关于复杂智能软件密集型系统国际会议

# 微搜索：一个简单的网页搜索引擎

Andri Mirzal：毕业于日本北海道大学信息科学技术研究生学院

北领地西北9号，札幌060-0814

andri@complex.eng.hokudai.ac.jp

# 摘要

我们描述了一种使用python语言来获取网页内容的搜索引擎。选择python的原因是因为它语法简单明了容易学习 和调试，并且它还有一个有特点是几乎支持所有的操作系统。这个程序的显著特点是它是可调式的，允许用户使用若干组合函数对文档进行排序，在锚文本分析的基础上我们提供了4种额外的计算比分。我们还提供了一个额外的基于链路添加过程的网络算法排名，我们提供合格这个算法的的动机是把页面排序和点击作为一个实验。这个算法是这个本文论述的核心。

# 简介

许多论文都是作为网页信息展示在网上的便于利用检索和利用自己的网络爬虫来抓取内容，包括索引和分析内容，超链接文本，和网络结构的网页，有时也使用搜索函数来返回已经排好序的相关网页便于用户查询。爬虫和搜索功能都被认为是一个搜索引擎的基本组成部分，他们中的每一个都有自己研究挑战和问题。

网络爬虫，也被称为蜘蛛或机器人，是负责获取页面，解析链接，管理爬取队列和索引页面内容的组件。在更复杂的情况下，这个组件还实现了智能策略如：遵守robots.txt指令，不重新下载具有重复的页面以及使用锚点标识了的垃圾邮件内容或具有不同的url但相似的内容。这些策略可以让爬虫避免下载上面描述的不友好的内容，

因此网络宽带和访问资源的过程都能够得到妥善处理。真正运行的爬虫像Mercator,Poly,UbiCrawler,Goolebot,iRobot和IRLbot都显然比这里描述的更复杂。尽管他们中的一些人提供了很好的文件设计并实现了，但是这些信息对可重复生产而言太笼统了没有什么可能行。

对于第二个组件的搜索功能也是一样。从基本从基本得到的搜索功能开始，内容基于只索引和搜索布尔值的分数页面内容与查询内容之间的匹配。更进一步对链接方法进行分析比如PageRank,HITS 和SalSA,然后延伸到更高级的方法，比如锚文本分析和用户点击行为。尽管这些方法都只是关于如何去发现相关网页并根据查询，算法的实际实现，数据库设计，优化问题和内存管理已经变得过于复杂，难以处理。这种情况变得更加困难，事实上网络引擎在大型企业中很有竞争性，所以设计实现真正的web引擎任然存在问题。处在这个公开源代码的时代真的是一件令人失望的事情，软件应该成为教学人类的工具和无限制第传播知识。当然我们可以找到大量的可以在网页中自由构建我们自己系统的代码。实际上，这种努力可以变得更容易，因为在这个领域工作中，已经有很好的开源项目了。一些值得注意的项目是Lemur1和Lucene2.但是由于他们复杂的设计和实现，这些项目仍然不是很好的起点。

这就是开源搜索引擎中项目强调代码和设计需要易于理解的重要性。因为如果学习是简单和有趣的，这个复杂的细节可以稍后介绍。相比较于以前已经做了的，我们想取=去研究一些新的东西，比如实施新的排序算法，在页面上做词干分析内容和超文本链接，并使用新的方法来进行锚文本分析。所有这些都要求我没改变现有的代码并向其添加新函数或模块代码。为了能够提供满足上述条件的软件，我们建议使用Toby Segaren在书中提到过的搜索引擎即编程集体智慧。这个程序，是用Python编写的.

Python是一个免费的开源脚本，它是最近非常流行的语言，它的语法和语义简单，支持的库非常全面，对主要的操作系统都有强大爬虫的支持。这个程序本身包含三个类，爬虫搜索引擎和神经网络。我只会使用，改进和添加新的功能来爬取和搜索，因为我们只对建立搜索引擎抓取网页和提供搜索功能给用户，不训练这些来自用户返回的数据（这需要用户日志数据集）。对那些对神经网络感兴趣的用户而言，建议直接看参数书籍[6]或Bawza-Yates et等人的论文[19]，他们使用搜索日志信息对那些分类 用户进行调查。

# 系统概述

系统顶层设计图如下：

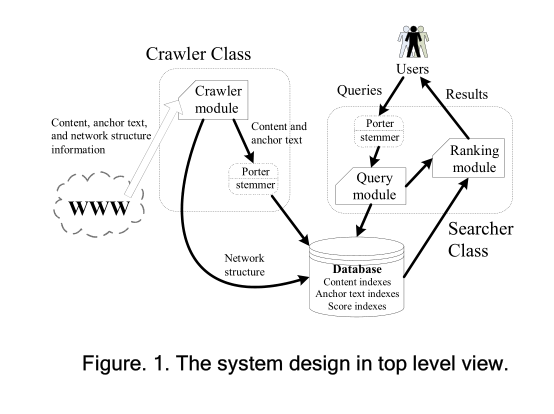
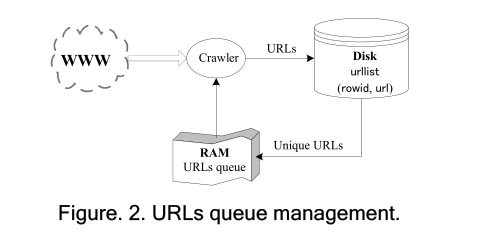


图1显示i了系统的总设计，该设计主要继承了它的原始结构，图2描述了url队列管理爬虫程序从RAM列表中下载url，解析页面上的超链接，并将其发送到磁盘。如果url是惟一的，那么它们将被添加到url列表然后被发送到url队列RAM中。下面的图描述了对原系统的修改。

URLs队列管理



## 2.1提高爬虫的可靠性

在原始代码中，爬虫程序将尝试打开用户提供的URL列表(python集)中的url.

如果失败，爬虫程序将从下一个URL开始。但是问题是即使URL可以打开，但它并不意味着内容可以被阅读。进一步,如果内容可以被读取，有时是索引器爬虫类中的函可能无法输入索引进入数据库中相应的表中(参见图3)，很多情况下，这些错误会导致程序停止工作，这是非常不方便的，因为在网上数据挖掘的能力，继续爬行的过程忽略不可读的页面是一种不可或缺的品质。try-except语句用于打开和读取url，和索引过程，其中包括内容，锚文本和网络结构索引(参见图3)。有了这种简单的方法，我们就可以连续三天没有停止和下载爬取页面74,243页(243mb)，摘自6月份的《学术百科全书4》2008年采用英特尔1.86 GHz处理器笔记本，2GB内存，100 Mbps(下载度:53 Mbps)上传速度:5.5 Mbps)互联网连接。与原系统相比，在爬取的过程中在哪出错就会在哪停止，这是一个很大的进步。我们还根据英语标准停顿字5添加了停止词来减少数据库的存储大小。但是“计算机”一词被列表中删除了，因为我们相信这个词在搜索目的上是有意义的。

## 2.2添加词干分析功能

我们还通过使用porter为两个类的stemmer算法6[20]添加词干分析函数。抑制是一个将词形变化或派生的词还原为它们的过程茎。阻塞过程经常会改进一个红外系统的性能，因为文字具有同一茎通常有相似的意思。阀杆本身不需要与它的形态相同，但一个好的词干分析算法应该返回应该返回词干相同的词表示词干相同的词形态根。例如“navigational”,“navigation”和“navigate”都有词根“navigate”。如果使用porter stemer ,那么将返回navig这些词而不是navigate. 如图1所示，词干分析函数位于内容和锚文本输入数据库爬虫类和查询进入searcher类中的查询模块之前。这个过程将确保单词列表中的所有单词是否以stem形式保存，以及查询中的所有术语将以stem形式输入查询模块。有一件重要的事要记住，如果爬行进程是在阻塞函数存在的情况下完成的激活，搜索过程也必须用正在激活阻塞函数。

## 2.3添加新的score函数

我们加入了一些标准的原程序中没有的分数函数。第一个函数是使用Okapi bm25的bm25score()算法[21]，一个计算查询项在页面中出现的频率假设每一项都是独立的。第二个函数是calculatelength()以urlid为主键的pagelength表页的长度作为返回值。这个函数根据较长的且比矮的分数高的页面的页面长度进行评分。计算页面长度的表并返回长度分数的函数时lengthscore()函数。注意，这个函数不仅返回在查询中包术语的页面，但也包含那些由包含其中的术语的锚文本链接查询。我们使用这种方法是因为锚文本是被证明是一个很好的页面内容指示器并表现为一个“一致标题”[18]。第三个函数是calculatehits()权威和枢纽得分的每一页基于查询独立点击[22 pp. 124-126]，一种算法计算全局权限和中心向量轻微减少了垃圾链接的影响。

