Graph Clustering and Community Detection

Ji Wang and Krunal Doshi

Nov 11, 2013
Betweenness Measures and Graph Partitioning (E-K*, 3.6)

Presented by: Ji Wang

Network and Communities

• In our mind, we always think the networks look like this way:

![Network Diagram](http://i.imgur.com/k0pv0.jpg)

Reddit: [http://i.imgur.com/k0pv0.jpg](http://i.imgur.com/k0pv0.jpg)
Network and Communities

• Question is Why?
  – Why we have this idea for social networks?
  – How it actually works in social networks?

• Answers in E-K:
  – Strong Ties
  – Weak Ties

Figure 3.11: The contrast between densely-knit groups and boundary-spanning links is reflected in the different positions of nodes A and B in the underlying social network. There is a lot of further insight to be gained by asking about the roles that different nodes play in this structure as well. In social networks, access to edges that span different groups is not equally distributed across all nodes: some nodes are positioned at the interface between multiple groups, with access to boundary-spanning edges, while others are positioned in the middle of a single group. What is the effect of this heterogeneity?

Following the positional lead of social network researchers including Ron Burt [87], we can formulate an answer to this question as a story about the different experiences that nodes have in a network like the one in Figure 3.11—particularly in the contrast between the experience of a node such as A, who sits at the center of a tightly-knit group, and node B, who sits at the interface between several groups.

Embeddedness. Let's start with node A. Node A's set of network neighbors has been subject to considerable triadic closure; A has a high clustering coefficient. (Recall that the clustering coefficient is the fraction of pairs of neighbors who are themselves neighbors.) To talk about the structure around A it is useful to introduce an additional definition. We define the embeddedness of an edge in a network to be the number of common neighbors the two endpoints have. Thus, for example, the A-B edge has an embeddedness of two, since A and B have the two common neighbors E and F. This definition relates to two notions from earlier in the chapter. First, the embeddedness of an edge is equal to the numerator in...
Strong Ties and Weak Ties

• Mark Granovetter (1960s)
  – How people find out about new jobs
  – People find the info through personal contacts
  – But, Contacts were often acquaintance rather than close friends

• Two views of friendships in personal contacts
  – Structural: Friendships span different parts of network
  – Interpersonal: Friendship between two people is Strong or Weak

Strong Ties and Weak Ties

• Granovetter defined the connection between social role and structural role of an edge

• In network structure view:
  – Structurally embedded edges are also socially strong
  – Edges spanning different parts of the network are socially weak

• In social phenomenon view:
  – The long range edges allow you to gather information from different parts of the network and get a job
  – Structurally embedded edges are heavily redundant in terms of information access
Strong Ties and Weak Ties

Figure 3.11: The contrast between densely-knit groups and boundary-spanning links is reflected in the different positions of nodes A and B in the underlying social network.

There is a lot of further insight to be gained by asking about the roles that different nodes play in this structure as well. In social networks, access to edges that span different groups is not equally distributed across all nodes: some nodes are positioned at the interface between multiple groups, with access to boundary-spanning edges, while others are positioned in the middle of a single group. What is the effect of this heterogeneity?

Following the positional lead of social network researchers including Ron Burt [87], we can formulate an answer to this question as a story about the different experiences that nodes have in a network like the one in Figure 3.11—particularly in the contrast between the experience of a node such as A, who sits at the center of a tightly-knit group, and node B, who sits at the interface between several groups.

Embeddedness. Let’s start with node A. Node A’s set of network neighbors has been subject to considerable triadic closure; A has a high clustering coefficient. (Recall that the clustering coefficient is the fraction of pairs of neighbors who are themselves neighbors.)

To talk about the structure around A it is useful to introduce an additional definition. We define the embeddedness of an edge in a network to be the number of common neighbors the two endpoints have. Thus, for example, the A-B edge has an embeddedness of two, since A and B have the two common neighbors E and F. This definition relates to two notions from earlier in the chapter. First, the embeddedness of an edge is equal to the numerator in

We define the embeddedness of an edge in a network to be the number of common neighbors the two endpoints have. Thus, for example, the A-B edge has an embeddedness of two, since A and B have the two common neighbors E and F. This definition relates to two notions from earlier in the chapter. First, the embeddedness of an edge is equal to the numerator in

Weak Ties

Strong Ties
Tie Strengths in Facebook Network

The number of links corresponding to maintained relationships, one-way communication, and reciprocal communication as a function of the total neighborhood size for users on Facebook.

Tie Strengths in Twitter Network

The total number of a user’s strong ties (defined by multiple directed messages) as a function of the number of followees he or she has on Twitter.

My Questions about Weak Ties

• Based on our life experience, FB and Twitter analysis results, do Weak Ties really work?
  – Most of our friendship are Weak Ties
  – Most of new info is from these connections
  – So, we have high chance to get “new job” info, right?

• What’s the role of Weak ties?

• How to define Weak Ties?

• How to evaluate and measure the information flow in real social network?
Network Communities

• Granovetter’s theory tells us that network are composed by tightly connected sets of nodes.

• What is network communities?
  – Sets of nodes
  – Which have more inside connections than outside connections
  – We can refer communities as groups, modules, clusters...

Stanford CS224W:
http://snap.stanford.edu/class/cs224w-2012/handouts.html
Network Communities

• How to automatically find communities in complex networks?
• Communities Detection in networks:
  – Undirected
  – Unweighted
A Network Example

3.6. ADVANCED MATERIAL: BETWEENNESS MEASURES AND GRAPH PARTITIONING

1
2
3
6
4
5
7
8
9
10
11
12
13
14

(a) A sample network

(b) Tightly-knit regions and their nested structure

Figure 3.14: In many networks, there are tightly-knit regions that are intuitively apparent, and they can even display a nested structure, with smaller regions nesting inside larger ones.

The Notion of Betweenness.

To motivate the design of a divisive method for graph partitioning, let's think about some general principles that might lead us to remove the 7-8 edge first in Figure 3.14(a).

A first idea, motivated by the discussion earlier in this chapter, is that since bridges and local bridges often connect weakly in interacting parts of the network, we should try removing these bridges and local bridges first. This is in fact an idea along the right lines; the problem is simply that it's not strong enough, for two reasons. First, when there are several bridges, it doesn't tell us which to remove first. As we see in Figure 3.14(a), where there are five bridges, certain bridges can produce more reasonable splits than others. Second, there can be graphs where no edge is even a local bridge, because every edge belongs to a triangle — yet there is still a natural division into regions. Figure 3.15 shows a simple example, where we might want to identify nodes 1-5 and nodes 7-11 as tightly-knit regions, despite...
3.6. ADVANCED MATERIAL: BETWEENNESS MEASURES AND GRAPH PARTITIONING

1 2 3 4 5 6 7 8 9 10 11 12 13 14

(a) A sample network

(b) Tightly-knit regions and their nested structure.

Figure 3.14: In many networks, there are tightly-knit regions that are intuitively apparent, and they can even display a nested structure, with smaller regions nesting inside larger ones.

The Notion of Betweenness. To motivate the design of a divisive method for graph partitioning, let's think about some general principles that might lead us to remove the 7-8 edge first in Figure 3.14(a).

A first idea, motivated by the discussion earlier in this chapter, is that since bridges and local bridges often connect weakly in interacting parts of the network, we should try removing these bridges and local bridges first. This is in fact an idea along the right lines; the problem is simply that it's not strong enough, for two reasons. First, when there are several bridges, it doesn't tell us which to remove first. As we see in Figure 3.14(a), where there are five bridges, certain bridges can produce more reasonable splits than others. Second, there can be graphs where no edge is even a local bridge, because every edge belongs to a triangle—and yet there is still a natural division into regions. Figure 3.15 shows a simple example, where we might want to identify nodes 1-5 and nodes 7-11 as tightly-knit regions, despite...
A Method: Strength of Weak Ties

• How to cut network to different communities?
• Edge Betweenness (EB):
  – Number of shortest path passing over the edge
  – If EB of an edge is high, it has high chance to be a weak tie. And it can be removed to distinguish different communities.
Girvan-Newman Algorithm

• Based on the notion of edge betweenness
  – Number of shortest paths via the edge
• Working on undirected unweighted networks
• Repeat until no edge left
  – Calculate betweenness of edges in networks
    • Need to re-calculate betweenness in every step
  – Delete the edges with highest betweenness value
Calculate EB in this network
Run G-N Algorithm Step by Step
One More Example

(a) Step 1

(b) Step 2

(c) Step 3

(d) Step 4
How to Compute Betweennessness?

(a) A sample network
(b) Breadth-first search starting at node A

Breadth Frist Search (BFS) starting from node A
Count the number of shortest paths from node A to all other nodes of the network:

# shortest A-K paths = # shortest A-I paths + # shortest A-J paths

# shortest A-J paths = # shortest A-G paths + # shortest A-H paths

# shortest A-I paths = # shortest A-F paths + # shortest A-G paths

Each node in the first layer is a neighbor of A, and so it has only one shortest path from A: the edge leading straight from A to it. So we give each of these nodes a count of 1.

Now, as we move down through the BFS layers, we apply the reasoning discussed above to conclude that the number of shortest paths to
Figure 3.20: The final step in computing betweenness values is to determine the floor values from a starting node A to all other nodes in the network. This is done by working up from the lowest layers of the breadth-first search, dividing up the floor value at a node in proportion to the number of shortest paths coming into it on each edge. Each node should be the sum of the number of shortest paths to all nodes directly above it in the breadth-first search. Working down through the layers, we thus get the number of shortest paths to each node, as shown in Figure 3.19. Note that by the time we get to deeper layers, it may not be so easy to determine these numbers by visual inspection—for example, to immediately list the six different shortest paths from A to K—but it is quite easy when they are built up layer-by-layer in this way.

Determine floor values. Finally, we come to the third step, computing how the flow from A to all other nodes spreads out across the edges. Here too we use the breadth-first search structure, but this time working up from the lowest layers. We first show the idea in Figure 3.20 on our running example, and then describe the general procedure.

- Let's start at the bottom with node K. A single unit of flow arrives at K, and an equal number of the shortest paths from A to K come through nodes I and J, so this unit of flow arrives at K, and an equal number of the shortest paths from A to K come through nodes I and J, so this unit...
Something Left?

• How to define the communities in formal way
  – Like a value or function?
• When to stop the Girvan-Newman Algorithm
• Answer:
  – Modularity \( Q \): A measure of how well a network is partitioned into communities
  – Given a partitioning of the network into groups \( s \in S \)

\[
Q \propto \sum_{s \in S} \left[ (\text{# edges within group } s) - \left(\text{expected # edges within group } s\right) \right]
\]

Need a null model!

Stanford CS224W:
http://snap.stanford.edu/class/cs224w-2012/handouts.html
Something Left?

• Check this reference
#2 Community Detection: Trawling

Presented by: Ji Wang

Trawling

- Search for small communities in a web graph
  - One and half years’ web archive in 1998.
- Find the signature of a community in a Web Graph
  - Many people are discussing about the same things

Use this to define “topics”:
- What the same people on the left talk about on the right
- Remember HITS!
Trawling

• Enumerate complete bipartite subgraphs $K_{s,t}$
• Where $K_{s,t}$: $s$ nodes on the “left” where each links to the same $t$ other nodes on the “right”

\[ |X| = s = 3 \]
\[ |Y| = t = 4 \]

Stanford CS224W:
http://snap.stanford.edu/class/cs224w-2012/handouts.html
Trawling

• How to find Complete Bipartite Graphs
  – View each node \(i\) as a set \(S_i\) of nodes \(i\) point to
  – \(K_{s,t}\) = a set \(Y\) of size \(t\) that occurs in \(s\) sets \(S_i\)
  – Looking for \(K_{s,t}\) set of frequency threshold to \(s\) and look at layer \(t\) – all frequent sets of size \(t\)

Stanford CS224W:
http://snap.stanford.edu/class/cs224w-2012/handouts.html
Summary of Trawling

• **Analytical result:**
  – Complete bipartite subgraphs $K_{s,t}$ are embedded in larger dense enough graphs (*i.e.*, the communities)
  – Bipartite subgraphs act as “signatures” of communities

• **Algorithmic result:**
  – Frequent itemset extraction and dynamic programming finds graphs $K_{s,t}$

Stanford CS224W:
http://snap.stanford.edu/class/cs224w-2012/handouts.html
#3 Spectral Clustering

Presented by: Krunal

Spectral Clustering

• **Given:**
  – Data points $X_1, ..., X_n$
  – Some function to find pairwise similarities

• **Objective of a clustering algorithm:**
  – Points in the same group are similar
  – Points in different groups are dissimilar
Similarity Graph & Notations

Undirected graph: \( G = (V,E) \)

- Data points \( X_i \) as Vertex set: \( V = \{v_1, ..., v_n\} \)
- Weighted adjacency matrix: \( W = (w_{ij}) \ i, j = 1, ..., n \quad w_{ij} \geq 0 \)
- Degree: \( d_i = \sum_{j=1}^{n} w_{ij} \)
- Degree matrix: \( D = \text{Diagonal matrix} \ (d_i) \)

- For a subset \( A \subset V \)
  - the number of vertices in \( A \) = \( |A| \)
  - Volume of subset, \( \text{vol}(A) := \sum_{i \in A} d_i \)
Different Similarity Graphs

• \textbf{\(\varepsilon\)-neighborhood}
  – Connect all points whose pairwise distance is less than \(\varepsilon\)

• \textbf{k-nearest neighbors}
  – Symmetric knn \(\rightarrow\) \(e_{(vi,vj)} \in E : \text{if} \ \text{knn}(v_i) \ \text{OR} \ \text{knn}(v_j)\)
  – Mutual knn \(\rightarrow\) \(e_{(vi,vj)} \in E : \text{if} \ \text{knn}(v_i) \ \text{AND} \ \text{knn}(v_j)\)

• \textbf{Fully connected}
  – Similarity function like Guassian: \(\exp(-\|x_i-x_j\|^2/(2\sigma^2))\)
  – \textit{All edges with similarity >0 are connected.}
Graph Laplacians

- **Unormalized Graph Laplacian**

\[ L = D - W \]

1. **Main property of matrix L:**

For every \( f \in \mathbb{R}^n \) we have,

\[ f'Lf = \frac{1}{2} \sum_{i,j=1}^{n} w_{ij}^2 (f_i - f_j)^2 \]

**Proof:**

\[
\begin{align*}
    f'Lf &= f'Df - f'Wf \\
    &= \sum_{i=1}^{n} d_i f_i^2 - \sum_{i,j=1}^{n} f_i f_j w_{ij} \\
    &= \frac{1}{2} \left( \sum_{i=1}^{n} d_i f_i^2 - 2 \sum_{i,j=1}^{n} f_i f_j w_{ij} + \sum_{j=1}^{n} d_j f_j^2 \right) \\
    &= \frac{1}{2} \sum_{i,j=1}^{n} w_{ij} (f_i - f_j)^2
\end{align*}
\]
Other properties

1. $L$ is symmetric and positive semi-definite.

2. The smallest eigenvalue of $L$ is 0, the corresponding eigenvector is the constant one vector $\mathbf{1}$

3. $L$ has $n$ non-negative, real-valued eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$.

4. The multiplicity $k$ of the eigenvalue 0 of $L$ equals the number of connected components $A_1, \ldots, A_k$ in the graph. The eigenspace of eigenvalue 0 is spanned by the indicator vectors $\mathbbm{1}_{A_1}, \ldots, \mathbbm{1}_{A_k}$ of those components.
Normalized Graph Laplacians

Random walk,

\[ L_{rw} := D^{-1}L = I - D^{-1}W \]

Symmetric matrix,

\[ L_{sym} := D^{-\frac{1}{2}}LD^{-\frac{1}{2}} = I - D^{-\frac{1}{2}}WD^{-\frac{1}{2}} \]
Properties of Normalized Graph Laplacian

1. For every $f \in \mathbb{R}^n$ we have
   \[ f' L_{\text{sym}} f = \frac{1}{2} \sum_{i,j=1}^{n} w_{ij}^2 \left( \frac{f_i}{\sqrt{d_i}} - \frac{f_j}{\sqrt{d_j}} \right)^2 \]

2. $\lambda$ is an eigenvalue of $L_{\text{rw}}$ with eigenvector $u$ if and only if $\lambda$ is an eigenvalue of $L_{\text{sym}}$ with eigenvector $w = D^{1/2}u$.

3. $\lambda$ is an eigenvalue of $L_{\text{rw}}$ with eigenvector $u$ if and only if $\lambda$ and $u$ solve the generalized eigen problem $Lu = \lambda Du$.

4. 0 is an eigenvalue of $L_{\text{rw}}$ with the constant one vector $1$ as eigenvector. 0 is an eigenvalue of $L_{\text{sym}}$ with eigenvector $D^{1/2}1$.

5. $L_{\text{sym}}$ and $L_{\text{rw}}$ are positive semi-definite and have $n$ non-negative real-valued eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$.

6. Then the multiplicity $k$ of the eigenvalue 0 of both $L_{\text{sym}}$ and $L_{\text{rw}}$ equals the number of connected components $A_1, \ldots, A_k$ in the graph. For $L_{\text{rw}}$ the eigenspace of 0 is spanned by the indicator vectors $1_{A_i}$ of those components. For $L_{\text{sym}}$, the eigenspace of 0 is spanned by the vectors $D^{1/2}1_{A_i}$. 
Spectral Clustering algorithm
I. Unnormalized Graph Laplacian

\[ L = D - W \]

---

Unnormalized spectral clustering

Input: Similarity matrix \( S \in \mathbb{R}^{n \times n} \), number \( k \) of clusters to construct.

- Construct a similarity graph by one of the ways described in Section 2. Let \( W \) be its weighted adjacency matrix.
- Compute the unnormalized Laplacian \( L \).
- Compute the first \( k \) eigenvectors \( u_1, \ldots, u_k \) of \( L \).
- Let \( U \in \mathbb{R}^{n \times k} \) be the matrix containing the vectors \( u_1, \ldots, u_k \) as columns.
- For \( i = 1, \ldots, n \), let \( y_i \in \mathbb{R}^k \) be the vector corresponding to the \( i \)-th row of \( U \).
- Cluster the points \( (y_i)_{i=1, \ldots, n} \) in \( \mathbb{R}^k \) with the \( k \)-means algorithm into clusters \( C_1, \ldots, C_k \).

Output: Clusters \( A_1, \ldots, A_k \) with \( A_i = \{ j \mid y_j \in C_i \} \).
II. Normalized Graph Laplacian -1

\[ L_{rw} := D^{-1}L = I - D^{-1}W \]

Normalized spectral clustering according to Shi and Malik (2000)

Input: Similarity matrix \( S \in \mathbb{R}^{n \times n} \), number \( k \) of clusters to construct.
- Construct a similarity graph by one of the ways described in Section 2. Let \( W \) be its weighted adjacency matrix.
- Compute the unnormalized Laplacian \( L \).
- Compute the first \( k \) generalized eigenvectors \( u_1, \ldots, u_k \) of the generalized eigenproblem \( Lu = \lambda Du \).
- Let \( U \in \mathbb{R}^{n \times k} \) be the matrix containing the vectors \( u_1, \ldots, u_k \) as columns.
- For \( i = 1, \ldots, n \), let \( y_i \in \mathbb{R}^k \) be the vector corresponding to the \( i \)-th row of \( U \).
- Cluster the points \((y_i)_{i=1,\ldots,n}\) in \( \mathbb{R}^k \) with the \( k \)-means algorithm into clusters \( C_1, \ldots, C_k \).

Output: Clusters \( A_1, \ldots, A_k \) with \( A_i = \{j \mid y_j \in C_i\} \).
III. Normalized Graph Laplacian - 2

\[ L_{sym} := D^{-\frac{1}{2}}LD^{-\frac{1}{2}} = I - D^{-\frac{1}{2}}WD^{-\frac{1}{2}} \]

Normalized spectral clustering according to Ng, Jordan, and Weiss (2002)

Input: Similarity matrix \( S \in \mathbb{R}^{n \times n} \), number \( k \) of clusters to construct.
- Construct a similarity graph by one of the ways described in Section 2. Let \( W \) be its weighted adjacency matrix.
- Compute the normalized Laplacian \( L_{sym} \).
- Compute the first \( k \) eigenvectors \( u_1, \ldots, u_k \) of \( L_{sym} \).
- Let \( U \in \mathbb{R}^{n \times k} \) be the matrix containing the vectors \( u_1, \ldots, u_k \) as columns.
- Form the matrix \( T \in \mathbb{R}^{n \times k} \) from \( U \) by normalizing the rows to norm 1, that is set \( t_{ij} = u_{ij}/(\sum_k u_{ik}^2)^{1/2} \).
- For \( i = 1, \ldots, n \), let \( y_i \in \mathbb{R}^k \) be the vector corresponding to the \( i \)-th row of \( T \).
- Cluster the points \( (y_i)_{i=1,\ldots,n} \) with the \( k \)-means algorithm into clusters \( C_1, \ldots, C_k \).

Output: Clusters \( A_1, \ldots, A_k \) with \( A_i = \{j \mid y_j \in C_i \} \).
Graph Cut Point of View

• **Goal**: Find a partition of the graph such that:
  – edges **within** a group have **high weights**
  – edges **across** different groups have **low weights**

• **MinCut**: choosing a partition $A_1, A_2, \ldots, A_K$ which minimizes

$$
cut(A_1, \ldots, A_k) := \frac{1}{2} \sum_{i=1}^{k} W(A_i, \overline{A_i})
$$
Sensitive to outliers

What we get

What we want

Source & credit: Arik Azran
• Balanced Cuts:

$$\text{RatioCut}(A_1, \ldots, A_k) := \frac{1}{2} \sum_{i=1}^{k} \frac{W(A_i, \overline{A_i})}{|A_i|} = \sum_{i=1}^{k} \frac{\text{cut}(A_i, \overline{A_i})}{|A_i|}$$

$$\text{Ncut}(A_1, \ldots, A_k) := \frac{1}{2} \sum_{i=1}^{k} \frac{W(A_i, \overline{A_i})}{\text{vol}(A_i)} = \sum_{i=1}^{k} \frac{\text{cut}(A_i, \overline{A_i})}{\text{vol}(A_i)}$$

MinCut can be solved efficiently, but RatioCut or Ncut is NP hard.
• Eigenvectors of $L$ for unnormalized spectral is same as relaxed \textit{RatioCut} problem.

• Eigenvectors of $L_{rw}$ for unnormalized spectral is same as relaxed \textit{Ncut} problem.
Random Walk point of View

• Random walk: A stochastic process in which there are randomly jumps from one vertex to another of a Graph.

• Clustering Problem Translation: Finding a partition such that a random walk stays long within a cluster and seldom jumps between clusters.
• Transition probability $p_{ij}$ of jumping from $v_i$ to $v_j$ is given by:

$$p_{ij} = \frac{w_{ij}}{d_i}$$

• So transition matrix

$$P = D^{-1}W$$

• Relationship between $L_{rw}$ and $P$

$$L_{rw} = I - P$$
Spectral clustering and commute distance

- The commute distance $c_{ij}$ between two vertices $v_i$ and $v_j$ is the expected time it takes for the random walk to travel from vertex $v_i$ to vertex $v_j$ and back.

- Intuitively, the commute distance between two vertices decrease if there are many different short paths from vertex $v_i$ to vertex $v_j$.

- Can be expressed in terms of the pseudo-inverse of the unnormalized graph Laplacian.

\[ c_{ij} = (e_i - e_j)' L^\dagger (e_i - e_j) \]

- Spectral clustering is intuitively based on commute distance.
Perturbation Theory point of view

- Perturbation theory studies the question of how eigenvalues and eigenvectors of a matrix $A$ change if we add a small perturbation $H$.

- For graph Laplacian, the smaller the perturbation $H = L - \tilde{L}$ leads to the larger the eigengap $|\lambda_k - \lambda_{k+1}|$.

- As a side note, this eigengap can be useful in selecting cluster size $k$. 
Recommendation for Practical usage

• Similarity graph construction: Use k-nearest neighbour similarity matrix so that graph is sparse but still well connected.

• Graph laplacian preference: $L_{rw}$

• Cluster size k selection: Eigengap heuristic can be used.
References

- Reddit: [http://i.imgur.com/k0pv0.jpg](http://i.imgur.com/k0pv0.jpg)
- [videolectures.net/ulrike_von_luxburg/](http://videolectures.net/ulrike_von_luxburg/)
- [people.cs.pitt.edu/~milos/courses/cs3750/lectures/](http://people.cs.pitt.edu/~milos/courses/cs3750/lectures/)