# 第六章 爬虫

## 一．常用库与模块

#### **试列出至少三种目前流行的大型数据库的名称: 、 、 , 其中您最熟悉的是 ,从 年开始使用**

（考察对数据可的熟悉程度，同时考察你的工作年限注意和自己简历一致）。Oracle，Mysql， SQLServer、MongoDB 根据自己情况（推荐 Mysql 、MongoDB）。

#### **列举您使用过的Python 网络爬虫所用到的网络数据包?**

requests、urllib、urllib2、httplib2。

#### **列举您使用过的Python 网络爬虫所用到的解析数据包**

BeautifulSoup、pyquery、Xpath、lxml。

#### **爬取数据后使用哪个数据库存储数据的，为什么？**

MongoDB 是使用比较多的数据库，这里以 MongoDB 为例，大家需要结合自己真实开发环境回答。

原因：1）与关系型数据库相比，MongoDB 的优点如下。

①弱一致性（最终一致），更能保证用户的访问速度

举例来说，在传统的关系型数据库中，一个 COUNT 类型的操作会锁定数据集，这样可以保证得到“当前”情况下的较精确值。这在某些情况下，例 如通过 ATM 查看账户信息的时候很重要， 但对于 Wordnik 来说，数据是不断更新和增长的，这种“较精确”的保证几乎没有任何意义，反而

158

会产生很大的延 迟。他们需要的是一个“大约”的数字以及更快的处理速度。

但某些情况下 MongoDB 会锁住数据库。如果此时正有数百个请求，则它们会堆积起来，造成许多问题。我们使用了下面的优化方式来避免锁定。

每次更新前，我们会先查询记录。查询操作会将对象放入内存，于是更新则会尽可能的迅速。在主/ 从部署方案中，从节点可以使用“-pretouch”参数运行，这也可以得到相同的效果。

使用多个 mongod 进程。我们根据访问模式将数据库拆分成多个进程。

②文档结构的存储方式，能够更便捷的获取数据。

对于一个层级式的数据结构来说，如果要将这样的数据使用扁平式的，表状的结构来保存数据， 这无论是在查询还是获取数据时都十分困难。

③内置 GridFS，支持大容量的存储。

GridFS 是一个出色的分布式文件系统，可以支持海量的数据存储。内置了 GridFS 了 MongoDB， 能够满足对大数据集的快速范围查询。

④内置 Sharding。

提供基于 Range 的 Auto Sharding 机制：一个 collection 可按照记录的范围，分成若干个段， 切分到不同的Shard 上。Shards 可以和复制结合，配合 Replica sets 能够实现Sharding+fail-over， 不同的 Shard 之间可以负载均衡。查询是对 客户端是透明的。客户端执行查询，统计，MapReduce 等操作，这些会被 MongoDB 自动路由到后端的数据节点。这让我们关注于自己的业务，适当的 时候可以无痛的升级。MongoDB 的 Sharding 设计能力较大可支持约 20 petabytes，足以支撑一般应用。

这可以保证 MongoDB 运行在便宜的 PC 服务器集群上。PC 集群扩充起来非常方便并且成本很低， 避免了“sharding”操作的复杂性和成本。

⑤第三方支持丰富。(这是与其他的 NoSQL 相比，MongoDB 也具有的优势)

现在网络上的很多 NoSQL 开源数据库完全属于社区型的，没有官方支持，给使用者带来了很大的风险。而开源文档数据库 MongoDB 背后有商业公司 10gen 为其提供供商业培训和支持。

而且 MongoDB 社区非常活跃，很多开发框架都迅速提供了对 MongDB 的支持。不少知名大公司和网站也在生产环境中使用 MongoDB，越来越多的创新型企业转而使用 MongoDB 作为和Django，RoR 来搭配的技术方案。

⑥性能优越

在使用场合下，千万级别的文档对象，近 10G 的数据，对有索引的 ID 的查询不会比 mysql 慢， 而对非索引字段的查询，则是全面胜出。 mysql 实际无法胜任大数据量下任意字段的查询，而mongodb 的查询性能实在让我惊讶。写入性能同样很令人满意，同样写入百万级别的数 据， mongodb 比我以前试用过的 couchdb 要快得多，基本 10 分钟以下可以解决。补上一句，观察过程中 mongodb 都远算不上是 CPU 杀手。

2)Mongodb 与 redis 相比较

①mongodb 文件存储是 BSON 格式类似 JSON，或自定义的二进制格式。

mongodb 与 redis 性能都很依赖内存的大小，mongodb 有丰富的数据表达、索引；最类似于关系数据库，支持丰富的查询语言，redis 数据丰富，较少的 IO ，这方面 mongodb 优势明显。

②mongodb 不支持事物，靠客户端自身保证，redis 支持事物，比较弱，仅能保证事物中的操作按顺序执行，这方面 redis 优于 mongodb。

③mongodb 对海量数据的访问效率提升，redis 较小数据量的性能及运算,这方面 mongodb 性能优于redis .monbgodb 有mapredurce 功能，提供数据分析，redis 没有 ，这方面 mongodb优 于 redis 。

#### **你用过的爬虫框架或者模块有哪些？谈谈他们的区别或者优缺点？**

Python 自带：urllib、urllib2 第三方：requests

框架： Scrapy

urllib 和 urllib2 模块都做与请求 URL 相关的操作，但他们提供不同的功能。urllib2：urllib2.urlopen 可以接受一个 Request 对象或者 url，（在接受 Request 对象时候，并以

此可以来设置一个 URL 的 headers），urllib.urlopen 只接收一个 url。

urllib 有 urlencode,urllib2 没有，因此总是 urllib，urllib2 常会一起使用的原因。

scrapy 是封装起来的框架，它包含了下载器，解析器，日志及异常处理，基于多线程，twisted 的方式处理，对于固定单个网站的爬取开发，有优势，但是对于多网站爬取，并发及分布式处理方面，不够灵活，不便调整与括展。

request 是一个 HTTP 库， 它只是用来，进行请求，对于 HTTP 请求，他是一个强大的库，下载， 解析全部自己处理，灵活性更高，高并发与分布式部署也非常灵活，对于功能可以更好实现

Scrapy 优点：

scrapy 是异步的；

采取可读性更强的 xpath 代替正则； 强大的统计和 log 系统；

同时在不同的 url 上爬行；

支持 shell 方式，方便独立调试；

写 middleware,方便写一些统一的过滤器； 通过管道的方式存入数据库；

Scrapy 缺点：

基于 python 的爬虫框架，扩展性比较差；

基于 twisted 框架，运行中的 exception 是不会干掉 reactor，并且异步框架出错后是不会停掉其他任务的，数据出错后难以察觉。

#### **写爬虫是用多进程好？还是多线程好？ 为什么？**

IO 密集型代码(文件处理、网络爬虫等)，多线程能够有效提升效率(单线程下有 IO 操作会进行 IO 等待，造成不必要的时间浪费，而开启多线程能在线程 A 等待时，自动切换到线程 B，可以不浪费 CPU 的资源，从而能提升程序执行效率)。在实际的数据采集过程中，既考虑网速和响应的问题，也需要考虑自身机器的硬件情况，来设置多进程或多线程。

#### **常见的反爬虫和应对方法？**

通过 Headers 反爬虫：

从用户请求的 Headers 反爬虫是最常见的反爬虫策略。很多网站都会对 Headers 的 User-Agent 进行检测，还有一部分网站会对 Referer 进行检测（一些资源网站的防盗链就是检测 Referer）。如果遇到了这类反爬虫机制，可以直接在爬虫中添加 Headers，将浏览器的 User-Agent 复制到爬虫的Headers 中；或者将 Referer 值修改为目标网站域名。对于检测 Headers 的反爬虫，在爬虫中修改或者添加 Headers 就能很好的绕过。

基于用户行为反爬虫：

还有一部分网站是通过检测用户行为，例如同一 IP 短时间内多次访问同一页面，或者同一账户短时间内多次进行相同操作。

大多数网站都是前一种情况，对于这种情况，使用 IP 代理就可以解决。可以专门写一个爬虫，爬取网上公开的代理 ip，检测后全部保存起来。这样的代理 ip 爬虫经常会用到，最好自己准备一个。有了大量代理 ip 后可以每请求几次更换一个 ip，这在 requests 或者 urllib2 中很容易做到，这样就能很容易的绕过第一种反爬虫。

对于第二种情况，可以在每次请求后随机间隔几秒再进行下一次请求。有些有逻辑漏洞的网站，可以通过请求几次，退出登录，重新登录，继续请求来绕过同一账号短时间内不能多次进行相同请求的限制。

动态页面的反爬虫：

上述的几种情况大多都是出现在静态页面，还有一部分网站，我们需要爬取的数据是通过 ajax 请求得到，或者通过 JavaScript 生成的。首先用 Fiddler 对网络请求进行分析。如果能够找到 ajax 请求， 也能分析出具体的参数和响应的具体含义，我们就能采用上面的方法，直接利用 requests 或者 urllib2 模拟 ajax 请求，对响应的 json 进行分析得到需要的数据。

能够直接模拟 ajax 请求获取数据固然是极好的，但是有些网站把 ajax 请求的所有参数全部加密了。我们根本没办法构造自己所需要的数据的请求。这种情况下就用 selenium+phantomJS，调用浏览器内核，并利用 phantomJS 执行 js 来模拟人为操作以及触发页面中的 js 脚本。从填写表单到点击按钮再到滚动页面，全部都可以模拟，不考虑具体的请求和响应过程，只是完完整整的把人浏览页面获取数据的过程模拟一遍。

用这套框架几乎能绕过大多数的反爬虫，因为它不是在伪装成浏览器来获取数据（上述的通过添加Headers 一定程度上就是为了伪装成浏览器），它本身就是浏览器，phantomJS 就是一个没有界面的浏览器，只是操控这个浏览器的不是人。利 selenium+phantomJS 能干很多事情，例如识别点触式

（12306）或者滑动式的验证码，对页面表单进行暴力破解等。

#### **解析网页的解析器使用最多的是哪几个?**

lxml，html5lib，html.parser,lxml-xml，正则表达式。

#### **需要登录的网页，如何解决同时限制ip，cookie,session（其中有一些是动态生成的）在不使用动态爬取的情况下？**

解决限制 IP 可以使用代理 IP 地址池、服务器；

不适用动态爬取的情况下可以使用反编译 JS 文件获取相应的文件，或者换用其他平台（比如手机端） 看看是否可以获取相应的 json 文件。

#### **10. 验证码的解决?**

图形验证码：干扰、杂色不是特别多的图片可以使用开源库 Tesseract 进行识别，太过复杂的需要借助第三方打码平台。

点击和拖动滑块验证码可以借助 selenium、无图形界面浏览器（chromedirver 或者 phantomjs） 和 pillow 包来模拟人的点击和滑动操作，pillow 可以根据色差识别需要滑动的位置。

#### **11. 使用最多的数据库（Mysql，Mongodb，redis 等），对他们的理解？**

MySQL 数据库：开源免费的关系型数据库，需要实现创建数据库、数据表和表的字段，表与表之间可以进行关联（一对多、多对多），是持久化存储。

Mongodb 数据库：是非关系型数据库，数据库的三元素是，数据库、集合、文档，可以进行持久化存储，也可作为内存数据库，存储数据不需要事先设定格式，数据以键值对的形式存储。

redis 数据库：非关系型数据库，使用前可以不用设置格式，以键值对的方式保存，文件格式相对自由，主要用与缓存数据库，也可以进行持久化存储。

#### **12. 字符集和字符编码**

字符是各种文字和符号的总称，包括各个国家文字、标点符号、图形符号、数字等。

字符集是多个字符的集合，字符集种类较多，每个字符集包含的字符个数不同，常见字符集有： ASCII 字符集、ISO 8859 字符集、GB2312 字符集、BIG5 字符集、GB18030 字符集、Unicode 字符集等。

字符编码就是以二进制的数字来对应字符集的字符。常见的编码字符集（简称字符集）如下所示：

Unicode：也叫统一字符集，它包含了几乎世界上所有的已经发现且需要使用的字符

（如中文、日文、英文、德文等）。

ASCII：ASCII 既是编码字符集，又是字符编码。早期的计算机系统只能处理英文，所以 ASCII 也就成为了计算机的缺省字符集，包含了英文所需要的所有字符。

GB2312：中文字符集，包含 ASCII 字符集。ASCII 部分用单字节表示，剩余部分用双字节表示。

GBK：GB2312 的扩展，但完整包含了 GB2312 的所有内容。

GB18030：GBK 字符集的超集，常叫大汉字字符集，也叫 CJK（Chinese，Japanese， Korea）字符集，包含了中、日、韩三国语。

注意：Unicode 字符集有多种编码方式，如 UTF-8、UTF-16 等；ASCII 只有一种；大多数 MBCS（包括 GB2312）也只有一种。

#### **写一个邮箱地址的正则表达式？**

[A-Za-z0-9\_-]+@[a-zA-Z0-9\_-]+(\.[a-zA-Z0-9\_-]+)+$

#### **编写过哪些爬虫中间件？**

user-agent、代理池等。

#### **15. “极验”滑动验证码如何破解？**

* 1. selenium 控制鼠标实现，速度太机械化，成功率比较低

|  |
| --- |
|  |
|  |  |

* 1. 计 算 缺 口 的 偏 移 量 （ 推 荐 博 客 ：<http://blog.csdn.net/paololiu/article/details/52514504?%3E>

* 1. “极验”滑动验证码需要具体网站具体分析，一般牵扯算法乃至深度学习相关知识。

#### **爬的那些内容数据量有多大，多久爬一次，爬下来的数据是怎么存储？**

京东整站的数据大约在 1 亿左右，爬下来的数据存入数据库，mysql 数据库中如果有重复的 url 建议去重存入数据库，可以考虑引用外键。评分，评论如果做增量，Redis 中 url 去重，评分和评论建议建立一张新表用 id 做关联。

多久爬一次这个问题要根据公司的要求去处理，不一定是每天都爬。

Mongo 建立唯一索引键（id）可以做数据重复 前提是数据量不大 2 台电脑几百万的情况 数据库需要做分片 （数据库要设计合理）。

例：租房的网站数据量每天大概是几十万条 ，每周固定爬取。

#### **cookie 过期的处理问题？**

因为 cookie 存在过期的现象，一个很好的处理方法就是做一个异常类，如果有异常的话 cookie 抛出异常类在执行程序。

#### **动态加载又对及时性要求很高怎么处理？**

Selenium+Phantomjs

尽量不使用 sleep 而使用 WebDriverWait

关于 HTTP/HTTPS 的区别，分别应该在什么场合下。

#### **HTTPS 有什么优点和缺点**

优点：1、使用 HTTPS 协议可认证用户和服务器，确保数据发送到正确的客户机和服务器；

2、HTTPS 协议是由 SSL+HTTP 协议构建的可进行加密传输、身份认证的网络协议，要比 http 协议安全，可防止数据在传输过程中不被窃取、改变，确保数据的完整性。

3、HTTPS 是现行架构下最安全的解决方案，虽然不是绝对安全，但它大幅增加了中间人攻击

的成本

缺点：

* 1. HTTPS 协议的加密范围也比较有限，在黑客攻击、拒绝服务攻击、服务器劫持等方面几乎起不到什么作用
  2. HTTPS 协议还会影响缓存，增加数据开销和功耗，甚至已有安全措施也会受到影响也会因此而受到影响。
  3. SSL 证书需要钱。功能越强大的证书费用越高。个人网站、小网站没有必要一般不会用。
  4. HTTPS 连接服务器端资源占用高很多，握手阶段比较费时对网站的相应速度有负面影响。
  5. HTTPS 连接缓存不如 HTTP 高效。

#### **HTTPS 是如何实现安全传输数据的。**

HTTPS 其实就是在HTTP 跟TCP 中间加多了一层加密层TLS/SSL。SSL 是个加密套件，负责对 HTTP 的数据进行加密。TLS 是 SSL 的升级版。现在提到 HTTPS，加密套件基本指的是 TLS。原先是应用层将数据直接给到 TCP 进行传输，现在改成应用层将数据给到 TLS/SSL，将数据加密后，再给到 TCP 进行传输。

#### **21.****TTL，MSL，RTT？**

MSL：报文最大生存时间”，他是任何报文在网络上存在的最长时间，超过这个时间报文将被丢弃。

TTL：TTL 是 time to live 的缩写，中文可以译为“生存时间”，这个生存时间是由源主机设置初始值但不是存的具体时间，而是存储了一个 ip 数据报可以经过的最大路由数，每经过一个处理他的路由器此值就减 1，当此值为 0 则数据报将被丢弃，同时发送 ICMP 报文通知源主机。RFC 793 中规定 MSL 为 2 分钟，实际应用中常用的是 30 秒，1 分钟和 2 分钟等。TTL 与 MSL 是有关系的但不是简单的相等的关系，MSL 要大于等于 TTL。

RTT： RTT 是客户到服务器往返所花时间（round-trip time，简称 RTT），TCP 含有动态估算 RTT 的算法。TCP 还持续估算一个给定连接的 RTT，这是因为 RTT 受网络传输拥塞程序的变化而变化。

#### **谈一谈你对Selenium 和PhantomJS 了解**

Selenium 是一个 Web 的自动化测试工具，可以根据我们的指令，让浏览器自动加载页面，获取需要的数据，甚至页面截屏，或者判断网站上某些动作是否发生。Selenium 自己不带浏览器，不支持浏览器的功能，它需要与第三方浏览器结合在一起才能使用。但是我们有时候需要让它内嵌在代码中运行， 所以我们可以用一个叫 PhantomJS 的工具代替真实的浏览器。Selenium 库里有个叫 WebDriver 的API。WebDriver 有点儿像可以加载网站的浏览器，但是它也可以像 BeautifulSoup 或者其他Selector 对象一样用来查找页面元素，与页面上的元素进行交互 (发送文本、点击等)，以及执行其他动作来运行网络爬虫。

PhantomJS 是一个基于 Webkit 的“无界面”(headless)浏览器，它会把网站加载到内存并执行页面上的 JavaScript，因为不会展示图形界面，所以运行起来比完整的浏览器要高效。相比传统的 Chrome或 Firefox 浏览器等，资源消耗会更少。

如果我们把 Selenium 和 PhantomJS 结合在一起，就可以运行一个非常强大的网络爬虫了，这个爬虫可以处理 JavaScript、Cookie、headers，以及任何我们真实用户需要做的事情。

主程序退出后，selenium 不保证 phantomJS 也成功退出，最好手动关闭 phantomJS 进程。（有可能会导致多个 phantomJS 进程运行，占用内存）。

WebDriverWait 虽然可能会减少延时，但是目前存在 bug（各种报错），这种情况可以采用 sleep。phantomJS 爬数据比较慢，可以选择多线程。如果运行的时候发现有的可以运行，有的不能，可

以尝试将 phantomJS 改成 Chrome。

#### **代理 IP 里的“透明”“匿名”“高匿”分别是指？**

透明代理的意思是客户端根本不需要知道有代理服务器的存在，但是它传送的仍然是真实的 IP。你要想隐藏的话，不要用这个。

普通匿名代理能隐藏客户机的真实 IP，但会改变我们的请求信息，服务器端有可能会认为我们使用了代理。不过使用此种代理时，虽然被访问的网站不能知道你的 ip 地址，但仍然可以知道你在使用代理，当然某些能够侦测 ip 的网页仍然可以查到你的 ip。

高匿名代理不改变客户机的请求，这样在服务器看来就像有个真正的客户浏览器在访问它，这时客户的真实 IP 是隐藏的，服务器端不会认为我们使用了代理。

设置代理有以下两个好处：

1，让服务器以为不是同一个客户端在请求

2，防止我们的真实地址被泄露，防止被追究

#### **requests 返回的content 和text 的区别？**

1. response.text 返 回 的 是 Unicode 型 数 据 ； a)response.content 返回的是 bytes 类型，也就是二进制数据；

1. 获取文本使用，response.text；
2. 获取图片,文件，使用 response.content； c)response.text

类型：str

解码类型： 根据 HTTP 头部对响应的编码作出有根据的推测，推测的文本编码如何修改编码方式：response.encoding=”gbk”

1. response.content 类型：bytes

解码类型： 没有指定

如何修改编码方式：response.content.decode(“utf8”)

#### **25.** **robots 协议**

Robots 协议：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。

#### **为什么requests 请求需要带上header？**

原因是：模拟浏览器，欺骗服务器，获取和浏览器一致的内容header 的形式：字典

headers = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/54.0.2840.99 Safari/537.36"}

用法： requests.get(url,headers=headers)

#### **dumps,loads 与dump,load 的区别？**

json.dumps()将 pyhton 的 dict 数据类型编码为 json 字符串； json.loads()将 json 字符串解码为 dict 的数据类型；

json.dump(x,y) x 是 json 对象, y 是文件对象，最终是将 json 对象写入到文件中； json.load(y) 从文件对象 y 中读取 json 对象。

#### **通用爬虫 ：通常指搜索引擎的爬虫**

聚焦爬虫 ：针对特定网站的爬虫

|  |
| --- |
|  |
|  |  |

通用搜素引擎的局限性：

通用搜索引擎所返回的网页里 90%的内容无用。

图片、数据库、音频、视频多媒体的内容通用搜索引擎无能为力。不同用户搜索的目的不全相同，但是返回内容相同。

#### **requests 使用小技巧**

1、reqeusts.util.dict\_from\_cookiejar 把 cookie 对象转化为字典

1.1. requests.get(url,cookies={}) 2、设置请求不用 SSL 证书验证

response = requests.get("https:[//www.12306.cn/mormhweb/](http://www.12306.cn/mormhweb/) ", verify=False) 3、设置超时

response = requests.get(url,timeout=10) 4、配合状态码判断是否请求成功

assert response.status\_code == 200

#### **30. 平常怎么使用代理的 ？**

1. 自己维护代理池
2. 付费购买（目前市场上有很多 ip 代理商，可自行百度了解，建议看看他们的接口文档

（API&SDK））

#### **31.** **IP 存放在哪里？怎么维护IP？对于封了多个ip 的，怎么判定IP 没被封？**

存放在数据库(redis、mysql 等)。维护多个代理网站：

一般代理的存活时间往往在十几分钟左右，定时任务，加上代理 IP 去访问网页，验证其是否可用， 如果返回状态为 200，表示这个代理是可以使用的。

#### **32. 怎么获取加密的数据？**

1. Web 端加密可尝试移动端（app）
2. 解析加密，看能否破解
3. 反爬手段层出不穷，js 加密较多，只能具体问题具体分析

#### **33. 假如每天爬取量在 5、6 万条数据，一般开几个线程，每个线程ip 需要加锁限定吗？**

1. 5、6 万条数据相对来说数据量比较小，线程数量不做强制要求(做除法得一个合理值即可）
2. 多线程使用代理，应保证不在同时一刻使用一个代理 IP

#### **34. 怎么监控爬虫的状态**

1. 使用 python 的 STMP 包将爬虫的状态信心发送到指定的邮箱
2. Scrapyd、pyspider
3. 引入日志



## 二．Scrapy

#### **1.** **描述下scrapy 框架运行的机制？**

从 start\_urls 里获取第一批 url 并发送请求，请求由引擎交给调度器入请求队列，获取完毕后，调度器将请求队列里的请求交给下载器去获取请求对应的响应资源，并将响应交给自己编写的解析方法做提取处理：

1. 如果提取出需要的数据，则交给管道文件处理；
2. 如果提取出 url，则继续执行之前的步骤（发送 url 请求，并由引擎将请求交给调度器入队列...)， 直到请求队列里没有请求，程序结束。

#### **2.** **谈谈你对Scrapy 的理解？**

scrapy 是一个为了爬取网站数据，提取结构性数据而编写的应用框架，我们只需要实现少量代码， 就能够快速的抓取到数据内容。Scrapy 使用了 Twisted 异步网络框架来处理网络通讯，可以加快我们的下载速度，不用自己去实现异步框架，并且包含了各种中间件接口，可以灵活的完成各种需求。scrapy 框架的工作流程：

1. 首先 Spiders（爬虫）将需要发送请求的 url(requests)经 ScrapyEngine（引擎）交给 Scheduler

（调度器）。

1. Scheduler（排序，入队）处理后，经 ScrapyEngine，DownloaderMiddlewares(可选，主要有 User\_Agent， Proxy 代理)交给 Downloader。
2. Downloader 向互联网发送请求，并接收下载响应（response）。将响应（response）经ScrapyEngine，SpiderMiddlewares(可选)交给 Spiders。

1. Spiders 处理 response，提取数据并将数据经 ScrapyEngine 交给 ItemPipeline 保存（可以是本地，可以是数据库）。提取 url 重新经 ScrapyEngine 交给 Scheduler 进行下一个循环。直到无 Url 请求程序停止结束。

#### **怎么样让 scrapy 框架发送一个 post 请求（具体写出来）**

|  |
| --- |
|  |
|  |  |

使 用 FormRequest

#### **怎么判断网站是否更新？**

使用 MD5 数字签名：

每次下载网页时，把服务器返回的数据流 ResponseStream 先放在内存缓冲区，然

后对 ResponseStream 生成 MD5 数字签名 S1，下次下载同样生成签名 S2，比较 S2 和 S1， 如果相同，则页面没有更新，否则网页就有更新。

#### **图片、视频爬取怎么绕过防盗连接，或者说怎么获取正确的链接地址？**

自定义 Referer(建议自行 Google 相关知识)。

#### **你爬出来的数据量大概有多大？大概多长时间爬一次？**

无标准答案，根据自己爬取网站回答即可（几百万，几千万，亿级）。

#### **7.** **增量爬取**

增量爬取即保存上一次状态，本次抓取时与上次比对，如果不在上次的状态中，便视为增量，保存下来。对于 scrapy 来说，上一次的状态是抓取的特征数据和上次爬取的request 队列（url 列表），request 队列可以通过 request 队列可以通过

scrapy.core.scheduler 的 pending\_requests 成员得到，在爬虫启动时导入上次爬取的特征数据，并且用上次 request 队列的数据作为 start url 进行爬取，不在上一次状态中的数据便保存。

选用 BloomFilter 原因：对爬虫爬取数据的保存有多种形式，可以是数据库，可以是磁盘文件等，不管是数据库，还是磁盘文件，进行扫描和存储都有很大的时间和空间上的开

销，为了从时间和空间上提升性能，故选用 BloomFilter 作为上一次爬取数据的保存。保存的特征数据可以是数据的某几项，即监控这几项数据，一旦这几项数据有变化，便视为增

量持久化下来，根据增量的规则可以对保存的状态数据进行约束。比如：可以选网页更新的时间，索引次数或是网页的实际内容，cookie 的更新等。

#### **爬虫向数据库存数据开始和结束都会发一条消息，是scrapy 哪个模块实现的？**

#### 

Scrapy 使用信号来通知事情发生，因此答案是 signals 模块。

#### **爬取下来的数据如何去重，说一下具体的算法依据**

1. 通过 MD5 生成电子指纹来判断页面是否改变
2. nutch 去重。nutch 中 digest 是对采集的每一个网页内容的 32 位哈希值，如果两个网页内容完全一样，它们的 digest 值肯定会一样。

数据量不大时，可以直接放在内存里面进行去重，python 可以使用 set()进行去重。当去重数据需要持久化时可以使用 redis 的 set 数据结构。

当数据量再大一点时，可以用不同的加密算法先将长字符串压缩成 16/32/40 个字符，再使用上面两种方法去重。

当数据量达到亿（甚至十亿、百亿）数量级时，内存有限，必须用“位”来去重，才能够满足需求。Bloomfilter 就是将去重对象映射到几个内存“位”，通过几个位的 0/1 值来判断一个对象是否已经存在。

然而 Bloomfilter 运行在一台机器的内存上，不方便持久化（机器 down 掉就什么都没啦），也不方便分布式爬虫的统一去重。如果可以在 Redis 上申请内存进行 Bloomfilter，以上两个问题就都能解决了。

simhash 最牛逼的一点就是将一个文档，最后转换成一个 64 位的字节，暂且称之为特征字，然后判断重复只需要判断他们的特征字的距离是不是<n（根据经验这个 n 一般取值为 3），就可以判断两个文档是否相似。

可见 scrapy\_redis 是利用 set 数据结构来去重的，去重的对象是 request 的 fingerprint（其实就是用 hashlib.sha1()对 request 对象的某些字段信息进行压缩）。其实 fp 就是 request 对象加密压缩后的一个字符串（40 个字符，0~f）。

#### **10.****Scrapy 的优缺点?**

优点：

1. scrapy 是异步的
2. 采取可读性更强的 xpath 代替正则
3. 强大的统计和 log 系统
4. 同时在不同的 url 上爬行
5. 支持 shell 方式，方便独立调试

5）写 middleware,方便写一些统一的过滤器6）通过管道的方式存入数据库

缺点：

1. 基于 python 的爬虫框架，扩展性比较差
2. 基于 twisted 框架，运行中的 exception 是不会干掉 reactor（反应器），并且异步框架出错后是不会停掉其他任务的，数据出错后难以察觉。

#### **11.怎么设置深度爬取?**

通过在 settings.py 中设置 depth\_limit 的值可以限制爬取深度，这个深度是与 start\_urls 中定义 url 的相对值。也就是相对 url 的深度。若定义 url 为[http://www.domz.com/game/,depth\_limit=1](http://www.domz.com/game/%2Cdepth_limit%3D1) 那么限制爬取的只能是此 url 下一级的网页。深度大于设置值的将被忽视。



## 三．Scrapy-redis

#### **scrapy 和scrapy-redis 有什么区别？为什么选择redis 数据库？**

scrapy 是一个 Python 爬虫框架，爬取效率极高，具有高度定制性，但是不支持分布式。而

scrapy-redis 一套基于 redis 数据库、运行在 scrapy 框架之上的组件，可以让 scrapy 支持分布式策略， Slaver 端共享 Master 端 redis 数据库里的 item 队列、请求队列和请求指纹集合。

为什么选择 redis 数据库，因为 redis 支持主从同步，而且数据都是缓存在内存中的，所以基于 redis 的分布式爬虫，对请求和数据的高频读取效率非常高。

#### **分布式爬虫主要解决什么问题？**

* 1. ip 2.带宽3.cpu 4.io

#### **3.** **什么是分布式存储？**

传统定义：分布式存储系统是大量 PC 服务器通过 Internet 互联，对外提供一个整体的服务。分布式存储系统具有以下的几个特性：

可扩展 ：分布式存储系统可以扩展到几百台甚至几千台这样的一个集群规模，系统的

整体性能线性增长。

低成本 ：分布式存储系统的自动容错、自动负载均衡的特性，允许分布式存储系统可以构建在低成本的服务器上。另外，线性的扩展能力也使得增加、减少服务器的成本低， 实现分布式存储系统的自动运维。

180

高性能 ：无论是针对单台服务器，还是针对整个分布式的存储集群，都要求分布式存储系统具备高性能。

易用 ：分布式存储系统需要对外提供方便易用的接口，另外，也需要具备完善的监控、运维工具，并且可以方便的与其他的系统进行集成。

布式存储系统的挑战主要在于数据和状态信息的持久化，要求在自动迁移、自动容错和并发读写的过程中，保证数据的一致性。

容错：可以快速检测到服务器故障，并自动的将在故障服务器上的数据进行迁移。

负载均衡：新增的服务器在集群中保障负载均衡？数据迁移过程中保障不影响现有的服务。事务与并发控制：实现分布式事务。

易用性：设计对外接口，使得设计的系统易于使用。

#### **你所知道的分布式爬虫方案有哪些？**

三种分布式爬虫策略：

* 1. Slaver 端从 Master 端拿任务（Request/url/ID）进行数据抓取，在抓取数据的同时也生成新任务，并将任务抛给 Master。Master 端只有一个 Redis 数据库，负责对 Slaver 提交的任务进行去重、加入待爬队列。

**优点：** scrapy-redis 默认使用的就是这种策略，我们实现起来很简单，因为任务调度

等工作 scrapy-redis 都已经帮我们做好了，我们只需要继承 RedisSpider、指定 redis\_key 就行了。

**缺点：** scrapy-redis 调度的任务是 Request 对象，里面信息量比较大（不仅包含 url， 还有 callback 函数、headers 等信息），导致的结果就是会降低爬虫速度、而且会占用Redis 大量的存储空间。当然我们可以重写方法实现调度 url 或者用户 ID。

* 1. Master 端跑一个程序去生成任务（Request/url/ID）。Master 端负责的是生产任务， 并把任务去重、加入到待爬队列。Slaver 只管从 Master 端拿任务去爬。

**优点**： 将生成任务和抓取数据分开，分工明确，减少了 Master 和 Slaver 之间的数据交流；Master 端生成任务还有一个好处就是：可以很方便地重写判重策略（当数据量大时优化判重的性能和速度还是很重要的）。

**缺点：** 像 QQ 或者新浪微博这种网站，发送一个请求，返回的内容里面可能包含几十个待爬的用户 ID，即几十个新爬虫任务。但有些网站一个请求只能得到一两个新任务，并且返回的内容里也包含爬虫要抓取的目标信息，如果将生成任务和抓取任务分开反而会降低爬虫抓取效率。毕竟带宽也是爬虫的一个瓶颈问题，我们要秉着发送尽量少的请求为原则，同时也是为了减轻网站服务器的压力，要做一只有道德的 Crawler。所以，视情况而定。

* 1. Master 中只有一个集合，它只有查询的作用。Slaver 在遇到新任务时询问 Master 此任务是否已爬，如果未爬则加入 Slaver 自己的待爬队列中，Master 把此任务记为已爬。它和策略一比较像，但明显比策略一简单。策略一的简单是因为有 scrapy-redis 实现了scheduler 中间件，它并不适用于非 scrapy 框架的爬虫。

**优点**： 实现简单，非 scrapy 框架的爬虫也适用。Master 端压力比较小，Master 与Slaver 的数据交流也不大。

**缺点**：“健壮性”不够，需要另外定时保存待爬队列以实现“断点续爬”功能。各 Slaver 的待爬任务不通用。

如果把 Slaver 比作工人，把 Master 比作工头。策略一就是工人遇到新任务都上报给工头，需要干活的时候就去工头那里领任务；策略二就是工头去找新任务，工人只管从工头那里领任务干活；策略三就是工人遇到新任务时询问工头此任务是否有人做了，没有的话工人就将此任务加到自己的“行程

表”。

#### **除了 scrapy-redis，有做过其他的分布式爬虫吗？**

Celery、gearman 等，参考其他分布式爬虫策略。

#### **在爬取的时候遇到某些内容字段缺失怎么判断及处理？**

判读字段缺失，做异常处理即可。

## 四．自定义框架