

文章编号: 1001 - 9081(2004)06Z - 0270 - 03

主题爬虫的设计与实现

汪涛^{1,2}, 樊孝忠¹

(1. 北京理工大学 计算机科学与工程系, 北京 100081;
2. 中国人民解放军炮兵学院 三系, 安徽 合肥 230031)

摘要: 文章以主题相关度为核心研究了设计主题爬虫的相关技术, 通过实验给出了具体实现。实验结果充分论证了其可行性与实用性, 为进行主题搜索引擎设计和主题信息采集奠定了良好的基础。

关键词: 搜索引擎; 主题爬虫; 主题相关度; 链接分析; 信息采集

中图分类号: TP393.08 **文献标识码:** A

1 引言

按照信息搜集方法和服务提供方式的不同, 搜索引擎系统可以分为三大类:

目录式搜索引擎 以人工方式或半自动方式搜集信息, 由编辑员查看信息之后, 人工形成信息摘要, 并将信息置于事先确定的分类框架中。信息大多面向网站, 提供目录浏览服务和直接检索服务。

机器人搜索引擎 一个称为蜘蛛 (Spider) 的机器人程序以某种策略自动地在互联网中搜集和发现信息, 由索引器为搜集到的信息建立索引, 由检索器根据用户的查询输入检索索引库, 并将查询结果返回给用户。服务方式是面向网页的全文检索服务。

元搜索引擎 这类搜索引擎没有自己的数据, 而是将用户的查询请求同时向多个搜索引擎递交, 将返回的结果进行重复排除、重新排序等处理后, 作为自己的结果返回给用户。服务方式为面向网页的全文检索。

目前的搜索引擎大多数是面向所有信息的, 可以称之为综合性搜索引擎, 但随着信息多元化的增长, 适用于所有用户的综合性搜索引擎显然已经不能满足特定用户更深入的查询需求, 他们对信息的需求往往是针对受限领域和面向特定主题的, 综合性搜索引擎的召回率和精确率都是很低的。针对这种情况, 需要一个分类精确、数据全面、更新及时的面向主题搜索引擎。

主题爬虫是主题搜索引擎的基础与核心, 本文主要是讨论主题爬虫设计的相关技术, 是作为北京理工大学自然语言处理实验室承担的深圳某投资公司的领域信息评估系统项目的子课题展开研究的, 目的是从网上尽可能多获取受限领域的针对特定主题的信息, 从而为后续的信息评估提供充分的材料。

2 主题爬虫设计方案

2.1 系统组成

主题爬虫的设计是以普通爬虫为基础的, 实际上它是对一个普通爬虫进行功能上的扩充。在对网页的整个处理过程中需要增加模块: 主题确立模块、优化初始种子模块、主题相关度分析模块、排序模块。主题确立模块用于确立爬虫面向的主题; 主题

相关度分析模块用来进行网页主题相关度的计算; 初始种子模块用于生成面向特定主题的较好的种子站点, 使爬行模块能够顺利展开爬行工作; 主题相关度分析模块是主题爬虫的核心模块, 它决定页面的取舍; 排序模块是对页面的最终处理, 给与主题相关页面的价值一个较为全面的评价排序。

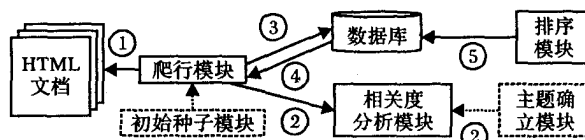


图 1 系统组成

其中, 初始种子模块和主题确立模块是两个辅助模块, 不参与数据流的处理。(1) 爬虫模块取回网页; (2) 调用相关性分析模块, 对网页进行相关性分析; (3) 爬行模块根据分析的不同结果进行相应的处理; (4) 爬行模块从数据库取出等待处理的 URL 继续工作, 循环到第一步, 直至没有新的 URL; (5) 对网页的重要程度进行排序。

2.2 主题确立模块

主题确立是主题爬虫工作的基础, 本文采用的方法是用关键词集来确立主题, 其中每个关键词拥有指定的不同权值。

权值的设置有两种方法: 手工设置和特征提取。特征提取是指给定一个跟主题有关的网页集合, 由程序自动提取这些网页里面共同的特征, 并根据频率确定权值。手工设置的好处是实现简单, 同时人的经验一般比较准确, 跟实际情况不会出现大的偏差, 缺点是可能有缺漏, 权值的量化定义不够精确; 特征提取的优点是权值量化定义精确, 但要求选取用来提取特征的网页集合必须是很具有代表性和全面概括性的, 否则就可能出现很大的偏差。最佳的方法是综合二者的优点:

(1) 手工 (主要是通过咨询领域专家获取) 设置一组关键词并分配权值;

(2) 用这组关键词到元搜索引擎中查找出对应的网页;

(3) 按权值的比例选取一定数量的网页;

(4) 用这些网页组成的集合作为特征提取程序的输入, 得到一组新的关键词及权值。

2.3 优化初始种子模块

由于主题爬虫是面向选定主题的, 所以初始种子的赋予

收稿日期: 2003 - 10 - 09; 修订日期: 2003 - 12 - 07

作者简介: 汪涛 (1977 -), 男, 湖北钟祥人, 博士研究生, 主要研究方向: 计算机网络、Web 信息处理; 樊孝忠 (1948 -), 男, 河南叶县人, 教授, 博士生导师, 主要研究方向: 自然语言处理、数字化网络教学。

应该来自本领域,否则爬虫无法展开爬行工作,具体做法是采用元搜索引擎搜索出的网页,从中选取质量较高的主题 URL,通常由人工完成筛选,这样可信度会更高。

2.4 主题相关度分析模块

为了保证爬虫获取的网页能够尽量向主题靠拢,必须对网页进行过滤,将主题相关度较低的网页(小于设定的阈值)剔除,这样就不会在下一步爬行中处理该页面中的链接。因为一个页面的主题相关度如果很低,说明该网页很可能只是偶尔出现某些关键词,而页面的主题可能和指定主题几乎没有什么关系,处理其中的链接意义很小,这是主题爬虫和普通爬虫的根本区别。普通爬虫是根据设定的搜索深度,对所有链接进行处理,结果返回了大量无用的网页,而且进一步增加了工作量。

主题相关度的计算是采用向量空间模型算法。

把关键词的个数 n 作为向量空间的维数,每个关键词的权值 w_i 作为每一维分量的大小,则主题用向量表示为:

$$= (a_1, a_2, \dots, a_n), i = 1, 2, \dots, n, a_i = w_i$$

对于页面进行分析,统计关键词出现的频率,并求出频率之比,以出现频率最高的关键词作为基准,其频率用 $x_i = 1$ 表示,通过频率比,求出其他关键词的频率 x_i ,则该页面对应向量的每一维分量为 $x_i w_i$,页面主题用向量表示为:

$$= (x_1 w_1, x_2 w_2, \dots, x_n w_n), i = 1, 2, \dots, n$$

用两个向量夹角的余弦表示页面的主题相关度:

$$\cos \langle \cdot, \cdot \rangle = \frac{(\quad)}{\sqrt{w_1^2 + w_2^2 + \dots + w_n^2} \sqrt{x_1^2 w_1^2 + x_2^2 w_2^2 + \dots + x_n^2 w_n^2}}$$

指定一个阈值 r ,当 $\cos \langle \cdot, \cdot \rangle \geq r$ 时就可以认为该页面和主题是比较相关的, r 的取值需要根据经验和实际要求确定,如果想获得较多的页面,可以把 r 设小一点,要获得较少的页面可以把 r 设的大一点。

2.5 排序模块

排序模块的作用是对网页的重要程度进行排序,把价值高的网页排到前面,以便它们更容易地被选择到。影响网页排序的因素很多,通过主题相关度的计算已经能够得到一个比较合理的排序,但是为了得到一个更加合理的排序,必须辅助考虑其他因素。

Web 结构挖掘研究如何利用超链接来对 Web 结构进行挖掘,也即 Web 超链分析,充分利用它,可以极大提高检索结果的质量。考虑网页 p 指向 q ,说明 p 和 q 在某个主题下可能很相关,或者说 p 的作者很认同 q 的内容,基于这种超链分析的思想, Sergey Brin 和 Lawrence Page 在 1998 年提出了 PageRank 算法。算法认为一个网页被多次引用,则它可能是很重要的;一个网页虽然没有被多次引用,但是被重要的网页引用,则它也可能是很重要的,这就是权威(Authoritative)网页,对每个网页计算它的权威值,就可以对网页进行排序,从而找到最重要的权威网页。1998 年 J Kleinberg 提出了 HITS 算法,引入 Hub 网页(提供指向权威网页链接集合的网页),它本身可能并不重要,或者说没有几个网页指向它,但是 Hub 网页提供了指向就某个主题而言最为重要的站点的链接集合。其他一些学者也相继提出了另外的链接分析算法,如 SALSA、PHITS、Bayesian 等算法。这些算法有的已经在实际的系统中实现和使用,并且取得了良好的效果。

在对网页进行排序时,可以综合考虑主题相关度和链接

分析两个关键因素,链接分析主要模仿 PageRank 算法,因为 Google 的成功实践证明 PageRank 算法的结果还是比较令人满意的。具体的算法实现中,应该对主题相关度和 PageRank 赋予不同的权重,则网页的重要程度值可以表示为:

$$P = w_1 \cdot \cos \langle \cdot, \cdot \rangle + w_2 \cdot R(u)$$

其中 $\cos \langle \cdot, \cdot \rangle$ 是上述通过主题相关度模块计算出的主题相关度的大小, $R(u) = d \sum_{v \in B(u)} R(v) / N(v) + (1 - d)$ 是利用 PageRank 算法计算出的可用于页面排序的网页 PageRank 值, w_1 为主题相关度的权重, w_2 为 $R(u)$ 的权重,二者的取值可以根据实验需求选定,必须保证 $w_1 + w_2 = 1$,本算法拟定 w_1 取 0.6, w_2 取 0.4。

下面对 PageRank 算法进行进一步说明,其根本思想是网页的 PageRank 值是一个由网络的超链接结构所产生的一个网页重要性等级值,所有网页的 PageRank 值都可以根据指向它的网页的 PageRank 值和超链接数来计算,即所有链接到它的网页的 PageRank 值除以各自向外的链接数的商进行求和,其对应的直观冲浪模型是冲浪者随机通过点击超链接在网上冲浪,每个网页到达的可能性大小就是网页的 PageRank 值,链接到某网页的超链接越多,到达可能性越大,PageRank 值就越高, u 是一个网页, $F(u)$ 是 u 指向的网页集合, $B(u)$ 是指向 u 的网页集合, $N(u)$ 是 u 指向外的链接数,则有 $N(u) = |F(u)|$;另一方面,由于考虑到实际的冲浪模型中有一种特殊情况,便会不通过所在页面的超链接而跳到一个随机的网页中,引入衰变因子 d (通常取 0.8)就是针对这种情况的,这样网页的 PageRank 值仅 d 部分在它所链接到的网页中分配,剩下的在整个网络的所有网页中分配,最终网页的 PageRank 值可以用 $R(u) = d \sum_{v \in B(u)} R(v) / N(v) + (1 - d)$ 来计算。

3 主题爬虫的实现

1) URL 队列的维护

为了能够方便的处理链接和主题相关度的计算,需要使用 5 个 URL 队列,每个队列保存着同一处理状态的 URL:

等待队列 在这个队列中,URL 等待被爬虫处理,新发现的 URL 被加入到该队列。

处理队列 爬虫开始处理 URL 时,被传送到这一队列,为了保证同一个 URL 不能多次被处理,当一个 URL 被处理过后,被移送到错误队列或者抛弃队列或者完成队列。

错误队列 如果在下载网页时发生错误,它的 URL 将被加入到错误队列,一旦移入错误队列,爬虫不会对它作进一步处理。

抛弃队列 如果下载网页没有发生错误,且经过主题相关度的计算小于阈值,则放入该队列,一旦移入抛弃队列,爬虫不会对它作进一步处理。

完成队列 如果下载网页没有发生错误,如果下载网页没有发生错误,且经过主题相关度的计算大于等于阈值,就要把从中发现的 URL 放入等待队列,处理完毕把它加入到完成队列,到达这一队列将等待排序模块的处理。

同一时间一个 URL 只能在一个队列中,这也叫做 URL 的状态,图 2 说明了这些状态的关系以及网页如何从一个状态转换到另一个状态。

2) 数据库结构

数据库采用 Oracle9i,由以下 7 个表构成:

keywords 记录关键词和权值,由 word 和 weight 两个字段

构成：

- seeds 记录初始种子；
- waiting-queue 记录等待队列，由 id 和 url 两个字段构成；
- processing-queue 记录处理队列，由 id 和 url 两个字段构成；
- error-queue 记录错误队列，由 id 和 url 两个字段构成；
- discard-queue 记录抛弃队列，由 id, url, relativity 三个字段构成，relativity 表示主题相关度；
- complete-queue 记录完成队列，由 id, url, relativity, priority 四个字段构成，priority 表示网页的最终排序值。

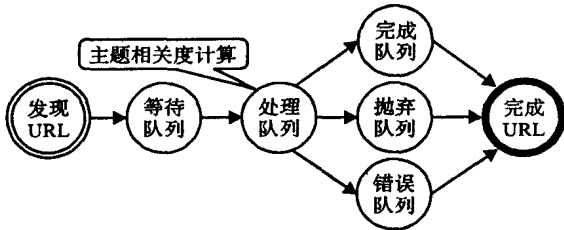


图 2 URL 状态流程图

以上各表只是为了简单实现主题爬虫，所以表的结构比较简单，若要详细记录网页的标题、作者、编码、长度、修改日期等信息，可以对 discard-queue 和 complete-queue 两个表的进行扩充，这些信息在处理网页的过程中很可能得到。

3) 全文索引

本主题爬虫的设计主要目的是获取领域相关信息，为进行领域信息评估提供素材，不需要实现一个完整的搜索引擎。如果为了构建一个完整的搜索引擎，提供全文检索功能，可以直接利用 Oracle9i 数据库提供的全文索引功能对 complete-queue 表的网页建立全文索引。

4) 开发环境

主题爬虫的开发选用 Java 语言，因为 Java 是非常适合为了编程的语言，开发工具选用 Jbuilder9。

5) 部分关键代码段示例

以下代码段是爬虫的页面处理函数 processPage (对于下载出错的网页将放入错误队列，不会被处理到这一步)，它调用相关度计算类进行相关度计算，根据计算的结果和设定阈值进行比较，决定网页取舍，对于大于阈值的网页，将调用 geturl () 函数取出页面中的链接放入等待队列做进一步的爬行，对于小于阈值的页面，放入抛弃队列。

```

package com. bitnlp. crawler;
public class ThemeCrawler
{
    .....
    protected ComputeCorrelativity computing1;
    // 用于相关度计算的类
    .....
    public void processPage (HTTP page)
    // HTTP 是包中定义的类，对网页进行封装
    {
        float i;
        // 用于记录相关度
        i = computing1. analysis (page);
        if (i > = 0. 1)
        {
            geturl (page); // 取出超链，放入数据库等待队列
            complete (page); // 放入完成队列
        }
        else
            discard (page); // 放入抛弃队列
        }
    .....
}
  
```

4 实验结果分析

目前该领域信息评估系统主要是做和监视器相关的信息评估，用主题爬虫和普通爬虫分别进行信息收集，结果表明主题爬虫耗时虽然比普通爬虫长，但差距比预计的要小的多，分析原因可知，尽管主题爬虫要进行比较耗时的主题相关度计算，然而它也带来了正面效应，使爬行的工作量减少了，页面一旦进入抛弃队列将不再被处理，而普通爬虫只会没有选择的对所有页面进行处理，实验结果显示随着搜索深度的增加，主题爬虫的耗时还可能小于普通爬虫。由此可见，用主题爬虫进行面向受限领域信息的收集是切实可行的。另外，主题爬虫的精度和耗时可以通过阈值进行调节，阈值设的越大精度就越高，耗时就越小，因为随着阈值的增大，要求主题相关度增大，需要处理的页面就减少了。部分实验数据如下：

实验条件：搜索深度 = 2 (设的较小，为了防止搜索规模过大)，线程数 = 200 (要求在网络环境较好的情况下)，起始种子 = 10 (都是经过人工选择的较好的种子)，阈值 r = 0. 1，中文分词主要以计算所免费版的分词工具 (C 语言开发) 为基础。机器配置：P4 2. 4C CPU，内存 1G。实验结果如表 1、2 所示。

表 1 主题爬虫

表 2 普通爬虫

名称	总计	名称	总计
提取文档	2568	提取文档	5034
提取失败	436	提取失败	1258
拒绝文档	1	拒绝文档	2
发现文档	3005	发现文档	6294
收集数据总数	93135421 字节	收集数据总数	191008993 字节
总搜索时间	0 12 59	总搜索时间	0 15 46
索引	0 7 8	索引	0 13 8
实际爬行时间	0 5 51	实际爬行时间	0 2 38

5 结语

通过本论文的研究，充分说明了一个主题爬虫设计方案的可行性，以主题爬虫为基础可以开发主题搜索引擎，结合到具体应用，主题爬虫可以在受限领域内进行面向主题的信息采集，为进行评估准备大量的素材。

参考文献

- [1] Heaton J. 网络机器人 Java 编程指南 [M]. 童兆丰, 李纯, 刘润杰, 译. 北京: 电子工业出版社, 2002.
- [2] 厉亮, 等. 主题搜索引擎的探讨 [A]. 李晓明, 李星. 搜索引擎与 Web 挖掘进展 [C]. 北京: 高度教育出版社, 2003. 34 - 40.
- [3] 李盛韬, 等. 主题 Web 信息采集的研究与设计 [A]. 孙茂松, 陈群秀. 语言计算与基于内容的文本处理 [C]. 北京: 清华大学出版社, 2003. 488 - 494.
- [4] Page L, Brin S, Motwani R, et al. The PageRank Citation Ranking: Bringing order to the Web [EB/OL]. http://www. db. stanford. edu/~backrub/ pageranksub. ps January, 1998.
- [5] Brin S, Page L. The Anatomy of a Large - scale Hypertextual Web Search Engine [EB/OL]. http://www7. scu. edu. au/programme/ full-papers/ 1921/ com1921. htm, 1998.
- [6] 曹军. Google 的 PageRank 技术剖析 [J]. 情报杂志, 2002, (10): 15 - 18.
- [7] Oracle Corporation. Oracle9iR2 Text, Technical White Paper [EB/OL]. http://otn. oracle. com/products/text/pdf/ 9ir2text-twp-f. pdf, 2002.