第5章

互联网数据采集

本章学习目标

- 了解爬虫概念、爬行策略、Robots 协议等基础知识；
- 掌握各种网络爬虫方法；
- 熟练掌握爬虫工具使用方法；
- 能够运用 Python 语言编写爬虫软件。

随着网络的飞速发展，网络已经成为世界上最大的信息载体，每天都有大量的新数据涌入网络。因此，如何从网络中提取出有效的信息并加以利用，是开发人士面对的新课题。互联网中大量的相关信息可以反映用户的偏好倾向、事件发展趋势等。更重要的是，互联网数据以共享开放的形式存储在互联网中，这意味着互联网数据收集的成本往往更低。因此，相关互联网数据的采集与整合几乎成为大数据项目建设的必然选择。从网络中自动抓取信息的最常见、最有效的方法是使用网络爬虫。

5.1 网络爬虫概述


微课视频

随着大数据时代的来临，我们每天面对的数据数不胜数，为了在繁多数据中找到有效的内容，网络爬虫扮演着越来越重要的角色。互联网中的数据是海量的。如何从这些海量信息中自动、高效地获取用户感兴趣的内容，是互联网数据行业面临的重要问题，网络爬虫技术正是为解决这些问题而诞生的。

5.1.1 网络爬虫的基本概念

1. 互联网数据采集面临的困难和挑战

如何在蕴含海量信息的互联网中采集到自己感兴趣的数据，将面临以下的困难和

挑战。

（1）每个门户网站的建设水平不同，每个网站的结构往往因用户体验不同而不同，这意味着通过统一的方法从互联网上收集数据几乎是不可能的。

（2）互联网数据的结构一般比较复杂，通常以文本、表格、图片、视频等非结构化形式存在，这也给互联网数据的采集带来了挑战和困难。

（3）大型互联网公司，如百度，总数据量超过1000PB，覆盖中文网页、百度视频、百度日志等部分，拥有70%以上的中文搜索市场。对于如此海量的数据集，需要研究分布式架构来满足其采集需求。

（4）对于需要从网页获取的互联网数据，可以通过网络爬虫程序自动获取数据，但由于对爬虫程序的监管，不同的网站往往会设置很多障碍，从而增加了互联网数据采集的难度。

2. 网络爬虫的定义

互联网数据的收集通常是借助网络爬虫来完成的。“网络爬虫”（简称爬虫）就是定向或不定向地按照一定的规则，对互联网上的网页数据进行抓取的程序或脚本。抓取网页数据的一般方法是定义一个入口页面，这个页面一般会包含指向其他页面的URL，所以从当前页面获取这些URL，添加到爬虫的爬行队列中，进入新页面后再递归进行上述操作。爬虫数据收集方法可以从网页中提取非结构化数据，将其存储为一个统一的本地数据文件，并以结构化的方式存储。支持图片、音频、视频等文件或附件等各式的信息，附件可以自动与正文关联。

该技术被广泛应用于互联网搜索引擎、信息收集、舆情监测等方面，以获取或更新这些网站的内容和检索方式。同时，这项技术还可以自动收集所有可访问的页面内容，供搜索引擎进一步处理，使用户能够更快地检索到自己需要的信息。

3. 网络爬虫的基本原理

网络爬虫通过网页中的一些超链接信息，辅以一定的算法，按照一定的规则自动采集其所能访问的所有页面内容，为搜索引擎和大数据分析提供数据源，网络爬虫一般具有数据采集、数据处理和数据存储三大功能。

网络爬虫一般从一个或多个初始URL下载网页内容，然后通过搜索或内容匹配的方式获取网页中感兴趣的内容。同时，从当前页面中不断提取新的URL，依照爬虫策略按一定顺序放入待爬取的URL队列中。整个过程循环执行，直到满足系统相应的停止条件，然后对捕获的数据进行清洗、整理、索引并存入数据库或文件中。最后根据查询需要，从数据库或文件中提取相应的数据，以文本或图表的形式显示出来。

4. 网络爬虫的应用

网络爬虫应用广泛。常见的应用包括：

（1）抓取网站上的图片，重点浏览。

（2）抓取相关财务信息，进行投资分析。

(3) 从多个新闻网站上爬取新闻信息,集中阅读。

(4) 利用爬虫对相应网页上的信息进行爬取,自动过滤网页中的广告,方便信息的阅读和使用。

(5) 使用爬虫,可以设置相应的规则,从网上自动收集目标用户的公开信息,便于营销。

(6) 抓取网站用户活跃度、发言次数、热门文章等信息,进行相关分析。

网络爬虫的本质,是一种自动抓取网页的程序,是搜索引擎的重要组成部分。

5. 网络爬虫算法

网络爬虫算法定义爬取范围、如何过滤重复页面等爬行策略的要求和规则。根据不同的目的,选择不同的爬虫算法,爬虫的运行效率和得到的结果也会不同。

1) 基于网络拓扑的分析算法

该算法的思想是,基于网页之间的链接(网络拓扑),对与已知网页有直接或间接链接关系的对象(可以是网页或网站等)作出评价,然后确定爬取的范围和顺序。该算法有三种不同的类型,分别是网页粒度、网站粒度和网页块粒度。

2) 基于网页内容的分析算法

该算法是指利用网页内容(文本、数据等资源)的特征,对网页进行评价,然后确定爬取的范围和顺序。网页的内容已经从超文本发展到动态页面(或隐藏 Web)数据,后者的数据量大约是直接可见页面数据(PIW,公共索引 Web)的 400～500 倍。除此之外,多媒体产生的数据、Web 服务等形式的网络资源日益丰富。因此,这种算法已经从最初单纯的网页文本检索,发展到包括网页数据抽取、机器学习、数据挖掘、语义理解等多方面的综合应用。根据网页数据加载的不同形式,可以将算法分为以下三类:第一类,针对文本和超链接为主的非结构化或简单网页;第二类,针对结构化数据源动态生成的网页,这种页面的数据不能直接批量访问;第三类,介于前两种之间,具有良好的结构,遵循一定的模式和规律,可以直接访问的网页。

5.1.2　网络爬虫的爬行策略

在使用网络爬虫时,URL 队列是一个非常重要的部分,待抓取队列中的 URL 按什么顺序排列是一个非常重要的问题,这就涉及网络爬虫的抓取策略。

网络爬虫的抓取策略是指,在使用爬虫系统时,待抓取 URL 队列中 URL 排序的方法。不同的网页抓取策略会对应不同的网页抓取算法,抓取效率会根据算法不同有所不同。常见的爬虫抓取策略有深度优先策略、广度优先策略、本地 PageRank 策略、OPIC 策略、大站优先策略、反向链接数策略和最佳优先搜索策略。

1. 深度优先策略

深度优先策略是按照深度从低到高的顺序,逐一访问下一层的网页链接,直到无法打开下一层的网页链接为止。在完成一个爬行分支后,爬虫返回到上一个链接节点,进一步搜索其他分支的链接。当遍历所有链接时,爬网任务结束。例如,见图 5-1 的网页链接关

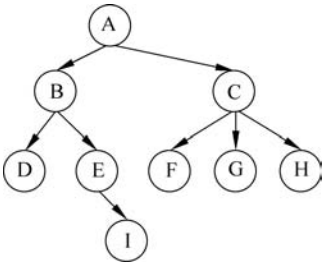

图 5-1　网页链接关系示意图

系，根据深度优先策略，爬虫的爬行顺序为 A→B→D→E→I→C→F→G→H。

深度优先策略更适合垂直搜索或站内搜索，但当爬行页面包含的内容层次较深时，会造成巨大的资源浪费。

2. 广度优先策略

广度优先策略是按照广度优先的搜索思想，逐层抓取 URL 池中每个 URL 的内容，并将每一层的 URL 添加到 URL 池中，按照广度优先策略继续遍历。图 5-1 的 Web 链接关系，按照广度优先策略的爬行顺序为 A→B→C→D→E→F→G→H→I。由此可见，这种策略属于盲目搜索。它不考虑结果的可能位置，只对整个网络进行彻底搜索，效率较低。但是，若想要覆盖尽可能多的网页，广度优先是一个较好的选择。这个策略多用在主题爬虫中，因为网页离初始 URL 越近，它的主题相关性就越大。

3. 局部 Page Rank 策略

本地 Page Rank 策略基于 Page Rank 的思想，根据一定的网页分析算法，预测候选 URL 与目标页面的相似度，或者与主题的相关性，选择一个或几个评价最好的 URL 进行抓取，即将下载的网页与要抓取的 URL 队列中的 URL 一起组成一个网页集，计算每个页面的 Page Rank 值。计算结束后，将 URL 队列中待抓取的 URL 按照 Page Rank 的大小排列，按顺序抓取页面。但由于网络中广告链接和作弊链接的存在，这种策略容易导致 Page Rank 的值不能完全描述其重要性，从而导致捕获的数据无效。

4. OPIC 策略

OPIC（Online Page Importance Computation）策略实际上是一个页面的重要性评分。在开始时，所有页面都被给予相同的初始“现金”（cash）。当一个页面 P 被下载时，P 的“现金”被分配给从 P 解析出的所有链接，P 的“现金”被清空。要爬网的 URL 队列中的所有页面必须根据“现金”的数量进行排序。与 Page Rank 相比，Page Rank 每次都需要迭代计算，而 OPIC 策略不需要迭代过程。因此，OPIC 计算速度明显比局部 Page Rank 策略快，是一种更好的重要性度量策略，适用于实时计算场景。

5. 大站优先策略

大站优先策略是指将 URL 队列中所有要抓取的网页按照其网站进行分类。对于需要下载页面数量较多的网站，会优先下载。因为大型网站往往包含的页面较多，往往著名企业的网页质量就更高，这种策略会倾向优先加载大型网站。大量的实际应用表明该策略优于深度优先策略。

6. 反向链接数策略

反向链接数指一个网页被其他网页指向的链接数量，表示网页内容被其他网页推荐

的程度。受推荐和认可的指数越高，则被指向的链接数越多，因此很多时候，搜索引擎也会参考这个指标来评估网页的重要性，从而决定不同网页的抓取顺序。

7. 最佳优先搜索策略

最佳优先搜索策略根据一定的网页分析算法，预测候选 URL 与目标网页的相似度，选择一个或几个备选 URL 进行抓取。该策略并不会访问所有网页，而只访问被算法预测为“有用”的页面。因为最佳优先级策略是一种局部最优搜索算法，使用这种策略时，许多相关网页可能会被忽略。因此，并不推荐单一使用最佳优先搜索策略，一般都将最佳优先级与具体应用相结合进行改进，才能跳出局部最优。

在实际应用中，通常会结合几种策略来捕获网络信息。例如，百度蜘蛛的抓取策略是以广度优先策略为主，辅以局部 Page Rank 策略。

5.1.3　Web 更新策略

Internet 中的网页信息是经常更新的，网页更新后，网络爬虫必须对这些网页重新进行爬取。然而，互联网中网页信息的更新速度并不统一；如果网页更新太慢，网络爬虫太频繁，必然会增加爬虫和 Web 服务器的压力；如果网页更新较快，但抓取时间间隔较长，则抓取的内容不能真实反映网页的信息。由此可见，网页的更新率与爬虫的更新率越接近，爬虫的效果就越好。当然，在爬虫服务器资源有限的情况下，爬虫还需要根据相应的策略，使不同的网页具有不同的更新优先级，更新优先级较高的网页会得到更快的抓取响应。

常见的网页更新策略如下。

1. 用户体验策略

通常在搜索引擎查询一个关键词时，结果中会给出大量的网页，这些网页会按照一定的规则进行排名。大多数用户只会关注排名靠前的网页。因此，在服务器资源有限的情况下，爬虫策略会优先更新排名靠前的网页。这种更新策略以用户体验为优先参考，被称为用户体验策略。

在用户体验策略中，爬虫程序会保留相应网页的多个历史版本，并根据多个历史版本的内容更新、搜索质量影响、用户体验等信息进行相应分析，确定这些网页的抓取周期。

2. 历史数据策略

历史数据策略是根据网页的历史更新数据，采用泊松分布建模的方法预测网页的下次更新时间，从而确定网页的下次抓取时间，即更新周期。

以上两种策略都需要历史数据作为依据。但如果一个网页是新的网页，就不会有相应的历史数据，爬虫服务器就需要采用新的更新策略。比较常见的策略是聚类分析策略。

3. 聚类分析策略

聚类分析策略是将聚类分析算法应用于爬虫更新网页的策略。其基本原理是对大量

的网页进行聚类(即根据相似度进行分类)。一般来说,类似网页的更新率是差不多的。聚类后,这些海量网页会被划分为多个类,每个类中的网页具有相似的属性,即它们一般具有相似的更新频率。然后对聚类结果中每个类中的网页进行采样,计算采样网页的平均更新频率,从而确定每个集群的网页抓取频率,见图 5-2。

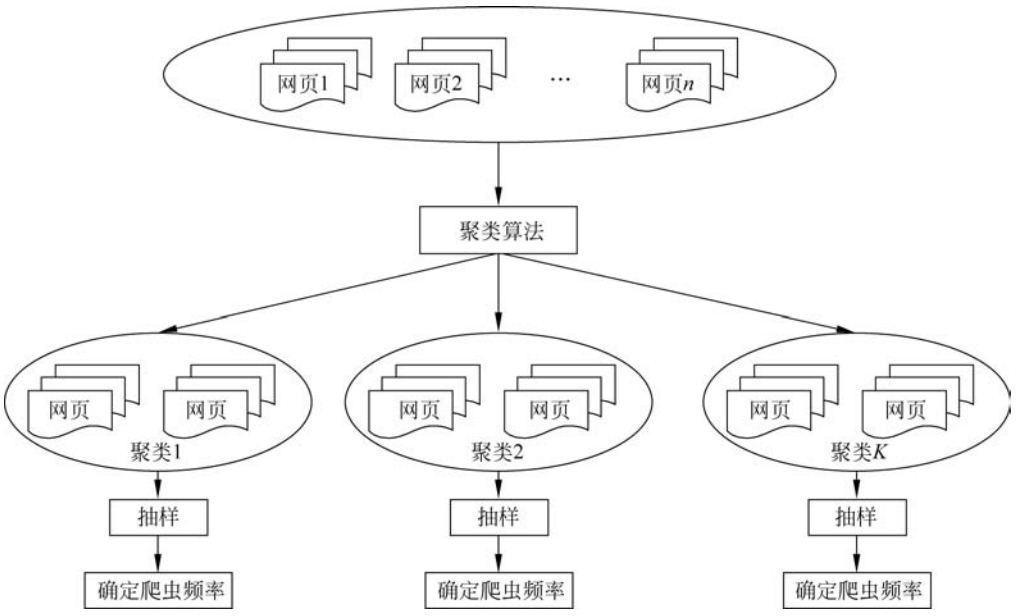

图 5-2　爬虫频率聚类分析策略

在图 5-2 中,利用一定的聚类算法将大量的网页划分为 K 个聚类(K 是由聚类算法确定的)。在图 5-2 的 K 个簇中,每个集群具有相似的更新频率。然后对每个聚类进行采样,提取一些网页,计算这些网页的平均更新频率。最后将每个集群的平均更新频率确定为该集群中所有网页的爬虫频率。

根据网页的更新策略,爬虫可以更高效地执行,执行逻辑更合理。

5.1.4　robots 协议

1. robots 协议简介

robots 是网站和爬虫之间的协议。它以简单直接的 txt 格式文本告诉相应爬虫的允许权限,即 robots.txt 是在搜索引擎中访问网站时首先要查看的文件。爬取某站点信息时,首先在根目录检查是否存在 robots.txt；如果存在,搜索机器人会根据文件内容确定访问范围；如果该文件不存在,所有爬虫将能够访问网站上所有不受密码保护的页面。

robots.txt 是存储在网站根目录中的 ASCII 编码文本文件。它通常会告诉网络搜索引擎的爬虫,网站中哪些内容是爬虫无法获取的,哪些内容是爬虫可以获取的。由于某些系统中的 URL 区分大小写,robots.txt 的文件名应该统一使用小写。robots.txt 应该放在网站的根目录上。

robots 协议不是规范，只是约定，因此不能保证网站的隐私性。注意，robots.txt 使用字符串比较来确定是否获取 URL，因此，目录末尾带有或不带有斜杠“/”的目录，指示的是两个不同的 URL，不能使用“disallow：*.gif”等通配符。

2. robots 协议的使用技巧

(1) 每当用户试图访问一个不存在的 URL 时，服务器就会记录一个 404 错误(文件找不到)。每当访问一个不存在 robots.txt 文件的网站时，服务器也会记录一个 404 错误，所以应该在网站上添加一个 robots.txt 文件。

(2) 一般要让爬虫远离某些服务器上的目录——以保证服务器性能，如大多数网站服务器都有程序存储在 cgi-bin 目录中，所以在 robots.txt 文件中添加 disallow:/cgi-bin 是个不错的主意，这样可以避免索引所有程序文件，节省服务器资源，一般网站中不需要爬取的文件有：后台管理文件、程序脚本、编码文件、样式表文件、模板文件、导航图片和背景图片等。

(3) 如果你的网站是一个动态页面，并且你已经为搜索爬虫创建了这些动态页面的静态副本以便更容易地进行爬取，那么你需要在 robots.txt 文件中进行设置，防止动态页面被爬虫索引，以确保这些页面被认为包含重复内容。

(4) robots.txt 文件也可以直接在站点地图文件中包含一个链接 http://www.***.com/sitemap.xml。

目前支持这项搜索引擎公司包括谷歌、雅虎、Ask 和 MSN。这样做的好处是，网站管理员不需要将他的站点地图文件，提交给每个搜索引擎的网站管理员，搜索引擎会抓取 robots.txt 文件，读取其中的站点地图路径，然后抓取链接的网页。

(5) 适当使用 robots.txt 文件也可以避免访问错误，比如搜索者不能直接进入购物车页面，因为购物车没有被收录的理由，所以可以在 robots.txt 文件中设置，防止搜索者直接进入购物车页面。

3. robots.txt 文件格式

robots.txt 文件包含一个或多个由空行分隔的记录(以 CR、CR/NL 或 NL 终止)，每个记录的格式如下：

```
"<field>: <option space><value><option space>".
```

可以使用 # 在这个文件中进行注释，方法与 UNIX 中的约定相同。文件中的记录通常以一行或多行 User-agent 开头，后跟几行 Disallow 行。

(1) user-agent

user-agent 是用来描述搜索引擎机器人的名称，在 robots.txt 文件中，如果有多个 user-agent 记录，多个机器人就会受到协议的限制，对于这个文件，至少要有一个 user-agent 记录，如果本项的值设置为 *，协议对任何机器人都有效，在 robots.txt 文件中，只能有一个这样的记录“user-agent：*”。

（2）disallow

disallow 用于描述不希望被访问的 URL。此 URL 可以是完整路径，也可以是部分路径。任何以 disallow 开头的 URL 都不会被机器人访问。例如，“disallow：/help”不允许搜索引擎访问/help. HTML 或/help/index. HTML，而“disallow：/help/”允许机器人访问/help. HTML，但不允许访问/help/index. HTML。任何 disallow 记录为空，表示允许访问网站的所有部分。“/robots. txt”文件中必须至少有一条 disallow 记录。如果“/robots. txt”是空文件，则该网站对所有搜索引擎机器人开放。

（3）allow

allow 用来描述可以访问的一组 URL，与 disallow 项类似，这个值可以是一个完整的路径，也可以是路径的前缀，以 allow 项的值开头的 URL 允许机器人访问，例如“allow：/hibaidu”允许机器人访问/hibaidu. htm、/hibaiduca. com. html、/hibaidu/com. html，默认情况下允许一个网站的所有 URL，所以 allow 通常与 disallow 一起使用，允许访问某些网页，同时禁止访问其他所有 URL。

注意 disallow 和 allow 行的顺序是有意义的，并且机器人根据 allow 或 disallow 行的第一个成功匹配来确定是否访问 URL。

（4）使用“*”和“$”

robots 支持使用通配符“*”和“$”模糊匹配 URL，“$”匹配行终止符；“*”匹配 0 个或更多任意字符。


微课视频

5.2 网络爬虫方法

为了解决网络搜索和 Internet 数据收集问题，学者们通过不断的研究和实践，总结出了多种网络爬虫方法。为了研究的方便，这些方法可以按照网络爬虫的功能、系统结构和实现技术进行划分。按照网络爬虫的功能可以分为批量爬虫、增量爬虫和垂直爬虫。根据网络爬虫系统的结构和实现技术，可分为通用网络爬虫、聚焦网络爬虫、深度网络爬虫、分布式网络爬虫等方法。

5.2.1 按功能分类的网络爬虫

1. 批量型爬虫

批量爬虫根据用户配置对网络数据进行爬行。用户通常需要配置的信息包括 URL 或 URL 池、爬虫累计工作时间和爬虫累计获取的数据量等。也就是说，批量爬虫有比较明确的抓取范围和目标。当爬虫到达这个设定的目标时，抓取过程就会停止。该方法适用于 Internet 数据获取的任何场景，通常用于评估算法的可行性和审计目标 URL 数据的可用性。批量爬虫实际上是增量爬虫和垂直爬虫的基础。

2. 增量型爬虫

增量爬虫根据用户的配置对网络数据进行连续爬行。用户通常需要配置的信息包括

URL 或 URL 池、单个 URL 数据爬取频率和数据更新策略。因此，增量爬虫是持续爬行的，被爬行的网页要定期更新，增量爬虫需要及时反映这种变化。这种方法可以实时获取互联网数据，一般的商业搜索引擎基本都采用了这种爬虫技术。

3. 垂直型爬虫

垂直爬虫根据用户配置连续爬行指定的网络数据。用户通常需要配置的信息包括 URL 或 URL 池、敏感词、数据策略等信息。垂直爬虫的关键是如何识别网页内容是否属于指定的行业或主题。从节约系统资源的角度出发，往往要求爬虫在抓取阶段动态识别某个网站是否与主题相关，尽量不抓取无关页面，以达到节约资源的目的。该方法可以实时获取 Internet 中与指定内容相关的数据。垂直搜索网站或垂直行业网站通常会使用这种爬虫技术。

5.2.2　通用网络爬虫

这种爬虫又称全网爬虫。它从一个或几个预设的初始种子 URL 开始，获取初始网页上的 URL 列表。在抓取过程中，它不断地从 URL 队列中获取新的 URL，然后访问和下载页面。页面下载完成后，页面解析器分析网页之后，删除 HTML 标记，获取页面内容，并将摘要、URL 等信息保存到数据库中，提取当前页面上的新 URL 保存到 URL 队列中，直到满足系统停止条件。

通用网络爬虫的主要组成部分包括：初始 URL 集、URL 队列、页面爬行模块、页面分析模块，此外还有页面数据库等其他部分。在进行爬行时会采用一定的爬行策略，主要有深度优先的爬行策略和广度优先的爬行策略。

其工作过程见图 5-3，首先，通用网络爬虫获取初始 URL，初始 URL 地址可以由用户指定，也可由用户指定的一个或几个初始爬行网页确定；其次，根据初始 URL 对页面进行爬取，并获得新的 URL，同时将网页存储在原数据库中，并将爬取网页得到的 URL 地址存储在 URL 列表中；最后，把新的 URL 放入在 URL 队列中；重复上述爬取过程，直到满足条件，停止爬取。

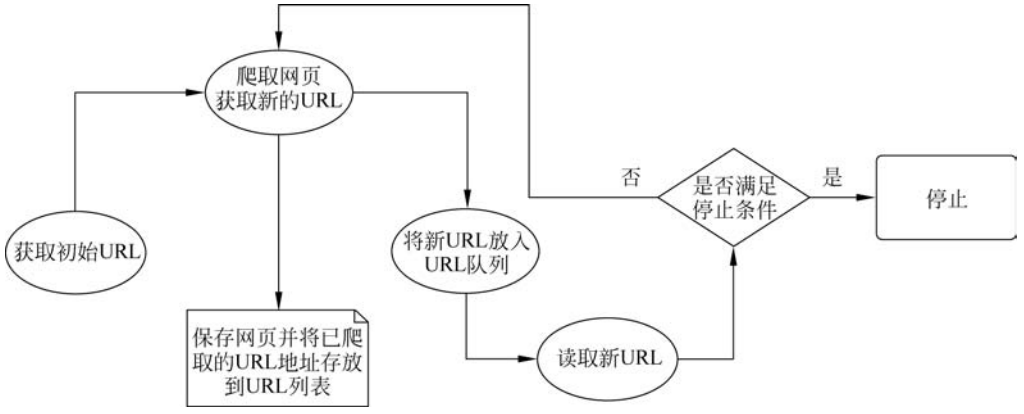

图 5-3　工作流程图

一般的网络爬虫主要是为搜索引擎和大型网站提供商收集数据。由于商业原因，一般网络爬虫的技术细节很少公布。通常这种大型的网络爬虫，爬行范围和数量都有一定规模，对爬行速度和存储空间有着一定的要求，而对爬行顺序要求较少。同时，由于需要刷新的页面太多，通常采用并行工作模式，但每个页面的刷新时间耗时较长。一般的网络爬虫主要有以下几个局限性：

（1）抓取范围较大时的抓取结果包含了大量不相关的网页。

（2）获得的数据较松散，没有连贯性，针对有一定结构的数据资料效果不佳。

（3）通用搜索引擎大多基于关键词，缺少灵活性，难以满足支持语义信息查询和智能搜索引擎的要求。

由此可见，保证网页的质量和数量的同时，兼顾保证网页的时效性，做到信息的实时更新，仅靠通用的网络爬虫实现有一定的难度。而针对搜索范围广泛的主题，通用网络爬虫仍然有难以替代的应用价值。

5.2.3 焦点网络爬虫

焦点网络爬虫还可以叫作主题网络爬虫。顾名思义，焦点网络爬虫是一种根据预定义的主题，对网页进行选择性抓取，而非全部爬取的爬虫技术。焦点网络爬虫不像一般的网络爬虫在整个互联网中定位目标资源，而是筛选与主题相关的页面，对其进行定位。这样可以过滤掉一些无用网页，一定程度上节省爬取时所需要的带宽和服务器资源。

由于聚焦式网络爬虫是有目的地抓取信息的，因此与一般的网络爬虫相比，它必须增加目标定义和过滤机制。其工作过程见图 5-4。

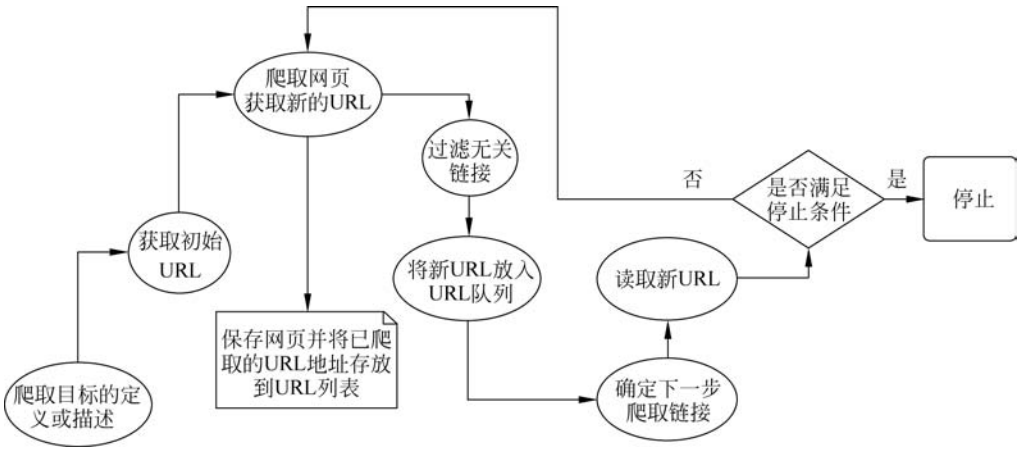

图 5-4 焦点式网络爬虫工作流程图

从图 5-4 中可以看出。第一，焦点网络爬虫要根据爬行需求定义爬虫的目标和相关描述；第二，得到一个初始 URL，根据该初始 URL 得到一个新的 URL；第三，从新 URL 中过滤掉与抓取目标无关的网页，同时需要将抓取的有效 URL 存储在 URL 列表中；第四，将过滤后的链接放入 URL 队列，根据搜索算法确定 URL 在 URL 队列中的优先级，确定下一步要抓取的 URL 地址；第五，读取刚才更新的新 URL，根据新 URL 地址抓取

内容，重复之前的抓取过程，直到满足停止条件。

在焦点网络爬虫的过程中，因有进行筛选的过程，与通用爬虫相比，额外需要一个控制模块对整个爬虫过程进行管理和控制，主要包括控制爬虫初始化、确定爬取主题的筛选、协调各模块之间协同工作、控制爬虫过程等。从控制模块角度出发，可以分为以下几个模块：页面采集模块、页面分析模块、页面相关性计算模块、页面过滤模块、链接排序模块和内容评价模块。

1. 页面获取模块

页面采集模块主要是根据要访问的主题，将 URL 加入到队列中，之后由分析模块进行分析处理，提取符合主题的网页。该模块也是任何爬虫系统中较为重要的模块。

2. 页面分析模块

页面分析模块的功能是对页面获取模块中获得的网页进行分析和处理，主要用于辅助超链接排序模块和计算页面相关性，判断是否与主题相关。

3. 页面相关性计算模块

该模块是整个系统的核心模块，主要用于评估获取的网页与爬取主题的相关性，并提供相关的爬行策略，以改进爬虫的爬行过程，提高效率。其主要思想是在系统抓取之前，模块根据用户输入的关键词进行学习，训练出一个页面相关性评价模型。当遇到一个页面与主题相关度较高时，继续向下爬行，该页面会被发送到页面相关性模型器，计算其与主题的相关性程度。如果相关性大于或等于给定阈值，则该页存储在数据库中，否则将被删除，不继续进行爬取。

4. 页面过滤模块

该模块过滤掉与主题无关的 URL，并移除 URL 及其相关的子链接。通过模块过滤之后，系统无须遍历与主题无关的 URL，保证了爬行效率。

5. 链接排序模块

链接排序模块根据优先级，将经过过滤模块处理后的页面进行排序，把结果添加到要访问的 URL 队列中。

6. 内容评价模块

内容评估模块评估网页内容的重要性。根据重要性程度，确定页面优先级，有用的页面将被优先访问，免去访问无效页面，提高爬行效率。

5.2.4　Deep Web 爬虫

1994 年，迈克尔·伯格曼提出了 Deep Web(Deep Page)的概念。Deep Web 是指使用大众普通搜索引擎并不会被发现的信息内容。Deep Web 中常常隐藏着比普通网页更

多的信息量，质量也略有不同。但由于技术限制，普通搜索引擎无法收集到这些高质量、权威的信息，这些信息通常隐藏在深度网页的大型动态数据库中，涉及数据整合、语义识别等很多深层领域。如此庞大的信息资源，如果没有合理、高效的获取途径，将是对数据资源的一种巨大浪费。因此，对 Deep Web 爬虫技术的研究具有重大的现实意义和理论价值。

常规的网络爬虫无法执行某些操作，缺乏一定的主动性和智能性。例如，需要输入用户名和密码的页面或者包含页码导航的页面，这会导致无法发现隐藏在普通网页中的信息。Deep Web 网络爬虫比之前提到的几种爬虫方法更复杂。在访问和解析 URL 之后，它需要继续分析页面是否包含深度页面条目的形式。如果包括在内，表格要模拟人的行为进行分析、填写和提交。最后，要从返回页面中提取所需内容，添加到搜索引擎中，参与索引，供用户查找。其工作过程见图 5-5。

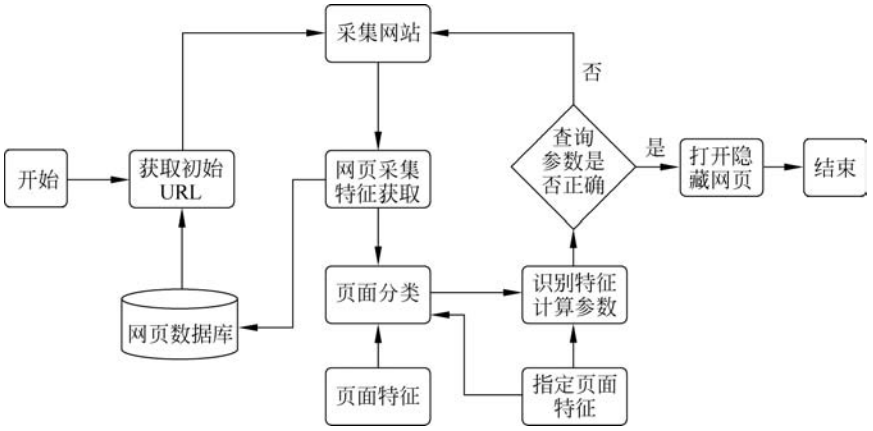

图 5-5　深层网络爬虫工作流程图

Deep Web 爬虫与常规网络爬虫的不同之处在于，Deep Web 爬虫并不是在下载页面后立即遍历所有的超链接，而是使用一定的算法对已经加载的 URL 进行分类，针对不同类别的 URL 采用不同的方法，计算查询参数，并将查询参数重新提交给服务器使用。如果提交的查询参数正确，将得到隐藏的页面和 URL。

5.2.5　分布式网络爬虫

分布式网络爬虫不仅仅是一个爬虫，而是由多个爬虫组成。每个爬虫都需要完成类似于单个爬虫的任务，从 Internet 下载网页，将网页保存在本地磁盘上，从中提取 URL 并沿着这些 URL 的方向继续爬行。分布式网络爬虫结构见图 5-6。

从图 5-6 可以看出，分布式网络爬虫是三层结构，最上层是互联网的网页；中间层是网络爬虫，或者说是单个爬虫；最底层是分布在不同地理位置的数据中心。每个数据中心有几个爬虫服务器，每个爬虫服务器上可能部署几个爬虫程序。

分布式网络爬虫的重点是不同爬虫之间如何通信。目前分布式网络爬虫根据通信方式的不同，主要分为主从型和对等型。对于主从模式，有一个专门的主服务器来维护要爬

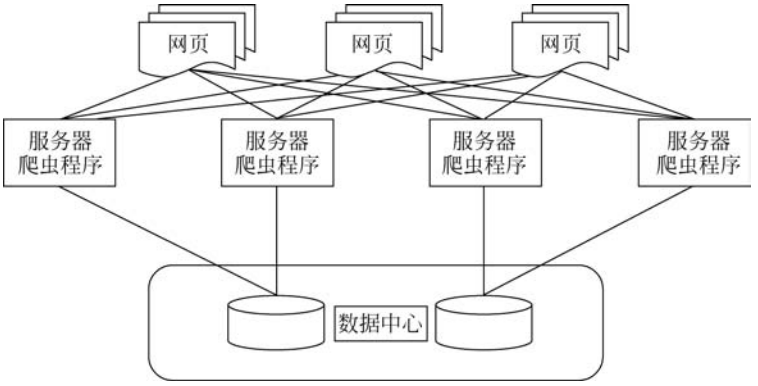

图 5-6　分布式网络爬虫体系结构

网的 URL 队列。它负责每次向不同的从服务器分发 URL，而从服务器则负责实际的网页下载。除了维护要抓取的 URL 队列和分发 URL，主服务器还负责调解每个 Slave 服务器的负载，每个 Slave 不需要相互通信。因此，该方法实现简单，易于管理。主从结构见图 5-7。

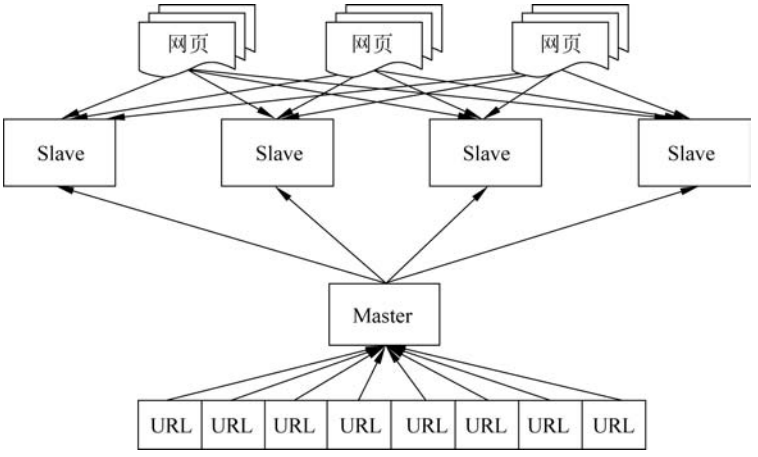

图 5-7　主从结构

对于对等方式来说，所有爬行服务器的分工没有区别。每个爬网服务器都可以从待爬取的 URL 队列中检索 URL。为了使服务器合理分工，通常采用哈希算法将要抓取的 URL 分配给不同的服务器，即计算 H mod m，其中 H 为 URL 主域名的哈希值，m 为服务器数，计算出的数为处理该 URL 的主机数。点对点结构见图 5-8。

在抓取一个网站信息时，可以设置为 $H=7$ 和 $m=3$，然后 $H \bmod m=1$，这样编号为 1 的服务器抓取该链接。

分布式网络爬虫技术是一种大规模并发收集技术，能够在最短的时间内收集尽可能多的网页，是一种高效的爬虫技术。

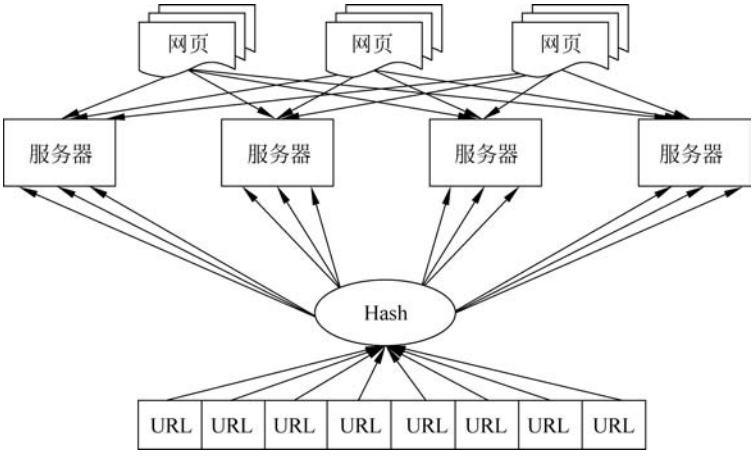

图 5-8　对等结构

5.3　网络爬虫工具

目前已经有很多成熟的网络爬虫，既有 ParseHub、Web Scraper 等浏览器拓展插件，还有八爪鱼收集器、后裔采集器等简单方便的爬虫工具。这些工具可以在极短的时间内轻松获取各种网站或网页中的大量标准化数据，帮助客户实现数据的采集、编辑和标准化，摆脱对人工搜索和数据采集的依赖，从而降低获取信息的成本，提高效率。

5.3.1　ParseHub

ParseHub 是一个免费且功能强大的 Web 抓取工具，可作为一个客户端工具，也可用作 Firefox 扩展。下载之后在本机电脑上可以操作想要爬取的数据，只需单击所需的数据即可轻松提取数据。该软件可以从多个页面获取数据，包括 AJAX、表单、下拉列表等进行交互操作。爬取的结果可以通过 JSON、Excel 和 API 访问数据。

1. 适合编程人员使用

可以在短时间内构建爬取项目，只需通过鼠标单击想要爬取的标签，就开始用 ParseHub 根据 XPath、CSS 和 Regex 等语言构建爬取规则。提供各种 API 接口，可以使用不同编程语言建立爬取规则，比如 Python、PHP、Ruby、node 和 Go 等语言。使用 ParseHub 客户端或 API 下载的 JSON 或 CSV 格式的数据，可以直接使用在网站搭建上。

可以在多种项目提供爬取功能，如：允许使用 XPath 和 CSS 选择器来获取所需的数据。使用常规表达式仅提取需要的数据。使用 ParseHub 的服务器收集数据，存储在云上的数据，无须经常维护，随时随地下载。使用 API 将任何 Web 数据连接到多种 Web 数据库和移动应用。使用 ParseHub 的调度功能，用户可以根据需要经常刷新数据，爬取新的数据。

2. 多种模板选择

ParseHub 提供一系列模板和命令组合。每个模板都由一组适用于特定网站布局的命令组成。对于网站上每种不同类型的页面布局，将创建一个独特的模板，命令 ParseHub 在该布局上采取特定操作。例如，如果要爬取一个电子商务网站的信息，可能有一个模板"main_template"，该模板会爬取产品列表页面上的所有结果。

ParseHub 工具箱中有 15 个命令可用，每个命令指示 ParseHub 在项目中采取不同的操作，一些最常见的命令如：

select：此命令选择页面上的元素。如果单击一个元素，它会选择一个元素，如果单击另一个类似的元素，它会自动选择该类型的所有元素，并插入一个开始新条目命令（隐藏在列表图标下），以确保每个都有它自己的条目。

relative select：此命令嵌套在"选择"命令下，并将一个元素链接到另一个元素。选择项目后，可以使用“相对选择”命令单击该项目并将其链接到另一个项目。例如，将日期与标题、带有名称的电话号码或产品名称的价格关联在一起。

click：此命令允许项目中单击到已通过"选择"命令选择的元素。

extract：此命令允许使用"选择"命令从已选定的元素中提取数据。例如，如果选择链接，它会自动提取链接的名称和网址本身，如果只对名称感兴趣，则可以使用“提取”命令仅提取名称。

5.3.2　Web Scraper

Web Scraper 是 Chrome 浏览器上的一个扩展插件，具有易于使用鼠标单击的图形界面，适用于所有人免费易用的网络抓取工具，通过简单的单击界面仅需花费几分钟的时间就可以进行抓取程序设置从网站提取成千上万条记录的功能。Web Scraper 利用由选择器组成的模块化结构，该结构可指示抓取程序如何遍历目标站点以及要提取哪些数据。由于这种结构，Web Scraper 能够从一个支持动态页面的网站（例如 Amazon、Tripadvisor、eBay 等）以及规模较小的鲜为人知的网站中提取信息。

1. Web Scraper 提供的功能

1）从多个页面抓取数据

不仅可以爬取当前页面的数据，还可以通过 URL 设置深层次 URL 的数据。

2）多种数据提取类型（文本、图像、URL 等）

Web Scraper 主要支持三类数据类型：数据类型、链接类型和元素类型。数据类型包括基本的 Text、Image、Table、HTML、Grouped 这些常见的类型。链接类型包括 Link、Popup Link 和 Sitemap. xml links 三种类型。元素类型包括 Element、Element Attribute、Element scroll down、Element click 以上类型。以上这些类型为 Web Scraper 提供了多种抓取方式。

3）从动态页面中抓取数据（JavaScript＋AJAX）

现如今的网站多建立在 JavaScript 框架上，这些框架虽然更易于使用，但很多爬虫无

法访问。Web Scraper 可以执行完整的 JavaScript 命令，等待 AJAX 请求，分页处理程序并自行进行页面滚动，可解决这些问题。

4）浏览抓取的数据

Web Scraper 拥有独特的思维导图模式，可以直观地查看爬虫的层次结构。通过模块化选择器系统，可以使数据提取不同层次的数据，爬取结果可直接显示在浏览器开发者模式下。

5）将抓取的数据从网站导出到 Excel

直接从浏览器中抓取工具之后，抓取网站并以 CSV 格式导出数据。使用 Web Scraper Cloud 以 csv、xlsx 和 json 格式导出数据，通过 API，webhooks 访问数据或通过 Dropbox 导出数据。

6）仅取决于 Web 浏览器，无需额外软件

Web Scraper 直接在浏览器中运行，不需要在计算机上安装任何东西。不需要任何 Python、PHP 或 JavaScript 编码经验即可开始使用 Web Scraper 进行抓取。此外，Web Scraper 还提供了完全自动化 Web Scraper Cloud 中数据提取的功能。

2. Web Scraper 使用方法

在 Chrome 浏览器中，输入 chrome://extensions 安装 Web Scraper 插件，并在开发者状态下中打开 Web Scraper 选项卡（必须将窗口放置在屏幕底部才可以显示 Web Scraper）。单击 Create Sitemap 创建新的爬取规则，将数据提取选择器添加到 Sitemap 中，最后，启动 Web Scrape 并导出抓取的数据。

Chrome 浏览器除了 Web Scraper 插件之外，还有其他爬虫插件，如 Data Scraper、Listly 等。

5.3.3 后羿采集器

后羿采集器是杭州快忆科技有限公司旗下的一款采集软件，由前谷歌搜索技术团队基于人工智能技术研发的新一代网页采集软件。

该软件功能强大，操作简单，是为广大无编程基础的产品、运营、销售、金融、新闻、电商和数据分析从业者，以及政府机关和学术研究等用户量身打造的一款产品。

后羿采集器不仅能够进行数据的自动化采集，而且在采集过程中还可以对数据进行清洗。在数据源头即可实现多种内容的过滤。通过使用后羿采集器，用户能够快速、准确地获取海量网页数据，从而彻底解决了人工收集数据所面临的各种难题，降低了获取信息的成本，提高了工作效率，可以同时支持 Windows、macOS 和 Linux 全操作系统的采集器，具有以下特点。

1. 智能识别数据

智能模式：基于人工智能算法，只需输入网址就能智能识别列表数据、表格数据和分页按钮，不需要配置任何采集规则，一键采集。支持单个网址的采集和多个网址的批量采集，支持从本地 TXT 文档中批量导入网址，并且支持批量生成网址。

可自动识别列表、表格、链接、图片、价格等。

2. 可视化单击操作

流程图模式：只需根据软件提示在页面中进行单击操作，完全符合人为浏览网页的思维方式，简单几步即可生成复杂的采集规则，结合智能识别算法，任何网页的数据都能轻松采集。

可模拟操作：输入文本、单击、移动鼠标、下拉框、滚动页面、等待加载、循环操作和判断条件等。

3. 支持多种数据导出方式

采集结果可以导出到本地，支持 txt、Excel、csv 和 HTML 文件格式，也可以直接发布到数据库（MySQL、MongoDB、SQL Server、PostgreSQL）。

4. 提供企业级服务

提供丰富的采集功能，无论是采集稳定性或是采集效率，都能够满足个人、团队和企业级采集需求，支持定时采集，自动导出，下载文件，引擎加速，按组启动和导出，Webhook，RESTful API，智能识别 SKU 和大图等。

5. 全平台支持

同时支持 Windows、macOS 和 Linux 全操作系统的采集软件，各平台版本完全相同，无缝切换。

5.3.4　八爪鱼收集器

八爪鱼收集器可以简单快速地将网页数据转化为更易解读和理解的结构化数据，以 Excel 或数据库等形式存储，并提供云采集解决方案，实现精准、高效、大规模的数据采集。其智能模式可实现输入网站数据的全自动导出，节约了用户爬取完处理数据的时间，提高了爬取效率。

1. 功能介绍

简单地说，使用八爪鱼可以轻松地从任何网页中准确地收集你所需要的数据，并生成自定义的、常规的数据格式。八爪鱼数据采集系统可以用在以下场景：

（1）财务数据，如季报、年报、财务报告，包括自动收集最近一天的净值。

（2）对各大新闻门户网站实时监控，自动爬取最新信息并将结果进行上传。

（3）监控社交平台和应用程序，自动抓取企业产品的相关评论。

（4）收集最新最全的岗位招聘信息。

（5）监控主要房地产相关网站，收集最新新房、二手房市场行情。

（6）在各大汽车网站上收集新车、二手车的具体信息。

（7）发现和收集潜在客户信息。

（8）收集各种产品官网网站的产品目录和产品信息。

（9）在各大电商平台间同步商品信息，将结果集中发布在一个平台。

2. 数据采集器的优点

（1）操作简单。八爪鱼采集器使用可视化图形操作界面，不需要专业 IT 人员，凡是会用电脑上网的人都可以轻松掌握。

（2）可进行云采集。采集任务自动分配给云端多个服务器同时执行，提高了采集效率，可以在很短时间内获取数千条信息。

（3）灵活的采集流程。可对采集流程进行折叠、拖动，还可以进行登录、输入数据、单击链接、按钮等行为，也可以针对不同的情况采用不同的采集流程。

（4）内置 OCR 功能。八爪鱼采集器内置了可扩展的 OCR 接口，支持图片中文字的分析，用户可以很方便地提取图片上的文字。

（5）定时自动采集。采集任务自动操作，按规定时段自动采集，还支持最快一分钟一次实时采集。

（6）免费使用。

3. 收集原则

八爪鱼收集器是一个模拟人的思维访问 Web 文档的互联网数据收集器。通过设计工作流程，实现采集程序的自动化，从而快速采集和整合网页数据，完成用户数据采集的目的。

规则（也叫任务）是八爪鱼规则配置程序记录手工操作过程，显示在八爪鱼客户端中，可以进行导入和导出操作的程序脚本。当规则配置好后，八爪鱼可以根据配置好的规则自动采集数据，而不是手工采集。

八爪鱼收集器的核心原理：

（1）内置火狐内核，输入要采集的网址，在八爪鱼内置的火狐浏览器中打开。

（2）模拟人浏览网页，思考如何浏览，根据网页显示数据：单击、翻页、列表等。

（3）设计工作流程，打开网页-循环翻页-循环列表-单击元素-提取数据等。

（4）自动采集数据，选择启动本地采集/云采集，自动采集数据。

4. 客户端程序

在八爪鱼客户端中，采集和导出数据主要经过以下三个步骤：

（1）配置任务。

（2）选择采集方式，本地采集或云采集。

（3）采集完成并导出数据。

八爪鱼用三个程序来完成这三个步骤：主程序负责任务配置和管理。任务采集程序负责云采集控制和收集数据的管理（导出、清洗和发布），本地收集程序根据工作流程，通过正则表达式和 XPath 原则快速收集网页数据。数据导出程序负责数据导出，导出格式支持 Excel、csv、HTML、txt 和导出到数据库。最多支持一次导出百万级数据。

5. 数据采集方式

八爪鱼采集器提供本地采集和云端采集两种采集方式，满足不同的数据采集需求。

1）本地采集

获取立即在本地计算机上进行，可用于测试任务配置是否正确并按预期运行。如果任务配置正确，采集任务可以在本地完成。

本地采集（单机采集），即使用自己的计算机进行采集。它可以抓取绝大多数网页数据，并在采集过程中对数据进行初步清洗。例如，利用八爪鱼提供的正则工具，用正则表达式对数据进行格式化，就可以在数据源处实现去除空格、过滤日期等各种操作。其次，八爪鱼还提供了分支判断功能，可以从逻辑上判断网页中的信息对不对，从而实现用户的筛选需求。

2）云采集

如果任务配置正确，立即在八爪鱼云收集集群上启动收集，并且只运行一次。云采集用户移动后，可以在任务栏中看到采集的数据。

任务在八爪鱼云采集集群上定期运行。任务将根据定期设置的时间周期性运行多次。如果多次运行收集到的数据重复（所有数据字段完全一致，即重复），则自动过滤重复数据。

云采集使用八爪鱼自身的服务集群进行数据采集，不占用本地计算机资源，节约空间。用户自行设置好采集规则后，启动云采集，可关闭自己的电脑，实现无人值守，直接获得采集后的数据。具有以下三大优势：①功能：定期采集、实时监控、数据自动去重存储、增量采集、验证码自动识别、通过 API 接口实现多样化数据导出；②速度：多节点并发操作，多服务器并行处理，采集速度比本地采集（单机采集）大幅度提高；③反封存：具有多节点、多 IP，可避免网站 IP 屏蔽，最大限度收集数据。

5.4　Python 爬虫技术

常用的网络爬虫语言有很多，包括 Python、Java、PHP、C++语言等。其中，Python 语言拥有非常丰富的爬虫框架，强大的多线程处理能力，学习简单，代码简洁；Java 语言适合开发大型爬虫项目；PHP 的后端处理能力强，代码简洁，模块丰富，但并发性相对较弱；C++运行速度快，适合开发大型爬虫项目，但成本较高。本节将介绍 Python 语言的爬虫技术。

5.4.1　Python 爬行器基础知识

如果把 Internet 比作一个蜘蛛网，那么网络爬虫就是一个在 Wb 上爬行的蜘蛛。网络爬虫通过网页的链接地址，对网页进行搜索，爬取内部内容。因此，网络爬虫的基本操作就是对网页进行抓取。

抓取网页的过程与使用浏览器浏览网页的过程类似。例如，在浏览器的地址栏中输入 https://www.baidu.com 时，浏览器作为浏览“客户端”向服务器发出请求，将服务器

上的文件“抓取”到本地，然后进行解释显示。网页的 HTML 是一种标记语言，用于对内容进行标记、解析和区分。浏览器的作用是解析得到的 HTML 代码，然后以网页的形式显示出来。

1. URL

网络爬虫从被称为种子（又称 URL 池或 URL 队列）的统一资源地址列表开始，然后从网站的一个页面（通常是首页）读取网页内容，找到网页中的其他链接地址，并通过这些链接地址找到下一个网页，不断重复上述循环，直到爬完该网站的所有网页。因此，确定 URL 是首要任务。

Internet 上的每个文件都有一个唯一的 URL，其中包含指示文件位置和浏览器应如何处理的信息。URL 由三部分组成，即协议（或服务模式）、存储资源的主机 IP 地址（有时包括端口号）以及主机资源的具体地址，如目录和文件名。因此，网络爬虫抓取数据时，必须有目标 URL 才能获取数据。

2. requests 库

Requests 是使用 Apache2 licensed 许可证的 HTTP 库，比 urllib2 模块更简洁。Requests 支持 HTTP 连接保持和连接池，支持使用 cookie 保持会话，支持文件上传，支持自动响应内容的编码，支持国际化的 URL 和 POST 数据自动编码。

Requests 库在 Python 内置模块的基础上进行了封装，从而在进行网络请求时，更加灵活，使得 Requests 可以轻而易举的完成浏览器的操作。

Requests 库全部信息可以在 https://requests.readthedocs.io/zh_CN/latest/index.html 网页上查看，常用的方法见表格 5-1。

表 5-1 requests 库七种方法

方　法	说　明
requests.request()	构造一个 requests 请求，是支撑以下各方法的基础
requests.get()	获取 HTML 网页的主要方法，对应 HTTP 的 GET
requests.head()	获取 HTML 网页头的信息方法，对应 HTTP 的 HEAD
requests.post()	向 HTML 网页提交 POST 请求方法，对应 HTTP 的 POST
requests.put()	向 HTML 网页提交 PUT 请求的方法，对应 HTTP 的 PUT
requests.patch()	向 HTML 网页提交局部修改请求，对应于 HTTP 的 PATCH
requests.delete()	向 HTML 页面提交删除请求，对应 HTTP 的 DELETE

在不传递参数的情况下，所有方法的接口样式如下：

```
requests.get("https://www.baidu.com/")        # GET 请求
requests.post("https://www.baidu.com/")       # POST 请求
requests.put("https://www.baidu.com/")        # PUT 请求
requests.delete("https://www.baidu.com/")     # DELETE 请求
requests.head("https://www.baidu.com/")       # HEAD 请求
requests.options("https://www.baidu.com/")    # OPTIONS 请求
```

网页请求中常见的两种方法为 get()和 post()：

get()，最常见的请求方式，一般用于获取或者查询资源信息，响应速度快，多数网站会选择使用这种方式来获取信息。

post()，以表单形式上传参数的请求方式，使用该方法时，除查询信息外，还可以修改信息。

网页请求的过程分为两个环节，request 和 response。

request(请求)：用户向服务器发送访问请求，是所有用户想看到网页时，所发生的第一步操作。

response(响应)：服务器在接收到用户的请求后，验证请求的有效性，信息无误后，向用户(客户端)发送响应(response)的内容。用户接收服务器响应的内容，网页上显示出请求的内容。

服务器响应(response)的内容，称为一个 response 对象，如果想具体检查每个服务器返回 response 对象的属性，需要用到 response 方法，常用的属性见表 5-2。

表 5-2　response 方法常用属性表

属　　性	说　　明
r. status_code	HTTP 请求的返回状态，常见如：200-连接成功，404-失败
r. text	HTTP 响应内容的字符串形式，即 URL 对应的页面内容
r. encoding	从 HTTP 头部中猜测的响应内容编码方式
r. apparent_encoding	从内容中分析出的响应内容编码方式(备选编码方式)
r. content	HTTP 响应内容的二进制形式

3. BeautifulSoup

BeautifulSoup 是一个 Python 的函数库，主要功能之一是从网页中抓取数据。该库提供了一些简单的函数来处理导航、搜索、修改分析树等。此外，BeautifulSoup 库还是一个工具箱，可用来解析文档，为用户提供要爬取的数据。因为它很简单，所以编写一个完整的应用程序不需要太多代码。BeautifulSoup 可自动将输入文档转换为 Unicode 编码，将输出文档转换为 UTF-8 编码，不需要用户过多考虑编码方式。BeautifulSoup 已经成为与 lxml 和 HTML6lib 一样好的 Python 解释器，为用户提供了使用不同解析策略或更多的灵活性。

BeautifulSoup3 目前已经开发完毕，推荐在当前项目中使用 BeautifulSoup4，但是已经迁移到 BeautifulSoup4，这就意味着在导入的时候需要导入 BeautifulSoup4。BeautifulSoup4 支持 Python 标准库中的 HTML 解析器，以及一些第三方解析器。如果不安装它，Python 将使用 Python 的默认解析器。lxml 解析器功能更强大、速度更快。

BeautifulSoup4 库通过安装解析器，将复杂 HTML 文档转换成一个树形结构，每个节点都是 Python 对象。常用的四种解析器见表 5-3。

表 5-3 四种解析器

解析器	使用方法	条 件
bs4 的 HTML 解析器	BeautifulSoup(mk,'HTML. parser')	安装 BeautifulSoup4 库
lxml 的 HTML 解析器	BeautifulSoup(mk,'lxml')	pip install lxml
lxml 的 xml 解析器	BeautifulSoup(mk,'xml')	pip install lxml
HTML5lib 的解析器	BeautifulSoup(mk,'HTML5lib')	pip install HTML5lib

BeautifulSoup 中的五种基本元素见表 5-4。

表 5-4 五种基本元素

基本元素	说 明
Tag	标签，最基本的信息组织单元，<>开头和</>结尾
Name	标签的名字，< a >…的名字是 a，格式：< tag >. name
Attribute	标签的属性，字典形式进行组织，格式：< tag >. attrs
NavigatableString	标签内非属性字符串，<>…</>中的字符串，格式：< tag >. string
Comment	标签内字符串注释部分

5.4.2 反爬虫与反爬虫技术

在互联网上抓取开放资源并不违法。作为互联网大数据采集的技术手段，网络爬虫本身是中立的。而抓取未经授权、未经授权的数据会影响服务器的正常运行，抓取的数据会被用于商业目的，并在未经授权的情况下公开展示。这些突破爬虫大数据收集法律和技术界限的“不友好”爬虫，应该被“抵制”，也就是反爬虫。

反爬虫的主要工作包括两个方面，一是对不友好爬虫的识别，二是对爬虫行为的预防。爬虫识别的主要任务是区分不友好的爬行行为和正常浏览行为的区别。阻断爬虫是为了防止恶意爬行，同时可以在识别错误时为正常用户提供一个释放通道。

1. “不友好”爬虫的特征

1）不遵守 robots 协议

友好的爬虫应该遵守 robots 协议。

2）大规模并发访问

友好爬虫的爬行频率和策略合理。

3）对服务器的持续或瞬时压力

友好的爬虫对服务器的压力较小。

2. 反爬虫技术

与爬虫技术相比，反爬虫其实更复杂。目前，不少互联网公司都花更多的精力研究“反爬虫”。爬虫不仅会占用大量网站流量，导致有真实需求的用户无法进入网站；还可能造成网站关键信息泄露等问题。爬虫存在于互联网的各个角落，所以爬虫有优点也有缺点。这里为大家介绍一下与爬虫一同诞生的反爬虫技术，以及如何防止别人爬取自己

的网站。

1）控制用户对报头请求

这是一种最常见的反爬行策略，很多网站会检测头部请求中的 User-Agent，有些网站会检测 Referer。因此，在爬虫代码中，需要将 User-Agent 伪装成浏览器发出的请求，混淆反爬虫策略。有时服务器也可能会检查引用器，因此也需要设置引用器（用于指示此时请求是从哪个页面链接的）。

2）限制 IP。

当使用同一 IP 多次频繁访问服务器时，服务器会检测到该请求可能是爬虫操作。因此无法正常响应页面的信息。这种反爬虫技术可以利用 IP 代理池技术，互联网上有很多提供代理的网站。

3）动态页面的反爬行。

1）和 2）大多出现在静态页面中，还有一些网站需要抓取的数据是通过 AJAX 请求或者 Java 生成的。首先，使用 Firebug 或 HttpFox 分析网络请求。如果通过浏览器的开发者模式能找到 AJAX 请求，并分析具体参数和响应的具体含义，就可以使用 requests 库或 urllib2 库模拟 AJAX 请求，分析响应的 JSON，得到所需的数据。有些网站可能会对参数进行加密或拼接后发送到服务器，以达到反爬虫的目的。这时我们可以尝试使用 JS 代码进行破解。也可以使用“PhantomJS”，这是一个基于 WebKit 的“无头”浏览器，它将网站加载到内存中，并在页面上执行 JavaScript。因其不显示图形界面，所以运行起来比完整的浏览器更有效率。

4）验证码验证

验证码（CAPTCHA）的全称是：“Completely Automated Public Turing test to tell Computers and Humans Apart”，意思是“全自动区分计算机和人类的图灵测试”。它是一种区分用户是计算机还是人类的公共全自动程序，一般被用来防止恶意破解密码、刷票、论坛灌水等活动，有效防止黑客利用特定程序暴力破解手段不断登录注册用户。现在验证码在很多网站上被广泛使用。因为计算机一般无法直接回答 CAPTCHA 的问题，所以回答问题的用户可以被视为人类。

3. 反反爬虫技术

1）IP 和访问间隔的限制

爬虫时并没有使用用户自己的真实 IP，而是使用代理服务器或云主机来不断切换 IP，在请求中使用代理。

2）报头内容验证

使用 Selenium 或其他嵌入式浏览器访问，构造合理的头信息，主要包括用户代理和主机信息。使用 Selenium 将调用浏览器。也可以根据规则自行组装头信息，在爬虫实现中尽可能完整地填写头的属性值。

3）根据 cookie 验证

在请求头的信息上使用不同的线程记录。如使用 requests 库中的 requests. session 方法，为每个线程保存 cookie，将获取到的 cookies 附加到头部信息中上，或者根据站点需

求正确使用 cookies 中的数据(例如,使用 cookies 中指定的密钥进行加密验证)。

4）验证码表单

目前 Web 站点经常使用的验证码可以分为四大类：计算验证码、滑块验证码、地图识别验证码和语音验证码。目前流行的验证码破解技术有两种：机器图像识别和人工编码。此外,还可以使用浏览器插件来绕过验证码。

5）JS 解析

对于异步加载的网页,可以使用 Selenium 或 PhantomJS 对页面进行 JS 解析,并执行页面内容获取所需的正确 JS 方法或请求。当然,一个真实的浏览器也可以作为收集工具的媒介,如可以封装一个自定义的 Firefox 浏览器,以插件的形式实现收集工具。

6）动态调整页面结构

对于这种反爬虫技术,最好的方法是先收集页面,然后根据收集到的页面进行分类,特别是爬虫程序中的异常捕获。如果一个页面的 HTML 是不规则的,那么它的显示会是一个问题。因此,对于结构动态调整的页面,可以使用 Selenium 加载浏览器,根据信息区域尝试采集。此外,可以尝试使用正则表达式排除结构中的随机因素。

7）蜜罐模式拦截

蜜罐的设置使得爬虫无法收集到真实的信息。在这种情况下,只有一种策略。爬虫分析一个超链接后,不要贸然进入该超链接。首先分析蜜罐的结构,判断这个蜜罐中隐藏的信息,包括表单字段、页面等等。分析异常后,在提交表单和收集页面时绕过蜜罐。

5.5 本章小结

采集互联网中大量数据最常见、最有效的方法是使用网络爬虫。本章介绍了网络爬行策略、网站和爬虫之间的协议(robots)、爬取数据的方法；然后介绍了 4 种简单的爬虫工具的使用；最后,较详细地介绍了 Python 爬虫技术常用的 requests 库及 BeautifulSoup 库的介绍和使用。

习题

1. 网络爬虫的网络爬行策略有哪些?
2. 爬虫的网页更新策略有哪些?
3. 简述 robots 协议的作用。
4. 简述通用爬行器的原理。
5. 简述了八爪鱼采集器的功能和优点。
6. 简述 Python 的 requests 库的主要函数。
7. 反爬虫的主要技术有哪些?