A Mini-project Report

on

Web Extraction Algorithm based on Ontology and DOM Tree

carried out as part of the course Web Technology and Applications (IT302)

Submitted by

Abhishek R S (12IT02)
Nikhil V(12IT46)
Pradnyesh Vineet Joshi(12IT49)
Shashank S R(12IT58)
Raju S N

V Sem B.Tech (IT)

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY
in
INFORMATION TECHNOLOGY

Department of Information Technology

National Institute of Technology Karnataka, Surathkal

November 2014

CERTIFICATE

This is to certify that the project entitled **"____________________Web Extraction Algorithm based on Ontology and DOM Tree**

____

___" is a bonafide work carried out as part of the course **Web Technology and Applications (IT302)**, under my guidance by _____________________________________, student of V Sem B.Tech (IT) at the Department of Information Technology, National Institute of Technology Karnataka, Surathkal, during the academic semester _____________________, in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Information Technology, at NITK Surathkal.

Place:

Date: Signature of the Instructor

DECLARATION

I hereby declare that the project entitled "**Web Extraction Algorithm based on Ontology and DOM Tree** " submitted as part of the partial course requirements for the course Web Technology and Applications (IT302) for the award of the degree of Bachelor of Technology in Information Technology at NITK Surathkal during the Jul - Nov 2014 semester has been carried out by me. I declare that the project has not formed the basis for the award of any degree, associateship, fellowship or any other similar titles elsewhere.

Further, I declare that I will not share, re-submit or publish the code, idea, framework and/or any publication that may arise out of this work for academic or profit purposes without obtaining the prior written consent of the course Faculty Mentor and Course Instructor.

Signature of the Student:

Place:

Date:

TABLE OF CONTENTS

Introduction

1.Literature Survey

1. **Background**
2. **Outcome of Literature survey**
3. **Problem Statement**
4. **Objectives**

2. Methodology

3. Implementation

1.**Work done**

2**. Result and Analysis**

4. Conclusion

5. Future Work

References

Abstract

Due to the information on the Web being tremendous, dynamic and irregular, it is difficult to search and integrate information from the Web. The paper proposes a Web information extraction algorithm based on Ontology and DOM tree. The areas are accurately found out and the interested information is extracted exactly by information extraction rules generated by ontology. Furthermore this algorithm implements information extraction through traveling DOM tree. Finally, we implement information extraction system and test its performance on news site.

INTRODUCTION

Web is a giant resource of data. Some of the websites represent this data in a well-structured format like JSON which allows easy downloading of data for further analysis and study. But, not all websites provide data in such a format. In such cases a process called Web Scraping or Web extraction is used. Web Scraping is a computer software technique which simulates human exploration of web.

Web scraping is widely used for product comparison, combining data from different websites to form a integrated data set. In one line it's the automation of web surfing to find required data.

Web scraping is closely related to web indexing, which indexes information on the web using a bot or web crawler and is a universal technique adopted by most search engines. In contrast, web scraping focuses more on the transformation of unstructured data on the web, typically in HTML format, into structured data that can be stored and analyzed in a central local database or spreadsheet. Web scraping is also related to web automation, which simulates human browsing using computer software. Uses of web scraping include online price comparison, contact scraping, weather data monitoring, website change detection, research, web mash up and web data integration.

Usually, such software programs simulate human exploration of the World Wide Web by either implementing low-level Hypertext Transfer Protocol (HTTP), or embedding a fully-fledged web browser, such as Internet Explorer or Mozilla Firefox.

LITERATURE SURVEY

1. Background

Web scraping is the process of automatically collecting information from the World Wide Web. It is a field with active developments sharing a common goal with the semantic web vision, an ambitious initiative that still requires breakthroughs in text processing, semantic understanding, and artificial intelligence and human-computer interactions. Web scraping, instead, favors practical solutions based on existing technologies that are often entirely ad hoc. Therefore, there are different levels of automation that existing web-scraping technologies can provide:

2. Outcome of Literature survey

There are many techniques used in web scraping. Human copy / paste is the crudest representation of web scraping. Techniques like HTML and DOM parsing are prominent.

- **Human copy-and-paste** : Sometimes even the best web-scraping technology cannot replace a human's manual examination and copy-and-paste, and sometimes this may be the only workable solution when the websites for scraping explicitly set up barriers to prevent machine automation.

- **Text grepping and regular expression matching:** A simple yet powerful approach to extract information from web pages can be based on the UNIX grep command or regular expression-matching facilities of programming languages

- **HTML parsers**: Many websites have large collections of pages generated dynamically from an underlying structured source like a database. Data of the same category are typically encoded into similar pages by a common script or template. In

data mining, a program that detects such templates in a particular information source, extracts its content and translates it into a relational form called a wrapper. Wrapper generation algorithms assume that input pages of a wrapper induction system conform to a common template and that they can be easily identified in terms of a URL common scheme. Moreover, some semi-structured data query languages, such as XQuery and the HTQL, can be used to parse HTML pages and to retrieve and transform page content.

- **DOM parsing**: By embedding a full-fledged web browser, such as the Internet Explorer or the Mozilla browser control, programs can retrieve the dynamic content generated by client-side scripts. These browser controls also parse web pages into a DOM tree, based on which programs can retrieve parts of the pages.

- **Vertical aggregation platforms**: There are several companies that have developed vertical specific harvesting platforms. These platforms create and monitor a multitude of “bots” for specific verticals with no man-in-the-loop, and no work related to a specific target site. The preparation involves establishing the knowledge base for the entire vertical and then the platform creates the bots automatically. The platform's robustness is measured by the quality of the information it retrieves and its scalability . This scalability is mostly used to target the Long Tail of sites that common aggregators find complicated or too labor-intensive to harvest content from.

- **Semantic annotation recognizing**: The pages being scraped may embrace metadata or semantic markups and annotations, which can be used to locate specific data snippets. If the annotations are embedded in the pages, as Microformat does, this technique can be viewed as a special case of DOM parsing. In another case, the annotations, organized into a semantic layer, are stored and managed separately from

the web pages, so the scrapers can retrieve data schema and instructions from this layer before scraping the pages.

- **Computer vision web-page analyzers**: There are efforts using machine learning and computer vision that attempt to identify and extract information from web pages by interpreting pages visually as a human being might.

All of the web extractors heavily use the Document Object Model (DOM) which is a cross-platform and language-independent convention for representing and interacting with objects in HTML, XHTML and XML documents. The nodes of every document are organized in a tree structure, called the DOM tree. Objects in the DOM tree may be addressed and manipulated by using methods on the objects. The public interface of a DOM is specified in its application programming interface . The paper presents under implementation uses xml DOM for data storage and extraction . All the links are crawled by a 'crawler' .

3. Problem Statement

Web page extraction from a web-site using DOM and Xpath . The HTML documents are analyzed using the ontology and can be converted to XML DOM. The interested nodes in XML DOM are extracted. The extracted nodes in the DOM tree can be located by XPath, and information can be extracted through using Xpath API. The paper mentions an algorithm for the process which would be implemented.

4. Objectives

The objective of this project is to learn and understand the usage of web semantics for developing better web related applications . The information extraction provides valuable insights into the working of the Web which can be used for further study .

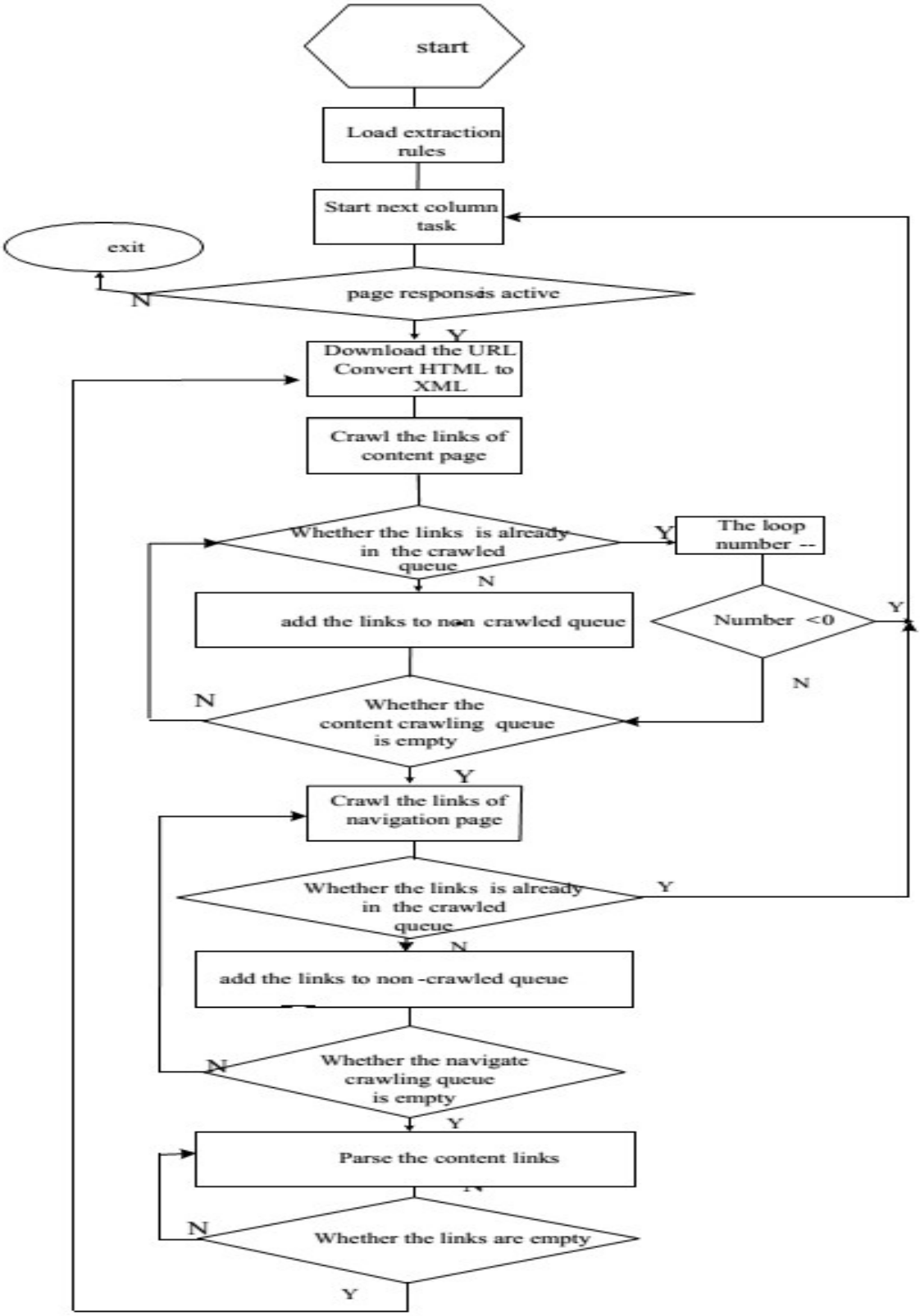
Our aim is to extract the information from a News Website for offline viewing. The paper gives an ontology of web page. Ontology, being the description of the concepts in a certain domain, is developed by the authors by asking domain experts and researching a lot of websites.

Methodology

The IEEE paper under implementation discusses about a methodology to scrape a website using Web Page Ontology to form extraction rules for xpath and use of regular expression.

The HTML documents analyzed can be converted to XML DOM. The extracted nodes in the DOM tree can be located by XPath, and information can be extracted through using XpathAPI. The following figure describes the flow chart of information extraction algorithm.

Usually, in order to judge whether the news has been crawled or not we need to know whether its headlines, release time, contents and other fields have been extracted. This paper presents a method which only needs to compare the page URL to infer whether the page has been processed or not. The illation algorithm eliminates the crawled content page and the influence of page noise on the page information processing effectively, which also improves the efficiency of the program greatly.


start
Load extraction rules
Start next column task
exit
page responses active
N
Y
Download the URL Convert HTML to XML
Crawl the links of content page
Whether the links is already in the crawled queue
Y
The loop number --
N
add the links to non crawled queue
Number <0
Y
N
N
Whether the content crawling queue is empty
Y
Crawl the links of navigation page
Whether the links is already in the crawled queue
Y
N
add the links to non -crawled queue
N
Whether the navigate crawling queue is empty
Y
Parse the content links
N
Whether the links are empty
Y

Implementation

1. Work Done:

A program to scrape a given web page has been developed. The programming language used for development is python .A crawler is used to crawl through the links of a website which stores all the links present on the web page in an array The crawling of the links uses Breadth First Search algorithm which is more accurate of a website with too many levels . This crawling is a simulated form of human crawling. The *urllib* module in python provides a simple http client interface for network resource access. The urlopen() function opens the url for reading and fetches the data across the world wide web. Using urllib library the links crawled and the html pages are searched for the required contents.

Once the required level of links are crawled the html files of the links are parsed for the required content which is established during the ontology construction of the website.

The required data is extracted from the web page using Xpath . XPath is used to navigate through elements and attributes in the html files and the content we are interested in is extracted. The scraped content, in turn, is to be placed into a well-structured xml file categorically. Mechanize and lxml modules are used to carry out this task. *lxml* module is used to build XML DOM. This library helps in parsing the xml . Using lxml implementation uses xpath to select nodes . Mechanize is a library used to simulate browsing to that of a human . This process can help in automation of web scraping in case of any changes to the website.

2. Results and Analysis:

The program fetches links without any noise, i.e no link gets repeated and there is no loss in accuracy. Only those elements which are requested are downloaded. The data obtained from the website is structured into an XML format for futher analysis . The experimental environment of this system are: Intel 2.8GHz CPU, 4.0Gmemory, Windows8 Operating system and python Programming language.

The information extraction system has an efficient recall and precision. For the page processing rate, the time of downloading picture is longer than the time of analyzing content, so this will decrease the extraction rate, but in general the extraction rate of this system is satisfactory.

Conclusion

Web scraping can be made faster and accurate with use of noble algorithms. Today most websites give APIs to extract data from their websites like Bing, Google and Facebook. However when such APIs are not given by the websites we can use web scraping. Web extraction can be made faster and accurate with the use of better algorithms. But due to the slow internet speeds the extraction may halt due server hang up, how to improve the extraction rate in such cases is our further research .

Future Work

The extraction process can be made faster by crawling only through the required links . This can be done by parsing the given HTML document to find the required tag or link , instead of crawling through all the links . In such case we can use Dept First Search as the crawling algorithm . In this method we can reduce the overload of opening unwanted web-pages thereby making extraction faster . Also when the structure of the web-page changes , new xpath values for extraction should be formulated . Automatically achieving this is requires further study.

References :

[1] Web Information Extraction based on Ontology and Dom Tree , IEEE paper

[2] www.w3schools.com/xpath

[3] www.w3schools.com/xmldom

[4] http://en.wikipedia.org/wiki/Semantic_Web

[5] http://en.wikipedia.org/wiki/Web_scraping