasa and Vanessa Silva)

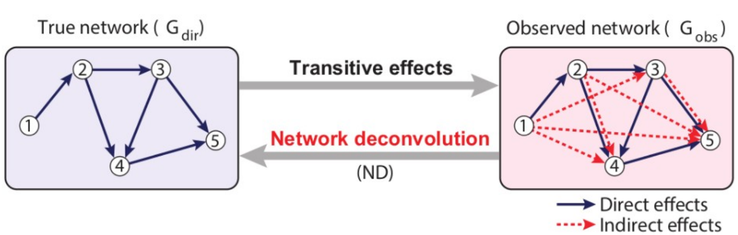
How to construct networks?



Today: How to construct and infer networks from raw data?

Network Deconvolution

- **Goal:** Reverse the effect of transitive information flow across all indirect paths:
  - Recover **true direct network (blue edges, G_{dir})** based on **observed network (combined blue and red edges, G_{obs})**

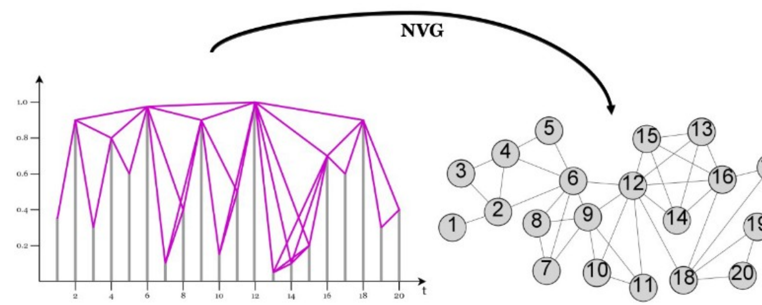


Feizi *et al.*, *Nature Biotechnology*, 31:8, 2013.

Visibility Graphs

$$y_c = y_b + (y_a - y_b)\frac{(t_b - t_c)}{t_b - t_a}, \qquad t_a < t_c < t_b$$

Natural Visibility Graph

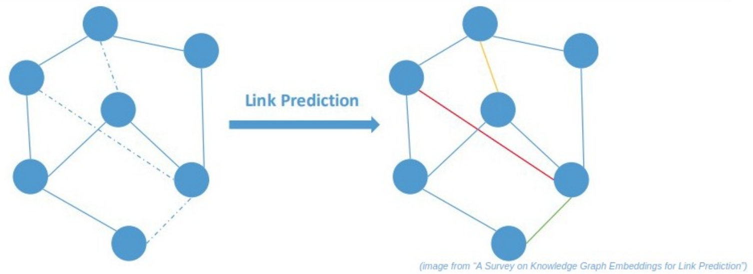


10 - Link Prediction

Link Prediction: an introduction

Pedro Ribeiro
(DCC/FCUP & CRACS/INESC-TEC)

U.PORTO
FACULDADE DE CIÊNCIAS UNIVERSIDADE DO PORTO



(based on slides used by myself at PBS and by Marcia Oliveira)

Link Prediction: Methods

A Survey of Link Prediction in Complex Networks

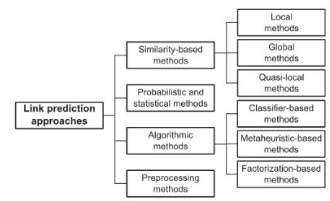
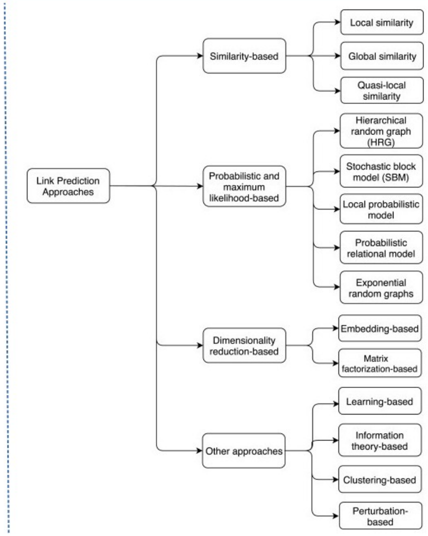


Fig. 1. Our proposed taxonomy for link prediction techniques.



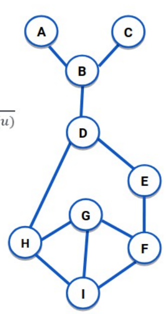
Resource Allocation Index

- Fraction of a "resource" that a node can send to another through their common neighbors

How to compute?

$$\text{res\_alloc}(i, j) = \sum_{u \in N(i) \cap N(j)} \frac{1}{|N(u)|} = \sum_{u \in N(i) \cap N(j)} \frac{1}{degree(u)}$$

N(i)/N(j) – set of neighbors of node i/ node j



Example:

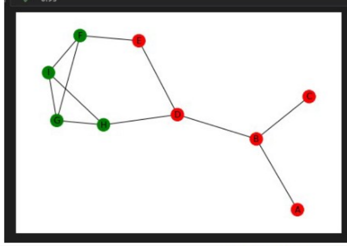
$$\text{res\_alloc}(F, H) = \frac{1}{3} + \frac{1}{3} = \frac{2}{3}$$

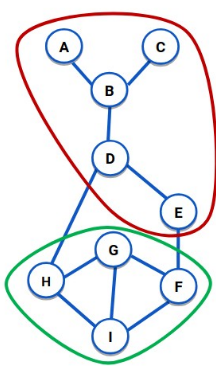
Community-based measures

Community-based measures

```python
G.nodes['A']['community'] = 0
G.nodes['B']['community'] = 0
G.nodes['C']['community'] = 0
G.nodes['D']['community'] = 0
G.nodes['E']['community'] = 0

G.nodes['F']['community'] = 1
G.nodes['G']['community'] = 1
G.nodes['H']['community'] = 1
G.nodes['I']['community'] = 1

colors = { 0 : 'red', 1 : 'green'}
communities = [colors[G.nodes[node]['community']] for node in G.nodes()]

nx.draw(G, with_labels=True, node_color=communities)
```

✓ 0.9s Python

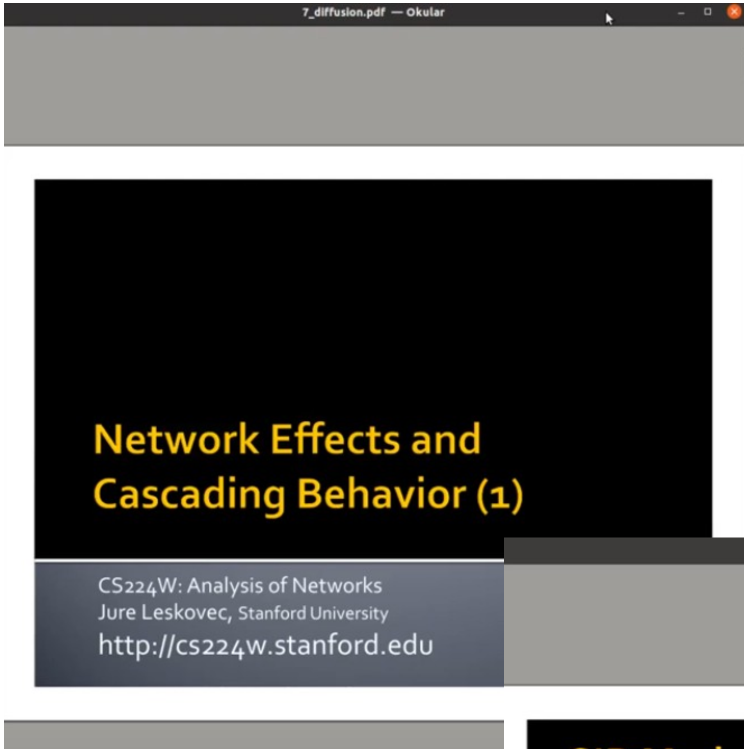


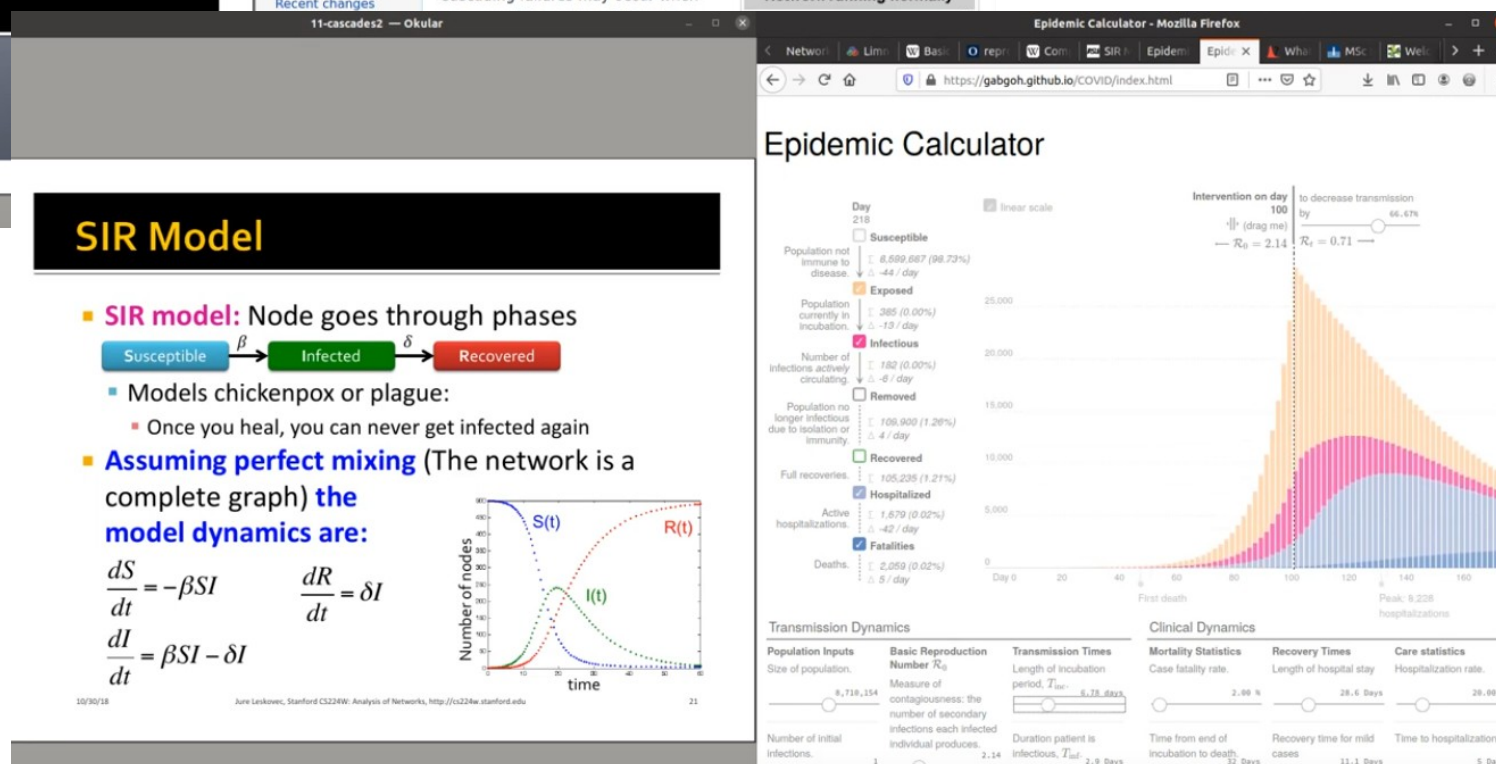


NetworkX
Network Analysis in Python

Other Material

Diffusion and Cascading Behavior



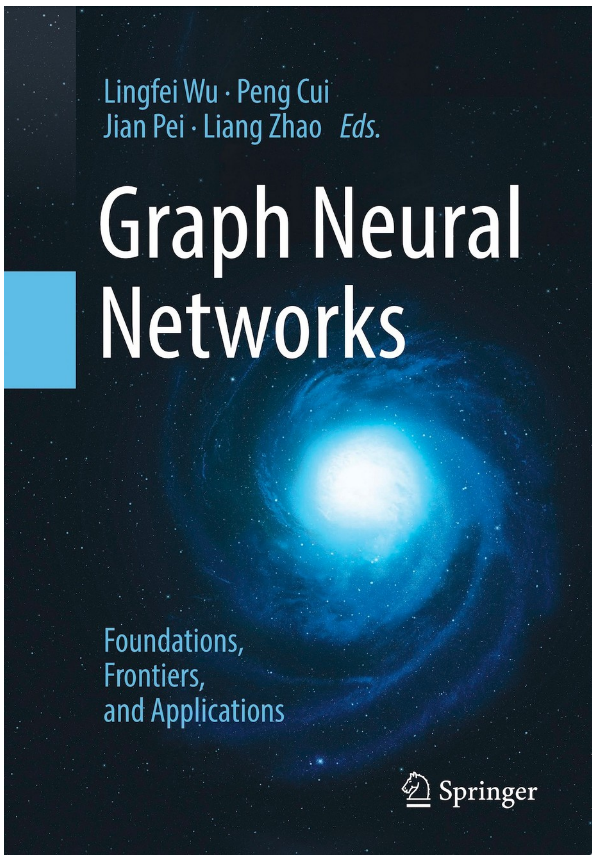


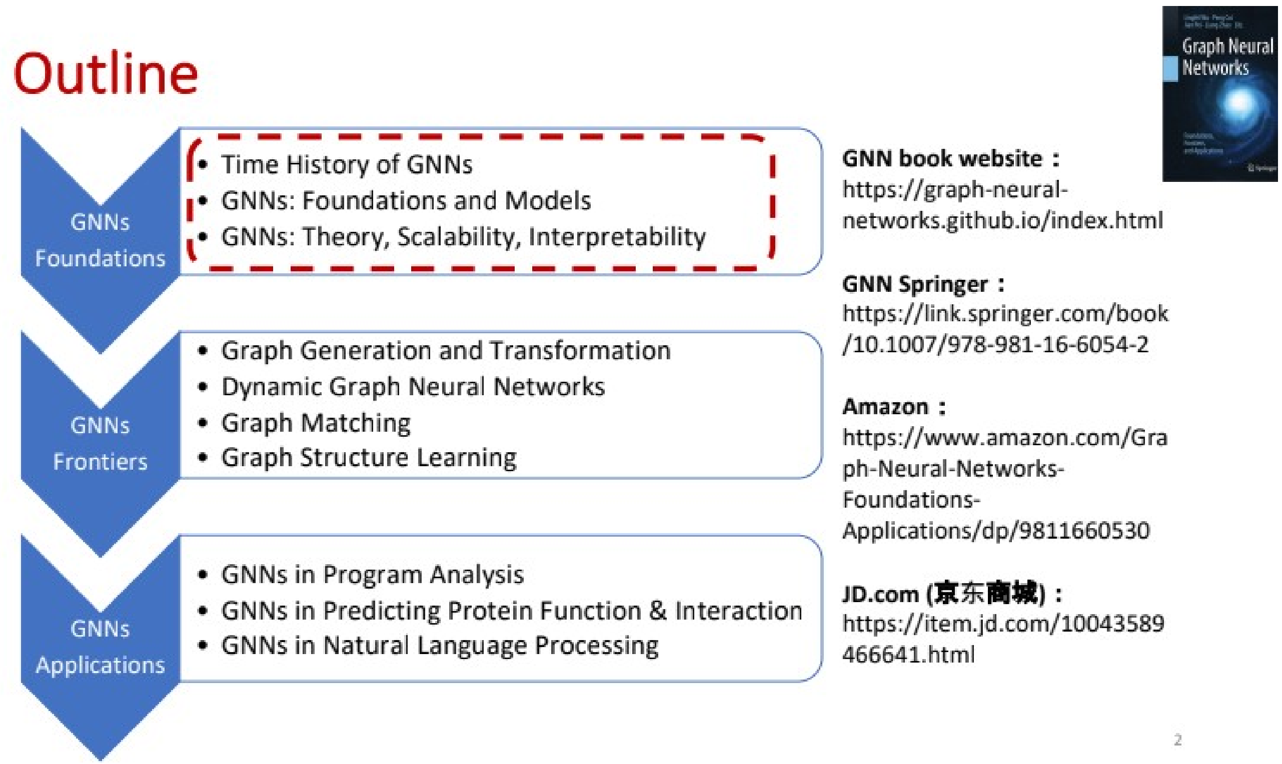
Slides and Videos from previous courses are available on the website

Graph Neural Networks

- *“deep learning meets network science”*

https://graph-neural-networks.github.io/



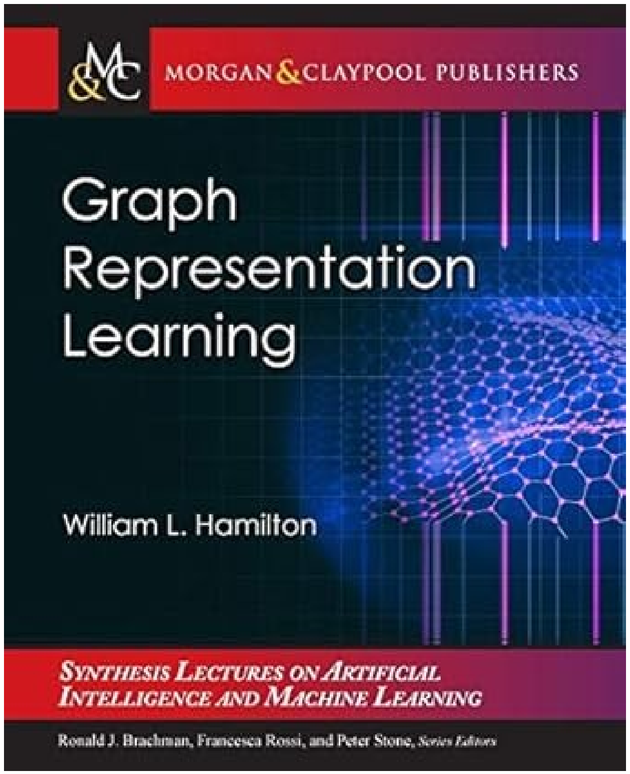


PyG

DEEP GRAPH LIBRARY

Graph Representation Learning

https://www.cs.mcgill.ca/~wlh/grl_book/



Preface

Acknowledgments

1 Introduction
1.1 What is a graph?
1.1.1 Multi-relational Graphs
1.1.2 Feature Information
1.2 Machine learning on graphs
1.2.1 Node classification
1.2.2 Relation prediction
1.2.3 Clustering and community detection
1.2.4 Graph classification, regression, and clustering

2 Background and Traditional Approaches
2.1 Graph Statistics and Kernel Methods
2.1.1 Node-level statistics and features
2.1.2 Graph-level features and graph kernels . . . 13
2.2 Neighborhood Overlap Detection . . . 16
2.2.1 Local overlap measures
2.2.2 Global overlap measures
2.3 Graph Laplacians and Spectral Methods
2.3.1 Graph Laplacians
2.3.2 Graph Cuts and Clustering
2.3.3 Generalized spectral clustering
2.4 Towards Learned Representations

I Node Embeddings

3 Neighborhood Reconstruction Methods
3.1 An Encoder-Decoder Perspective
3.1.1 The Encoder
3.1.2 The Decoder

II Graph Neural Networks . . . 46

5 The Graph Neural Network Model . . . 47
5.1 Neural Message Passing . . . 48
5.1.1 Overview of the Message Passing Framework . . . 48
5.1.2 Motivations and Intuitions . . . 50
5.1.3 The Basic GNN . . . 51
5.1.4 Message Passing with Self-loops . . . 52
5.2 Generalized Neighborhood Aggregation . . . 52
5.2.1 Neighborhood Normalization . . . 53
5.2.2 Set Aggregators . . . 54
5.2.3 Neighborhood Attention . . . 56
5.3 Generalized Update Methods . . . 58
5.3.1 Concatenation and Skip-Connections . . . 59
5.3.2 Gated Updates . . . 61
5.3.3 Jumping Knowledge Connections . . . 61
5.4 Edge Features and Multi-relational GNNs . . . 62
5.4.1 Relational Graph Neural Networks . . . 62
5.4.2 Attention and Feature Concatenation . . . 63
5.5 Graph Pooling . . . 64
5.6 Generalized Message Passing . . . 66

III Generative Graph Models . . . 102

8 Traditional Graph Generation Approaches . . . 103
8.1 Overview of Traditional Approaches . . . 103
8.2 Erdös-Rényi Model . . . 104
8.3 Stochastic Block Models . . . 104
8.4 Preferential Attachment . . . 105
8.5 Traditional Applications . . . 107

9 Deep Generative Models . . . 108
9.1 Variational Autoencoder Approaches . . . 109
9.1.1 Node-level Latents . . . 111
9.1.2 Graph-level Latents . . . 113
9.2 Adversarial Approaches . . . 115
9.3 Autoregressive Methods . . . 117
9.3.1 Modeling Edge Dependencies . . . 117
9.3.2 Recurrent Models for Graph Generation . . . 118
9.4 Evaluating Graph Generation . . . 120
9.5 Molecule Generation . . . 121

Machine Learning With Graphs

https://web.stanford.edu/class/cs224w/

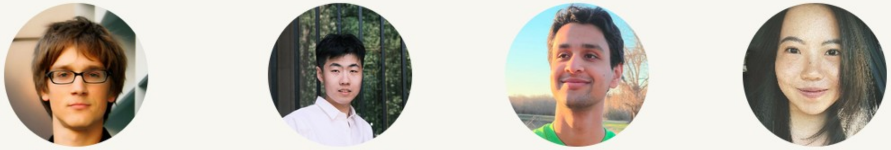
CS224W: Machine Learning with Graphs

Stanford / Fall 2024

Logistics

- **Lectures:** are on Tuesday/Thursday 3:00-4:20pm **in person** in the NVIDIA Auditorium.
- **Lecture Videos:** are available on Canvas for all the enrolled Stanford students.
- **Public resources**: The lecture slides and assignments will be posted online as the course progresses. We are happy for anyone to use these resources, but we cannot grade the work of any students who are not officially enrolled in the class.
- **Contact**: Students should ask *all* course-related questions on Ed (accessible from Canvas), where you will also find announcements. For external inquiries, personal matters, or in emergencies, you can email us at *cs224w-aut2425-staff@lists.stanford.edu*.
- **Academic accommodations**: If you need an academic accommodation based on a disability, you should initiate the request with the Office of Accessible Education (OAE). The OAE will evaluate the request, recommend accommodations, and prepare a letter for the teaching staff. Once you receive the letter, send it to our staff email address. Students should contact the OAE as soon as possible since timely notice is needed to coordinate accommodations.

Instructor **Course Assistants**



Jure Leskovec | Kexin Huang (Head CA) | Aman Patel | Harper Hua

1 

Stanford CS224W: Machine Learning with Graphs | 2021 | Lecture 1.1 - Why Graphs

Stanford Online • 522 mil visualizações • há 4 anos

2 

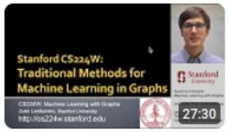
Stanford CS224W: Machine Learning with Graphs | 2021 | Lecture 1.2 - Applications of Graph ML

Stanford Online • 164 mil visualizações • há 4 anos

3 

Stanford CS224W: Machine Learning with Graphs | 2021 | Lecture 1.3 - Choice of Graph Representation

Stanford Online • 126 mil visualizações • há 4 anos

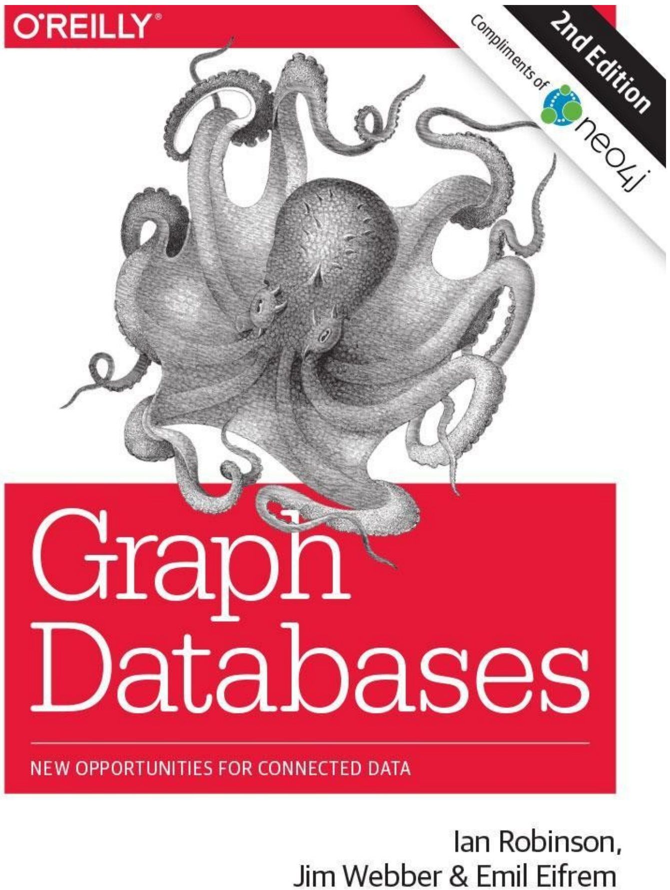
4 

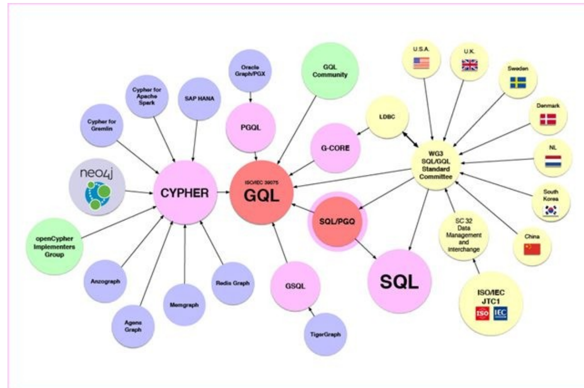
Stanford CS224W: ML with Graphs | 2021 | Lecture 2.1 - Traditional Feature-based Methods: Node

Stanford Online • 136 mil visualizações • há 4 anos

Graph Databases

https://en.wikipedia.org/wiki/Graph_database





neo4j

InfiniteGraph

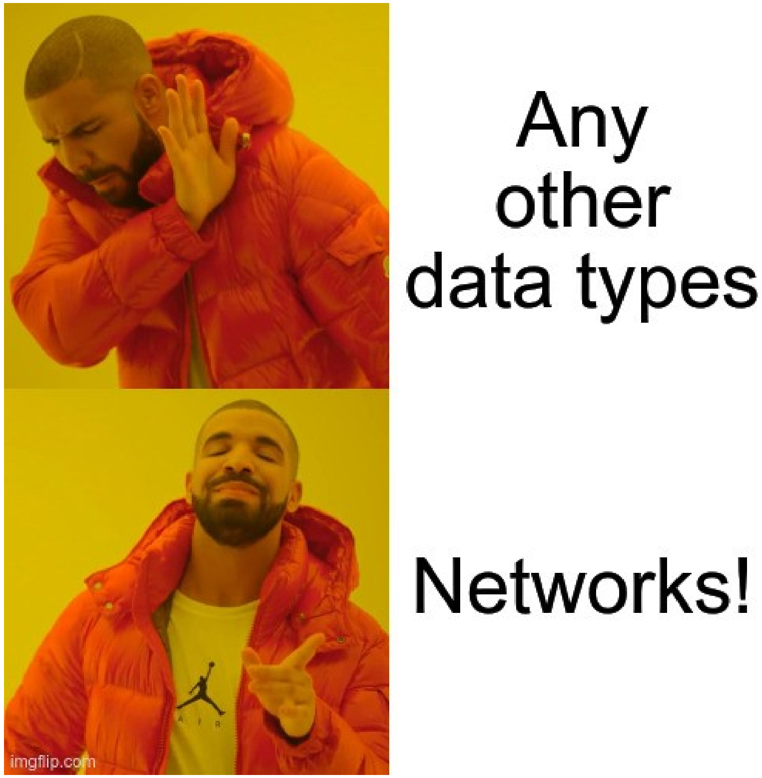
blazegraph

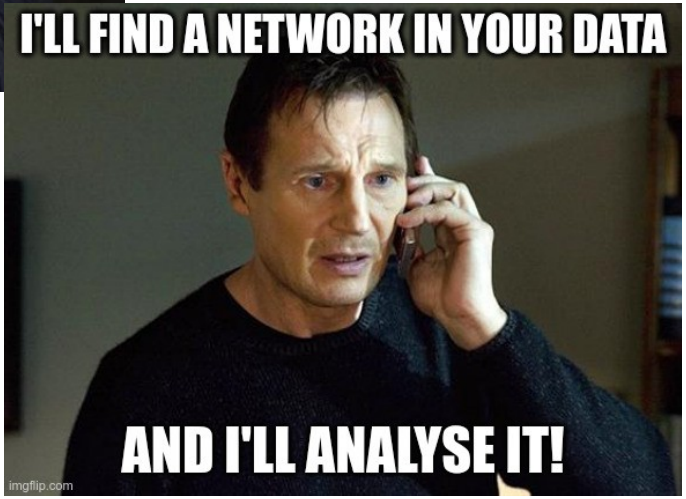
GraphDB

AllegroGraph

MSc Thesis in/with Network Science

Me, Myself... and Graphs!





ChatGPT

write lyrics for bohemian rhapsody bout about the beauty of graphs and network science

🎶 **Bohemian Graphsody – A Tribute to Network Science** 🎶

(To the tune of "Bohemian Rhapsody" by Queen)

[Intro – slow piano ballad]
Is this adjacency?
Is this just symmetry?
Caught in a structure
No escape from topology

Open your mind
Look to the sky and seeeee—
I'm just a node, now
I need no sympathy
Because I'm
In-degree, out-degree
Weighted edge, randomly
Any way it flows, the links really matter to me...
To meeee...

[Verse 1 – soft rock begins]
Mama, just built a graph
Put some edges in the file
Gave some weights, now watch it spiral
Mama, my code had just begun
But now I've gone and found a giant cliiiiique

Mama, ooooh…
Didn't mean to blow your mind
If I find a hub again this time tomorrow
Carry on, carry on—as if Erdős still mattered

🎶 Bohemian Graphsody – A Tribute to Network Science 🎶

(To the tune of "Bohemian Rhapsody" by Queen)

[Verse 2 – growing intensity]
Too late, my model's run
Sends a message to each friend
Through the shortest path it sends
Goodbye to random ties
I've got centrality—
Betweenness ranking's climbing up on me

Mama, ooooh (any way the flow goes…)
I can't resist this graph
I sometimes wish I'd never
seen PageRaaaaaank at all...

[Guitar Solo]

[Operatic Bridge – dramatic, many voices]
I see a little silhouetto of a node
Modularity! Modularity!
Will you do the detection?

Communities dividing, very very enlightening me!
(Oh-oh-oh) Clustering! (Oh-oh-oh) Clustering!
Clustering! Clustering! Figaro—metrics gooo!

[Heavy Rock Section]
I'm just a small node, nobody loves me
(He's just a small node in a dense community!)
Spared from isolation by connectivity!

Easy come, easy go, will this edge let it grow?
Bipartite! No, we will not let it grow! (Let it grow!)
Weighted edge! We will not let it grow! (Let it grow!)
Directed? No! (Let it grow!) Undirected? No! (Let it grow!)
Never never let it grow—oooooo!

🎶 Bohemian Graphsody – A Tribute to Network Science 🎶

(To the tune of "Bohemian Rhapsody" by Queen)

[Breakdown – dramatic, soft then loud]
No, no, no, no, no, no, no!
Oh my modularity, my modularity!
Modularity, let me gooooooo—
Erdős has a devil put aside for meee!
For meeeee! For meeeeeee!

[Finale – epic and reflective]
So you think you can cluster and leave me to cry?!
So you think you can label my node and not say why?!
Oh baby—can't do this to me baby—
Just gotta trace out, just gotta trace the whole graph here—

[Outro – soft piano, fading]
Oooh—
Nodes really matter…
Anyone can see…
Nodes really matter—
Nodes really matter...
To meeeeee...

(Any way the flow goes...)