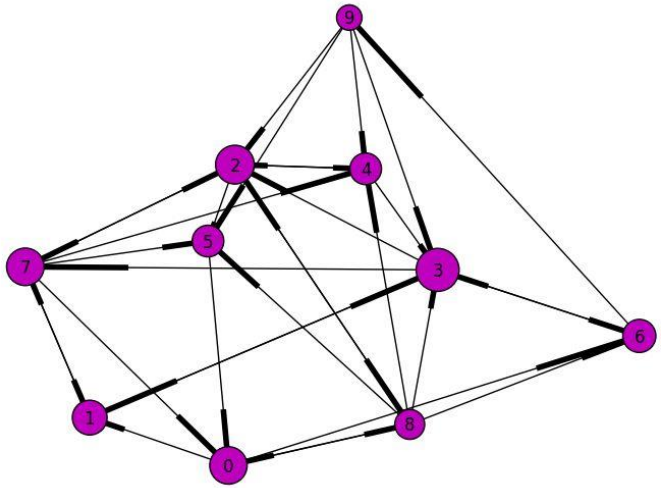
Python程序设计

陈远祥

chenyxmail@gmail.com

北京邮电大学 电子工程学院

网络爬虫



网络爬虫

网络爬虫定义

- 网络爬虫（Crawler）又被称为网页蜘蛛（Spider），网络机器人，网页追逐者，它是一种按照一定的规则，自动的抓取万维网信息的程序或者脚本
- 狭义与广义定义：狭义上指遵循标准的http协议，利用超链接和Web文档检索方法遍历万维网的软件程序；而广义的定义则是能遵循http协议，检索Web文档的软件都称之为网络爬虫

网络爬虫

网络爬虫用途

- 主要用途：数据采集

  金融，金融新闻/数据，制定投资策略，进行量化交易

  旅游，各类信息，优化出行策略

  电商，商品信息，比价系统

  游戏，游戏论坛，调整游戏运营

  银行，个人交易信息，征信系统/贷款评级

  招聘，职位信息，岗位信息

  舆情，各大论坛，社会群体感知，舆论导向

- 其他用途：12306抢票、各种抢购、投票、刷票、短信轰炸、网络攻击、Web漏洞扫描器

网络爬虫

- 爬虫原理
  - ✓ 网络蜘蛛从一组要访问的URL链接开始（种子URL），爬虫访问这些链接，它辨认出这些页面的所有超链接，然后添加到这个URL列表并按照一定的策略反复访问这些URL

网络爬虫

- URL含义
  - ✓ URL(Uniform Resource Locator)，即统一资源定位符，也就是我们说的网址，统一资源定位符是对可以从互联网上得到的资源的位置和访问方法的一种简洁的表示，是互联网上标准资源的地址。互联网上的每个文件都有一个唯一的URL，它包含的信息指出文件的位置以及浏览器应该怎么处理它
- URL的格式由三部分组成：
  - ✓ 第一部分是协议(或称为服务方式)
  - ✓ 第二部分是存有该资源的主机IP地址(有时也包括端口号)
  - ✓ 第三部分是主机资源的具体地址，如目录和文件名等

网络爬虫

- 浏览网页的过程
  - ✓ 比如浏览http://image.baidu.com/，我们会看到几张的图片以及百度搜索框
  - ✓ 过程：用户输入网址之后，经过DNS服务器，找到服务器主机，向服务器发出一个请求，服务器经过解析之后，发送给用户的浏览器HTML、JS、CSS等文件，浏览器解析出来，便可以看到形形色色的图片了
  - ✓ 因此，用户看到的网页实质是由HTML代码构成的，爬虫爬来的便是这些内容，通过分析和过滤这些HTML代码，实现对图片、文字等资源的获取

网络爬虫

- HTML、 CSS、 JavaScript
  - ✓ 网页主要由三部分组成：结构(Structure)、表现(Presentation)和行为(Behavior)
  - ✓ HTML-结构， 决定网页的结构和内容(是什么)，<head>元素标记头部文件，用<title>元素标记网页名称，用<body>元素标记网页主体，用<table>元素标记表格等等
  - ✓ CSS-表现(样式)，设定网页的表现样式(什么样子)，将网页样式提取出来方便更改某一类元素的样式，通过<style>元素插入CSS代码，<style>元素放在<head>元素中
  - ✓ JavaScript(JS)-行为，控制网页的行为(做什么)，微博评论点赞等），通过<script>元素来插入JS代码

网络爬虫

- 网络爬虫是否合法？
  - ✓ 网络爬虫目前还处于早期的蛮荒阶段，“允许哪些行为”这种基本秩序还处于建设之中。从目前的实践来看，如果抓取数据的行为用于个人使用，则不存在问题：而如果数据用于转载或者商业用途，那么抓取的数据类型就非常关键
  - ✓ 从很多历史案件来看，当抓取的数据是现实生活中的真实数据（比如，营业地址、电话清单）时，是允许转载的。但是，如果是原创数据（比如，意见和评论），通常就会受到版权限制

网络爬虫

- 网络爬虫是否合法？
  - ✓ 无论如何，当你抓取某个网站的数据时，请记住自己是该网站的访客，应当约束自己的抓取行为，否则他们可能会封禁你的IP，甚至采取更进一步的法律行动。这就要求下载请求的速度需要限定在一个合理值之内，并且还需要设定一个专属的用户代理来标识自己

网络爬虫

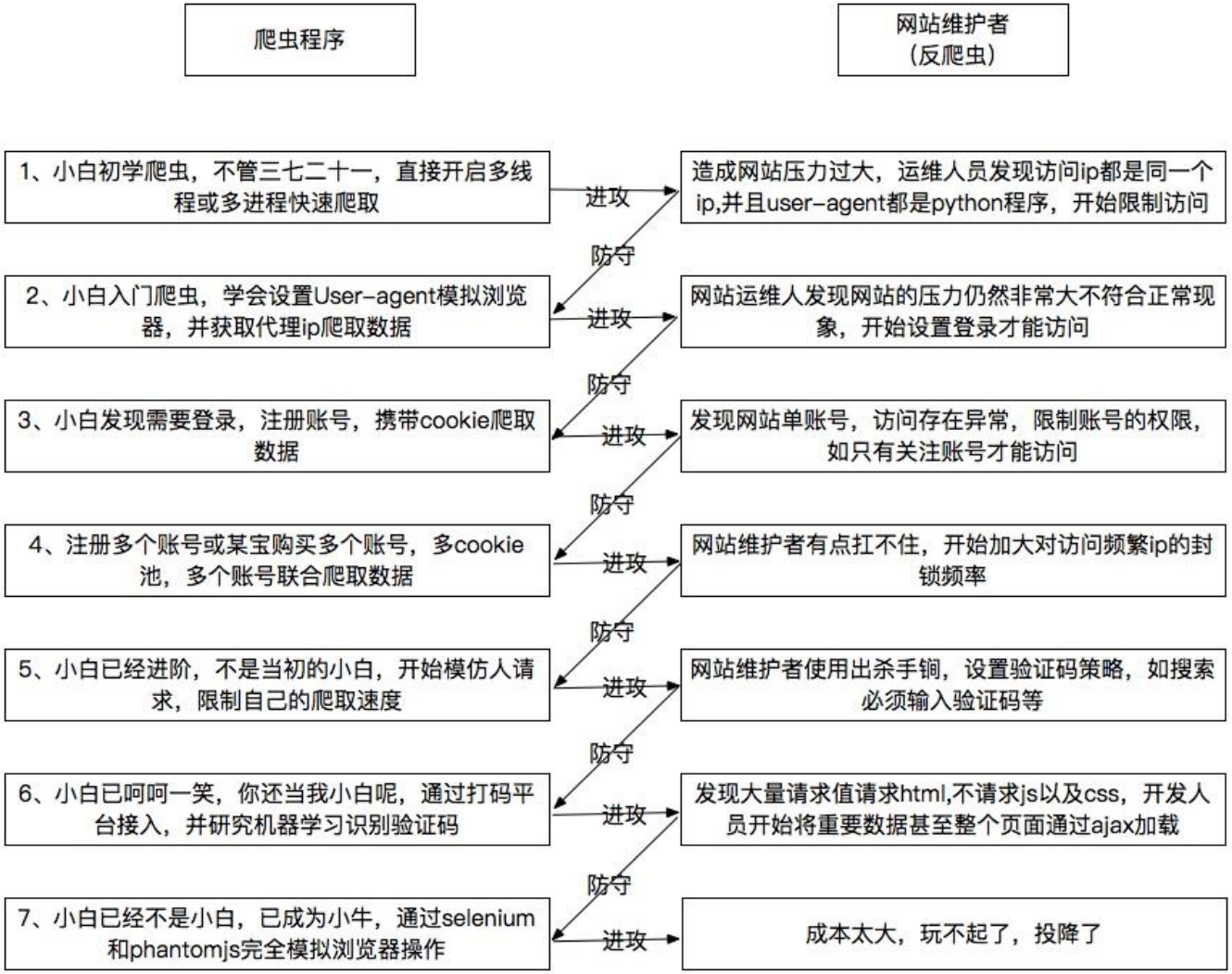
- 反爬虫
  - ✓ 初学者写的爬虫：简单粗暴，不管对端服务器的压力，甚至会把网站爬挂掉了
  - ✓ 数据保护：很多的数据对某些公司网站来说是比较重要的不希望被别人爬取
  - ✓ 商业竞争问题：这里举个例子是关于京东和天猫，假如京东内部通过程序爬取天猫所有的商品信息，从而做对应策略这样对天猫来说就造成了非常大的竞争

网络爬虫

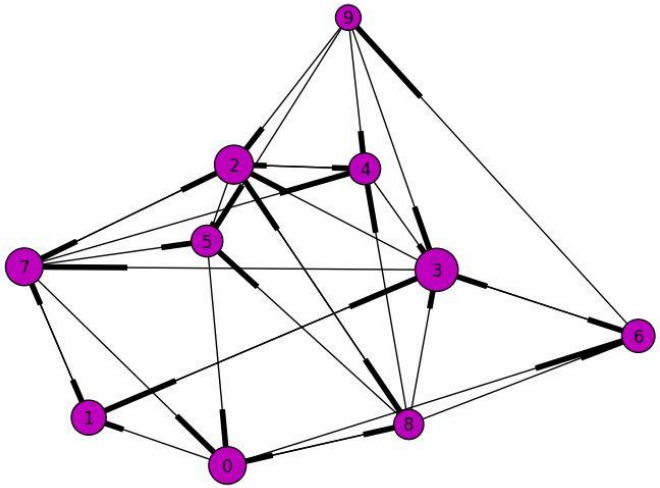
- 反爬虫
  - ✓ 爬虫行为分为搜索引擎爬虫及扫描程序爬虫，某一些网站可屏蔽特定的搜索引擎爬虫以节省带宽和性能，也可屏蔽扫描程序爬虫，避免网站被恶意抓取页面（网站洁癖）

网络爬虫

爬虫与反爬虫大战



网络爬虫分类



网络爬虫分类

- 网络爬虫分类
  - ✓ 根据使用场景，网络爬虫可分为通用爬虫和聚焦爬虫两种
  - ✓ 通用爬虫，搜索引擎和web服务商用的爬虫系统。通用网络爬虫是搜索引擎抓取系统（Baidu、Google、Yahoo等）的重要组成部分。主要目的是将互联网上的网页下载到本地，形成一个互联网内容的镜像备份
  - ✓ 聚焦爬虫，是"面向特定主题需求"的一种网络爬虫程序，它与通用搜索引擎爬虫的区别在于：聚焦爬虫在实施网页抓取时会对内容进行处理筛选，尽量保证只抓取与需求相关的网页信息

网络爬虫分类

- 通用搜索引擎工作原理
  - ✓ 尽可能的把互联网上的所有的网页下载下来，放到本地服务器里形成备份，再对这些网页做相关处理(提取关键字、去掉广告)，最后提供一个用户检索接口
  - ✓ 通用网络爬虫从互联网中搜集网页，采集信息，这些网页信息用于为搜索引擎建立索引从而提供支持，它决定着整个引擎系统的内容是否丰富，信息是否即时，因此其性能的优劣直接影响着搜索引擎的效果

网络爬虫分类

- 通用搜索引擎工作原理
  - ✓ 第一步：抓取网页
  - ✓ 首先选取一部分种子URL，把这些URL放到待爬取队列
  - ✓ 从队列取出URL，然后解析DNS得到主机IP，然后保存这个IP对应的服务器里下载HTML页面，保存到搜索引擎的本级服务器，之后把这个爬过的URL放入已爬过的队列
  - ✓ 分析这些网页内容，找出网页里其他的URL链接，继续执行第二步，知道爬取结束

网络爬虫分类

- 通用搜索引擎工作原理

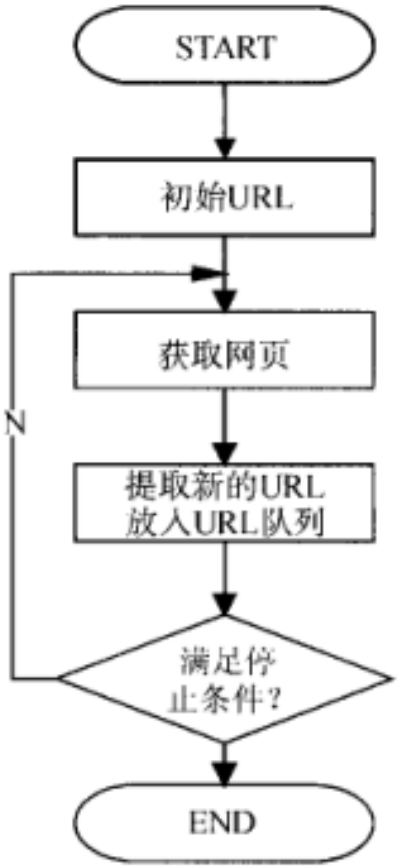


图 1　通用网络爬虫工作流程图

网络爬虫分类

- 通用搜索引擎工作原理
  - ✓ 第二步：数据存储
  - ✓ 搜索引擎通过爬虫爬取到的网页，将数据存入原始页面数据库，其中的页面数据与用户浏览器得到的HTML是完全一样的
  - ✓ 搜索引擎蜘蛛在抓取页面时，也做一定的重复内容检测，一旦遇到访问权重很低的网站上有大量抄袭、采集或者复制的内容，很可能就不再爬行

网络爬虫分类

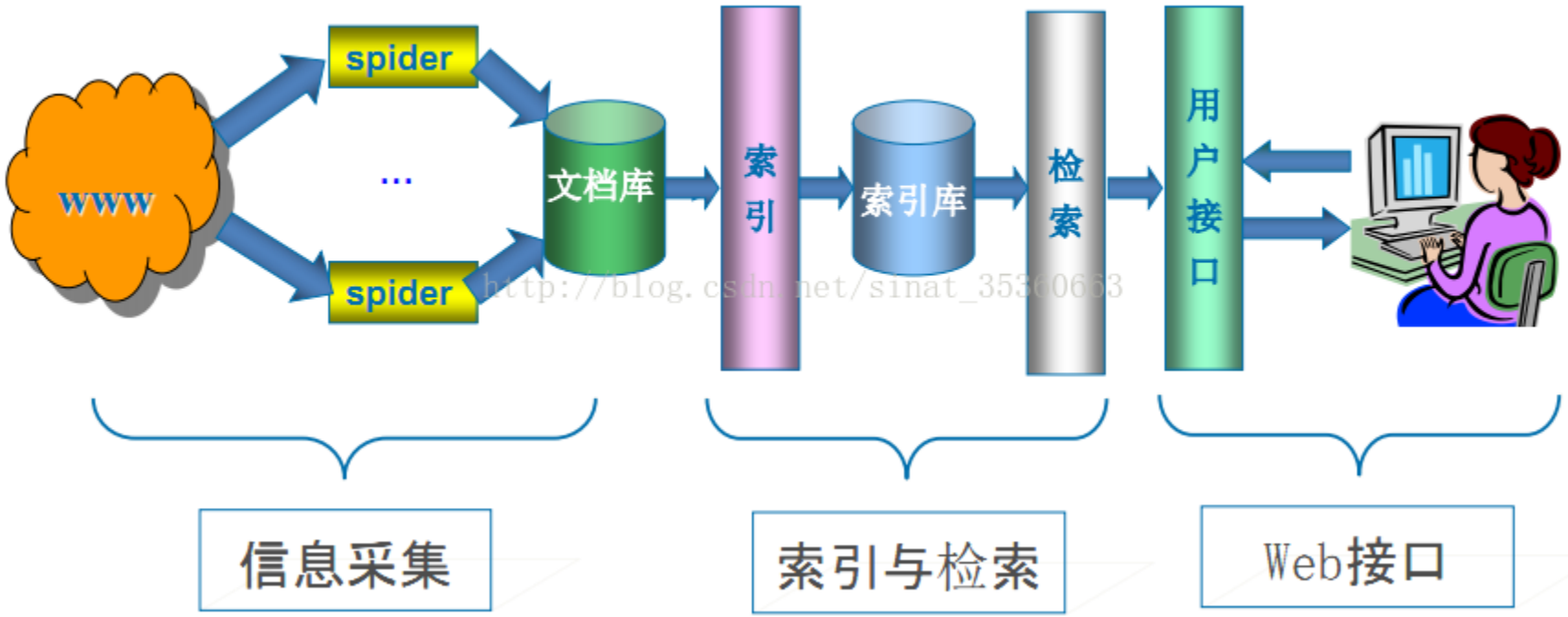
- 通用搜索引擎工作原理
  - ✓ 第三步：数据预处理，搜索引擎将爬虫抓取回来的页面，进行各种步骤的预处理
  - ✓ 提取文字
  - ✓ 中文分词
  - ✓ 消除噪音（比如版权声明文字、导航条、广告等……）
  - ✓ 索引处理
  - ✓ 链接关系计算
  - ✓ 特殊文件处理
  - ✓ ....

网络爬虫分类

- 通用搜索引擎工作原理
  - ✓ 第四步：提供检索服务，网站排名
  - ✓ 搜索引擎在对信息进行组织和处理后，为用户提供关键字检索服务，将用户检索相关的信息展示给用户
  - ✓ 同时会根据页面的PageRank值（链接的访问量排名）来进行网站排名，这样Rank值高的网站在搜索结果中会排名较前，当然也可以直接使用Money购买搜索引擎网站排名，简单粗暴

网络爬虫分类

- 通用搜索引擎工作原理



网络爬虫分类

- 通用搜索引擎的局限性
  - ✓ 通用搜索引擎所返回的结果都是网页，而大多情况下，网页里90%的内容对用户来说都是无用的
  - ✓ 不同领域、不同背景的用户往往具有不同的检索目的和需求，搜索引擎无法提供针对具体某个用户的搜索结果
  - ✓ 万维网数据形式的丰富和网络技术的不断发展，图片、数据库、音频、视频多媒体等不同数据大量出现，通用搜索引擎对这些文件无能为力，不能很好地发现和获取
  - ✓ 通用搜索引擎大多提供基于关键字的检索，难以支持根据语义信息提出的查询，无法准确理解用户的具体需求

网络爬虫分类

- 解决通用爬虫的缺点，聚焦爬虫出现了
  - ✓ 聚焦爬虫，爬虫程序员写的针对某种特定内容爬虫
  - ✓ 面向主题爬虫、面向需求爬虫：会针对某种特定的容去爬取信息，而且保证内容需求尽可能相关

网络爬虫分类

- 聚焦爬虫流程
  - ✓ 聚焦爬虫根据一定的网页分析算法过滤与主题无关的链接，保留有用的链接并将其放入等待抓取的URL队列。然后，它将根据一定的搜索策略从队列中选择下一步要抓取的网页URL，并重复上述过程，直到达到系统的某一条件时停止

网络爬虫分类

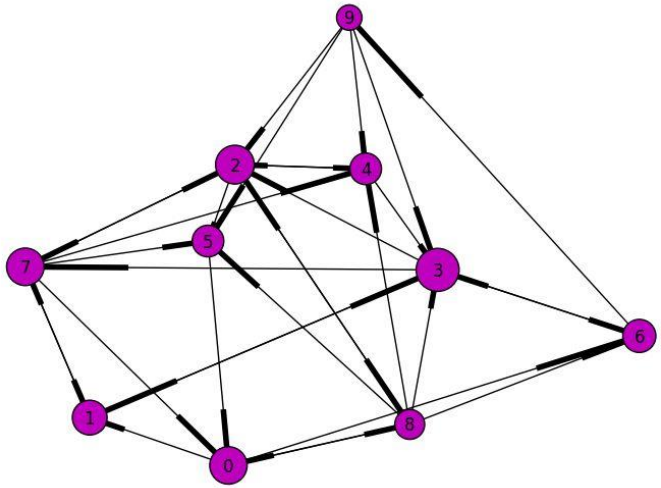
- 相对于通用网络爬虫，聚焦爬虫还需要解决三个主要问题：
  - ✓ 对抓取目标的描述或定义
  - ✓ 对URL的搜索策略：深度优先、广度优先和最佳优先
  - ✓ 对网页或数据的分析与过滤：基于网络拓扑、基于网页内容和基于用户访问行为
- 三者关系
  - ✓ 抓取目标的描述和定义是决定网页分析算法与URL搜索策略如何制订的基础。而网页分析算法和候选URL排序算法是决定搜索引擎所提供的服务形式和爬虫网页抓取行为的关键所在。这两个部分的算法又是紧密相关的

网络爬虫分类

- 两种爬虫比较

	通用网络爬虫	聚焦爬虫
目标	通用网络爬虫的目标是尽可能多的采集信息页面，而在这一过程中它并不太在意页面采集的顺序和被采集页面的相关主题。这需要消耗很多的系统资源和网络带宽，并且对这些资源的消耗并没有换来采集页面的较高利用率	聚焦爬虫的目标是尽可能快地爬行、采集尽可能多的与预先定义好的主题相关的网页。聚焦爬虫可以通过对整个Web按主题分块采集，并将不同块的采集结果整合到一起，以提高整个Web的采集覆盖率和页面利用率

聚焦爬虫关键技术分析



关键技术分析

- 关键技术
  - ✓ 抓取目标的定义与描述
  - ✓ 网页URL的搜索策略
  - ✓ 网页的分析与信息的提取

关键技术分析

- 抓取目标的定义与描述
  - ✓ 针对有目标网页特征的网页级信息：对应网页库级垂直搜索，抓取目标网页，后续还要从中抽取出需要的结构化信息。稳定性和数量上占优，但成本高、性活性差。
  - ✓ 针对目标网页上的结构化数据：对应模板级垂直搜索，直接解析页面，提取并加工出结构化数据信息。快速实施、成本低、灵活性强，但后期维护成本高。

关键技术分析

- URL的搜索策略
  - ✓ 基于IP地址搜索策略
  - ✓ 广度优先
  - ✓ 深度优先
  - ✓ 最佳优先

关键技术分析

- 基于IP地址搜索策略
  - ✓ 先赋予爬虫一个起始的IP地址，然后根据IP地址递增的方式搜索本口地址段后的每一个WWW地址中的文档，它完全不考虑各文档中指向其它Web站点的超级链接地址
  - ✓ 优点是搜索全面，能够发现那些没被其它文档引用的新文档的信息源
  - ✓ 缺点是不适合大规模搜索

关键技术分析

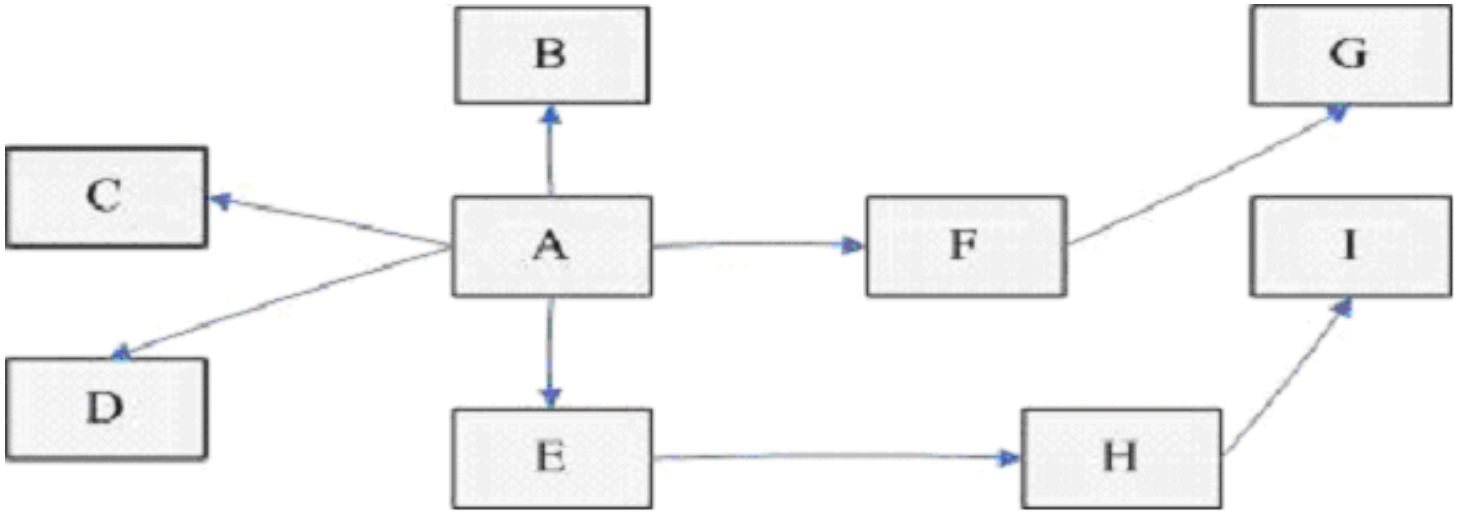
- 广度优先搜索策略
  - ✓ 广度优先搜索策略是指在抓取过程中，在完成当前层次的搜索后，才进行下一层次的搜索。这样逐层搜索，依此类推
  - ✓ 该算法的设计和实现相对简单。为覆盖尽可能多的网页，一般使用广度优先搜索方法

关键技术分析

- 广度优先搜索策略
  - ✓ 另外一种方法是将广度优先搜索与网页过滤技术结合使用，先用广度优先策略抓取网页，再将其中无关的网页过滤掉。这些方法的缺点在于，随着抓取网页的增多，大量的无关网页将被下载并过滤，算法的效率将变低

关键技术分析

- 广度优先搜索策略



使用广度优先策略抓取的顺序为：

A-B、C、D、E、F-G、H-I

关键技术分析

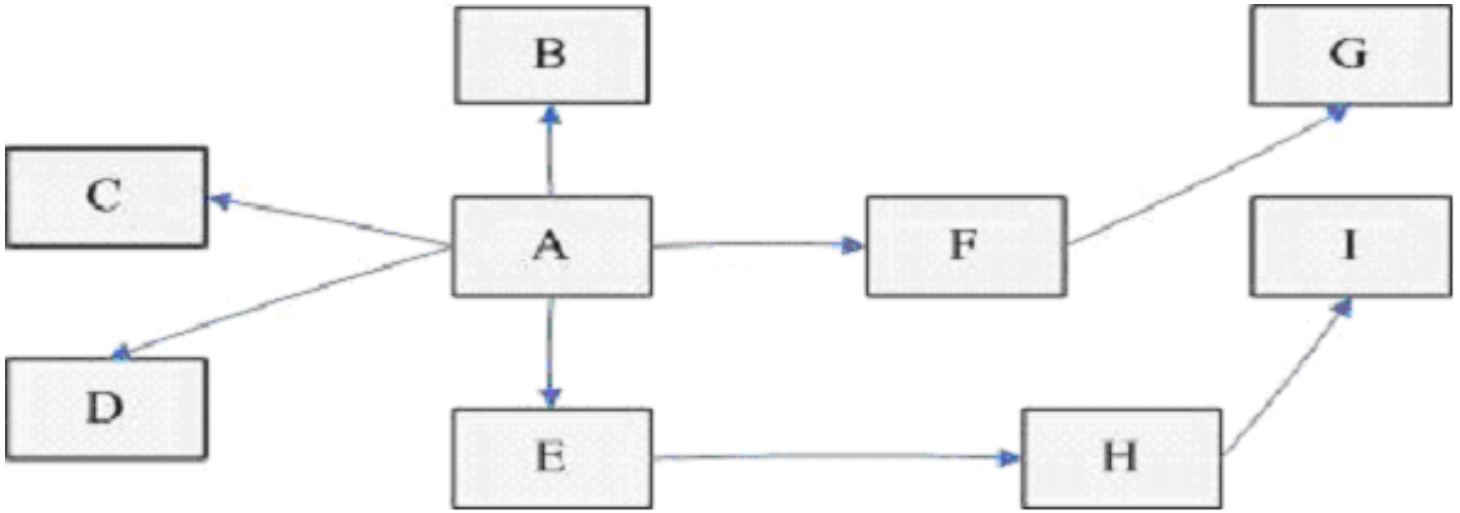
- 深度优先搜索策略
  - ✓ 深度优先搜索在开发网络爬虫早期使用较多的方法之一，目的是要达到叶结点，即那些不包含任何超链接的页面文件
  - ✓ 从起始页开始在当前HTML文件中，当一个超链被选择后，被链接的HTML文件将执行深度优先搜索，一个链接一个链接跟踪下去，处理完这条线路之后再转入下一个起始页，继续跟踪链接，即在搜索其余的超链结果之前必须先完整地搜索单独的一条链

关键技术分析

- 深度优先搜索策略
  - ✓ 深度优先搜索沿着HTML文件上的超链走到不能再深入为止，然后返回到某一个HTML文件，再继续选择该HTML文件中的其他超链。当不再有其他超链可选择时，说明搜索已经结束
  - ✓ 这个方法的优点是网络蜘蛛在设计的时候比较容易

关键技术分析

- 深度优先搜索策略



使用深度优先策略抓取的顺序为：

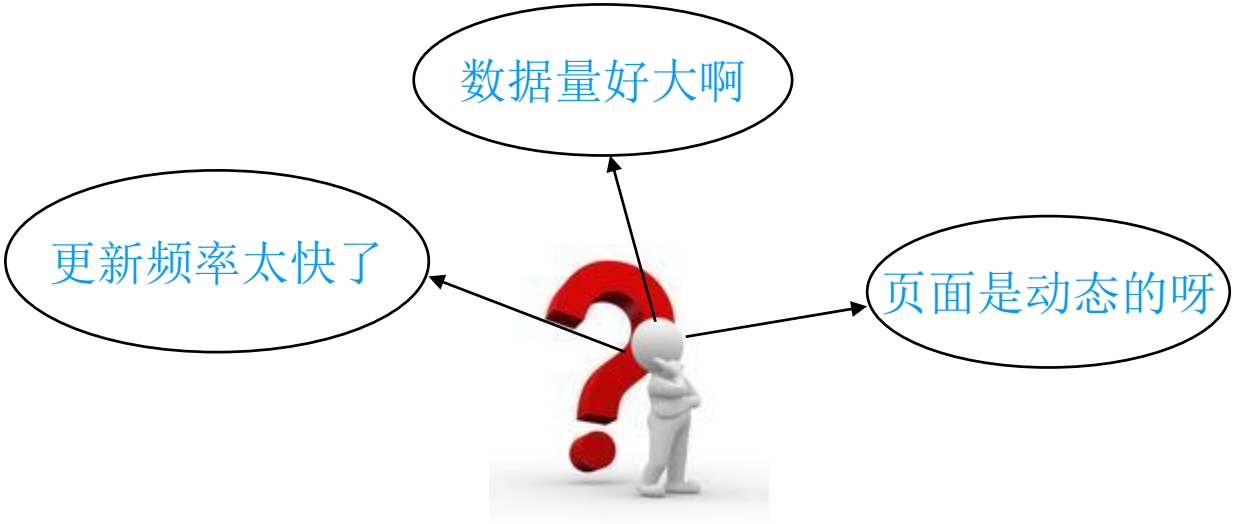
A-F-G、E-H-I、B、C、D

关键技术分析

- 最佳优先搜索策略
  - ✓ 最佳优先搜索策略按照一定的网页分析算法，先计算出URL描述文本的目标网页的相似度，设定一个值，并选取评价得分超过该值的一个或几个URL进行抓取。它只访问经过网页分析算法计算出的相关度大于给定的值的网页
  - ✓ 存在的一个问题是，在爬虫抓取路径上的很多相关网页可能被忽略，因为最佳优先策略是一种局部最优搜索算法。因此需要将最佳优先结合具体的应用进行改进，以跳出局部最优点
  - ✓ 有研究表明，这样的闭环调整可以将无关网页数量降低30%-90%

关键技术分析

- 爬行策略



三种网络特征使得设计网页爬虫抓取策略变得很难

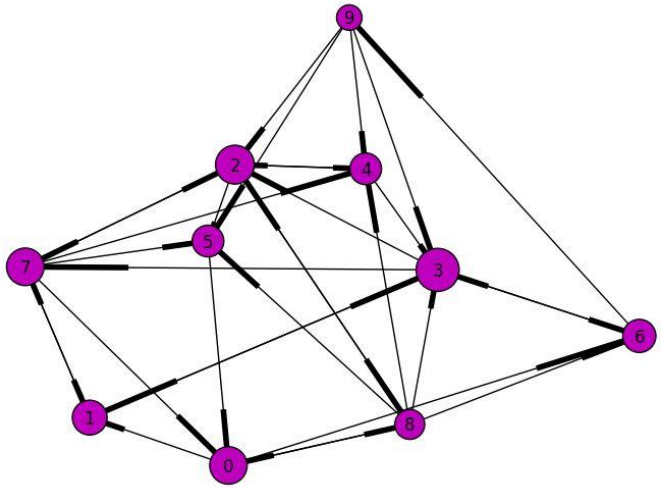
关键技术分析

- 网页爬虫的行为通常是四种策略组合的结果：
  - ✓ 选择策略，决定所要下载的页面
  - ✓ 重新访问策略，决定什么时候检查页面的更新变化
  - ✓ 平衡礼貌策略，指出怎样避免站点超载
  - ✓ 并行策略，指出怎么协同达到分布式抓取的效果

关键技术分析

- 网页的分析及信息的提取
  - ✓ 基于网络拓扑关系的分析算法：根据页面间超链接引用关系，来对与已知网页有直接或间接关系对象作出评价的算法。网页粒度PageRank，网站粒度SiteRank
  - ✓ 基于网页内容的分析算法：从最初的文本检索方法，向涉及网页数据抽取、机器学习、数据挖掘、自然语言等多领域综合的方向发展
  - ✓ 基于用户访问行为的分析算法：有代表性的是基于领域概念的分析算法

网络爬虫之前期工作



背景调研

- 检查robots.txt
  - ✓ Robots协议（也称为爬虫协议、机器人协议等）的全称是“网络爬虫排除标准”（Robots Exclusion Protocol），网站通过Robots协议告诉搜索引擎哪些页面可以抓取，哪些页面不能抓取
  - ✓ 当一个搜索蜘蛛访问一个站点时，它会首先检查该站点根目录下是否存在robots.txt，如果存在，搜索机器人就会按照该文件中的内容来确定访问的范围；如果该文件不存在，所有的搜索蜘蛛将能够访问网站上所有没有被口令保护的页面

背景调研

- 检查robots.txt
  - ✓ 豆瓣网：https://www.douban.com/robots.txt
  - ✓ http://example.webscraping.com/robots.txt

```
User-agent: *
Disallow: /subject_search
Disallow: /amazon_search
Disallow: /search
Disallow: /group/search
Disallow: /event/search
Disallow: /celebrities/search
Disallow: /location/drama/search
Disallow: /forum/
Disallow: /new_subject
Disallow: /service/iframe
Disallow: /j/
Disallow: /link2/
Disallow: /recommend/
Disallow: /trailer/
Disallow: /doubanapp/card
Sitemap: https://www.douban.com/sitemap_index.xml
Sitemap: https://www.douban.com/sitemap_updated_index.xml
# Crawl-delay: 5

User-agent: Wandoujia Spider
Disallow: /
```

背景调研

- 检查Sitemap文件
  - ✓ 网站提供的Sitemap文件（即网站地图）可以帮助爬虫定位网站最新的内容，而无须爬取每一个网页
  - ✓ 网站地图提供了所有网页的链接，可用于爬虫
  - ✓ 虽然Sitemap文件提供了一种爬取网站的有效方式，但是我们仍需对其谨慎处理，因为该文件经常存在缺失、过期或不完整的问题

背景调研

- 估算网站大小
  - ✓ 目标网站的大小会影响我们如何进行爬取。如果是像我们的示例站点这样只有几百个URL的网站，效率并没有那么重要，但如果是拥有数百万个网页的站点，使用串行下载可能需要持续数月才能完成，这时就需要使用分布式下载来解决了
  - ✓ 估算网站大小的一个简便方法是检查Google或者百度爬虫的结果，因为Google或者百度很可能已经爬取过我们感兴趣的网站
  - ✓ site:example.webscraping.com
  - ✓ site:www.douban.com

背景调研

- 识别网站所用技术
  - ✓ 构建网站所使用的技术类型也会对我们如何爬取产生影响。有一个十分有用的工具可以检查网站构建的技术类型-builtwith模块
  - ✓ 该模块将URL作为参数，下载该URL并对其进行分析，然后返回该网站使用的技术

```
import builtwith
builtwith.parse("http://example.webscraping.com/")
builtwith.parse("https://www.douban.com/")
```

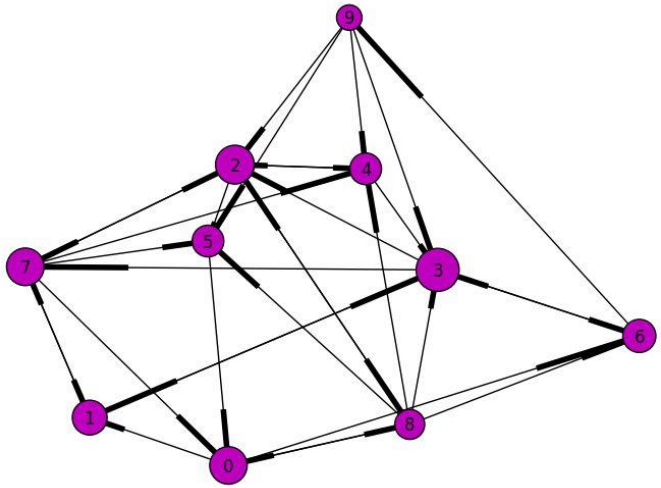
背景调研

- 识别网站所用技术
  - ✓ 网站的内容很有可能是嵌入在HTML中的，相对而言比较容易抓取；如果采用AngularJS构建该网站，此时的网站内容就很可能是动态加载的；如果网站使用了ASP.NET，那么在爬取网页时，就必须要用到会话管理和表单提交

背景调研

- 寻找网站所有者
  - ✓ 对于一些网站，我们可能会关心其所有者是谁。比如，我们已知网站的所有者会封禁网络爬虫，那么我们最好把下载速度控制得更加保守一些。为了找到网站的所有者，我们可以使用WHOIS协议查询域名的注册者是谁
  - ✓ import whois
  - ✓ print whois.whois('https://www.douban.com/’)
  - ✓ print whois.whois('https://movie.douban.com/')

编写第一个网络爬虫



编写第一个网络爬虫

- 下载网页
  - ✓ 要想爬取网页，我们首先需要将其下载下来。使用Python的urllib2模块下载URL
  - ✓ #download1
  - ✓ 这个代码片段存在一个问题，即当下载网页时，我们可能会遇到一些无法控制的错误，比如请求的页面可能不存在。此时，urllib2会抛出异常，然后退出脚本。安全起见，给出一个更健壮的版本，可以捕获这些异
  - ✓ #download2

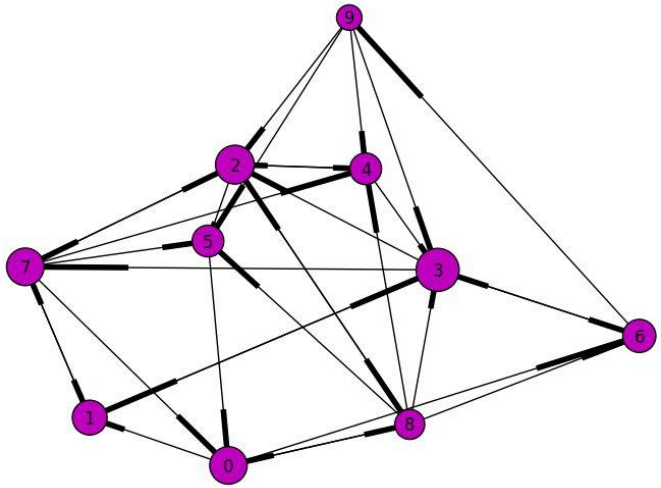
编写第一个网络爬虫

- 重试下载
  - ✓ 下载时遇到的错误经常是临时性的，比如服务器过载时返回的503 Service Unavailable错误。对于此类错误，我们可以尝试重新下载，因为这个服务器问题现在可能已解决。不过，我们不需要对所有错误都尝试重新下载。如果服务器返回的是404 Not Found这种错误，则说明该网页目前并不存在，再次尝试同样的请求一般也不会出现不同的结果
  - ✓ 到4xx错误发生在客户端请求存在问题时，而5xx错误则发生在服务端存在问题时。所以，我们只需要确保download函数在发生5xx错误时重试下载即可
  - ✓ #download3

编写第一个网络爬虫

- 设置用户代理
  - ✓ 默认情况下，urllib 2使用Python urllib/2.7作为用户代理下载网页内容
  - ✓ 如果能使用可辨识的用户代理则更好，这样可以避免网络爬虫碰到一些问题。一些网站曾经历过质量不佳的Python网络爬虫造成的服务器过载，会封禁这个默认的用户代理
  - ✓ #download4

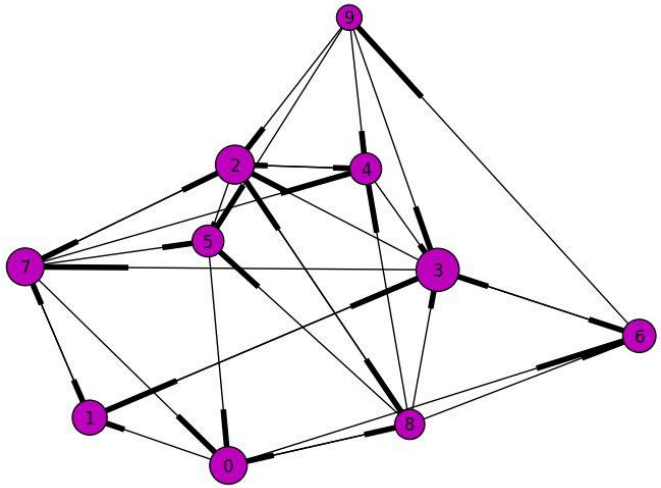
网站地图爬虫



网站地图爬虫

- 网站地图爬虫
  - ✓ 使用示例网站robots.txt文件中发现的网站地图，利用正则表达式下载所有网页
  - ✓ #sitemap_crawler

ID遍历爬虫



ID遍历爬虫

- ID遍历爬虫
  - ✓ 利用网站结构的弱点，更加轻松地访问所有内容

下面是一些示例国家的URL

- ✓ http://example.webscraping.com/places/default/view/Afghanistan-1
- ✓ http://example.webscraping.com/places/default/view/Aland-Islands-2
- ✓ http://example.webscraping.com/places/default/view/Albania-3

ID遍历爬虫

- ID遍历爬虫
  - ✓ 可以看出，这些URL只在结尾处有所区别，包括国家名（作为页面别名）和ID
  - ✓ 在URL中包含页面别名是非常普遍的做法，可以对搜索引擎优化起到帮助作用。一般情况下，Web服务器会忽略这个字符串，只使用ID来匹配数据库中的相关记录
  - ✓ 加载

http://example.webscraping.com/places/default/view/1

http://example.webscraping.com/places/default/view/Afghanistan-1

  - ✓ #iteration_crawler1.py

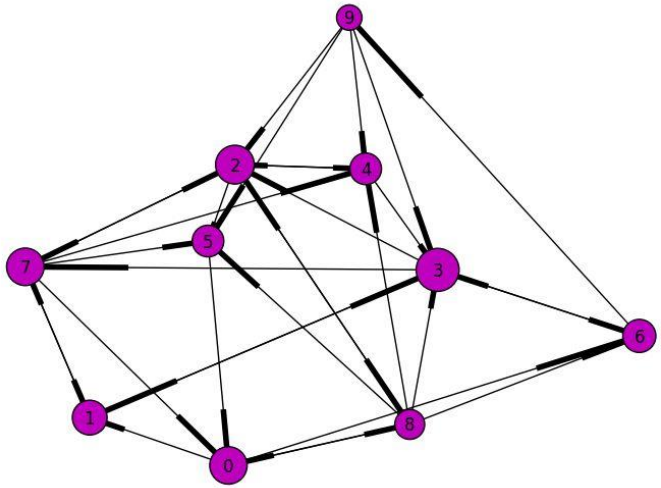
ID遍历爬虫

- ID遍历爬虫
  - ✓ 这种实现方式存在一个缺陷，那就是某些记录可能已被删除，数据库ID之间并不是连续的。此时，只要访问到某个间隔点，爬虫就会立即退出。下面是这段代码的改进版本，在该版本中连续发生多次下载错误后才会退出程序
  - ✓ #iteration_crawler2.py

ID遍历爬虫

- ID遍历爬虫
  - ✓ 在爬取网站时，遍历ID是一个很便捷的方法，但是和网站地图爬虫一样，这种方法也无法保证始终可用。比如，一些网站会检查页面别名是否满足预期，如果不是，则会返回404 Not Found错误。而另一些网站则会使用非常连续大数作为ID，或是不使用数值作为ID，此时遍历就难以发挥其作用了

链接爬虫



链接爬虫

- 链接爬虫
  - ✓ 要让爬虫表现得更像普通用户，跟踪链接，访问感兴趣的内容
  - ✓ 链接爬虫：通过跟踪所有链接的方式，我们可以很容易地下载整个网站的页面。但是，这种方法会下载大量我们并不需要的网页，利用正则表达式匹配想爬取的网页

链接爬虫

- 链接爬虫
  - ✓ 要爬取的是国家列表索引页和国家页面

其中，索引页链接格式如下

  - ✓ http://example.webscraping.com/index/1

国家页链接格式如下。

  - ✓ http://example.webscraping.com/view/Afghanistan-1

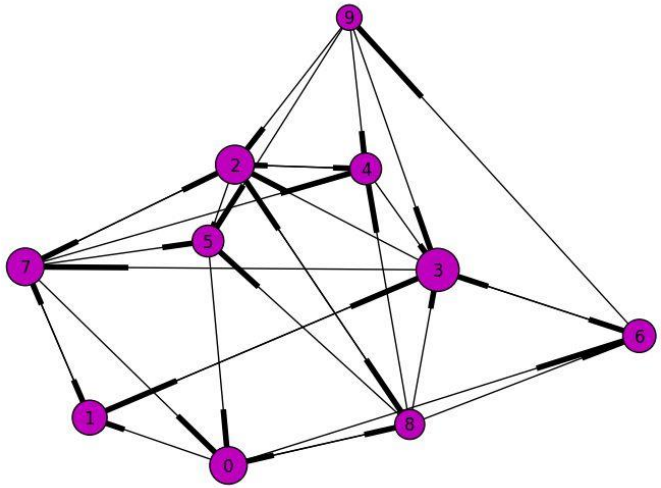
我们可以用/(index|view)/这个简单的正则表达式来匹配这两类网页

  - ✓ #link_crawler1

链接爬虫

- 为链接爬虫添加一些功能，使其在爬取其他网站时更加有用
  - ✓ 解析robost.txt，以避免下载禁止爬取的URL
  - ✓ 支持代理，访问某些屏蔽的网站
  - ✓ 下载限速，降低被封禁或是造成服务器过载的风险
  - ✓ 避免爬虫陷阱，我们的爬虫会跟踪所有之前没有访问过的链接。但是，一些网站会动态生成页面内容，这样就会出现无限多的网页，通过定义爬虫深度，避免爬虫陷阱

数据抓取



数据抓取

- 三种网页抓取方法
  - ✓ 正则表达式
  - ✓ BeautifulSoup模块
  - ✓ lxml模块

数据抓取

- 正则表达式
  - ✓ 抓取面积数据
  - ✓ #regex_example
  - ✓ 正则表达式特点：简单正则表达式无法适应网页未来变化，微小的布局变化也会使该正则表达式无法满足，复杂正则表达式难以构造、可读性差

数据抓取

- Beautiful Soup
  - ✓ Beautiful Soup是一个非常流行的Python模块。该模块可以解析网页，并提供定位内容的便捷接口
  - ✓ 使用Beautiful Soup的第一步是将已下载的HTML内容解析为soup文档。由于大多数网页都不具备良好的HTML格式，因此Beautiful Soup需要对其实际格式进行确定，存在属性值两侧引号缺失和标签未闭合的问题
  - ✓ #fixed_html
  - ✓ #bs_example

数据抓取

- Lxml
  - ✓ Lxml是基于libxml2这一XML解析库的Python封装。该模块使用C语言编写，解析速度比Beautiful Soup更快
  - ✓ 和Beautiful Soup一样，使用lxml模块的第一步也是将有可能不合法的HTML解析为统一格式，再选择抓取数据（css选择器）
  - ✓ #fixed_lxml.html
  - ✓ #lxml

数据抓取

- 性能对比
  - ✓ 三种抓取方法评估取舍，我们需要对其相对效率进行对比
  - ✓ 一般情况下，爬虫会抽取网页中的多个字段。因此，为了让对比更加真实，我们将为每个爬虫实现一个扩展版本，同时进行抽取国家网页中的每个可用数据
  - ✓ #performance

数据抓取

- 性能对比
  - ✓ Beautiful Soup比其他两种方法慢了超过6倍之多，因为lxml和正则表达式模块都是C语言编写的，而Beautiful Soup则是纯Python编写的
  - ✓ lxml和正则表达式性能差不多，由于lxml在搜索元素之前，必须将输入解析为内部格式，因此会产生额外的开销。而当抓取同一网页的多个特征时，这种初始化解析产生的开销就会降低，lxml也就更具竞争力

数据抓取

- 性能对比

抓取方法	性能	使用难度	安装难度
正则表达式	快	困难	简单（内置模块）
Beautiful Soup	慢	简单	简单（纯Python）
Lxml	快	简单	相对困难

谢谢