基于 Chrome 扩展的爬虫系统设计与实现

魏少鹏，夏小玲

（东华大学 计算机科学与技术学院，上海 201620）

摘　要：为了提高网页数据抓取效率，降低爬虫对系统资源的消耗，提出了一种基于 Chrome 扩展的爬虫系统。利用 Chrome 浏览器对网页进行解析，防止被爬取对象屏蔽和网页异步加载问题，并且实现数据结构化；通过选择普通用户版扩展和服务器版扩展，既可以实现无人值守主动抓取，也可以在用户浏览网页的同时抓取信息。整个系统前后端分离，并且采用面向接口编程，具有良好的扩展性。通过从搜达足球网站抓取英超赛程，验证了程序的高效可行性。

关键词：爬虫系统；Chrome 扩展；Netty

DOI：10.11907/rjdk.1511495

中图分类号：TP319　　**文献标识码**：A　　**文章编号**：1672-7800(2016)003-0076-05

0　引言

在大数据时代，信息呈“爆炸”式增长，为企业和个人提供了丰富的信息来源。以新浪微博为例，截至 2015 年 8 月，有注册用户 6 亿，日均活跃用户 6 600 万，日均发微博 1.2 亿条[1]，只通过搜索引擎搜集、获取整合数据非常困难[2-3]。在大数据时代信息空前丰富的背景下，数据获取，即如何有效整合散落在互联网各个角落的数据，从而为用户提供更为精准的信息至关重要，解决该问题归根结底涉及到网络爬虫技术[4]。

网页爬虫几乎与网页技术一同出现，刚开始的爬虫技术主要是利用图论知识，将整个互联网看作一个连通图，采用深度优先或广度优先算法来抓取数据。随着互联网规模的扩大，对采集速度和数据质量要求不断提高，由此出现了主题爬虫，即根据给定的主题，抓取与主题相关的网页。当互联网规模和技术进一步发展，出现了基于分布式的爬虫，例如 Google Crawler、Internet Archive Crawler 等，并且还出现了针对 Ajax 网站的爬虫，例如 Google Groups、Google Suggest 等。无论是传统爬虫还是主题爬虫，抑或是分布式爬虫和针对 Ajax 的爬虫[5-7]，其核心技术都是使用程序模拟 IE 浏览器的功能，将 URL 作为 HTTP 请求的内容发送到对方服务器端，然后读取对方服务器端的响应资源[8]。

但在实际使用中，传统爬虫技术存在很大的局限性：①实现 IE 客户端模拟难度大，由于网络环境复杂，不仅实现模拟 IE 客户端的工作很困难——即使调用开源的 API，也是非常繁琐的，而且网页的解析工作也异常困难复杂；②使用场景存在很大局限性[9]，比如不能处理包含异步请求的网页、抓取的数据中含有很多脏数据（如通过网页中的 URL 抓到广告等）、对服务器配置要求高（所有操作都在中央服务器）、容易被抓取对象屏蔽（客户端浏览器模拟不完善）和无法获得网页详情（网页的 URL 通过 JavaScript 动态生成加密信息）等一系列问题；③使用不友好，主要表现是部署难度大，只适合专业用户，不支持用户在浏览网页的同时爬取数据。

员使用，可应用于智慧城市、智慧交通前期阶段的交通基础数据处理。

参考文献：

[1] 卫翀，邵春福. 考虑交通量随机波动的随机用户均衡配流模型[J]. 吉林大学学报：工学版，2015，45(5)：1408-1413.
[2] 林宇洪，沈嵘枫，邱荣祖，等. 南方林区林产品运输监管系统的研发[J]. 北京林业大学学报，2011，33(5)：130-135.
[3] 蓝岚，伍炜. 基于交通量特性分析的山地城市交通发展策略[J]. 重庆交通大学学报：社会科学版，2015(1)：24-26.
[4] 林宇洪，林森，景锐，等. 木材运输 IC 卡读写器的开发[J]. 福建农林大学学报：自然科学版，2010，39(4)：435-438.
[5] 肖颖，刘晓建. 轨道交通客流预测的扩展反馈四阶段法研究[J]. 都市快轨交通，2014，27(5)：48-51.
[6] 曹志成. 交通量预测中分布交通量预测的几种计算方法解析[J]. 建筑工程技术与设计，2015(11)：1912-1917.
[7] 林宇洪，胡连珍，蒋新华，等. 基于二维码的农超对接供应链追溯系统的设计[J]. 黑龙江八一农垦大学学报，2015，27(6)：83-87.
[8] 孟博翔. 基于四阶段法的兰州市雁滩商业圈交通需求预测[J]. 交通科技与经济，2014，16(2)：27-30.

（责任编辑：孙　娟）

作者简介：魏少鹏（1990－），男，甘肃天水人，东华大学计算机科学与技术学院硕士研究生，研究方向为计算机软件与数据可视化；夏小玲（1966－），女，上海人，博士，东华大学计算机科学与技术学院教授，研究方向为计算机软件与数据可视化。

针对传统爬虫的以上缺点，设计了基于 Chrome 扩展的爬虫系统，其优势如下：①实现难度小。因为本身就是基于 Chrome 浏览器，所以不需要模拟浏览器、解析网页等；②使用局限性小。基于 Chrome 扩展的爬虫系统本身就是浏览器访问，不仅可以解决对方服务器端对 user-agent 验证的屏蔽问题，而且可以解决 URL 带有动态加密信息造成的问题。Chrome 扩展的爬虫系统是对解析好的网页信息进行提取，因而只会提取所需要的结构化信息，降低脏数据，提高数据质量。通过控制 Chrome 扩展中 JavaScript 脚本的执行时间，可以解决网页的异步加载问题；③使用友好。因为是基于 Chrome 的扩展，所以安装非常方便；信息抓取模块不仅有可以自动抓取的服务器版还有普通用户版，普通用户版可以在用户浏览网页时实现信息抓取，降低了对双方服务器的负载。

1　系统结构

通过对传统爬虫系统和 Chrome 扩展的研究，本文设计的基于 Chrome 扩展的爬虫系统是一个严格的前后端分离的系统，信息抓取模块负责信息提取和结构化，中央服务器模块负责数据去重和持久化，两者之间没有业务上的重合。系统架构如图 1 所示。

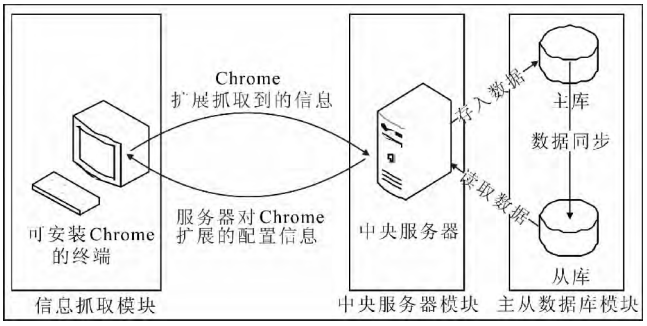

图 1　系统架构

如图 1 所示，基于 Chrome 扩展的爬虫系统主要由 3 个模块构成，即基于 Chrome 扩展的信息抓取模块、接受并处理 Chrome 扩展信息的中央服务器模块和进行数据存储的数据库模块。

1.1　中央服务器模块

中央服务器主要接受信息抓取模块的请求，根据需求返回配置数据或者接受抓取到的数据，并对接受的数据去重之后进行持久化。因为该过程不解析数据，所以相较于传统爬虫系统更节省资源。根据信息抓取模块的设计要求，中央服务器模块必须是一个支持高并发的后台系统。鉴于此，采用 Netty 框架[10-11] 实现高并发。并且，为了整个爬虫系统的扩展性，中央服务器模块采用面向接口编程思想，引入 Spring[12] 对 Bean 进行管理。

1.2　信息抓取模块

信息抓取模块的主要任务是从网页中提取信息并且结构化，然后把结构化的数据用 Ajax 发送到中央服务器，其本质是 Chrome 浏览器的一个扩展，是对浏览器功能的扩充。也正是该核心模块基于 Chrome 扩展，才能利用 Chrome 扩展自有的性能，在工作量小于传统爬虫的同时，实现优于传统爬虫系统的功能。

信息抓取模块主要由 3 类文件构成，第一类是 manifest.json 文件，是 Chrome 扩展的配置文件；第二类是公有的 JavaScript 脚本文件，主要用于实现与中央服务器的交互并控制其它 JavaScript 脚本；第三类是从页面中提取信息的 JavaScript 脚本文件。

信息抓取模块有两个版本：一个是服务器版，根据从中央服务器获取的信息，自动打开网页并从中提取信息，当信息提取结束之后关闭该标签页，如果有必要会继续打开网页中的链接，并从链接页提取信息；另一个是普通用户版，与服务器版的区别在于不会从中央服务器获取信息并打开网页，而是当用户浏览网页的 URL 匹配网页抓取规则时，从网页中提取信息，不符合则不会进行信息提取。相对于服务器版，普通用户版不需要专门为了抓取数据而单独运行程序，也不会单独请求对方的服务器，不仅节约了用户的资源和时间，也减少了对方服务器的额外负载。

1.3　主从数据库模块

为了满足中央服务器模块的高并发请求，对数据库采用主从库设计，主从库的设计不仅提高了数据读写的效率，而且提高了数据安全性[13]。通过在中央服务器对 JDBC 接口的封装，使主库只承担写的功能，从库只承担读的功能。

2　系统实现

本文设计的爬虫系统包含基于 Chrome 扩展的信息抓取模块、基于 Netty 的中央服务器模块和采用主从配置的数据库模块。其中，Chrome 扩展模块分为个人版和服务器版，相对于个人版，服务器版的信息抓取模块更复杂。

图 2 是服务器版系统时序图。通过系统框架图、系统模块介绍和系统时序图可知，基于 Chrome 扩展的爬虫系统实现过程实际就是开发一个具有信息提取模块的 Chrome 扩展，并且通过 Netty 框架实现一个支持高并发的中央服务器模块，即通过使用 Chrome 扩展和 Netty 框架技术，实现一个比现有爬虫系统更高效好用的爬虫系统。

2.1　中央服务器模块

中央服务器模块的主要功能是对信息抓取模块提交的信息进行去重并持久化。为了使系统支持高并发，引入 Netty 框架和主从库设计；为了扩展方便，采用面向接口编程思想，主要有 HTTPServer、JDBC 封装和 API 核心模块。

2.1.1　HTTPServer

HTTPServer 主要实现创建服务端 NIO 线程组和端口监听。通过以下代码来实现服务端 NIO 线程组设置：

```
EventLoopGroup bossGroup = new NioEventLoopGroup
();
```

```
EventLoopGroup workerGroup = new NioEventLoopGroup();
ServerBootstrap serverBootstrap = new ServerBootstrap();
serverBootstrap
.group(bossGroup, workerGroup)
.channel(NioServerSocketChannel.class)
.childOption(ChannelOption.TCP_NODELAY, true)
.childOption(ChannelOption.CONNECT_TIMEOUT_MILLIS, 60 * 1000)
.childOption(ChannelOption.SO_REUSEADDR, true)
.option(ChannelOption.SO_KEEPALIVE, true)
.childOption(ChannelOption.ALLOCATOR,
    PooledByteBufAllocator.DEFAULT)
.option(ChannelOption.SO_BACKLOG, 4096)
.childHandler(new HttpServerInitializer());
```

使用“serverBootstrap.bind(port).sync();”来绑定监听端口，其中 port 是要监听的端口号。

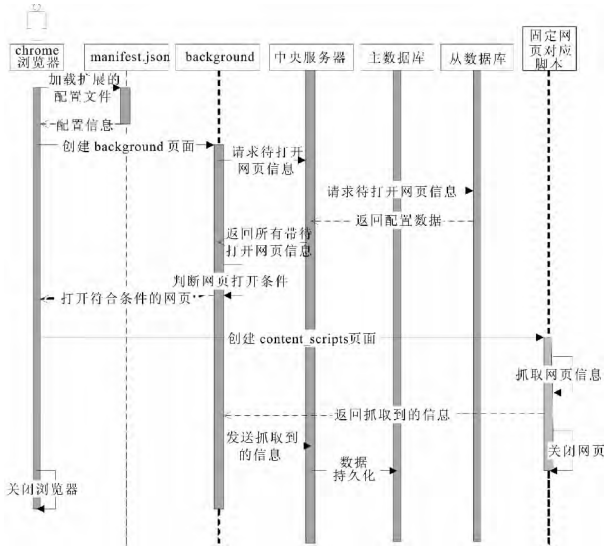

图 2 系统时序

2.1.2 JDBC 封装

JDBC(Java Database Connectivity)是 Java 提供的执行 SQL 语句的 API，其封装了多种数据库的访问方式。本系统由于支持高并发和主从库设置，因而需要对 JDBC 进行封装以支持主从配置，提高与数据库交互的速度。

通过 Java 的 Properties 文件来配置主从库，并且通过 APIService 来封装 JDBC，所有查询操作都是用从库，所有更新或删除操作都使用主库。通过这种封装，不仅完成了读写分离，而且提高了效率。

2.1.3 API 开发

通过上述步骤，即可完成对中央处理模块核心功能的搭建。在此基础上，对不同主题信息进行抓取时，只需要开发对应的 API 即可。API 的主要作用是接受 Chrome 扩展发送的信息，并且对信息进行处理(验证是否为脏数据并去重等)，将满足条件的数据写入主库。

在前面步骤基础上，开发 API 很简单，只需要注入 Bean 和声明所需参数，然后在逻辑中验证参数即可。以添加赛程为例：

```
@APIContent(value = "AddGame", description = "添加赛事信息")
@DataRequest(value = "AddGame", params = {@DataRequestParameter(name = "matchName", description = "联赛中文名"),
@DataRequestParameter(name = "homeTeamName", description = "主队名称"),
@DataRequestParameter(name = "homeScore", description = "主队得分"),
@DataRequestParameter(name = "awayTeamName", description = "客队名称"),
@DataRequestParameter(name = "awayScore", description = "客队得分"),
@DataRequestParameter(name = "year", description = "年份"),
@DataRequestParameter(name = "month", description = "月份"),
@DataRequestParameter(name = "day", description = "天数"),
@DataRequestParameter(name = "time", description = "时间"),
@DataRequestParameter(name = "round", description = "轮次"),
@DataRequestParameter(name = "seasonId", description = "赛季", defaultValue = "2015", required = false)})
```

其中，AddGame 为 Bean 名称，也是类名称；DataRequestParameter 表示该接口需要的参数，其中“name”表示参数名称，“description”表示对参数名称的说明；required 表示该参数是否为必需。

API 类的属性和方法主要实现对数据的处理，例如验证必选参数是否有值、数据是否为脏数据等。将 API 从中央服务器的基础功能中分离出来，可极大增强系统的扩展功能。

2.2 信息抓取模块

信息抓取模块提取信息的本质是利用 Chrome 扩展自带的功能，对需要抓取信息的网页进行 JavaScript 脚本注入，然后利用 JavaScript 从 HTML 提取信息。但是信息抓取模块还需要其它功能，例如服务器版需要从中央服务器获取待抓取网页信息，在抓取结束时自动关闭标签页等；普通用户版需要控制是否运行扩展等，因而不仅需要信息提取脚本，还需要其它脚本。

2.2.1 manifest.json 配置

manifest.json 文件是 Chrome 扩展的核心配置文件，对哪些网页注入哪些 JavaScript 脚本进行控制。为了从 soda 抓取内容，manifest.json 文件核心代码如下：

```
{
  //省略次要内容
  "backgroud": {
```

```
        "persistent": true,
        "scripts": ["common－3.2.0.js","core.js","backgroud.js"]
      },
      "permissions": ["notifications","storage","tabs","cookies","http://sodasoccer.com/*"]
      "content_scripts": [
        {
          "matches": ["http://www.sodasoccer.com/*"]
    ,      "css": ["core.js"],
          "js": ["common－3.2.0.js","core.js","soda.js"]
        }
      ]
    //省略次要内容
  }
```

其中，“background”属性表示 Chrome 扩展运行时会有脚本在后台运行，“script”属性中的 JavaScript 文件就是在后台运行的脚本文件。“content_script”属性控制哪些网页注入哪些脚本，其中“matches”属性决定对哪些网页注入脚本，“js”属性表示对网页注入哪些脚本。

2.2.2　backgroud.js 文件

background.js 文件是当 Chrome 扩展运行后会一直在后台运行的脚本文件。服务器版信息抓取模块用来控制浏览器打开哪些网页，即把从服务器获取的网页信息放入一个队列，background.js 用来控制该队列的内容，当从队列中拿出信息，则打开一个网页，如果队列为空则再从服务器请求下一批网页信息。

2.2.3　core.js 和 core.css 文件

core.js 和 core.css 文件是公有基础文件，core.js 文件主要用来和中央服务器交互，并且在调试时将页面提取到的信息按 core.css 文件申明的规则进行展示。

2.2.4　soda.js 文件

soda.js 文件是 Chrome 扩展信息抓取的核心文件，主要从网页中提取信息，并且如果网页有异步加载，则对异步加载网页进行处理。

(1)普通信息提取功能。为了简化 HTML 和 JavaScript 之间的操作，信息提取功能主要使用 JavaScript 的一个版本库 JQuery[14]。而从网页中提取信息主要使用 JQuery 的选择器[15]，即通过浏览器自带的“审查元素”功能查看某一类网站的网页元素，然后使用 JQuery 选择器从中提取内容。例如从 souda 中提取球队信息的代码如下：

```
$("li", ".xin").each(function(index, element) {
    if (index == 0) {
        chineseName = $(this).text().replace("简称:", "").trim();
    }else if (index == 1) {
        englishName = $(this).text().replace("英名:", "").trim();
    }else if (index == 2) {
        homeCourt = $(this).text().replace("主场:", "").trim();
    }else if (index == 4) {
        site = $(this).text().replace("官网:", "").trim();
    }
});
```

(2)提取网页中的动态链接。有时需要打开网页链接并从中抓取信息，但是某些网站的链接中含有动态生成的验证信息。传统爬虫系统很难解决该问题，因为其无法抓取网页的所有资源并使它们运行，然而对于 Chrome 扩展爬虫系统而言，这些信息与普通内容并无区别，因为提取的链接本身就是从运行的浏览器中获取，因而必定带有最新的验证信息。

(3)从包含异步加载技术的网页中提取数据。随着网站浏览的激增，传统的 Web 同步请求模式已难以适应高并发高密度的网络请求，异步加载方式被越来越多的互联网服务提供商所采用[16]。相对于传统 Web 技术，Ajax 技术的广泛应用，不仅在一定程度上促进了页面表现和页面数据的分离，而且使 Web 应用的交互特性、反应速度及柔性迈向了更高、更新的层次[17]。但 Ajax 技术对爬虫技术并不友好，传统的爬虫技术无法处理异步加载问题，即使专门针对异步加载的爬虫系统也需要调用浏览器 API 才能解决问题。但是基于 Chrome 扩展的爬虫系统只要使用 JavaScript 的 setTimeout 函数来控制信息抓取脚本的运行时机即可。例如，将信息提取功能放在一个函数中，然后使用 SetTimeout 函数延迟一定时间再执行信息抓取模块，这样可以确保网页中的异步加载信息完全加载结束。这样不仅解决了传统爬虫不能爬取异步加载的问题，而且相对于调用浏览器 API 大大降低了开发成本。

3　系统验证

3.1　安装扩展

安装爬虫扩展和 Chrome 其它扩展一样，根据安装程序来源可分为从 Chrome 应用商店获取和加载本地解压的扩展。对于本扩展，只需要在“chrome://extensions/”页面勾选“开发者选项”，然后点击“加载已解压的扩展程序”即可运行。

3.2　数据验证

由本文研究可知，系统可以从搜达足球[18]上抓取赛程。以英超为例，发现英超 2015/2016 赛季一共有 38 轮，每轮有 10 场比赛，总共有 380 场比赛。运行系统后，查询 game 表，发现 380 场比赛都已完全抓到。图 3 为数据库中存储的 2015/2016 赛季英超的所有比赛，为了方便显示，删除了某些属性。

1	id	matchName	seasonId	gameDate	homeTeamName	awayTeamName	round
2	1	英超	2016	2015-08-08 00:00:00	切尔西	斯旺西城	1
3	2	英超	2016	2015-08-08 00:00:00	莱斯特	桑德兰	1
4	3	英超	2016	2015-08-08 00:00:00	阿森纳	西汉姆联	1
5	4	英超	2016	2015-08-08 00:00:00	曼彻斯特联	托特纳姆	1
378	377	英超	2016	2016-05-07 00:00:00	利物浦	沃特福德	37
379	378	英超	2016	2016-05-15 00:00:00	埃弗顿	诺维奇	38
380	379	英超	2016	2016-05-15 00:00:00	曼彻斯特联	伯恩茅斯	38
381	380	英超	2016	2016-05-15 00:00:00	沃特福德	桑德兰	38

图 3　英超赛程

3.3　系统优点

通过上述实现过程可知，基于 Chrome 扩展的爬虫系统相较于传统爬虫系统具有以下优点：

(1)开发简单。直接从浏览器解析好的页面中提取信息，省去了传统爬虫系统最难的步骤——模拟浏览器。

(2)扩展方便。如果有新需求,只需要对新类型网页编写信息提取脚本——示例中的 soda.js 文件,再通过 manifest.json 配置即可。

(3)基本支持所有包含异步请求的网页,从中抓取到所需数据。

(4)可以降低被对方屏蔽的概率。由于是从浏览器解析的页面提取信息,因而对方服务器无法通过 user-agent 等浏览器信息来屏蔽,也不能通过 URL 中的验证参数来屏蔽请求,相对于服务器版可能会因为请求频繁被屏蔽。而普通版不会被屏蔽,因为是用户正常访问,所以不会对对方服务器造成额外的负载压力。

(5)安装使用简单。传统爬虫系统需要专业知识才能完成部署安装,而基于 Chrome 扩展的爬虫系统在安装上更加方便。

4 结语

本文针对传统爬虫系统开发难度大、无法处理异步请求、容易被屏蔽和使用不友好等缺点,提出了一种基于 Chrome 扩展的爬虫系统。本系统通过使用 Chrome 浏览器提供的功能,简化了开发难度,可以处理异步请求网页,降低了被屏蔽的概率且提高了用户使用的友好度,并且在开发过程采用面向接口开发思想,保持了系统的高可扩展性。但其仍存在一些缺点有待改进:①无法处理某些脏数据,因为只从网页中提取内容而不复用脚本文件,所以如果网页中还有动态隐藏的脏数据则无法处理;②服务器版扩展依然存在被对方屏蔽的可能性。

参考文献:

[1] 微博广告中心. 产品介绍[EB/OL]. [2015-08-18]. http://tui.weibo.com/intro/product/sea.

[2] 罗刚,王振东. 自己动手写网络爬虫[M]. 北京:清华大学出版社,2010.

[3] 张敏,孙敏. 基于限定爬虫的设计与实现[J]. 计算机应用与软件,2013,30(4):33-35.

[4] 罗成,程耀东,胡庆宝,等. 可配置聚焦爬虫设计与实现[J]. 核电子学与探测技术,2014,34(3):353-358.

[5] 郭若飞. 支持 Ajax 的 Deep Web 爬虫技术研究[D]. 苏州:苏州大学,2010.

[6] 林碧霞. 基于领域本体的主题爬虫研究及实现[D]. 成都:西南交通大学,2010.

[7] 张莹. 面向动态页面的网络爬虫系统的设计与实现[D]. 天津:南开大学,2012.

[8] 汪海. [Python]网络爬虫(二):利用 urllib2 通过指定的 URL 抓取网页内容[EB/OL]. (2013-05-13)[2015-08-20]. http://blog.csdn.net/pleasecallmewhy/article/details/8923067.

[9] 孔淼. 一看就明白的爬虫入门讲解:基础理论篇[EB/OL]. [2015-11-18]http://www.csdn.net/article/2015-11-13/2826205.

[10] Netty 项目组. Netty 官方首页[EB/OL]. [2015-08-18]http://netty.io/.

[11] 李林峰. Netty 权威指南[M]. 北京:电子工业出版社,2014.

[12] 李刚. 轻量级 Java EE 企业应用实战——struts2+Spring3+Hibernate 整合开发[M]. 第 3 版. 北京:电子工业出版社,2012.

[13] DAREN. MySQL 主从同步部署[EB/OL]. [2015-08-18] http://www.linuxidc.com/Linux/2012-12/76276.htm.

[14] THE JQUERY FOUNDATION. JQuery 官网[EB/OL]. [2015-08-8]. http://jquery.com/.

[15] PHP100 中文网. JQuery 在线手册,CHM1.7[EB/OL]. [2015-08-18]. http://www.php100.com/manual/jquery/.

[16] 邬柏. 支持 AJAX 的分布式爬虫系统的研究与实现[D]. 武汉:华中科技大学,2013.

[17] 潘杰,周传生. 基于框架的研究与实现[J]. 沈阳师范大学学报:自然科学版,2015,33(1):96-99.

[18] 北京求之易数据有限公司. 搜达足球英超联赛首页[EB/OL]. [2015-08-18]. http://www.sodasoccer.com/dasai/league/133.html.

(责任编辑:孙 娟)

Web Crawler System Based on Chrome Extension

Abstract: This article introduces a new crawler system based on Chromeextension in order to improve data collection efficiency from web pages and reduce consumption of the system resource from crawler. This crawler system uses chrome browser to analysis web pages to prevent shielding of crawling object and asynchronous loading of web pages problems, as well as the realization of structured data. Unattended active crawl can be achieved, and information can be grabbed at the time when users are browsing Web pages by selecting the common user extension and server extension. Front and back is separated in the whole system, and Program To Interface is used to cover it high expansibility. Finally, verifying efficiency and feasibility of the program by gaining premiership schedule from Sodasoccer website.

Key Words: Web Crawler; Chrome Extension; Netty