



華東師範大學

EAST CHINA NORMAL UNIVERSITY

# 数据科学与工程算法基础

Algorithm Foundations of Data Science and Engineering

## 第十三章 社团发现

$$(1+x)^n = 1 + \frac{nx}{1!} + \frac{n(n-1)x^2}{2!} + \dots$$

# 课程提纲

## Content

1 算法引入

2 模块度

3 谱方法

4 Louvain方法

# 课程提纲

## Content

1 算法引入

2 模块度

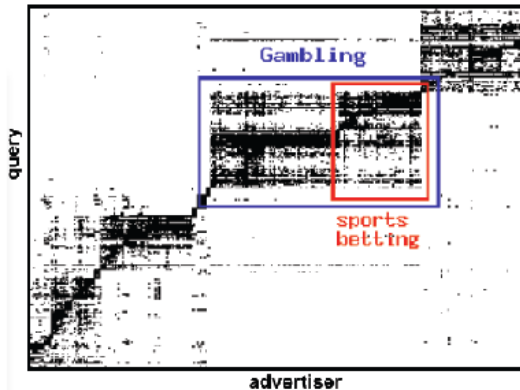
3 谱方法

4 Louvain方法

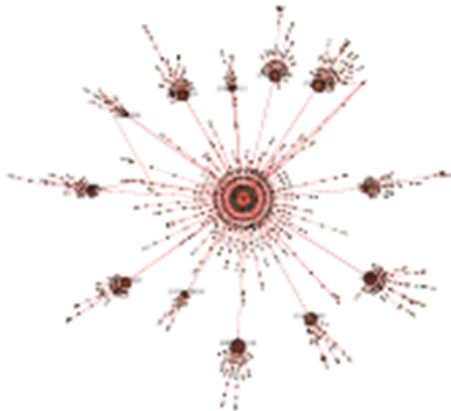
# 网络与社区结构

---

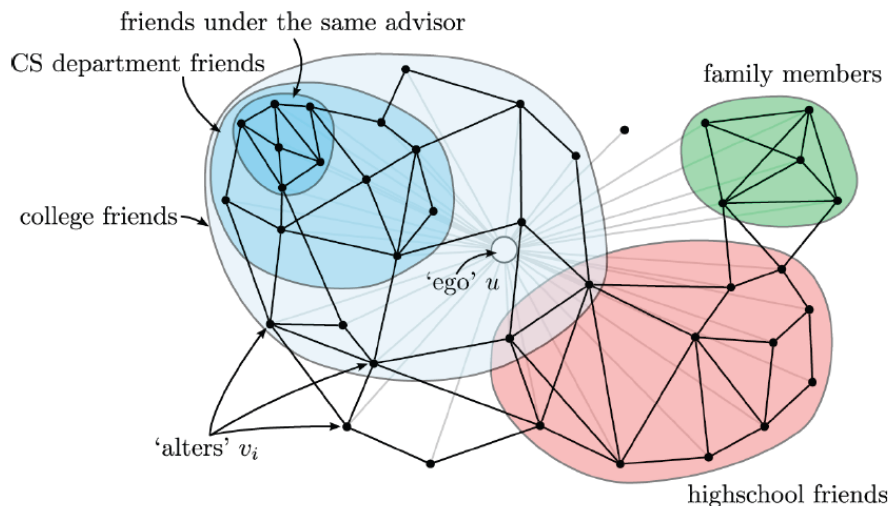
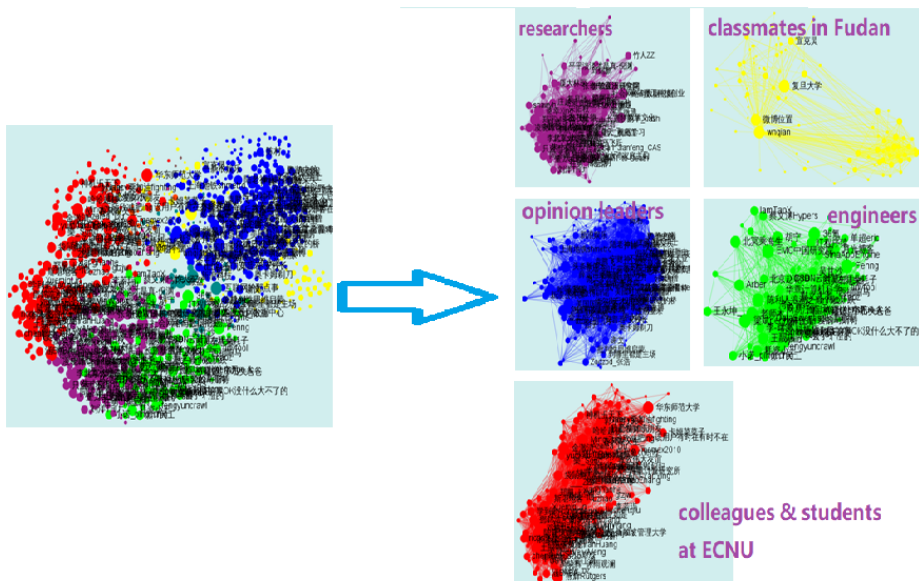
- 社区是一个子图
- 如果子图内部连接比较紧密，子图与子图之间连接比较稀疏，则表明图具有社区结构



- 社区发现有助于
  - 图的可视化
  - 异常检测（水军）
  - 用户画像（物以类聚、人以群分）
- 社区发现是一类广义的聚类方法



# 社区示例



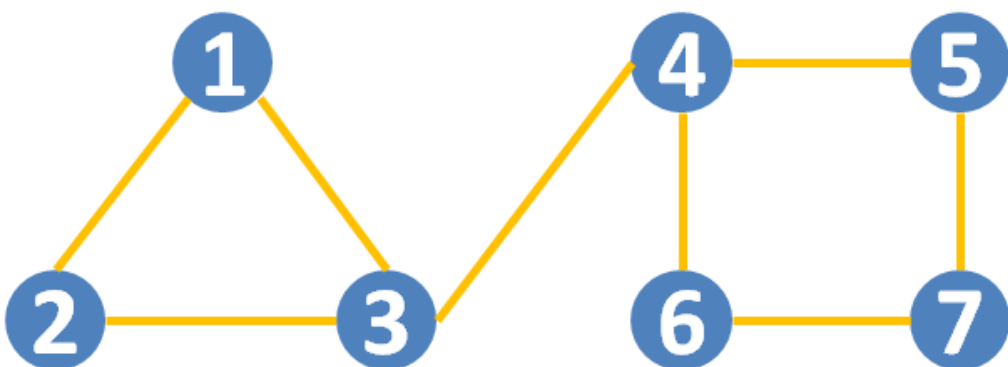
- 在信任网络中，社区类似于“朋友圈”
  - 可视化
  - 用户行为理解
- 社区分类
  - 是否有重叠
    - 无重叠社区
    - 有重叠社区
  - 是否涉及所有顶点
    - 全局社区
    - 局部社区
- 仅探讨无重叠的全局社区结构

# 问题定义

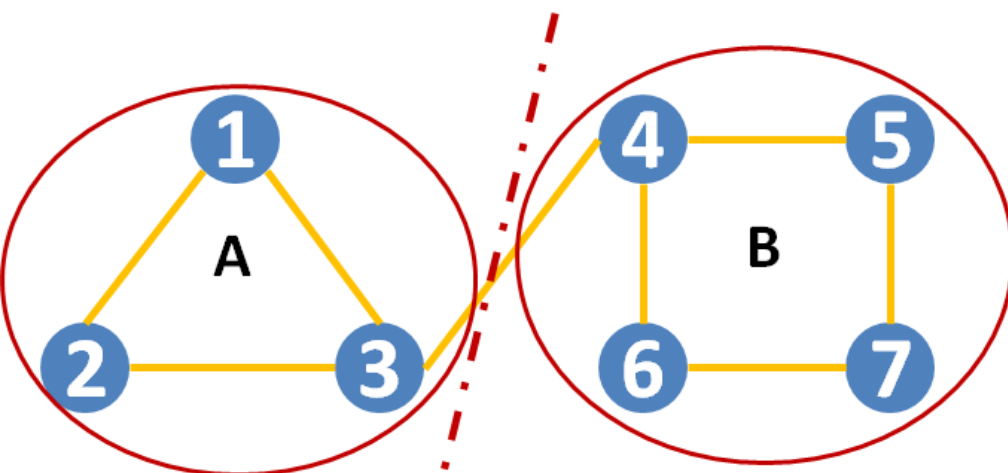
---

- **社区结构**表明网络可以被划分成子图，其内部连接比较紧密，相互间连接比较稀疏
- 然而，实际应用中的图通常都很大
  - 一个拥有 200M 顶点和 2B 边的图，需要 16 GB 内存
  - 由于图太大，运行在图上的算法最多只能是线性复杂度
- 社区发现旨在将子图划分成不相交的子图
  - 基于顶点相似度的聚类
  - 隐空间模型
  - 谱聚类方法
  - 最大化模块度

# 社区划分



- 给定无向图  $G = (V, E)$ 
  - 将其划分为两个不相交的部分  $A$  和  $B = V \setminus A$
  - 除此以外，还有很多种划分方法
- 如何划分才能算是一个好的社区？



# 社区划分标准

---

- 社区划分标准

- 最大化社区内部连接数量
- 最小化社区之间的连接数量

- 割是表达社区好坏的一个标准，定义为  $Cut(A) = \sum_{i \in A, j \notin A} w_{ij}$

- 好的社区划分旨在最小割

- 解决最小割问题的算法复杂度为  $O(|V| \cdot |E|^2)$
- 对于大图行不通



# 课程提纲

## Content

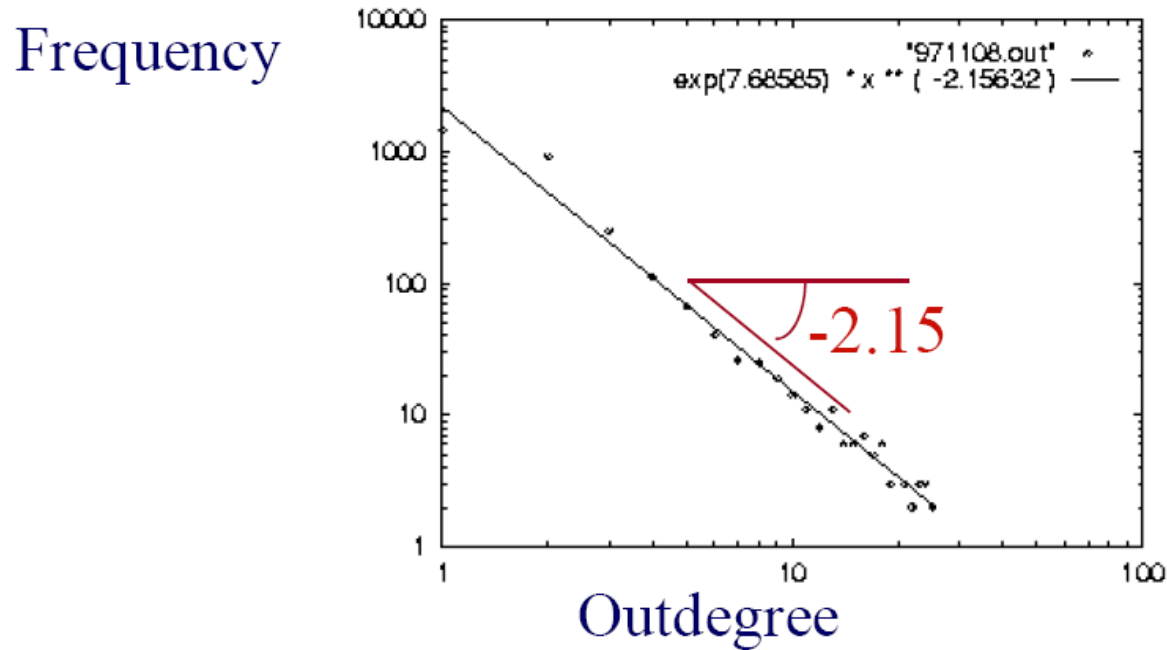
1 算法引入

2 模块度

3 谱方法

4 Louvain方法

# 图模型 — Power Law



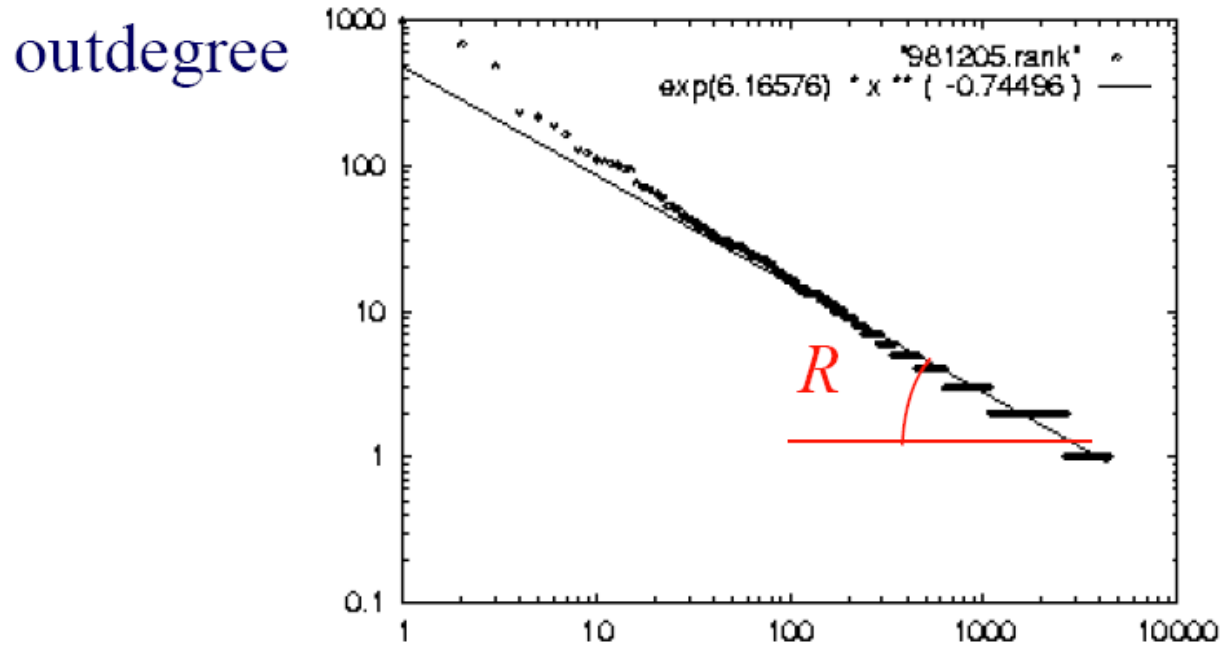
- Internet 拓扑结构图
  - Log-log scale
  - 分布在斜率为  $-2.15$  的直线上
  - $freq \propto deg^{-2.15}$

# Power Law

---

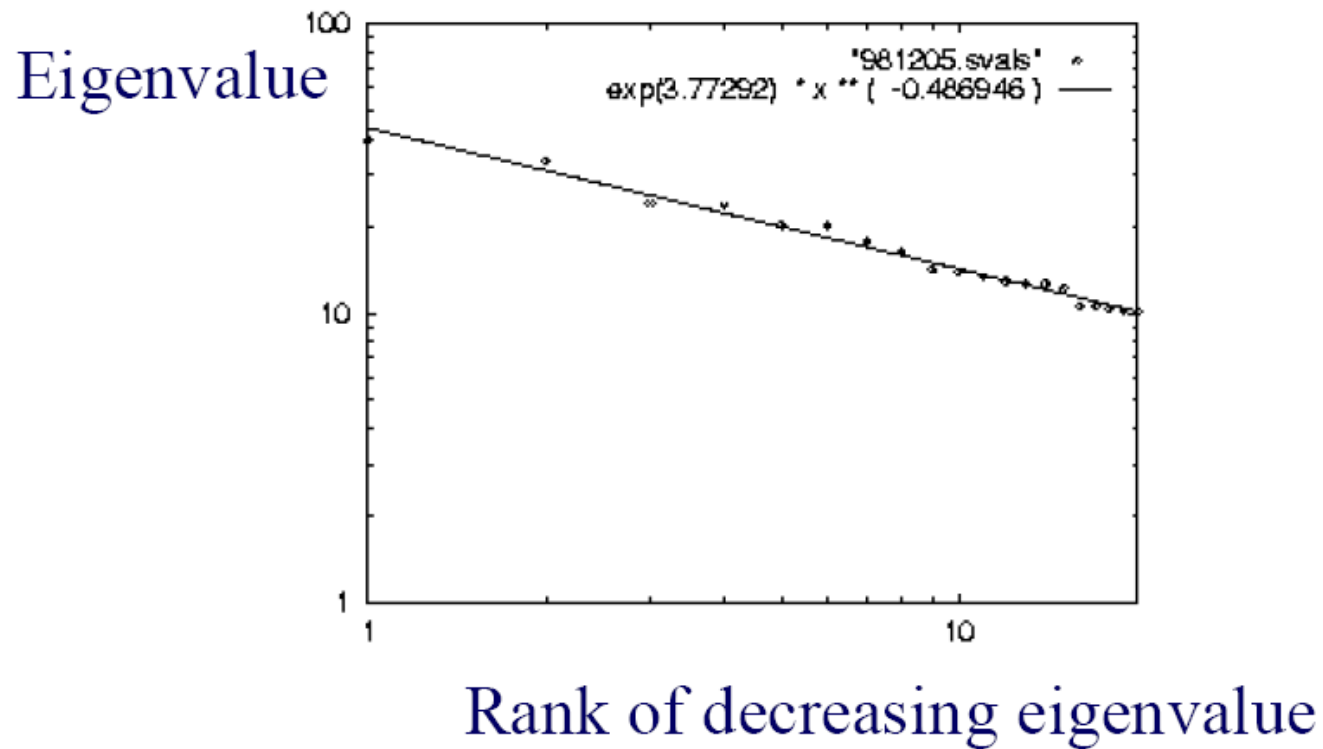
- 又称为“马太效应”、“Pareto规则”、“富人越富”、或者“二八定律”
  - 80% 意大利的土地被 20% 的人所拥有
  - 世界上最富有的 20% 的人拥有 82.70% 的财富
  - 圣经：词频排序 VS. 频率
  - 网页：点击次数 VS. 网页数量
  - 文件：文件数量 VS. 文件大小
  - 商业领域：
    - 20% 的客户贡献企业 80% 的利润
    - 80% 的投诉来自 20% 的客户
    - 80% 的公司利润来自员工 20% 的时间所创造的
    - 80% 的销售利润来自 20% 的员工

# Power Law (II)



- 图中顶点出度的排序
  - 顶点按照出度升序排列
  - $deg \propto rank^{-2.15}$

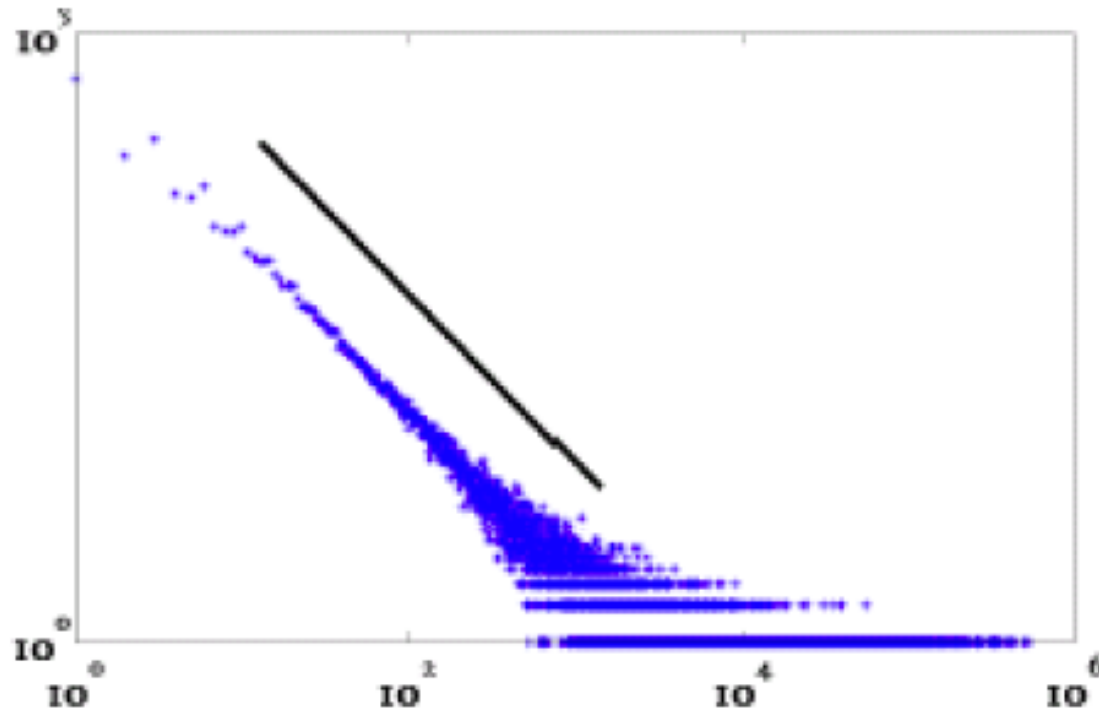
# Power Law (III)



- 特征值排序
  - 邻接矩阵特征值按升序排列
  - $eigen . \propto rank^{-2.15}$

# Power Law (IV)

---



- 顶点涉及Triangle的数量
  - Y轴：顶点涉及Triangle的数量，X轴：顶点数量
  - 在 log-log 尺度下，也分布在一条直线上

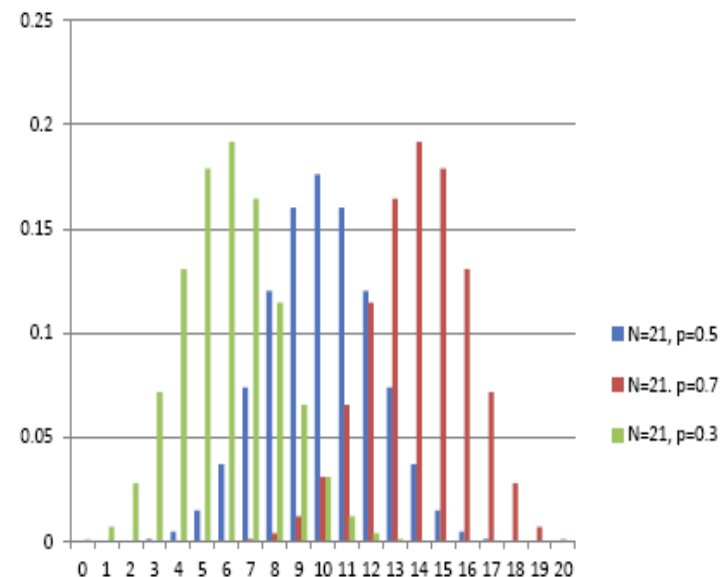
# Erdos–Renyi 模型

- Erdos–Renyi (ER 模型) 是一个随机图模型，用于产生无向随机图

- 参数：  $N$  (顶点数量) 和  $p$  (生成一条无向边的概率)
- 对每组顶点配对，ER 模型以概率  $p$  形成一条无向边
- 因此，该随机图中边的总数为  $|E| = \frac{n(n-1)}{2} \cdot p$
- 顶点度的分布

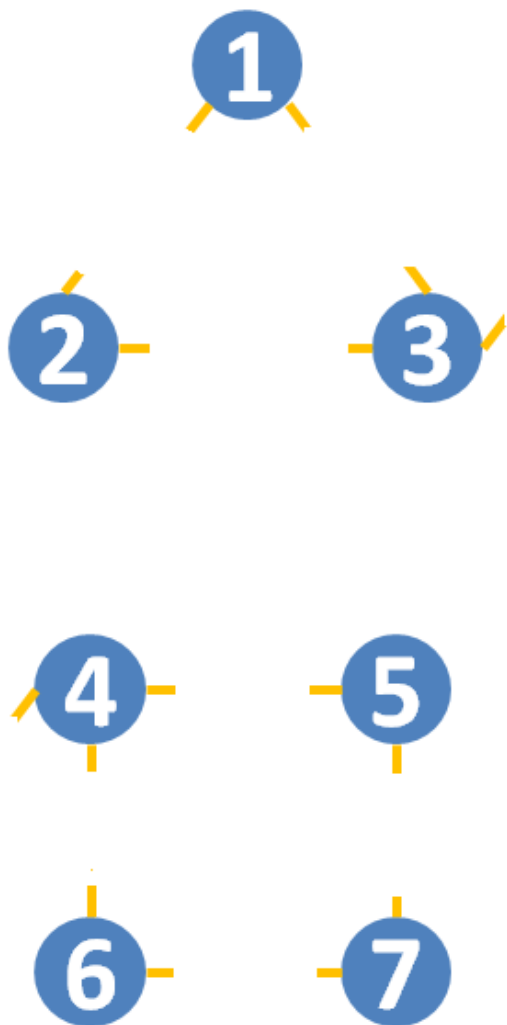
- $P(\deg = k) = \binom{N-1}{k} \cdot p^k \cdot (1-p)^{N-1-k}$

- 服从二项分布
- 期望为  $(N-1) \cdot p$
- 方差为  $(N-1) \cdot p \cdot (1-p)$
- 不是一个长尾分布



# NULL 模型

---



- 顶点  $v_i$  和  $v_j$  间期望边的数量
  - 假设边的起点为  $v_i$
  - 定义随机变量  $X_{ij} = \begin{cases} 1, & \text{目标顶点为 } v_j \\ 0, & \text{otherwise} \end{cases}$
  - 因此,  $P[X_{ij} = 1] = P_{ij}$  为一个伯努利分布
- 进一步, 我们有
  - $\sum_{i,j} E(X_{ij}) = \sum_{i,j} P_{ij} = 2m$
  - $\sum_j E(X_{ij}) = \sum_j P_{ij} = k_i$



# NULL 模型 (续)

---

- 因为  $P_{ij}$  和  $k_i, k_j$  有关, 令  $P_{ij} = f(k_i)f(k_j)$
- 由于  $P_{ij} = P_{ji}$ , 因此  $\sum_j P_{ij} = \sum_j f(k_i)f(k_j) = f(k_i) \sum_j f(k_j) = k_i$
- 因为  $\sum_j f(k_j)$  与 顶点  $v_i$  无关, 所以  $f(k_i) = ck_i$
- 进一步的  $\sum_j f(k_i)f(k_j) = c^2 \sum_{i,j} k_i k_j = 2m$
- 因此,  $P_{ij} = c^2 k_i k_j = \frac{k_i k_j}{2m}$

# NULL 模型的直观意义

---

- 令  $X_i$  为独立同分布的  $\text{Bernoulli}(\frac{k_i}{2m})$ , 其中  $X_i = 1$  意味着起点为  $v_i$ , 否则取值为 0
- 类似的, 令  $Y_i$  为独立同分布的  $\text{Bernoulli}(\frac{k_i}{2m})$ , 其中  $Y_i = 1$  意味着终点为  $v_i$ , 否则取值为 0
- 对于边  $e_{ij} \in E$ 
  - 意味着起点和终点分别为  $v_i$  和  $v_j$
  - 令  $Z_{ij} = \begin{cases} 1, & e_{ij} \in E \\ 0, & \text{otherwise} \end{cases}$

# NULL 模型的直观意义 (续)

---

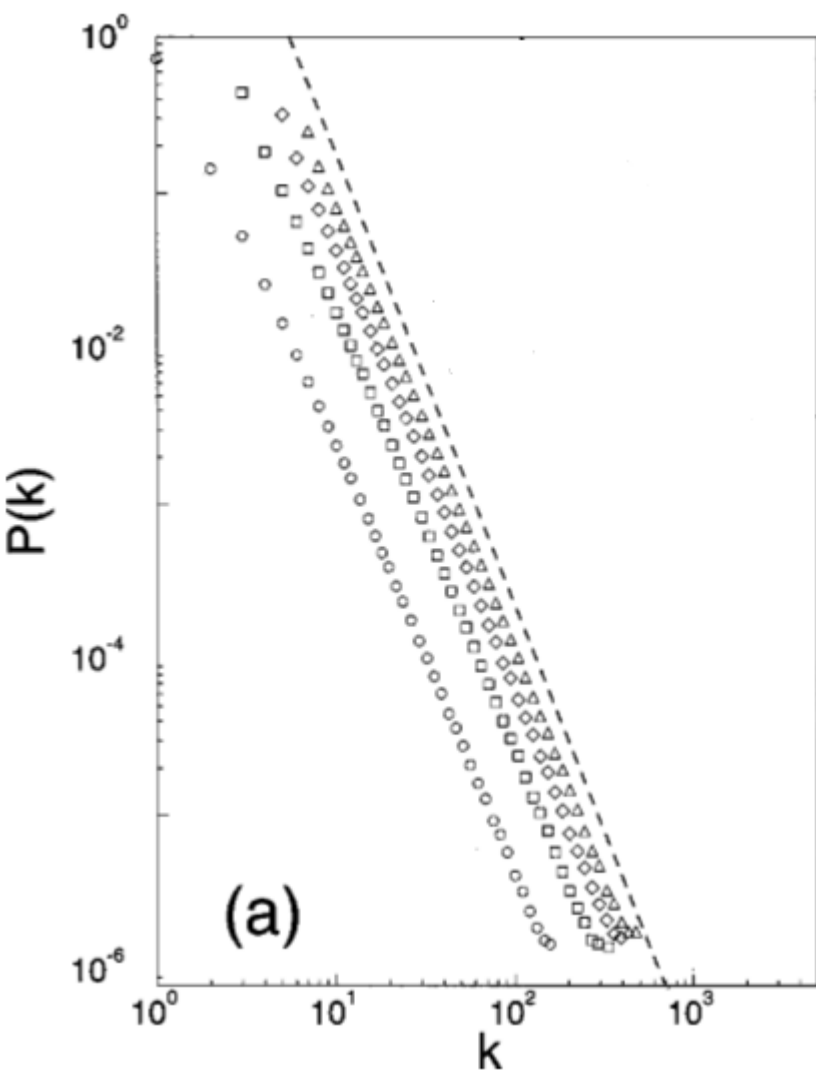
- 给定一个网络, 有  $2m$  条有向边
  - 每次往图中插入一条边是一个伯努利试验
  - 因此,  $\sum_{i,j} Z_{ij} \sim \text{Binomial}(2m, p)$
- 因为  $p = P(Z_{ij} = 1) = \frac{k_i}{2m} \cdot \frac{k_j}{2m}$
- 因此, 顶点  $v_i$  和  $v_j$  之间边的数量的期望为
$$E\left(\sum_{i,j} Z_{ij}\right) = 2mP(Z_{ij} = 1) = \frac{k_i k_j}{2m}$$

# Scale-Free 网络

---

- Preferential Attachment 模型
  - 连接比较多的顶点更可能获得新的连接（富人越富）
  - 价格模型
  - Barabasi-Albert 模型
- 价格模型
  - 以引用网络为例
  - 每篇新论文产生  $m$  个引用
  - 新论文引用已有论文的概率和它的度成正比
  - 产生的网络满足参数  $\alpha = 2 + \frac{1}{m}$  的长尾分布

# Barabasi–Albert 模型



- Barabasi–Albert 模型

- 初始化拥有  $m_0$  个顶点的网络
- 每个顶点的度不小于 1
- 对每一个新顶点，连接到  $m$  个存在顶点中的顶点  $i$  的概率为  $p_i$ ，其中

$$p_i = \frac{k_i}{\sum_j k_j}$$

- 产生具有一个连通分量的 Scale-Free 网络，其中度的分布参数为  $\alpha = 3$

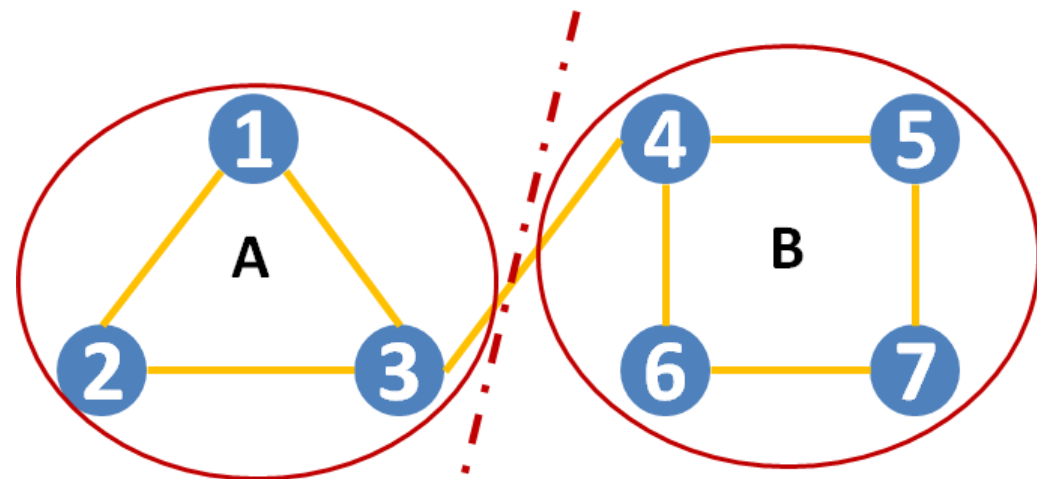
# Modularity 定义

---

- Modularity (模块度) 是网络社区结构的一个度量指标
- 给定无向图  $G = (V, E)$ ,  $A$  为邻接矩阵, 其中
$$A_{ij} = \begin{cases} 1, & (v_i, v_j) \in E \\ 0, & \text{otherwise} \end{cases}$$
- Modularity 定义为  $Q = \frac{1}{2m} \sum_{i,j} \left( A_{ij} - \frac{k_i k_j}{2m} \right) \delta(C_i, C_j)$ , 其中  
 $m$  和  $C_i$  表示图中顶点数量和顶点  $v_i$  所在的社区,  $k_i$  为顶点  $v_i$  的度, 且  $\delta(C_i, C_j) = \begin{cases} 1, & \text{if } C_i = C_j \\ 0, & \text{otherwise} \end{cases}$

# Modularity 示例

---

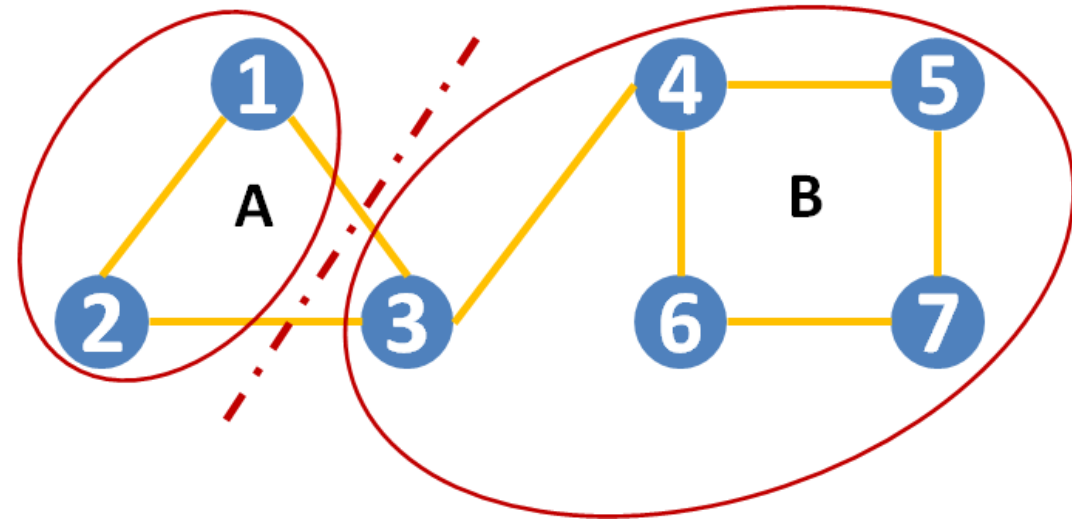


$$m = 8, k_1 = 2, k_2 = 2, k_3 = 3$$

$$k_4 = 3, k_5 = 2, k_6 = 2, k_7 = 2$$

$$Q = \frac{1}{2m} \sum_{i,j} \left( A_{ij} - \frac{k_i k_j}{2m} \right) \delta(C_i, C_j) = \frac{47}{128}$$

# Modularity 示例 (续)



$$m = 8, k_1 = 2, k_2 = 2, k_3 = 3$$
$$k_4 = 3, k_5 = 2, k_6 = 2, k_7 = 2$$

$$Q = \frac{1}{2m} \sum_{i,j} \left( A_{ij} - \frac{k_i k_j}{2m} \right) \delta(C_i, C_j) = \frac{1}{8}$$



# Modularity 的直观含义

---

- Modularity 度量了社区结构的好坏

- 给定网络的一个划分  $C$ , 对每个社区  $c \in C$ ,

$$\begin{aligned} Q &\propto \sum_{i,j} \left( A_{ij} - \frac{k_i k_j}{2m} \right) \delta(C_i, C_j) = \left( \sum_{i,j} A_{ij} - \sum_{i,j} \frac{k_i k_j}{2m} \right) \delta(C_i, C_j) \\ &= \sum_{i,j} A_{ij} \delta(C_i, C_j) - \sum_{i,j} \frac{k_i k_j}{2m} \delta(C_i, C_j) \\ &= \sum_{c \in C} [(\text{num. edges in } c) - (\text{expected num. edges in } c)] \end{aligned}$$

- 给定具有  $n$  个顶点和  $m$  条边的真实网络  $G$ , 依据 NULL 模型构建网络  $G'$ 
  - $G'$  与  $G$  的顶点具有相同度的分布
  - 网络  $G'$  是一个多图

# Modularity: 社区 VS. NULL 模型

---

$$Q \propto \sum_{c \in C} [(\text{num. edges in } c) - (\text{expected num. edges in } c)]$$

- Modularity 计算了真实网络结构与基于 NULL 模型的网络结构间的偏差大小
  - NULL 模型是随机连接两个顶点的，没有特别的网络结构
  - 对于给定划分，Modularity 比较了真实网络与随机网络中的社区结构
  - Modularity 的值越大意味着社区结构越强
  - 因此，modularity 可用于评价社区结构的好坏
  - 也可以用于发现社区结构

# 计算 Modularity

---

$$Q = \frac{1}{2m} \sum_{c \in C} \sum_{i \in c} \sum_{j \in c} (A_{ij} - \frac{k_i k_j}{2m})$$

- Modularity 的值属于  $[-1, 1]$
- 其值为正，表明网络划分具有社区结构
- 其值大于 0.3，表明网络划分具有较好的社区结构
- 给定一个划分，利用 modularity 来衡量该划分的好坏

# 权重图的 Modularity

---

- 给定一个无向权重图  $G = (V, E)$ , 其权重矩阵为

$$W_{ij} = \begin{cases} w_{ij}, & A_{ij} = 1 \\ 0, & A_{ij} = 0 \end{cases}$$

- Modularity 定义为  $Q = \frac{1}{2m} \sum_{i,j} \left( w_{ij} - \frac{k_i k_j}{2m} \right) \delta(C_i, C_j)$
- 注意  $m, k_i, k_j$  的含义变化

# 有向图的 Modularity

---

- Modularity（模块度）也可以定义有向图上的社区结构
- 给定有向图  $G = (V, E)$ ,  $A$  为邻接矩阵
- Modularity 定义为  $Q = \frac{1}{m} \sum_{i,j} \left( A_{ij} - \frac{k_i^{out} k_j^{in}}{m} \right) \delta(C_i, C_j)$

# 课程提纲

## Content

1 算法引入

2 模块度

3 谱方法

4 Louvain方法

# Modularity 矩阵

---

- 给定某网络包含  $n$  个顶点，假定该网络被划分成 2 个社区，令  $s_i = \begin{cases} 1, & \text{if } v_i \in c_1 \\ -1, & \text{otherwise} \end{cases}$
- 该划分对应的 modularity 可以重写为

$$\begin{aligned} Q &= \frac{1}{4m} \sum_{i,j} \left( A_{ij} - \frac{k_i k_j}{2m} \right) (s_i s_j + 1) \\ &= \frac{1}{4m} \sum_{i,j} \left( A_{ij} - \frac{k_i k_j}{2m} \right) s_i s_j \\ &= \frac{1}{4m} \mathbf{s}^\top \mathbf{B} \mathbf{s} \end{aligned}$$

其中  $B_{ij} = A_{ij} - \frac{k_i k_j}{2m}$  被称为 Modularity 矩阵

# 谱方法

---

- 注意到 Modularity 矩阵行和、列和均为 0
  - 意味着  $(1,1,\dots,1)$  为特征值 0 对应的特征向量
  - 将向量  $\mathbf{s}$  表示成 Modularity 矩阵  $\mathbf{B}$  的正交特征向量  $\mathbf{u}_i$  的线性组合, 即  $\mathbf{s} = \sum_{i=1}^n \mathbf{a}_i \mathbf{u}_i$ , 其中  $\mathbf{a}_i = \mathbf{u}_i^\top \cdot \mathbf{s}$
- 因此,  $Q = \frac{1}{4m} \sum_i \mathbf{a}_i \mathbf{u}_i^\top \mathbf{B} \sum_j \mathbf{a}_j \mathbf{u}_j = \frac{1}{4m} \sum_{i=1}^n (\mathbf{u}_i^\top \cdot \mathbf{s})^2 \beta_i$ , 其中  $\beta_i$  为矩阵  $\mathbf{B}$  的特征值, 其对应的特征向量为  $\mathbf{u}_i$
- 不妨假设特征值按照降序排列, 即  $\beta_1 \geq \beta_2 \geq \dots \geq \beta_n$



# 谱方法（续）

---

- 社区发现旨在找到一个划分使得 Modularity 值最大
  - 相当于找到合适的向量  $\mathbf{s}$  使得  $\frac{1}{4m} \sum_{i=1}^n (\mathbf{u}_i^\top \cdot \mathbf{s})^2 \beta_i$  的值最大
  - 仅考虑最大特征值  $\beta_1$
  - 如果向量  $\mathbf{s}$  没有任何限制则  $\mathbf{s} = c\mathbf{u}_1$ ，其中  $c$  为常数
  - 不幸的是， $\mathbf{s}$  的每个分量取值为  $\pm 1$
  - 为了使得 modularity 取得最大值，定义  $s_i = \begin{cases} 1, & \text{if } \mathbf{u}_{1i} > 0 \\ 0, & \text{otherwise} \end{cases}$
  - 换句话说，向量  $\mathbf{u}_1$  的所有正分量对应的顶点被划分到同一个社区，其他顶点在另外一个社区

# 多社区划分方法

---

- 按照谱聚类方法，一个网络仅被划分成两个社区
- 如何利用该方法划分更多社区进一步增加 Modularity 的值呢？
  - 重复运用该方法不断将大的社区划分成小社区
  - 这种方法可行吗？
  - 存在以下问题
    - 最终的 Modularity 可能会改变，因为一些边在后续划分中被删除了
    - 后续每次 Modularity 最大化的划分不一定实现 Modularity 的最大化
  - 最好的方法应该计算 Modularity 的增量

# Modularity 增量计算方法

---

- 最大化  $\Delta Q = \frac{1}{4m} \mathbf{s}^\top \mathbf{B}^{(g)} \mathbf{s}$ , 其中  $B_{ij}^{(g)} = B_{ij} - \delta_{ij} \sum_{k \in g} B_{ik}$
- 问题转化为一个新的谱聚类方法
- 重复以上过程, 知道 Modularity 不再增加, 算法停止

# 课程提纲

## Content

1 算法引入

2 模块度

3 谱方法

4 Louvain方法

# Louvain 方法

---

- 一种基于贪心策略的社区发现算法
  - 支持无向图、有向图和权重图
  - 产生层次状的社区结构
  - 社区数量不是一个超参数
  - 可以有效发现大图中的社区结构
    - 运行时间仅为  $O(|E|)$
    - Modularity 逐渐增加，而且收敛速度快
- 基于贪心策略，Louvain 算法逐渐增加 modularity
  - 每轮迭代由两个阶段组成
  - 不停迭代，直到 modularity 不再增加为止
  - 提供自底向上的社区结构

# 改写 Modularity

$\Sigma_{in}$ :



$$Q = \frac{1}{2m} \sum_{i,j} \left( A_{ij} - \frac{k_i k_j}{2m} \right) \delta(C_i, C_j)$$
$$= \sum_{c \in C} \left[ \frac{\Sigma_{in}^c}{2m} - \left( \frac{\Sigma_{tot}^c}{2m} \right)^2 \right]$$

- $\Sigma_{in}^c$  为社区  $c$  内部边上权重之和
- $\Sigma_{tot}^c$  为社区  $c$  内部顶点连接的边上权重之和

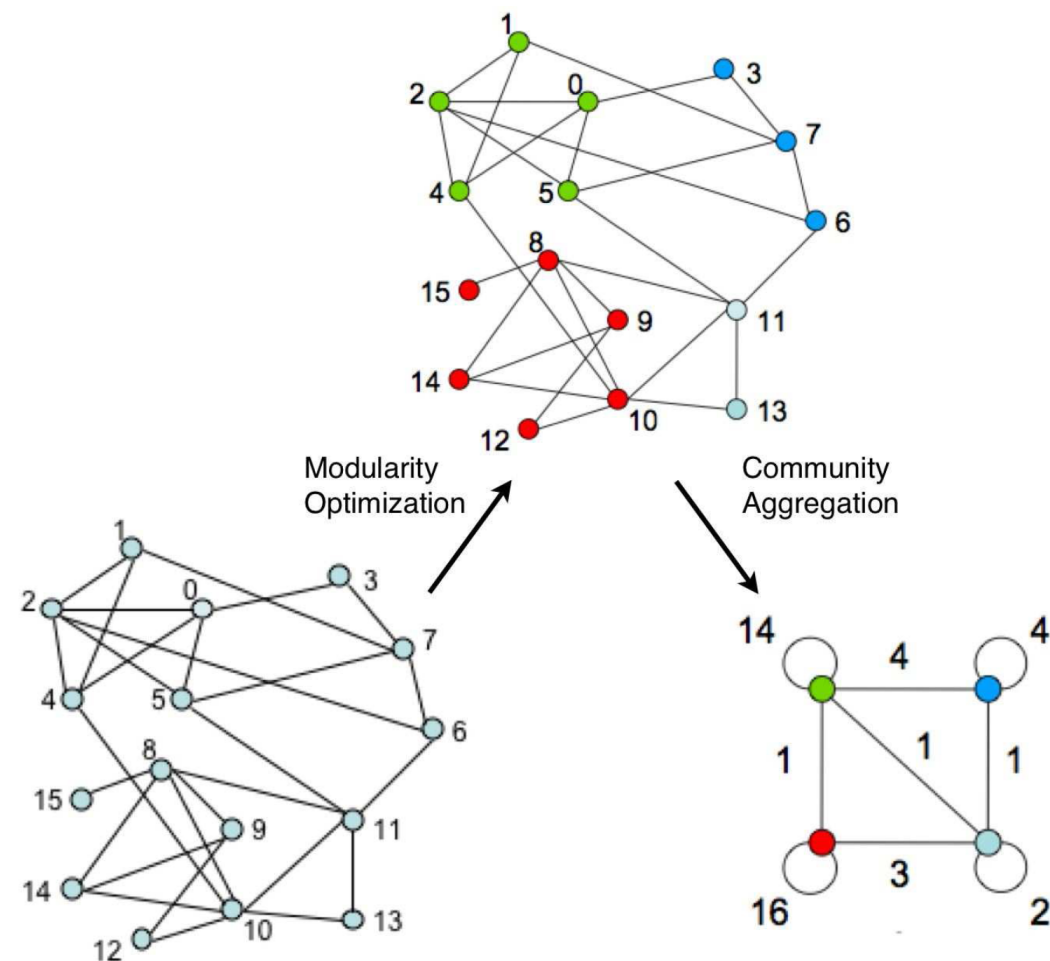
$\Sigma_{tot}$ :



# Louvain 算法

- 每轮迭代

- **第一阶段：**通过局部调整社区实现 modularity 最大化（仅移动一个顶点）
- **第二阶段：**一个社区看作一个超级顶点，构建一个新的权重图



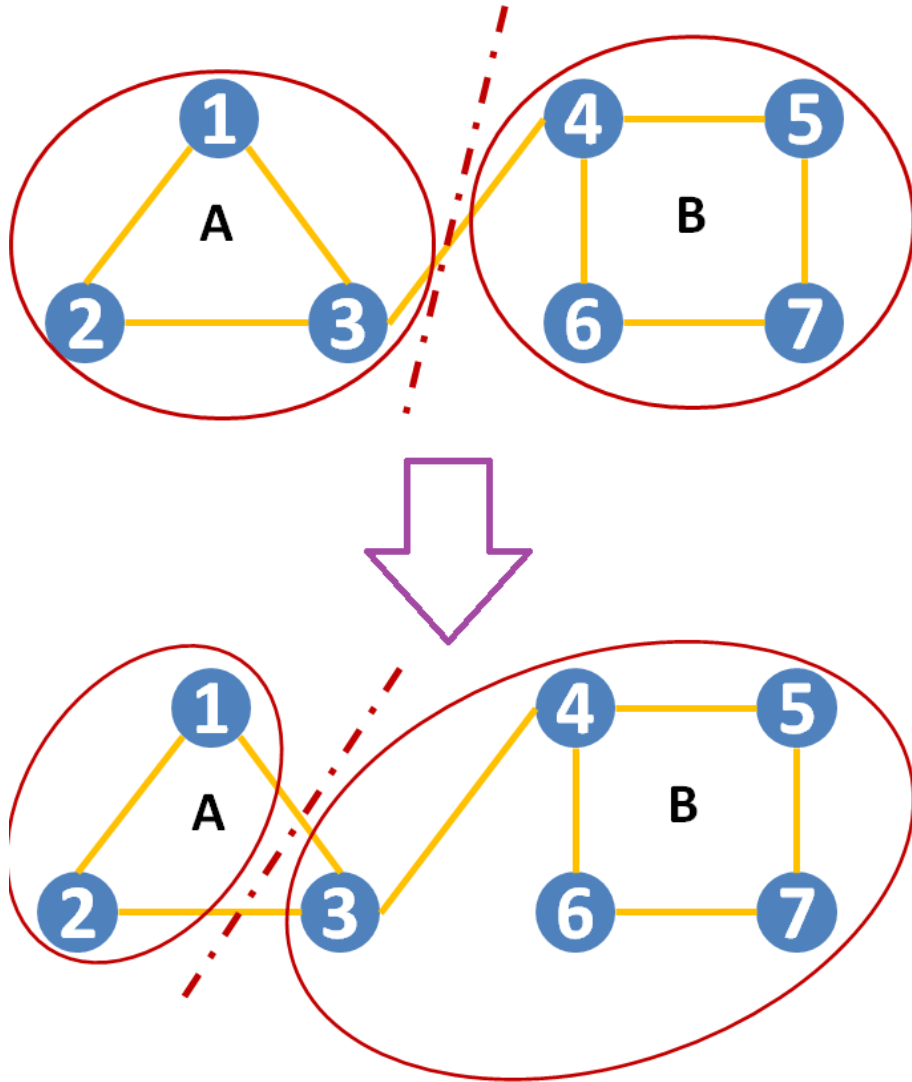
# Louvain 算法第一阶段

---

- 初始状态
  - 每个顶点属于不同的社区
  - 自底向上的方式构成层次状社区结构
- 第一阶段的局部调整，对每个顶点  $v_i$ ，计算
  - 当顶点  $v_i$  移动到它邻居  $v_j$  所在社区时，Modularity 增加量  $\Delta Q$
  - 将顶点  $v_i$  移动到使得  $\Delta Q$  增加最大的那个社区
  - 循环执行，直到移动任何顶点都不会增加 Modularity 为止
- 第一阶段实现了 modularity 的局部最大化
- 虽然顶点访问顺序会影响到算法结果，但是已有研究表明：顶点访问顺序不会对 Modularity 产生大的影响



# Modularity 增加量 $\Delta Q$



- Modularity 的增加量体现在两部分
  - 收益:  $\Delta Q(v_i \rightarrow C)$  表示当顶点  $v_i$  移进社区  $C$  中时 Modularity 的增加量
  - 损失:  $\Delta Q(D \rightarrow v_i)$  表示当顶点  $v_i$  从社区  $D$  中移出 Modularity 的减少量
- 是否移动顶点  $v_i$  需要综合考虑这两部分

# 计算 $\Delta Q(v_i \rightarrow C)$

---

$$\begin{aligned}\Delta Q(v_i \rightarrow C) &= Q_{C+v_i} - (Q_C + Q_{v_i}) \\ &= \frac{k_{i,in}^C}{2m} - \frac{\Sigma_{tot}^C k_i}{2m^2}\end{aligned}$$

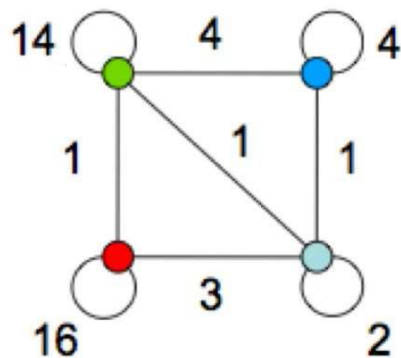
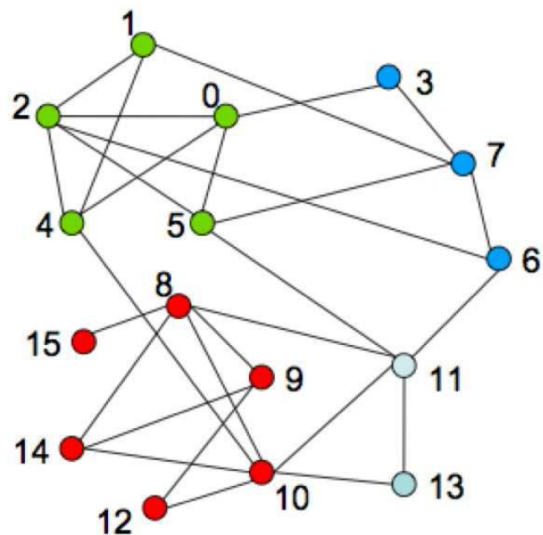
- $\Sigma_{in}^C$  为社区  $C$  内部边上权重之和
- $\Sigma_{tot}^C$  为社区  $C$  内部顶点连接的边上权重之和
- $k_{i,in}^C$  为顶点  $v_i$  与社区  $C$  之间的边的权重和
- $k_i$  为顶点  $v_i$  的度或者边的权重和

# 计算 $\Delta Q(D \rightarrow v_i)$

---

- 令  $D'$  表示顶点  $v_i$  从社区  $D$  中移出后的社区结构
- 因此,  $\Delta Q(D \rightarrow v_i) = -\Delta Q(v_i \rightarrow D')$
- 进一步的,  $\Delta Q(D \rightarrow v_i) = \frac{\sum_{tot}^{D'} k_i}{2m^2} - \frac{k_{i,in}^{D'}}{2m}$
- 最终  $\Delta Q(v_i, C, D) = \Delta Q(v_i \rightarrow C) + \Delta Q(D \rightarrow v_i)$
- 顶点  $v_i$  将被移到使得  $\Delta Q$  增加最大的那个社区

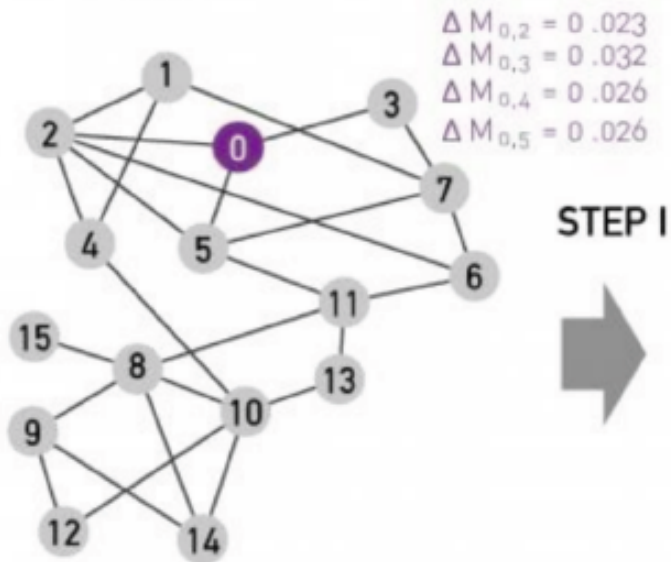
# Louvain 算法第二阶段



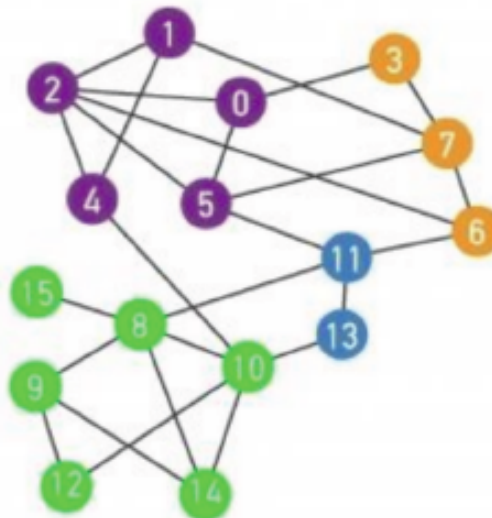
- 构建新的超图
  - 顶点：每个社区抽象成一个顶点
  - 边：如果两个社区间存在至少一条边，则两个超级顶点是连通的
  - 边的权重：为两个社区间所有连接边的权重之后
- 超图
  - 是一个权重图
  - 比输入图小很多
- 因此，复杂度最高的操作是在第一轮迭代中，而且复杂度为  $O(|E|)$

# Louvain 算法示例

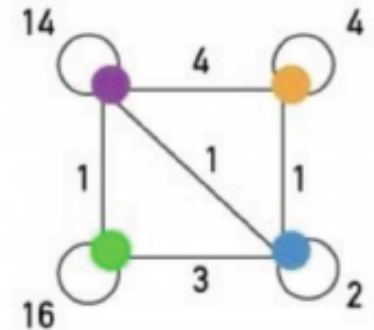
## 1<sup>ST</sup> PASS



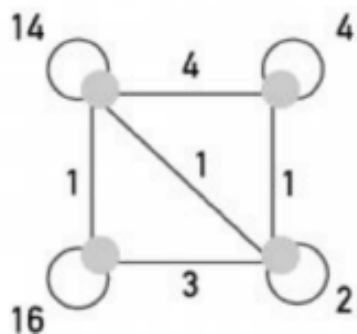
STEP I



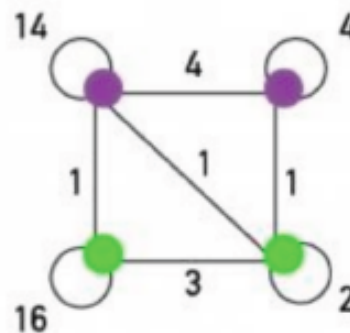
STEP II



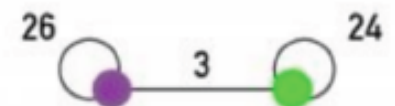
## 2<sup>ND</sup> PASS



STEP I

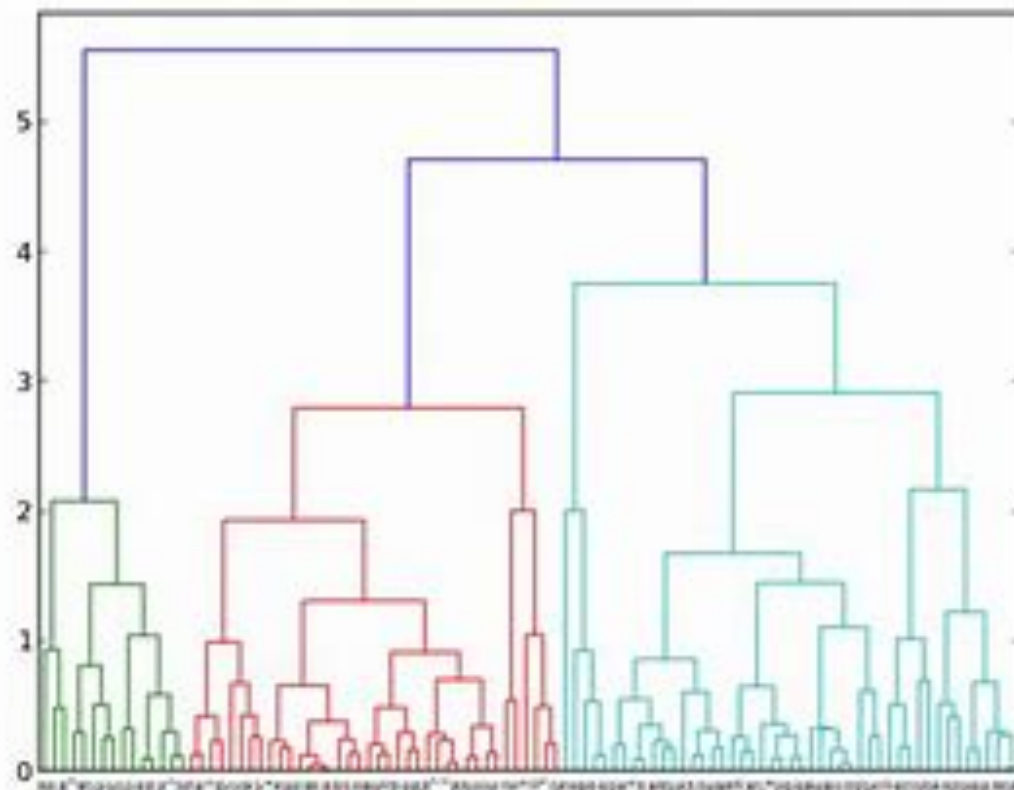


STEP II



# Louvain 算法优势

---



层次状社区结构

# 实验结果

---

|               | Karate | Arxiv     | Internet  | Web nd.edu |
|---------------|--------|-----------|-----------|------------|
| Nodes/links   | 34/77  | 9k/24k    | 70k/351k  | 325k/1M    |
| CNM           | .38/0s | .772/3.6s | .692/799s | .927/5034s |
| PL            | .42/0s | .757/3.3s | .729/575s | .895/6666s |
| WT            | .42/0s | .761/0.7s | .667/62s  | .898/248s  |
| Our algorithm | .42/0s | .813/0s   | .781/1s   | .935/3s    |

---

---

|               | Phone     | Web uk-2005 | Web WebBase 2001 |
|---------------|-----------|-------------|------------------|
| Nodes/links   | 2.6M/6.3M | 39M/783M    | 118M/1B          |
| CNM           | -/-       | -/-         | -/-              |
| PL            | -/-       | -/-         | -/-              |
| WT            | .56/464s  | -/-         | -/-              |
| Our algorithm | .769/134s | .979/738s   | .984/152mn       |

---

# 本章小结

---

- 社区发现是常见的一类图数据挖掘任务
  - 图可视化
  - 异常检测
  - 用户画像
- 社区类型
  - 全局社区结构
  - 局部社区结构 (Clique、Biclique、Quasi-clique)
- 全局社区发现
  - 谱聚类方法
  - Louvain 算法