Complex Networks
mathematics of networks

2015.10.19(Mon)

contents (1)

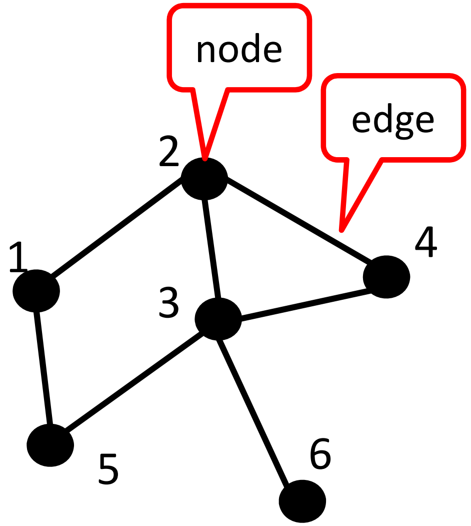
- networks and their representation
- adjacency matrix
- weighted networks
- directed networks
- hypergraphs
- bipartite networks
- trees

contents (2)

- planar networks
- degree
- paths
- components
- independent paths, connectivity, and cut sets
- the graph laplacian
- random walks

networks and their representation

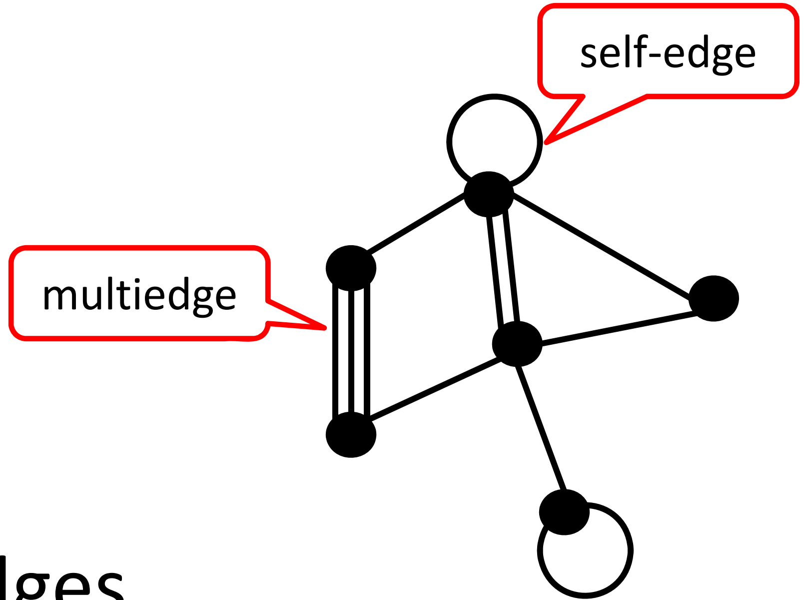
- a network (a graph) is a collection of vertices (nodes) joined by edges (links).



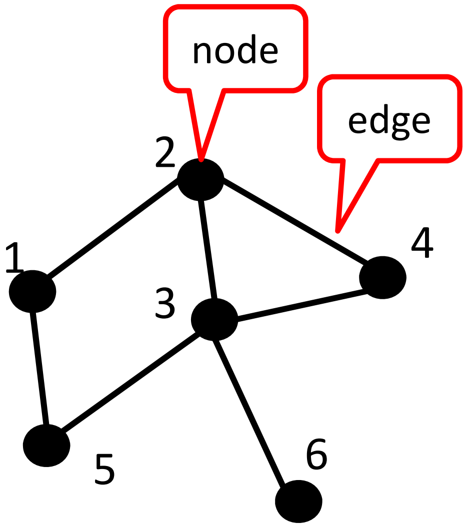
Network	Vertex	Edge
Internet	Computer or router	Cable or wireless data connection
World Wide Web	Web page	Hyperlink
Citation network	Article, patent, or legal case	Citation
Power grid	Generating station or substation	Transmission line
Friendship network	Person	Friendship
Metabolic network	Metabolite	Metabolic reaction
Neural network	Neuron	Synapse
Food web	Species	Predation

notations

- n: the number of vertices in a network
- m: the number of edges
- multiedge, self-edge
- multigraph: with multiedges



edge list & adjacency matrix



edge list

n=6

(1,2),(1,5),(2,3),(2,4),(3,4),(3,5),(3,6)

adjacency matrix

$$A_{ij} = \begin{cases} 1 & \text{if there is an edge between vertices i and j,} \\ 0 & \text{otherwise.} \end{cases}$$

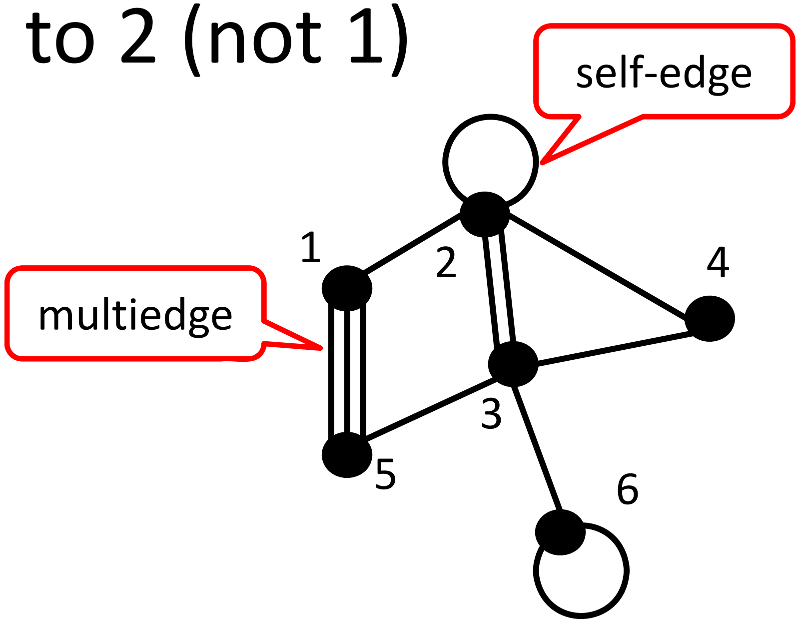
$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

adjacency matrix

- no self-edge -> diagonal elements are all zero
- symmetric (for undirected networks)

- multiedge: setting A_{ij} equal to the multiplicity
- self-edge: setting A_{ij} equal to 2 (not 1)

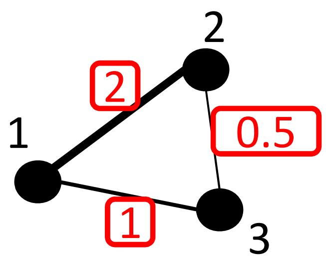
$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 3 & 0 \\ 1 & 2 & 2 & 1 & 0 & 0 \\ 0 & 2 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 3 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 2 \end{pmatrix}$$



weighted networks

- weights represent
  - the amount of data flowing/bandwidth (Internet)
  - total energy flow (food web)
  - frequency of contact (social network)

$$A = \begin{pmatrix} 0 & 2 & 1 \\ 2 & 0 & 0.5 \\ 1 & 0.5 & 0 \end{pmatrix}$$

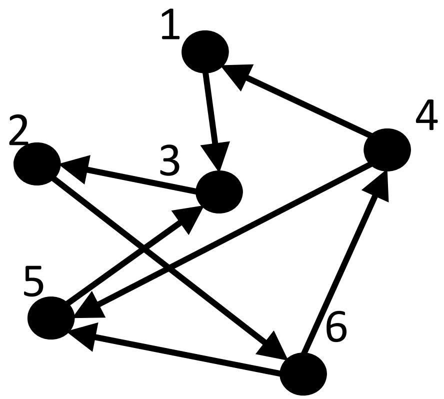


- weighted edge vs multiedge
  - switching between the two can be useful for analysis
- weights can be negative
  - animosity (social network)

directed network (digraph)

- each edge has a direction
  - hyperlink from one page to another (WWW)
- adjacency matrix is asymmetric

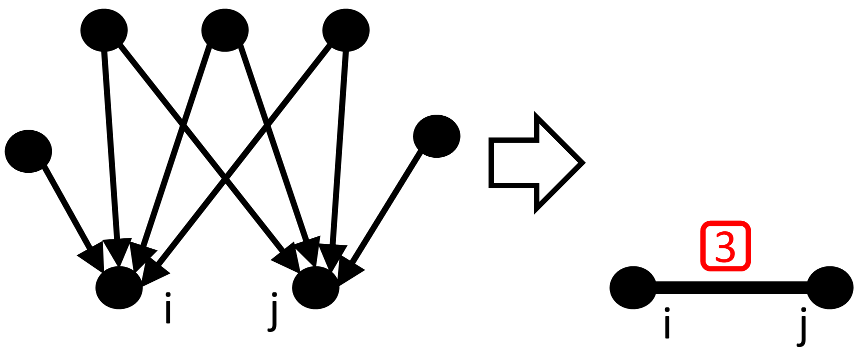
$$A_{ij} = \begin{cases} 1 & \text{if there is an edge from } j \text{ to } i, \\ 0 & \text{otherwise.} \end{cases}$$



$$A = \begin{array}{cc} & \begin{array}{cccccc} & & & j & & \end{array} \\ \begin{array}{c} \\ \\ i \\ \\ \\ \\ \end{array} & \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

cocitation and bibliograhic coupling

- a directed network -> an undirected one
  - just ignoring the edge directions is easy, but it may lose valuable information
  - cocitation: # of vertices that have outgoing edges pointing to both i and j

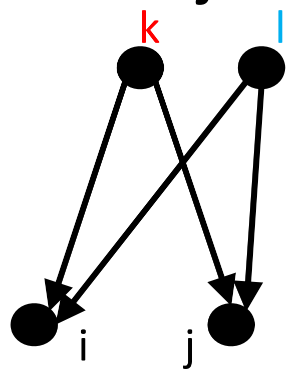


papers i & j are often co-cited
-> they are closely related

adjacency matrix of cocitation

- C (cocitation matrix)
  - Cij : # of columns whose ith & jth elements are 1

$$A = \begin{array}{cc} & \begin{array}{cccc} & k & l & \end{array} \\ \begin{array}{c} \\ i \\ j \\ \\ \end{array} & \begin{pmatrix} \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot \\ \cdot\cdot & 1 & \cdot\cdot & \cdot\cdot \\ \cdot\cdot & 1 & \cdot\cdot & \cdot\cdot \\ \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot \end{pmatrix} \end{array}$$



$$C_{ij} = \sum_{k=1}^{n} A_{ik} A_{jk} = \sum_{k=1}^{n} A_{ik} A_{kj}^{T} \qquad i \neq j$$

A^T : transpose of A

$$C = \begin{array}{cc} & \begin{array}{cccc} & k & l & \end{array} \\ \begin{array}{c} \\ i \\ j \\ \\ \end{array} & \underbrace{\begin{pmatrix} \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot \\ \cdot\cdot & 1 & \cdot\cdot & \cdot\cdot \\ \cdot\cdot & 1 & \cdot\cdot & \cdot\cdot \\ \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot \end{pmatrix}}_{A} \end{array} \begin{array}{cc} \begin{array}{cccc} & i & j & \end{array} & \\ \underbrace{\begin{pmatrix} \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot \\ \cdot\cdot & 1 & 1 & \cdot\cdot \\ \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot \\ \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot \end{pmatrix}}_{A^T} & \begin{array}{c} \\ k \\ l \\ \\ \end{array} \end{array}$$

$$C = AA^T$$

C is symmetric because

$$C^T = (AA^T)^T = AA^T = C$$

more on citation matrix

- if all elements in A are zero or one,

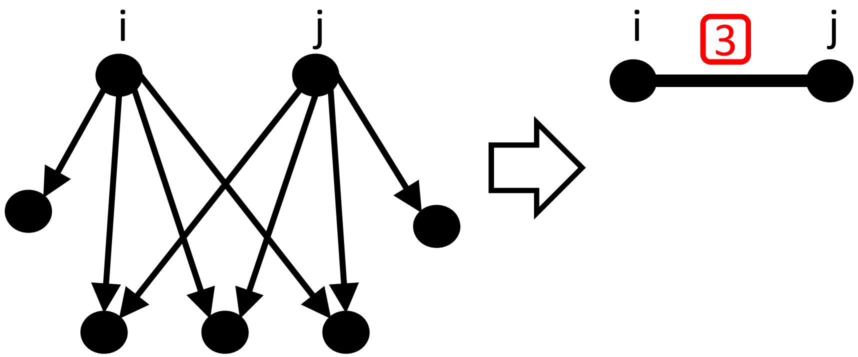
$$C_{ii} = \sum_{k=1}^{n} A_{ik}^2 = \sum_{k=1}^{n} A_{ik}$$ -> # of 1s in ith row

- we ignore these diagonal elements

$$C_{ij} = \begin{cases} \sum_{k=1}^{n} A_{ik} A_{kj}^T & i \neq j \\ 0 & i = j \end{cases}$$

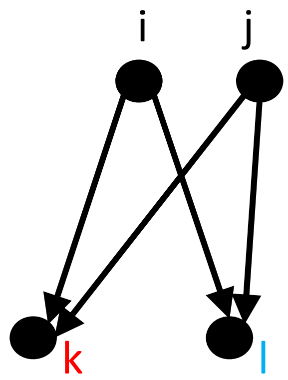
bibliographic coupling

- # of other vertices to which both point



i & j often cite the same papers
-> they are closely related

$$A = \begin{array}{c|cccc} & & i & j & \\ \hline & \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot \\ k & \cdot\cdot & 1 & 1 & \cdot\cdot \\ l & \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot \\ & \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot \end{array}$$



- B (bibliographic coupling)

$$B_{ij} = \sum_{k=1}^{n} A_{ki} A_{kj} = \sum_{k=1}^{n} A_{ik}^{T} A_{kj} \quad i \neq j$$

$$B = \underbrace{\begin{array}{c|cccc} & & k & l & \\ \hline & \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot \\ i & \cdot\cdot & 1 & \cdot\cdot & \cdot\cdot \\ j & \cdot\cdot & 1 & \cdot\cdot & \cdot\cdot \\ & \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot \end{array}}_{A^T} \underbrace{\begin{array}{cccc|c} & i & j & & \\ \hline \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot & \\ \cdot\cdot & 1 & 1 & \cdot\cdot & k \\ \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot & l \\ \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot & \end{array}}_{A}$$

$$B = A^T A$$

B is symmetric

$$B_{ij} = \begin{cases} \sum_{k=1}^{n} A_{ik}^{T} A_{kj} & i \neq j \\ 0 & i = j \end{cases}$$

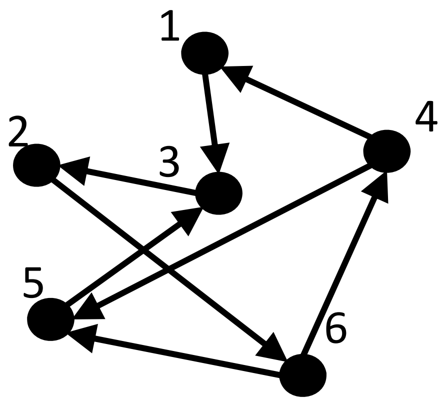
cocitation & bibliographic coupling

- mathematically similar, but practically different
- cocitation
  - is limited to influential papers
  - may change over time as the papers receive new citations
- bibliographic coupling
  - is more uniform indicator of similarity than cocitation
    - because the size of bibliography vary less than # of citations paper receive
  - can be computed as soon as a paper is published

Example with R

```
> a <- rbind(c(0,0,0,1,0,0),
+                c(0,0,1,0,0,0),
+                c(1,0,0,0,1,0),
+                c(0,0,0,0,0,1),
+                c(0,0,0,1,0,1),
+                c(0,1,0,0,0,0))
> a
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    0    0    0    1    0    0
[2,]    0    0    1    0    0    0
[3,]    1    0    0    0    1    0
[4,]    0    0    0    0    0    1
[5,]    0    0    0    1    0    1
[6,]    0    1    0    0    0    0
```

definition of matrix A




```
> c <- a %*% t(a)
> diag(c) <- 0
> c
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    0    0    0    0    1    0
[2,]    0    0    0    0    0    0
[3,]    0    0    0    0    0    0
[4,]    0    0    0    0    1    0
[5,]    1    0    0    1    0    0
[6,]    0    0    0    0    0    0
> b <- t(a) %*% a
> diag(b) <- 0
> b
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    0    0    0    0    1    0
[2,]    0    0    0    0    0    0
[3,]    0    0    0    0    0    0
[4,]    0    0    0    0    0    1
[5,]    1    0    0    0    0    0
[6,]    0    0    0    1    0    0
>
```

$$\boxed{C = AA^T}$$

diagonal elements=0

1&5 are cocited by 4

4&5 are cocited by 6

$$\boxed{B = A^T A}$$

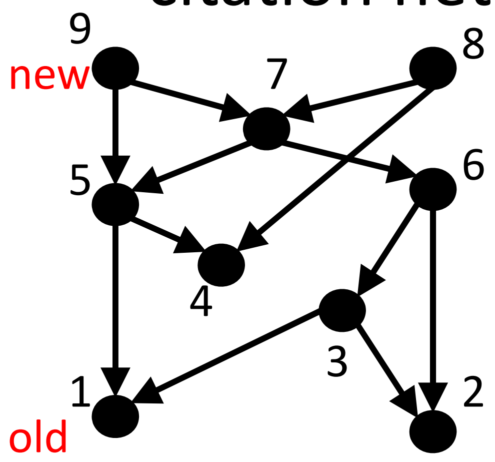
diagonal elements=0

1&5 cocite 3

4&6 cocite 5

acyclic directed networks

- cycle : a closed loop (including self-edge)
- acyclic network (DAG) : without loop
- acyclic directed network
  - citation network : vertices are time-ordered



no upward edges -> no loop

“acyclic -> no upward edges”

<proof>

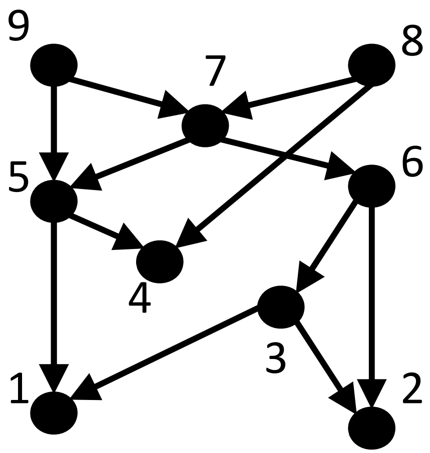
- an acyclic network of n vertices
- there must be at least one vertex that has no outgoing edges
  - a path across the network by following edges (at most n-1 times) will encounter a vertex with no outgoing edges
- then put the vertex at the bottom of the picture and remove the vertex and attached edges
- repeat the above process

cyclic or acyclic?

1. Find a vertex with no outgoing edges
2. If no such vertex exists, the network is <u>cyclic</u>. Otherwise, if such a vertex does exist, remove it and all its ingoing edges from the network.
3. If all vertices have been removed, the network is <u>acyclic</u>. Otherwise, go back to step 1

adjacency matrix of DAG is triangular

- vertices are numbered in the order they are removed in the previous algorithm
  - an edge from j to i only if j > i
  - no self-edge -> diagonal elements are 0



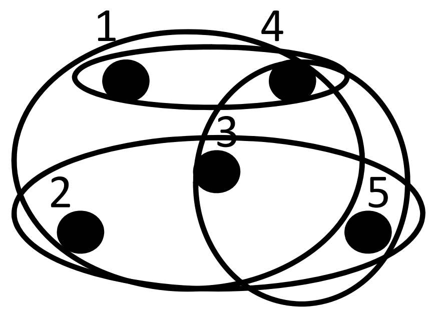
$$A = \begin{array}{cc} & \begin{array}{ccccccccc} & & & & j & & & & \end{array} \\ \begin{array}{c} \\ \\ \\ i \\ \\ \\ \\ \\ \\ \end{array} & \begin{pmatrix} 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

acyclic <-> eigenvalues are zero

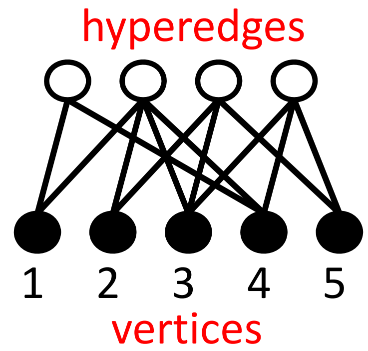
- ->
  - acyclic -> order the vertices described previously
  - adjacency matrix is strictly upper triangular
  - eigenvalues (diagonal elements) are all zero
- <-
  - prove contraposition
    - “cyclic -> at least one nonzero eivenvalue”
  - the total number Lr of cycles of length r is $L_r = \sum_{i=1}^{n} \kappa_i^r$
    - κ_i : ith eigenvalue
  - cyclic -> Lr > 0 -> at least one κ_i is greater than zero

hypergraphs

- links sometimes join more than two vertices
  - families
  - actors in a film



{1,4}
{1,2,3,4}
{2,3,5}
{3,4,5}



Network	Vertex	Group	Section
Film actors	Actor	Cast of a film	3.5
Coauthorship	Author	Authors of an article	3.5
Boards of directors	Director	Board of a company	3.5
Social events	People	Participants at social event	3.1
Recommender system	People	Those who like a book, film, etc.	4.3.2
Keyword index	Keywords	Pages where words appear	4.3.3
Rail connections	Stations	Train routes	2.4
Metabolic reactions	Metabolites	Participants in a reaction	5.1.1

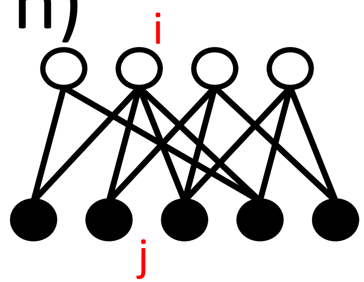
bipartite networks

- two kinds of vertices
  - original vertices and the groups to which they belong
- edges run only between vertices of unlike types
- incidence matrix **B** (g x n)

# of groups

# of vertices

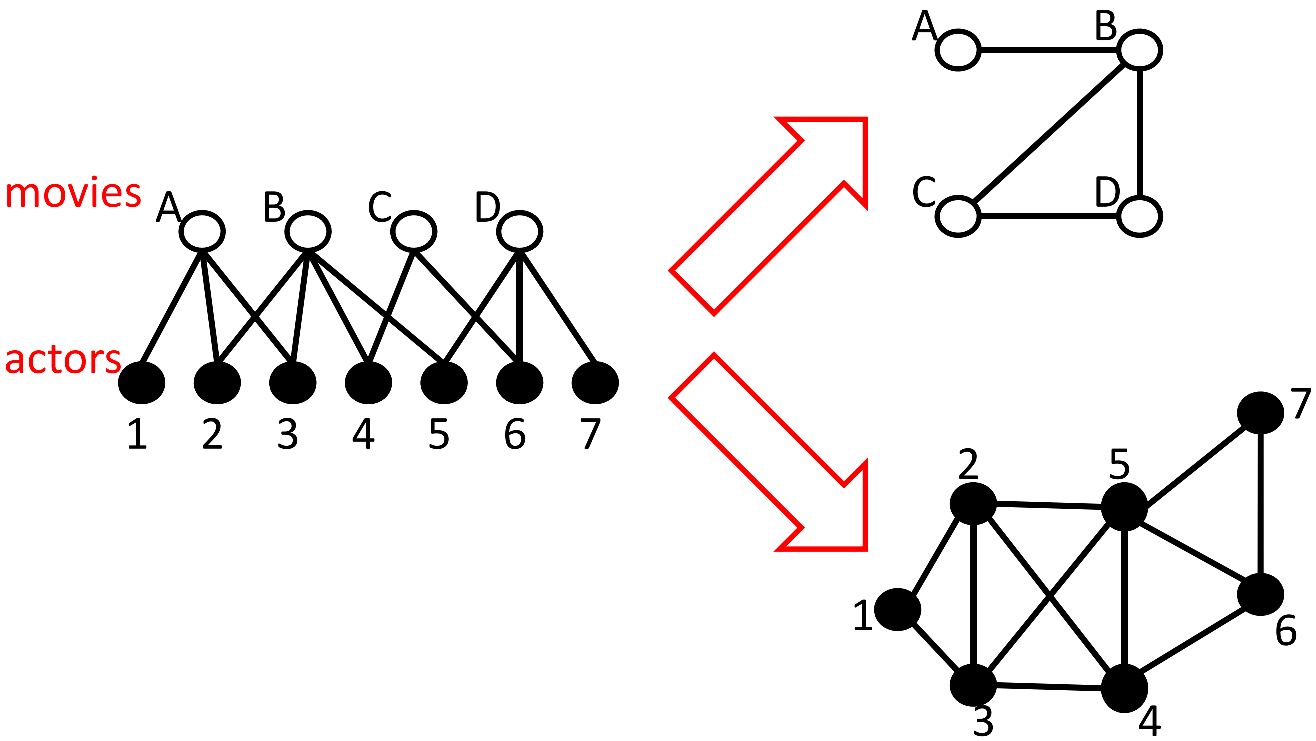
$$B_{ij} = \begin{cases} 1 & \text{if vertex j belongs to group i,} \\ 0 & \text{otherwise.} \end{cases}$$



$$\begin{array}{cc} & \begin{array}{ccccc} & \text{vertices} & & & \\ & j & & & \end{array} \\ \begin{array}{r} \text{groups} \\ i \\ B = \\ \\ \end{array} & \begin{pmatrix} 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 \end{pmatrix} \end{array}$$

one-mode projection

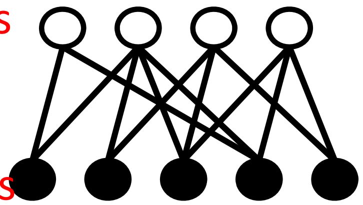
- bipartite -> unipartite
- discards a lot of the information



weighted projection

- projection onto (original) vertices
  - $B_{ki}B_{kj} = 1$ <-> i and j both belong to group k

$$P_{ij} = \sum_{k=1}^{g} B_{ki}B_{kj} = \sum_{k=1}^{g} B_{ik}^{T}B_{kj}$$



n vertices

$$\begin{array}{cc} & \begin{array}{ccccc} i & j & & & \end{array} \\ \text{g groups} \quad B = & \begin{pmatrix} 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 \end{pmatrix} \end{array}$$

k

$$\mathbf{P} = \mathbf{B}^{T}\mathbf{B}$$ n x n matrix

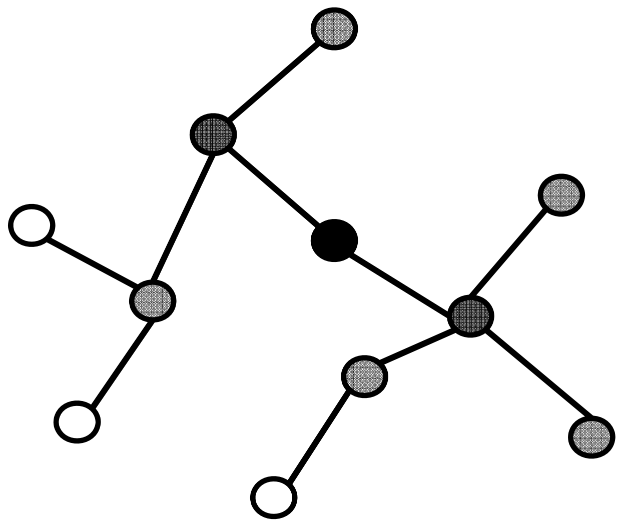
  - diagonal elements $P_{ii} = \sum_{k=1}^{g} B_{ki}^{2} = \sum_{k=1}^{g} B_{ki}$
    - # of groups to which vertex i belong

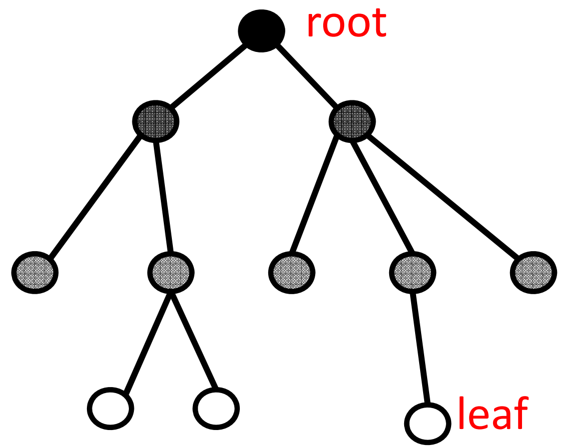
- projection onto groups

$$\mathbf{P}' = \mathbf{B}\mathbf{B}^{T}$$ g x g matrix

trees

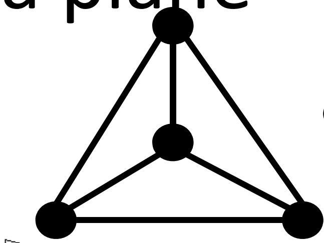
- connected, undirected network without any closed loop
- forest : collection of trees
- exactly one path between any pair of vertices
- (# of vertices) = (# of edges) + 1

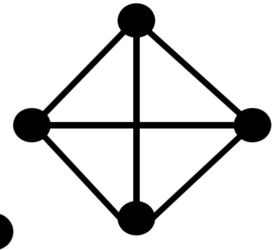


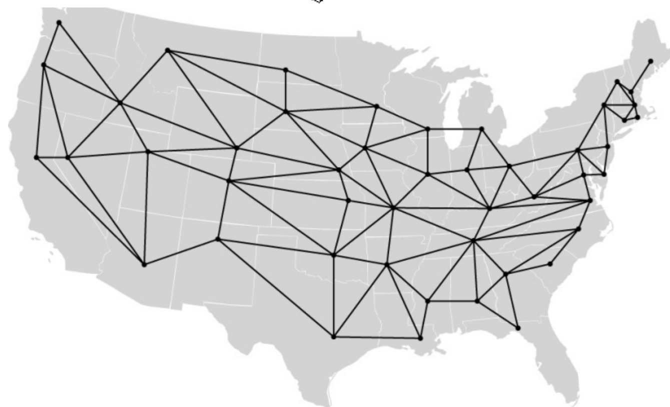


planar network

- a network that can be drawn on a plane without having any edges cross
- trees are planar
- examples
  - road network (without bridges)
  - shared borders between countries
    - four-color theorem



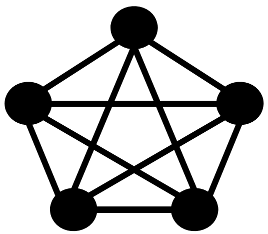




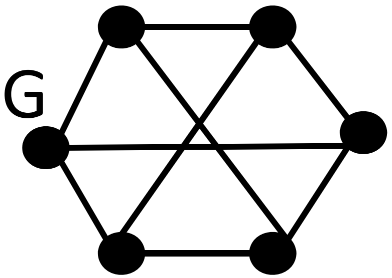
planar or not?

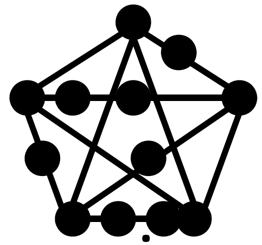
- Any network that contains a subset of vertices in the form of K_5 or UG is not planar.

K_5



UG



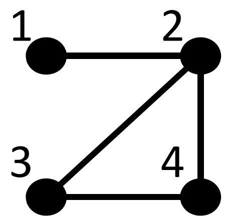


expansion of K_5

- Any expansion of K_5 or UG is not planar.
- Kuratowski's theorem
  - Every non-planar network contains at least one subgraph that is an expansion of K_5 or UG.

degree

- k_i : the degree of vertex i $\quad k_i = \sum_{j=1}^{n} A_{ij}$
  - – # of edges connected to it



$$\begin{array}{cc} & A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix} \end{array}$$

k_2

m

- (sum of all degrees) = 2 x (# of edges)

$$\sum_{i=1}^{n} k_i = \sum_{i=1}^{n}\sum_{j=1}^{n} A_{ij} = 2m$$

- c : mean degree

$$c = \frac{1}{n}\sum_{i=1}^{n} k_i = \frac{2m}{n}$$

- maximum possible number of edges

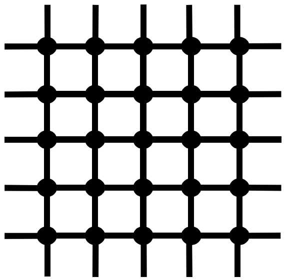
$$_nC_2 = \binom{n}{2} = \frac{1}{2}n(n-1)$$

density

- density (or connectance) $\rho = \frac{m}{\binom{n}{2}} = \frac{2m}{n(n-1)} = \frac{c}{n-1} \approx \frac{c}{n}$
  $0 \le \rho \le 1$

  networks is sufficiently large

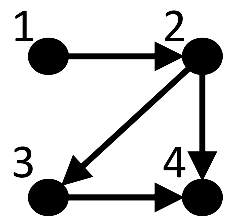
- dense : $\rho \to const$ as $n \to \infty$
- sparse : $\rho \to 0$ as $n \to \infty$
- almost all of the networks we consider are sparse (except food webs)
  - important for developing algorithms and models
- k-regular : all vertices have degree k



degrees in directed networks

- in-degree $k_i^{in} = \sum_{j=1}^{n} A_{ij}$
- out-degree $k_i^{out} = \sum_{i=1}^{n} A_{ij}$

$$m = \sum_{i=1}^{n} k_i^{in} = \sum_{j=1}^{n} k_j^{out} = \sum_{ij} A_{ij}$$



$$\begin{array}{cc} & \begin{array}{cccc} & k_2^{out} & & \end{array} \\ \begin{array}{c} \\ k_2^{in} \\ \\ \\ \end{array} & A = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{pmatrix} \end{array}$$

- mean in-degree c_{in}
- mean out-degree c_{out}

$$c_{in} = \frac{1}{n}\sum_{i=1}^{n} k_i^{in} = \frac{1}{n}\sum_{j=1}^{n} k_j^{out} = c_{out}$$

-> we will just denote both by c $\quad c = \frac{m}{n}$

path

- a route across the network that runs from vertex to vertex along the edges of the network
- self-avoiding path : a path that does not intersect itself
- length : # of edges traversed along the path
- # of paths of a given length r
  - $A_{ik}A_{kj} = 1$ if there is a path j -> k -> i
  - # of paths of length 2 from j to i : $N_{ij}^{(2)} = \sum_{k=1}^{n} A_{ik}A_{kj} = \left[\mathbf{A}^2\right]_{ij}$
  - # of paths of length r from j to i : $N_{ij}^{(r)} = \left[\mathbf{A}^r\right]_{ij}$

cycles

- paths of length r that start and end at the same vertex $L_r = \sum_{i=1}^{n} \left[\mathbf{A}^r\right]_{ii} = Tr\mathbf{A}^r$ 1->2->3->1 and 2->3->1->2 are distinct
  - counting each loop only once is not easy
- L_r in terms of eigenvalues of **A** (undirected)
- $\mathbf{A} = \mathbf{UKU}^T$

diagonal matrix of eigenvalues

orthogonal matrix of eigenvectors

undirected graph -> **A** is symmetric
-> **A** is diagonalizable

$$\mathbf{A}^r = \left(\mathbf{UKU}^T\right)^r = \mathbf{UK}^r\mathbf{U}^T \qquad \because \mathbf{UU}^T = \mathbf{U}^T\mathbf{U} = \mathbf{I}$$

$$L_r = Tr(\mathbf{UK}^r\mathbf{U}^T) = Tr(\mathbf{U}^T\mathbf{UK}^r) = Tr\mathbf{K}^r = \sum_i \kappa_i^r$$

$$\because Tr(\mathbf{AB}) = Tr(\mathbf{BA})$$

κ_i : ith eigenvalue of **A**

cycles of directed graphs

- $L_r = \sum_i \kappa_i^r$ is true also for directed graphs
  - although A cannot be diagonalized
- proof
  - Every real matrix can be written in the form
    - Schur decomposition

upper triangular matrix

$$\mathbf{A} = \mathbf{QTQ}^T$$

orthogonal matrix

  - Eigenvalues of T are the same as those of A

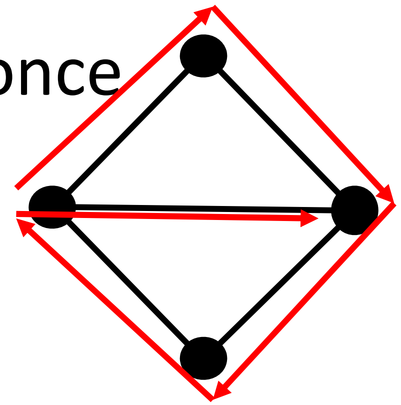
$$L_r = Tr\mathbf{A}^r = Tr(\mathbf{QT}^r\mathbf{Q}^T) = Tr(\mathbf{Q}^T\mathbf{QT}^r) = Tr\mathbf{T}^r = \sum_i \kappa_i^r$$

geodesic path (shortest path)

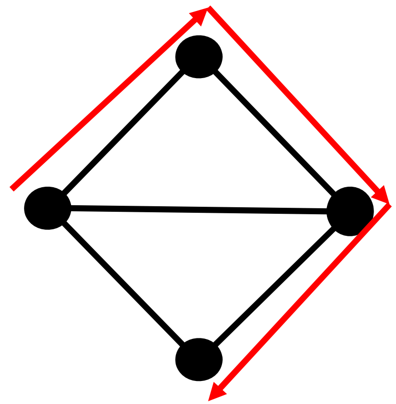
- geodesic distance between vertices i and j
  - smallest value of r such that $\left[\mathbf{A}^r\right]_{ij} > 0$
- self-avoiding: no loop
- diameter : the longest geodesic path between any pair of vertices in the network

Eulerian and Hamiltonian path

- Eulerian path
  - a path that traverses each edge exactly once

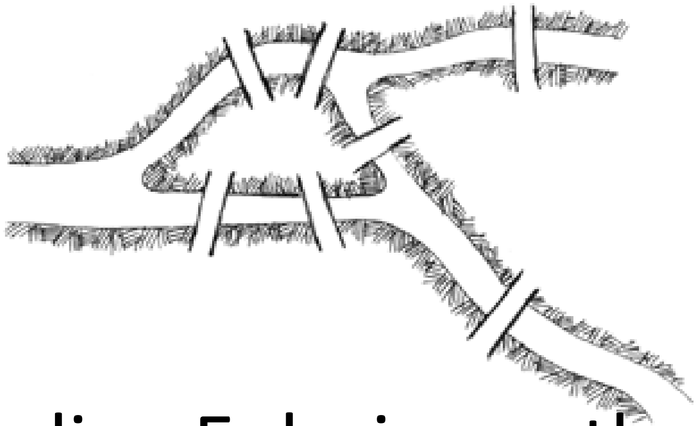


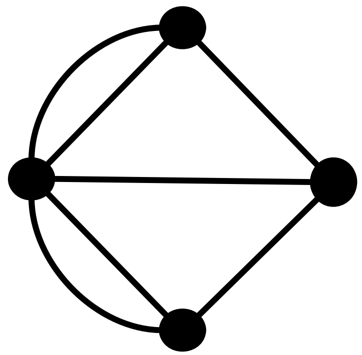
- Hamiltonian path
  - a path that visit each vertex exactly once
  - self-avoiding



Königsberg bridge problem

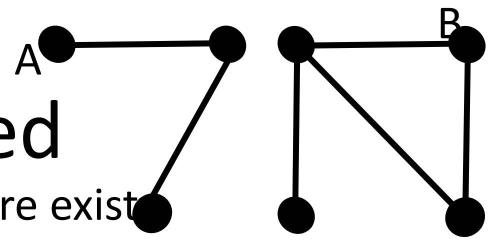
- Does there exist any walking route that crosses all seven bridges exactly once each?





- -> finding Eulerian path on the right network
  - at most two vertices with odd degree
  - all four vertices have odd degree -> no solution

components

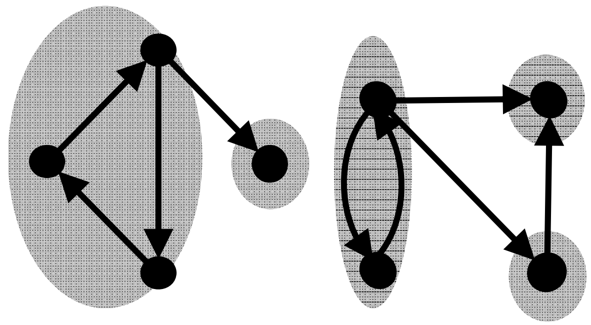


- no path from A to B -> disconnected
- component : subgroups in a network such that there exist at least one path from each member to each other member
- block diagonal matrix

$$\mathbf{A} = \begin{pmatrix} \square & 0 & \cdots \\ 0 & \square & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}$$

non-zero elements

- components in directed networks
  - two (undirected network)
  - five (directed network)



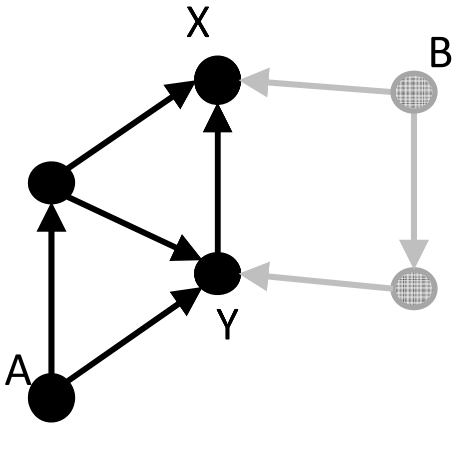
strongly connected components (SCC)

- A and B are connected if and only if there exists both A->B and B->A
- SCC is a maximal subset of vertices such that there is a directed path in both directions between every pair in the subset
- each vertex belongs to exactly one SCC
- every SCC with more than one vertex must contain at least one cycle

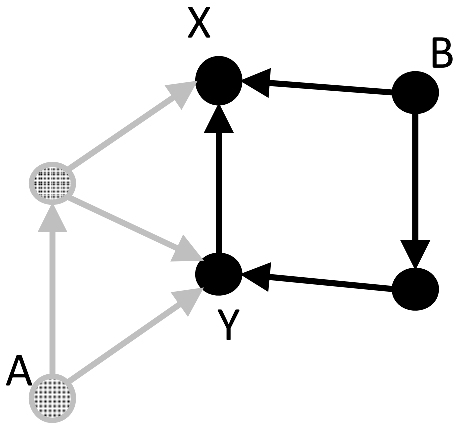
out-component in a directed network

- the set of vertices that are reachable via directed paths starting at a specific vertex A
- depends on network and starting vertex

out-component of vertex A

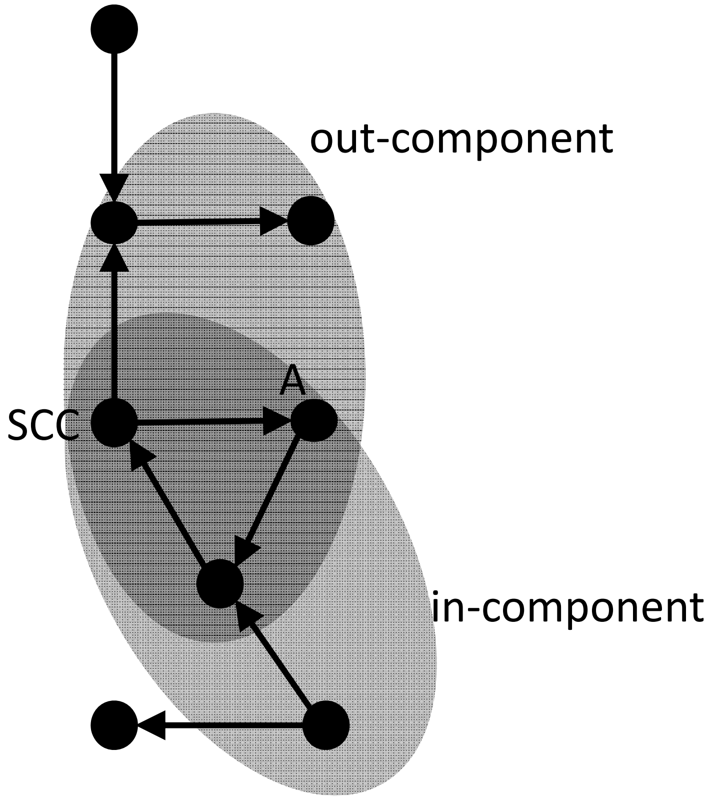


out-component of vertex B

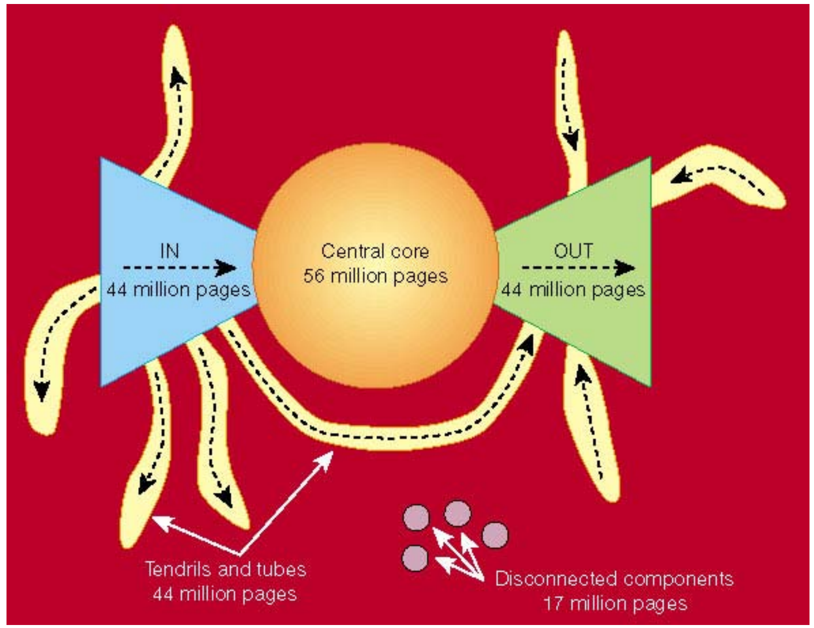


in-component & out-component

- in-component :reachable to vertex A
- out-component : reachable from vertex A
- SCC: intersection of in and out

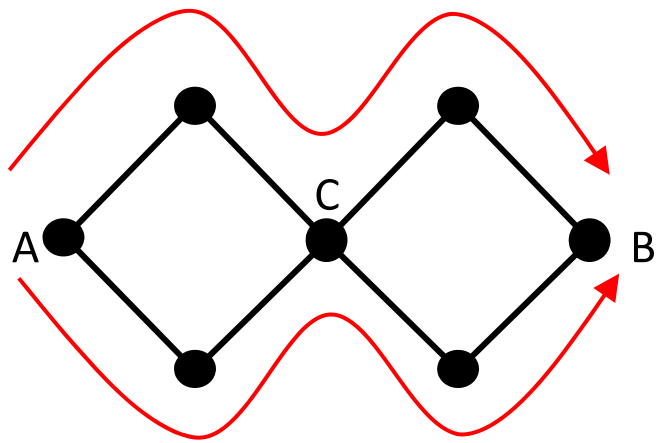


The Web is a bow tie



independent paths

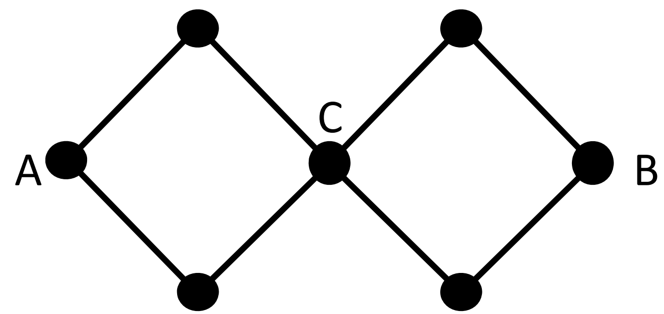
- edge-independent path share no edges
- vertex-independent path share no vertices (except starting and ending vertices)
- vertex-independent -> edge-independent
  - but the reverse is not true



2 edge-independent paths
1 vertex-independent path

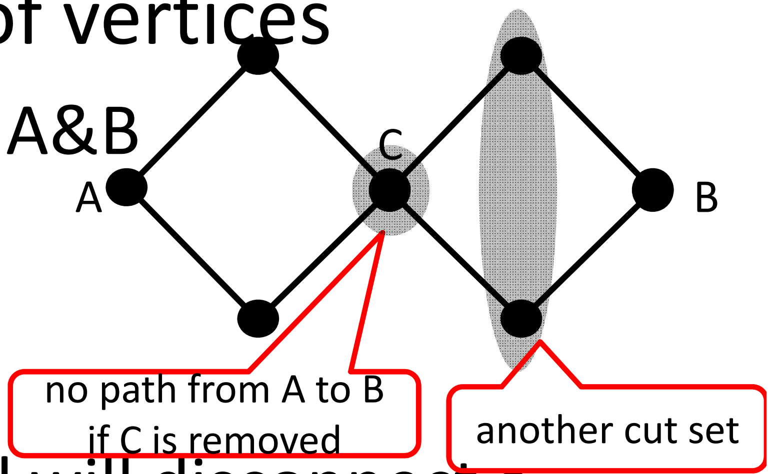
more on independent paths

- There can be only a finite number of independent paths between any two vertices in a finite network
- connectivity : # of independent paths between a pair of vertices
  - A and B have edge connectivity 2 but vertex connectivity 1
  - strength of connection
    - discovering communities
    - finding bottlenecks



cut set

- a set of vertices whose removal will disconnect a specified pair of vertices
  – C forms a cut set of size 1 for A&B

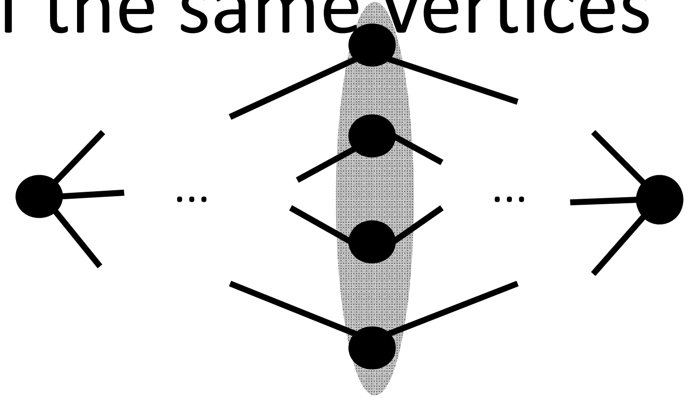


- edge cut set
  – a set of edges whose removal will disconnect a specified pair of vertices
- minimum cut set : the smallest cut set

Menger’s theorem

- If there is no cut set of size less than n between a given pair of vertices, then there are at least n independent paths between the same vertices
  – this theorem applies both to edges and to vertices
- The size of the minimum vertex cut set that disconnects a given pair of vertices is equal to the vertex connectivity of the same vertices

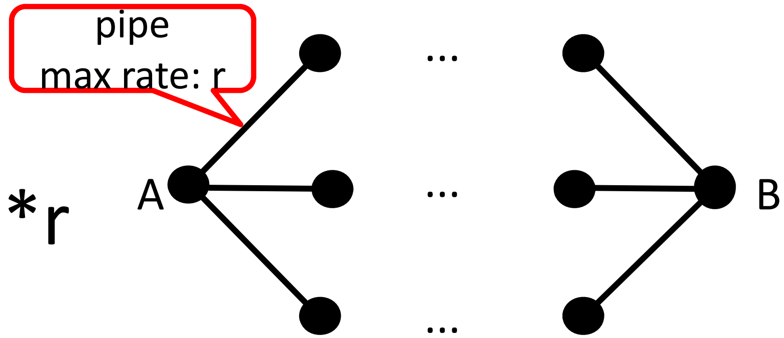
min cut set		independent paths
n	->	n or more
n or more	<-	n



maximum flow

- a network of water pipes
- the max rate from A to B =

(# of edge-independent paths)*r



- proof
  - n independent paths -> at least n*r of flows (lower bound)
  - a cut set of n edges -> at most n*r of flows (upper bound)
  - the max rate is exactly n*r
- max-flow/min-cut theorem
  - individual pipes can have different capacities

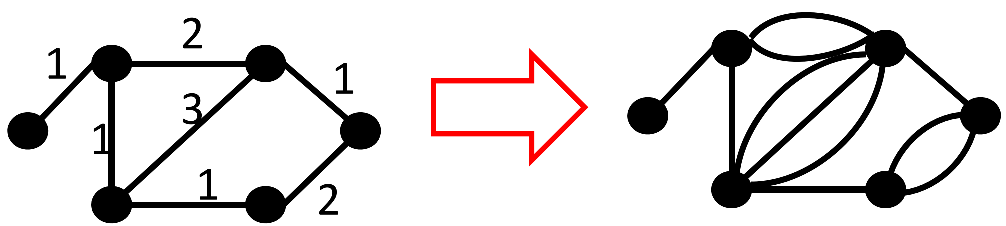
these three are numerically equal

- the edge connectivity of a pair of vertices
  - the number of edge-independent paths
- the size of the minimum edge cut set
  - the number of edges that must be removed to disconnect them
- the maximum flow between the vertices

these are equal for directed network as well

max-flows on weighted networks

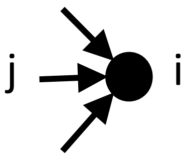
- max-flows/min-cut theorem can be extended to weighted networks
  - the maximum flow between a given pair of vertices in a network is equal to the sum of the weights on the edges of the minimum edge cut set that separate the same two vertices
- proof
  - transform weighted edges to multiedges



diffusion process on networks

- spreading (ideas/diseases/...) on networks
- ψ_i:some commodity or substance at vertex i
- $C(\psi_i-\psi_j)$: flow from i to j (C:constant)

$$\frac{d\psi_i}{dt} = C\sum_j A_{ij}(\psi_j - \psi_i)$$



degree of i

$$\frac{d\psi_i}{dt} = C\sum_j A_{ij}\psi_j - C\psi_i\sum_j A_{ij} = C\sum_j A_{ij}\psi_j - C\psi_i k_i$$

$$= C\sum_j (A_{ij} - \delta_{ij}k_i)\psi_j$$

1 if i=j
0 otherwise

$$\frac{d\psi}{dt} = C(A-D)\psi$$

$$D = \begin{pmatrix} k_1 & 0 & 0 & \cdots \\ 0 & k_2 & 0 & \cdots \\ 0 & 0 & k_3 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix}$$

graph Laplacian (1)

$$\frac{d\psi}{dt} = C(A - D)\psi$$

graph Laplacian

$$L = D - A$$

$$\Rightarrow \frac{d\psi}{dt} + CL\psi = 0$$

similar to diffusion equation

- graph Laplacian is for
  - random walk
  - resistor networks
  - graph partitioning
  - network connectivity

graph Laplacian (2)

$$L = D - A$$

$$L_{ij} = \begin{cases} k_i & \text{if i = j,} \\ -1 & \text{if i} \neq \text{j and there is an edge (i,j),} \\ 0 & \text{otherwise} \end{cases}$$

degree

edge

$$D = \begin{pmatrix} k_1 & 0 & -1 & \cdots \\ 0 & k_2 & 0 & \cdots \\ -1 & 0 & k_3 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix}$$

$$L_{ij} = \delta_{ij} k_i - A_{ij}$$

- ψ as linear combination of eigenvectors of L

$$\psi(t) = \sum_i a_i(t) v_i$$

v_i:eigenvectors of L $\quad Lv_i = \lambda_i v_i$

eigenvectors of a symmetric matrix are orthogonal

$$\frac{d\psi}{dt} + CL\psi = 0 \Rightarrow \sum_i \left(\frac{da_i}{dt} + C\lambda_i a_i \right) v_i = 0$$

$$\frac{da_i}{dt} + C\lambda_i a_i = 0 \qquad a_i(t) = a_i(0) e^{-C\lambda_i t}$$

eigenvalues of graph Laplacian (1)

- Laplacian is symmetric, so it has real eigenvalues. They are also non-negative.
- G=(V,E), |V|=n, |E|=m
- edge incidence matrix

Each row of B has one +1 and one -1

$$B_{ij} = \begin{cases} +1 & \text{if end 1 of edge i is attached to vertex j,} \\ -1 & \text{if end 2 of edge i is attached to vertex j,} \\ 0 & \text{otherwise} \end{cases}$$

$i \neq j$

$$\sum_k B_{ki} B_{kj} = -1$$

$\because$ The only non-zero terms will occur when an edge connects i and j.

$i = j$

$$\sum_k B_{ki}^2 = k_i$$

$$\therefore \sum_k B_{ki} B_{kj} = L_{ij} \Rightarrow \mathbf{L} = \mathbf{B}^{\mathbf{T}}\mathbf{B}$$

eigenvalues of the graph Laplacian (2)

- v_i: eigenvector of L with eigenvalue λ_i $\quad Lv_i = \lambda v_i$

$$\mathbf{L} = \mathbf{B}^{\mathbf{T}}\mathbf{B}$$

$$\mathbf{v}_{\mathbf{i}}^{\mathbf{T}}\mathbf{B}^{\mathbf{T}}\mathbf{B}\mathbf{v}_{\mathbf{i}} = \mathbf{v}_{\mathbf{i}}^{\mathbf{T}}\mathbf{L}\mathbf{v}_{\mathbf{i}} = \lambda_i \mathbf{v}_{\mathbf{i}}^{\mathbf{T}}\mathbf{v}_{\mathbf{i}} = \lambda_i$$

$$\lambda_i = (\mathbf{v}_{\mathbf{i}}^{\mathbf{T}}\mathbf{B}^{\mathbf{T}})(\mathbf{B}\mathbf{v}_{\mathbf{i}})$$

inner product of a real vector Bv_i and itself

$$\therefore \lambda_i \geq 0$$

- Laplacian always has at least one zero eigenvalue.

$$\mathbf{1} = (1,1,1,\ldots)$$

$$\sum_j L_{ij} \times 1 = \sum_j \left(\delta_{ij} - A_{ij}\right) = k_i - \sum_j A_{ij} = k_i - k_i = 0$$

$$\therefore \mathbf{L} \cdot \mathbf{1} = 0$$

1 is always an eigenvector of the graph Laplacian with eigenvalue zero

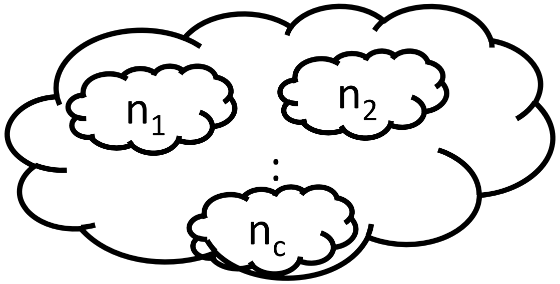
$$0 = \lambda_1 \leq \lambda_2 \leq \lambda_3 \leq \ldots \leq \lambda_n$$

Laplacian has no inverse because its determinant is always zero. It is singular.

components and connectivity

suppose we have a network that is divided into c components

- Laplacian -> block diagonal



Laplacian of each component

$$L = \begin{pmatrix} \square & 0 & \cdots \\ 0 & \square & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}$$

row sum = 0

$$D = \begin{pmatrix} k_1 & 0 & -1 & \cdots \\ 0 & k_2 & 0 & \cdots \\ -1 & 0 & k_3 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix}$$

$$v = \begin{pmatrix} 1 \\ \vdots \\ 0 \\ \vdots \end{pmatrix} \begin{matrix} \left.\vphantom{\begin{matrix}1\\ \vdots\end{matrix}}\right\} n_1 \text{ ones} \\ \left.\vphantom{\begin{matrix}0\\ \vdots\end{matrix}}\right\} \text{zeros} \end{matrix}$$

is an eigenvector of L with eigenvalue zero

$$Lv = 0v$$

⇨ at least c eigenvectors with eigenvalue zero

- (# of zero eigenvalues) = (# of components)

->the second eigenvalue of graph Laplacian λ_2 is non-zero if and only if the network is connected

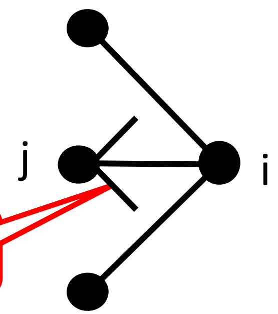
random walk

- a path across a network created by taking repeated random steps
  - used for sampling and ranking
- $p_i(t)$:probability that the walk is at vertex i at time t

$$p_i(t) = \sum_j \frac{A_{ij}}{k_j} p_j(t-1)$$

vector with element p_i

$$\mathbf{p}(t) = \mathbf{A}\mathbf{D}^{-1}\mathbf{p}(t-1)$$



degree:k_i

$$\mathbf{D}^{-1} = \begin{pmatrix} 1/k_1 & 0 & 0 & \cdots \\ 0 & 1/k_2 & 0 & \cdots \\ 0 & 0 & 1/k_3 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix} \qquad \mathbf{D}^{1/2} = \begin{pmatrix} \sqrt{k_1} & 0 & 0 & \cdots \\ 0 & \sqrt{k_2} & 0 & \cdots \\ 0 & 0 & \sqrt{k_3} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix}$$

reduced adjacency matrix

$$\mathbf{p}(t) = \mathbf{A}\mathbf{D}^{-1}\mathbf{p}(t-1)$$

$$\mathbf{D}^{-1/2}\mathbf{p}(t) = [\mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}][\mathbf{D}^{-1/2}\mathbf{p}(t-1)]$$

Repeated multiplication of this symmetric matrix

symmetric

$$\mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2} = \begin{cases} 1/\sqrt{k_i k_j} & \mathbf{A}_{ij} = 1 \\ 0 & otherwise \end{cases}$$

This matrix is called reduced adjacency matrix

- when $t \to \infty$

$$\mathbf{p} = \mathbf{A}\mathbf{D}^{-1}\mathbf{p}$$

$$(\mathbf{I} - \mathbf{A}\mathbf{D}^{-1})\mathbf{p} = (\mathbf{D} - \mathbf{A})\mathbf{D}^{-1}\mathbf{p} = \mathbf{L}\mathbf{D}^{-1}\mathbf{p} = 0$$

->D^{-1}p is an eigenvector of the Laplacian with eigenvalue 0

- connected network ->only one eigenvector (with eigenvalue 0) whose components are all equal

$$\mathbf{D}^{-1}\mathbf{p} = a\mathbf{1} \qquad p_i = ak_i$$

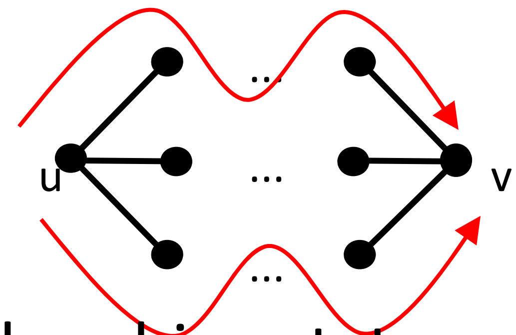
$$\mathbf{p} = a\mathbf{D}\mathbf{1} \qquad p_i = \frac{k_i}{\sum_j k_j} = \frac{k_i}{2m}$$

normalize

-> probability is proportional to the degree of the vertex

random walk with absorbing state(1)

- first passage time:# of steps from u first reaches v
- $p_v(t)$: probability that a walk is at v at time t
- $p_v(t) - p_v(t-1)$: prob. that a walk has first passage time exactly t



v:absorbing state
(never go out from v)

- mean first passage time

$$\tau = \sum_{t=0}^{\infty} t[p_v(t) - p_v(t-1)]$$

- trick for calculating $p_v(t)$ is in the next slides

random walk with absorbing state(2)

- $A_{iv} = 0$ ∵ v is absorbing state

  $A_{vi} \neq 0$ A is asymmetric

A_{iv} =0 and the terms with j=v don't contribute to the sum

$$p_i(t) = \sum_j \frac{A_{ij}}{k_j} p_j(t-1) = \sum_{j(\neq v)} \frac{A_{ij}}{k_j} p_j(t-1)$$

$$\mathbf{p}'(t) = \mathbf{A}'\mathbf{D}'^{-1}\mathbf{p}'(t-1)$$

$$\mathbf{M} = \mathbf{A}'\mathbf{D}'^{-1}$$

p'(t):p with vth element removed

A',D': A and D with vth row and column removed

these are symmetric

$$\mathbf{p}'(t) = [\mathbf{A}'\mathbf{D}'^{-1}]^t \mathbf{p}'(0)$$

$$p_v(t) = 1 - \sum_{i(\neq v)} p_i(t) = 1 - \mathbf{1}^T \mathbf{p}'(t) \qquad \mathbf{1} = (1,1,1,\ldots)$$

$$\tau = \sum_{t=0}^{\infty} t[p_v(t) - p_v(t-1)] = \sum_{t=0}^{\infty} t\mathbf{1}^T[\mathbf{p}'(t-1) - \mathbf{p}'(t)] = \mathbf{1}^T[\mathbf{I} - \mathbf{A}'\mathbf{D}'^{-1}]^{-1}\mathbf{p}'(0)$$

$$\because \sum_{t=0}^{\infty} t(\mathbf{M}^{t-1} - \mathbf{M}^t) = [\mathbf{I} - \mathbf{M}]^{-1}$$

random walk with absorbing state(3)

$$[\mathbf{I}-\mathbf{A}'\mathbf{D}'^{-1}]^{-1} = \mathbf{D}'[\mathbf{D}'-\mathbf{A}']^{-1} = \mathbf{D}'\mathbf{L}'^{-1} \qquad \because [\mathbf{AB}]^{-1} = \mathbf{B}^{-1}\mathbf{A}^{-1}$$

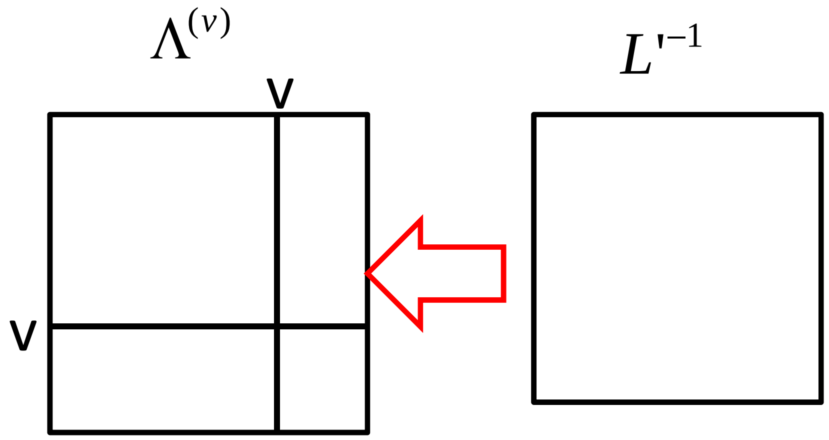
$\mathbf{I} = \mathbf{D}'\mathbf{D}'^{-1}$

L’ : graph Laplacian with the vth row and column removed (vth reduced Laplacian)

$$\tau = \mathbf{1}^T[\mathbf{I}-\mathbf{A}'\mathbf{D}'^{-1}]^{-1}\mathbf{p}'(0) = \mathbf{1}\cdot\mathbf{D}'\mathbf{L}'^{-1}\cdot\mathbf{p}'(0)$$

- L’ can have inverse $\because \mathbf{1} = (1,1,\ldots,1)$ is not an eigenvalue of L’
- $\Lambda^{(v)}$: equal to L'^{-1} with a vth row and column reintroduced

$$\Lambda_{ij}^{(v)} = \begin{cases} 0 & \text{if } i = v \text{ or } j = v \\ [L'^{-1}]_{ij} & \text{if } i < v \text{ and } j < v \\ [L'^{-1}]_{i-1,j} & \text{if } i > v \text{ and } j < v \\ [L'^{-1}]_{i,j-1} & \text{if } i < v \text{ and } j > v \\ [L'^{-1}]_{i-1,j-1} & \text{if } i > v \text{ and } j > v \end{cases}$$



random walk with absorbing state(4)

$$\tau = \mathbf{1} \cdot \mathbf{D}'\mathbf{L}'^{-1} \cdot \mathbf{p}'(0)$$

a walk starting at vertex u at time 0

$$\mathbf{p}'(0) = (0,0,\ldots,\underset{u}{1},0,\ldots,0)$$

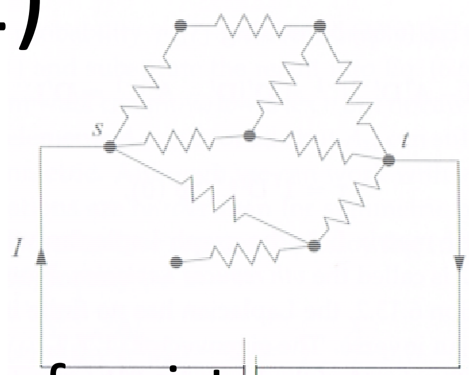
$$\therefore \tau = \sum_i k_i \Lambda_{iu}^{(v)}$$

- calculate L' (vth reduced Laplacian)
- the sum over the elements in the uth column

⇨ the first passage time from u to v

- sums over the other columns ⇨ the first passage time from other starting vertices to v

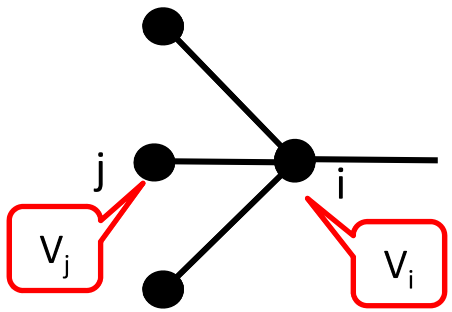
resistor networks(1)



- connection between
  - random walks on networks and
  - calculation of current flows in networks of resistors
- edges: identical resistors of resistance R
- vertices: junctions between resistors
- apply a voltage between s and t such that a current I flows from s to t
- What is the current flow through any given resistor?

resistor networks(2)

- Kirchhoff's current law: electricity is conserved
- V_i: the voltage at vertex i
- I_i : current injected into vertex i



$$I_i = \begin{cases} +I & \text{for i=s,} \\ -I & \text{for i=t,} \\ 0 & \text{otherwise.} \end{cases}$$

$$\sum_i A_{ij} \frac{V_j - V_i}{R} + I_i = 0$$

$$k_i V_i - \sum_j A_{ij} V_j = RI_i \quad \because \sum_j A_{ij} = k_i$$

$$\sum_j \left(\delta_{ij} k_i - A_{ij} \right) V_j = RI_i$$

$$\therefore \mathbf{LV} = R\mathbf{I} \qquad \mathbf{L} = \mathbf{D} - \mathbf{A} \quad \text{graph Laplacian}$$

resistor networks(3)

- the Laplacian has no inverse ⇨ we cannot simply invert $\mathbf{LV} = R\mathbf{I}$ to get V
- this is because we can add any multiple of vector $\mathbf{1} = (1,1,1,\ldots)$ $\quad \mathbf{L}(\mathbf{V} + c\mathbf{1}) = \mathbf{LV} + \mathbf{L1} = \mathbf{LV} = R\mathbf{I}$
- If we fix reference potential at a particular value, then the equation for V will become solvable

the voltage at this point is zero

- ⇨ the voltage at V_t is set to zero

resistor networks(4)

- remove the element V_t=0 from V in $\mathbf{LV} = R\mathbf{I}$

along with the corresponding column t in L

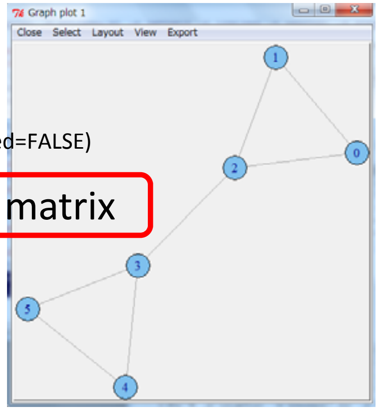
$$\Rightarrow \mathbf{L}'\mathbf{V}' = R\mathbf{I}'$$

- L': tth reduced Laplacian

$$\mathbf{V}' = R\mathbf{L}'^{-1}\mathbf{I}'$$

$V_i = RI\Lambda_{is}^{(t)}$ $\mathbf{\Lambda}^{(t)}$: inverse of the tth reduced Laplacian with the tth row and column reintroduced having elements all zero

graph Laplacian with R+igraph




```
> library(igraph)
> g0 <- graph(c(0,1,1,2,2,0,2,3,3,4,4,5,5,3), directed=FALSE)
> tkplot(g0)
> get.adjacency(g0)
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    0    1    1    0    0    0
[2,]    1    0    1    0    0    0
[3,]    1    1    0    1    0    0
[4,]    0    0    1    0    1    1
[5,]    0    0    0    1    0    1
[6,]    0    0    0    1    1    0
> (gl<-graph.laplacian(g0))
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    2   -1   -1    0    0    0
[2,]   -1    2   -1    0    0    0
[3,]   -1   -1    3   -1    0    0
[4,]    0    0   -1    3   -1   -1
[5,]    0    0    0   -1    2   -1
[6,]    0    0    0   -1   -1    2
> eigen(gl)
$values
[1]  4.561553e+00  3.000000e+00  3.000000e+00  3.000000e+00  4.384472e-01
[6]  7.985906e-17

$vectors
           [,1]          [,2]       [,3]       [,4]       [,5]       [,6]
[1,]  0.1845241  0.000000e+00  0.0000000 -0.7637626  0.4647051 -0.4082483
[2,]  0.1845241  4.196338e-17 -0.5345225  0.5455447  0.4647051 -0.4082483
[3,] -0.6571923 -4.196338e-17  0.5345225  0.2182179  0.2609565 -0.4082483
[4,]  0.6571923 -4.196338e-17  0.5345225  0.2182179 -0.2609565 -0.4082483
[5,] -0.1845241 -7.071068e-01 -0.2672612 -0.1091089 -0.4647051 -0.4082483
[6,] -0.1845241  7.071068e-01 -0.2672612 -0.1091089 -0.4647051 -0.4082483
```

adjacency matrix

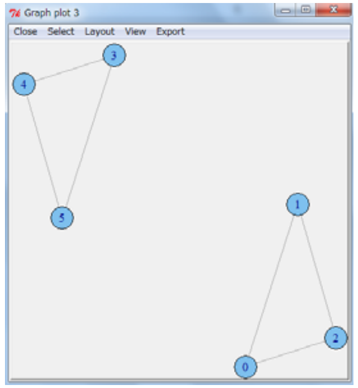
graph Laplacian

eigenvalues / eigenvectors

no.clusters(g0) →1

(# of zero eigenvalues) = 1

$\lambda_2 \neq 0 \longrightarrow$ network is connected




```
> g2 <- graph(c(0,1,1,2,2,0,3,4,4,5,5,3), directed=FALSE)
> tkplot(g2)
[1] 3
> get.adjacency(g2)
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    0    1    1    0    0    0
[2,]    1    0    1    0    0    0
[3,]    1    1    0    0    0    0
[4,]    0    0    0    0    1    1
[5,]    0    0    0    1    0    1
[6,]    0    0    0    1    1    0
> (g2l<-graph.laplacian(g2))
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    2   -1   -1    0    0    0
[2,]   -1    2   -1    0    0    0
[3,]   -1   -1    2    0    0    0
[4,]    0    0    0    2   -1   -1
[5,]    0    0    0   -1    2   -1
[6,]    0    0    0   -1   -1    2
> eigen(g2l)
$values
[1] 3.000000e+00 3.000000e+00 3.000000e+00 3.000000e+00 1.776357e-15
[6] 1.776357e-15

$vectors
           [,1]       [,2]       [,3]       [,4]       [,5]       [,6]
[1,]  0.0000000  0.0000000  0.8164966  0.0000000  0.0000000 -0.5773503
[2,]  0.0000000  0.0000000 -0.4082483 -0.7071068  0.0000000 -0.5773503
[3,]  0.0000000  0.0000000 -0.4082483  0.7071068  0.0000000 -0.5773503
[4,]  0.0000000  0.8164966  0.0000000  0.0000000 -0.5773503  0.0000000
[5,] -0.7071068 -0.4082483  0.0000000  0.0000000 -0.5773503  0.0000000
[6,]  0.7071068 -0.4082483  0.0000000  0.0000000 -0.5773503  0.0000000
```

no bridge between components

block diagonal

no.clusters(g2) →2

(# of zero eigenvalues) = 2

Example: spectral partitioning with R+igraph

signs of the eigenvector corresponding to the second smallest eigenvalue

```
> library(igraph)
> g0 <- graph(c(0,1,1,2,2,0,2,3,3,4,4,5,5,3), directed=FALSE)
> tkplot(g0)
> get.adjacency(g0)
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    0    1    1    0    0    0
[2,]    1    0    1    0    0    0
[3,]    1    1    0    1    0    0
[4,]    0    0    1    0    1    1
[5,]    0    0    0    1    0    1
[6,]    0    0    0    1    1    0
```

adjacency matrix

```
> gl<-graph.laplacian(g0)
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    2   -1   -1    0    0    0
[2,]   -1    2   -1    0    0    0
[3,]   -1   -1    3   -1    0    0
[4,]    0    0   -1    3   -1   -1
[5,]    0    0    0   -1    2   -1
[6,]    0    0    0   -1   -1    2
```

graph Laplacian

```
> eigen(gl)
$values
[1]  4.561553e+00  3.000000e+00  3.000000e+00  3.000000e+00  4.384472e-01
[6] -7.985906e-17

$vectors
           [,1]          [,2]       [,3]       [,4]       [,5]       [,6]
[1,]  0.1845241  0.000000e+00  0.0000000 -0.7637626  0.4647051 -0.4082483
[2,]  0.1845241  4.196338e-17 -0.5345225  0.5455447  0.4647051 -0.4082483
[3,] -0.6571923 -4.196338e-17  0.5345225  0.2182179  0.2609565 -0.4082483
[4,]  0.6571923 -4.196338e-17  0.5345225  0.2182179 -0.2609565 -0.4082483
[5,] -0.1845241 -7.071068e-01 -0.2672612 -0.1091089 -0.4647051 -0.4082483
[6,] -0.1845241  7.071068e-01 -0.2672612 -0.1091089 -0.4647051 -0.4082483
```

eigenvalues / eigenvectors

second smallest

```
> sign(eigen(gl)$vectors[,length(eigen(gl)$values)-1])
[1]  1  1  1 -1 -1 -1
> (sign(eigen(gl)$vectors[,length(eigen(gl)$values)-1])+1)/2
[1] 1 1 1 0 0 0
> V(g0)$color <- (sign(eigen(gl)$vectors[,length(eigen(gl)$values)-1])+1)/2
> tkplot(g0)
```