xScraper:基于 Web-Harvest 技术批量与深度获取无结构化 Web 信息

朱 焱 朱 凯
(西南交通大学信息科学与技术学院 成都 610031)

摘 要 通过分析 Web-Harvest 数据提取规则的设计原理,设计实现了一个 xScraper 系统。该系统的主要功能有:(1)定制设计满足不同需求的 Web 数据提取规则模板,驱动 Web-Harvest 内核进行无结构化信息提取;(2)批量可控提取同一网址中的 Web 信息(含图像);(3)跨网站深度提取主题相关信息;(4)提取 Web 信息元数据并将其转换为 XML 标签;(5)实现无结构化多媒体信息的数据库管理。应用结果表明,系统提供了超出 Web-Harvest 的加值功能,可满足不同的信息提取需求,其简单实用,便于扩展。

关键词 Web 信息提取,xScraper 系统,Web-Harvest 内核技术
中图法分类号 TP393.4 **文献标识码** A

xScraper: Bulk- and Deep-extracting Non-structured Web Information Based on Web-Harvest Techniques

ZHU Yan ZHU Kai
(School of Information Science and Technology, Southwest Jiaotong University, Chengdu 610031, China)

Abstract A system named xScraper was developed based on the data extraction rules investigation in Web-Harvest. 5 main functions of this system are (1) flexible specification of extraction rules to meet different application requirements; (2) controllable bulk non-structured data (incl. images) extraction from the same Web site; (3) deep extraction of topic-related information across many Web sites; (4) extraction of metadata from Web sites and transformation in to XML tags; (5) non-structured multi-media information management in databases. xScraper is a simple, practical and extendable system. It provides value-added services over Web-Harvest and can meet different requirements of Web information extraction.

Keywords Web information extraction, xScraper, Web-Harvest core techniques

1 前言

万维网作为重要的信息资源,正在成为各类智能信息处理(管理)系统的基本组成部分。从万维网中提取重要信息加以集成和管理,为 Web 信息检索、分类管理、知识发现和决策提供优良的数据基础是当前倍加重视的研究和应用领域。Web 信息提取方法很多,文献[1]将其分为 5 类。

1. 基于 HTML 结构法:将 HTML 文档解析为标签树或语法树,在此基础上半自动地产生提取规则,将其施用在类似结构的网页上,提取信息内容。

2. 自然语言处理法:通过应用过滤、词性标注、句法分析、语义标注等技术对文本中的词、句元素间的关系进行定义,导出提取规则。这类方法适用于 Web 含有大量文字内容的情形。

3. 包装器归纳法:通过分析一组训练用网页集的格式特点,归纳出基于分隔符的提取规则。应用这些规则提取相似网页。

4. 基于建模的方法:按照某个数据模型的基本要素(元祖、表、XML 标签等)定义一个模型。在 Web 页面中定位与该模型结构相符合的数据段落,加以提取。

5. 基于本体(ontology)方法:此方法确定应用域的本体知识,根据本体对网页中固定文字表达进行定位,利用这些定位点提取相关对象。与上述几种方法不同,此方法对网页信息的结构依赖较少。

近年来,大量 Web 信息提取方面的研究成果采用了上述策略,有些工作在此基础上进行了方法的扩展或组合。文献[2,3]主要应用上述第 1 类提取方案,将 Web 信息结构解析为树结构,根据树根结点到叶结点(信息内容)的路径信息,半自动地产生提取规则,以提取相同结构类型的 Web 信息。文献[4]的方法组合第 1 到第 4 类方法的特点,通过分析 HTML 文档的结构和信息词句与预定义模式的匹配关系,推导出生成该 HTML 文档的后台模板,利用模板可以提取相同类型的 HTML 文档内容。文献[5]结合第 1 和第 4 种方法,根据 HTML 标签树分离出结构体,结合网页链接分类,产生

到稿日期:2012-02-12 返修日期:2012-06-25 本文受中央高校基本科研业务费专项基金(云计算与智能技术,SWJTU11ZT08),铁道部科技研究开发计划重大课题(2011X007)资助。

朱 焱(1965—),女,博士,教授,主要研究方向为数据挖掘、Web 资源质量挖掘、Web 信息智能管理,E-mail:yzhu@swjtu.edu.cn;朱 凯(1986—),男,学士。

模板，利用模板对同类 Web 信息进行提取。文献[6]利用分类技术，通过标注 Web 信息片段的结构形成样板，将待提取的信息片段与样本之间的相似度作为分类的依据，从而直接从分类的角度提取信息。文献[7]的信息提取基于两类本体。一是与特定网站无关的外部信息，称为领域本体；另一类是针对特定网站的应用本体。应用本体中最重要的概念是 HTML 语法树的路径表达式，其用来具体引导程序，完成信息提取。Web-Harvest1.0 是 2007 年 10 月推出的 Web 信息提取开源工具[8]。它融合了上述第 1、第 2 和第 4 种 Web 信息提取方法的特点，组合了 XSLT、Xquery、正则表达式等多种技术，旨在从信息记录和页面两级提取信息，是一个可扩展的实用的软件工具。

由于获取 Web 信息的应用需求不同，希望能够充分利用 Web-Harvest 的提取机制，灵活组合它的核心技术，建立用户可自定义的信息提取规则平台，进行大批量和跨网站的(半)自动化信息提取，完成提取中文元数据并将其自动转换为 XML 标签以及管理信息数据库等多项任务，因此本文在 Web-Harvest 内核之上设计实现了一个 xScraper 系统。该系统实现以下主要功能：

(1) 定制设计满足不同需求的 Web 数据提取规则模板，驱动 Web-Harvest 内核进行 Web 信息提取；

(2) 通过分页数或条数实现对同一网站的信息(含图像等)的可控批量提取；

(3) 根据网址超链结构实现主题相关的 Web 信息跨网站深度提取；

(4) 支持中英文 Web 元数据提取，并实现将中英文元数据转换为 XML 标签；

(5) 实现 Web 信息的数据库管理，为后续数据分析和知识发现做数据准备。

2 Web-Harvest 的工作原理

2.1 管线处理方式

Web-Harvest 的提取过程是通过定义 XML 格式的配置文件来完成的，配置文件中需要组合 BeanShell、XPath、XQuery 等技术，定义各种信息提取处理器。一个处理器执行的输出是另一个处理器的输入，因此可以管线的方式执行提取任务。表 1 是一个提取 Web 图片的配置文件，对应的提取原理如图 1 所示。

表 1 一个 Web 图片提取配置文件

```
〈? xml version="1.0" encoding="GB2312"?〉
〈config charset="GB2312"〉
 〈var-def name="url"〉http://www.swjtu.edu.cn/web.htm〈/var-def〉
  〈var-def name="urlList"〉
   〈xpath expression="//img/@src"〉
    〈html-to-xml〉〈http url="${url}" /〉〈/html-to-xml〉
   〈/xpath〉
  〈/var-def〉
  〈loop item="link" index="i"〉
   〈list〉〈var name="urlList" /〉〈/list〉
   〈body〉
    〈file action="write" type="binary" path="images/${i}.gif"〉
     〈http url="${sys.fullUrl(url,link)}" /〉
    〈/file〉
   〈/body〉
  〈/loop〉
〈/config〉
```

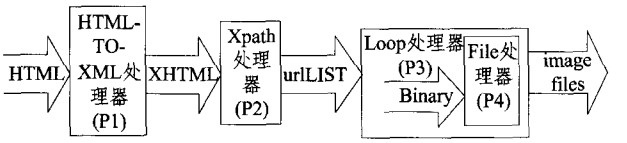

图 1 基于表 1 规则的图片提取原理

在图 1 中，给定一个起始 URL(Uniform Resource Locator)，如 http://www.swjtu.edu.cn/web.htm，该 URL 指向一个 HTML 文档。处理器 P1 先将该文档预处理为符合 XML 规范的 XHTML，然后根据 P2 定义的提取规则获取文档中所有图片的超链地址并存放在 urlLIST 中，在 P3 循环处理中对 urlLIST 中的图片逐一定位，P4 从网站中下载对应图片存放在本地的 images 目录中。

2.2 数据类型和变量的使用

Web-Harvest 内核常用 3 种数据类型：文本(text)、二进制(binary)和表(list)。空值是一种特殊的数据类型，如空字符串和空数组等。处理器可按默认或显示指定的方式处理数据类型，如图 1 中，P4 的默认类型为文本。由于图片须按二进制形式处理，因此通过显式定义数据类型为 binary 来提取图片。

Web-Harvest 没有限制变量命名，w[1]、345 或！#%等都是合法的变量名，但动态执行脚本或模板时，某些变量名会造成程序异常，因此通常遵循程序语言中通用的变量名定义方法。

2.3 提取规则模板设计和脚本语言

信息提取规则采用脚本语言撰写。Web-Harvest 支持多种脚本语言，如 BeanShell、Groovy 和 Javascript。撰写配置文件可以选用任一种语言，也可组合不同的脚本语言。

配置文件中用“${”和“}”标注提取模板，解析引擎据此识别模板并完成解析和执行。表 1 中有两个模板：(1) path="images/${i}.gif"是根据循环次数为图片文件命名；(2) url="${sys.fullUrl(url,link)}"是调用系统函数 fullUrl()对图片的 URL 进行处理。

3 xScraper 系统设计

xScraper 在 Web-Harvest 内核技术之上提供了 3 类信息提取功能，以及配置文件编辑和数据导出管理辅助模块。图 2 显示了 xScraper 的模块结构。

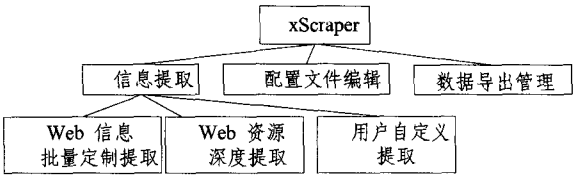

图 2 xScraper 功能模块结构

3.1 信息提取主功能模块

信息提取主功能分为 Web 信息批量定制提取、深度提取和用户自定义提取。此外，提取元数据转换为 XML 标签和应用数据库技术管理所提取的 Web 信息也是主功能模块的重要部分。

3.1.1 Web 信息批量定制提取

Web 信息量很大，许多网站使用分页方法组织信息，为

此该模块在配置文件中设定所需的信息条数或页数，从而实现用户指定的某个网站中各类信息的批量可控提取。表 2 是批量提取信息的配置文件片段。

表 2　Web 信息提取量控制设计举例

```
〈var-def name="linkList"〉
 〈call name="getItem"〉
  〈call-param name="pageUrl"〉〈var name="url" /〉〈/call-param〉
  〈call-param name="itemPerPage"〉20〈/call-param〉
  〈call-param name="startIndex"〉〈var name="startIndex" /〉〈/call-param〉
  〈call-param name="endIndex"〉〈var name="endIndex" /〉〈/call-param〉
  〈call-param name="pageMarker"〉page〈/call-param〉
  〈call-paramname="nextXPath"〉//div[@id='divTopPageNavi']//a[@name='link_page_next']/@href〈/call-param〉
  〈call-param name="keyXPath"〉//div[@class="list_r_list"]//a[@name="link_prd_name"]/@href〈/call-param〉
 〈/call〉
〈/var-def〉
```

配置文件由 getItem 函数调用和参数传递组成。getItem 函数是 Beanshell 形式的提取规则，实现信息定位和信息循环提取。该函数接收外部传递的 7 个参数，执行后返回用户所需要的结果(URL 列表)。这一片段之后的工作将采用类似表 1 的规则，逐一处理 URL 地址，获得其所指向页面的信息，完成提取。传递给 getItem 函数的 7 个参数作用如下：

1. pageUrl：信息提取的开始网址(起点)。

2. itemPerPage：每页的信息项数。

3. startIndex：从第 startIndex 项开始提取信息。

4. endIndex：到第 endIndex 项结束提取信息。

5. pageMarker：获取当前页的第一项信息在全部信息项中的序列数。如“http://search.book.dangdang.com/search.aspx?key=java&p=10”中，PageMarker 获取 p 的值。例中 $p=10$ 指明现在需要提取信息总数中的第 11 条。又例如“http://www.baidu.com/s?wd=%B9%FA%C7%EC&pn=5”中，PageMarker 获取 pn 的值。

6. nextXPath：寻找下一页链接的 XPath 路径表达式。

7. keyXPath：定位需要提取的信息内容的 XPath 路径表达式。

在这个功能模块中，通过分析网页 HTML 结构，组合 Xquery 与 XPath 技术，设计了模糊提取规则，构建了适应面广的 Web 信息提取配置文件，从而能使用一个配置文件尽可能提取最多类型的信息。例如，xScraper 将当当网的商品信息粗分为图书、音乐、影视、其他(包括所有剩下的类别)4 类，每类又可分为多个子类，涵盖了海量信息。提取每类信息只需分别设计配置文件。表 3 给出了模糊提取的配置文件片段。例如使用了 XPath 的//div 进行模糊定位，然后使用 XQeury 的 substring-after 方法对模糊定位所得到的结果进行精确处理，以获得准确的信息内容。表 3 只给出了一种处理方案的例子。

在这个功能模块中，xScraper 还实现从网页中分别提取信息自带的元数据和信息内容，将元数据转换为 XML 标签，数据作为对应标签的元素内容。例如，图书信息中的“作者：古龙”，提取后的 XML 输出为〈作者〉古龙〈/作者〉。这样处理使得提取出的信息仍然保持了 Web 信息原有的语义关联，减少了信息损失，提高了信息提取的自动化程度。

表 3　提取当当网图书信息的模糊提取规则的片段

```
declare variable $item as node() external;
declare variable $imgPath as xs:string external;
  let $name:=data( $item//span[@class="black000"]//b )
  let $author:=translate(data(substring-after( $item//div[@id="author_"],":" )),"&#3000;著编译","")
...
  let $price:=data( substring-after( $item//div[@class="price"]/span[1],"¥" ) )
  let $curPrice:=data( substring-after( $item//div[@class="price"]/span[2],"¥" ) )
  let $count:=data(substring-before( $item//div[@class="price"]/span[3],"折"))
  ...
  let $save:=if ( $hasVip)
     then data(substring-before(substring-after( $item//div[@class="price"]/span[3]/following-sibling::text()[1],"¥"),"  "))
    else substring-after( $item//div[@class="price"]/span[3]/following-sibling::text()[1],"¥")
...
```

提取时，xScraper 将自动关联所对应的配置文件，根据用户的关键字和所需信息量搜索与关键字相匹配的相关内容，按照配置文件中的规则提取出相关信息。提取完成后，可浏览结果或调用数据导出模块，将信息导出到数据库中进行管理。

3.1.2　Web 资源深度提取

xScraper 的 Web 信息深度提取功能旨在以某网页为起点，沿链接跨网站深度追踪同一主题信息，并可逐一提取相关内容。该功能适合新闻主题信息的溯源和追踪获取，因为大量相同主题的信息内容并不仅仅存放在同一网站中，需要沿超链路的嵌套层次纵深挖掘，从一个网址跳转到另一网址。该功能具有很大的动态性和提取规则的不确定性。

xScraper 深度提取的原理是：调用配置文件搜索起始网址，获得第一个信息页，在该页中检索与主题词相匹配的所有链接，获得一组 URL。每个 URL 代表的网页又包含另一组主题相关的 URL 列表，从而形成一个树结构，树结构的深度由提取深度决定。按深度优先策略遍历该 URL 树，提取信息。提取过程结束条件是：(1)纵向达到指定的提取深度；(2)横向将匹配主题词的超链遍历完毕。

在深度信息提取配置文件中，首先确定初始链接(起点)、主题词、提取深度以及提取出信息的存储设置，然后设计两个 BeanShell 函数(isValidUrl()和 getFullUrl())来规范 URL 格式。配置文件的关键是根据预定义的深度设计嵌套循环进行关键字匹配和信息提取。最内层循环首先使用 XPath 判断页面中的所有链接是否包含关键字，包含关键字的链接构成一张表 List。依次处理 List 中的链接，提取链接所指向的信息。List 中的链接处理完毕后，返回外一层。在这一层中，系统遍历本层的未访问链接表，如果下一个链接尚未访问，则将该链接放到已访问链接表中，然后以该链接为输入参数调用最内层循环；如果未访问链接表已遍历完毕，则返回更外一层。在最外层循环中，系统检测本层的未访问链接表是否为空，如为空表明所有链接已全部处理完毕，输出提取的信息内容，提取过程结束。

3.1.3　用户自定义提取

xScraper 预定义通用配置文件，用户在此基础上定义初始化参数，将参数传递到配置文件中形成用户自定义的信息提取规则，随后驱动 Web-Harvest 内核开始执行配置文件完

成信息提取。

该功能的另一个用处是调试配置文件。通过设置各种参数并传递给配置文件，然后检查信息提取正确与否，来判断配置文件的正确性，以便进一步完善。

3.2 配置文件编辑和提取后的 Web 信息管理模块

（1）配置文件编辑

该功能模块的主要作用是对配置文件进行编辑。设计出的配置文件可以通过调用 Web-Harvest 的内核，使用监听器来监听运行时的各个线程信息，并将信息显示在信息窗口，以完善配置文件的各项功能。

（2）数据导出管理

该功能将提取出来的 Web 信息存放到关系数据库中，以利于结构化地管理 Web 数据。该功能集成在 Web 资源批量定制提取功能中。

xScraper 首先检测提取出的 Web 信息是否为 XML 文档形式。如果是，则合并相同语义的 XML 标签，抚平 XML 的嵌套层次，将已转换为 XML 标签的元数据作为关系表的属性创建数据库关系模式。随后，xScraper 将 XML 中的信息内容添加到数据库表中。xScraper 用不同的表管理从不同网站提取出的信息。

4 xScraper 系统实现

4.1 Web 信息批量定制提取的实现

首先，根据应用需求设计正确的 Web 信息提取配置文件。随后，xScraper 分析配置文件，动态加载网页，从用户界面获取参数，调用配置文件，驱动 Web-Harvest 内核完成信息批量提取。提取出的信息按用户设定的路径自动以 XML 文档格式存放。用户可选择将这些信息导出到数据库中加以管理。图 3 显示了从网页中提取出的图书信息，图 4 是用关系数据库表管理已提取的数据。

```
-<书>
   <名称>我的爸爸是巨人 My Daddy Is a Giant</名称>
   <作者>Carl Norac</作者>
   <出版社/>
   <出版时间>2005-5-1</出版时间>
   <isbn>9781405021685</isbn>
   <定价>60.00</定价>
   <当当价>57.00</当当价>
   <折扣>95</折扣>
   <节省>3.00</节省>
   <字数/>
   <版次>1</版次>
   <页数/>
   <印刷时间/>
   <开本>大16开</开本>
   <印次>1</印次>
   <纸张/>
   <包装>平装</包装>
   <图片路径>C:/Users/Kamil/Desktop/daddy/img/daddy_2.JPEG</图片路径>
 </书>
```

图 3 提取出的图书信息

19	九州：英雄	唐缺	新世界出版社
20	欢乐英雄（绘图珍藏本）一古龙作品集	古龙	珠海出版社
21	英雄书	（西）葛拉西安，李汉昭	安徽教育出版社
22	说富天下 垂 楞福的帝国（一部讲述血色王朝帝王将...	巽天较云	花山文艺出版社
23	英雄人物故事/小学生语文课外阅读丛书	潘炼静 改	世界知识出版社
24	和爸爸去购物 I SHOP WITH MY DADDY	本社	[illegible]
25	[illegible]	Carl Norac	[illegible]
26	爸爸的女孩DADDY'S GIRL	Lisa Scottoline	音像供货

图 4 已提取的信息存放在数据库表

4.2 Web 资源深度提取

此功能的实现难点在于动态监测信息的提取过程，确定信息追踪深度，判断深度搜索的入口、出口条件。系统使用了 Web-Harvest 的 ScraperRuntimeListener 接口，该接口预定义 7 种运行状态，使系统能监测信息提取过程并及时根据不同状态进行处理。图 5 的实例片段以 http://news.google.cn 为起始 URL，主题词为"国庆"，搜索深度为 5，提取出关于"国庆"信息的不同网站的 URL 集合。

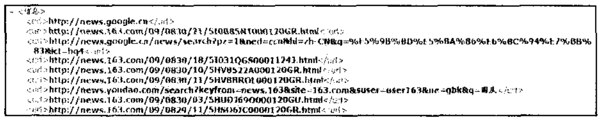

图 5 Web 信息深度提取的结果举例

4.3 配置文件编辑的实现

配置文件编辑模块中同样使用 ScraperRuntimeListener 接口对每个线程的状态进行跟踪，并返回运行状态，在信息窗口中显示正在运行的线程情况。图 6 显示了配置文件编辑界面。

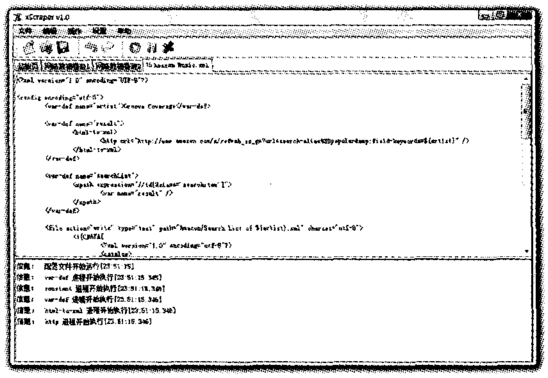

图 6 配置文件编辑界面

结束语 本文通过分析 Web 信息提取需求，剖析 Web-Harvest 软件内核，设计了一个 xScraper 系统。该系统实现了 Web 信息的提取规则的设计与调试、Web 信息的批量可控提取与深度提取、中英文 Web 元数据提取及转换、Web 信息的数据库管理。本系统组合运用了 Java、XML、XPath、Xquery、BeanShell、SQL 技术，能很好地满足大批量和深度信息提取的应用需求，提供了 Web-Harvest 之上的加值功能，系统简单实用，便于扩展。

当前正在对 xScraper 的系统逻辑进行改进提高，并完善系统对配置文件错误的检查和报告机制。

参考文献

[1] Laender A H F, Ribeiro-Neto B A, da Silva A S, et al. A brief survey of web data extraction tools [J]. SIGMOD Records, 2002, 31(2): 84-93

[2] 李效东，顾毓清. 基于 DOM 的 Web 信息提取[J]. 计算机学报，2002, 25(5): 526-533

[3] 冀高峰，汤庸，道炜，等. 基于 XML 的自动学习 Web 信息抽取[J]. 计算机科学，2008, 35(3): 87-90

[4] Yang Shao-hua, Lin Hai-lue, Han Yan-bo. Automatic data extraction from template-generated Web pages[J]. Journal of Software, 2008, 19(2): 209-223

[5] 梅雪，程学旗，郭岩，等. 一种全自动生成网页信息抽取 Wrapper 的方法[J]. 中文信息学报，2008, 22(1): 22-29

[6] 李向阳. 基于竞争分类的 Web 信息抽取[J]. 电子学报，2004, 32(11): 1915-1917

[7] 高军，王腾蛟，杨冬青，等. 基于 Ontology 的 Web 内容二阶段半自动提取方法[J]. 计算机学报，2004, 27(3): 310-317

[8] Web-Harvest Team[OL]. http://Web-Harvest. sourceforge. net/. 2009