Learning in Big Data Analytics
Lecture 6

Alexander Schönhuth

UNIVERSITÄT BIELEFELD
Faculty of Technology

Bielefeld University
December 22, 2020

Direct Discovery of Communities

INTRODUCTION

- So far, we partitioned graphs into disjoint communities
- But communities might be overlapping
- *Solution:* Determine communities as (induced) subgraphs of a certain type
- Subgraphs should contain unusually large amount of edges
- Will treat two types briefly here:
  - Cliques
  - Complete bipartite subgraphs

FINDING CLIQUES

DEFINITION [INDUCED SUBGRAPH]

Let $G = (V, E)$ be a graph. A subgraph $C = (V' \subset V, E' \subset E)$ is *induced* iff

$$(v', w') \in E \quad \text{implies} \quad (v', w') \in E'$$

for any $v', w' \in V'$.

DEFINITION [CLIQUE]

Let $G = (V, E)$ be a graph.

- An induced subgraph $C = (V', E')$ is called a *clique* iff any pair of nodes in C is connected by an edge.
- A clique $C = (V', E')$ is *maximal* iff extending the clique by any node and its edges implies that the clique property no longer holds.

Communities as Cliques

- *Possible idea:* Determine communities as maximal cliques
- *Caveat:* The number of maximal cliques in a graph may be exponential in the number of nodes
- So, listing all maximal cliques is a computationally demanding problem
- Nevertheless, identifying communities as clique like arrangements is popular

Complete Bipartite Graphs

Definition [(Complete) Bipartite Graphs]

A graph $G = (V, E)$ with vertices V and edges E is referred to as *bipartite* iff

- there are $V_1, V_2 \subset V$ such that

$$V = V_1 \dot\cup V_2 \quad \text{and} \quad E \subset (V_1 \times V_2)$$

- A bipartite graph $G = (V, E)$ is *complete* iff

$$V = V_1 \dot\cup V_2 \quad \text{and} \quad E = (V_1 \times V_2)$$

  that is iff each node from V_1 is connected with each node from V_2

- A complete bipartite graph where $|V_1| = s, |V_2| = t$ is referred to as $K_{s,t}$
- A complete bipartite graph is also referred to as *biclique*

Complete Bipartite Graphs and Communities

- *Strategy:* Seek to discover all sufficiently large bicliques
- Treat them as "nuclei" (or seeds) of communities
- *Theoretical Advantage over Cliques:* While it is not possible to guarantee the existence of large cliques for graphs with many edges, one can guarantee the existence of large bicliques

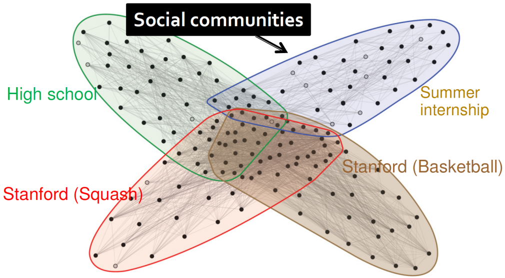
Finding Complete Bipartite Graphs

Frequent Itemset Mining Problem

- Let $G = (V, E)$ on $V = V_1 \dot{\cup} V_2$ be a (large) bipartite graph
- Items are nodes from V_1
- Baskets are nodes from V_2
- Items in baskets are nodes from V_1 connected to basket node
- $K_{s,t}$ in G is itemset of size s that appears in t baskets
- So mining for frequent itemsets at threshold t dicovers all $K_{s,t}$

The Graph Affiliation Model

Overlapping Communities

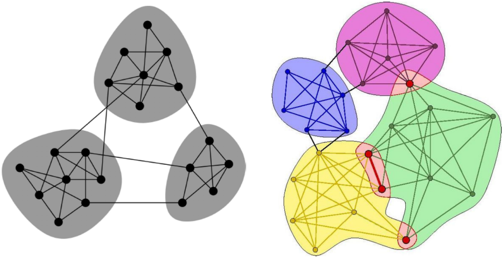


Subgraph from Facebook

Adopted from `mmds.org`

- *Observation:* Communities in social networks can overlap
- Graph partitioning does not help in these cases
- Would like to have a statistical interpretation of network data

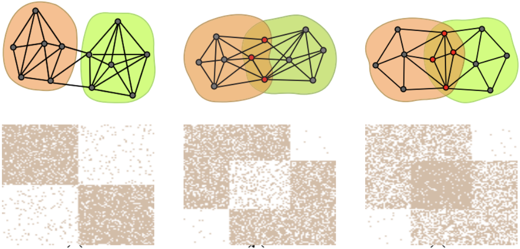
Nonoverlapping versus Overlapping Communities



Left: Nonoverlapping communities
Right: Overlapping communities
Adopted from mmds.org

- Communities may overlap or not
- *Issue:* How to determine communities correctly?

Affiliation Graph Model: Introduction

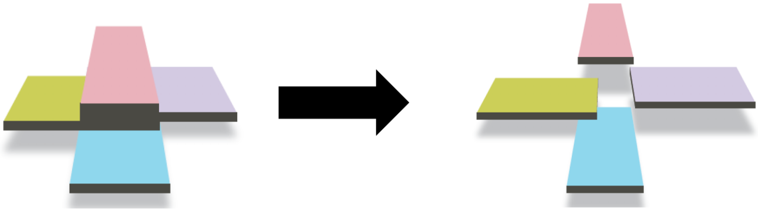


Networks and their adjacency matrices

Adopted from `mmds.org`

- Left: No overlap, adjacency matrix sparse across communities
- Middle: Loose overlap, adjacency matrix less sparse in shared part
- Right: Tight overlap, adjacency matrix dense in shared part

Community Discovery: Goal



Revealing (overlapping) communities

Adopted from `mmds.org`

- *Goal:* Discover communities correctly
- Regardless of whether they overlap or not

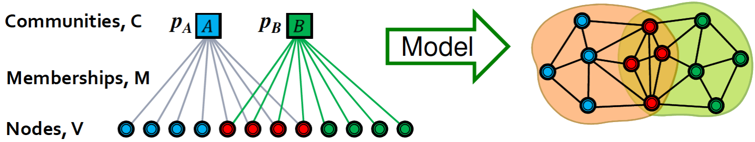
Determine the statistically most likely community structure

Affiliation Graph Model: Introduction

- *Issue:* Statistical control over community structure of a network
- *Idea:* Design *generative probability distribution*
- Given a number of nodes, this generative distribution generates edges
- The generative distribution represents a particular community structure
  - The distribution knows about nodes belonging to communities
  - It generates more edges within communities
  - It generates less edges between communities

Affiliation Graph Model: Introduction

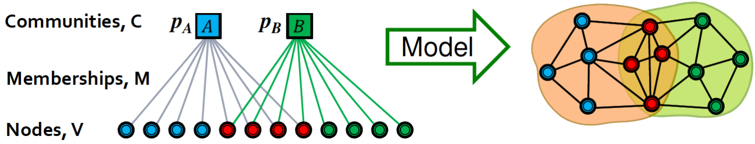
- The generative distribution represents community structures
  - The distribution knows about nodes belonging to communities
  - It generates more edges within communities
  - It generates less edges between communities



Distribution representing a community structure generating network

Adopted from mmds.org

Affiliation Graph Model: Introduction



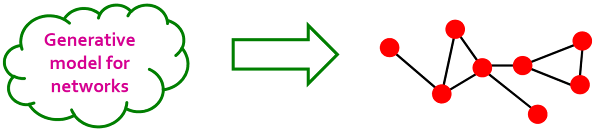
Distribution representing a community structure (left) generating network (right)

Adopted from `mmds.org`

- We can generate networks when knowing community structure
- *But:* We would like to determine the community structure when knowing the network

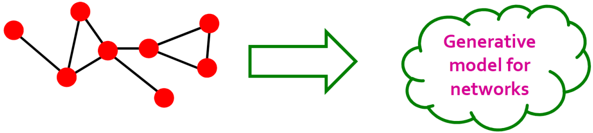
Isn't that exactly the opposite?

Generative Distributions



We can do this: generating network from distribution...

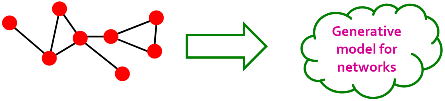
Adopted from `mmds.org`



...but we want this: inferring distribution from network

Adopted from `mmds.org`

GENERATIVE DISTRIBUTIONS: MAXIMUM LIKELIHOOD INFERENCE



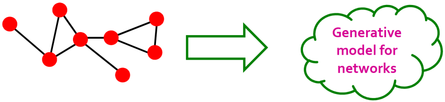
We want to infer distribution from network

Adopted from `mmds.org`

Maximum Likelihood Estimation

- Let Θ be a *parameterized class of probability distributions* that generate networks
  - We identify the different distributions with the different parameterizations
  - ☞ Formally not 100% correct, but doesn't matter here
- Let $\mathbf{P}(N \mid \theta)$ be the probability that distribution $\theta \in \Theta$ generates network N

Generative Distributions: Maximum Likelihood Inference



We want to infer distribution from network

Adopted from `mmds.org`

Maximum Likelihood Estimation

- Let $\mathbf{P}(N \mid \theta)$ be the probability that distribution $\theta \in \Theta$ generates network N
- *Maximum likelihood estimation:* Determine distribution $\hat{\theta}$ that generated N with greatest likelihood:

$$\hat{\theta} := \arg\max_{\theta \in \Theta} \mathbf{P}(N \mid \theta) \tag{1}$$

- This computes most reasonable distribution $\hat{\theta}$ for network N

Affiliation Graph Model: Definition I

- An AGM θ generates a network $N = (V, E)$ by adding edges E to a given set of nodes V
- For $u, v \in V$, edge (u, v) is generated with probability $\mathbf{P}_\theta((u, v))$
- $\mathbf{P}_\theta((u, v))$ depends on the parameters θ
- Recall that θ specifies community structure

So, what exactly is θ supposed to be?

AFFILIATION GRAPH MODEL: PARAMETERS

- $\mathcal{C}$, as a set of *communities*
- $M \in \{0,1\}^{\mathcal{C} \times V}$, specifying *assignment of nodes $v \in V$ to communities* $C \in \mathcal{C}$, where

$$M_{C,v} = \begin{cases} 1 & v \text{ belongs to } C \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

  - M specifies "affiliations" of nodes $v \in V$
  - Note that one can vary $\mathcal{C}$, as a parameter, but not V
- $(p_C)_{C \in \mathcal{C}}$ as probabilities to generate edges (u, v) because $u, v \in C$
- *Summary:* A particular AGM θ corresponds to

$$\theta = (\mathcal{C}, M, (p_C)_{C \in \mathcal{C}}) \tag{3}$$

Affiliation Graph Model: $\mathbf{P}_\theta((u,v))$

Several C containing both u, v

- Let $M_u, M_v \subset \mathcal{C}$ be the subsets of communities that contain u and v, respectively
- Existence of communities that contain both u, v means

$$M_u \cap M_v \neq \emptyset$$

- Memberships in different communities have no influence on each other
- That is, we assume *statistical independence*

AFFILIATION GRAPH MODEL: $\mathbf{P}_\theta((u, v))$

Several C **containing both** u, v

- Statistical independence is expressed by

$$\prod_{C \in M_u \cap M_v} (1 - p_C)$$

as probability of *no edge* (u, v) *in any community* $C \in M_u \cap M_v$

- Hence, the probability to generate (u, v) is

$$1 - \prod_{C \in M_u \cap M_v} (1 - p_C) \tag{4}$$

Done? No: What about $M_u \cap M_v = \emptyset$?

Affiliation Graph Model: $\mathbf{P}_\theta((u,v))$

No C containing both u, v

- For $M_u \cap M_v = \emptyset$, computing (4) yields (empty product is 1)

$$1 - \prod_{C \in \emptyset} (1 - p_C) = 1 - 1 = 0$$

- No edges across communities makes no sense
- Let $\epsilon > 0$ be small; we generate an edge (u, v) with probability

$$\mathbf{P}_\theta((u,v)) = \epsilon \quad \text{if} \quad M_u \cap M_v = \emptyset$$

Affiliation Graph Model: $\mathbf{P}_\theta((u,v))$

Affiliation Graph Model (AGM)

- An edge (u,v) is generated with probability

$$\mathbf{P}_\theta((u,v)) = \begin{cases} 1 - \prod_{C \in M_u \cap M_v}(1-p_C) & M_u \cap M_v \neq \emptyset \\ \epsilon & M_u \cap M_v = \emptyset \end{cases} \tag{5}$$

- Edges (u,v) are generated independently from one another
- *Overall:* The probability $\mathbf{P}_\theta(E)$ to generate edges E given AGM θ computes as

$$\mathbf{P}_\theta(E) = \prod_{(u,v)\in E} \mathbf{P}_\theta((u,v)) \times \prod_{(u,v)\notin E} 1 - \mathbf{P}_\theta((u,v)) \tag{6}$$

where $\mathbf{P}_\theta((u,v))$ are computed following (5), with $\theta = (\mathcal{C}, M, p_C)$ determining p_C and M_u, M_v and so on.

Affiliation Graph Model: Overall Probability

Affiliation Graph Model (AGM)

- The probability $\mathbf{P}_\theta(E)$ to generate E given θ is

$$\mathbf{P}_\theta(E) = \prod_{(u,v)\in E} \mathbf{P}_\theta((u,v)) \times \prod_{(u,v)\notin E} 1 - \mathbf{P}_\theta((u,v)) \quad (7)$$

- *Reminder:* For a given network $N = (V, E)$, the *goal* is to determine

$$\hat{\theta} := \arg\max_{\theta\in\Theta} \mathbf{P}_\theta(E)$$

- That is, we need to vary $\theta = (\mathcal{C}, M, p_C)$ until $\mathbf{P}_\theta(E)$ is maximal

How to *systematically vary* $\theta = (\mathcal{C}, M, p_C)$?

Computing the MLE $\hat{\theta}$

Issues

- Search space of combinations of
  - Communities $\mathcal{C}$,
  - Assignments of nodes to communities M, and
  - Probabilities p_C for communities

  tends to be huge
- Concise formulas of (7) for $\mathbf{P}_\theta(E)$ as function of θ too difficult
- Analytical solution for determining $\hat{\theta} := \arg\max_{\theta \in \Theta} \mathbf{P}_\theta(E)$ not available
- Moreover, parameters are both discrete ($\mathcal{C}, M$) and continuous ($(p_C)_{C \in \mathcal{C}}$)

Computing the MLE $\hat{\theta}$

Approach

1. Pick initial set of parameters θ_0
2. Vary θ such that $\mathbf{P}_\theta(E)$ iteratively increases
3. Vary $\mathcal{C}$ or M first
   - ☞ Partial derivates of $\mathbf{P}_\theta(E)$ wrt. p_C computable on fixed $\mathcal{C}, M$
4. Determine optimal $(p_C)_{C \in \mathcal{C}}$, e.g. by gradient descent
5. Keep change if $\mathbf{P}_\theta(E)$ has increased, discard otherwise

Computing the MLE $\hat{\theta}$

Iterative variations of $\mathcal{C}, M$

- *Varying M:*
  - Delete node from community, i.e. for $M_{C,v} = 1$, set $M_{C,v} = 0$
  - Add node to community, i.e. for $M_{C,v} = 0$, set $M_{C,v} = 1$
- *Varying $\mathcal{C}$:*
  - Merge two communities
  - Split community
  - Delete community
  - Add new community, with initial random selection of members

Computing the MLE $\hat{\theta}$

Soft Community Membership

- Instead of $M_{C,v} \in \{0,1\}$, allow any real-numbered $M_{C,v} \geq 0$
- For (u,v) to be generated because of $u, v \in C$, let

$$\mathbf{P}_\theta((u,v)) = 1 - e^{-M_{C,u} M_{C,v}} \tag{8}$$

  be the individual probability
- Proceeding exactly as before, we obtain

$$\mathbf{P}_\theta(E) = \prod_{(u,v) \in E} (1 - e^{-\sum_C M_{C,u} M_{C,v}}) \prod_{(u,v) \notin E} e^{-\sum_C M_{C,u} M_{C,v}} \tag{9}$$

Computing the MLE $\hat{\theta}$

Soft Community Membership

- Probability for edges E:

$$\mathbf{P}_\theta(E) = \prod_{(u,v)\in E} (1 - e^{-\sum_C M_{C,u} M_{C,v}}) \prod_{(u,v)\notin E} e^{-\sum_C M_{C,u} M_{C,v}} \tag{10}$$

- On fixed communities, include M in gradient descent (or related) optimization step
- *Advantages:*
  - Only one gradient descent run necessary
  - Less prone to get stuck in unfavorable local optima
- If necessary, add or delete communities, and re-run

General / Further Reading

Literature

- Mining Massive Datasets, Sections 10.3, 10.5
  `http://infolab.stanford.edu/~ullman/mmds/ch10.pdf`