Learning in Big Data Analytics
Lecture 5

Alexander Schönhuth

UNIVERSITÄT
BIELEFELD
Faculty of Technology

Bielefeld University
December 15, 2020

Social Networks as Graphs

Social Networks: Introduction

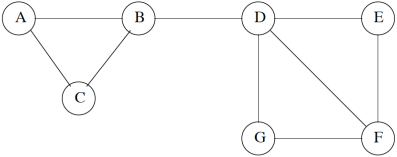
Basic Examples

- Facebook, Twitter, Google+

Defining Properties

- Collection of entities participating in network
  - Usually people, but other entities conceivable
- There is a relationship between the entities
  - Being friends is frequent relationship
  - Relationship can be of 0-1 type, or weighted
- Assumption of nonrandomness or locality
  - Hard to formalize, intuition is that relationships tend to cluster
  - If entity A is related with both B and C, B and C are related with larger probability

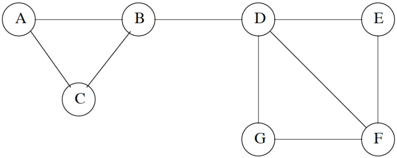
Social Network Graphs: Entities and Relationships



Adopted from `mmds.org`

- *Entities:* Nodes A to G
- *Relationships:* Represented by edges between nodes
  - *Example:* A is "friends" with B and C

Social Network Graphs: Locality



Adopted from `mmds.org`

- *Locality:*
  - There are 9 out of 21 possible edges: $\frac{9}{21} = 0.429$
  - Given nodes X, Y, Z such that there are edges $(X, Y), (Y, Z)$, random occurrence of (X, Z) is $\frac{7}{19} = 0.368$
  - However, across all pairs of existing edges $(X, Y), (Y, Z)$, probability that (X, Z) exists is $\frac{9}{16} = 0.563$
  - ☞ Network exhibits locality

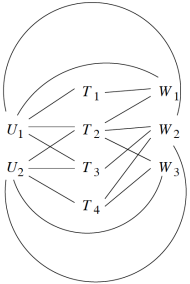
Social Networks: Examples

- *Telephone Networks:*
  - *Nodes* are phone numbers, *edges* exist if one number called another
  - *Edge weights:* Number of calls (within certain period of time)
  - *Communities:* Groups of friends, members of a club, people working at same company
- *Email Networks:*
  - *Nodes* are email addresses, *edges* indicate exchange of emails
  - *Edge directionality* may matter, so graph with directed edges
  - *Communities:* Similar to telephone networks

Social Networks: Examples

- *Collaboration Networks:*
  - *Nodes* e.g. represent authors, *edges* indicate working on same document
  - *Alternatively:* nodes represent documents, edges indicate that identical author contributed
  - *Communities:* Groups interested in / working on same subjects; documents sharing related content
- *Other:*
  - *Information networks:* Documents, web graphs, patents
  - *Infrastructure networks:* Roads, planes, water pipes, power grids
  - *Biological networks:* Genes, proteins, drugs
  - *Co-purchasing networks:* E.g. Gropon

SEVERAL TYPES OF NODES



Adopted from `mmds.org`

EXAMPLES

- Figure: Users (U) put tags (T) on documents (D): tri-partite network
- Put documents and authors into one bi-partite network

Social Networks: Topics

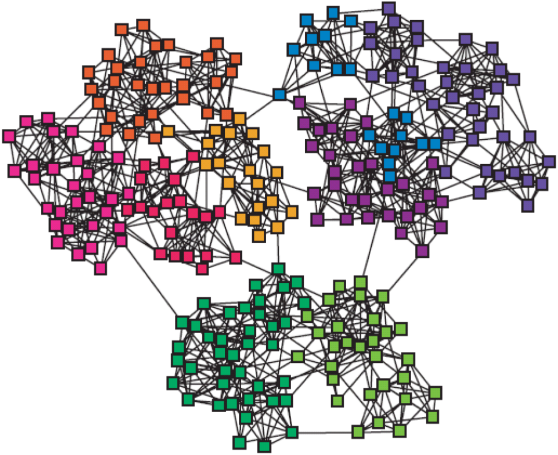
- Clustering, Betweenness & Girvan-Newman Algorithm (today)
- Direct Discovery of Communities & the Graph Affiliation Model (planned for December 22)
- Counting Triangles (planned for January 5)

Clustering Social Networks

Clustering Social Networks: Introduction

- An important aspect of social networks are *communities*
- Communities reveal themselves as groups of nodes that share unusually many edges
- Clustering social networks relates to the discovery of such communities

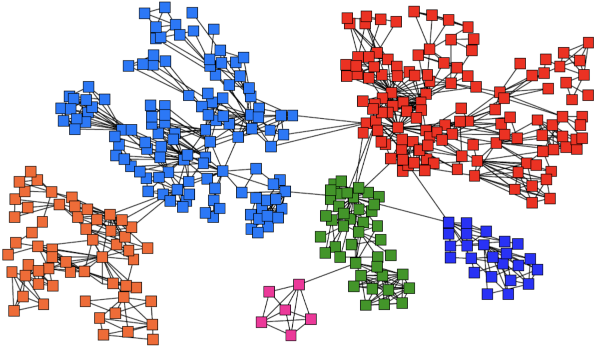
Communities



Differently Colored Communities in Social Network

Adopted from mmds.org

Clustered Network



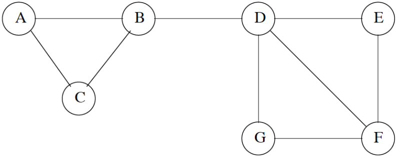
Differently Colored Clusters in Social Network

Adopted from `mmds.org`

Distance Measures in Social Networks

- Standard clustering techniques work with distance measures
- Distance measures are not obvious to define in social networks
  - Let $x, y \in V$ be two nodes in a social network $G = (V, E)$. The measure
    $$d(x,y) = \begin{cases} 0 & (x,y) \in E \\ 1 & (x,y) \notin E \end{cases}$$
    violates the triangle inequality, hence is no distance measure
  - Exchanging 0 with 1, and 1 with ∞ does not help
  - But other binary-valued measures (e.g. 1 and 1.5) agree with triangle inequality
- *But:* Additional issues apply

Social Networks: Clustering Issues



Communities: A-B-C and D-E-F-G

Adopted from `mmds.org`

- *Hierarchical Clustering:* Randomly picks closest nodes/clusters
- Distance between clusters: distance between closest points
- As soon as clusters are joined on B and D, clusters not as desired
- *Summary:* Standard clustering techniques difficult/impossible to sensibly implement

Betweenness

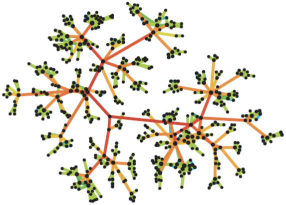
Idea: Identify edges that are least likely to be within community

Definition [Betweenness]

The *betweenness* of an edge (a, b) is

- the number of pairs of nodes (x, y) such that (a, b) makes part of the *shortest path* leading from x to y
- If for (x, y) there are several shortest paths, (a, b) is credited the fraction of shortest paths leading through (a, b) when computing its betweenness

BETWEENNESS



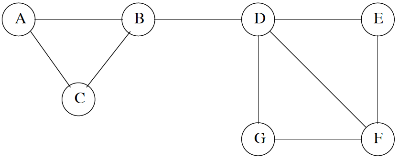
Telephone network:
Links between communities have great betweenness

Adopted from `mmds.org`

Explanation

- High betweenness means that (a, b) is a bottleneck for shortest paths
- If nodes (a, b) lie within community, there are too many options for shortest paths to circumvent (a, b) (so (a, b) gets credited only small fractions)

BETWEENNESS: EXAMPLE

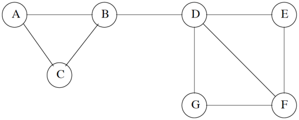


Adopted from `mmds.org`

- (B, D) has the greatest betweenness, 12
  - It is on any shortest path between A, B, C and D, E, F, G
- (D, F) has betweenness 4
  - It lies on all shortest paths between A, B, C, D and F

Algorithmic Principle

- Visit each node X once
- Compute shortest paths from X to any other node Y
- To visit nodes Y from X, perform breadth-first search (BFS)



Social Network; consider BFS from E

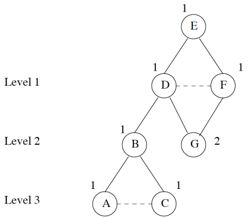
Adopted from `mmds.org`

The Girvan-Newman Algorithm

Calculating Betweenness

Algorithmic Principle

- Visit each node X once
- Compute shortest paths from X to any other node Y
- To visit nodes Y from X, perform breadth-first search (BFS)

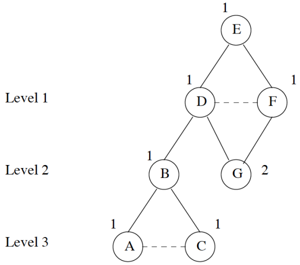


BFS starting from E on social network from slide before

Adopted from `mmds.org`

THE GIRVAN-NEWMAN ALGORITHM

CALCULATING BETWEENNESS



BFS starting from *E*

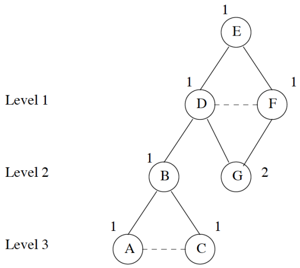
Adopted from `mmds.org`

INTUITION / NOTATION

- Length of shortest path from X to Y: level of BFS starting at X
- Edges within BFS level cannot be part of shortest paths from X
- Edges between different levels are referred to as *DAG (directed acyclic graph) edges*
- DAG edges are on at least one shortest path leaving from X

THE GIRVAN-NEWMAN ALGORITHM

CALCULATING BETWEENNESS



BFS starting from E

Adopted from `mmds.org`

EXAMPLE NOTATION

- Solid edges = DAG edges: e.g. (D, B), (E, F)
- Dashed edges = within level: e.g. (D, F), (A, C)
- For DAG edge (Y, Z) where Y is closer to root X than Z:
  - Y is said to be the *parent*
  - Z is said to be the *child*

The Girvan-Newman Algorithm

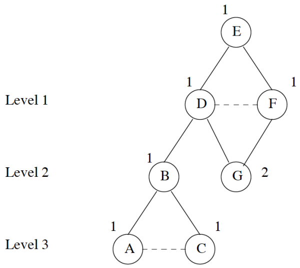
Calculating Betweenness

Two Stages

- *Labeling:* For each node, assign number of shortest paths from root to that node
  - Proceed from root to leaves in BFS order
- *Crediting:* For each edge, compute contribution of shortest paths from root to betweenness of that edge
  - Need to compute credits for nodes as well
  - Proceed from leaves to root, bottom-up

THE GIRVAN-NEWMAN ALGORITHM

CALCULATING BETWEENNESS



BFS starting from *E*

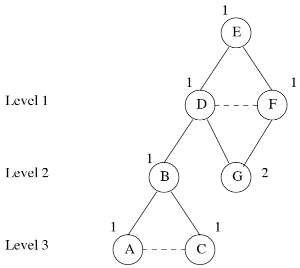
Adopted from `mmds.org`

LABELING NODES

- Label each node by the number of shortest path to the root
- Start by labeling the root with 1
- Top-down, label each node by the sum of labels of each parents

THE GIRVAN-NEWMAN ALGORITHM

CALCULATING BETWEENNESS



BFS starting from *E*: Labeling

Adopted from `mmds.org`

EXAMPLE LABELING

- Label the *root* *E* with 1
- *Level 1:* Each *D* and *F* have only *E* as parent; label both with 1
- *Level 2:*
  - *B* has only *D* as parent, label with 1
  - *G* has parents *D* and *F*, label with 2
- *Level 3:* Both *A*, *C* have only *B* as parent, so both are labeled with 1

Crediting Nodes

- Credit each *leaf* with 1
- Each *non-leaf node* v gets credit

$$1 + \sum_{e \in \mathcal{D}(v)} c(e) \tag{1}$$

where $\mathcal{D}(v)$ are the DAG edges leaving from v, and $c(e)$ is the credit of an edge e

How to credit edges?

Crediting Edges

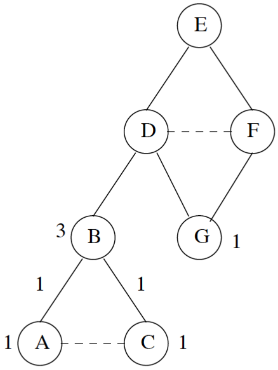
- Let $u_j, j = 1, ..., k$ be the parents of w; so (u_j, w) are the DAG edges entering w
- Let $N_j, j = 1, ..., k$ be the number of shortest paths running through edges (u_j, w)
- Let $c(w)$ be the credit of w
- We compute the credit of (u_i, w) as

$$c(u_i, w) := c(w) \times \frac{N_i}{\sum_{j=1}^{k} N_j} \tag{2}$$

- Note that N_j agrees with the *label* of u_j

The Girvan-Newman Algorithm

Calculating Betweenness



Crediting Nodes and Edges in Level 3 and 2

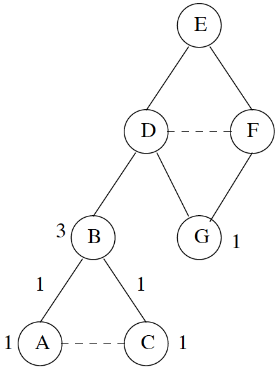
Adopted from `mmds.org`

Example Crediting

- *Level 3 Nodes:* Credit each of nodes A and C with 1
- *Level 2-3 Edges:* Both A and C have only one parent, so full credit 1 is assigned to both (B, A) and (B, C)

The Girvan-Newman Algorithm

Calculating Betweenness



Crediting Nodes and Edges in Level 3 and 2

Adopted from `mmds.org`

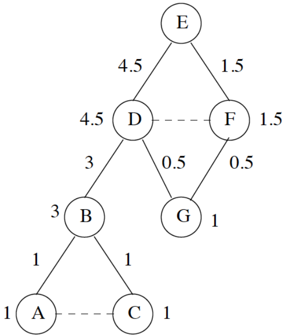
Example Crediting

Level 2 Nodes:

- G is a leaf, so gets credit 1
- B is not a leaf, so gets credit 1 + sum of credits 1 of DAG edges (B, A), (B, C) leaving from it: credit 3 overall
- Intuitively, credit 3 for B refers to all shortest paths from E to A, B, C going through B.

The Girvan-Newman Algorithm

Calculating Betweenness



Crediting Nodes and Edges

Adopted from `mmds.org`

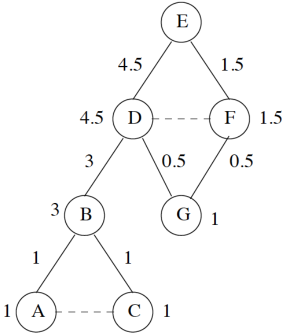
Example Crediting

Level 1-2 Edges:

- B has only one parent, D, so the edge (D, B) gets all of B's credit
- $(D, G), (F, G)$: Both D, F have label (not credit!) 1. So we credit both $(D, G), (F, G)$ with $1/(1 + 1) = 0.5$
- *Example:* If labels of D and F had been 3 and 5, the credit of (D, G) would be $3/(3 + 5) = 3/8$ and that of (F, G) would be $5/8$.

The Girvan-Newman Algorithm

Calculating Betweenness



Crediting Nodes and Edges

Adopted from `mmds.org`

Example Crediting

Level 1 Nodes / Edges:

- D gets credit 1 + credits of (D, B), (D, G) = credit 4.5 overall
- F gets credit 1 + credit of (F, G) = credit 1.5 overall
- Edges (E, D), (E, F) receive credits of D, F respectively, because D, F each have only one parent

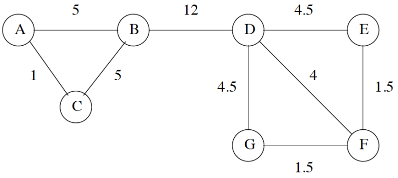
Summary: Credit on each edge is contribution to betweenness of that edge to shortest paths from E

The Girvan-Newman Algorithm

Summary

Completing the Algorithm

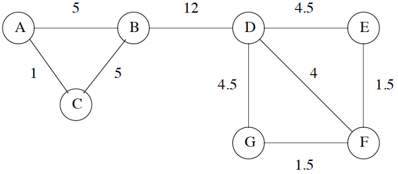
- Repeat the calculation illustrated for E for every other node
- Sum up the contributions for each edge across different roots
- Divide each edge weight by 2: each shortest path is counted twice, with each of its end points as root



Betweenness Scores

Adopted from `mmds.org`

Finding Communities with Betweenness



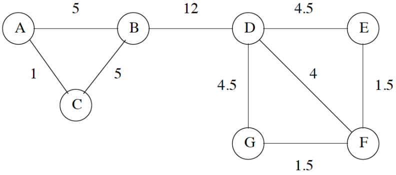
Betweenness Scores

Adopted from `mmds.org`

Computing Communities: Principle

- Remove edges in decreasing order of betweenness
- Stop at reasonably chosen threshold
- Communities are the resulting connected components

Finding Communities with Betweenness



Betweenness Scores

Adopted from `mmds.org`

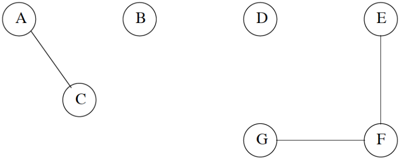
Computing Communities: Example Threshold 4

- First, remove (B, D): communities $\{A, B, C\}, \{D, E, F, G\}$
- Second, remove $(A, B), (B, C)$: communities $\{A, C\}, \{B\}, \{D, E, F, G\}$
- Third, remove $(D, E), (D, G)$: communities $\{A, C\}, \{B\}, \{D, E, F, G\}$
- Last, remove (D, F): communities $\{A, C\}, \{B\}, \{D\}, \{E, F, G\}$

Finding Communities with Betweenness

Computing Communities: Example Threshold 4

- First, remove (B, D): communities $\{A, B, C\}, \{D, E, F, G\}$
- Second, remove $(A, B), (B, C)$: communities $\{A, C\}, \{B\}, \{D, E, F, G\}$
- Third, remove $(D, E), (D, G)$: communities $\{A, C\}, \{B\}, \{D, E, F, G\}$
- Last, remove (D, F): communities $\{A, C\}, \{B\}, \{D\}, \{E, F, G\}$



Final Communities

Adopted from `mmds.org`

General / Further Reading

Literature

- Mining Massive Datasets, Sections 10.1, 10.2
  `http://infolab.stanford.edu/~ullman/mmds/ch10.pdf`