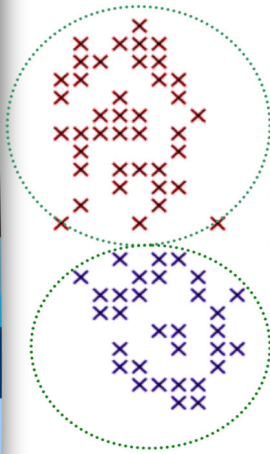


Clustering

■ Clustering – finding natural groupings of items.

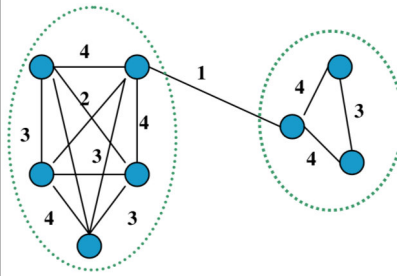
■ Vector Clustering



Each point has a vector, i.e.

- x coordinate
- y coordinate
- color

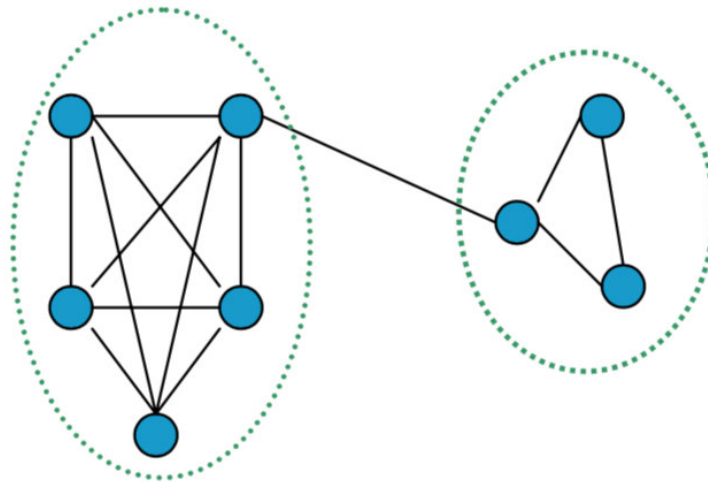
Graph Clustering



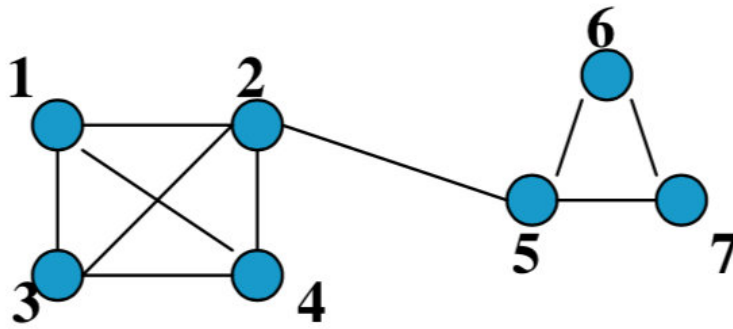
Each vertex is connected to others by (weighted or unweighted) edges.

- 基于vector聚类
- 基于graph聚类

1. random walks



针对一个图像，在一个cluster内可能存在很多个连线(联系)，同时不同的cluster之间连线很少；这意味着，从一个node开始出发，在相互连接的node间随机移动，便更容易在一个cluster内移动而不是在不同的cluster之间移动。这就是MCL聚类基础。通过随机移动，能发现倾向于聚集在一起的趋势，该趋势涉及的范围就是一个cluster。使用"Markov Chains"计算图中的随机移动。



例如，从节点1移动到2,3,4均具有33%的概率，而移动到5,6,7的概率为0；节点2移动到1,3,4,5均有25%的概率，而到6,7概率为0。对此，构建矩阵：

$$\begin{array}{c}
 \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array}
 \begin{pmatrix}
 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\
 0 & .25 & .33 & .33 & 0 & 0 & 0 \\
 .33 & 0 & .33 & .33 & .33 & 0 & 0 \\
 .33 & .25 & 0 & .33 & 0 & 0 & 0 \\
 .33 & .25 & .33 & 0 & 0 & 0 & 0 \\
 0 & .25 & 0 & 0 & 0 & .5 & .5 \\
 0 & 0 & 0 & 0 & .33 & 0 & .5 \\
 0 & 0 & 0 & 0 & .33 & .5 & 0
 \end{pmatrix}
 \end{array}
 \quad \text{(notice each column sums to one)}$$

例：矩阵乘法

$$A = \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \end{bmatrix}$$

$$B = \begin{bmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \\ b_{3,1} & b_{3,2} \end{bmatrix}$$

$$C = AB = \begin{bmatrix} a_{1,1}b_{1,1} + a_{1,2}b_{2,1} + a_{1,3}b_{3,1} & a_{1,1}b_{1,2} + a_{1,2}b_{2,2} + a_{1,3}b_{3,2} \\ a_{2,1}b_{1,1} + a_{2,2}b_{2,1} + a_{2,3}b_{3,1} & a_{2,1}b_{1,2} + a_{2,2}b_{2,2} + a_{2,3}b_{3,2} \end{bmatrix}$$

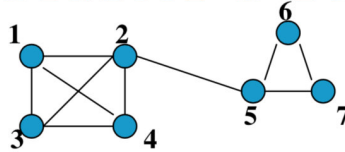
2. Markov Chain

为一个变量序列, X_1, X_2, X_3, \dots (本例子中为概率矩阵)，这里，针对下一步的概率仅考虑当前状态，过去和未来状态均不考虑。random walk为Markov Chain，只用转移概率矩阵。

增加weighted graphs和self loops，则为Markov Chain Cluster Structure

Markov Chain Cluster Structure

■ Example:



$$\begin{pmatrix}
 0 & .25 & .33 & .33 & 0 & 0 & 0 \\
 .33 & 0 & .33 & .33 & .33 & 0 & 0 \\
 .33 & .25 & 0 & .33 & 0 & 0 & 0 \\
 .33 & .25 & .33 & 0 & 0 & 0 & 0 \\
 0 & .25 & 0 & 0 & 0 & .5 & .5 \\
 0 & 0 & 0 & 0 & .33 & 0 & .5 \\
 0 & 0 & 0 & 0 & .33 & .5 & 0
 \end{pmatrix}
 \xrightarrow{\text{eventually}}
 \begin{pmatrix}
 .15 & .15 & .15 & .15 & .15 & .15 & .15 \\
 .2 & .2 & .2 & .2 & .2 & .2 & .2 \\
 .15 & .15 & .15 & .15 & .15 & .15 & .15 \\
 .15 & .15 & .15 & .15 & .15 & .15 & .15 \\
 .15 & .15 & .15 & .15 & .15 & .15 & .15 \\
 .1 & .1 & .1 & .1 & .1 & .1 & .1 \\
 .1 & .1 & .1 & .1 & .1 & .1 & .1
 \end{pmatrix}$$

3. [MCL][<https://www.zybuluo.com/chanvee/note/17815#算法基础>]

1. 给定一个无向图（有权无权都可），以及expansion 和 inflation 的参数 e 和 r。
2. 生成连接矩阵。
3. 给每个节点添加自环（可选）。
4. 归一化矩阵，得到转移矩阵。
5. 对矩阵进行 expansion 操作。
6. 对矩阵进行 inflation 操作。
7. 重复 5 和 6 直至收敛。
8. 根据最后得到的矩阵进行划分cluster。

算法核心：expansion，转移矩阵表示就是在初始状态下，某个节点转移到另一个节点的概率组成的矩阵。比如有1,2,3, 3个点的图，不考虑权重，两两之间互相连接，其转移矩阵就为： $P = \begin{bmatrix} 0 & 0.5 & 0.5 \\ 0.5 & 0 & 0.5 \\ 0.5 & 0.5 & 0 \end{bmatrix}$ 。而expansion这个步骤不断对这个转移矩阵进行自乘直到它不再改变为止，也就是求平稳分布的过程。expansion的作用就是为了能够将图中不同的区域连接起来；inflation，该过程目的是为了使得强连接更强，弱连接更弱，就是让转移矩阵中概率大的更大，小的更小。最直观操作就是幂操作，例如平方操作，然后在每一列归一化，就达到该目的了。

