关于机器人（爬虫）的学习

Web机器人（爬虫）其实是自活跃的用户代理！

Web机器人（爬虫）的其他名字有：爬虫、蜘蛛、蠕虫以及机器人。

**一、机器人（爬虫）的爬行方式**：

它们会递归的对各种信息性web站点进行遍历，获取第一个web页面，然后获取那个页面指向的所有web页面，以此类推。

1.从哪开始：根集

机器人（爬虫）开始访问的url初始集合被叫做根集（root set），根集应该选择足够多的不同站点url。

通常一个好的根集会包括一些大的流行web站点、一个新创建的页面的列表、和一个不经常被连接的无名页面列表。

2.机器人（爬虫）要提取链接以及相对链接的标准化

机器人（爬虫）要通过简单的HTML解析，将这些链接提取出来，并将相对URL转换为绝对形式。

3.避免环路的出现

机器人（爬虫）在爬行时，要小心不要陷入循环，或者环路（cycle）之中。

机器人（爬虫）必须知道他们到过什么地方，以避免环路的出现，这些环路会造成陷阱，减缓或者暂停爬行过程。

4.循环与复制

至少有以下三个原因，环路对机器人（爬虫）是有害的

循环会使未经良好设计的机器人（爬虫）不停的兜圈子。

机器人（爬虫）不断地获取相同页面时，会对另一端的web服务器进行打击，阻止所有真实用户访问这个站点。这种拒绝服务是可以作为法律诉讼理由的。

机器人（爬虫）会获取大量重复的页面【通常被称为dups（重复），以便于loops（循环）押韵】。这样获取重复内容的程序就变得无用。

5.**记录搜索过地方的技巧**（最好全部掌握）

**A.树和散列表（这些是加速URL查找的软件数据结构）**

复杂的机器人（爬虫）会用搜索树或者散列表来记录访问过的URL。

**B.有损的存在位图**

为了减小空间，一些大型的机器人（爬虫）会使用有损数据结构，比如存位数组（bit array）。用一个散列函数将每个URL都转换为一个定长的数字。这个数字在数组中有相应的“存在位”，爬过一个URL就将“存在位”置位。

**C.检查点**

一定要将已访问URL列表保存到硬盘上，以防机器人（爬虫）程序崩溃。

**D.分类**

因为URL的潜在数量是无限的，所有有些大型的web机器人（爬虫）会使用机器人（爬虫）“集群”，每个机器人（爬虫）负责一个特定的URL“片”。

6.别名与机器人（爬虫）环路

由于URL别名的存在，有时很难分辨出以前是否访问过某个页面。如果两个URL看起来不一样，但其实指向同一资源，就称这两个URL互为“别名”。

以下：

1. 添加了端口号
2. ~与%7其实是一个东西，相同东西的不同编码
3. 换了某些标签，但是内容无修改
4. 没有区分大小写
5. 加了index默认页面跟没加
6. 使用了IP地址

**7.规范化URL**

将URL规范化来消除上面那些显而易见的别名。

**A．如果没有指定端口的话，就向主机名中添加“:80”。**

**B. 将所有转义符%xx都转换为成等价字符。**

**C. 删除# 标签**

如果不知道特定web服务器的相关信息，机器人（爬虫）就没什么好办法来避免D-F的问题。

8.文件系统连接环路

“邪恶的网管”或者服务器管理员的有心或无心下回造成这样的机器人（爬虫）陷阱。

比如：

某一超链接指向了地址“/”，但是由于URL看起来不同，所以机器人（爬虫）无法单从URL本身判断出文档是相同的。如果没有某种循环的检测方式，这个环路就会继续下去。

9.动态虚拟WEB空间

恶意网管会有意创建包含了到同一服务器上虚构URL链接的HTML。

请求这些虚构的URL时，这个邪恶的服务器就会捏造出一个带有新的虚构URL的新HTML页面出来

比如，一个基于CGI的日历程序，它会生成一个月历和一个指向下个月的链接，真正的用户是不会不停的请求下个月的连接的，但不了解其内容的动态特性机器人（爬虫）可能会不断的请求获取这些资源。

**10.避免循环和重复**

良好设计的机器人（爬虫）要包涵一组试探方法，以避免环路的出现。

**A.规范化URL**

将URL转换为标准形式避免语法上的别名（见第7条）

**B.广度优先的爬行**

以广度优先的方式来调度URL去访问Web站点，就可以将环路的影响最小化。即使碰到了陷阱，也可以回到环路中获取下一个页面。就是说优先搜索，域名不同的web站点。

大体上是：优先搜索第一层结构的网站然后搜索第二层…

详情请查询：广度优先搜索

**C.节流**

限制一段时间内机器人（爬虫）可以从一个Web站点获取的页面数量，也可以通过节流来限制重复的页面总数和对服务器的访问总数。

**D.限制URL的大小**

机器人（爬虫）可能会拒绝爬行超出特定长度（通常是1KB）的URL。如果环路的长度增加，长度限制最终会终止这个环路。

这种技术会错过某些内容。现在很多站点会用URL来管理用户状态，比如储存用户ID。

**E.URL/站点黑名单**

维护一个与机器人（爬虫）环路和陷阱相对应的已知站点及URL列表，然后躲开他们，发现新问题就将其加入黑名单。这需要人工干预。

**F. 模式监测**

拒绝爬行带有多余两个或者三个重复组件的URL

比如：”subdir/subdir/subdir…”

或者：“subdir/images/subdir/images…”

查找多种不同周期的重复模式

**G. 内容指纹**

一些更复杂的Web机器人（爬虫）会使用指纹这种方式来监测重复。使用内容指纹的机器人（爬虫）会获取页面内容中的字节，并计算出一个效验和（checksum）。这个效验和是页面内容的压缩表示形式。如果机器人（爬虫）获取了一个页面，而这个页面的效验和它见过，就不会去爬取了。

MD5这样的报文摘要函数就常被用于指纹计算。

由于某些动态页面的原因，所以有时在计算效验和的时候会忽略web页面内容中的某些部分，比如嵌入的链接。

**H.人工监视**

设计所有产品级机器人（爬虫）时都要有诊断和日志功能，这样人类才能很方便的监视机器人（爬虫）的进展。如果发生什么不寻常的事情就发出警告。

**二、机器人（爬虫）的HTTP**

机器人（爬虫）和其他的HTTP客户端程序没什么区别。他们也要遵守HTTP规范中的规则。比如：伪装成HTTP/1.1客户端的机器人（爬虫）也要使用正确的HTTP请求首部。

很多机器人（爬虫）会发出HTTP/1.0的请求，因为这个协议的要求很少。

1.识别请求首部

鼓励使用以下内容（用来追踪犯错的机器人（爬虫）,也是留下自己痕迹的证据）：

A. User-Agent

将发起请求的机器人（爬虫）名字告知服务器

B. From

提供机器人（爬虫）的用户/管理者的E-mail地址

C. Accept

告知服务器可以发送哪些媒体类型。这有助于确保机器人（爬虫）只接收它感兴趣的内容（文本、图片等）。

如果对特定版本感兴趣，发送类似Accept-Charset之类的首部都是很有帮助的。

D. Referer

提供包含了当前请求URL文档的URL。

2.虚拟主机

机器人（爬虫）实现者要支持HOST首部，不然会有悲剧发生？？？

可能会使机器人（爬虫）将错误的内容和服务器上的其他站点联系起来。

3.条件请求

只有内容发生变化时才重新获取内容是很有意义的。

大体上与HTTP缓存查看已获取资源的本地副本是否有效的方法类似。

4.对相应的处理

一般机器人（爬虫）主要用简单的GET方法来获取内容，但是，使用了某些HTTP特性的机器人（爬虫）,并且与服务器进行交互的机器人（爬虫）则要能够对各种不同类型的HTTP响应进行处理。

A. 状态码

所有机器人（爬虫）都应该理解200 OK 和 404 NOT Found这样的状态码。当然，还有一些其他的，也需要进行处理。

B. 实体

除了HTTP首部信息之外，机器人（爬虫）也会在实体中查找信息。HTML元标签，比如http-equiv，就是用于嵌入附加信息的一种方式。

所以，有些机器人（爬虫）会搜索html文档的head部分来查找http-equiv信息。

5. User-Agent向导

Web管理者应该记住，会有很多的机器人（爬虫）来访问他们的站点，因此要做好接收机器人（爬虫）的准备。站点要对浏览器类型进行检测，以确保能够支持各种站点的特性。

站点管理者应该设计一个处理机器人（爬虫）请求的策略，比如单独开发一个页面？？？

机器人（爬虫）在爬行时也需要设定浏览器类型，方法就是在首部加上User-Agent:<浏览器信息>。

**三、行为不当的机器人（爬虫）**

不守规矩的机器人（爬虫）会造成很多严重问题。

例如：

1. 失控机器人（爬虫）

机器人（爬虫）发起的HTTP请求的速度比人类要快得多，它们通常都运行在有快速网络连接的高速计算机上，如果它们存在编程逻辑错误，或者陷入了环路之中，可能会向Web服务器发出大量的负载——可能会使服务器过载。

所以机器人（爬虫）编写者都必须小心的设计一些保护措施，以免机器人（爬虫）带来危害。

2.失效的URL

有些机器人（爬虫）会访问URL列表，这些列表比较老了，管理者不会喜欢它们的错误日志中充满了对不存在文档的访问请求，也不希望提供出错页面的开销降低其他服务器的处理能力。

3.爱打听的机器人（爬虫）

有些机器人（爬虫）可能会得到一些指向私有数据的URL，如果数据的所有者没有主动宣传这些Web页面，那么他不会希望机器人（爬虫）访问这些页面的。

应该有某种机制可以将这些数据丢弃。

4.动态网管访问

因为机器人（爬虫）并不总是知道它们访问的是什么内容，所以有可能访问到网关，在这种情况下，计算的开销可能很高，web站点管理员不喜欢那些请求网关文档的幼稚机器人（爬虫）。

**四、拒绝机器人（爬虫）访问**

拒绝机器人（爬虫）访问标准——robots.txt

Robots.txt的思想很简单。所有服务器都可以在服务器的文件根目录提供一个可选的、名为robots.txt的文件。这个文件包含的信息说明了机器人（爬虫）可以访问服务器的那些部分。如果机器人（爬虫）遵循这个自愿约束标准，那么它会在访问这个站点之前，先请求robots.txt文件。

1. 拒绝机器人（爬虫）访问标准

主要介绍1.0版本

2. Web站点和robots.txt文件

A. 获取robots.txt

机器人（爬虫）会用HTTP的GET方法来获取robots.txt资源，如果有这个文件的话，服务器会将其放在一个text/plain主体中返回。如果以404 NOT FOUND状态码响应的话，机器人（爬虫）可以认为没有人限制它的访问，它可以请求任意文件。

例如：

GET /robots.txt HTTP/1.0

Host: [www.joes-hardware.con](http://www.joes-hardware.con)

User-Agent: Slurp/2.0

Date: Wed Oct 3 20:22:48 EST 2001

B.响应码

很多Web站点都没有robots.txt资源，但是机器人（爬虫）并不知道。机器人（爬虫）会根据对robots.txt检索的结果采取不同的行动。

成功（2xx）：需要对返回内容进行解析，并使用排斥规则从站点上获取内容。

不存在（404）：不受robots.txt的限制。

访问限制（401或403）：这个站点的访问是完全受限的。

服务器故障（503）：推迟访问，直到获取该资源为止。

重定向（3xx）：跟着重定向，直到获取该资源为止。

3. robots.txt文件的格式

有三种类型的行：空行、注释行、规则行。

规则行看起来像HTTP首部（<Field>:<value>）一样

例如：

# 这个文件允许机器人（爬虫）Slurp和Webcrawler访问除了private子目录下的那些文件之外的内容

User-Agent: slurp

User-Agent: webcrawler

Disallow: /private

User-Agent: \*

Disallow:

来看看具体内容

A. User-Agent行

每个机器人（爬虫）记录都可以以一个或多个下列形式的User-Agent行开始：

User-Agent: <robot-name>

或

User-Agent: \*

第一个<robot-name>如果不是\*的话，必须是机器人（爬虫）名的大小写无关的子字符串。

B. Disallow和 Allow行

Disallow和 Allow行紧跟在机器人（爬虫）排斥记录的User-Agent行之后。用来说明禁止或允许特定机器人（爬虫）使用那些URL路径。

如果没有找到匹配项，就说明允许使用这个URL

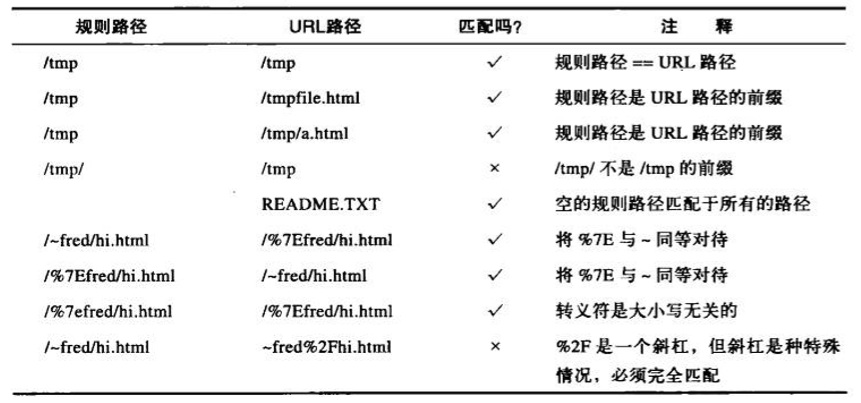
要使Disallow/Allow行与一个URL相匹配，那么需要URL路径大小写相关的前缀。如：Disallow: /tmp就和下面所有的匹配

<http://www.xxx.com/tmp>

<http://www.xxx.com/tmp/pliers.html>

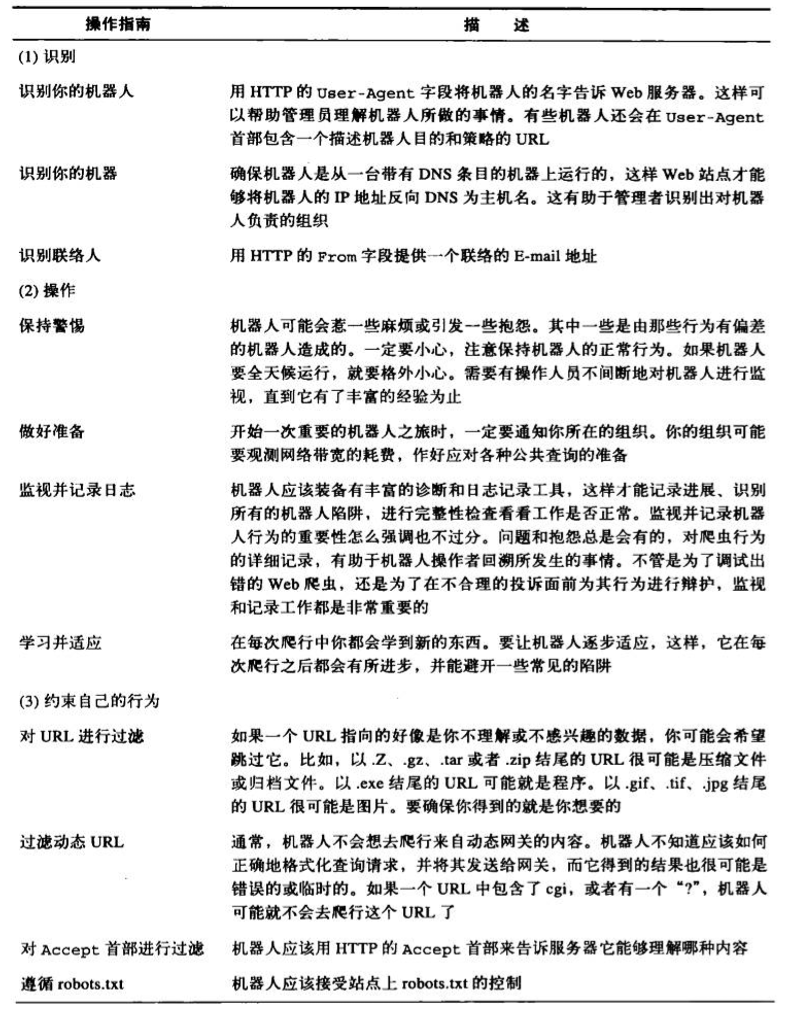
<http://www.xxx.com/tmpspc/stuff.txt>

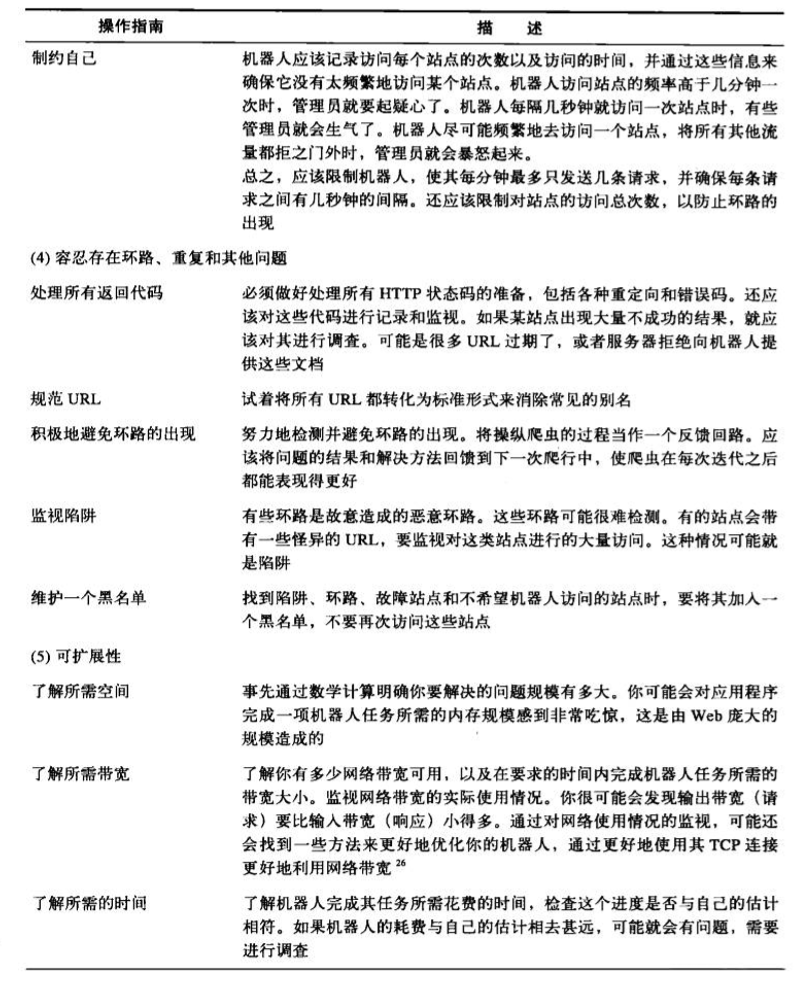
C. Disallow/Allow前缀匹配

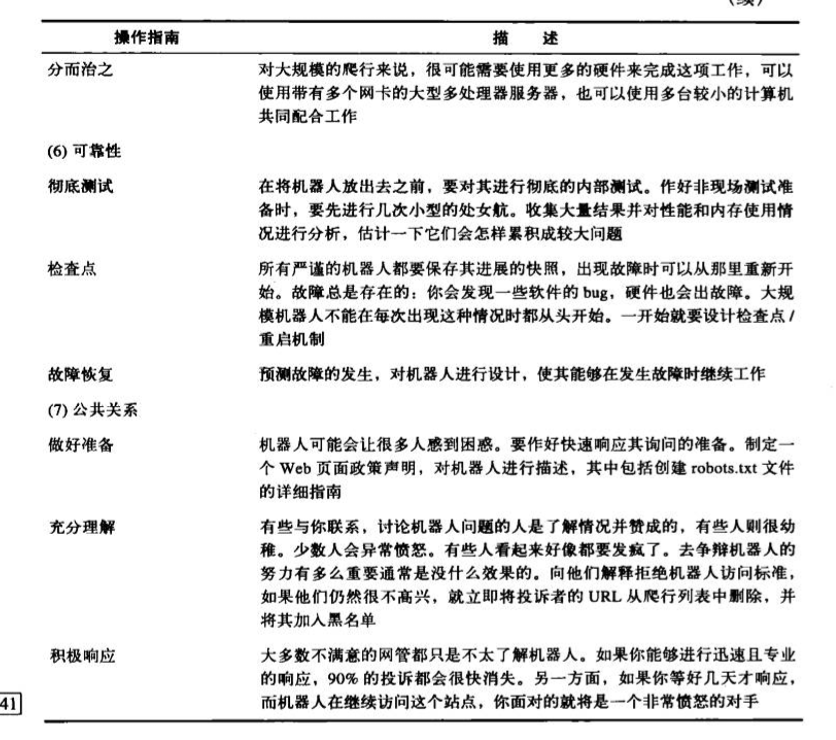


**五、机器人（爬虫）规范**

如下表：







**六、信息**

[**http://www.robotstxt.org/wc/robots.html**](http://www.robotstxt.org/wc/robots.html)

**Web机器人（爬虫）页面——机器人（爬虫）开发者所需的资源，包括网络机器人（爬虫）的注册。**

[**http://www.searchengineworld.com**](http://www.searchengineworld.com)

**搜索引擎世界——搜索引擎和机器人（爬虫）的相关资源**

[**http://www.searchtools.com**](http://www.searchtools.com)

**搜索工具和机器人（爬虫）的相关资源。**