计算机应用编程 实验三

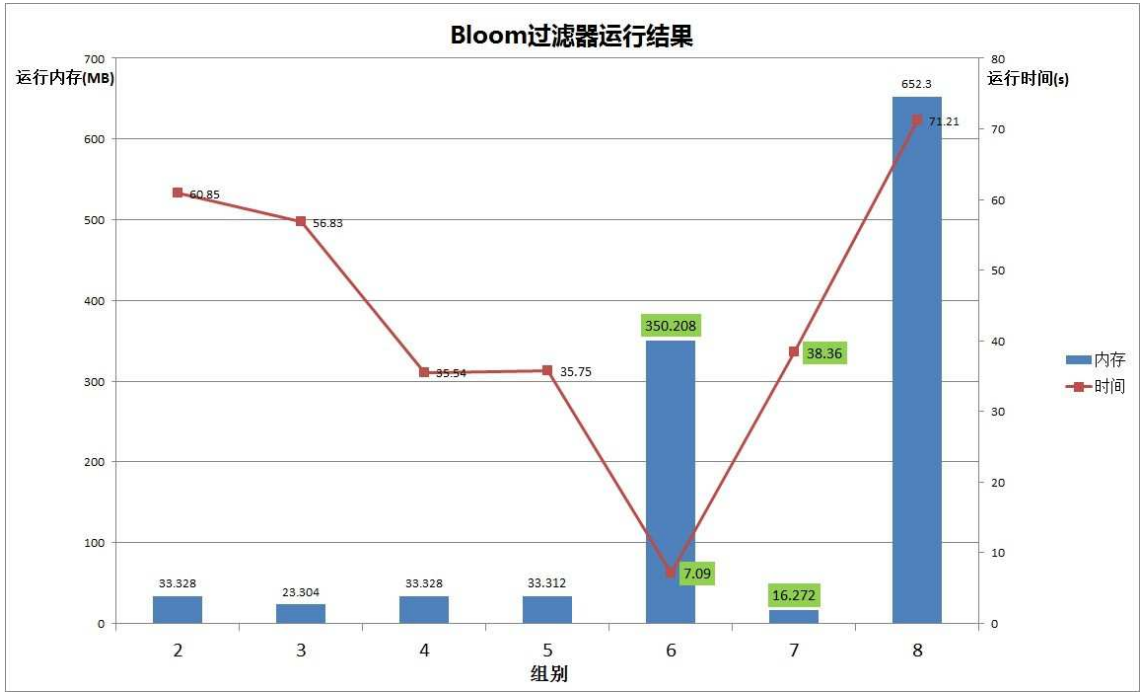
链接分析器

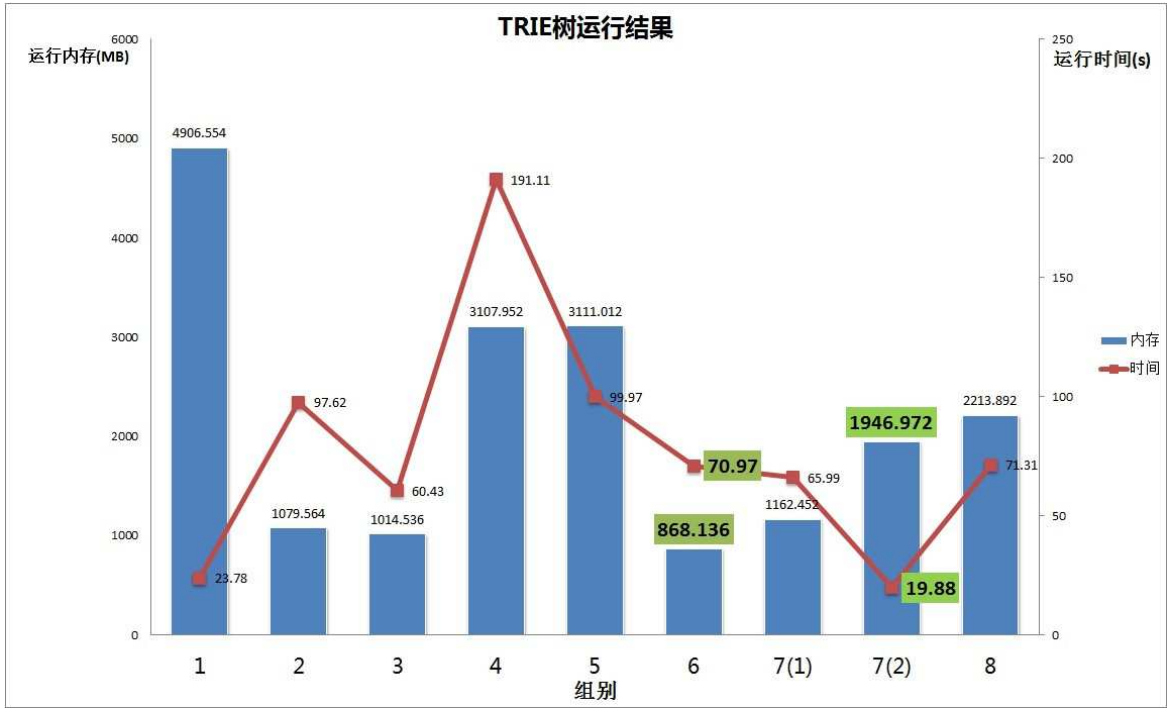
熊永平@网络技术研究院

ypxiong@bupt.edu.cn

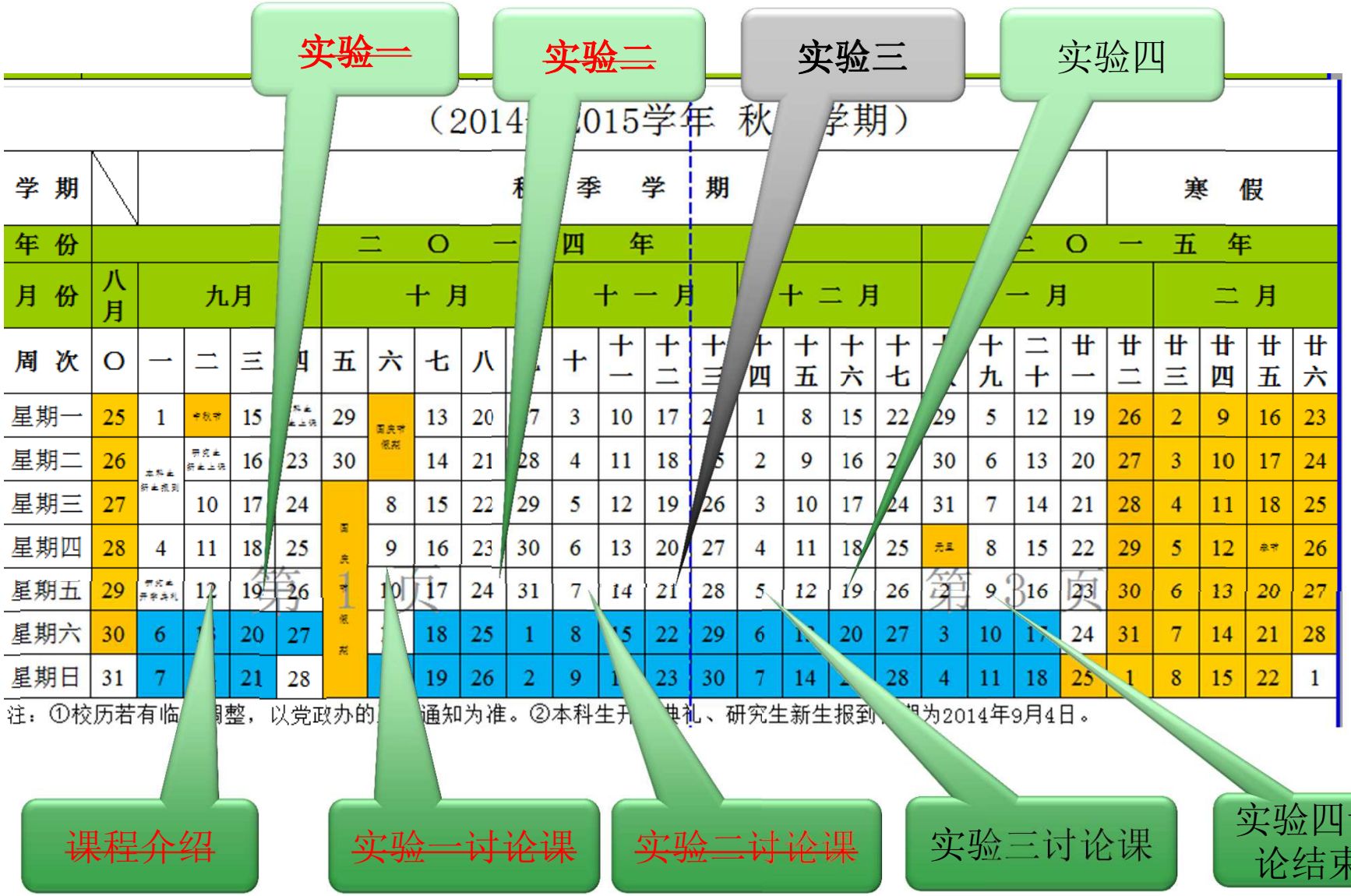
周五10:10-12:00@3-130

2014.11.20


Bloom过滤器运行结果
运行内存(MB)
运行时间(s)
700
600
500
400
300
200
100
0
80
70
60
50
40
30
20
10
0
652.3
71.21
60.85
56.83
35.54
35.75
350.208
38.36
7.09
33.328
23.304
33.328
33.312
16.272
内存
时间
2
3
4
5
6
7
8
组别


TRIE树运行结果
运行内存(MB)
运行时间(s)
6000
5000
4000
3000
2000
1000
0
250
200
150
100
50
0
4906.554
191.11
97.62
3107.952
3111.012
99.97
70.97
65.99
1946.972
2213.892
71.31
60.43
1079.564
1014.536
868.136
1162.452
19.88
23.78
内存
时间
1
2
3
4
5
6
7(1)
7(2)
8
组别

课程进度


实验一
实验二
实验三
实验四
（2014-2015学年 秋 学期）
学 期
季 学 期
寒 假
年 份
二 〇 一 四 年
二 〇 一 五 年
月 份
八月
九月
十 月
十 一 月
十 二 月
一 月
二 月
周 次
星期一
星期二
星期三
星期四
星期五
星期六
星期日
注：①校历若有临 调整，以党政办的 通知为准。②本科生升 典礼、研究生新生报到 期为2014年9月4日。
课程介绍
实验一讨论课
实验二讨论课
实验三讨论课
实验四讨论结束

实验二：链接分析器

1、实验任务

2、相关基础原理

3、分析器设计与实现

4、实验说明

实验目标

- 目标
  - 设计一个链接分析程序，实现
  - 构建16W个网页链接图
  - 利用PageRank算法计算重要度最高的前5个页面
- 编程技能
  - C++语言练习
  - 图分析
  - 随机过程实现
  - 稀疏矩阵
  - PageRank算法

1、实验任务

2、相关基础原理

3、分析器设计与实现

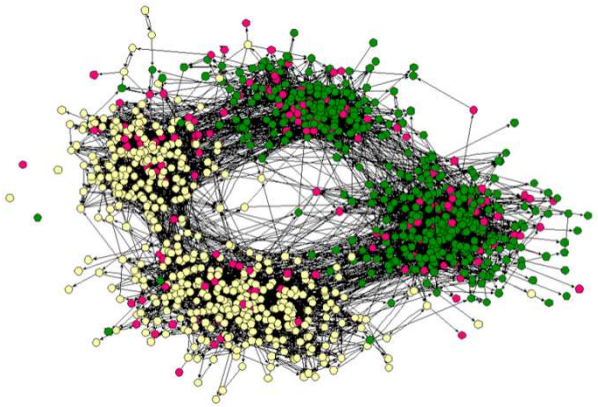
4、实验说明

相关基础原理

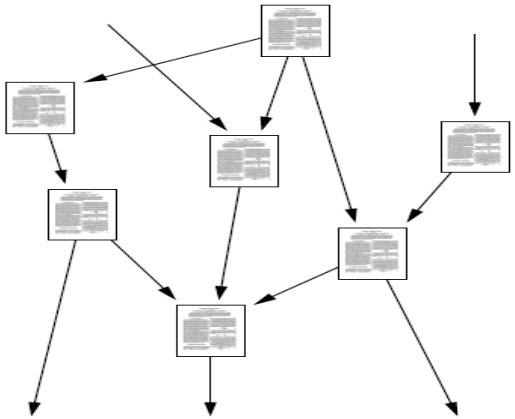
- 网络与图
- 谱结构与马尔可夫更新过程
- PageRank算法

真实网络（1）：社会网络

朋友关系网

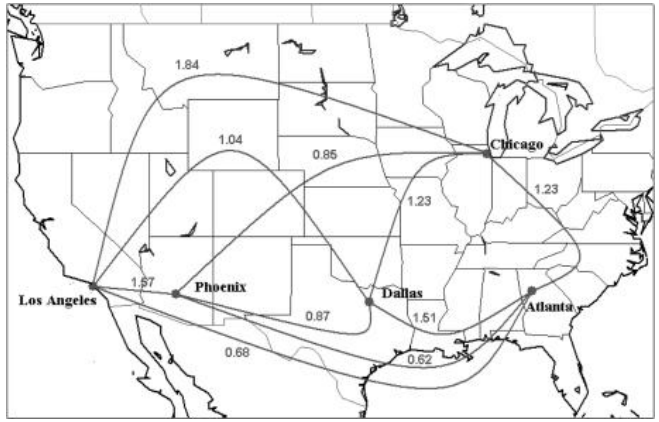


科学引文网

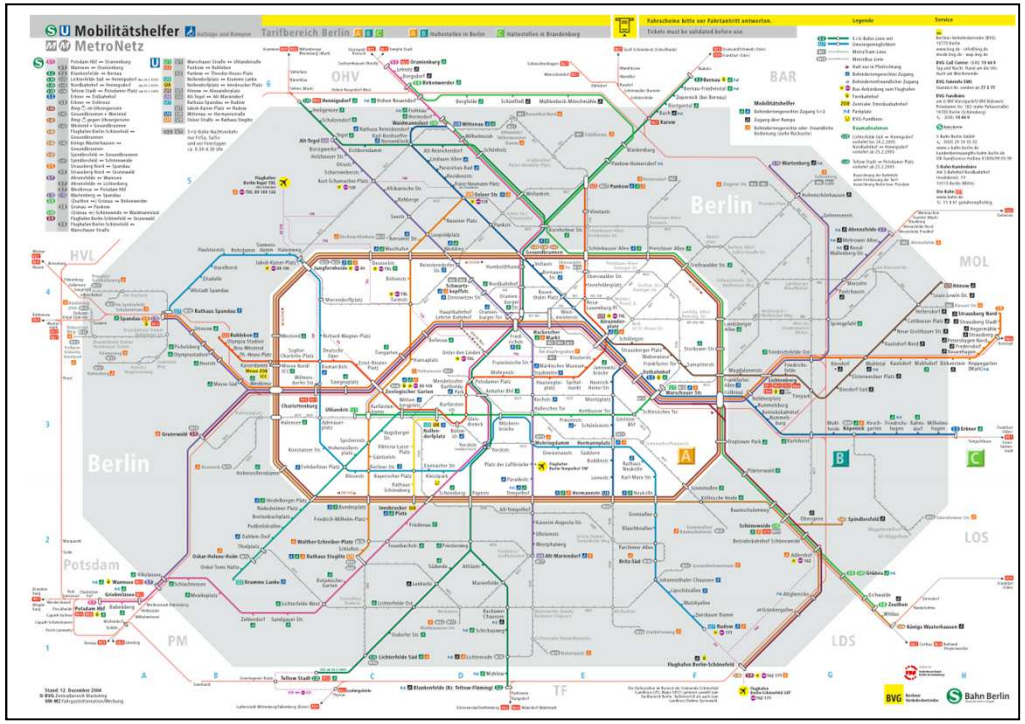


真实网络（2）：交通网络

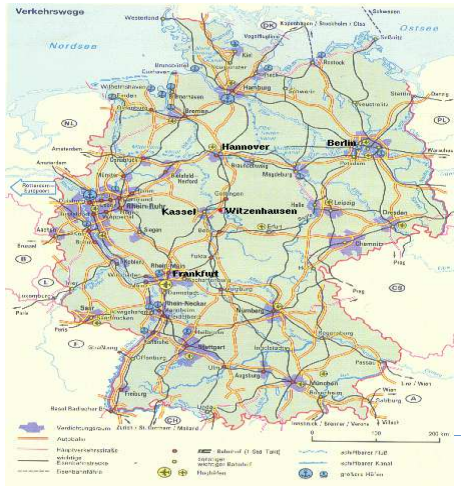
航空网



城市公共交通网

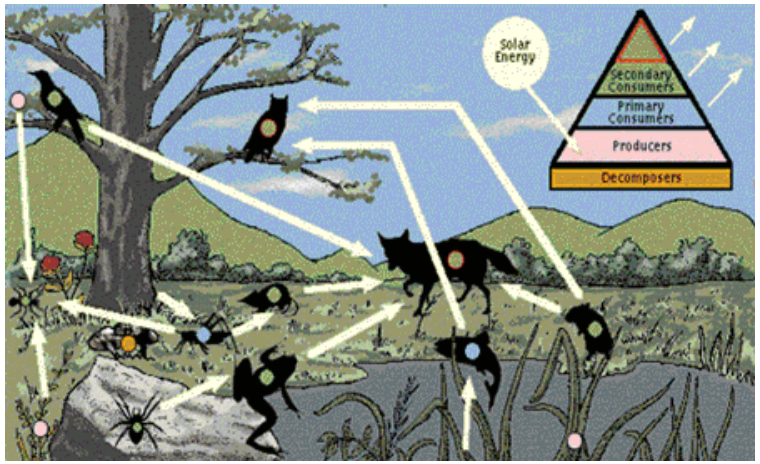


道路交通网

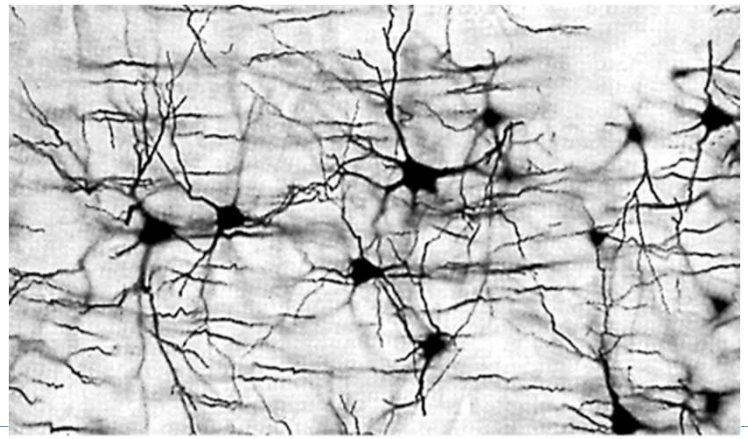


真实网络（3）：生物网络

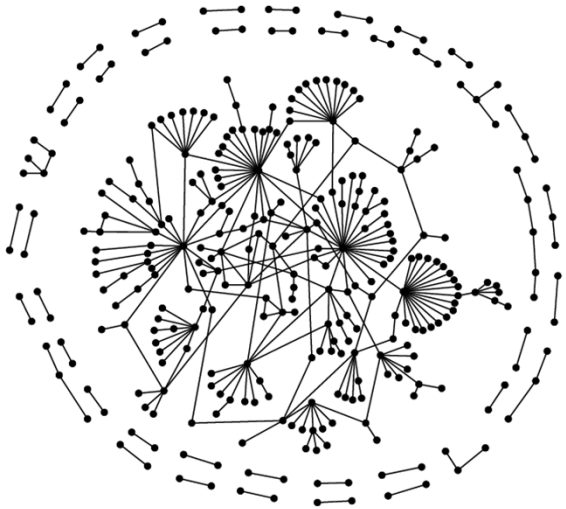
生态网络



神经网络



蛋白质相互作用网络



不同领域的真实网络

- 社会网：演员合作网，友谊网，姻亲关系网，科研合作网，Email网
- 生物网：食物链网，神经网，新陈代谢网，蛋白质网，基因网络
- 信息网络：**WWW**，专利使用，论文引用，计算机共享
- 技术网络：电力网，Internet，电话线路网
- 交通运输网：航线网，铁路网，公路网，自然河流网

网络和图的概念

- 网络
  - 抽象为它用于表示事物之间的相互联系。
  - 节点通常用来表示系统中的部件；
  - 边通常用来表示系统中部件之间的关系。
  - 网络就是由节点与边构成的一张图，一般认为是加权图
- 图
  - 图由一个顶点的有穷非空集合V(G)和一个弧的集合E(G)组成，通常记做G=(V,E)。用有序对<v,w>表示从v到w的一条弧。弧具有方向性，以带箭头的线段表示，通常称v为弧尾或始点，称w为弧头或终点，此时的图称为有向图。若图中从v到w有一条弧，同时从w到v也有一条弧，则以无序对(v,w)代替这两个有序对<v,w>和<w,v>,表示v和w之间的一条边。此时的图在顶点之间不再强调方向性的特征，称为无向图。

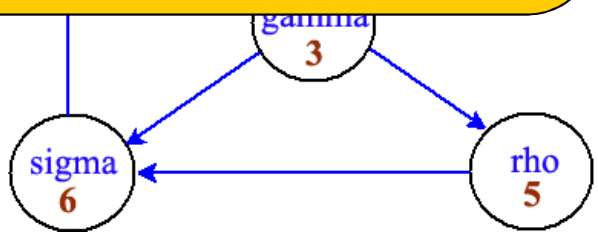
图的表示：邻接矩阵

- **定义邻接矩阵**

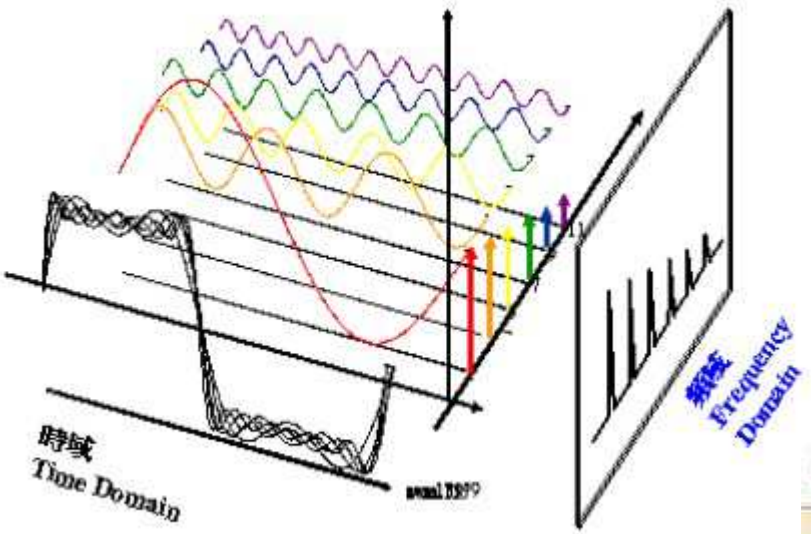
$$G = (g_{ij}), \quad \text{其中 } g_{ij} = \begin{cases} 1, & \text{如果存在从 } j \text{ 到 } i \text{ 的弧} \\ 0, & otherwise \end{cases}$$

“图”(Graph)和“矩阵”(Matrix)结合
矩阵是代数学中最重要的概 念，给了图一个矩阵表达，建立了用代数方法研究图的途径

$$\begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}$$

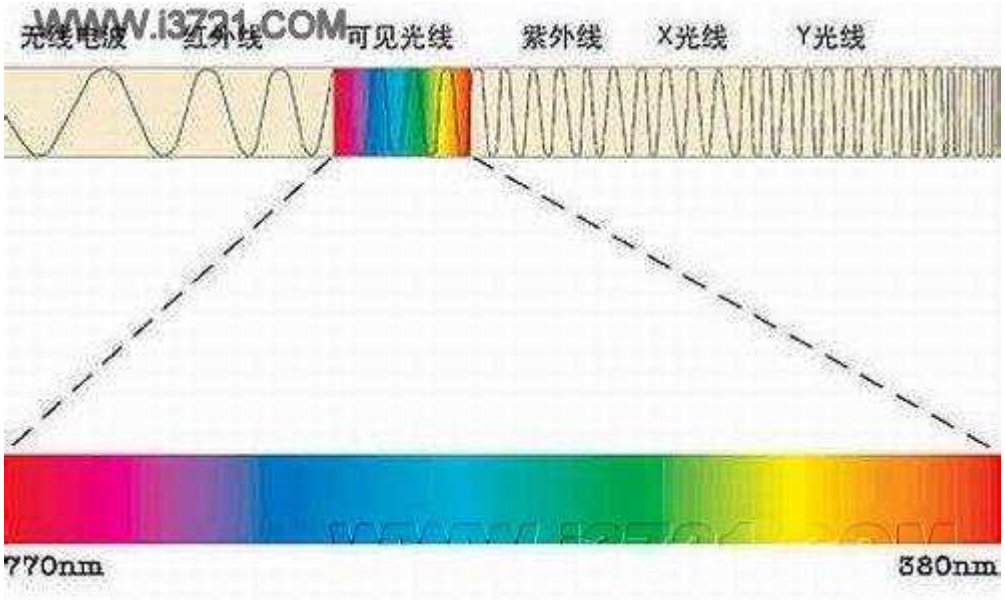


谱的概念



圖二 時域與頻域的差異

- 简单地说，谱这个概念来自“分而治 之”的策略。一个复杂的东西不好直接研究，就把它分解成简单的分量。
- 如果我们把一个东西看成是一些分量叠加而成，那么这些分量以及它们各自所占的比例，就 叫这个东西的谱。
- 所谓频谱，就是把一个信号分解成多个频率单一的分量。

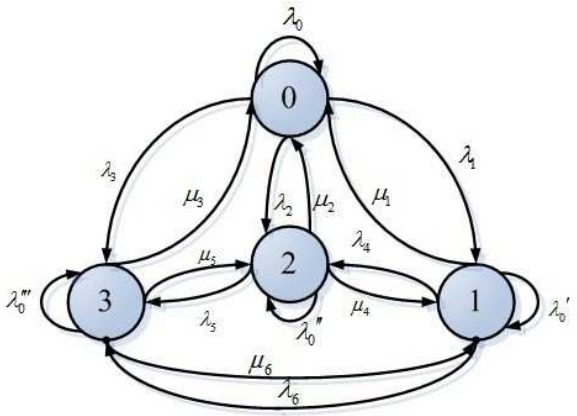


矩阵的谱结构

- 定义
  - 矩 阵的谱，就是它的特征值和特征向量，普通的线性代数课本会告诉你定义：如果A v = c v，那么c就是A的特征值，v就叫特征向量。这仅仅是数学家发明的一种数学游戏么？
  - 其实，这里的谱代表了一种分量结构，它使用“分而治之”策略来研究矩阵，这里可以把矩阵理解为一个操作 (operator)，它的作用就是把一个向量变成另外一个向量：y = A x。对于某些向量，矩阵对它的作用很简单，A v = cv，相当于就把这个向量v拉长了c倍。把这种和矩阵A能如此密切配合的向量v1, v2, ... 叫做特征向量，这个倍数c1, c2,...叫特征值。
  - 当出现一个新的向量x 的时候，可以把x 分解为这些向量的组合，x = a1 v1 + a2 v2 +...，那么A对x的作用就可以分解：A x = A (a1 v1 + a2 v2 + ...) = a1 c1 v1 + a2 c2 v2... 所以，矩阵的谱就是用于分解一个矩阵的。

马尔可夫更新过程-图的时间角度

- 马尔可夫过程
  - “将来只由现在决定，和过去无关”。
  - 考虑一个图，图上每个点有一个值，会被不断更新。每个点通过一些 边连接到其它一些点上，对于每个点，这些边的值都是正的，和为1。在图上每次更新一个点的值，就是对和它相连接的点的值加权平均。



$$P=\begin{bmatrix} \lambda_0 & \lambda_1 & \lambda_2 & \lambda_3 \\ \mu_1 & \lambda_0' & \lambda_4 & \lambda_6 \\ \mu_2 & \mu_4 & \lambda_0'' & \lambda_5 \\ \mu_3 & \mu_6 & \mu_5 & \lambda_0''' \end{bmatrix}$$

转**移矩**阵

  - 如果图是联通并且非周期 （数学上叫各态历经性， ergodicity)，那么这个过程最后会收敛到一个唯一稳定的状态（平衡状态)。

图-谱联姻(1)

- 矩阵谱结构
  - 根 据上面的定义，可以理解邻接矩阵A其实就是这个马尔可夫过程的转移概率矩阵。
  - 把各个节点的值放在一起可以得到一个向量v，那么我们就可以获得对这个过 程的代数表示， v(t+1) = A v(t)。
  - 稳态情况下，v = Av。我们可以看到稳定状态就是A的一个特征向量，特征值就是1。
  - 这里谱的概念进来了。我们把A的特征向量都列出来v1, v2, ...，它们有 Avi = ci vi。vi其实就是一种很特殊，但是很简单的状态，对它每进行一轮更新，所有节点的值就变成原来的ci倍。如果0 < ci< 1，那么，相当于所有节点的值呈现指数衰减，直到大家都趋近于0。

图-谱联姻(2)

- 收敛速度
  - 一 般情况下，更新过程开始于一个任意一个状态u
  - 用谱的方法来分析，把u分解成 u = v1 +c2 v2 + c3 v3 + ...（在数学上可以严格证明，对于上述的转移概率矩阵，最大的特征值就是1，这里对应于平衡状态v1，其它的特征状态v2, v3, ...,对应于特征值1 > c2 > c3 > ... > -1)。
  - 可以看到，当更新进行了t 步之后，状态变成u(t) = v1 + c2^t v2 + c3^t v3 + ...，除了代表平衡状态的分量保持不变外，其它分量随着t增长而指数衰减，最后，其它整个趋近于平衡状态。
  - 从 上面的分析看到，这个过程的收敛速度，和衰减得最慢的那个非平衡分量是密切相关的，它的衰减速度取决于第二大特征值c2，c2的大小越接近于1，收敛越慢，越接近于0，收敛越快。

图-谱联姻(3)

- 矩阵谱的意义
  - 它帮助把一个图上运行的马尔可夫过程分解为多个简单的字过程的叠加，这里面包含一个平衡 过程和多个指数衰减的非平衡过程。
  - 它指出平衡状态是对应于最大特征值1的分量，而收敛速度主要取决于第二大特征值。
- 谱的作用
  - 第二大特征值c2对于描述这个过程至关重要
  - 如果要设计一个采样过程或者 更新过程，那么就要追求一个小的c2，它一方面提高过程的效率，另外一方面，使得图的结构改变的时候，能及时收敛，从而保证过程的稳定。

聚类结构-图的空间角度

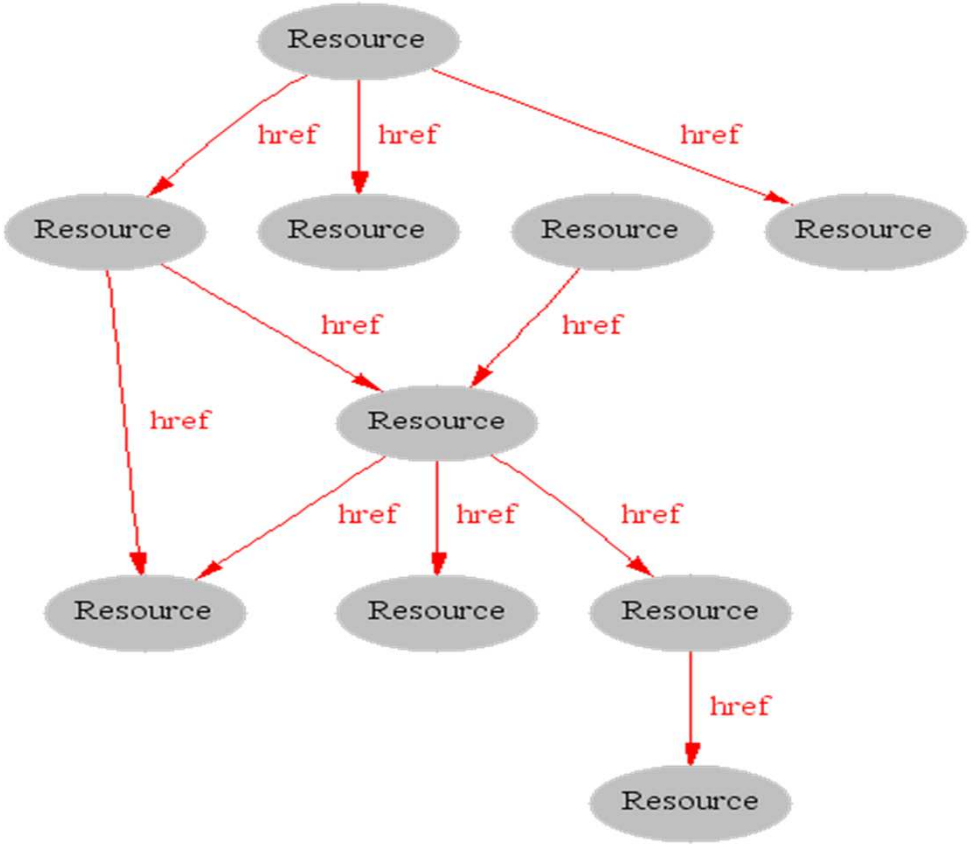
- 聚类结构
  - c2 的大小取决于图上的聚类结构。图 上的聚类结构越明显， c2 越大，c2 =1时，整个图就断裂成非连通的两块或者多块。
  - c2越大，越容易对这个图上的点进行聚类。机器学习中一个重要课题叫做聚类，近十年 来基于代数图论发展出来的一种新的聚类方法，就是利用了第二大特征值对应的谱结构，这种聚类方法叫做谱聚类(Spectral Clustering)。
  - 如果图上的点分成几组，各自聚成一团，缺乏组与组之间的联系，那么这种结构是很不利于扩散的。在某些情况下，甚至需要 O(exp(N))的时间才能收敛。这也符合我们的直观想象，好比两个大水缸，它们中间的只有一根很细的水管相连，那么就需要好长时间才能达到平衡。有兴 趣的朋友可以就这个水缸问题推导一下，这个水缸系统的第二大特征值和水管流量与水缸的容积的比例直接相关，随比例增大而下降。

图-谱-马尔可夫过程-聚类结构

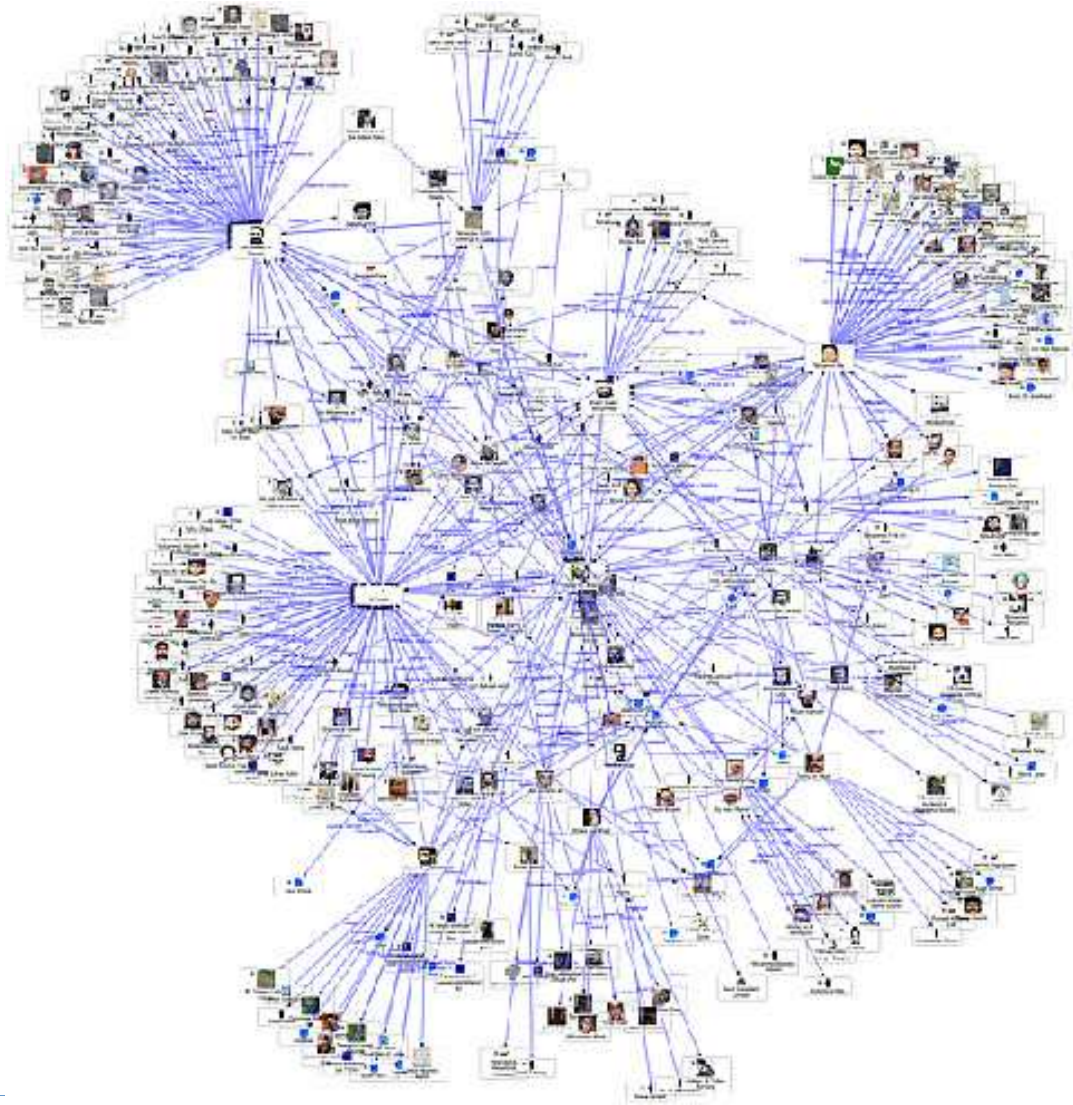
- 总结
  - 图是表达事物关系和传递扩散过程的重要数学抽象
  - 图的矩阵表达提供了使用代数方法研究图的途径
  - 谱，作为一种重要的代数方法，其意义在于对复杂对象和过程进行分解
  - 图上的马尔可夫更新过程是很多实际过程的一个重要抽象
  - 图的谱结构的重要意义在于通过它对马尔可夫更新过程进行分解分析
  - 图的第一特征值对应于马尔可夫过程的平衡状态，第二特征值刻画了这个过程的收敛速度（采样的效率，扩散和传播速度，网络的稳定程度）。
  - 图的第二特征分量与节点的聚类结构密切相关。可以通过谱结构来分析图的聚类结构。
  - **马尔可夫过程代表了一种时间结构，聚类结构代表了一种空间结构，“谱”把它们联系在一起了，在数学刻画了这种时与空的深刻关系。**

Web结构

```
<!-- HEADER -->
<div id="header">

    <h1><a href="index.html" onclick="return link(th

    <a class="header-button toggle">Menu</a>

    <!-- NAVIGATION -->
    <div id="nav">

        <ul>
            <li><a href="index.html" onclick="return
            <li><a href="about.html" onclick="return
            <li><a href="portfolio.html" onclick="re
            <li><a href="gallery.html" onclick="retu
            <li><a href="contact.html" onclick="retu
            <li><a href="styles.html" onclick="retur
        </ul>

    </div> <!-- /nav -->
```



哪个网页最重要？



节点重要度计算（一）

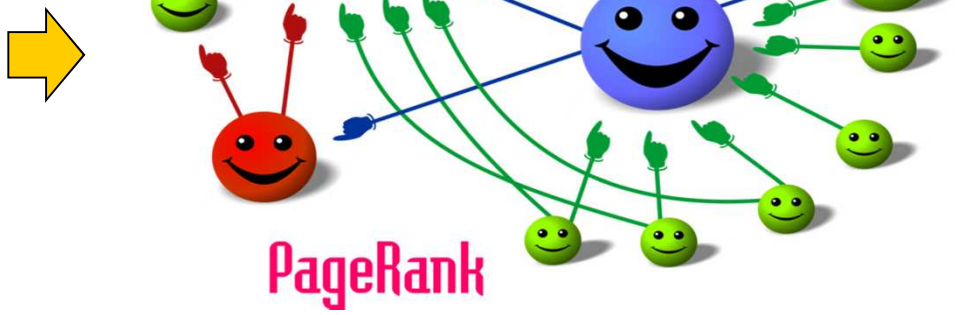
- 导出链接（出度）
  - 一个网页指向其它网页的链接数越多，该网页就越重要？

节点重要度计算（二）

- 导入链接（入度）
  - 指向一个网页A的网页数越多，则网页A就越重要？

节点重要度计算（三）

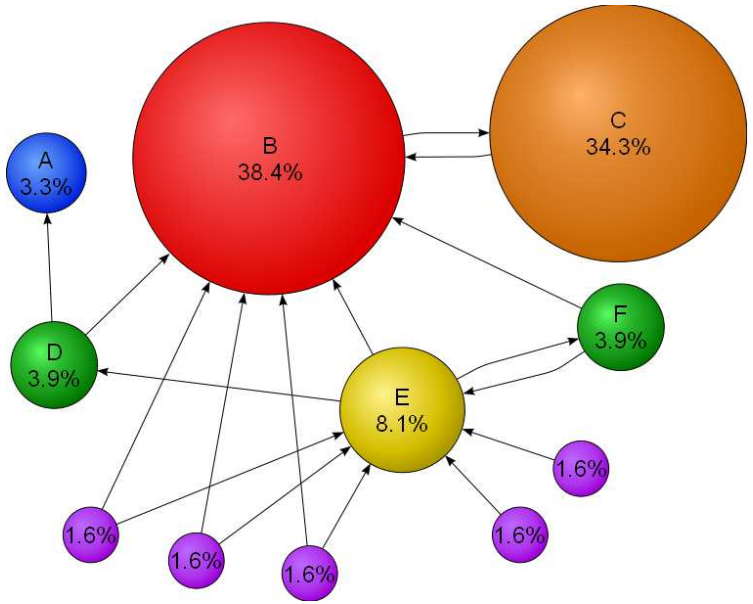
- PageRank方法
  - 导入页面越多，并且导入页面越重要，当前页重要度就越高？



网页重要度计算之PageRank思想

- 思想
  - 若 B 网页上有连接到 A 网页的链接 说明 B 认为 A 有链接价值，是一个“重要”的网页。当 B 网页级别 (重要性) 比较高时，则 A 网页可从 B 网页这个导入链接分得一定的级别 (重要性)，并平均分配给 A 网页上的所有导出链接
- 一个页面P的Rank值r(P)的大小取决于三个因素：
  - 链入网页数
  - 链入网页的质量
  - 链入网页的链出网页数

链向网页E的导入链接数远大于链向网页C的链接，但是网页C的重要性却大于网页E。这是因为因为网页C被网页B所链接，而网页B有很高的重要性。



PageRank 算法

- 设 u 是某个网页，其级别（重要性）为 $r(u)$，记 F_u 为 u 的导出链接的集合，B_u 为 u 的导入链接的集合，$n_u = |F_u|$ 即是 u 的导出链接总数。
- 设 v 是 u 的一个导入链接，根据 PageRank 理论，u 从 v 处分得的级别（重要性）为 $r(v)/n_v$。将 u 从所有导入链接处分得的重要性相加，即为网页 u 的最终级别

$$r(u) = \sum_{v \in B_u} \frac{r(v)}{n_v}$$

PageRank 的数学模型

- 设共有 m 个网页，分别编号为 1、2、3、…、m，它们的级别（重要性）分别记为 r_1、r_2、r_3、…、r_m，G 表示由这些网页组成的有向图的邻接矩阵。根据有向图理论：

$$r(u)=\sum_{v\in B_u}\frac{r(v)}{n_v} \quad \Rightarrow \quad r_i=\sum_{j=1}^{m}\frac{g_{ij}}{n_j}r_j$$

n_j：G 中第 j 列的列和

矩阵形式

$$r=G_m\cdot r$$

其中 $r=(r_1,r_2,\ldots,r_m)^T$，$G_m=\{g_{ij}/n_j\}$

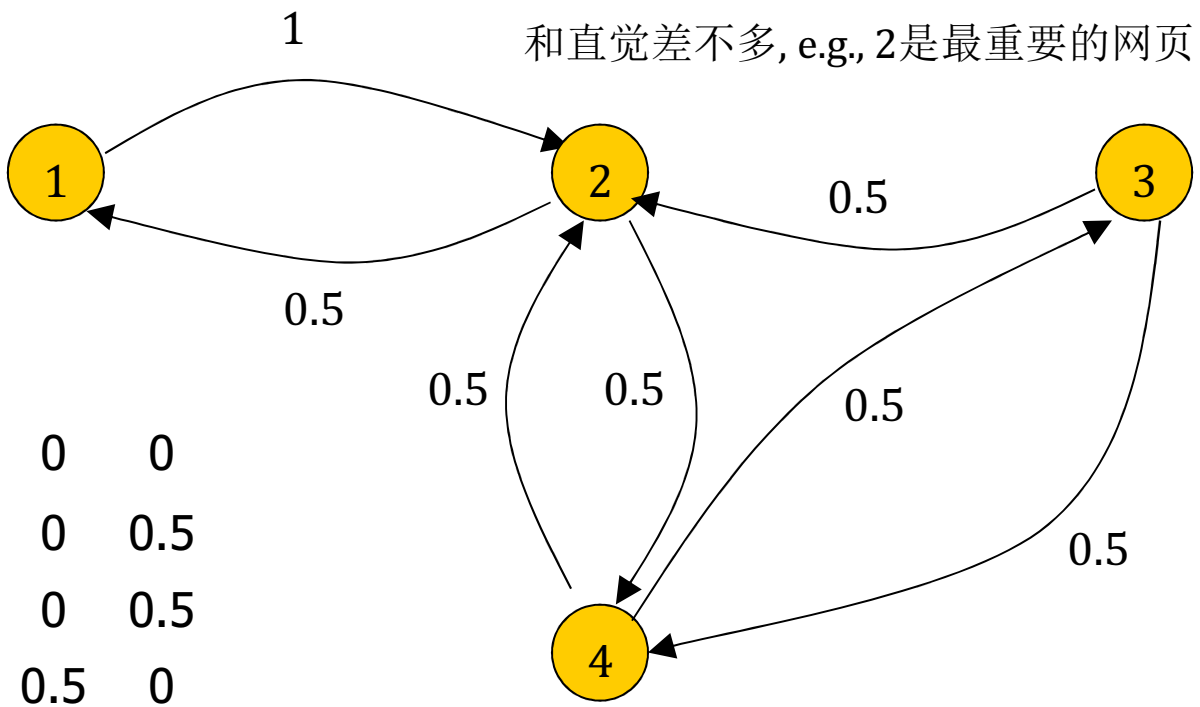
- 可知 r 是 G_m 的对应于特征值为 1 的特征向量

计算举例

W的特征向量为

$\underline{r}$ = [0.2 0.4 0.133 0.2667]

和直觉差不多, e.g., 2是最重要的网页



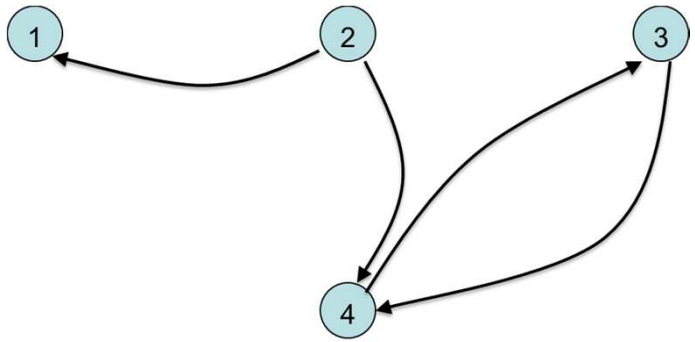
G

0	1	0	0
0.5	0	0	0.5
0	0.5	0	0.5
0	0.5	0.5	0

这么快就搞定了?

PageRank的问题—矩阵特性

- 矩阵特性
  - 必须保证矩阵是不可约的（强连通、非周期）
  - 如右图所示，由于1没有链出页面，3和4形成一个周期，所以该矩阵是可约的，不存在特征值1对应的特征向量



矩阵 G_m 一定有特征值 1 吗？即上面的方程是否有解？

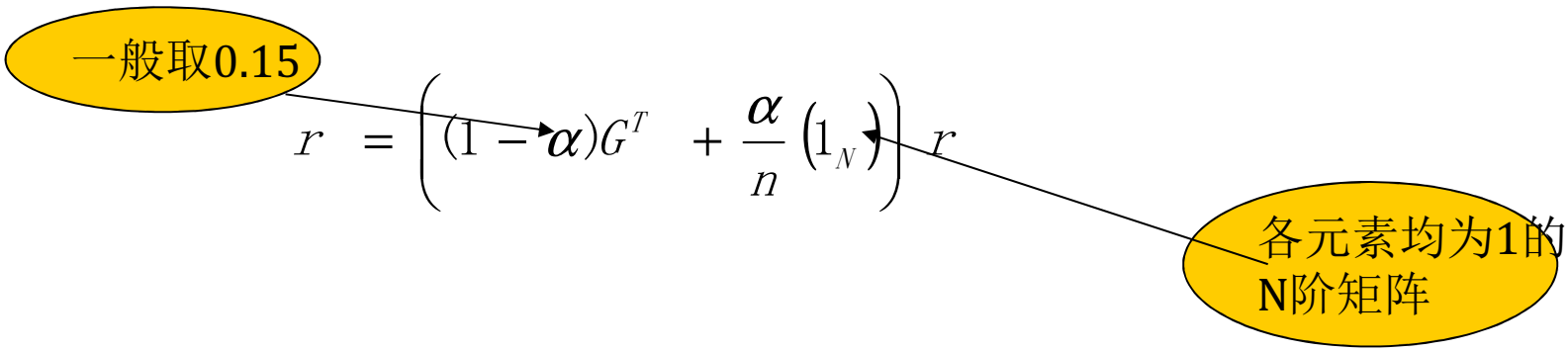
如果 $G=\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}$，则 $r_1=r_2$ ，此时就无法进行求解

PageRank问题之理解

- PR的随机游走模型
  - 从浏览者的角度理解PageRank
  - 设想有一个永不休止浏览网页的人，每次随机选择一个链出的链接继续访问。我们问，在稳态情况下（足够长时间后），他会正在看哪一篇网页呢？
  - 等价于：稳态情况下，每个网页v会有一个被访问的概率，p(v)，它可以作为网页的重要程度的度量，依赖于上一个时刻到达“链向”v的网页的概率，以及那些网页中超链的个数。
- 矩阵特性问题
  - 当浏览者所浏览的网页矩阵存在不可达或周期连通分量时，该浏览者将无法继续浏览其它网页
  - 矩阵是可约的

PageRank改进

- PR修正模型
  - 让浏览者每次以一定的概率（1-α）沿着超链走，以概率（α）重新随机选择一个新的起始节点
  - α选在0.1和0.2之间，被称为阻尼系数
  - **$M=(1-\alpha)G^T+\alpha/N(1_N)$**被称为Google Matrix



PageRank修正算法

- 问题转化为求解方程组r=G·r，且r满足 $\sum_{i=1}^{n} r_i = 1$
  - p有稳态解，需要G满足两个条件
    - 强连通
    - 非周期
  - 这两个条件都可以通过增加的阻尼系数，即以概率β重新随机选择一个新的起始节点继续进行游走，来得到实现。
- 根据Perron-Frobenius定理

$$\boldsymbol{x = A x}\text{，}\ \boldsymbol{x}\text{满足：}\sum_{i=1}^{n} x_i = 1$$

  - 该方程组解存在且唯一
    - 而且是A的最大特征值1所对应的特征向量

PageRank的计算问题

- 计算问题
  - 特征向量计算规模是$O(n^3)$
  - 互联网的页面数以百亿计，本实验中是16万个，大规模矩阵的向量计算用普通方法不可行
- 解决方法-Power Iteration幂迭代

当矩阵 A 的阶很大，无法直接计算其特征值和特征向量时，需要使用该方法

1) 输入矩阵 A 和迭代初始向量 v，以及精度 $\varepsilon > 0$ 例如0.0001，向量各元素对应差值绝对值），令 $k = 0$;

2) 计算：$v_{k+1} = Av_k$;

3) 如果 $|v_{k+1} - v_k| < \varepsilon > 0$ ，则计算 PageRank 值并停机。否则转第二步。

$$x = A^k v / sum(A^k v)$$

$A^k v$ 即 v_k

PageRank算法只有两步：构造矩阵A和迭代求解

幂迭代计算

- 幂迭代算法
  - 确定合适的初始向量v，例如v= indegree(node)/|E|
  - 每次迭代是一次矩阵向量乘法复杂度>$O(n^2)$，但A是稀疏矩阵，所以整个迭代速度非常快
  - 收敛速度取决于c2
  - 3亿个页面的Web Graph
    -> 50 iterations to convergence (Brin and Page, 1998)
- 幂迭代算法的马尔科夫链模型
  - **page importance ⇔ steady-state Markov probabilities ⇔ eigenvector**
  - **Larry Page和Sergey Brin 的贡献，一方面加入随机游走解决了矩阵收敛问题，另一方面由于互联网网页数量巨大，生成的二维矩阵巨大，两人利用稀疏矩阵计算简化了计算量。**

PageRank计算举例

- 页面框架图
  - 先建立一个网页间的链接关系的模型,即我们需要合适的数据结构表示页面间的连接关系。
  - 首先我们使用图的形式来表述网页之间关系：
  - 现在假设只有四张网页集合：A、B、C，其抽象结构如下图1：

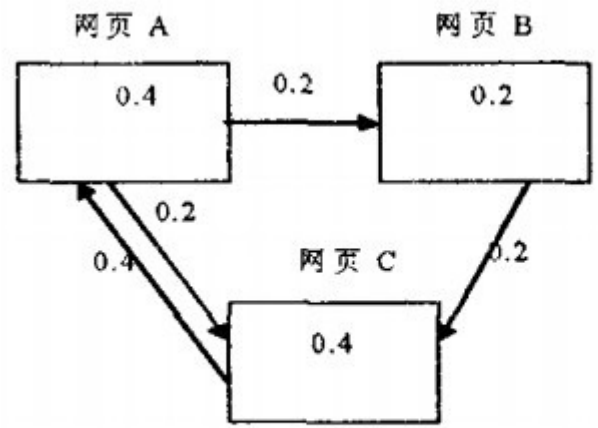


图1 网页间的链接关系

计算概率转移矩阵

- 用矩阵表示连通图：
  - 用邻接矩阵 P表示这个图中顶点关系，如果顶（页面）i向顶点（页面）j有链接情况，则pij = 1，否则pij = 0。如图2所示。如果网页文件总数为N，那么这个网页链接矩阵就是一个N x N 的矩 阵 。
- 网页链接概率矩阵
  - 然后将每一行除以该行非零数字之和，即（每行非0数之和就是链接网个数）则得到新矩阵P’，如图3所示。 这个矩阵记录了 每个网页跳转到其他网页的概率，即其中i行j列的值表示用户从页面i 转到页面j的概率。图1 中A页面链向B、C，所以一个用户从A跳转到B、C的概率各为1/2。
- 概率转移矩阵P
  - 采用P’ 的转置矩 阵进行计算， 也就是上面提到的概率转移矩阵P 。

$$P=\begin{array}{c} \\ A \\ B \\ C \end{array}\begin{array}{c}\begin{array}{ccc} A & B & C \end{array}\\ \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}\end{array}$$

图2 网页链接矩阵：

$$P'=\begin{bmatrix} 0 & 1/2 & 1/2 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

图3 网页链接概率矩阵：

$$P'^{T}=\begin{bmatrix} 0 & 0 & 1 \\ 1/2 & 0 & 0 \\ 1/2 & 1 & 0 \end{bmatrix}$$

图4 P’ 的转置矩 阵

计算google Matrix

1）P概率转移矩阵 ：

$$P=\begin{bmatrix}0 & 0 & 1\\ 1/2 & 0 & 0\\ 1/2 & 1 & 0\end{bmatrix}$$

2）ee^t/N为：

$$ee^t/N=\begin{bmatrix}1/3 & 1/3 & 1/3\\ 1/3 & 1/3 & 1/3\\ 1/3 & 1/3 & 1/3\end{bmatrix}$$

3）A矩阵为：$q \times P + (1-q) * ee^t/N = 0.85 \times P + 0.15 * ee^t/N$

$$A=\begin{bmatrix}0.05 & 0.05 & 0.9\\ 0.45 & 0.05 & 0.05\\ 0.45 & 0.9 & 0.05\end{bmatrix}$$

初始每个网页的 PageRank值均为1 ， 即X~t = (1 ， 1 ， 1) 。

迭代计算PageRank

第一步：

$$AX=\begin{bmatrix}0.05 & 0.05 & 0.9\\ 0.45 & 0.05 & 0.05\\ 0.45 & 0.9 & 0.05\end{bmatrix}*\begin{bmatrix}1\\1\\1\end{bmatrix}=\begin{bmatrix}0.05+0.05+0.9\\ 0.45+0.05+0.05\\ 0.45+0.9+0.05\end{bmatrix}=\begin{bmatrix}1\\0.55\\1.4\end{bmatrix}=R$$

因为X 与R的差别较大。 继续迭代。

第二步：

$$AX=\begin{bmatrix}0.05 & 0.05 & 0.9\\ 0.45 & 0.05 & 0.05\\ 0.45 & 0.9 & 0.05\end{bmatrix}*\begin{bmatrix}1\\0.55\\1.4\end{bmatrix}=\begin{bmatrix}1.337\\0.455\\1.109\end{bmatrix}=R$$

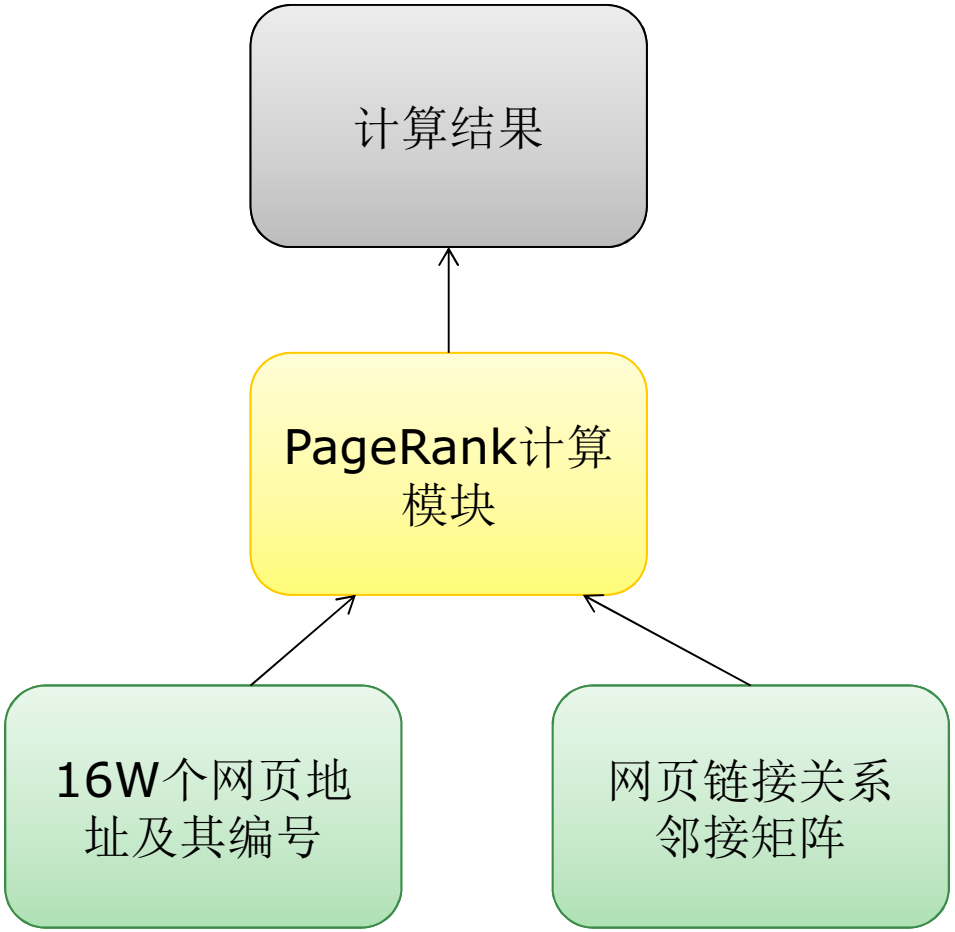
继续迭代这个过程...

直到最后两次的结果近似或者相同，即R最终收敛，R 约等于X，此时计算停止。最终的R 就是各个页面的PageRank 值。

用幂法计算PageRank 值总是收敛的，即计算的次数是有限的。


1、实验任务
2、相关基础原理
3、分析器设计与实现
4、实验说明

链接分析器总体设计



稀疏矩阵

- 概念
  - 简单说，设矩阵A中有s个非零元素，若s远远小于矩阵元素的总数（即s≦m×n），则称A为稀疏矩阵。
  - 精确点，设在矩阵A中，有s个非零元素。令 e=s/(m*n)，称e为矩阵的稀疏因子。通常认为e≦0.05时称之为稀疏矩阵。
- 存储方式
  - 在存储稀疏矩阵时，为了节省存储单元使用压缩存储方法。
  - 非零元素的分布一般没有规律，存储非零元素的同时，还必须同时记下它所在的行和列的位置（i,j)。
  - 两种存储方式：
    - 三元组(i,j,a_{ij})唯一确定了矩阵A的一个非零元。因此，稀疏矩阵可由表示非零元的三元组及其行列数唯一确定。
    - 十字链表方法，矩阵的每一个非零元素用一个结点表示， 该结点除了（row，col，value）以外，还要有以下两个链域： right：用于链接同一行中的下一个非零元素；down： 用于链接同一列中的下一个非零元素。

稀疏矩阵的三元组存储

```
#define  MAXSIZE 1000    /*非零元素的个数最多为1000*/
#define  MAXROW 1000    /*矩阵最大行数为1000*/
typedef struct
    {
        int    row,  col;    /*该非零元素的行下标和列下标*/
        ElementType  e;      /*该非零元素的值*/
    }Triple;
typedef struct
{
     Triple   data［MAXSIZE+1］; /* 非零元素的三元组表，data［0］未用*/
     int     first［MAXROW+1］;     /* 三元组表中各行第一个非零元素所在的
位置 */
     int     m,  n,  len;/*矩阵的行数、 列数和非零元素的个数*/
}TriSparMatrix;
```

三元组存储示例

所示的稀疏矩阵的三元组的表示如下：

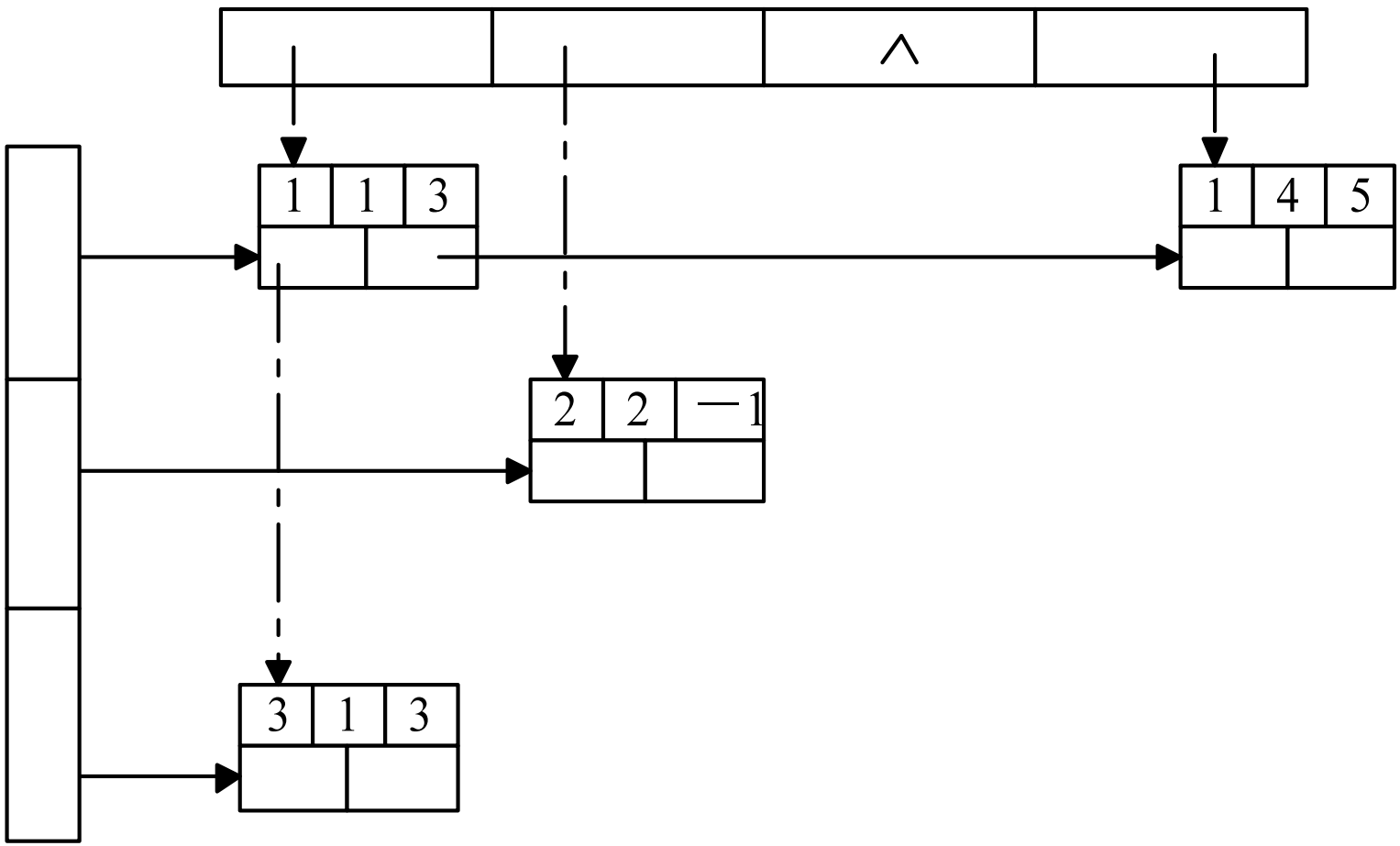
$$\begin{pmatrix} 0 & 12 & 9 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -3 & 0 & 0 & 0 & 0 & 14 & 0 \\ 0 & 0 & 24 & 0 & 0 & 0 & 0 \\ 0 & 18 & 0 & 0 & 0 & 0 & 0 \\ 15 & 0 & 0 & -7 & 0 & 0 & 0 \end{pmatrix}$$

i	j	v
1	2	12
1	3	9
3	1	-3
3	6	14
4	3	24
5	2	18
6	1	15
6	4	-7

十字链表存储结构

十字链表的结构类型说明如下：

```
typedef struct OLNode
{
    int             row,  col;        /* 非零元素的行和列下标 */
    ElementType     value;
    struct OLNode   * right, *down;   /* 非零元素所在行表、列表的后继链域 */
}OLNode;  *OLink;
typedef struct
{
    OLink  * row_head,  *col_head;    /* 行、 列链表的头指针向量 */
    int    m,  n,  len;  /* 稀疏矩阵的行数、 列数、 非零元素的个数 */
}CrossList;
```



十字链表的结构


1、实验任务
2、相关基础原理
3、分析器设计与实现
4、实验说明

实验说明

- 程序数据来源
  - 大约16万个网页
  - 其链接地址已保存到本地文件
- 运行要求
  - ./linkanalyzer urls.txt matrix.txt result.txt
  - result.txt：显示PageRank最大的前5个URL地址
  - 格式：第一列pagerank值，第二列url
- 参数设计建议
  - PageRank的α取值0.15，迭代精度　取值0.0001


THE END