动态网页解析下的分布式网络爬虫系统设计

陈春晖
(福建农业职业技术学院,福建 福州 350000)

摘 要:由于 Web 前端技术和反爬虫技术的发展,系统的复杂性也在逐渐增大,爬虫在爬取数据时获取动态网页数据、应对反网络爬虫以及集群稳定性还存在问题,这些问题对爬虫系统获取动态网页数据的效率存在着很大影响。文章基于分布式网络的爬虫器,对上述问题进行了深入的研究和分析。首先,对所设计的分布式爬虫系统进行了需求分析;其次,结合系统需求,给出了系统整体结构和各模块的设计;最后,重点分析了系统中的关键技术,即爬虫的健壮性分析、网页动态加载分析。

关键词:分布式网络爬虫;动态网页资料获取;防网络爬虫

0 引言

近年来,互联网数据的需求急剧增长,不仅是正在研发高性能爬虫技术的传统搜索引擎公司,还有大数据、人工智能等新兴企业和学术机构有着巨大的需求,这就需要一种低成本的网络爬虫器,以解决购买数据的资金问题[1-2]。本文在对系统需求进行分析的基础上,提出了一种基于分布式爬虫的算法。

1 系统需求

1.1 能够获取站点的非同步载入

随着异步负载技术的不断发展,动态页面的异步负载在网络产品中得到了越来越多的应用。当前主要的爬虫无法对动态页面进行有效分析,如何实现动态网页的爬虫是本设计的主要内容。

1.2 反爬处理

数据站点设计了一套防爬策略,以保护站点资料和普通使用者的存取,如当前 IP 访问被禁用、单位时间访问限制等。所以,在设计系统时,必须考虑如何处理好防爬虫问题。

1.3 健壮性

一个好的系统,不仅要有很好的容错率,还要有很高的稳定性。实例证明,这种分布式爬虫系统必须具有良好的性能,可以在几个星期甚至几个月内连续运行。

1.4 扩展性

在分布式网络中,可扩展性非常重要,因为分布的网络爬虫程序会经常添加或移除节点,而 Scrapy-Redis 分布体系结构恰好能解决这个问题,Scrapy-Redis 最大的优势在于可扩充性,可以随意添加和删除任何一个节点。

1.5 低成本和高效率

本设计的爬虫系统期望在硬件成本最小的情况下获得更高的性能,Scrapy-Redis 的分布架构正是最佳的解决办法,其所需的服务器数量很少,普通的计算机也能使用,即使是树莓派(一种卡式的微型计算机,造价约在 200 RMB)也可以充当 Slave 节点。而且,基于 Scrapy-Redis 结构的爬虫系统也是分布的,可以很好地实现对页面的高效爬取[3]。

2 系统结构与模块设计

本文根据系统需求提出了基于分布式爬虫体系结构的结构模型,并将其划分为数据层和业务逻辑两个层次,结合现有的分布式框架进行了详细的设计,如图 1 所示,实线箭头表示数据流方向,虚线箭头表示分布群集的 Master 节点和 Slave 节点间的“请求应答”通信。

2.1 初始化参数模块

初始化参数模块是为了让系统能够正常工作而初始化特定的参数,在 Slave 节点上,主节点 IP 位址和要爬取的站点的参数都要初始化。Slave 节点要在主节点上的 IP 位址,并从 Master 节点上的 Redis 获取爬取作业,主节点无需事先知晓 Slave 节点的 IP 位址,而 Slave 节点可以随时进出。此外,系统还会对网站的 URL 初始化,该 URL 将会出现在工作分配模组中的待执行工作序列中。

2.2 任务分配与爬取策略

2.2.1 任务分派

主要是对待完成的任务和已完成的任务进行记录,并根据已完成的任务排队来防止对爬取的重复访问。

当系统开始时,任务指派模块会收到种子 URL 的初始化,并将其放到要爬取的队列中。

集群中 Master 节点对 Slave 节点采取优先服务的原则,即从主节点的任务队列中提取爬取任务,提取完后记录在已爬取任务队列中[4]。

若某个 Slave 节点表现得更好,那么该节点可以更快地获得任务;相反,则会变得更慢。所以,这个策略可以最大限度地利用各个 Slave 节点的最高性能,而这些 Slave 节点本身是否在线,并不会对其他 Slave 节点的处理任务造成任何影响。

2.2.2 爬虫策略模块

不同的站点对数据的爬虫策略是不同的,一般采用广度优先策略和深度优先策略获取数据。

2.3 系统的稳健维护模块

系统的稳健性维护模块包括 4 个子模块:心跳探测、IP 代理、类人爬取模块、记录遗失任务模块。

作者简介:陈春晖(1980—),男,福州闽侯人,讲师,硕士;研究方向:虚拟现实,网页。

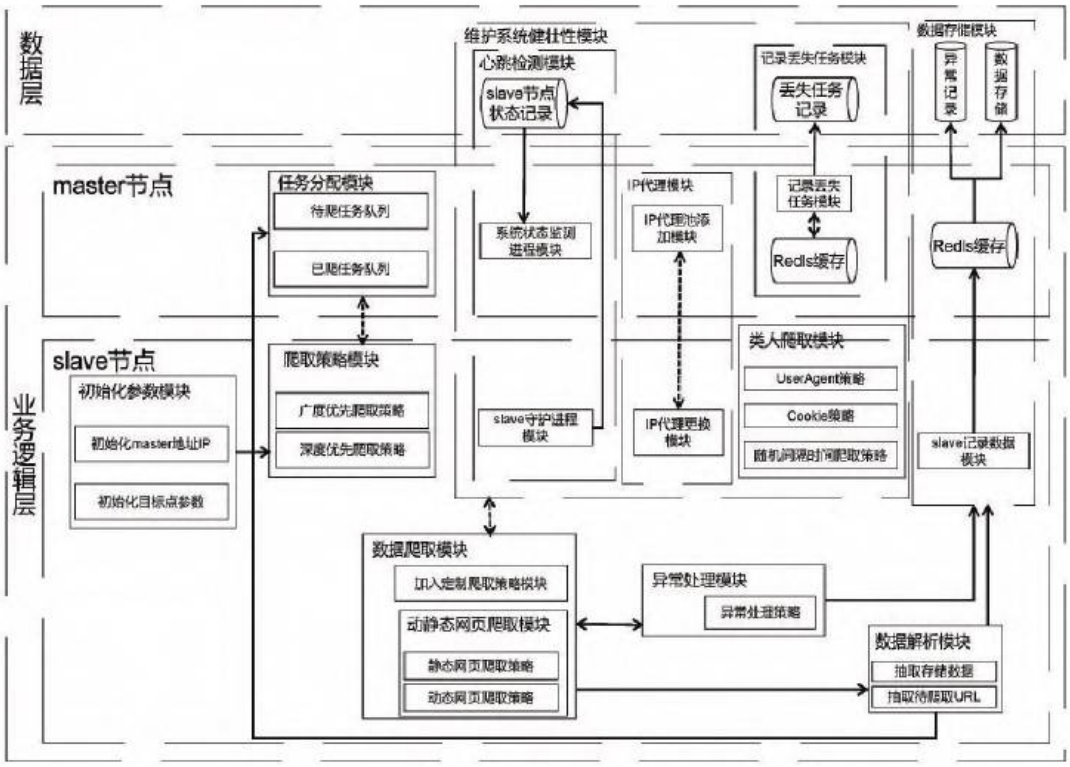

图 1　系统架构

2.3.1　心跳探测模块

模块包括监控进程、状态记录、保护进程等，是系统的一个关键部件，可以监控每一个 Slave 节点的登录和访问，包括当前的状态（比如：正常的爬虫、异常的要求、退出的情况）。

系统状况监控流程模块：位于主节点，作用是监控当前 Slave 节点的状态，并对监控结果进行处理和记录。

Slave 节点状态记录：将 Slave 节点的状态存入诸如 MySQL 之类的关系数据库中。

Slave 守护模块：在 Slave 的节点中，其功能是将自身的状态在特定的时间段内传递出去，并存储在一个资料库中。

Slave Device 模块对 Slave 节点的状态记录进行计时写入，而 Slave Development 通过该过程的状态信息来判定该节点的状态。

2.3.2　IP 代理模块

是非常关键的模块，能有效地防止网络爬虫，通过特定的策略，持续地修改 IP 地址，使其造成多个站点的访问，降低被拦截的概率。

IP 代理替换模块：在 Slave 节点上，主要作用是从 IP 代理池中随机提取 IP，用于爬虫系统的爬虫和记录 IP 的异常情况。

2.3.3　类人爬虫模块

作为系统的关键部件，其作用是通过仿真用户访问网站，欺骗网站的防爬机制，获取用户的数据。本文提出了一种类人爬取策略，主要内容如下。

用户代理策略：由一些常见的用户代理组成用户代理库，在用户使用时随机选择一个，并将其安装到爬虫系统中。

Cookie 战略：选择是否使用 Cookie，取决于目标站点对数据的爬虫。

随机时间爬虫策略：通过仿真用户在特定时间周期内的存取次数和存取时间，欺骗目标站点的频率和存取时间，从而判定该系统是不是爬虫；针对特定区域的爬虫限制，随机产生一定的时间间隔。

通常情况下，使用最佳策略进行组合，而在实际的爬行前，先对网站的忍耐程度进行检测（即严格的抗爬行战略）。

2.3.4　记录遗失任务模块

模块位于主计算机上，用于记录因异常退出 Slave 节点而导致的丢失任务。若 Slave 节点没有完成爬取，此时出现异常的 Slave 节点，爬虫任务将不能被发送到 Master 节点上，这就会导致任务丢失。本模块主要包括任务丢失和任务丢失的日志。

2.4　资料爬取组件

数据爬虫模块是利用特定的策略从页面中获取相应的 URL，包括自定义的爬虫策略和动态的网页爬虫。

2.4.1　添加自定义爬取策略模块

在实际的爬行过程中，通常只需要特定的类别或特定区域的信息，广度优先，深度优先。而爬行的覆盖区域太大，则无法快速准确地获取所需要的信息，这就需要一个特殊的爬行器策略为特定站点提供有吸引力的信息。此外，对不同站点的蠕动行为所能忍受的范围也不尽相同。为了应对更多的爬虫数据，在此将加入一些事先设置的战略配置。

2.4.2　动态、静态的网页爬虫模块

主要针对动态页面和静止页面采用不同的爬虫策

略,分析动态页面所需的时间一般要比静态页面多,区分动态页面和静态页面,可以提高爬虫效率。

在动态网页分析方面,根据邬柏[5]的建议,向白名单中添加网址,新网址将依据查询的白名单,决定是否使用动态页面的解析方式,并将不包含白名单但视为动态页面的 URL。在此基础上,本设计采用了如下的改进方法。

利用规则基础来判定 URL 对应于存储预先调查的目标站点动态页面 URL 的正则表达式的规则库,利用规则表达式,可以极大地减少规则库的容量,减少系统的维护费用。

对于异常解析的网站日志,由管理员进行分析,然后确定是否进行更新。在相同的网站上,由于用户关注的数据,动态页面的分布规则是有限制的,通常不会因为规则表达式的覆盖而无法进行动态分析,大多数的异常都是由网络异常或者防爬虫因子引起的。所以,管理员在分析不正常的日志之后,就可以决定是否进行规则的更新。

2.5 异常处理模块和解析模块

2.5.1 异常处理模块

处理在执行爬虫作业 URL 时出现的异常,并将其传回 Master 节点,由 Master 节点进行记录。

2.5.2 数据分析模块

根据预定的规则,对采集到的数据进行分析,生成 URL 或最终存储的数据。用户依据网站、调查需求,从网站中提取或存储的数据类型(页面文字格式、JSON 格式、URL 的链接类型)和 URL 的规则,对相关的规则进行配置。在同一网站上,使用者要抽取的数据与 URL 的链接是一样的,符合一定规则的 URL,就会相应地抽取规则或者行为,例如,从某一列中抽取某一段话、某一段文字、点击加载内容等。

2.6 数据储存模块

将 Redis 在 Master 端收到的数据(异常和爬虫)进行统一处理,包括:记录、缓存、异常记录、数据存储等。

2.6.1 Slave 记录数据模块

模块的作用是将 Slave 的例外访问、数据处理系统中的数据处理、被 Slave 节点获取的工作、Slave 节点请求的下载量(包括请求例外)返回到主要节点的 Redis 中,并大量地传送给关联资料库。

2.6.2 异常情况

保存异常数据,主要是利用传统的关系数据库,将异常爬虫记录(在爬虫过程中出现的 URL)保存下来,以便管理员查询异常,补充数据。

2.6.3 资料储存

储存爬取资料,采用传统的关联式资料库,存取主节点上的 Redis 中储存的资料,如最后储存使用者所需要的资料,以备日后使用者查询时所用。

3 结语

本文在基于动态网页解析的网络爬虫系统需求基础上,对系统进行了整体的设计,并将其划分为数据采集级和数据解析级、数据存储层、节点接入层和系统管理层。各层共包括存储层、网页下载和任务调度、网页信息提取、网页删除、节点管理、爬虫管理 6 大部分。随着大数据时代的到来,各种页面的涌现,传统基于计算机的爬虫系统已不能很好地适应目前的检索需求,同时也需要更高的、实时的、精确的信息采集。因此,如何有效地从网页中抽取网页信息,建立一个基于动态网页解析的网络爬虫系统是十分必要的。

[参考文献]

[1]孟庆昊,沈妍,李青君,等.基于爬虫技术的医疗行业舆情监控系统的设计与实现[J].科技创新与应用,2022(8):27-29.
[2]冯艳茹.基于 Python 的网络爬虫系统的设计与实现[J].电脑与信息技术,2021(6):47-50.
[3]刘宏嘉,王静,黄宇亮,等.基于 Web 爬虫技术的电子病历信息聚合工具的开发及验证[J].中国医学物理学杂志,2021(11):1444-1448.
[4]张晓宇.基于 DHT 网络爬虫原理的 P2P 监听系统研究[J].微处理机,2021(5):22-25.
[5]邬柏.支持 AJAx 的分布式爬虫系统的研究与实现[D].武汉:武汉华中科技大学,2013.

(编辑 沈强)

Design of distributed web crawler system under dynamic web page parsing

Chen Chunhui
(Fujian Vocational College of Agriculture, Fuzhou 350000, China)

Abstract: Due to the development of Web front-end technology and anti crawler technology, the complexity of the system is also gradually increasing. There are still problems for the crawler to obtain dynamic web page data when crawling data, deal with anti web crawlers and cluster stability. These problems have a great impact on the efficiency of the crawler system to obtain dynamic web page data. Based on the distributed network crawler, this paper makes an in-depth study and analysis of the above issues. Firstly, the requirements of the designed distributed crawler system are analyzed; Secondly, combined with the system requirements, the overall structure of the system and the design of each module are given; Finally, the key technologies in the system, namely, the robustness analysis of the crawler and the dynamic loading analysis of the web page, are emphatically analyzed.

Key words: distributed web crawler; dynamic web page data acquisition; anti web crawler