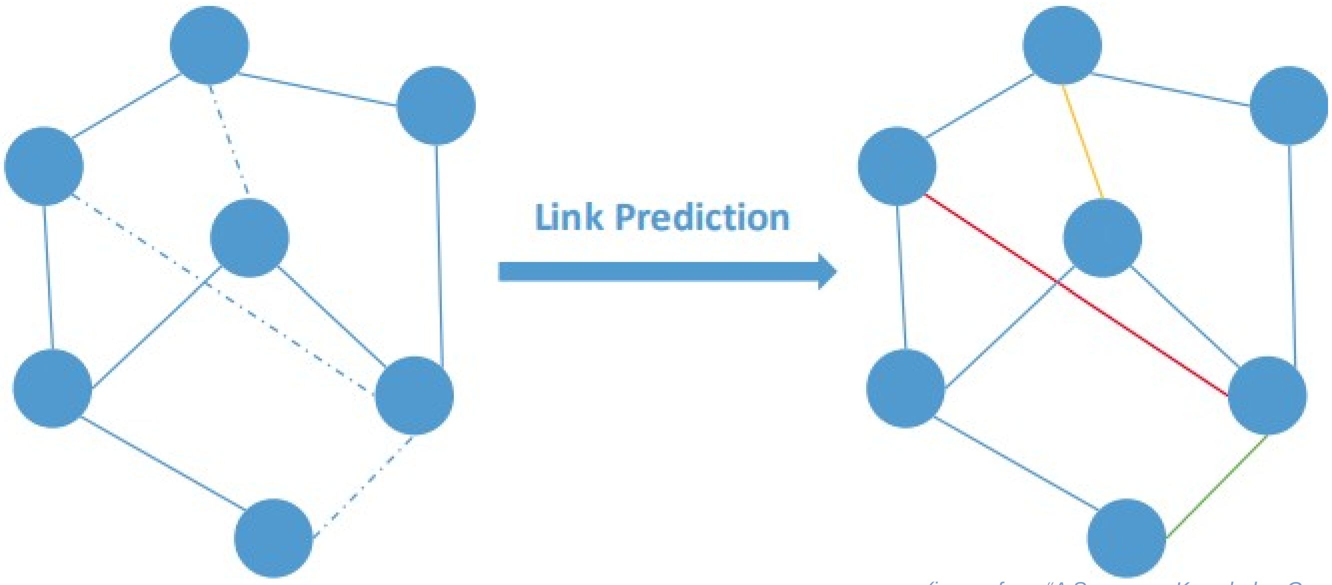
Link Prediction: an introduction

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



(image from "A Survey on Knowledge Graph Embeddings for Link Prediction")

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

The Link Prediction Problem

Link Prediction Problem

- To what extent can the **evolution** of a social network be modeled using **features intrinsic to the network** itself?

- **Intuitive definition:**

  Given a snapshot of a social network, can we infer/predict which new interactions (edges) among its entities are likely to occur in the near future?

- **A more rigorous definition:**

  Given a snapshot of a network at time t, link prediction seeks to accurately predict the edges that will be added to the network during the time t to a given future time t'.

Link Prediction Problem

- Example applications:

Recommender Systems

Who to follow and why: link prediction with explanations

Authors: Nicola Barbieri, Francesco Bonchi, Giuseppe Manco Authors Info & Claims

KDD '14: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining • August 2014 • Pages 1266–1275 • https://doi.org/10.1145/2623330.2623733

Online: 24 August 2014 Publication History

ABSTRACT

User recommender systems are a key component in any on-line social networking platform: they help the users growing their network faster, thus driving engagement and loyalty.

In this paper we study *link prediction with explanations* for user recommendation in social networks. For this problem we propose WTFW ("Who to Follow and Why"), a stochastic topic model for link prediction over directed and nodes-attributed graphs. Our model not only predicts links, but for each predicted link it decides whether it is a "topical" or a "social" link, and depending on this decision it produces a different type of explanation.

A topical link is recommended between a user interested in a topic and a user authoritative in that topic: the explanation in this case is a set of binary features describing the topic responsible of the link creation. A social link is recommended between users which share a large social neighborhood: in this case the explanation is the set of neighbors which are more likely to be responsible for the link creation.

Our experimental assessment on real-world data confirms the accuracy of WTFW in the link prediction and the quality of the associated explanations.

Link Prediction Problem

- Example applications:

Anomaly Detection

Social Network Analysis and Mining (2018) 8:27
https://doi.org/10.1007/s13278-018-0503-4

ORIGINAL ARTICLE

Generic anomalous vertices detection utilizing a link prediction algorithm

Dima Kagan[1] · Yuval Elovichi[1] · Michael Fire[2]

Received: 1 November 2017 / Revised: 18 February 2018 / Accepted: 21 March 2018 / Published online: 5 April 2018

Abstract
In the past decade, graph-based structures have penetrated nearly every aspect of our lives. The detection of anomalies in these networks has become increasingly important, such as in exposing infected endpoints in computer networks or identifying socialbots. In this study, we present a novel unsupervised two-layered meta-classifier that can detect irregular vertices in complex networks solely by utilizing topology-based features. Following the reasoning that a vertex with many improbable links has a higher likelihood of being anomalous, we applied our method on 10 networks of various scales, from a network of several dozen students to online networks with millions of vertices. In every scenario, we succeeded in identifying anomalous vertices with lower false positive rates and higher AUCs compared to other prevalent methods. Moreover, we demonstrated that the presented algorithm is generic, and efficient both in revealing fake users and in disclosing the influential people in social networks.

Link Prediction Problem

- Example applications:

Community Detection

Modern Physics Letters B | Vol. 32, No. 01, 1850004 (2018) | Research Papers

No Access

Community detection in complex networks using link prediction

Hui-Min Cheng, Yi-Zi Ning, Zhao Yin, Chao Yan, Xin Liu and Zhong-Yuan Zhang

https://doi.org/10.1142/S0217984918500045 | **Cited by:** 18

< Previous

Next >

PDF/EPUB

Tools **Share**

Abstract

Community detection and link prediction are both of great significance in network analysis, which provide very valuable insights into topological structures of the network from different perspectives. In this paper, we propose a novel community detection algorithm with inclusion of link prediction, motivated by the question whether link prediction can be devoted to improving the accuracy of community partition. For link prediction, we propose two novel indices to compute the similarity between each pair of nodes, one of which aims to add missing links, and the other tries to remove spurious edges. Extensive experiments are conducted on benchmark data sets, and the results of our proposed algorithm are compared with two classes of baselines. In conclusion, our proposed algorithm is competitive, revealing that link prediction does improve the precision of community detection.

Link Prediction Problem

- Example applications:

Predicting Citation Count of Scientists as a Link Prediction Problem

Publisher: IEEE Cite This PDF

Ertan Bütün ; Mehmet Kaya All Authors

Abstract:
The studies dealing with the problem of predicting scientific impacts in the scientific world mostly focus on predicting citation count of papers (PCCP). However, in the literature, only a little bit of research has been conducted on estimating the future influence of scientists individually. Estimating the impact of scientists individually is a worthwhile task for the following scientific research and cooperatives. From this point of view, a new supervised link prediction method is proposed to predict the citation count of scientists (PCCS). Many PCCP studies employ document-based attributes, such as titles, abstracts, and keywords of papers; institutions of scientists; impact factors of publishers; etc. and they do not take advantage of any topological features of complex networks formed with citations among papers. However, citation networks include valuable features for PCCP and PCCS. Therefore, we formulate the problem of PCCS as a link prediction problem in directed, weighted, and temporal citation networks. The proposed approach predicts not only links but also its weights. Our supervised link prediction method is tested on two citation networks in Experiment 1. The results of Experiment 1 confirm that our method achieves promising performances when considering pre its weights are addressed for the first time in terms of link prediction in directed, weighted, and networks. In Experiment 2, the performance of the proposed link prediction metric and five wel prediction metrics are compared in terms of prediction new links in complex networks. The resu Experiment 2 demonstrate that the proposed link prediction metric outperforms all baseline lin metrics.

Contents lists available at ScienceDirect
Neurocomputing
journal homepage: www.elsevier.com/locate/neucom
ELSEVIER

Classification using link prediction

Seyed Amin Fadaee, Maryam Amir Haeri*
Department of Computer Science and Information Technology, Amirkabir University of Technology, Iran

ARTICLE INFO
Article history:
Received 21 February 2019
Revised 24 May 2019
Accepted 5 June 2019
Available online 13 June 2019
Communicated by Prof. H. Zhang
Keywords:
Classification
Link prediction
Graph representation
Local similarity measure

ABSTRACT
Link prediction in a graph is the problem of detecting the missing links or the ones that would be formed in the near future. Using a graph representation of the data, we can convert the problem of classification to the problem of link prediction which aims at finding the missing links between the unlabeled data (unlabeled nodes) and their classes. To our knowledge, despite the fact that numerous algorithms use the graph representation of the data for classification, none are using link prediction as the heart of their classifying procedure. In this work, we propose a novel algorithm called CULP (Classification Using Link Prediction) which uses a new structure namely Label Embedded Graph or LEG and a link predictor to find the class of the unlabeled data. Different link predictors along with Compatibility Score - a new link predictor we proposed that is designed specifically for our settings - has been used and showed promising results for classifying different datasets. This paper further improved CULP by designing an extension called CULM which uses a majority vote (hence the M in the acronym) procedure with weights propor-

Analysis of offense tactics of basketball games using link prediction

Publisher: IEEE Cite This PDF

Tao Zhang ; Gongzhu Hu ; Qi Liao All Authors

Abstract:
Every basketball game has a lot of game records, also called match data. All the data are not only statistical but also logical and spatial. People normally use these kind of data to obtain statistical or summarized information of the games, but few have used these data to analyze the teams' tactics. In this paper, we present an approach to analyze the match data for detecting basketball teams' tactic using link prediction method. The main idea is to create a measure for the team offense tactics based on the Basketball Analysis Graph (BA graph) and use link prediction to extract the information about the cooperation between teammates and offense priority. The information may be used for basketball game strategy assistance.

Published in: 2013 IEEE/ACIS 12th International Conference on Computer and Information Science (ICIS)
Date of Conference: 16-20 June 2013 **INSPEC Accession Number:** 13797125

scientific reports

OPEN

Multimorbidity prediction using link prediction

Furqan Aziz[1,2], Victor Roth Cardoso[1,2], Laura Bravo-Merodio[1,2], Dominic Russ[1,2], Samantha C. Pendleton[1,2], John A. Williams[1,2], Animesh Acharjee[1,2,3] & Georgios V. Gkoutos[1,2,3,4,5,6]

Multimorbidity, frequently associated with aging, can be operationally defined as the presence of two or more chronic conditions. Predicting the likelihood of a patient with multimorbidity to develop a further particular disease in the future is one of the key challenges in multimorbidity research. In this paper we are using a network-based approach to analyze multimorbidity data and develop methods for predicting diseases that a patient is likely to develop. The multimorbidity data is represented using a temporal bipartite network whose nodes represent patients and diseases and a link between these nodes indicates that the patient has been diagnosed with the disease. Disease prediction then is reduced to a problem of predicting those missing links in the network that are likely to appear in the future. We develop a novel link prediction method for static bipartite network and validate the performance of the method on benchmark datasets. By using a probabilistic framework, we then report on the development of a method for predicting future links in the network, where links are labelled with a time-stamp. We apply the proposed method to three different multimorbidity datasets and report its performance measured by different performance metrics including AUC, Precision, Recall, and F-Score.

Link Prediction: Methods

- There is a multitude of possible methods

SURVEY

A Survey of Link Prediction in Complex Networks

Authors: Víctor Martínez, Fernando Berzal, Juan-Carlos Cubero Authors Info & Claims

ACM Computing Surveys, Volume 49, Issue 4 • December 2017 • Article No.: 69, pp 1–33 • https://doi.org/10.1145/3012704

Online: 20 December 2016 Publication History

Abstract

Networks have become increasingly important to model complex systems composed of interacting elements. Network data mining has a large number of applications in many disciplines including protein-protein interaction networks, social networks, transportation networks, and telecommunication networks. Different empirical studies have shown that

Link Prediction in Dynamic Social Networks: A Literature Review

Publisher: IEEE Cite This PDF

Mohammad Marjan; Nazar Zaki; Elfadil A. Mohamed All Authors

Abstract

Document Sections

I. Introduction

Abstract:

Social network link prediction has gained significant attention and become a key research focus over the last two decades. The prediction of missing links in the current network and emerging or broken links in future networks is essential for the understanding of their evolutionary nature. Social networks are changing dynamically over time. Link inference in dynamic social networks is an extremely challenging process and few

Contents lists available at ScienceDirect

Physica A

journal homepage: www.elsevier.com/locate/physa

Minireview

Link prediction in complex networks: A survey

Linyuan Lü [a,b,c], Tao Zhou [a,d,*]

[a] Web Sciences Center, University of Electronic Science and Technology of China, Chengdu 610054, People's Republic of China
[b] Research Center for Complex System Science, University of Shanghai for Science and Technology, Shanghai 200093, People's Republic of China
[c] Department of Physics, University of Fribourg, Chemin du Musée 3, CH-1700 Fribourg, Switzerland
[d] Department of Modern Physics, University of Science and Technology of China, Hefei 230026, People's Republic of China

ARTICLE INFO

Article history:
Received 5 October 2010
Received in revised form 10 November 2010
Available online 2 December 2010

Keywords:
Link prediction
Complex networks
Node similarity
Maximum likelihood methods
Probabilistic models

ABSTRACT

Link prediction in complex networks has attracted increasing attention from both physical and computer science communities. The algorithms can be used to extract missing information, identify spurious interactions, evaluate network evolving mechanisms, and so on. This article summaries recent progress about link prediction algorithms, emphasizing on the contributions from physical perspectives and approaches, such as the random-walk-based methods and the maximum likelihood methods. We also introduce three typical applications: reconstruction of networks, evaluation of network evolving mechanism and classification of partially labeled networks. Finally, we introduce some applications and outline future challenges of link prediction algorithms.

Contents lists available at ScienceDirect

Physica A

journal homepage: www.elsevier.com/locate/physa

Minireview

Link prediction techniques, applications, and performance: A survey

Ajay Kumar *, Shashank Sheshar Singh, Kuldeep Singh, Bhaskar Biswas

Department of Computer Science and Engineering, Indian Institute of Technology (BHU), Varanasi, 221–005, India

ARTICLE INFO

Article history:
Received 11 January 2019
Received in revised form 4 November 2019
Available online 8 February 2020

Keywords:
Link prediction
Similarity metrics
Probabilistic model
Embedding
Fuzzy logic
Deep learning

ABSTRACT

Link prediction finds missing links (in static networks) or predicts the likelihood of future links (in dynamic networks). The latter definition is useful in network evolution (Wang et al., 2011; Barabasi and Albert, 1999; Kleinberg, 2000; Leskovec et al., 2005; Zhang et al., 2015). Link prediction is a fast-growing research area in both physics and computer science domain. There exists a wide range of link prediction techniques like similarity-based indices, probabilistic methods, dimensionality reduction approaches, etc., which are extensively explored in different groups of this article. Learning-based methods are covered in addition to clustering-based and information-theoretic models in a separate group. The experimental results of similarity and some other representative approaches are tabulated and discussed. To make it general, this review also covers link prediction in different types of networks, for example, directed, temporal, bipartite, and heterogeneous networks. Finally, we discuss several applications with some recent developments and concludes our work with some future works.

Link Prediction: Methods

A Survey of Link Prediction in Complex Networks

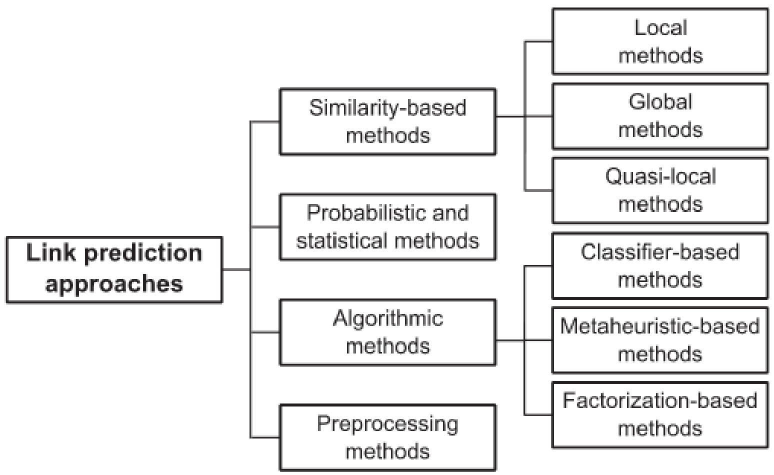
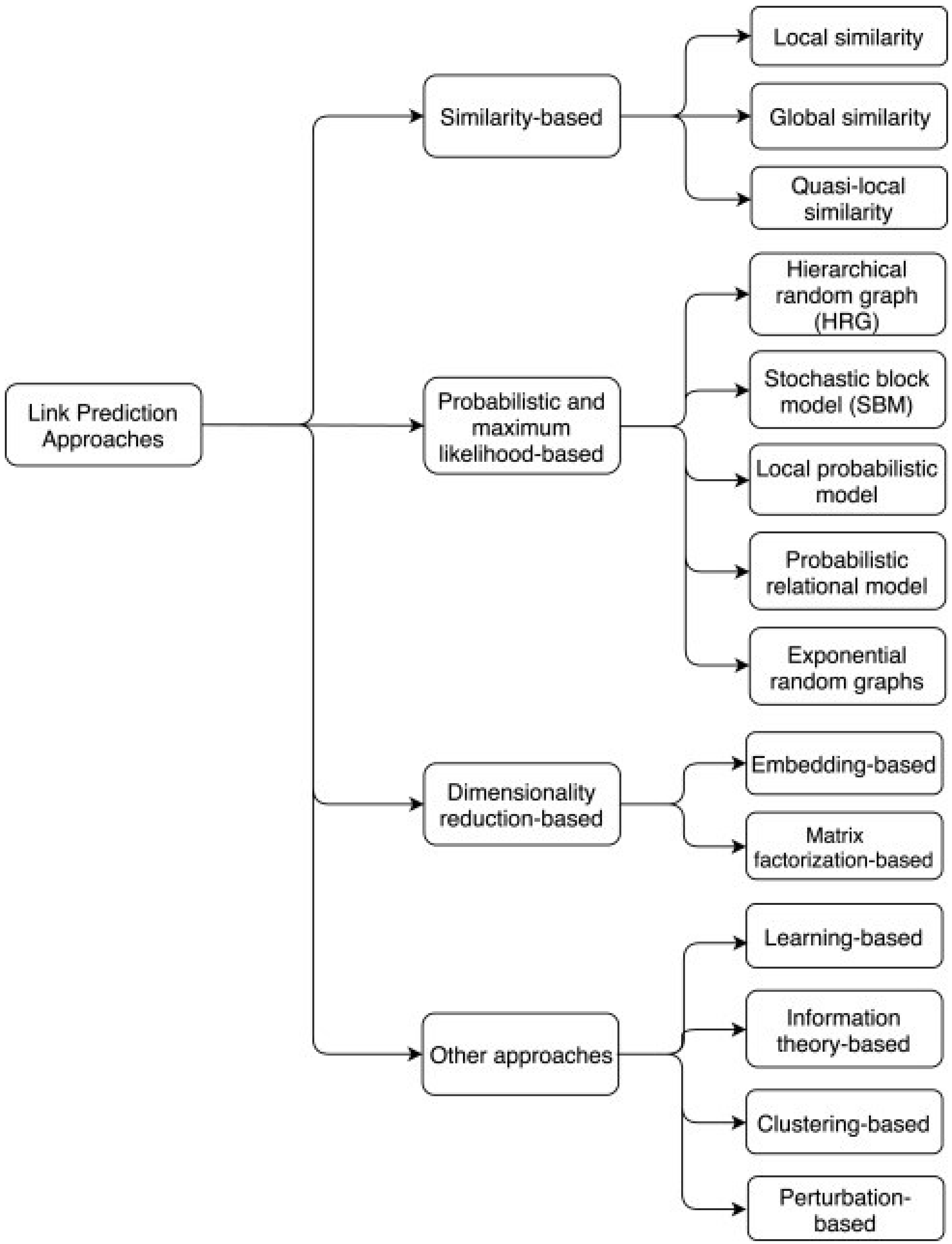


Fig. 1. Our proposed taxonomy for link prediction techniques.



Link Prediction Measures

- Today we will talk about some possible **similarity-based methods**

1	**Number of Common Neighbors**
2	**Jaccard Coefficient**
3	**Resource Allocation Index**
4	**Adamic-Adar Index**
5	**Preferential Attachment Score**
6	**Common Neighbor Soundarajan-Hopcroft Score**
7	**Resource Allocation Soundarajan-Hopcroft Score**

Example Network

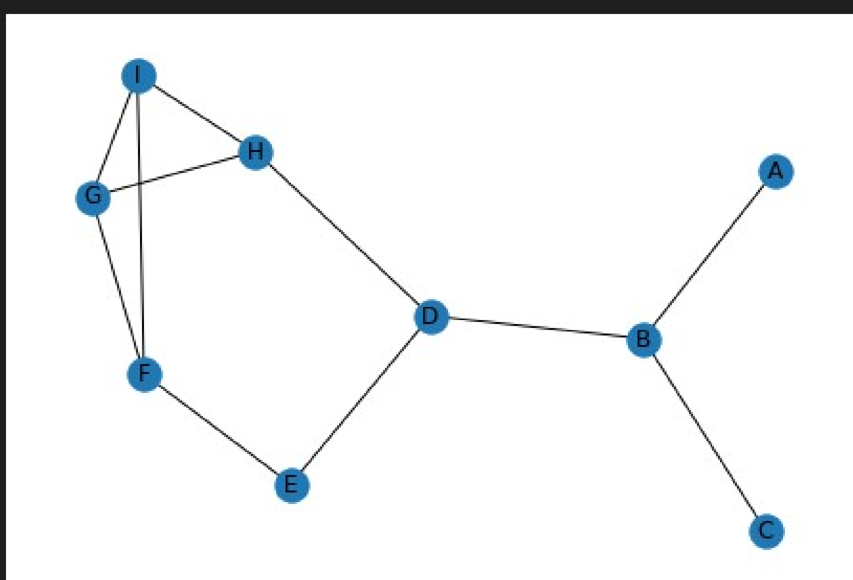
- We will use the following example network as we go through all the metrics

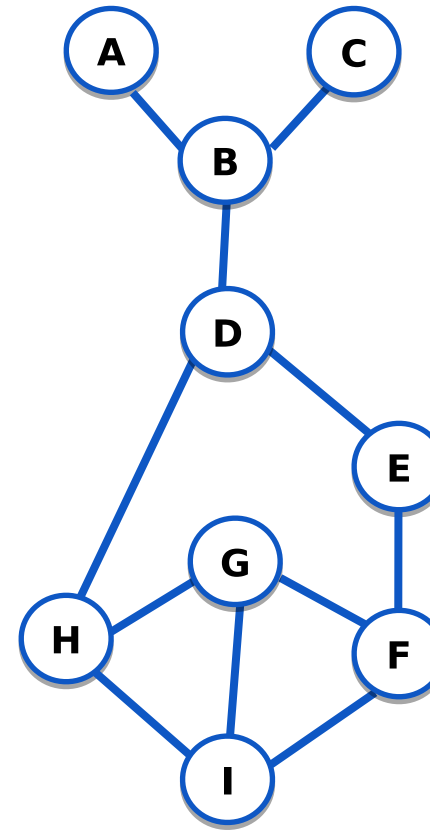
```python
import networkx as nx
import operator as op
```

✓ 0.2s Python

```python
G = nx.Graph()
edge_list = [('A', 'B'), ('B', 'C'), ('B', 'D'), ('D', 'E'), ('D', 'H'),
('H', 'G'), ('E', 'F'), ('G','F'), ('F', 'I'), ('H', 'I'), ('G', 'I')]
G.add_edges_from(edge_list)
nx.draw(G, with_labels=True)
```

✓ 0.1s Python

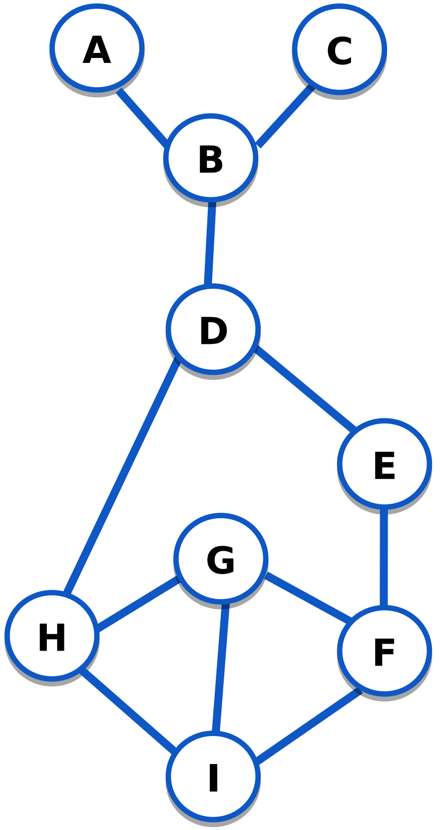




NetworkX
Network Analysis in Python

Number of Common Neighbors

- Very simple measure which is grounded on the network transitivity property of social networks
- **Triadic closure:** if two people in a social network have a friend (network neighbor) in common, then there is an increased likelihood that they will become friends themselves at some point in the future.



How to compute?

$$\text{comm\_neigh}(i, j) = |N(i) \cap N(j)|$$

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

Example:

$$\text{comm\_neigh}(F, H) = |\{G, I\}| = 2$$

Number of Common Neighbors

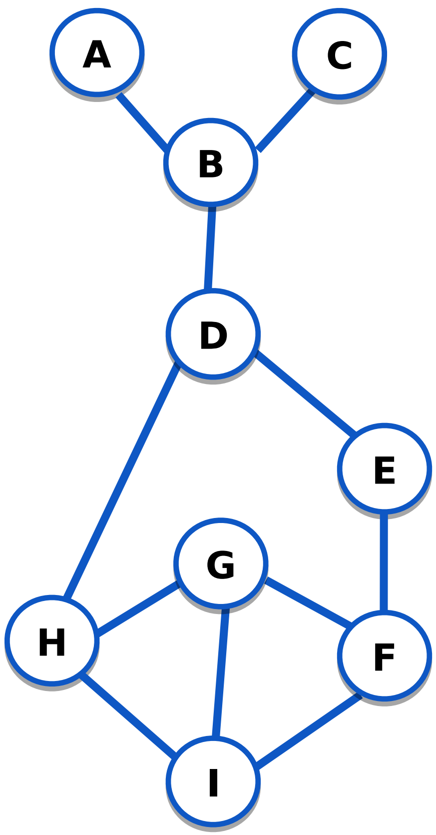
NetworkX
Network Analysis in Python

Number of Common Neighbors

```python
common_neigh = [(e[0], e[1], len(list(nx.common_neighbors(G, e[0], e[1]))))
                for e in nx.non_edges(G)]
common_neigh = sorted(common_neigh, key=op.itemgetter(2), reverse=True)

print(common_neigh)
```

0.3s Python

```
[('H', 'F', 2), ('B', 'H', 1), ('B', 'E', 1), ('A', 'D', 1), ('A', 'C', 1),
('D', 'G', 1), ('D', 'F', 1), ('D', 'C', 1), ('D', 'I', 1), ('H', 'E', 1),
('I', 'E', 1), ('G', 'E', 1), ('B', 'F', 0), ('B', 'I', 0), ('B', 'G', 0),
('A', 'H', 0), ('A', 'E', 0), ('A', 'F', 0), ('A', 'G', 0), ('A', 'I', 0),
('H', 'C', 0), ('C', 'G', 0), ('C', 'E', 0), ('C', 'F', 0), ('C', 'I', 0)]
```



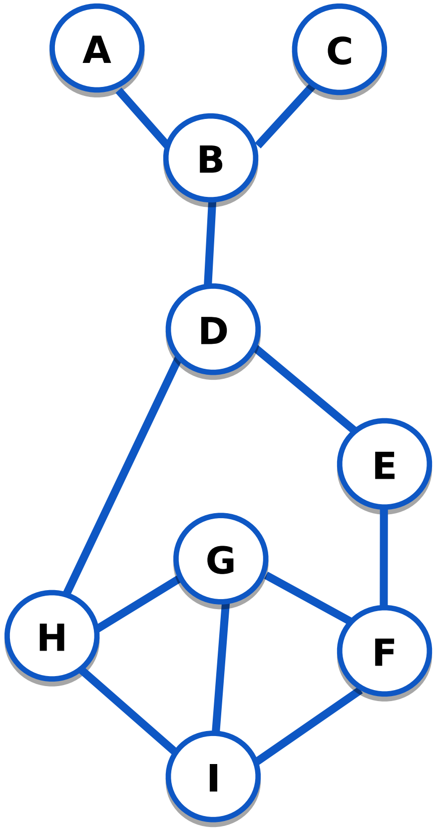
Jaccard Coefficient

- Number of common neighbors normalized by the total number of neighbors

How to compute?

$$\text{jacc\_coeff}(i, j) = \frac{|N(i) \cap N(j)|}{|N(i) \cup N(j)|}$$

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



Example:

$$\text{jacc\_coeff}(F, H) = \frac{|\{G, I\}|}{|\{D, G, I, E\}|} = \frac{2}{4} = \frac{1}{2}$$

Jaccard Coefficient

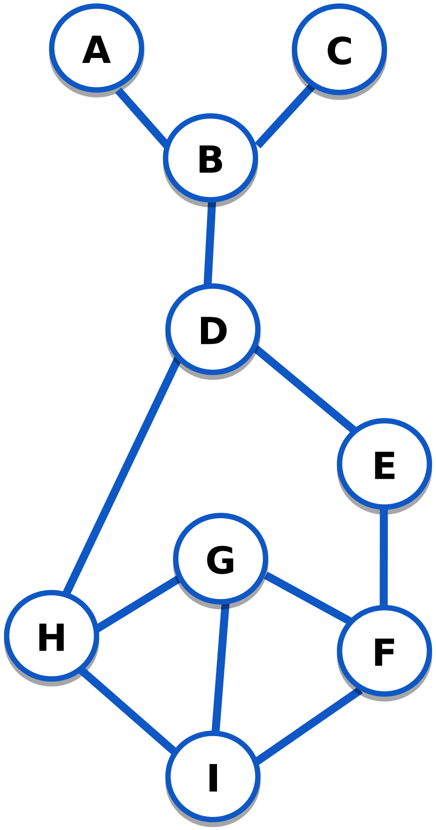
NetworkX
Network Analysis in Python

Jaccard Coefficient

```python
jacc_coeff = list(nx.jaccard_coefficient(G))
jacc_coeff.sort(key=op.itemgetter(2), reverse=True)

print(jacc_coeff)
```

✓ 0.6s Python

```
[('A', 'C', 1.0), ('H', 'F', 0.5), ('A', 'D', 0.3333333333333333), ('D', 'C',
0.3333333333333333), ('B', 'E', 0.25), ('H', 'E', 0.25), ('I', 'E', 0.25),
('G', 'E', 0.25), ('B', 'H', 0.2), ('D', 'G', 0.2), ('D', 'F', 0.2), ('D',
'I', 0.2), ('B', 'F', 0.0), ('B', 'I', 0.0), ('B', 'G', 0.0), ('A', 'H', 0.0),
('A', 'E', 0.0), ('A', 'F', 0.0), ('A', 'G', 0.0), ('A', 'I', 0.0), ('H', 'C',
0.0), ('C', 'G', 0.0), ('C', 'E', 0.0), ('C', 'F', 0.0), ('C', 'I', 0.0)]
```



Resource Allocation Index

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

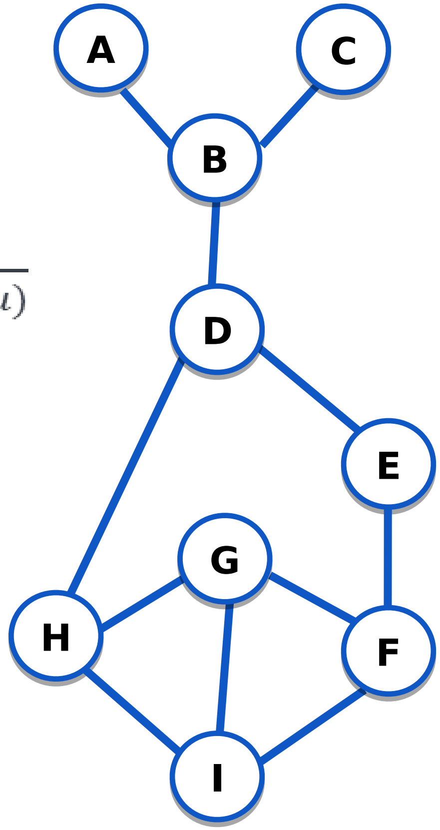
How to compute?

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

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

Example:

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



Resource Allocation Index

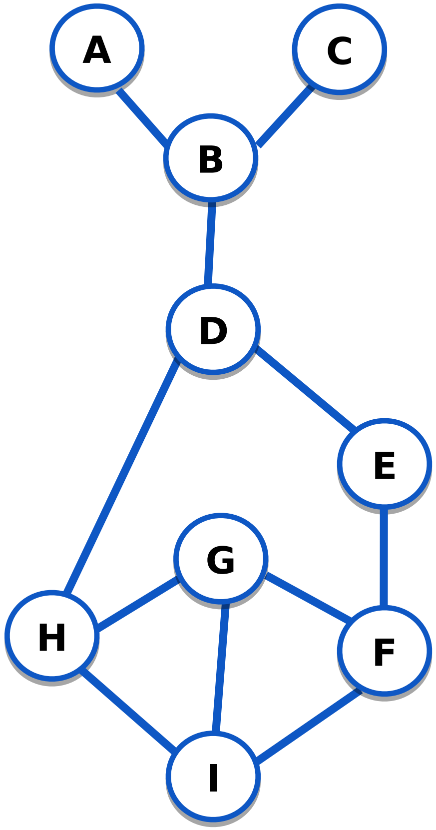
NetworkX
Network Analysis in Python

Resource Allocation Index

```python
res_alloc = list(nx.resource_allocation_index(G))
res_alloc.sort(key=op.itemgetter(2), reverse=True)

print(res_alloc)
```

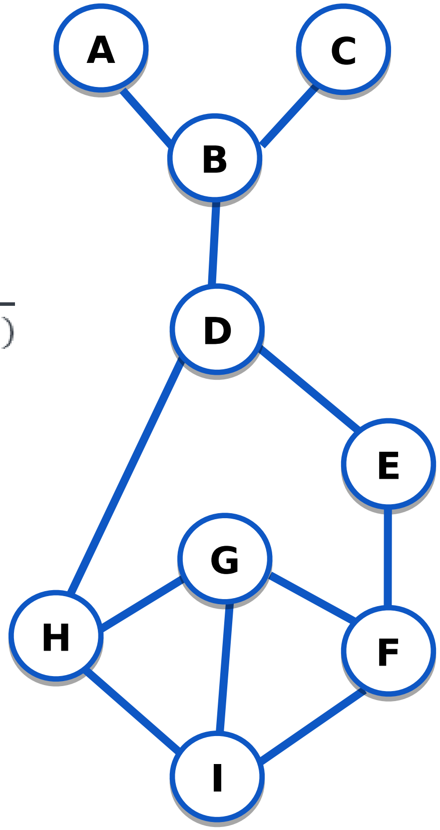
✓ 0.2s Python

```
[('H', 'F', 0.6666666666666666), ('D', 'F', 0.5), ('B', 'H',
0.3333333333333333), ('B', 'E', 0.3333333333333333), ('A', 'D',
0.3333333333333333), ('A', 'C', 0.3333333333333333), ('D', 'G',
0.3333333333333333), ('D', 'C', 0.3333333333333333), ('D', 'I',
0.3333333333333333), ('H', 'E', 0.3333333333333333), ('I', 'E',
0.3333333333333333), ('G', 'E', 0.3333333333333333), ('B', 'F', 0), ('B', 'I',
0), ('B', 'G', 0), ('A', 'H', 0), ('A', 'E', 0), ('A', 'F', 0), ('A', 'G', 0),
('A', 'I', 0), ('H', 'C', 0), ('C', 'G', 0), ('C', 'E', 0), ('C', 'F', 0),
('C', 'I', 0)]
```



Adamic-Adar Index

- Differs from the Resource Allocation Index, by computing the log of the degree. This measure formalizes the intuitive notion that rare characteristics/features are more telling, weighting more heavily these rare characteristics.



How to compute?

adamic_adar(i, j) = $\sum_{u \in N(i) \cap N(j)} \frac{1}{\log(|N(u)|)} = \sum_{u \in N(i) \cap N(j)} \frac{1}{\log(degree(u))}$

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

Example:

adamic_adar(F, H) = $\frac{1}{\log(3)} + \frac{1}{\log(3)} = 1{,}82$

Adamic-Adar Index

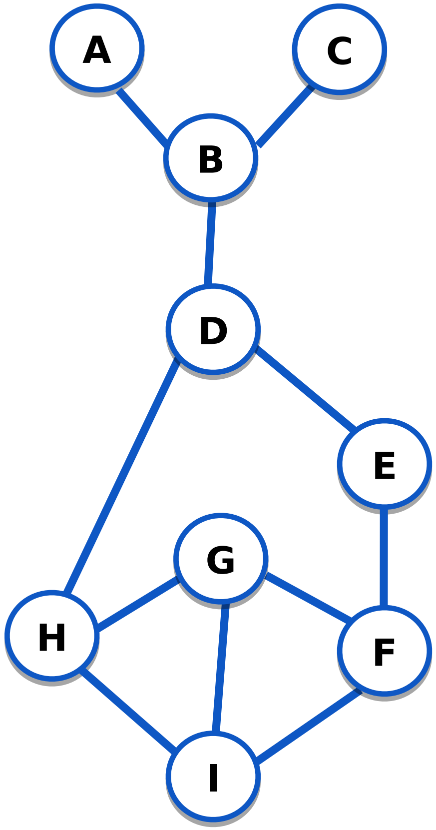
NetworkX
Network Analysis in Python

Adamic-Adar Index

```python
adamic_adar = list(nx.adamic_adar_index(G))
adamic_adar.sort(key=op.itemgetter(2), reverse=True)

print(adamic_adar)
```

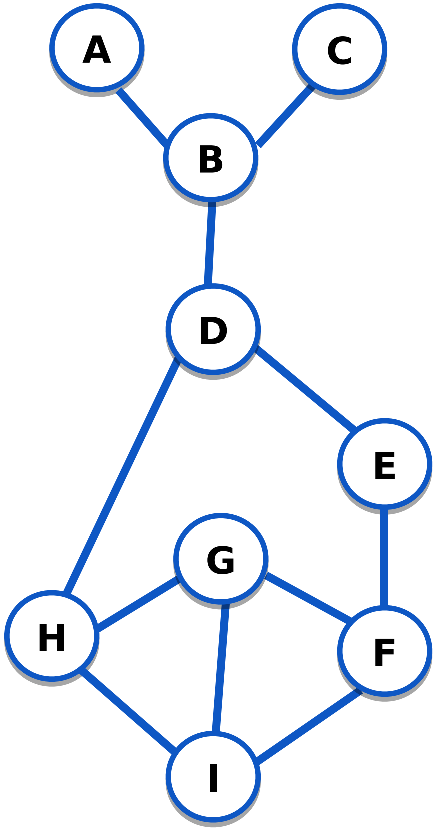
✓ 0.2s Python

```
[('H', 'F', 1.8204784532536746), ('D', 'F', 1.4426950408889634), ('B', 'H', 0.9102392266268373), ('B', 'E', 0.9102392266268373), ('A', 'D', 0.9102392266268373), ('A', 'C', 0.9102392266268373), ('D', 'G', 0.9102392266268373), ('D', 'C', 0.9102392266268373), ('D', 'I', 0.9102392266268373), ('H', 'E', 0.9102392266268373), ('I', 'E', 0.9102392266268373), ('G', 'E', 0.9102392266268373), ('B', 'F', 0), ('B', 'I', 0), ('B', 'G', 0), ('A', 'H', 0), ('A', 'E', 0), ('A', 'F', 0), ('A', 'G', 0), ('A', 'I', 0), ('H', 'C', 0), ('C', 'G', 0), ('C', 'E', 0), ('C', 'F', 0), ('C', 'I', 0)]
```



Preferential Attachment Score

- Relies on the preferential attachment mechanism since it assumes that nodes with high degrees are likely to get more neighbors in the future.



How to compute?

pref_attach(i, j) = $|N(i)|.|N(j)| = degree(i).degree(j)$

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

Example:

pref_attach(F, H) = $3x3 = 9$

Preferential Attachment Score

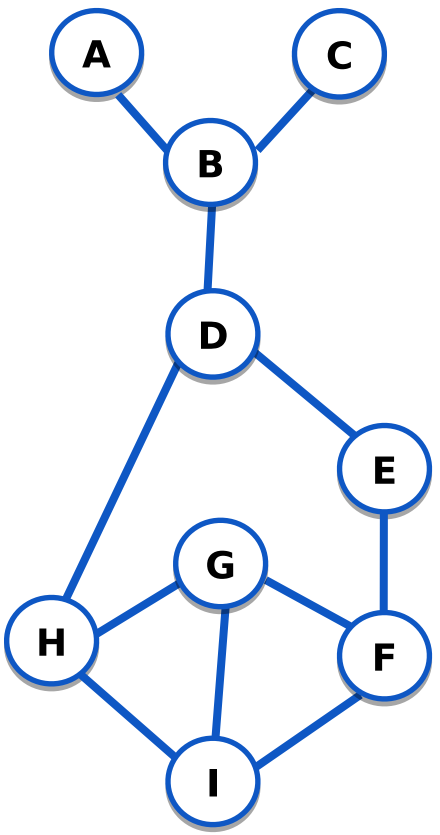
NetworkX
Network Analysis in Python

Preferential Attachment Score

```python
pref_attach = list(nx.preferential_attachment(G))
pref_attach.sort(key=op.itemgetter(2), reverse=True)

print(pref_attach)
```

✓ 0.2s Python

```
[('B', 'H', 9), ('B', 'F', 9), ('B', 'I', 9), ('B', 'G', 9), ('D', 'G', 9),
('D', 'F', 9), ('D', 'I', 9), ('H', 'F', 9), ('B', 'E', 6), ('H', 'E', 6),
('I', 'E', 6), ('G', 'E', 6), ('A', 'D', 3), ('A', 'H', 3), ('A', 'F', 3),
('A', 'G', 3), ('A', 'I', 3), ('D', 'C', 3), ('H', 'C', 3), ('C', 'G', 3),
('C', 'F', 3), ('C', 'I', 3), ('A', 'E', 2), ('C', 'E', 2), ('A', 'C', 1)]
```

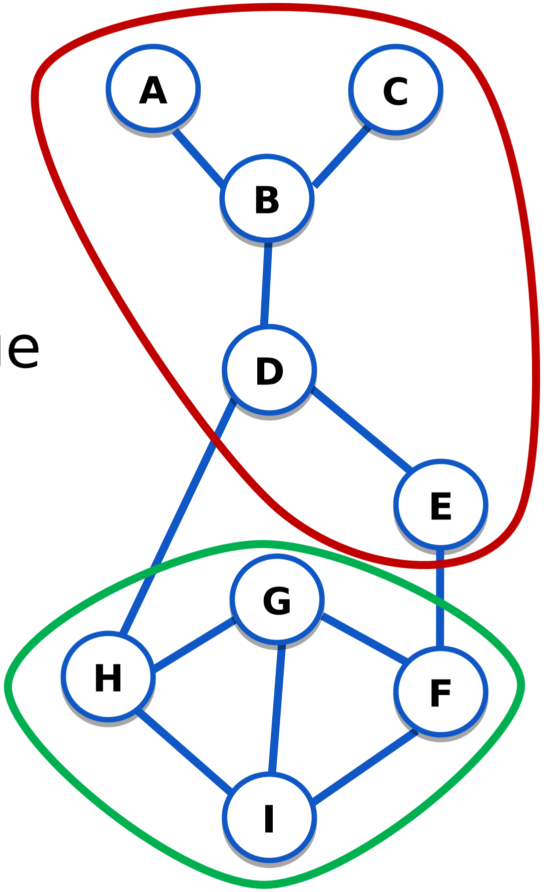


Community-based measures

- The next 2 measures are modifications of the Common Neighbors and Resource Allocation Index that <u>take into account the **community structure** of the network</u>

- **Main assumption:** nodes belonging to the same community are more likely to form an edge than nodes belonging to different communities

- Applies only to <u>disjoint communities</u> (each node belongs to only one community

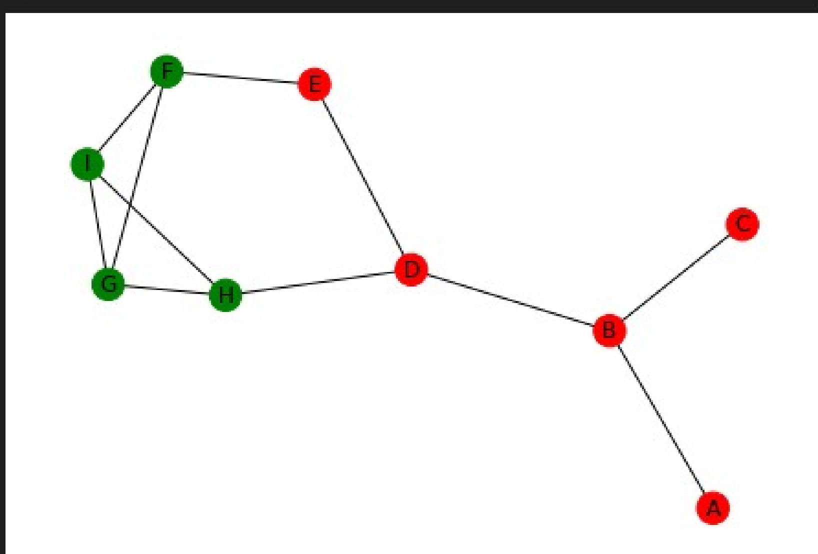


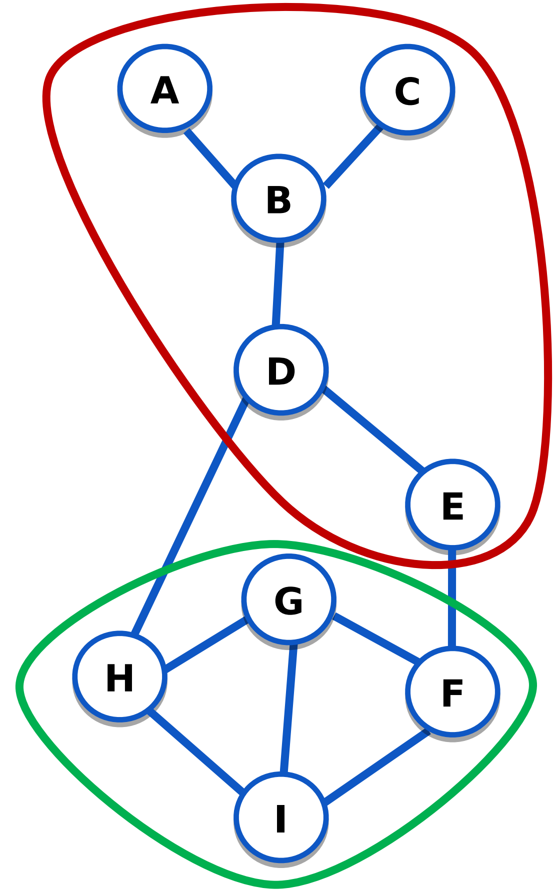
Community-based measures

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

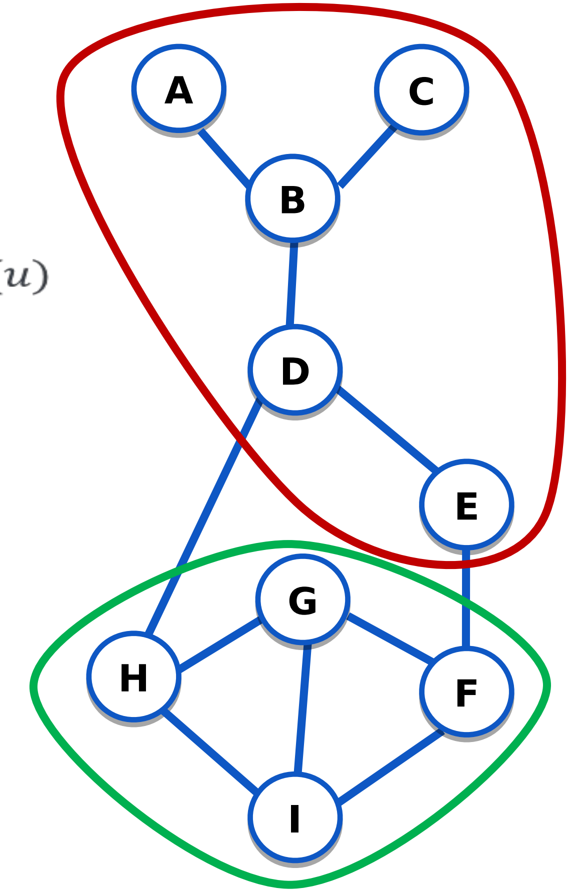




NetworkX
Network Analysis in Python

Common Neighbors Soundarajan-Hopcroft

- Number of common neighbors plus a bonus for neighbors that belong to the same community as the analyzed pair of nodes



How to compute?

$$\text{cn\_soundarajan\_hopcroft(i, j)} = |N(i) \cap N(j)| + \textstyle\sum_{u \in N(i) \cap N(j)} f(u)$$

$$\text{Where } f(u) = \begin{cases} 1, & u \text{ belongs to the same community as i and j} \\ 0, & \text{otherwise} \end{cases}$$

Example:

$$\text{cn\_soundarajan\_hopcroft(F, H)} = 2 + 1 + 1 = 4$$

$$\text{cn\_soundarajan\_hopcroft(B, H)} = 1 + 0 = 1$$

Common Neighbors Soundarajan-Hopcroft

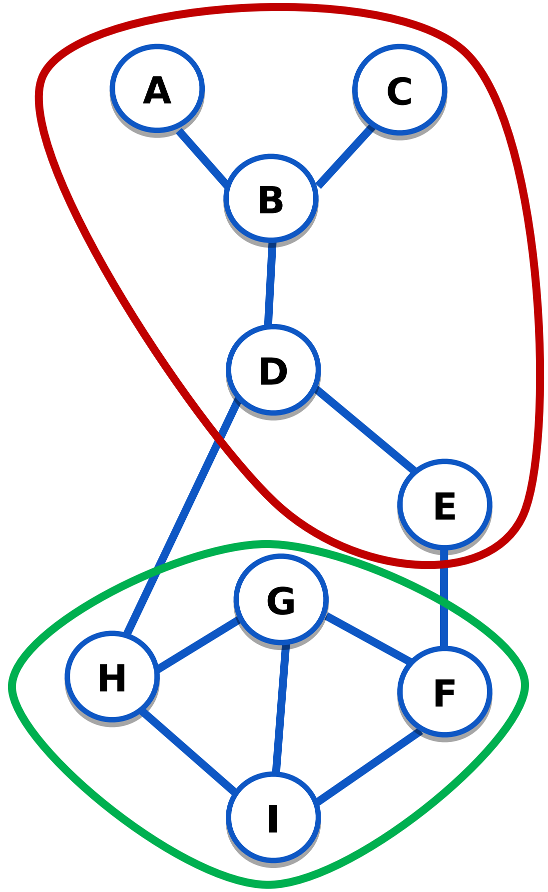
NetworkX
Network Analysis in Python

Common Neighbors - Soundarajan-Hopcroft

```python
cn_soundarajan_hopcroft = list(nx.cn_soundarajan_hopcroft(G))
cn_soundarajan_hopcroft.sort(key=op.itemgetter(2), reverse=True)

print(cn_soundarajan_hopcroft)
```

✓ 0.3s Python

```
[('H', 'F', 4), ('B', 'E', 2), ('A', 'D', 2), ('A', 'C', 2), ('D', 'C', 2),
('B', 'H', 1), ('D', 'G', 1), ('D', 'F', 1), ('D', 'I', 1), ('H', 'E', 1),
('I', 'E', 1), ('G', 'E', 1), ('B', 'F', 0), ('B', 'I', 0), ('B', 'G', 0),
('A', 'H', 0), ('A', 'E', 0), ('A', 'F', 0), ('A', 'G', 0), ('A', 'I', 0),
('H', 'C', 0), ('C', 'G', 0), ('C', 'E', 0), ('C', 'F', 0), ('C', 'I', 0)]
```



Resource Allocation Soundarajan-Hopcroft Index

- Similar to the Resource Allocation Index, but only takes into account nodes that are in the same community as the analyzed pair of nodes

How to compute?

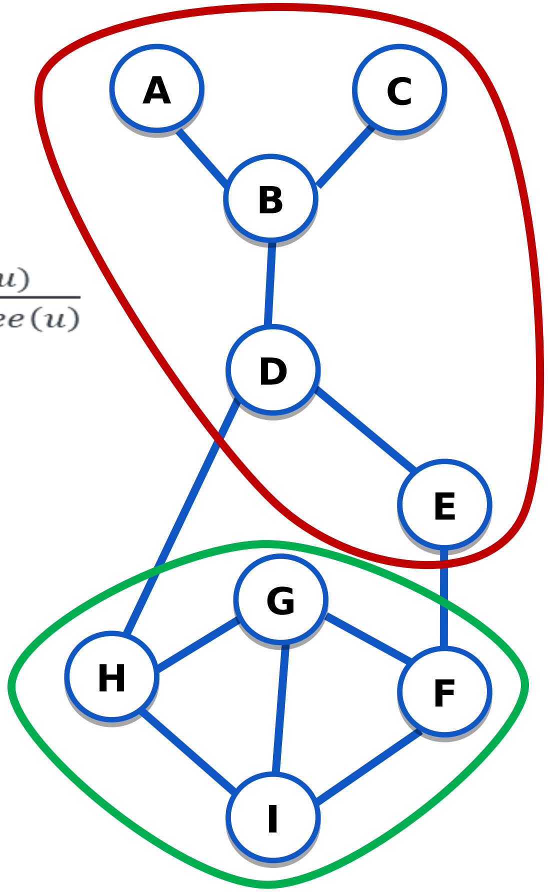
ra_soundarajan_hopcroft(i, j) = $\sum_{u \in N(i) \cap N(j)} \frac{f(u)}{|N(u)|} = \sum_{u \in N(i) \cap N(j)} \frac{f(u)}{degree(u)}$

Where $f(u) = \begin{cases} 1, & u \text{ belongs to the same community as i and j} \\ 0, & \text{otherwise} \end{cases}$

Example:

ra_soundarajan_hopcroft(F, H) = $\frac{1}{3} + \frac{1}{3} = \frac{2}{3}$

ra_soundarajan_hopcroft(B, H) = $\frac{0}{3} = 0$



Common Neighbors
Soundarajan-Hopcroft

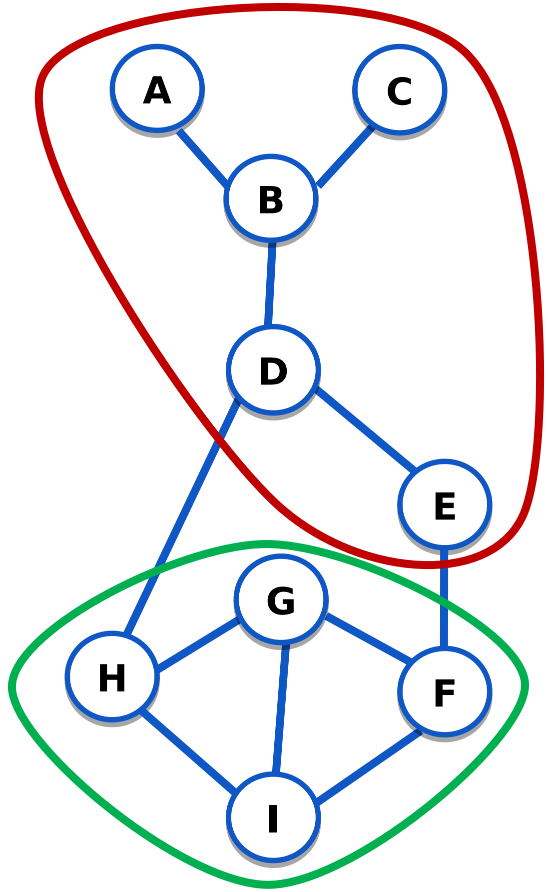
NetworkX
Network Analysis in Python

Resource Allocation - Soundarajan-Hopcroft Index

```python
ra_soundarajan_hopcroft = list(nx.ra_index_soundarajan_hopcroft(G))
ra_soundarajan_hopcroft.sort(key=op.itemgetter(2), reverse=True)

print(ra_soundarajan_hopcroft)
```

0.4s Python

```
[('H', 'F', 0.6666666666666666), ('B', 'E', 0.3333333333333333), ('A', 'D',
0.3333333333333333), ('A', 'C', 0.3333333333333333), ('D', 'C',
0.3333333333333333), ('B', 'H', 0), ('B', 'F', 0), ('B', 'I', 0), ('B', 'G',
0), ('A', 'H', 0), ('A', 'E', 0), ('A', 'F', 0), ('A', 'G', 0), ('A', 'I', 0),
('D', 'G', 0), ('D', 'F', 0), ('D', 'I', 0), ('H', 'C', 0), ('H', 'E', 0),
('C', 'G', 0), ('C', 'E', 0), ('C', 'F', 0), ('C', 'I', 0), ('I', 'E', 0),
('G', 'E', 0)]
```



Link Prediction - Limitations

- Link prediction measures “only” provide scores that give us a sense for whether two nodes are likely to connect in the future

- Lack of consistency across measures: different conclusions according to different measures

How can we turn them into useful information?

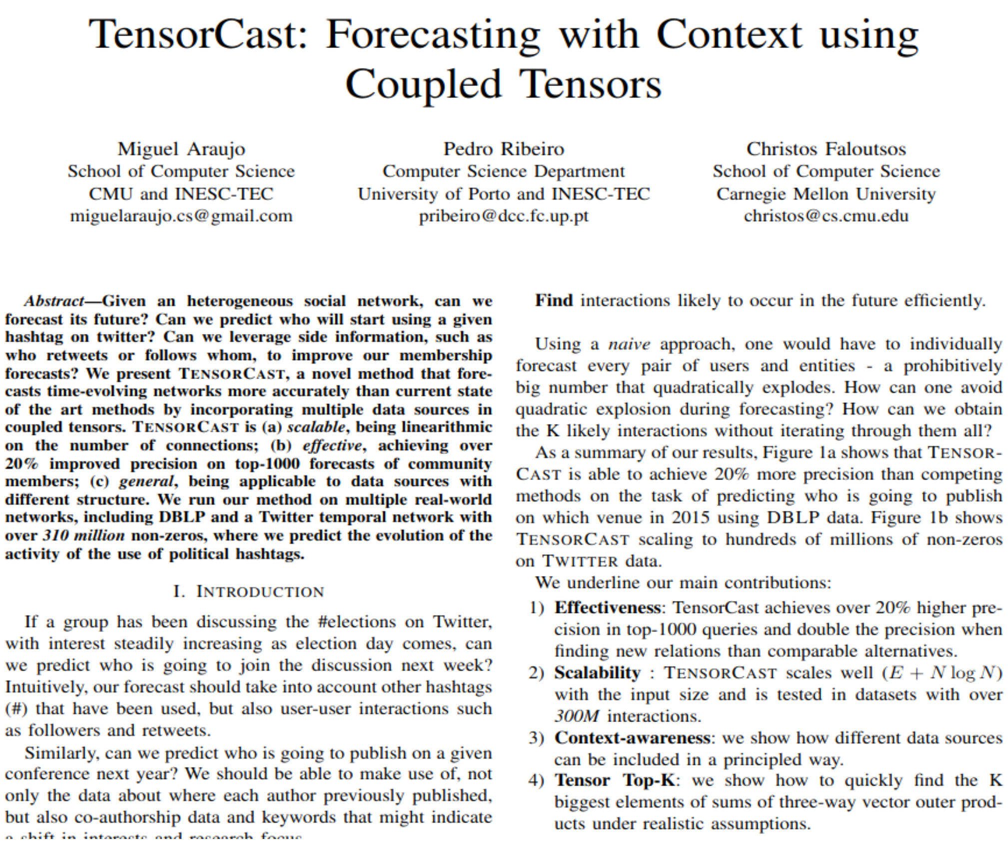
Link Prediction – War Story

War Story – A paper on predicting links

Miguel Araujo, Pedro Ribeiro and Christos Faloutsos

TensorCast: Forecasting with Context using Coupled Tensors *(Best Paper Award)*

Proceedings of the IEEE International Conference on Data Mining (ICDM), pp. 71-80, IEEE, New Orleans, USA, November, 2017.



TensorCast: Forecasting with Context using Coupled Tensors

Miguel Araujo
School of Computer Science
CMU and INESC-TEC
miguelaraujo.cs@gmail.com

Pedro Ribeiro
Computer Science Department
University of Porto and INESC-TEC
pribeiro@dcc.fc.up.pt

Christos Faloutsos
School of Computer Science
Carnegie Mellon University
christos@cs.cmu.edu

***Abstract*—Given an heterogeneous social network, can we forecast its future? Can we predict who will start using a given hashtag on twitter? Can we leverage side information, such as who retweets or follows whom, to improve our membership forecasts? We present TENSORCAST, a novel method that forecasts time-evolving networks more accurately than current state of the art methods by incorporating multiple data sources in coupled tensors. TENSORCAST is (a) *scalable*, being linearithmic on the number of connections; (b) *effective*, achieving over 20% improved precision on top-1000 forecasts of community members; (c) *general*, being applicable to data sources with different structure. We run our method on multiple real-world networks, including DBLP and a Twitter temporal network with over *310 million* non-zeros, where we predict the evolution of the activity of the use of political hashtags.**

I. Introduction

If a group has been discussing the #elections on Twitter, with interest steadily increasing as election day comes, can we predict who is going to join the discussion next week? Intuitively, our forecast should take into account other hashtags (#) that have been used, but also user-user interactions such as followers and retweets.

Similarly, can we predict who is going to publish on a given conference next year? We should be able to make use of, not only the data about where each author previously published, but also co-authorship data and keywords that might indicate

Find interactions likely to occur in the future efficiently.

Using a *naive* approach, one would have to individually forecast every pair of users and entities - a prohibitively big number that quadratically explodes. How can one avoid quadratic explosion during forecasting? How can we obtain the K likely interactions without iterating through them all?

As a summary of our results, Figure 1a shows that TENSORCAST is able to achieve 20% more precision than competing methods on the task of predicting who is going to publish on which venue in 2015 using DBLP data. Figure 1b shows TENSORCAST scaling to hundreds of millions of non-zeros on TWITTER data.

We underline our main contributions:

1) **Effectiveness**: TensorCast achieves over 20% higher precision in top-1000 queries and double the precision when finding new relations than comparable alternatives.
2) **Scalability** : TENSORCAST scales well ($E + N \log N$) with the input size and is tested in datasets with over *300M* interactions.
3) **Context-awareness**: we show how different data sources can be included in a principled way.
4) **Tensor Top-K**: we show how to quickly find the K biggest elements of sums of three-way vector outer products under realistic assumptions.