Markov chain.
一種時間離散的隨機過程

$$\xi_{n+1} = M\xi_n$$

M:Markov matrix不同时刻状态间的转移機率
￡n:狀態向量

性質：無記憶性。
Eg:一維的random walk

超市問題：某城市有兩個超市，分別記作A和B.
每次去A超市的人有20%下次仍然去A超市，其餘的80%去B超市；
每次去B超市的人有40%下次仍然去B超市，其餘的60%去A超市。
請問，經過足夠長的時間後，去這A超市的人口和去B超市的人數比例會不會趨近於一個常數？如果會，請問這個常數為多少？

记第n次去A超市的人数为a_n，去B超市的人数为b_n. 于是，我们可以得到一个递归关系：

$$a_{n+1} = 0.2a_n + 0.6b_n$$
$$b_{n+1} = 0.8a_n + 0.4b_n$$

$$\begin{pmatrix} a_{n+1} \\ b_{n+1} \end{pmatrix} = \begin{pmatrix} 0.2 & 0.6 \\ 0.8 & 0.4 \end{pmatrix} \begin{pmatrix} a_n \\ b_n \end{pmatrix}, n \geq 1 \qquad M = \begin{pmatrix} 0.2 & 0.6 \\ 0.8 & 0.4 \end{pmatrix}$$

$$\begin{pmatrix} a_{n+1} \\ b_{n+1} \end{pmatrix} = \begin{pmatrix} 0.2 & 0.6 \\ 0.8 & 0.4 \end{pmatrix}^n \begin{pmatrix} a_1 \\ b_1 \end{pmatrix}$$

去A超市的人数与去B超市的人数的和是一个常数:

$$\begin{pmatrix} 1 & 1 \end{pmatrix} \begin{pmatrix} a_{n+1} \\ b_{n+1} \end{pmatrix} = \begin{pmatrix} 1 & 1 \end{pmatrix} \begin{pmatrix} 0.2 & 0.6 \\ 0.8 & 0.4 \end{pmatrix} \begin{pmatrix} a_n \\ b_n \end{pmatrix} = \begin{pmatrix} 1 & 1 \end{pmatrix} \begin{pmatrix} a_n \\ b_n \end{pmatrix}$$

Markov chain 的每一個column vector 和為 1;
因此:

$$\mathbf{1}^T M = \mathbf{1}^T$$

$$M^T \mathbf{1} = \mathbf{1}$$

根據 Gershgorin circle theorem ➔Markov matrix的特徵值不可能大於1.
因為該矩陣的每一個元素都大於0
根據Perron-Frobenius theorem➔ 該矩陣除了一個等於1的特徵值之外,
其餘所有的特徵值的絕對值都小於1.

$$M = U \begin{pmatrix} 1 & 0 \\ 0 & -0.4 \end{pmatrix} U^{-1}$$

$$U = \begin{pmatrix} 3 & 1 \\ 4 & -1 \end{pmatrix}$$

Markov matrix 的n次幂为：

$$M^n = U \begin{pmatrix} 1 & 0 \\ 0 & (-0.4)^n \end{pmatrix} U^{-1}$$

当幂为无穷大的时候，我们得到:

$$\lim_{n\to\infty} M^n = U \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} U^{-1} = \begin{pmatrix} 3/7 & 3/7 \\ 4/7 & 4/7 \end{pmatrix}$$

$$a_\infty = \frac{3}{7}(a_1 + b_1)$$

$$b_\infty = \frac{4}{7}(a_1 + b_1)$$

$\Rightarrow$ 穩態;
歸一化

最后去A超市的人数与B超市的人数的比例为3:4.

Markov matrix: 狀態間的轉移概率

So，每一個元素都應該是非負的

記瑪律科夫矩陣元素為M_{ij},表示系統從狀態j轉移到狀態i的概率

Perron–Frobenius Theorem

If $\mathbf{A}_{n\times n} \geq \mathbf{0}$ is irreducible, then each of the following is true.

- $r = \rho(\mathbf{A}) \in \sigma(\mathbf{A})$ and $r > 0$. (8.3.6)
- $alg\ mult_{\mathbf{A}}(r) = 1$. (8.3.7)
- There exists an eigenvector $\mathbf{x} > \mathbf{0}$ such that $\mathbf{A}\mathbf{x} = r\mathbf{x}$. (8.3.8)
- The unique vector defined by

$$\mathbf{A}\mathbf{p} = r\mathbf{p}, \qquad \mathbf{p} > \mathbf{0}, \quad \text{and} \quad \|\mathbf{p}\|_1 = 1, \tag{8.3.9}$$

  is called the ***Perron vector.*** There are no nonnegative eigenvectors for $\mathbf{A}$ except for positive multiples of $\mathbf{p}$, regardless of the eigenvalue.

- The Collatz–Wielandt formula $r = \max_{\mathbf{x}\in\mathcal{N}} f(\mathbf{x})$,

$$\text{where } f(\mathbf{x}) = \min_{\substack{1\leq i\leq n \\ x_i \neq 0}} \frac{[\mathbf{A}\mathbf{x}]_i}{x_i} \text{ and } \mathcal{N} = \{\mathbf{x} \mid \mathbf{x} \geq \mathbf{0} \text{ with } \mathbf{x} \neq \mathbf{0}\}$$

  was established in (8.3.3) for all nonnegative matrices, but it is included here for the sake of completeness.

Primitive Matrices

- A nonnegative irreducible matrix $\mathbf{A}$ having only one eigenvalue, $r = \rho(\mathbf{A})$, on its spectral circle is said to be a ***primitive matrix.***

- A nonnegative irreducible matrix having $h > 1$ eigenvalues on its spectral circle is called ***imprimitive,*** and h is referred to as ***index of imprimitivity.***

- A nonnegative irreducible matrix $\mathbf{A}$ with $r = \rho(\mathbf{A})$ is primitive if and only if $\lim_{k\to\infty}(\mathbf{A}/r)^k$ exists, in which case

$$\lim_{k\to\infty}\left(\frac{\mathbf{A}}{r}\right)^k = \mathbf{G} = \frac{\mathbf{p}\mathbf{q}^T}{\mathbf{q}^T\mathbf{p}} > \mathbf{0}, \tag{8.3.10}$$

  where $\mathbf{p}$ and $\mathbf{q}$ are the respective Perron vectors for $\mathbf{A}$ and $\mathbf{A}^T$. $\mathbf{G}$ is the (spectral) projector onto $N(\mathbf{A} - r\mathbf{I})$ along $R(\mathbf{A} - r\mathbf{I})$.

判斷primitive matrix：

1.充分非必要條件：M的對角元素至少有一個大於0.

2.

Frobenius's Test for Primitivity

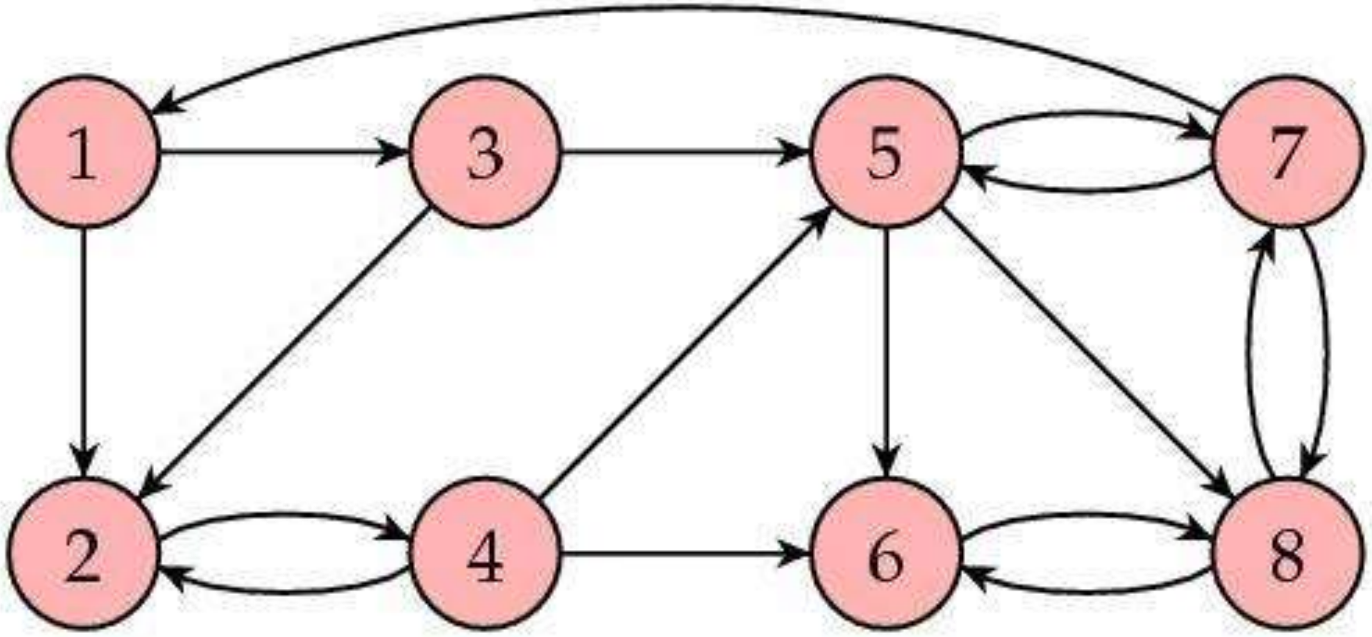
$\mathbf{A} \geq \mathbf{0}$ is primitive if and only if $\mathbf{A}^m > \mathbf{0}$ for some $m > 0$. (8.3.16)

性質：利用power iteration method

計算譜半徑。

$$\boldsymbol{v}_{k+1} = \frac{A\boldsymbol{v}_k}{||A\boldsymbol{v}_k||}, k \geq 0$$

Page rank:

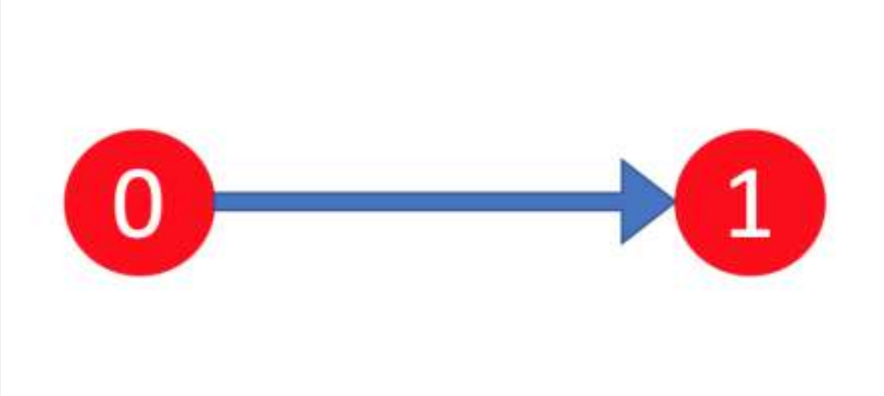


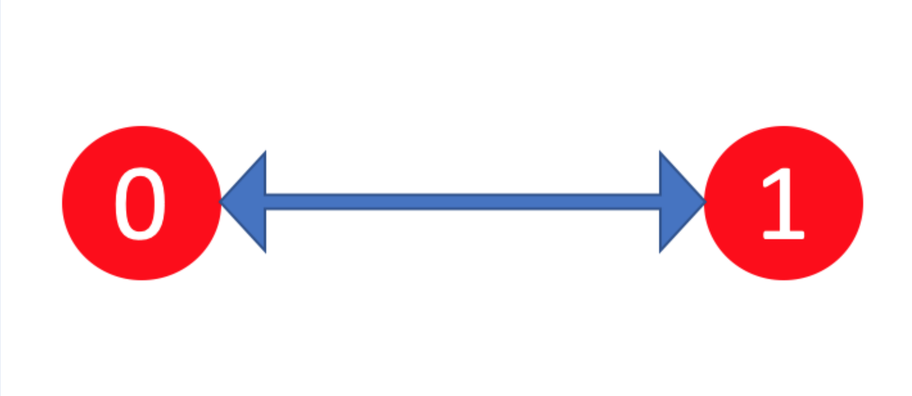
$$M = \begin{pmatrix} 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 1/3 & 0.0 \\ 1/2 & 0.0 & 1/2 & 1/3 & 0.0 & 0.0 & 0.0 & 0.0 \\ 1/2 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 1/2 & 1/3 & 0.0 & 0.0 & 1/3 & 0.0 \\ 0.0 & 0.0 & 0.0 & 1/3 & 1/3 & 0.0 & 0.0 & 1/2 \\ 0.0 & 0.0 & 0.0 & 0.0 & 1/3 & 0.0 & 0.0 & 1/2 \\ 0.0 & 0.0 & 0.0 & 0.0 & 1/3 & 1.0 & 1/3 & 0.0 \end{pmatrix}$$

$$\boldsymbol{\xi}_1 = M\boldsymbol{\xi}_0$$

$$\boldsymbol{\xi} = \begin{pmatrix} 0.05999997995306214 5 \\ 0.0675000087532483 5 \\ 0.0300000115184346 6 \\ 0.0674999899412349 3 \\ 0.0974999771878758 3 \\ 0.2025000561881978 6 \\ 0.180000052335199 5 \\ 0.294999924122746 7 \end{pmatrix}$$

PageRank算法与Perron-Frobenius定理





$$M \to M' = \alpha M + \frac{1-\alpha}{N}\mathbf{1}\mathbf{1}^T, \alpha \in (0,1)$$

无向图上的随机游走与拉普拉斯矩阵

$$M_{ij} = \frac{A_{ji}}{\sum_i A_{ji}}$$

$$D_{ij} = D_i \delta_{ij}$$

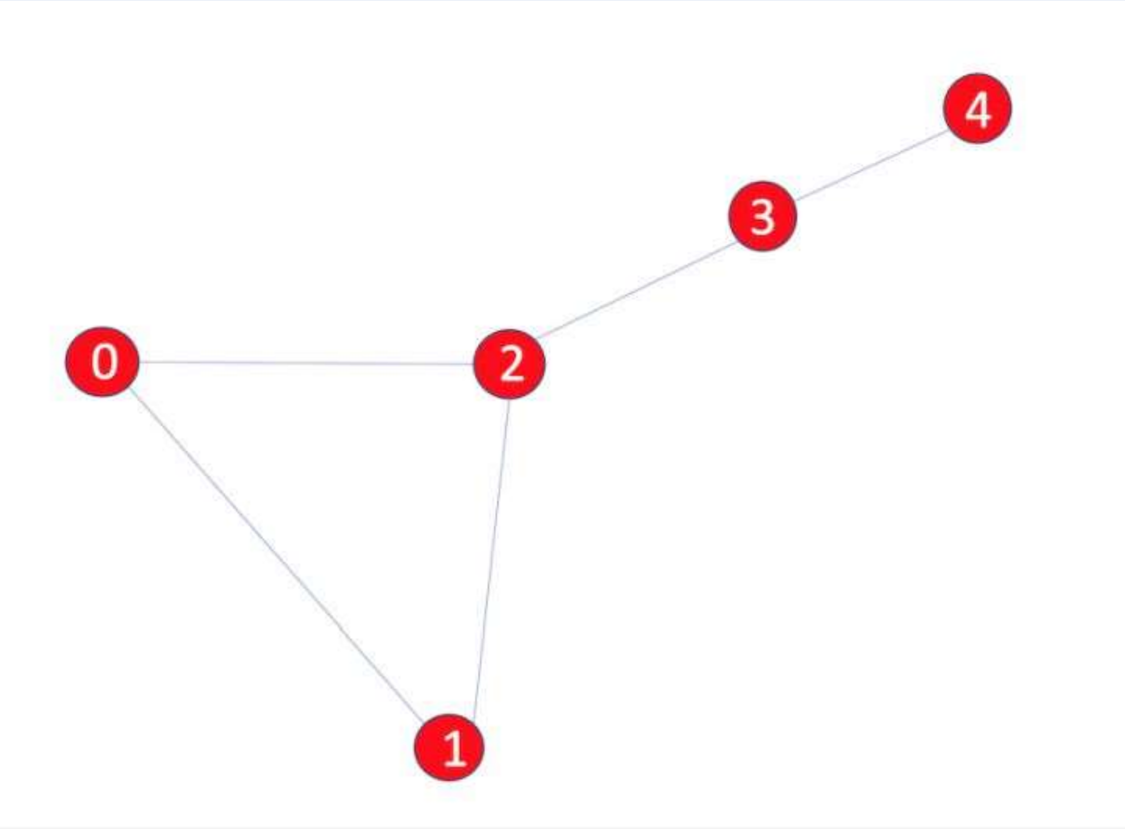
$$D_i = \sum_j A_{ij}$$

$$M^T = D^{-1} A$$

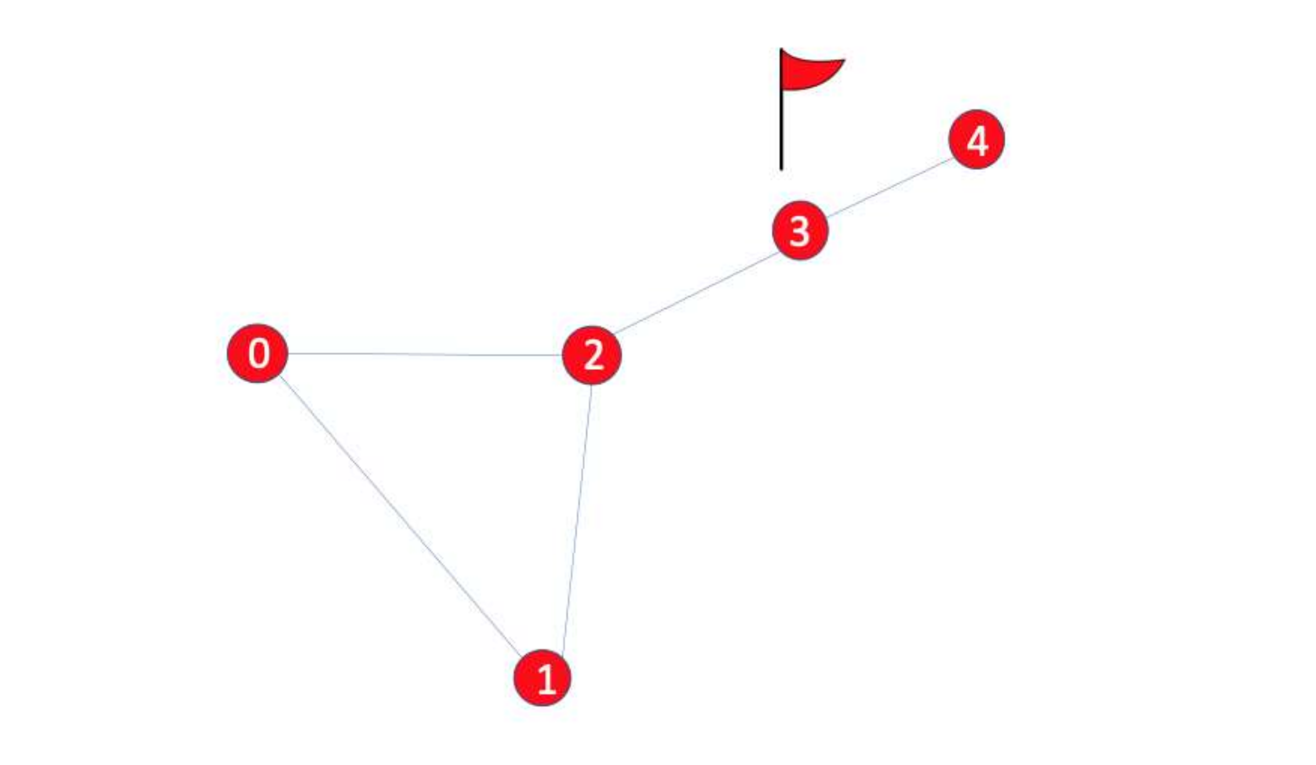
$$L = I - D^{-1} A$$

$$L = I - M^T$$

Random walk on graph ---hitting time 的精确计算



Hitting time:



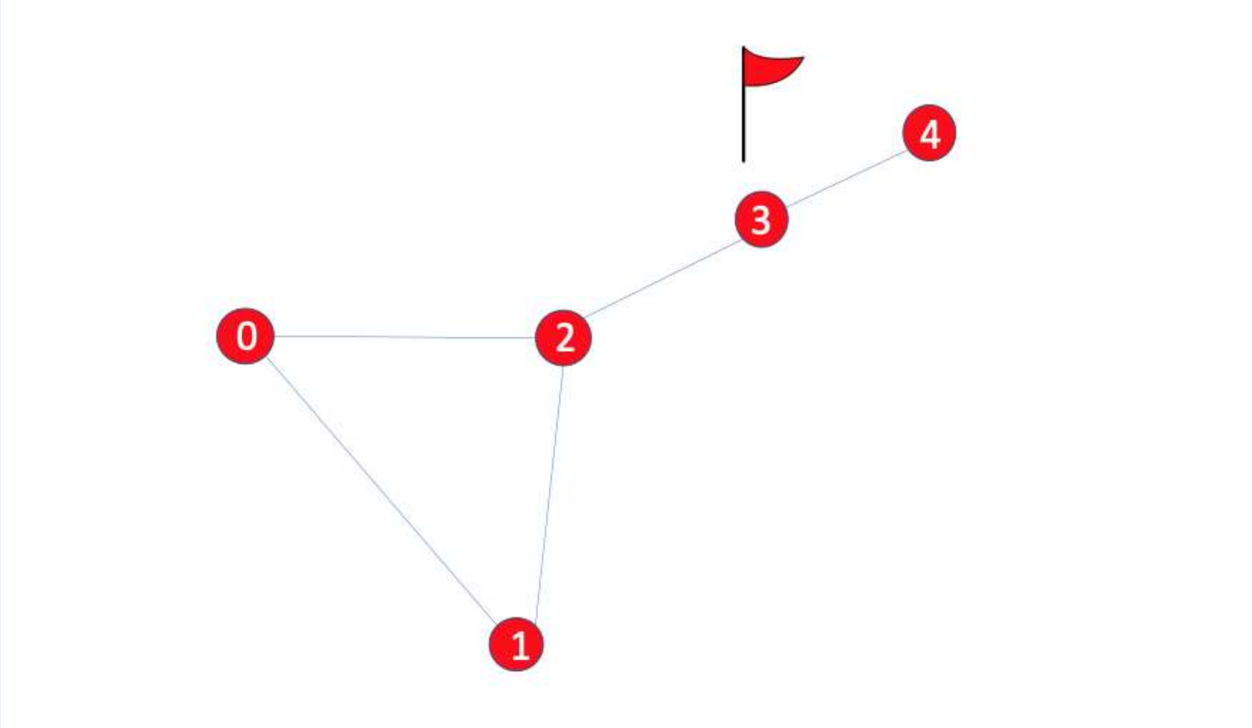
$$P(N_t^{(s)} = n) = \delta_{1,n}, n \geq 1$$

$$P(N_t^{(i)} = n) = \sum_{i_\alpha=1}^{m} P(i \to i_\alpha) P(N_t^{i_\alpha} = n-1)$$

$$P(N_t^{(i)} = n) = \sum_{i_\alpha=1, i_\alpha \neq t}^{m} P(i \to i_\alpha) P(N_t^{i_\alpha} = n-1), n \geq 2, i \neq t$$

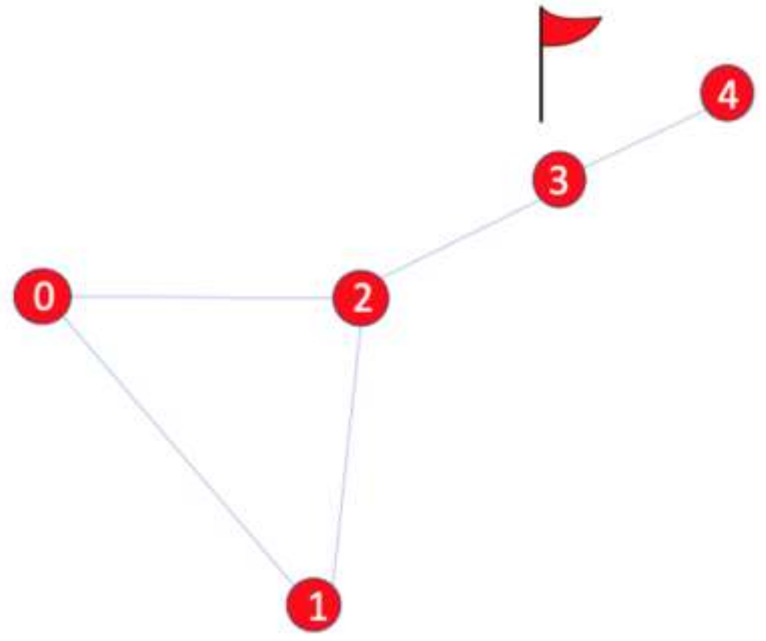
$$B_{ij} = \begin{cases} \frac{w_{ij}}{\sum_j w_{ij}} & A_{ij} \neq 0, j \neq t; \\ 0 & A_{ij} = 0, \text{ or } j = t. \end{cases}$$

$$P(N_t^{(i)} = n) = \sum_{\substack{j=1 \\ j \neq t}}^{|\mathbb{V}|} B_{ij} P(N_t^{(j)} = n-1), n \geq 2,$$



$$P(N_3^{(4)} = n) = \delta_{n,1}$$

$$\boldsymbol{X}_n = \begin{pmatrix} P(N_3^{(0)} = n) \\ P(N_3^{(1)} = n) \\ P(N_3^{(2)} = n) \end{pmatrix}$$



$$P(N_t^{(0)} = n) = \frac{1}{2} P(N_t^{(1)} = n-1) + \frac{1}{2} P(N_t^{(2)} = n-1)$$

$$P(N_t^{(1)} = n) = \frac{1}{2} P(N_t^{(0)} = n-1) + \frac{1}{2} P(N_t^{(2)} = n-1)$$

$$P(N_t^{(2)} = n) = \frac{1}{3} P(N_t^{(0)} = n-1) + \frac{1}{3} P(N_t^{(1)} = n-1)$$

$$\boldsymbol{X}_n = \begin{pmatrix} P(N_3^{(0)} = n) \\ P(N_3^{(1)} = n) \\ P(N_3^{(2)} = n) \end{pmatrix}$$

$$= \begin{pmatrix} 0 & 1/2 & 1/2 \\ 1/2 & 0 & 1/2 \\ 1/3 & 1/3 & 0 \end{pmatrix} \begin{pmatrix} P(N_3^{(0)} = n-1) \\ P(N_3^{(1)} = n-1) \\ P(N_3^{(2)} = n-1) \end{pmatrix}$$

$$= B\boldsymbol{X}_{n-1}, n \geq 2$$

解：$\boldsymbol{X}_n = B^{n-1}\boldsymbol{X}_1, n \geq 2$

$B^0 = I$ $\quad \boldsymbol{X}_n = B^{n-1}\boldsymbol{X}_1, n \geq 1$

$$\boldsymbol{X}_1 = \begin{pmatrix} 0 \\ 0 \\ 1/3 \end{pmatrix}$$

hitting time 期望值和方差的精确计算

$$\boldsymbol{X}_n^i = P(N_t^{(i)} = n), n \geq 1$$

$$\boldsymbol{X}_n = B\boldsymbol{X}_{n-1}, n \geq 2$$

$$\boldsymbol{X}_n = B^{n-1}\boldsymbol{X}_1, n \geq 1$$

记随机向量 X_n 的概率密度函数为:

$$\boldsymbol{f}(x) = \sum_{n=1}^{\infty} \boldsymbol{X}_n \delta(x-n)$$

它的特征函数为:

$$\hat{\boldsymbol{f}}(\omega) = \int_{x\in\mathbb{R}} \boldsymbol{f}(x) e^{i\omega x} = \sum_{n=1}^{\infty} \boldsymbol{X}_n e^{i\omega n}, \omega \in \mathbb{R}$$

$$z = e^{i\omega}$$

$$\tilde{\boldsymbol{f}}(z) = \hat{\boldsymbol{f}}(\omega) = \sum_{n=1}^{\infty} \boldsymbol{X}_n z^n, z = e^{i\omega}, \omega \in \mathbb{R}$$

$$\tilde{\boldsymbol{f}}(z) = \left(\sum_{n=1}^{\infty} z^n B^{n-1}\right) \boldsymbol{X}_1 = z(I - zB)^{-1} \boldsymbol{X}_1$$

hitting time的期望值為

$$\tilde{\boldsymbol{f}}'(1) = (I - B)^{-1}(B\tilde{\boldsymbol{f}}(1) + \boldsymbol{X}_1) = (I - B)^{-1} \tilde{\boldsymbol{f}}(1)$$

$$\tilde{\boldsymbol{f}}(1) = \begin{pmatrix} 1 & 1 & 1 \ldots & 1 \end{pmatrix}^{\mathrm{T}}$$

矩陣B的譜半徑小於 1 決定了. $\lim_{n \to \infty} B^n \tilde{\boldsymbol{f}}(1) = 0$

重新將求逆寫成冪級數的形式為:

$$\tilde{\boldsymbol{f}}'(1) = \sum_{n=0}^{\infty} B^n \tilde{\boldsymbol{f}}(1)$$

$$\tilde{\boldsymbol{f}}''(1) = 2B(I - B)^{-2} \tilde{\boldsymbol{f}}(1) = 2\sum_{n=1}^{\infty} nB^n \tilde{\boldsymbol{f}}(1)$$

不要計算疏鬆陣列的冪的原因

疏鬆陣列的冪不一定是疏鬆陣列。

疏鬆陣列乘以向量的運算量遠小於兩個疏鬆陣列相乘的運算量

演算法的正確性檢驗

已知概率轉移矩陣為

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

那麼hitting time的概率分佈的生成函數為

$$\tilde{\boldsymbol{f}}(z) = z(I - zB)^{-1}\boldsymbol{X}_1$$
$$= \frac{z}{3}\frac{1}{1 - \frac{7}{12}z^2 - \frac{z^3}{6}}\begin{pmatrix} \frac{z}{2} + \frac{z^2}{4} \\ \frac{z}{2} + \frac{z^2}{4} \\ 1 - \frac{z^2}{4} \end{pmatrix}$$

求微分得到hitting time的期望值為

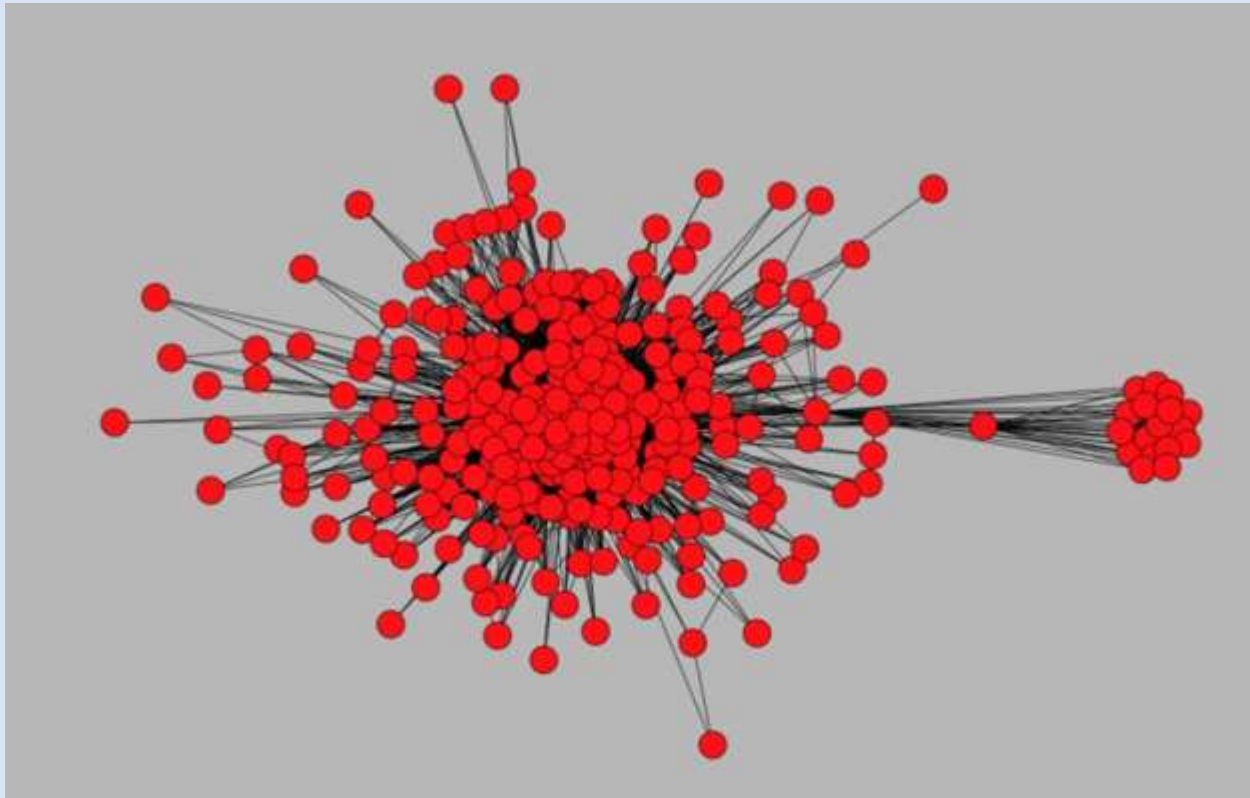
$$\tilde{\boldsymbol{f}}'(1) = \begin{pmatrix} \langle N_3^{(0)} \rangle \\ \langle N_3^{(1)} \rangle \\ \langle N_3^{(2)} \rangle \end{pmatrix} = \begin{pmatrix} 9 \\ 9 \\ 7 \end{pmatrix}$$

$\langle N_3^{(i)} \rangle$	Analytical	Monte Carlo	Numerical
$\langle N_3^{(0)} \rangle$	9	9.00323	8.99999999999999
$\langle N_3^{(1)} \rangle$	9	8.96693	8.99999999999999
$\langle N_3^{(2)} \rangle$	7	7.00013	7.00000000000002

演算法的程式實現以及在真實資料集上的運行結果

生成《紅樓夢》中的人物關係圖。
使用了如下假設:
1.每一段文字代表一個場景
2.如果兩個人名出現在了同一段，那麼我們假設這兩個人相互認識
3.兩個人出現在同一個場景裡面的次數越多，那麼這兩人的關係越密切

計算其他節點到“賈寶玉”的hitting time的期望值。



Targets = 贾宝玉 node -> targets:hitting_time_expectation,hitting_time_variance
王济仁 -> targets:6.60878021655381,100.21977910226013
王一贴 -> targets:7.993689781323544,114.22327387823792
茗玉 -> targets:9.278693958781961,128.21027251612435
龄官 -> targets:9.308905582567418,127.10286081255006
小鹊 -> targets:9.682179507899864,128.97898091741558
靛儿 -> targets:9.891742860405532,131.0611323115241
度恨菩提 -> targets:9.96198427587644,129.27202481155385
引愁金女 -> targets:9.96198427587644,129.27202481155385
痴梦仙姑 -> targets:9.96198427587644,129.27202481155382
媚人 -> targets:10.089225862075923,131.31268931793605
傅秋芳 -> targets:10.114989813496429,131.2104070950906
顽石 -> targets:10.191041554126159,132.61099062228175
李嬷嬷 -> targets:10.21416392613562,132.4462307733648
茗烟 -> targets:10.21535743199583,133.2311927983676
傅试 -> targets:10.271788215707486,132.43434176157876
小吉祥儿 -> targets:10.28688145754802,132.37147571738478
缕儿 -> targets:10.302565703603701,132.68713952270372
甄宝玉 -> targets:10.418421118946386,134.2295643874911
宝官 -> targets:10.421577503496536,132.28854980641864
玉官 -> targets:10.421577503496536,132.28854980641864
鲍太医 -> targets:10.504163744771263,134.35814774587627
宋嬷嬷 -> targets:10.543308984816829,133.42330086483082
李贵 -> targets:10.765711911974813,135.91963867344646
引泉 -> targets:10.784569054548784,136.46203720265532
王嬷嬷 -> targets:10.830019505119118,134.63844112666345
秋纹 -> targets:10.942037217615571,135.85113418151616
坠儿 -> targets:11.024281759834153,136.161639260118
扫花 -> targets:11.036600202715283,137.45435748854254
墨雨 -> targets:11.059956375430545,136.76135356238373

Targets = 贾宝玉 node -> targets:hitting_time_expectation,hitting_time_variance
茗烟 -> targets:10.068414800628682,130.40038959696477
李嬷嬷 -> targets:10.107201150498538,129.88104728836706
甄宝玉 -> targets:10.31624139950134,131.72056545675474
宋嬷嬷 -> targets:10.458277271459746,130.91467058523597
引泉 -> targets:10.674728790615026,133.61694827542823
李贵 -> targets:10.685131006901772,133.43225931904715
秋纹 -> targets:10.851587959250681,133.27507470940145
坠儿 -> targets:10.91692617076093,133.55674225505118
扫花 -> targets:10.93506432754307,134.65080138543956
墨雨 -> targets:10.963519367103848,134.07881876942957
四儿 -> targets:10.995115948304552,135.86114322094068
麝月 -> targets:11.007034073973653,133.8812057572318
秦钟 -> targets:11.088921755900607,135.823259530496
晴雯 -> targets:11.099291878420278,134.27368542741468
锄药 -> targets:11.114432574171522,140.83239593806383
袭人 -> targets:11.12033502494042,134.93983018121338
蒋玉菡 -> targets:11.131871657942694,150.22426256680035
云儿 -> targets:11.158117456267759,154.50371748350355
花自芳 -> targets:11.159635116158917,135.03488653940673
神瑛侍者 -> targets:11.19719308040166,135.26322807765024
紫鹃 -> targets:11.238032129449548,134.96893585870797
张若锦 -> targets:11.271190536915146,134.442190145199
王荣 -> targets:11.271190536915146,134.44219014519905
赵亦华 -> targets:11.271190536915146,134.442190145199
钱启 -> targets:11.271190536915146,134.442190145199
张道士 -> targets:11.293707888677178,136.54157131729812
春燕 -> targets:11.320469351943585,134.61149648125874
雪雁 -> targets:11.347823868777406,135.23259359088817
林黛玉 -> targets:11.370284367698098,135.77378174153088
赖尚荣 -> targets:11.392227888897665,136.6825253719723
茜雪 -> targets:11.39461319531203,135.30715674887114

Target = 贾宝玉
王济仁
王一贴
茗玉
龄官
小鹊
靛儿

Random Walk:

執行時間長，耗費記憶體大，模型超參數多，而且不具有可解釋性。

Deep Walk:

target = 贾宝玉	亲密度(归一化向量内积)
林黛玉	0.660261809826
史太君	0.656278014183
薛宝钗	0.624927401543
袭人	0.614130139351
王夫人	0.592175602913
史湘云	0.556256771088
麝月	0.548716425896
王熙凤	0.546379208565

$$d_{ij} := \theta_{ij} = \arccos\left(\frac{\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle}{||\boldsymbol{x}_i||||\boldsymbol{x}_j||}\right)$$

赋范线性空间中的距离函数要满足这三个条件:

1.对称性 : $d(x,y) = d(y,x)$
2.正定性 : $d(x,y) \geq 0$,其中等号成立的条件是$x = y$.
3.三角不等式 : $d(x,y) \leq d(x,z) + d(z,y)$

提出新演算法的原因:

1.簡單的隨機遊走沒法找到主角的親密朋友，只能給出隨從;
2.DeepWalk演算法計算量大，模型超參數多，而且給出的距離函數又永遠都是對稱的;
3.簡單隨機遊走可以給出不對稱的距離函數，但是我們要想辦法增加主角隨從到主角的距離，與此同時減小主角親密朋友到主角的距離。

frustrated random walks

simple random walks	frustrated random walks	DeepWalk
Target = 贾宝玉	Target = 贾宝玉	Target = 贾宝玉
王济仁	袭人	林黛玉
王一贴	林黛玉	史太君
茗玉	紫鹃	薛宝钗
龄官	秋纹	袭人
小鹊	麝月	王夫人
靛儿	薛宝钗	史湘云
痴梦仙姑	雪雁	麝月
度恨菩提	晴雯	王熙凤
引愁金女	史湘云	薛姨妈
媚人	史太君	紫鹃
傅秋芳	王夫人	晴雯
顽石	贾探春	李纨
李嬷嬷	李嬷嬷	秋纹
茗烟	李纨	鸳鸯
傅试	贾政	贾探春

$$\boxed{O(V^2)}$$

$$\mathbb{E}\boldsymbol{N}_t = \boldsymbol{f}'(1) = \sum_{n=0}^{\infty} B^n \mathbf{1}$$

$$\lambda_{max} \approx \frac{\sum_{ij} B_{ij}}{N_B}$$

$$\sum_{ij} B_{ij} = N_B - \sum_{i\in\{\bar{t}\}} \frac{w_{it}}{D_i}\frac{w_{ti}}{D_t}$$

$$N_{total} \approx (2E+V)V$$

$$N_{max} = \frac{N_B}{\sum_{i\in\{\bar{t}\}} \frac{w_{it}}{D_i}\frac{w_{ti}}{D_t}}$$

$$N_{max} \approx N_B \lesssim V$$

$$v_i' = \sum_j B_{ij} v_j, i = 0, 1, 2, \ldots, N_B - 1$$

$$\sum_i \left(D_i + 1\right) \lesssim 2E + V$$