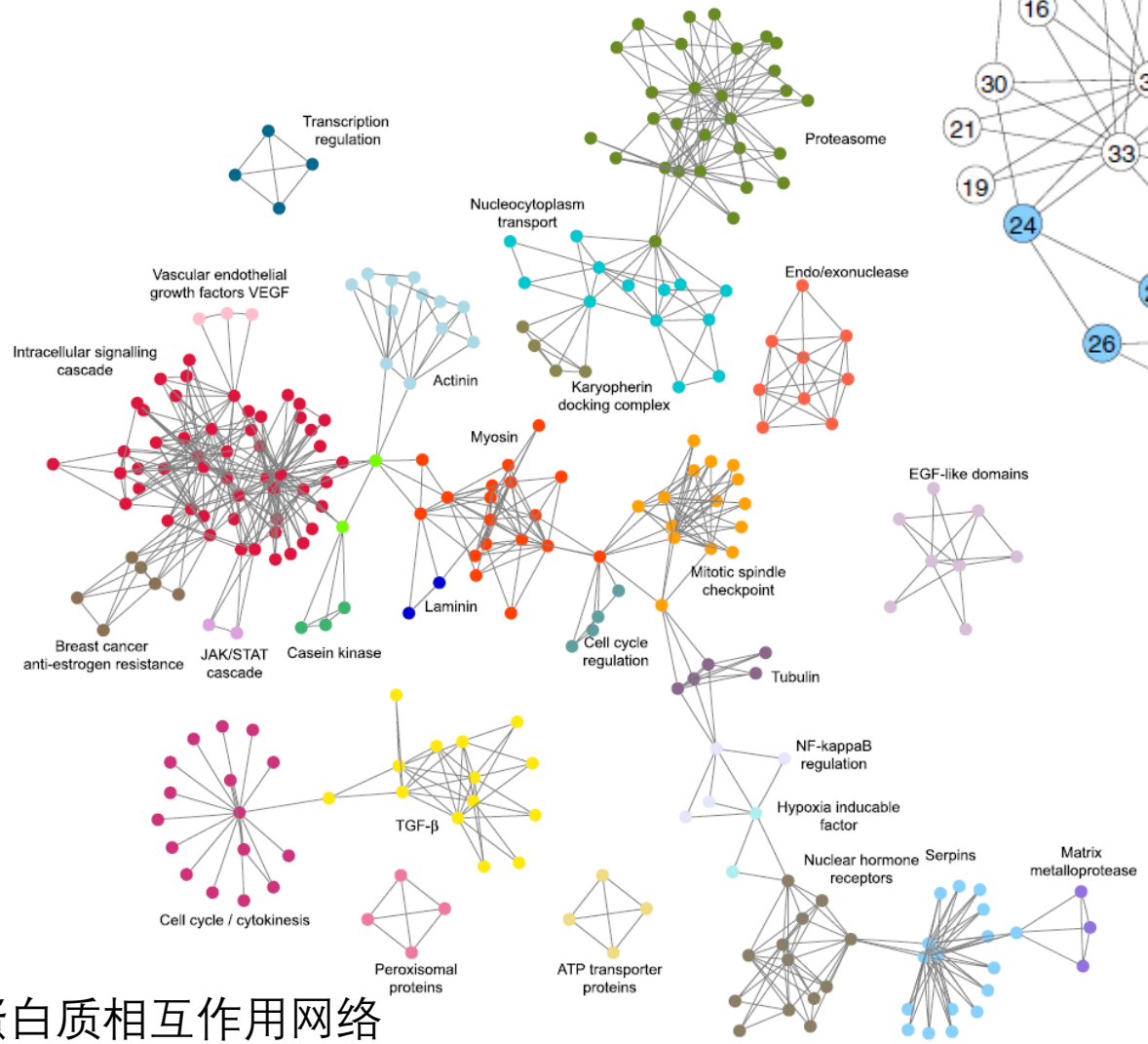
图聚类算法简介

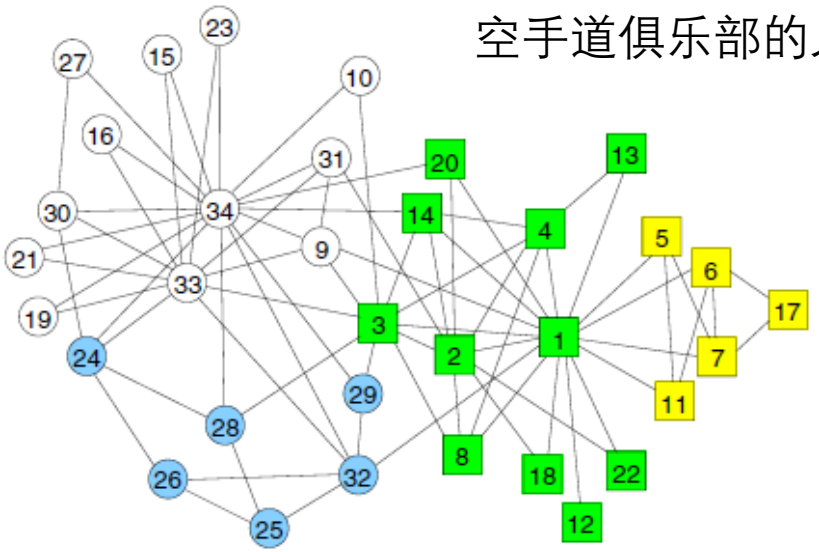
2017.10

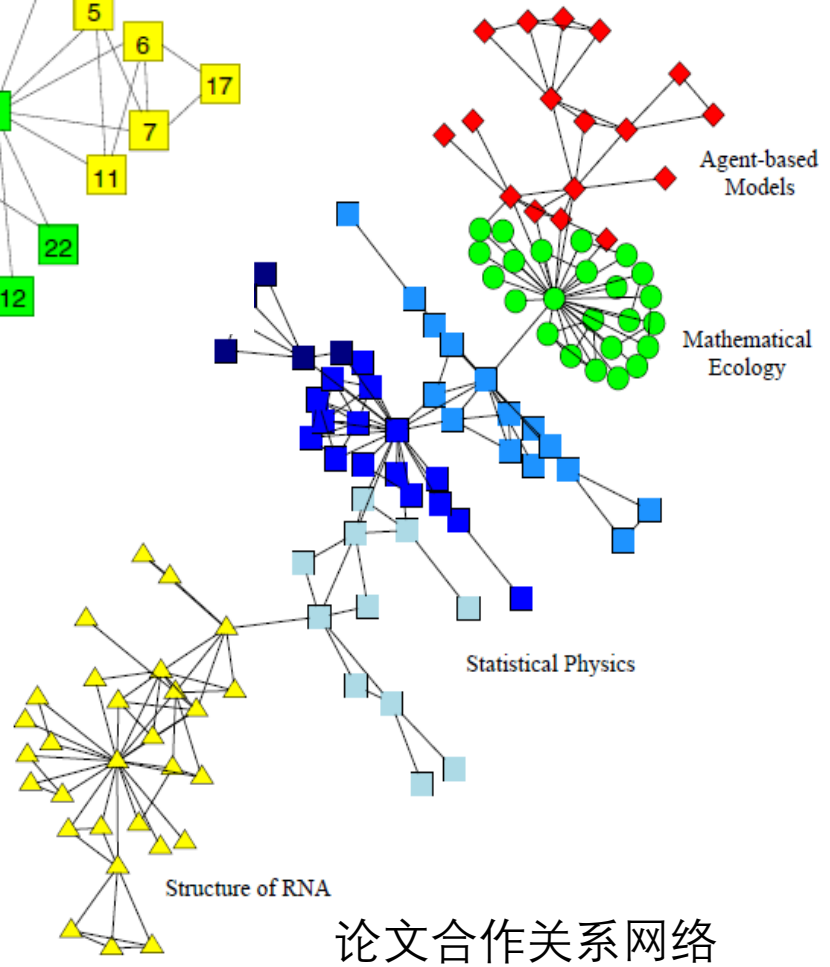
网络社区结构



蛋白质相互作用网络

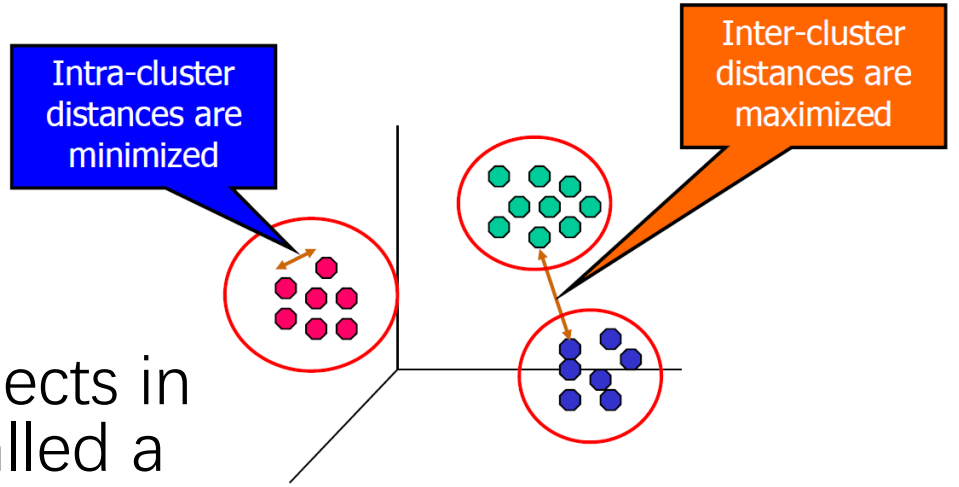
空手道俱乐部的人际关系网络



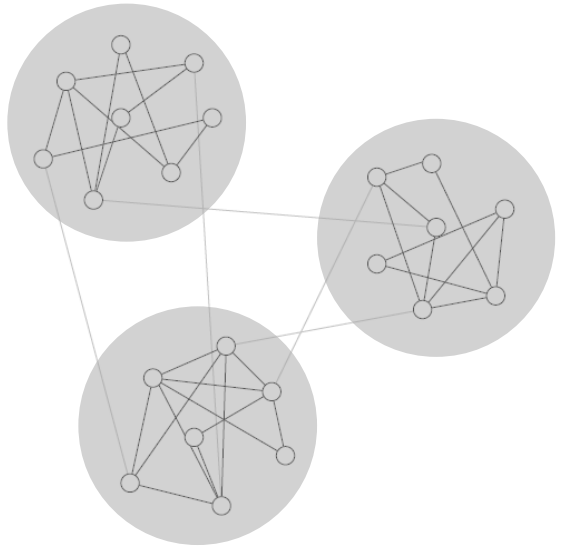


论文合作关系网络

聚类



- **维基百科上对于聚类的定义：**
- **Clustering** is the task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar (in some sense or another) to each other than to those in other groups (clusters).
- 对于什么是“类”，没有统一的、定量的定义，通常由聚类算法定义。
- 对象是空间中的点：类内距离尽可能小，类间距离尽可能大。
- 对象是网络中的节点：类内连接尽可能紧密，类间连接尽可能松散。
- 聚类 v.s. 分类？

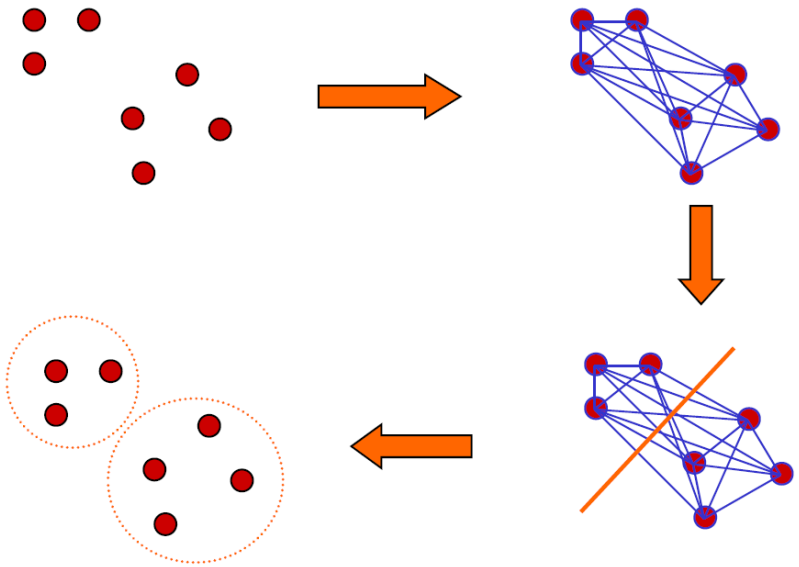


距离和相似性

- 距离越远，相似性越低。
- 欧几里得距离：$\mathrm{dis}(\boldsymbol{x},\boldsymbol{y}) = \sqrt{(\boldsymbol{x}-\boldsymbol{y})^2} = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$
- 曼哈顿距离：$\mathrm{dis}(\boldsymbol{x},\boldsymbol{y}) = |\boldsymbol{x}-\boldsymbol{y}| = \sum_{i=1}^{n}|x_i - y_i|$
- 切比雪夫距离：$\mathrm{dis}(\boldsymbol{x},\boldsymbol{y}) = \max_{1\le i\le n}|x_i - y_i|$
- Cosine相似度：$\cos(x,y) = \frac{\boldsymbol{x}\cdot\boldsymbol{y}}{||\boldsymbol{x}||||\boldsymbol{y}||}$
- 图上的相似性：
- 连边/边权
- 结构相似性：相同邻居数目越多，节点越相似

空间和图

- 图的顶点<->空间上的点
- 图的边权<->点与点之间的相似度（越相似，权重越大）



例如：

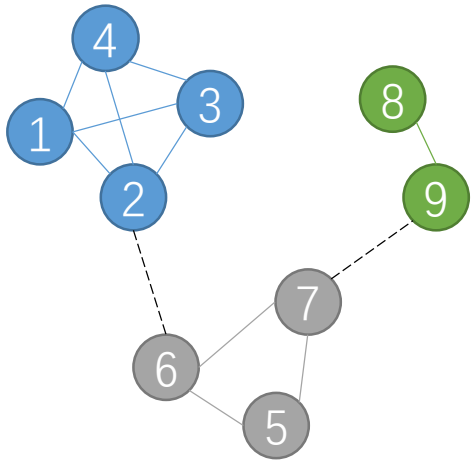
$$w_{ij} = \exp\left(-\frac{\left\|x_i - x_j\right\|^2}{\alpha^2}\right)$$

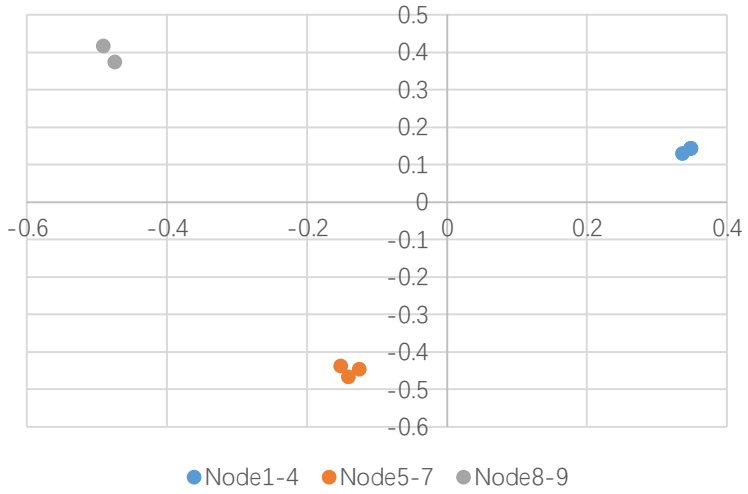
α是超参数。

（高斯分布假设）

图和空间

- 根据图的结构信息（邻接矩阵），将图上的节点表示成向量。在图上相近的节点，在向量空间中距离小。
  - 矩阵分解
  - 谱聚类
  - Network Embedding
- 用K-means等算法对向量聚类。





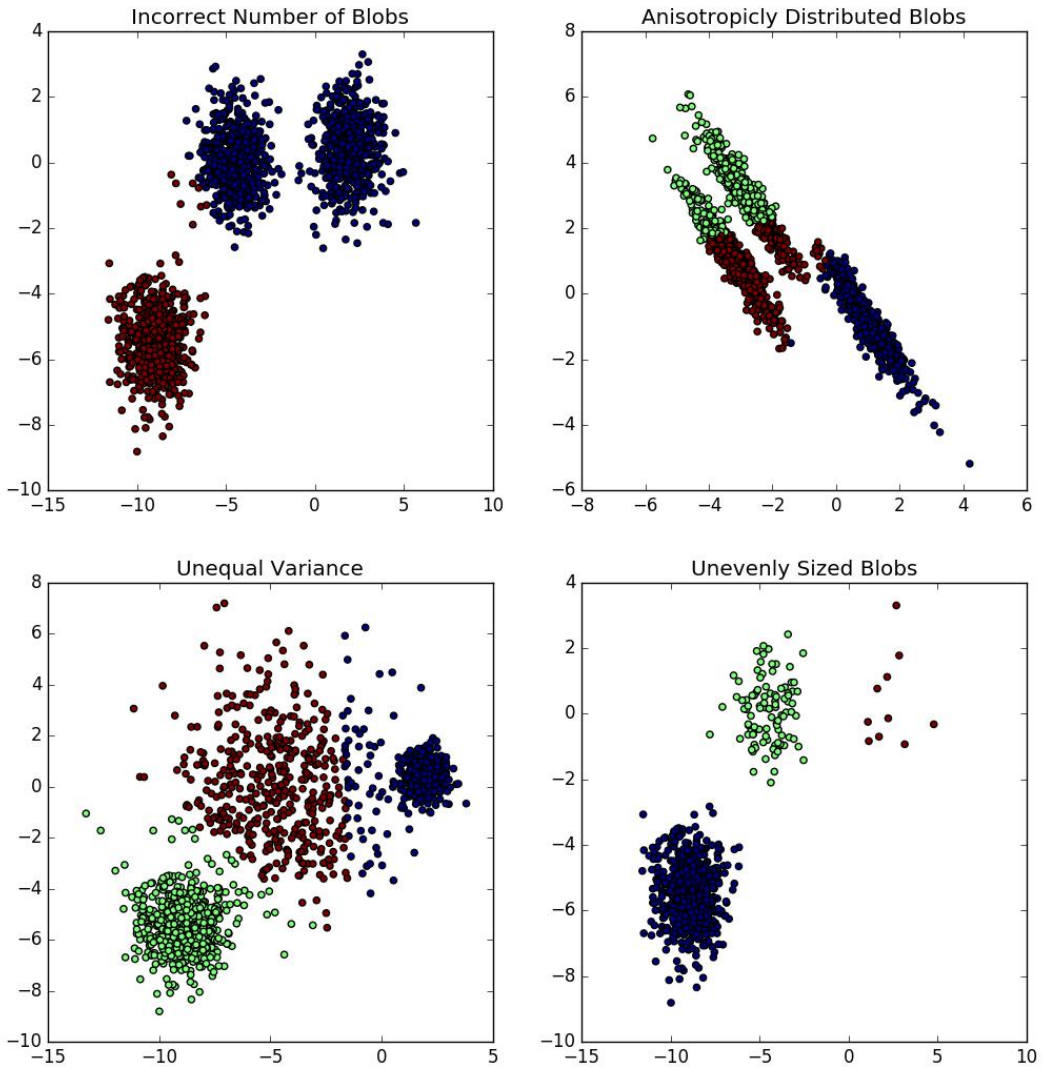
聚类算法

- K-means
- 基于分割的聚类
- 模块度最大化
- 谱聚类
- 层次化聚类
- 标签传播

K-means

- 需要指定聚类的数目k
- 1: 初始时，随机在特征空间中选取k个点作为类中心点
- 2: 根据距离类中心点的距离，将待分类点归到最近的类上。
- 3: 重新计算类中心向量，重复2-3步，直到收敛。

- 缺点：只能处理凸集

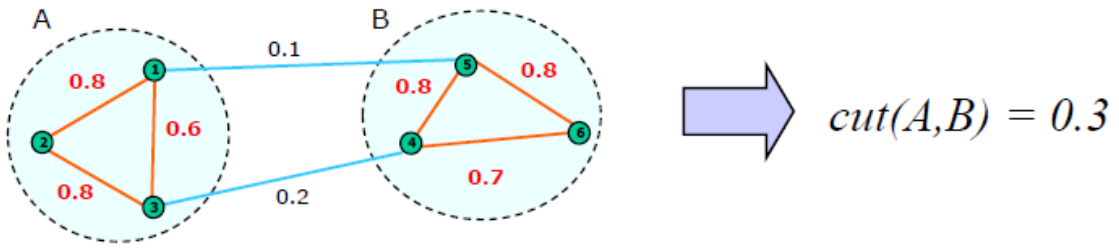


基于分割的聚类

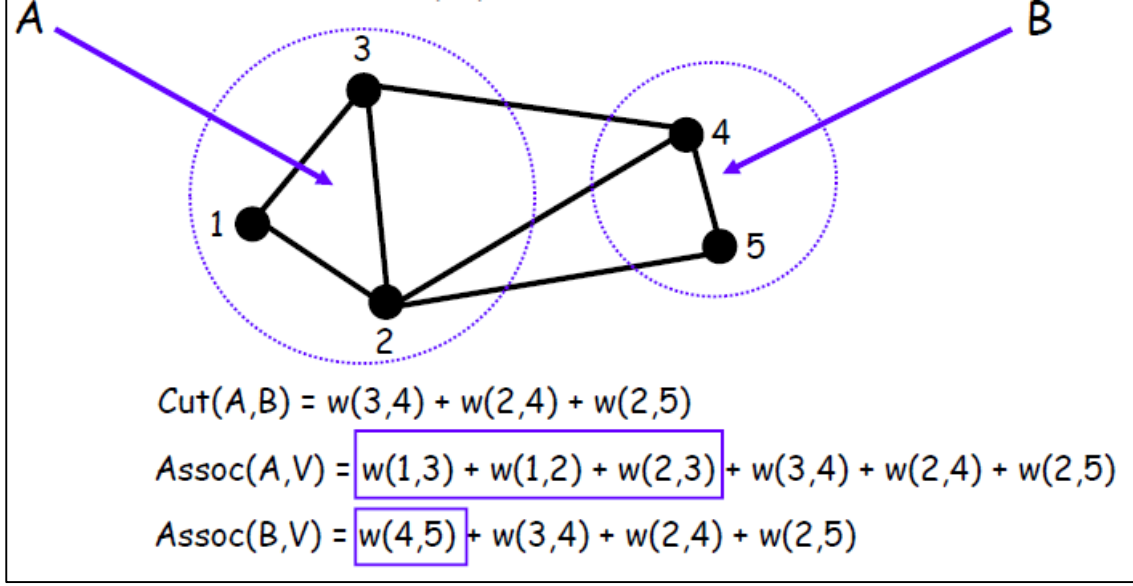
- 用一个**最小**切分，将图分割成两个或者多个子图。
- 切分（cut）：两个子图之间连边的边权和

$$Cut(A,B) = \sum_{u \in A, v \in B} w(u,v)$$

- 倾向于把连接数少的顶点切出来。（极端情况，一个类里一个点也没有，另一个类里有所有的点。Cut = 0）



Cut 改进



- 平衡两个类的大小。
- Ratio cut:

$$Rcut(A,B) = \frac{cut(A,B)}{|A|} + \frac{cut(A,B)}{|B|}$$

- Normalized cut:

$$Ncut(A,B) = \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(A,B)}{assoc(B,V)}$$

- $assoc(A,V) = \sum_{u \in A, v \in V} w(u,v)$
- 大部分图分割问题都是NP-hard的，只能近似求解

The Kernighan-Lin Algorithm

- 指定类的个数和类的大小。例如把所有的节点平均分成两个类。

- KL算法：
  1. 首先，随机把图分成相等的两个部分。
  2. 交换不属于同一类的一对顶点，算出交换后cut size减小的值，选择cut size减小最多（或者增长最少）的一对顶点，交换。
  3. 每个顶点在一轮里只能被交换一次。重复第2步，直到某一类中所有的顶点都被交换过了，回溯交换的过程，取cut size最小时的交换状态，作为这一轮的结果。
  4. 重复2-3步，直到cut size不再减小。
- 如果分类个数大于2，可以重复二分。但不能保证解最优。
- 缺点：依赖初始状态的选取。
- 通常以其他算法的结果作为初始状态，利用KL算法优化调整其他算法得到聚类结果。

谱聚类

- 分成两类，不指定类的大小。
- Cut size: $R = \frac{1}{2}\sum_{ij\ in\ different\ groups} A_{ij}$
- 定义$S_i = \begin{cases} +1, if\ vertex\ i\ belongs\ to\ group1 \\ -1, if\ vertex\ i\ belongs\ to\ group2 \end{cases}$
- $R = \frac{1}{4}\sum_{ij} A_{ij}(1 - s_i s_j) = \frac{1}{4}\sum_{ij}(k_{ij}\delta_{ij} - A_{ij}) = \frac{1}{4}\sum_{ij} L_{ij} s_i s_j$

$$\boldsymbol{R = \frac{1}{4} s^T L s}$$

- $\boldsymbol{L = D - A}$ 称为拉普拉斯（Laplacian）矩阵。
- 放松条件对s的限制（比如要求s的长度为$\sqrt{n}$, $\sum_i s_i = n_1 - n_2$）

$$\boldsymbol{s = \sum_i a_i v_i,\ R = \sum_i a_i^2 \lambda_i}$$

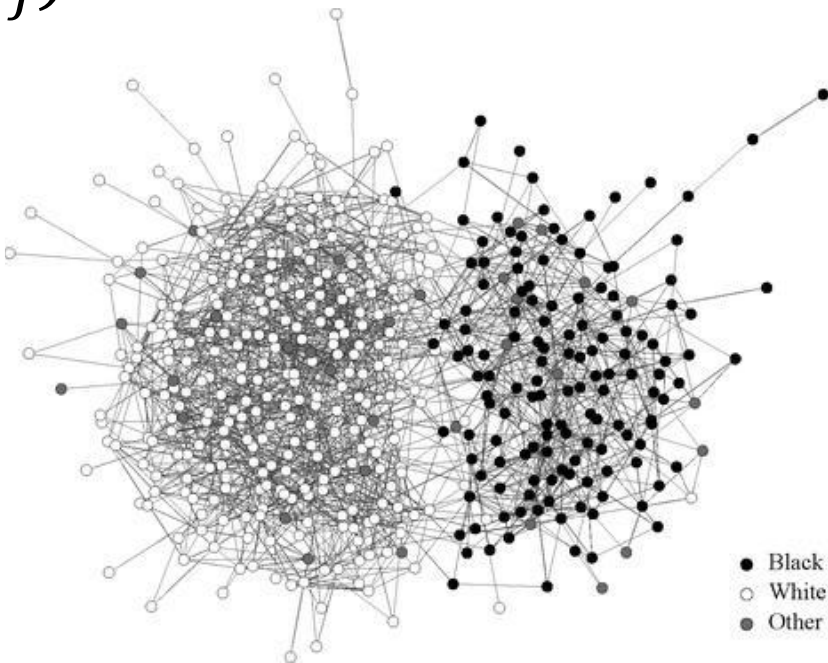
- $\boldsymbol{v_i}$是L的特征向量，对应特征值为λ_i
- $\boldsymbol{s}$取平行于第二小特征向量（ *Fiedler vector* ）
- 取$s_i = +1(-1)$ 如果$v_2^i > 0 (< 0)$

模块度

- 一个评价网络聚类程度的指标
- **模块度（Modularity）**定义为类内节点间实际边权和期望边权之差：

$$Q = \frac{1}{2m}\sum_{ij}\left(A_{ij} - \frac{k_i k_j}{2m}\right)\delta(c_i, c_j)$$

- 模块度越大，说明类间连接越紧密。



模块度

$$Q = \frac{1}{2m}\sum_{ij}\left(A_{ij} - \frac{k_i k_j}{2m}\right)\delta(c_i, c_j)$$

$$Q = \frac{1}{2m}\sum_{c=1}^{n_c}\sum_{i\in c}\sum_{j\in c}(A_{ij} - \frac{k_i k_j}{2m}) = \sum_{c=1}^{n_c}\left[\frac{l_c}{m} - \left(\frac{d_c}{2m}\right)^2\right]$$

- 其中，l_c是类内节点之间的边权和，d_c是类内节点的度数和。
- $Ex(l_c) = d_c^2/4m$是类内边权和的期望值。

模块度

$$
\begin{aligned}
Q_{\max} &= \max_{\mathcal{P}} \left\{ \sum_{c=1}^{n_c} \left[\frac{l_c}{m} - \left(\frac{d_c}{2m} \right)^2 \right] \right\} = \frac{1}{m} \max_{\mathcal{P}} \left\{ \sum_{c=1}^{n_c} \left[l_c - \mathrm{Ex}(l_c) \right] \right\} \\
&= -\frac{1}{m} \min_{\mathcal{P}} \left\{ -\sum_{c=1}^{n_c} \left[l_c - \mathrm{Ex}(l_c) \right] \right\},
\end{aligned}
$$

$$
\begin{aligned}
Q_{\max} &= -\frac{1}{m} \min_{\mathcal{P}} \left\{ \left[\left(m - \sum_{c=1}^{n_c} l_c \right) - \left(m - \sum_{c=1}^{n_c} \mathrm{Ex}(l_c) \right) \right] \right\} \\
&= -\frac{1}{m} \min_{\mathcal{P}} (|\mathrm{Cut}_{\mathcal{P}}| - \mathrm{ExCut}_{\mathcal{P}}).
\end{aligned}
$$

- 最大化模块度等价于最小化类间实际边权和类间边权期望值的差。

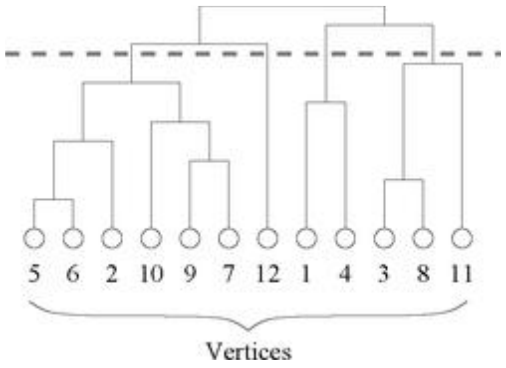
模块度最大化

- 可以和其他聚类方法结合，作为评估并选择聚类结果的指标。
- 也可以将模块度最大化直接作为聚类的目标。（难解问题，只能采用近似解法）

模块度最大化（贪心1）

- 类似KL算法。（二分类任务）
- 首先，对图随机二分类。
- 每一轮：
  - 计算每个顶点被移动到另一个类时模块度的变化，选择模块度增大最多或者减小最少的顶点，移动到另一个分类里。每个顶点在一轮里只移动一次。直到n个顶点都移动过了。这一轮移动停止。回溯移动的过程，找到模块度最大的状态，作为下一轮的初始状态。
- 重复以上过程，直到模块度不再增大。
- 每一轮复杂度$O(mn)$

模块度最大化（贪心2）



- 也是一种层次化聚类方法。
- 初始时，把每个顶点看成一个类。
- 对每两个类，计算合并后的模块度增量，选择模块度增加最大或者减少最小的两个类合并。
- 重复合并过程，直到最后只剩一个类。
- 回溯合并的过程，回到模块度最大的那个状态，就是最后的聚类结果。
- 得到的解相对较差，但是速度快，原始方法$O(n^2)$，优化后$O(n\log^2 n)$。（稀疏图）

模块度最大化（谱方法）

- 同样定义$s_i = \pm 1$代表第i个节点属于第1/2个类。定义B矩阵：

$$B_{ij} = A_{ij} - \frac{k_i k_j}{2m}$$

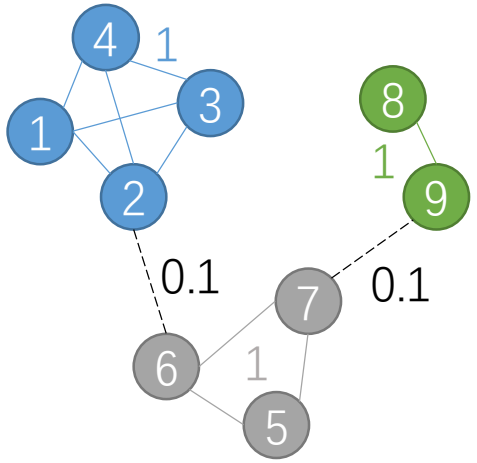
则模块度可以写成：

$$Q = \frac{1}{4m}\sum_{ij}\left(A_{ij} - \frac{k_i k_j}{2m}\right)(s_i s_j + 1) = \frac{1}{4m}\sum_{ij} B_{ij} s_i s_j = \frac{1}{4m} s^T B s$$

- s平行于B矩阵最大特征值λ_1对应的特征向量v_1
- 取$s_i = +1(-1)$ 如果$v_1^i > 0 (< 0)$
- 可以用类似KL的贪心算法在此结果上进一步优化。

谱聚类

- 谱聚类：包括了所有利用**特征值/特征向量**方法聚类的技术，无论是利用邻接矩阵A本身还是与之相关的其它矩阵（例如拉普拉斯矩阵、模块度矩阵等）。
- 将图上的顶点投影到前K个特征向量张成的空间中，在这个空间里做聚类（比如用K-means），可以解决直接用K-means分不出的类。（比如非凸集）

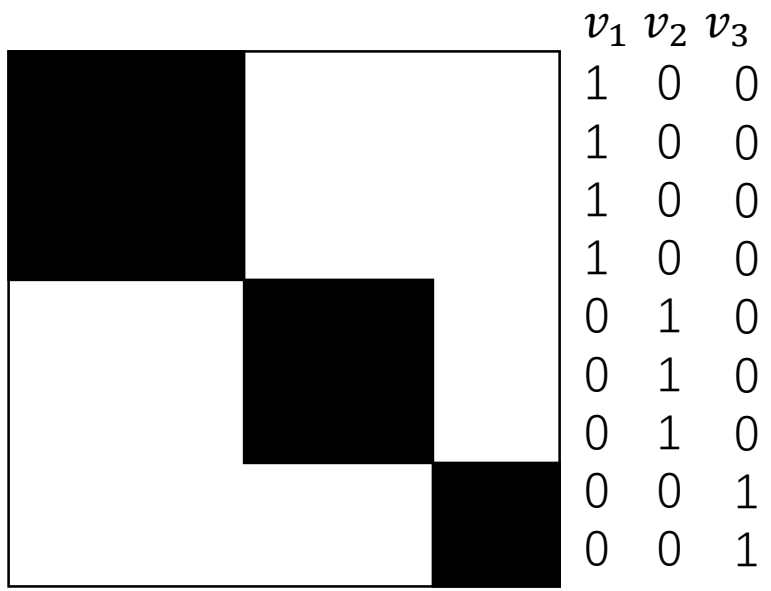


Laplacian 矩阵的特征向量（对应特征值左小右大）：

1	0.3333	0.3482	0.1437	0.0008	0.0146	-0.0226	0.8035	0.145	-0.2822
2	0.3333	0.3366	0.1289	-0.0009	-0.0297	0.0474	0	0	0.8694
3	0.3333	0.3482	0.1437	0.0008	0.0146	-0.0226	-0.2762	-0.7684	-0.2822
4	0.3333	0.3482	0.1437	0.0008	0.0146	-0.0226	-0.5273	0.6234	-0.2822
5	0.3333	-0.1407	-0.4661	-0.0369	0.8049	0.0476	0	0	0.0207
6	0.3333	-0.1251	-0.4462	-0.0335	-0.4582	0.6773	0	0	-0.0658
7	0.3333	-0.1515	-0.438	0.0353	-0.374	-0.7297	0	0	0.0229
8	0.3333	-0.4901	0.4166	-0.6888	-0.0127	-0.0228	0	0	0.0003
9	0.3333	-0.4737	0.3738	0.7224	0.0259	0.0479	0	0	-0.0009

谱聚类

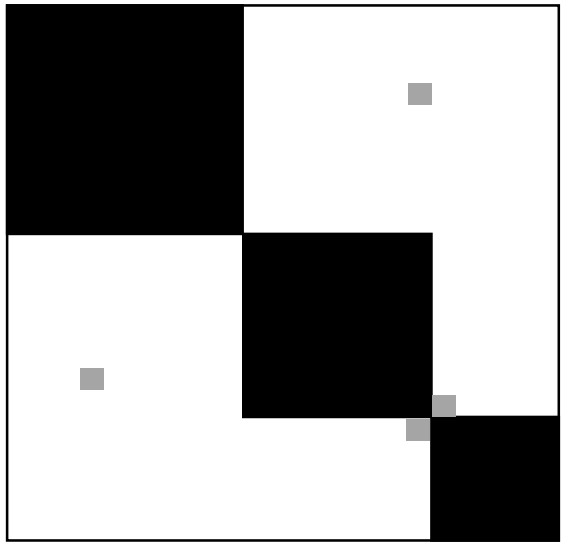
- 以Laplacian矩阵为例：$L = D - A$。Laplacian矩阵的特征值均大于等于0，最小的特征值$\lambda_1 = 0, v_1 = \mathbf{1}$。
- 如果有图上有k个连通分量（L矩阵为块对角形式），L有k个简并的最小特征向量（对应特征值=0）。取这k个特征向量为[11…11,00…00,…,00…00; 00…00,11…11,…,00…00;00…00,…,11…11]
- 将这k个特征向量拼成一个$n \times k$的矩阵，它的每一行对应图上节点经过变换后的向量。属于同一连通分量的节点向量是一样的。



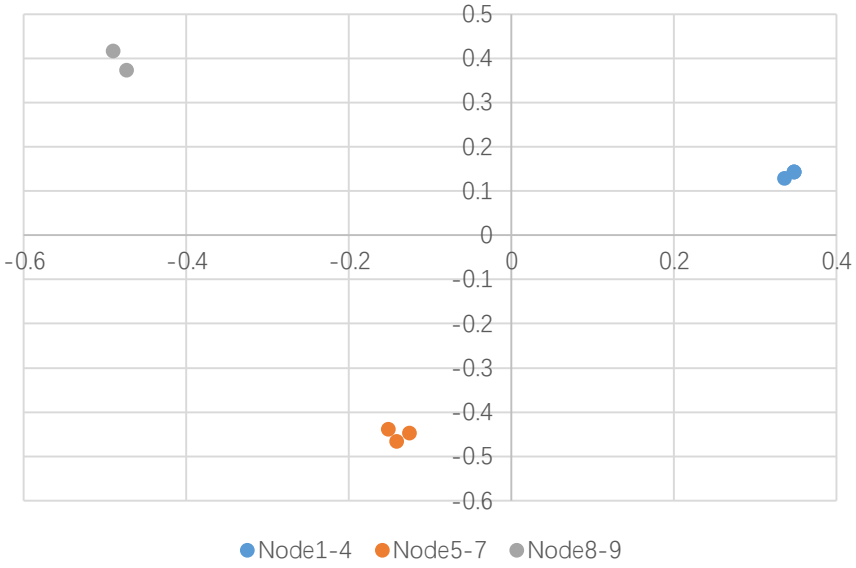
v_1	v_2	v_3
1	0	0
1	0	0
1	0	0
1	0	0
0	1	0
0	1	0
0	1	0
0	0	1
0	0	1

谱聚类

- 如果Laplacian矩阵并不是严格块对角的形式。在连通分量之间添加少量边作为扰动。最小特征值对应的k个特征向量解除简并。
- 但是前k-1个特征值接近0，同样将节点投影到前k个特征向量的空间中，属于同一个类的点在空间中接近。

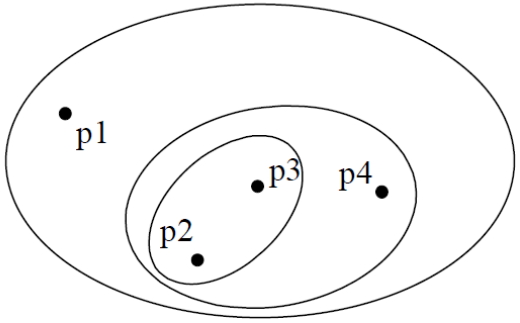


v_1	v_2	v_3
0.3333	0.3482	0.1437
0.3333	0.3366	0.1289
0.3333	0.3482	0.1437
0.3333	0.3482	0.1437
0.3333	-0.1407	-0.4661
0.3333	-0.1251	-0.4462
0.3333	-0.1515	-0.438
0.3333	-0.4901	0.4166
0.3333	-0.4737	0.3738

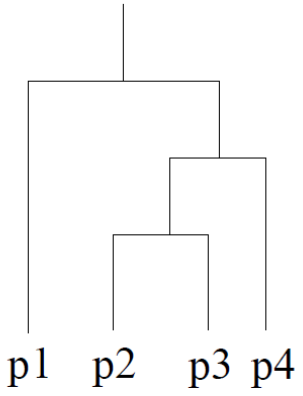


特征值：0, 0.0334, 0.1027, 2.0487, 3.0339, 3.1, 4, 4, 4.0812

层次化聚类



Hierarchical Clustering



Dendrogram

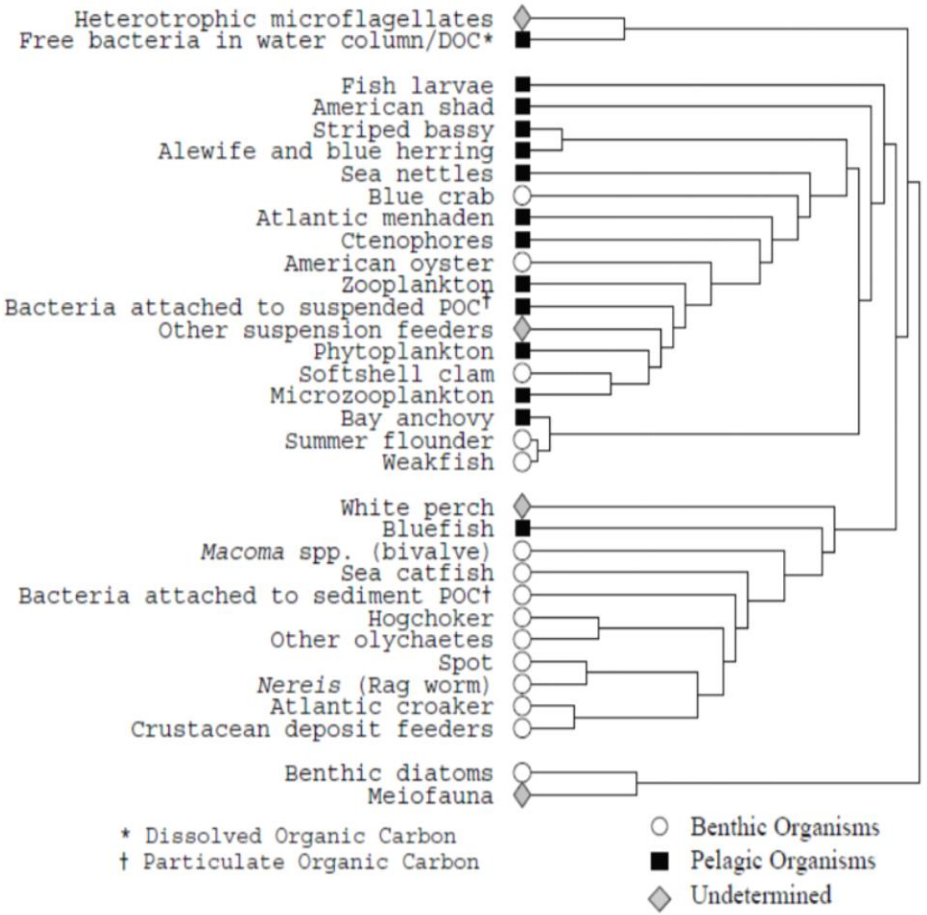
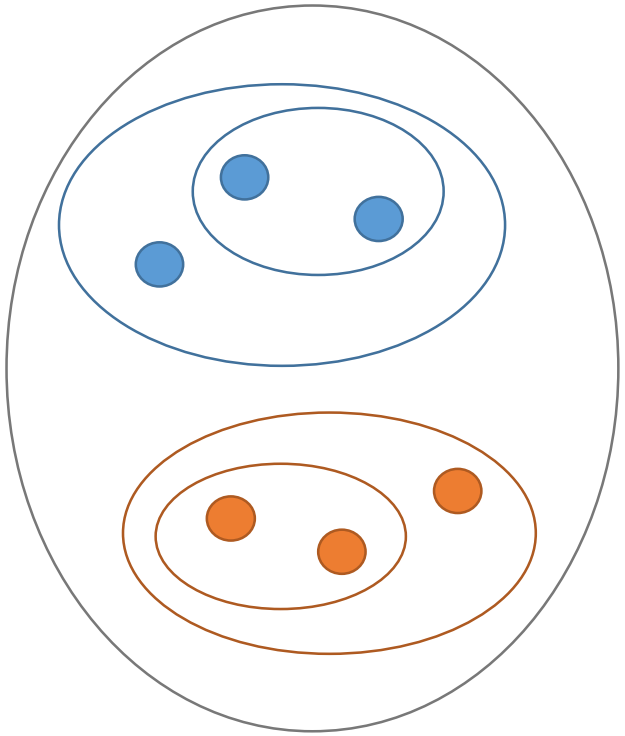
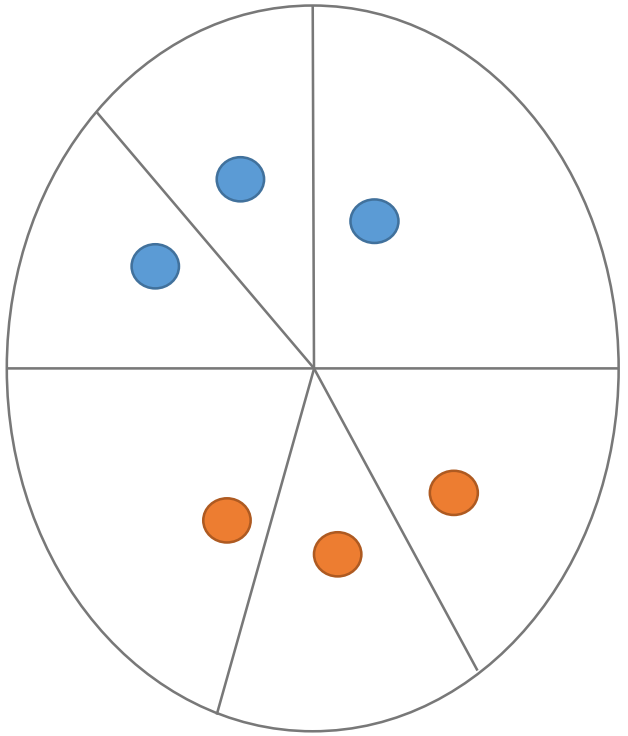


FIG. 7: Hierarchical tree for the Chesapeake Bay food web described in the text.

层次化聚类

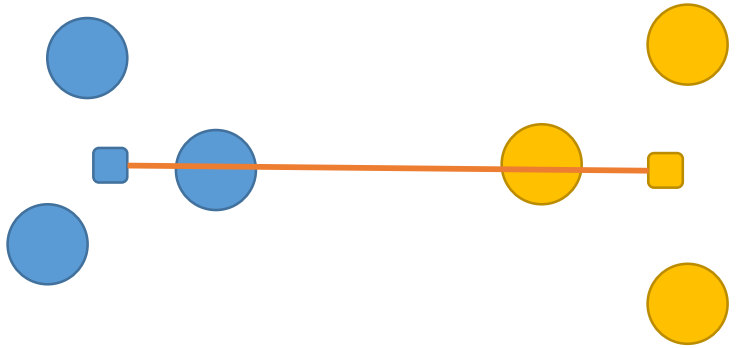
- 聚合型算法（自底向上）
- 分割型算法（自顶向下）



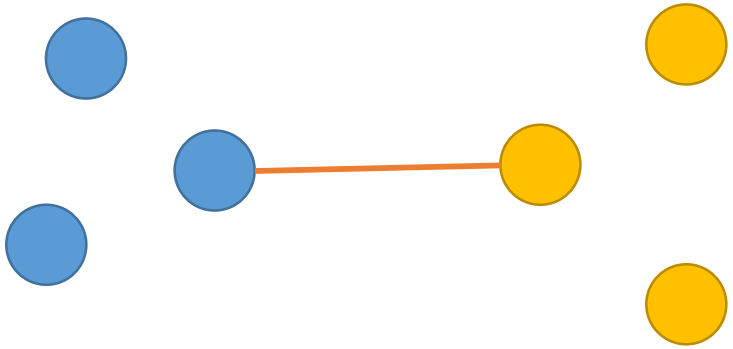


层次化聚类

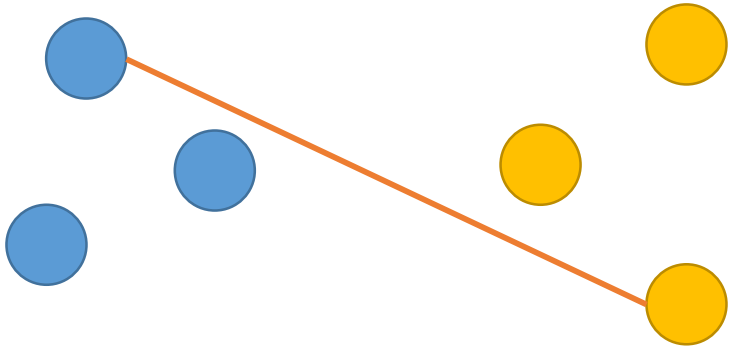
- 类与类之间相似度的度量：
- Single-linkage: 距离最小的
- Complete-linkage: 距离最大的
- Average-linkage: 平均距离
- Distance Between Centroids: 类重心距离



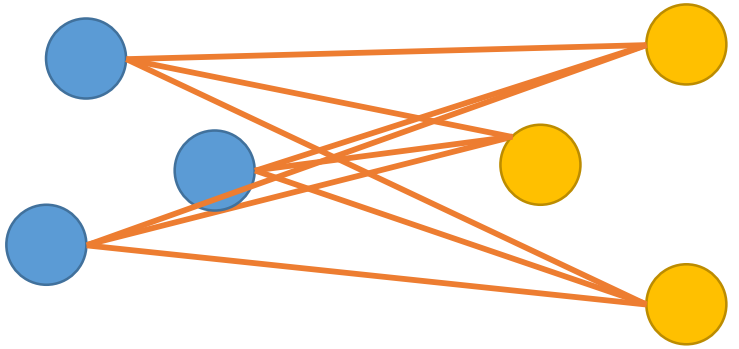
Centroid



Single



Complete



Average

层次化聚类（自底向上）

	1	2	3	4	5
1	1.00	**0.90**	0.10	0.65	0.20
2		1.00	0.70	0.40	0.30
3			1.00	0.30	0.20
4				1.00	0.80
5					1.00

① ② ③ ④ ⑤

层次化聚类（自底向上）

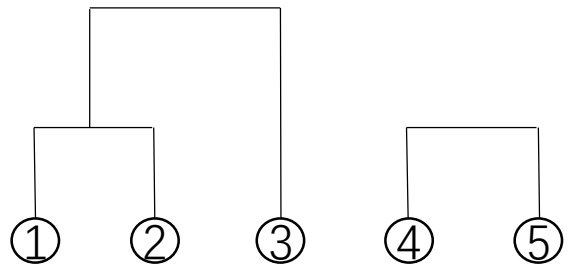
	-	1&2	3	4	5
-	-	-	-	-	-
1&2		1.00	0.70	0.65	0.30
3			1.00	0.30	0.20
4				1.00	**0.80**
5					1.00

层次化聚类（自底向上）

	-	1&2	3	4&5	-
-	-	-	-	-	-
1&2		1.00	**0.70**	0.65	-
3			1.00	0.30	-
4&5				1.00	-
-					-

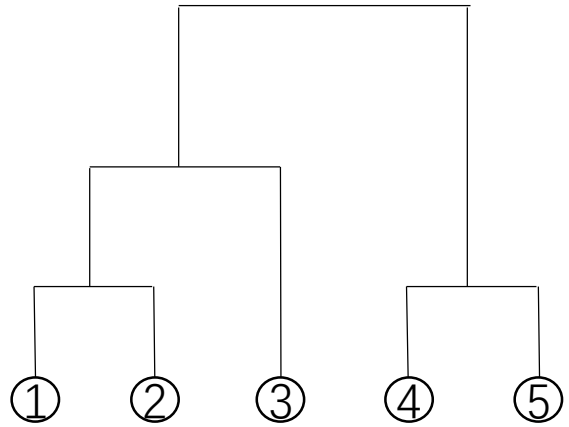
层次化聚类（自底向上）

	-	-	(1&2)&3	4&5	-
-	-	-	-	-	-
-		-	-	-	-
(1&2)&3			1.00	**0.65**	-
4&5				1.00	-
-					-



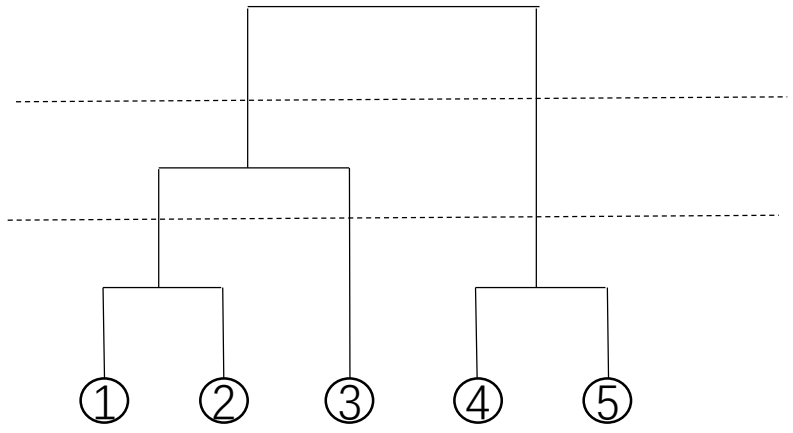
层次化聚类（自底向上）

	-	-	-	((1&2)&3）&（4&5）	-
-	-	-	-	-	-
-		-	-	-	-
			-	-	-
((1&2)&3）&（4&5）				1.00	-
-					-



层次化聚类（自底向上）

- 在聚类过程中停止，可以得到flat cluster
- 停止的条件：
  - 类的个数
  - 相似度小于某一阈值
  - 或者最优化其他指标（比如模块度等）



Dendrograms

基于模块度的聚合方法

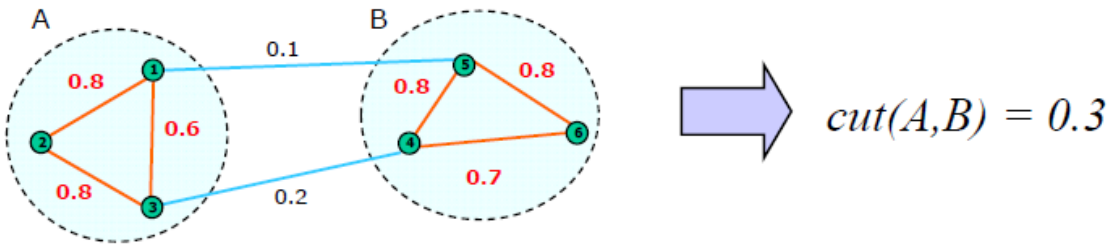
- Fast Unfolding
- Step1: 首先每个节点自成一类。将每个节点移动到相邻的类里，计算移动的模块度增量ΔQ，将增量最大的节点移动到相应的类里。重复以上过程，直到没有移动可以使得模块度增大。
- Step2: 将上一步得到的类看成节点，计算类与类之间连边的权重为相应的两个类之间各节点连边的权重之和。
- 重复Step2-3，得到层次化的聚类结构。

层次化聚类（自顶向下）

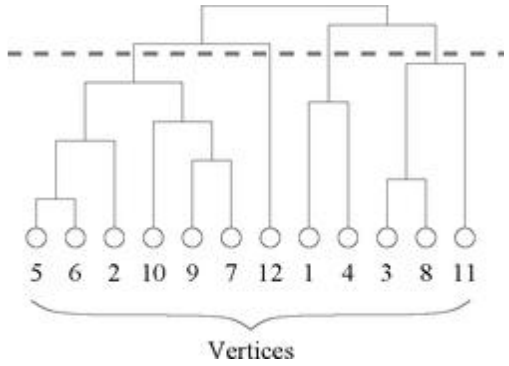
- 一个切分（cut）：两个子图之间的边的边权和

$$Cut(A,B) = \sum_{u \in A, v \in B} w(u,v)$$

- 目的：每次找到最小切分。
- 需要枚举所有可能的切分，难解问题。
- 只能用近似解法。



基于betweenness的分割方法



- 边的Betweenness：经过某条边的最短路径的数目
- Betweenness高的边，起到“枢纽”的作用。
- 算法：
- 计算每一条边的betweeness，选择最大的一条边，从图中删去。重新计算整张图的betweeness中心度。直到没有边可以删除。
- 层次化的结构。
- 是一个比较慢的算法。（求最短路径的代价比较高）

标签传播

- 无监督/半监督的分类方法。
- 连接到同类邻居的边>连接到不同类邻居的边
- 每个节点所属的类由邻居节点所属的类决定。

- 无监督的标签传播：不需要指定类的个数和类的大小。
- 半监督的标签传播：可以利用少量已经标注了类标签的数据点。聚类的数目和标注的类的数目保持一致。

- 优点：时间复杂度$O(knd)$（d是平均度数，k是迭代次数）。近乎线性时间。

无监督标签传播

1、初始化类标签：给每个节点都赋予一个不同的类。

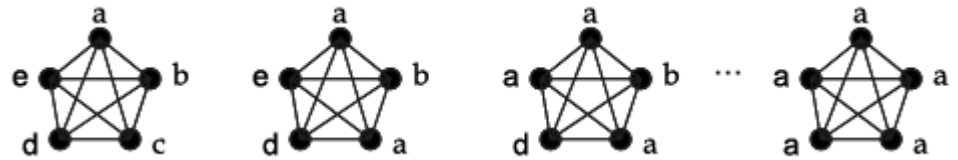
2、更新每个节点的类标签：

$$z_i = argmax_z \sum_{j \neq i} W_{ij} \delta_{z_j z}$$

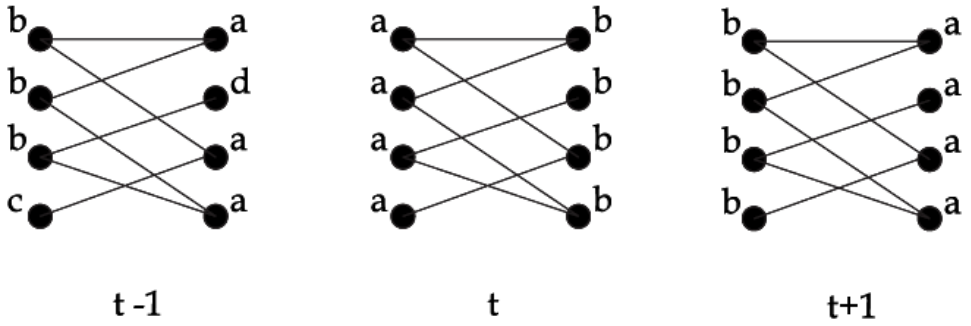
如果有不止一个类的贡献都最大，随机选择一个作为当前节点的类。

3、当每个节点的标签和它邻居中最多的标签相等，停止。否则重复第2步。

- 大部分的类会消失，紧密联系的子图有相同的类。



无监督标签传播



- 上述的标签传播算法的两个缺陷：
  1. 在类似二部图的结构中，会发生标签的振荡而无法收敛。
  2. 可能出现某个非常庞大的社区。

- 同步更新(synchronous)：用t时刻的标签更新t+1时刻的标签。
- 异步更新(asynchronous)：用已经更新的t+1时刻的标签和未更新的t时刻标签，更新t+1时刻当前节点的标签。

- 异步更新收敛更快，且更不容易陷入振荡。同步更新结果更稳定。

无监督标签传播

- 带有节点偏好和hop衰减的标签传播：

$$\mathcal{L}_i' = \underset{\mathcal{L}}{\operatorname{argmax}} \sum_{i' \in \mathcal{N}_i} s_{i'}(\mathcal{L}_{i'}) f(i')^m w_{i',i},$$

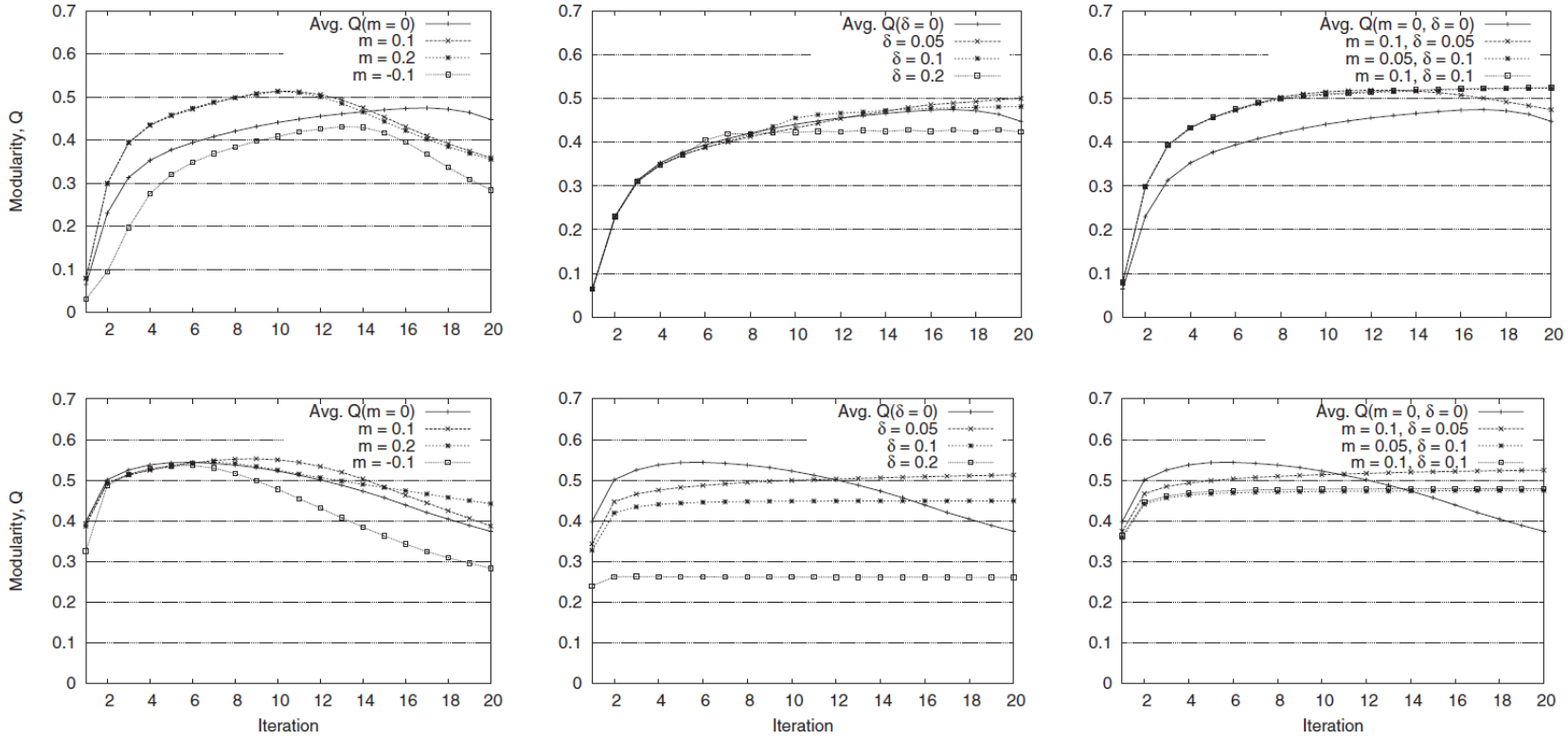
- $s_i(\mathcal{L})$是标签$\mathcal{L}$ 在节点i 处的hop score，$f(i)$是任意可比的节点i的特征（节点偏好）。例如，可以取$f(i) = Deg(i)$。参数m控制特征对于标签传播的影响。
- 标签的hop score会随着标签的传播衰减：

$$s_i'(\mathcal{L}_i') = \left[\max_{i' \in \mathcal{N}_i(\mathcal{L}_i')} s_i(\mathcal{L}_{i'}) \right] - \delta,$$

- $\mathcal{N}_i(\mathcal{L})$是i的标签为$\mathcal{L}$的邻居。$\delta$控制标签的hop score的衰减速度。当被选择的标签等于现在的标签时，$\delta = 0$。（避免负反馈循环）

无监督标签传播

- 节点偏好度的引入能够加快收敛、提升最终结果的模块度。
- 跳衰减的引入能够避免出现庞大社区，避免出现传播一定步数后模块度的下降。但是衰减太快社区也无法正常增长，限制模块度的升高。



半监督标签传播

- 假设某个点的标签能够以一定概率传播给与它相连的点。
- 概率转移矩阵

$$T_{ij} = P(i \to j) = \frac{w_{ij}}{\sum_k w_{ik}}$$

- t时刻标签矩阵：$Y^t \in N^{\mathrm{n}\times\mathrm{c}}$

$$Y_{ij}^0 = \begin{cases} 1, & x_i \in c_j \\ 0, otherwise \end{cases}$$ 对L个标注数据

对未标注数据，Y_{ij}^0可以取任意值。

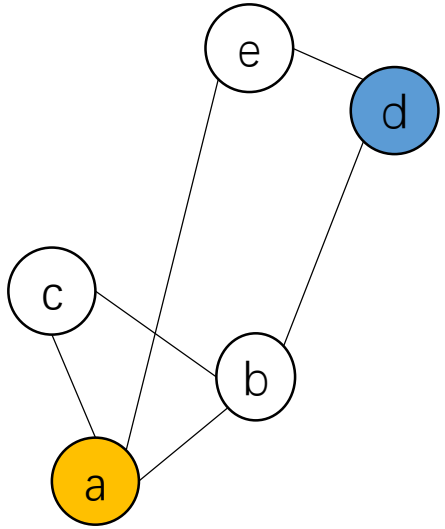
- 标签的传播：$Y_{ij}^{t+1} = \sum_k T_{ki} Y_{kj}^t$

标签传播

- 1: 按如上所述方式计算概率转移矩阵T。t=0时，按如上所述初始化标签矩阵Y。
- 2: 标签传播：$Y^{t+1} = T^T Y^t$
- 3: 对于已标注数据，恢复它们的标注标签。对于未标注数据，对行归一化。
- 4: 重复2-3步，直到Y^t收敛。
- 5: 最终节点x_i的标签$c_j = argmax_j Y_{ij}$

标签传播

- 已标数据：[a: 1, d: 0]
- 未标数据：[c: ?, b: ?, e: ?]
- 标签转移矩阵T：



初始化Y^0

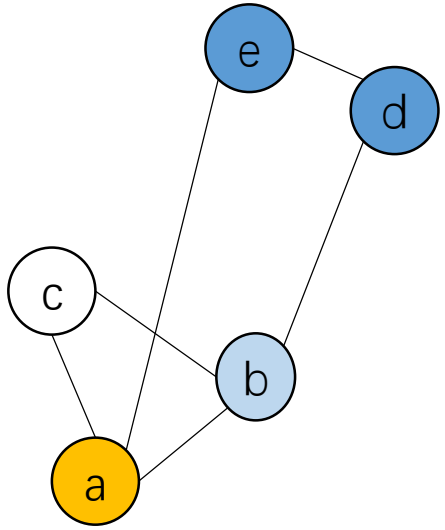
	a	b	c	d	e
a	0.00	0.50	0.40	0.00	0.10
b	0.50	0.00	0.40	0.10	0.00
c	0.50	0.50	0.00	0.00	0.00
d	0.00	0.20	0.00	0.00	0.80
e	0.20	0.00	0.00	0.80	0.00

	0	**1**
a	0	1
b	1	0
c	1	0
d	1	0
e	1	0

标签传播

- 1轮传播：
- 已标数据：[a: 1, d: 0]
- 未标数据：[c: ?, b: ?, e: ?]
- 标签转移矩阵T：

	a	b	c	d	e
a	0.00	0.50	0.40	0.00	0.10
b	0.50	0.00	0.40	0.10	0.00
c	0.50	0.50	0.00	0.00	0.00
d	0.00	0.20	0.00	0.00	0.80
e	0.20	0.00	0.00	0.80	0.00



Y^0

	0	**1**
a	0	1
b	1	0
c	1	0
d	1	0
e	1	0

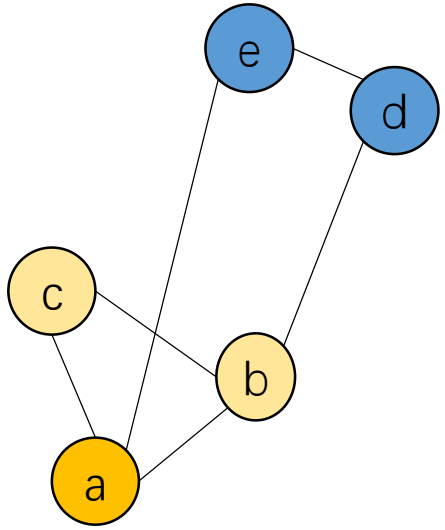
Y^1

	0	**1**
a	0	1
b	0.58	0.42
c	0.5	0.5
d	1	0
e	0.89	0.11

标签传播

- 2轮传播：
- 已标数据：[a: 1, d: 0]
- 未标数据：[c: ?, b: ?, e: ?]
- 标签转移矩阵T：

	a	b	c	d	e
a	0.00	0.50	0.40	0.00	0.10
b	0.50	0.00	0.40	0.10	0.00
c	0.50	0.50	0.00	0.00	0.00
d	0.00	0.20	0.00	0.00	0.80
e	0.20	0.00	0.00	0.80	0.00



Y^1

	0	**1**
a	0	1
b	0.58	0.42
c	0.5	0.5
d	1	0
e	0.89	0.11

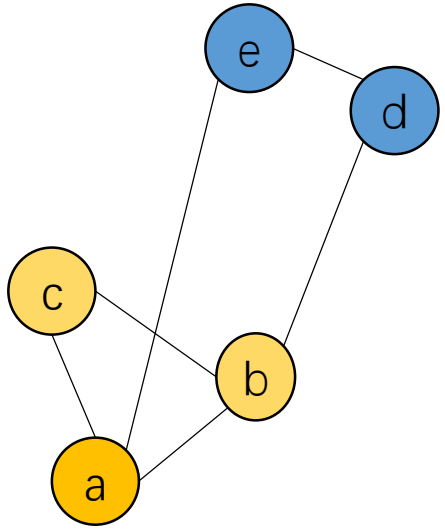
Y^2

	0	**1**
a	0	1
b	0.38	0.62
c	0.29	0.71
d	1	0
e	0.89	0.11

标签传播

- 3轮传播：
- 已标数据：[a: 1, d: 0]
- 未标数据：[c: ?, b: ?, e: ?]
- 标签转移矩阵T：

	a	b	c	d	e
a	0.00	0.50	0.40	0.00	0.10
b	0.50	0.00	0.40	0.10	0.00
c	0.50	0.50	0.00	0.00	0.00
d	0.00	0.20	0.00	0.00	0.80
e	0.20	0.00	0.00	0.80	0.00



Y^2

	0	**1**
a	0	1
b	0.38	0.62
c	0.29	0.71
d	1	0
e	0.89	0.11

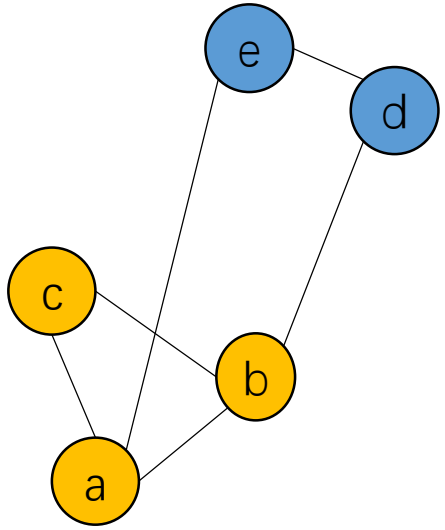
Y^3

	0	**1**
a	0	1
b	0.29	0.71
c	0.19	0.81
d	1	0
e	0.89	0.11

标签传播

- 4轮传播：
- 已标数据：[a: 1, d: 0]
- 未标数据：[c: ?, b: ?, e: ?]
- 标签转移矩阵T：

	a	b	c	d	e
a	0.00	0.50	0.40	0.00	0.10
b	0.50	0.00	0.40	0.10	0.00
c	0.50	0.50	0.00	0.00	0.00
d	0.00	0.20	0.00	0.00	0.80
e	0.20	0.00	0.00	0.80	0.00



Y^3

	0	**1**
a	0	1
b	0.29	0.71
c	0.19	0.81
d	1	0
e	0.89	0.11

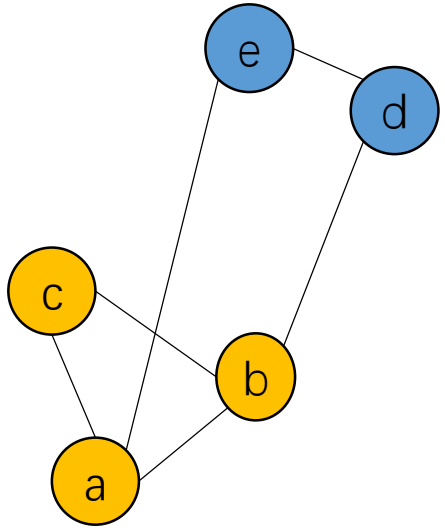
Y^4

	0	**1**
a	0	1
b	0.24	0.76
c	0.14	0.86
d	1	0
e	0.89	0.11

标签传播

- 5轮传播：
- 已标数据：[a: 1, d: 0]
- 未标数据：[c: ?, b: ?, e: ?]
- 标签转移矩阵T：

	a	b	c	d	e
a	0.00	0.50	0.40	0.00	0.10
b	0.50	0.00	0.40	0.10	0.00
c	0.50	0.50	0.00	0.00	0.00
d	0.00	0.20	0.00	0.00	0.80
e	0.20	0.00	0.00	0.80	0.00



Y^4

	0	**1**
a	0	1
b	0.24	0.76
c	0.14	0.86
d	1	0
e	0.89	0.11

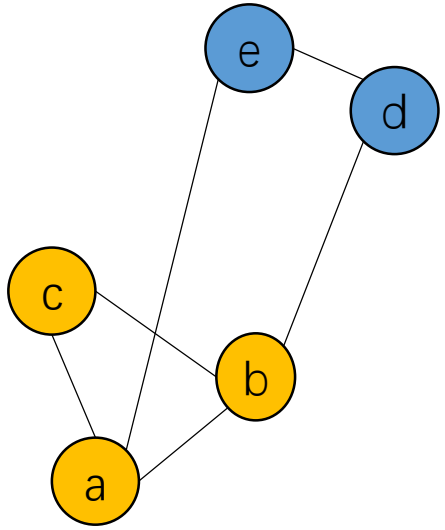
Y^5

	0	**1**
a	0	1
b	0.23	0.77
c	0.12	0.88
d	1	0
e	0.89	0.11

标签传播

- 6轮传播：
- 已标数据：[a: 1, d: 0]
- 未标数据：[c: ?, b: ?, e: ?]
- 标签转移矩阵T：

	a	b	c	d	e
a	0.00	0.50	0.40	0.00	0.10
b	0.50	0.00	0.40	0.10	0.00
c	0.50	0.50	0.00	0.00	0.00
d	0.00	0.20	0.00	0.00	0.80
e	0.20	0.00	0.00	0.80	0.00



Y^5

	0	**1**
a	0	1
b	0.23	0.77
c	0.12	0.88
d	1	0
e	0.89	0.11

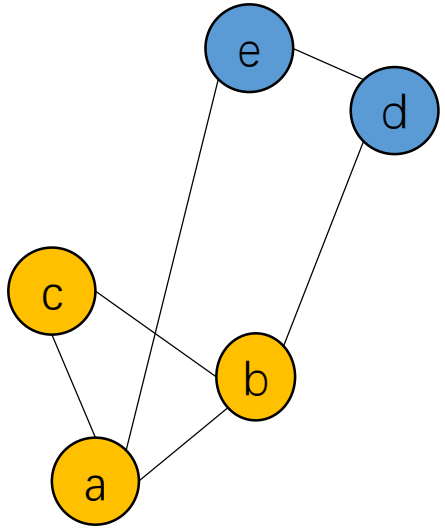
Y^6

	0	**1**
a	0	1
b	0.22	0.78
c	0.11	0.89
d	1	0
e	0.89	0.11

标签传播

- 7轮传播：
- 已标数据：[a: 1, d: 0]
- 未标数据：[c: ?, b: ?, e: ?]
- 标签转移矩阵T：

	a	b	c	d	e
a	0.00	0.50	0.40	0.00	0.10
b	0.50	0.00	0.40	0.10	0.00
c	0.50	0.50	0.00	0.00	0.00
d	0.00	0.20	0.00	0.00	0.80
e	0.20	0.00	0.00	0.80	0.00



Y^6

	0	**1**
a	0	1
b	0.22	0.78
c	0.11	0.89
d	1	0
e	0.89	0.11

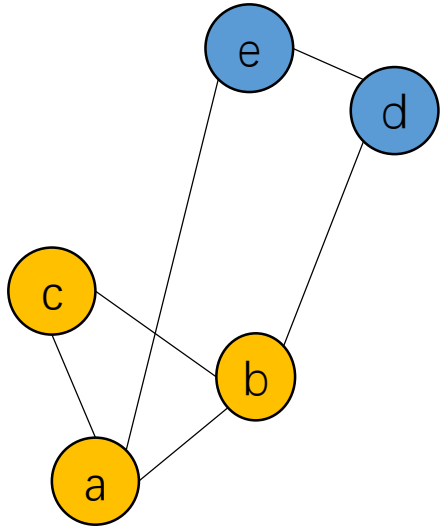
Y^7

	0	**1**
a	0	1
b	0.21	0.79
c	0.11	0.89
d	1	0
e	0.89	0.11

标签传播

- 8轮传播：
- 已标数据：[a: 1, d: 0]
- 未标数据：[c: 1, b: 1, e: 0]
- 标签转移矩阵T：

	a	b	c	d	e
a	0.00	0.50	0.40	0.00	0.10
b	0.50	0.00	0.40	0.10	0.00
c	0.50	0.50	0.00	0.00	0.00
d	0.00	0.20	0.00	0.00	0.80
e	0.20	0.00	0.00	0.80	0.00



Y^7

	0	**1**
a	0	1
b	0.21	0.79
c	0.11	0.89
d	1	0
e	0.89	0.11

Y^8

	0	**1**
a	0	1
b	0.21	0.79
c	0.11	0.89
d	1	0
e	0.89	0.11

扩展知识：异构信息网络

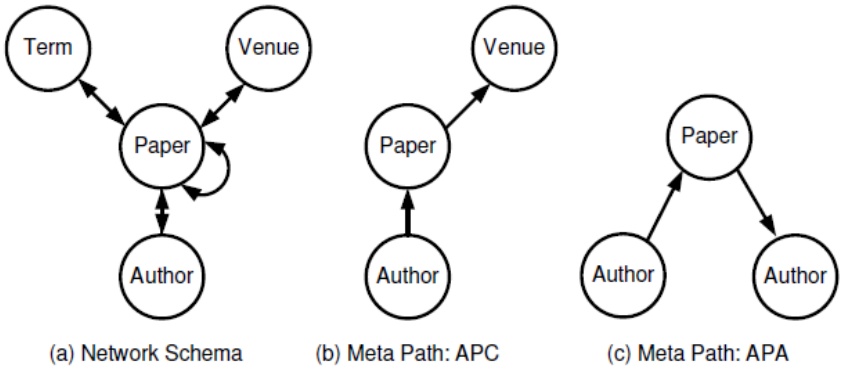
- 异构网络：网络中有不同类型的节点，和不同类型的边。
- 例如，在引用关系网络里，有四种类型的节点：作者（A）、文章（P）、术语（T）、期刊（C）
- 关系：

	Author	Paper	Term	Venues(i.e., conference / journals)
Author	-	Writing	-	-
Paper	Written-by	Citing / cited by	Using	Published in
Term	-	Used	-	-
Venues(i.e., conference / journals)	-	Publishing	-	-

扩展知识：异构信息网络

- **异构信息网络的定义：**
- $G = (V, E)$, 对象类型映射$\phi: V \to A$, 边类型映射$\psi: E \to R$. 对于对象集合中的每一个节点$v \in V$, 都有一个对象类型与之对应$\phi(v) \in A$, 同样地，对于边集里的每一条边$e \in E$, 都有一个边类型与之对应$\psi(e) \in R$。
- **Network schema:**
- 是异构网络$G = (V, E)$的一个元模板，同样是一个有向图，其顶点是G的对象类型，边是G中的边类型，表示成$T_G = (A, R)$。
- **Meta path**：定义在network schema上的路径。
- **Meta graph**：定义在network schema上的有向无环图，

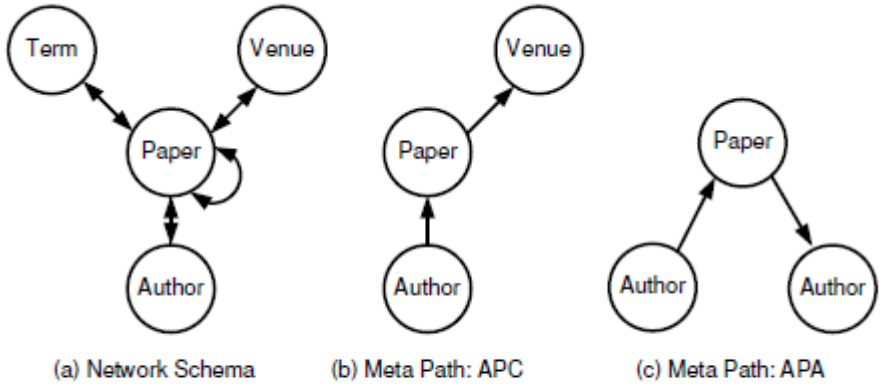
且有唯一的源点和汇点。



(a) Network Schema (b) Meta Path: APC (c) Meta Path: APA

扩展知识：异构信息网络

- 基于meta-path的节点相似度计算(Sun, 2011)：

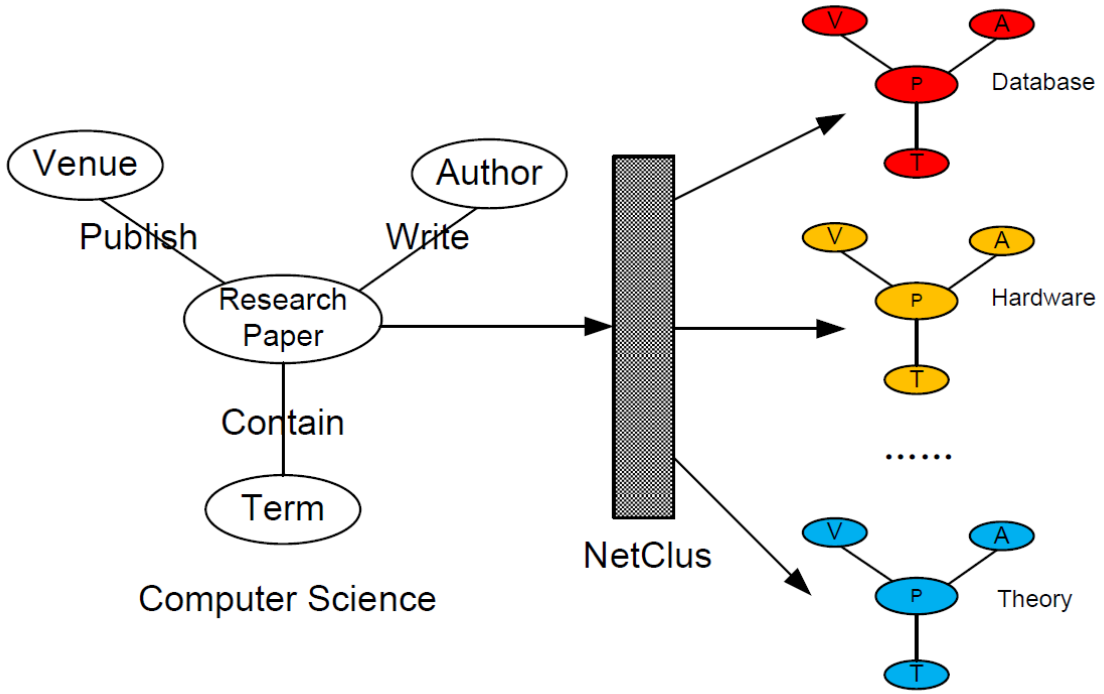


$$s(x,y) = \frac{2 \times |\{p_{x \rightsquigarrow y} : p_{x \rightsquigarrow y} \in \mathcal{P}\}|}{|\{p_{x \rightsquigarrow x} : p_{x \rightsquigarrow x} \in \mathcal{P}\}| + |\{p_{y \rightsquigarrow y} : p_{y \rightsquigarrow y} \in \mathcal{P}\}|}$$

Figure 1: Bibliographic network schema and meta paths.

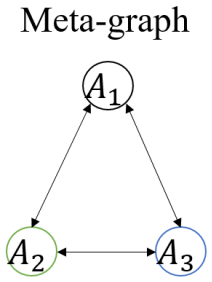
扩展知识：异构信息网络

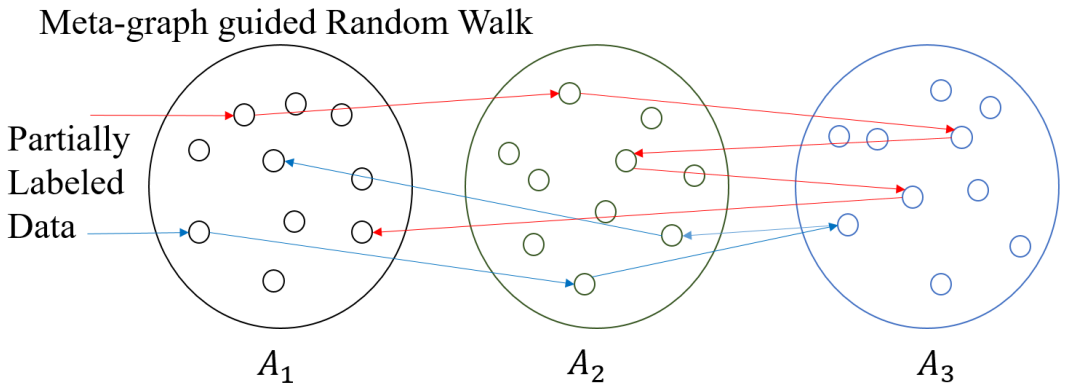
- 基于Ranking的聚类算法：NetClus (Sun, 2009)

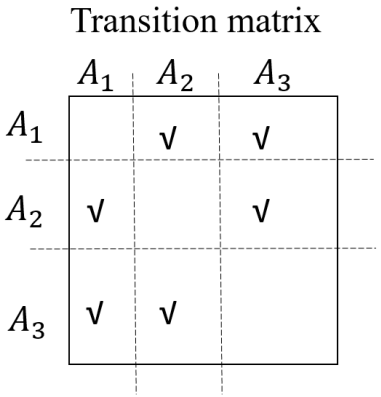


扩展知识：异构信息网络

- 基于meta-graph的半监督分类方法（Jiang, 2017）

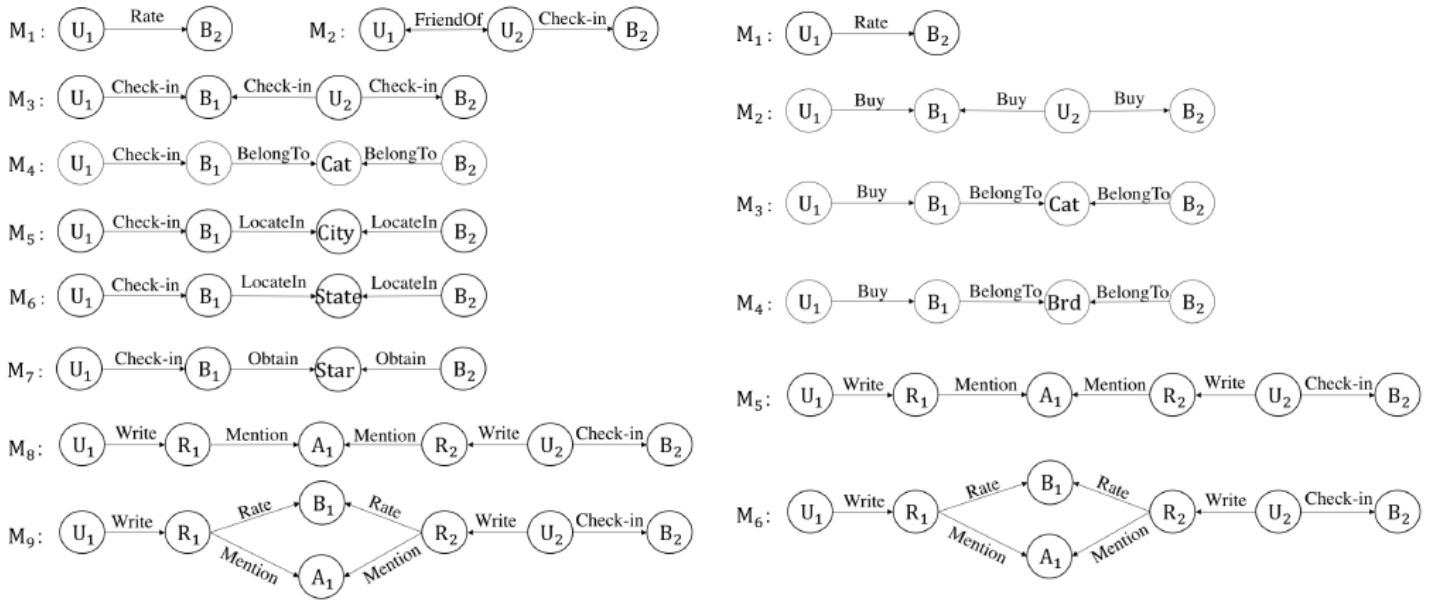






扩展知识：异构信息网络

- 基于meta-graph的推荐算法 (Zhao, 2017)



(a) Yelp-200K (Star: the average stars a business obtained).

(b) Amazon-200K (Brd: brand of the item).

Figure 3: Meta-graphs used for Amazon and Yelp datasets.

参考文献

- **综述类：**
  - Newman M. Networks: an introduction[M]. Oxford university press, 2010.
  - Fortunato S. Community detection in graphs[J]. Physics reports, 2010, 486(3): 75-174.
  - Schaeffer S E. Graph clustering[J]. Computer science review, 2007, 1(1): 27-64.
- **标签传播：**
  - Raghavan U N, Albert R, Kumara S. Near linear time algorithm to detect community structures in large-scale networks[J]. Physical review E, 2007, 76(3): 036106.
  - Leung I X Y, Hui P, Lio P, et al. Towards real-time community detection in large networks[J]. Physical Review E, 2009, 79(6): 066107.

参考文献

- **异构信息网络：**
  - Han J, Sun Y, Yan X, et al. Mining knowledge from data: An information network analysis approach[C]//Data Engineering (ICDE), 2012 IEEE 28th International Conference on. IEEE, 2012: 1214-1217.
  - Sun Y, Han J, Yan X, et al. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks[J]. Proceedings of the VLDB Endowment, 2011, 4(11): 992-1003.
  - Sun Y, Yu Y, Han J. Ranking-based clustering of heterogeneous information networks with star network schema[C]//Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2009: 797-806.
  - Jiang H, Song Y, Wang C, et al. Semi-supervised Learning over Heterogeneous Information Networks by Ensemble of Meta-graph Guided Random Walks[C]. IJCAI, 2017.
  - Zhao H, Yao Q, Li J, et al. Meta-Graph Based Recommendation Fusion over Heterogeneous Information Networks[C]//Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2017: 635-644.

Thanks