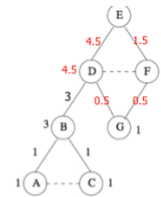


## Community detection:

### 1. Clustering by edge betweenness

- a. Edge betweenness: number of shortest passing over the edge.
  - i. High means an edge is important link connect groups
  - ii. Low is not important pass
- b. The girvan-newman algorithm:
  - i. Computer edge betweenness
  - ii. Remove the edge with the highest betweenness
  - iii. Recalculate edge betweenness
  - iv. Repeat the process until the network is splits into meaningful communities

#### Girvan-Newman Algorithm (6)



- In this case, **both D and F have just one shortest path from E to each of those nodes**
  - So, **give half credit of node G to each of those edges**
  - Credit =  $1/(1 + 1) = 0.5$
- In general, how we distribute credit of a node to its edges depends on number of shortest paths
  - Say there were 5 shortest paths to D and only 3 to F
  - Then credit of edge (D,G) =  $5/8$  and credit of edge (F,G) =  $3/8$
- Node D gets credit = **1 + credits of edges below it** =  $1 + 3 + 0.5 = 4.5$
- Node F gets credit =  $1 + 0.5 = 1.5$
- D has **only one parent**, so Edge (E,D) gets credit = 4.5 from D
- Likewise for F: Edge (E,F) gets credit = 1.5 from F

- c. Con for girvan-newman is computationally expensive for large graphs.

### 2. Define Modularity Q (higher is better)

- a. A measure of how well a network is partitioned into communities
- b. Given a partition of the network into groups  $s$  in  $S$
- c.  $Q = \text{Sum}((\text{number edges in groups } s) - (\text{expected number edges in } s))$

### 3. Modularity (range: -1 to 1):

#### Modularity of partitioning $S$ of graph $G$ :

- $Q \propto \sum_{s \in S} [ (\# \text{ edges within group } s) - (\text{expected \# edges within group } s) ]$
- $Q(G, S) = \frac{1}{2m} \sum_{s \in S} \sum_{i \in s} \sum_{j \in s} \left( A_{ij} - \frac{k_i k_j}{2m} \right)$ 
  - Normalizing const.:  $-1 < Q < 1$
  - All possible pairs of nodes in  $s$ : both  $(i,j), (j,i)$ , but not  $(i,i), (j,j)$
  - The actual existence of the edge in  $s$
  - The expected existence of the edge in  $s$

#### Hints

- Determine  $m$  and  $k_i, k_j$  based on  $G$  (before cuttings)
- Determine  $A_{ij}$  based on  $s$  in  $S$  (after cuttings)
- If  $s = \{n_x\}$  is a singleton, then use  $2 * (0 - k_x * k_x / 2m) = -k_x * k_x / m$

a.

#### 4. Spectral clustering (graph cut)

- a. Partitioning a graph to minimize the number of edges that connect different communities.
- b. Goal: minimize the cut size but smallest group size is not always best
  - i. Divide nodes into two sets so that the cut (set of edges that connect nodes in different sets) is minimized
  - ii. Want the two sets to be approximately equal in size
  - iii. Maximize the number of within-group connections
  - iv. Minimize the number of between-group connections

#### c. Normalized cut;

$$ncut(A, B) = \frac{cut(A, B)}{vol(A)} + \frac{cut(A, B)}{vol(B)}$$

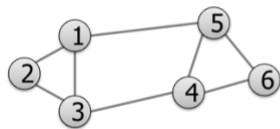
- i.
- ii.  $Vol(t)$  = edges (count even after edge cut)
- iii. Good cut = 0 Strongly connected groups with minimal between group edges
- iv. Bad cut = 1. Weak Separation many cross group connections

#### d. Matrix:

##### Adjacency matrix (A):

$n \times n$  matrix

$A = [a_{ij}]$ ,  $a_{ij} = 1$  if edge between node  $i$  and  $j$

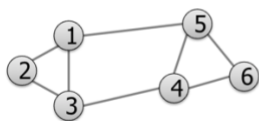


	1	2	3	4	5	6
1	0	1	1	0	1	0
2	1	0	1	0	0	0
3	1	1	0	1	0	0
4	0	0	1	0	1	1
5	1	0	0	1	0	1
6	0	0	0	1	1	0

##### Degree matrix (D):

$n \times n$  diagonal matrix


$D = [d_{ii}]$ ,  $d_{ii}$  = degree of node  $i$

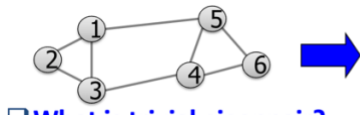


	1	2	3	4	5	6
1	3	0	0	0	0	0
2	0	2	0	0	0	0
3	0	0	3	0	0	0
4	0	0	0	3	0	0
5	0	0	0	0	3	0
6	0	0	0	0	0	2

#### e. Laplace matrix

- i. row sum = 0
- ii. Symmetric

 **Laplacian matrix (L):**  
 ➤  $n \times n$  symmetric matrix



	1	2	3	4	5	6
1	3	-1	-1	0	-1	0
2	-1	2	-1	0	0	0
3	-1	-1	3	-1	0	0
4	0	0	-1	3	-1	-1
5	-1	0	0	-1	3	-1
6	0	0	0	-1	-1	2

(degree matrix – adjacency matrix)

5. Bipartite graph:

- a. A complete bipartite graph: Contains all possible edges between a vertex of  $f$  and a vertex of  $c$

6. Direct discovery