# 从马尔可夫链到Google Page Rank算法优化设想

## 学生信息：

姓名：叶伟堂

学号：2021201613

班级：实用算法与程序设计2班

年级：2021级

专业：计算机科学与技术（图灵）

## （一）摘要与关键词

摘要：本文通过研究Google链接搜索算法的底层模型：马尔科夫链，探究了该算法的数学基础，概括了Google Page Rank中的过程和思路。基于马尔可夫链转换矩阵的概念，本文提出了矩阵快速幂算法可以对转换矩阵迭代过程进行加速，以及为了更加精确拟合现实模型，提出了用户状态变化（本文主要考虑的是用户连续访问相同页面导致的兴趣衰减）造成的转换矩阵动态变化的问题。针对这一问题，本文设想了一种算法，利用迭代单元整合一定次数的迭代，在每个迭代单元后利用衰减矩阵来更新转换矩阵，这样不会过多地破坏矩阵快速幂算法的连续性，以此来实现对算法效率和现实模型的兼顾。

## （二）研究背景与动机

### 1. 搜索引擎发展简史

搜索技术的发展大致可以分为三个阶段：

* 分类目录时期
* 文本检索时期
* 链接分析时期
* 以用户为中心的时期
* 分类目录时期是在互联网诞生的早期，搜索引擎还处于原始时期：hao123和Yahoo是那是人们熟悉的两个搜索引擎，它们通过人工收集和整理，把属于各个门类的不同优质网站进行过罗列，减少用户筛选网站的时间。
* 文本检索时期使使用了一系列信息检索的模型，分析网页文本内容和用户查询词之间的相关程度，后来人们又不断研究新的方法来降低时间复杂度。但是在搜索效果上仍有突破空间。
* 链接分析时期的搜索引擎建立在文本检索的基础上，通过深入地分析网页中链接所隐含的信息，获取网页之间的“推荐关系”，通过对于链接地分析能够为用户推荐相关地网页，并进行精准地排名，极大改善了搜索质量和搜索体验。
* 以用户为中心时期的搜索引擎大量运用了及机器学习技术，使得搜索的结果更加个性化。
* 值得一提的是，上述这些技术实际上经过长远的发展早已紧密相连，而不是更新替代，即使在机器学习算法层出不穷的时代，链接分析技术仍然有深入研究和改善的价值。
* 
* Google Page Rank算法发明者：Sergey Brin和Lawrence Page，当时二人是斯坦福大学研究生

### 相关工作

近年来对于链接搜索领域的研究仍然存在，机器学习事实上是链接搜索的补充，而不是替代——机器学习方法的出现为链接搜索领域提供了新的思路。工程师们也开始兼顾如何更好地用算法拟合现实模型和如何加速算法两个问题。本文基于Google提出的链接搜索模型，研究了这个模型的原理，并提出了一些加速算法和拟合现实情形的其它设想。

## 知识基础

### 随机游走模型

给定一个含有n个结点的有向图，在有向图上定义随机游走（ random walk）模型：图的每个节点结点表示一个特定的状态，有向边则用来表示状态之间的转移，假设从一个结点到通过有向边相连的所有结点的转移概率相等。具体地，转移矩阵是一个n阶矩阵M，M的第i行第j列的元素定义为从状态i向状态j的转移方向。事实上，我们总能得到以下结论：

 式（1）

这个模型就被称为随机游走模型，随机游走模型是马尔可夫链的基础。

### 2. 马尔可夫链

#### 2.1 马尔可夫链概念介绍

马尔可夫链是一种假设某一状态转移的概率只依赖于它的前一个状态。用数学语言可以表达为： 式（2）

概括而言，马尔可夫链应该具有核心三要素

* 状态空间（State Space）：状态只能在若干个状态间进行转换
* 无记忆性（Memorylessness）：当期选择的概率只受到上期状态的影响（即上述数学公式的涵义）
* 转移矩阵（Transition Matrix）： 我们可以使用一个矩阵来描述一个状态转移到所有状态（包括自身状态）的概率，如果不存在转移到某状态的路径，则直接用 0 表示其概率。
  + 有向图的表示方法有邻接矩阵和邻接表，而实际上这里的转移矩阵的实质是一种带权的邻接矩阵
  + 同理，转移矩阵不利于描述较疏的图，会照成空间复杂度的浪费，因此我们也可以使用“转移表”来表示马尔科夫链

#### 2.2 马尔可夫链的稳态

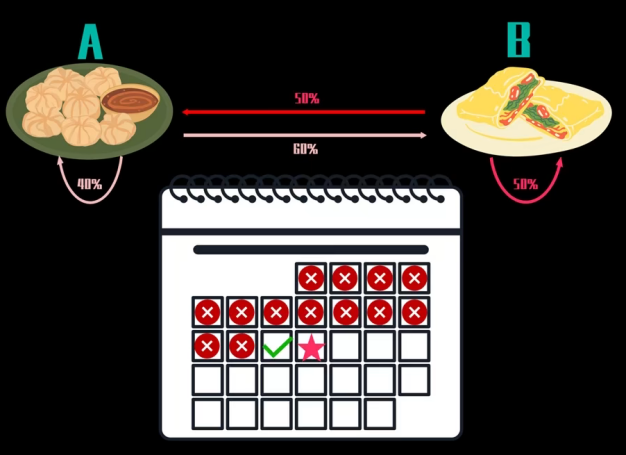
符合上述要求的马尔可夫链都会达到一个最终稳定的状态，如果我们将初始状态整合为一个状态向量，那么，每次得到的下一个状态向量的各状态概率可以用下式表示

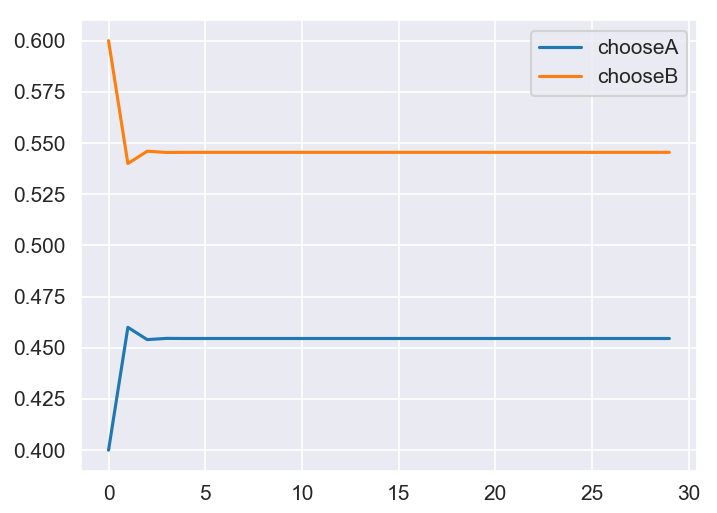
 式（3）

其中表示第j个状态经过i次迭代后得到的状态，表示第a个状态转移到第b个状态的概率。

我们可以通过一个简单的例子描述这样的模型：

W每天选择早餐时总是进行两种选择：A或B，在选择A后依旧选择A的概率为40%，在选择A后选择B的概率为60%，在选择B后依旧选择B的概率为50%，在选择B后选择A的概率为50%。这样我们统计W每天的早餐选择状态，会发现W每天的早餐选择倾向于B，并且概率为54.5455%，趋近于某种稳态。

 图（1）

 图（2）

#### 2.3 马尔可夫链的数学性质

除此之外，对于马尔可夫链，我们还可以发现这样的数学性质：如果我们经过足够多次的转移矩阵T迭代，得到的稳态矩阵为A，那么实际上由以下两式可知A是转移矩阵的特征向量。在这里我们当然可以借助高等代数的知识进行长篇大论的推演（应该会非常有趣），可惜这是一篇算法论文，如何借助这个模型来实现一些服务于现实世界的程序，才是本文的重点部分。

 式（4）

## Page Rank技术

### Page Rank的基本定义

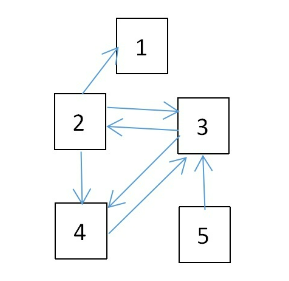
由上面对于马尔可夫链的讨论可知，对于含有n个节点的满足马尔可夫性质的有向图，我们在图上定义一个随机游走模型：对于随机游走模型而言，从图上一个节点转移到与该节点连通的任何一个节点的概率相等。根据这样的结果，我们不难直接定义出转移矩阵T，可以证明，这种情况下，马尔可夫链也是具有最终平稳分布R的，这样，只要得到最后的平稳分布，并且将平稳分布进行模长化（满足概率和为1的分布要求），可以得到：

  式（5）

这是Page Rank中的基础模型，现实场景下，为了模拟复杂的情况，我们还需要对Page Rank模型进一步修正。

### Page Rank算法模型

为更加清晰地解释下文的内容，我们利用一个简单的网站链接模型作为例子。如图（3），有五个网站，它们之间的链接关系如图所示

图（3）

#### 基于现实因素考虑的链接有效度

在现实场景的各种复杂因素影响下，我们会发现上述Page Rank的基本算法模型中事实上并不完全拟合现实中的情形：例如一个网站的访问量为1000，另一个网站的访问量为10，这两个网站的推荐效力很明显是不同的。

另外，不同网站的推荐效力也受到用户主观因素的影响：当下的“以用户为中心”的时代，机器学习等技术的兴起使得网站服务提供商有机会为用户“画肖像”，不用页面标题对于不同用户的吸引程度也是不同的，这就为转移矩阵的确定又增加了一层变量。

原有基于随机游走的模型应该被我们更新，现在，假设我们已经充分考虑了各种要素（事实上转化矩阵的获取方式并非本文的主要内容），得到一个新的转移矩阵M，它满足我们定义的马尔科夫性，同时又能够考虑到各种现实因素的影响。后面我们的讨论中都将基于这个转移矩阵M展开。

 式（6）

#### 关于链接有效度的再探讨：动态链接有效度

动态链接有效度是另一个拟合现实情形的方法：

用户的网页浏览意向事实上不完全由已经预设好的前置信息所决定：目标用户浏览网页的顺序，用户重复浏览同一个网页的次数等信息实际上都会对用户浏览网页的意向产生影响。本文考虑的主要是人们在连续访问网页后会产生一定的兴趣衰减，进而对转换矩阵的数值产生的影响，这也是属于动态链接有效度问题的一个范畴。

动态链接有效度虽然一定程度上提高了结果精确程度，但是由于每次迭代时，转换矩阵并不相同，因此会牺牲一些矩阵加速运算的特点。如何在考虑动态有效度的同时能够利用加速矩阵运算的方法成为本文研究的一大挑战。

本文提出的一种基于矩阵快速幂运算的转换矩阵迭代加速方法，同时提出了一种考虑动态信息对链接有效度影响并防止损失迭代速度的方法——引入”衰减矩阵“概念并利用它更新动态链接有效度。在后文中会对这些进行详细论述。

#### 2.3 规避转移到一个网站后无法继续转移的情况

如果进入到一个网站中，该网站不具有到任何其它网站的链接，那么一旦陷入这样的状态就会被当前页面局限而不再发生转移，这显然不是我们希望的状态。这种情况在转换矩阵中体现为：一列中只有对角线元素为1而其它元素为0。因此我们可以将这种不具备链接的网站近似成具备到所有其它网站的等效链接的网站，即对矩阵的相关维的分量进行修正。如矩阵的第一维分量 则，可以修正。同理，我们刚刚定义的转换矩阵应该修改为：

 式（7）

#### 2.4阻尼系数的引入

用户在点击进入网站时，除了使用通过链接到达其它相关的网站，还有另一种选择——就是直接通过引擎转到当前网页未链接的网站，这种情况在我们之前的模型中是一直缺乏考虑的。

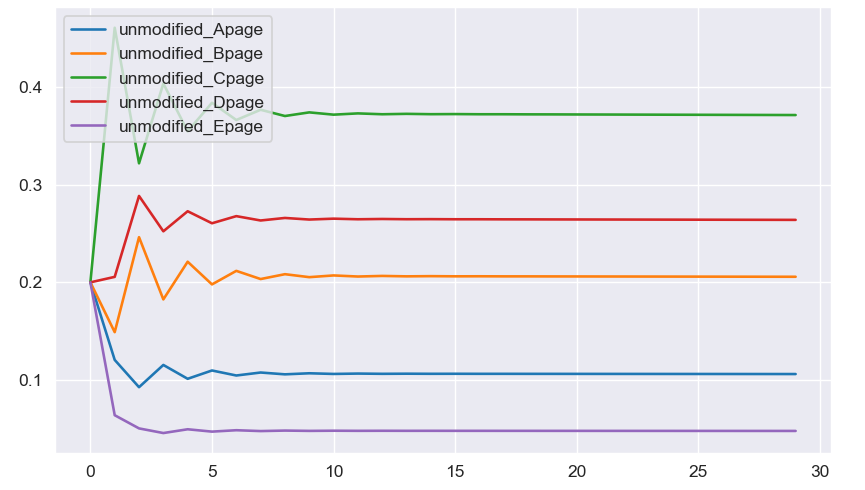
我们引入一个”阻尼系数“d，将它定义为用户通过链接跳转的概率，那么1-d就是用户直接通过搜索引擎跳转的概率，我们规定用户通过非链接方式前往各个网站的概率相等，那么新的转换矩阵S应该满足下面的公式

 式（8）

其中，K表示每个元素都是1的矩阵。Google的工程师统计得到的d取0.85。

如下，我们可以利用python求出引入阻尼系数以后得到的转换矩阵的值：

经过上述过程，我们定义一个合适的迭代次数（采用极限的方法定义迭代次数），经过这些迭代之后，得到的这几个网页的排名数据，下面（图4）展示了添加阻尼系数后的可视化数据。

图（4）

### Page Rank算法优化

#### 优化思路

* 对Page Rank算法进行速度优化，可以从以下思路开展
* 在不影响结果的情况下，减少转换矩阵迭代次数，加快迭代速度
* 在不影响迭代结果的情况下，近似转换矩阵迭代的终止条件
* 上述方法中，本文利用矩阵快速幂算法，影响了第一个过程，实现算法速度的提升。而前文提到的”利用‘衰减矩阵’概念更新动态链接有效度以达到不损失算法速度的目的“也会在下文一并提出，作为解决相关挑战的一个设想。

#### 矩阵快速幂算法

矩阵快速幂算法和普通快速幂算法逻辑上基本一致，本质是任何数字都可以使用二进制数进行表示，可以分为三步进行：

* 写出矩阵乘法的运算代码
* 将矩阵的次数转化为二进制数
* 求解得到的结果
* 代码设计思路可以采用递归或迭代的方式进行矩阵快速幂的算法设计。
* 在马尔科夫链迭代的过程中，需要大量进行转移矩阵的乘法计算，会带来极大的计算压力。而矩阵快速幂巧妙地减少转换矩阵迭代次数，加快迭代速度，是提高性能的方法之一。
  1. 衰减矩阵
* 挑战与措施

如前文所述，本文设计衰减矩阵的目的是为了更好地拟合现实场景中人们访问网页的情形：人们在连续访问网页后会产生一定的兴趣衰减，进而对转换矩阵的数值产生影响（就是前文提到的动态链接效果）。

然而，在常规算法中，如果在每次进行一次状态更迭时即进行一次链接效果更新计算，这样会导致每次更新后得到的转化矩阵都不一致，带来的影响就是原本用于加速迭代过程的

因此我们根据矩阵快速幂运算的特点，提出这样的计算思路：划分出迭代单元，每个迭代单元中进行一定次数的迭代运算，在期间统计用户访问的网站，用户在访问同一个网站后带来了一定的兴趣衰减，我们并不直接更新衰减的结果，而是在一次迭代单元后才对衰减结果进行分析。这样，在单个迭代单元内依旧可以使用矩阵快速幂算法，可以降低由于考虑动态链接效果带来的算法速度下降问题。

* 程序算法设计

1. 单元迭代的设计：封装矩阵快速幂算法，迭代一定次数。在单元迭代之后及时进行衰减处理，然后用状态矩阵与衰减后的转换矩阵作乘。
2. 衰减矩阵的设计：综合考虑当前状态中各网页的权重，上次迭代的转移矩阵显示出的“访问概率”对于下次迭代产生的影响。
3. 全过程迭代（全过程迭代可能包含若干次单元迭代）的思路：采用递归算法进行设计
   1. 如果满足需迭代次数大于单元迭代的迭代次数，先进行一次单元迭代，然后递归调用本函数。
   2. 如果需迭代次数不满足大于单元迭代的迭代次数，则直接迭代完毕，不需要调用衰减矩阵进行处理。

## （五）程序代码清单

1. 代码文件1：markov\_try.py 马尔可夫过程最终达到稳态的实验python代码

2. 代码文件2：rank.py 网页排名生成模拟python代码

3. 代码文件3：optimize.cpp 马尔可夫转换矩阵快速幂算法 & 马尔可夫迭代单元与衰减矩阵设计封装C++ 代码

备注：.py代码是实验程序，支持直接运行；C++代码是代码框架，不支持直接运行，需要根据具体情形进一步填充代码内容，但整体上能反映算法设计思路。

## （六）结论与问题

本文从马尔科夫链出发，通过探究该算法的数学基础和概括Google Page Rank中的过程和思路，对于该算法有了比较深入的了解。本文提出的关于加速迭代和更精确拟合现实的算法优化设计其实也建立在这种基础上。

当前工作仍存在的问题是，算法给出的对应代码只是阐述相关设想的架构而非具体工业化的实践代码，所以设想尚未经过实践检验是本文存在的一个问题。

期末作业开始时本来希望以马尔可夫链决策过程强化学习作为选题，但是因为一些原因最后一星期中在更换了选题，导致本篇内容准备略有些仓促，但也算是尽力完成了。

尽管如此，经过一学期的《实用算法与程序设计》课程的学习，对于算法在工业中的运用，一个算法如何从提出到落地，再到切实解决人们日常生活问题的过程都有很深的体会，这次期末作业更是让我深入体会到了一个具体领域中算法的重要意义。