Research on Web Crawler and the News Collector

Abstract: Website information retrieval or special web information collecting both needs a web crawler. The design of web crawler will be introduced in detail. And the crawler will be used to build a desktop application to collect news from websites and show the news in browser. In the end, there is future work to do to improve.

Introduction

Nowadays, information explosion makes the Internet to an unprecedented level. Google said that the pages they indexed were more than one thousand billion. (Alpert & Hajaj, 2008) This huge number makes people to create more efficient search engine. In this research, one important part of search engine will be discussed, the web crawler.

The web crawler links into the Internet directly. It is the information source for the data. The performance of web crawler determines content of the whole system. A good web crawler can make the data variable and updating on time. The operating principle of web crawler is starting from an initial URL or a set of URLs; gets one URL and downloads the webpage; then extracts the URLs in that webpage and adds them to the URL queue; and repeat this process until reach some standard.

The operating principle is simple, but to design a high performance web crawler is a really challenge. Building an efficient web crawler should consider aspects below. First, it can crawl huge data from the Internet and can be improved by using more

hardware resources. Second, when a web crawler can finish it job, the distribute system is a good method to improve the performance. Third, web crawler should not visit a website in high frequency in a short time. It will affect the normal users and may be prohibited by the website. Last, it can do different crawling from different website, such Blogs and BBS.

Building a web crawler is the first part of this project. After collected data from the websites, the data will be analyzed and show to the users.

Overview

Stanford University designed the web crawler for Google. (Brin S & Page L, 1998) The early web crawler system of Google has five modules. A URL server reads the URL queue from the disk and sends it to web crawler. Each web crawler operates in one single machine, and using single-thread asynchronizated IO method to crawl 300 links. The crawler sends the webpages to store server and the server will compress and save the data. The index process gets links from the webpages and stores them in different files. A URL parser reads these link files and transforms them to absolute path for URL server. Then Google has improved it crawler by using Google File System, BigTable to store the data and using MapReduce technology to make distribute computing. (Ghemawat S， Gobioff H & Leung Shun-Tak, 2003)

Allan Heydon and Marc Najork, from Compaq System Research Center, designed a web crawler named Mercator. (Heydon A & Najork M, 1999) The system uses JAVA multi-thread synchronizatism method to achieve parallel computing. And there are

many strategies, such as DNS buffer and delayed storing, to improve the performance. Their data structure uses little space in memory whether the data is large or small. Mercator has five parts, sorting the URLs which are downloading; parsing the host name to IP address; using HTTP to download the files; extracting links from HTTP files and checking the link is visited or not.

Internet Archive makes a crawler, which can crawl 64 websites at the same time. (Burner M, 1997) The crawler reads URLs from the disk, and uses asynchronous IO method to download the webpages and extracts the links. If the link is local crawling, it will be put into waiting list and store in disk, and be sent to crawler periodicity.

UbiCrawler is a high performance web crawler. (Boldi P, 2004) It is a distribute system and has high error-tolerant rate. It has many features, platform independence, efficient distribute algorithm, totally distributed modules and no one-point error problem.

IRLBOT is a large scale crawler developed and published by TAMU. (Lee, 2008)

As they declared, about 6 billion webpages has been crawled by this tool, and can han dle the crawling on tens of billions of webpages. IRLBOT is stretchable, and has great contribution “polite” crawling and dump page eliminating.

Maze is one of the crawlers published by Chinese people with high efficiency. (Maze, 2005) While developing this crawler, the programmers in Peaking University used dis tribute structure instead of centralized one. Maze can handle crawling on billions of w ebpages. Its two stage hash strategy based on websites solved the problem of dynamic

joining and exiting of crawlers while doing searching on the website.

Body

1. The basic process

At the beginning of this design, I planned to make a web crawler all by myself. So, I read many papers about that and get a solution.

The basic process of the web crawler is shown below. The web crawler will take web link from UnChecked URL database, the database is shown as table 1. And then it will check whether the link is available or not. The link is available means this link is not marked as visited. If the link is available, the HTML file will be stored in local disk. And then all the links in that HTML file will be extracted and added into the bottom of UnChecked URL database.

Name	Description	type	PK	NULL
URL_id	the id of record	number	no	no
URL	the URL of the HTML file	string	yes	no
UpdateCycle	update cycle	date	no	no
checked	whether the link visited	bolean	no	no

table 1: the UnChecked URL database

2. Database design

The UnChecked URL database is used to store the links which are extracted from HTML file. And then the crawler gets the link from this database, and visits next HTML file and downloads the page. The page will be stored in local disk.

In this database, the URL is the primary key. This design can prevent getting same link in the database. The checked is used to mark the start point for the web crawler. When the crawler gets link from the database, the checked will be marked 0. And the

new URLs which extracted from the HTML file will be marked 1 when stored into the database. When the web crawler restarts, it can continue previous work based on the checked value. This design is to avoid re-handling.

3. The algorithm of crawling

The extraction of links is one of the important parts of the design of web crawler. The algorithm of extracting links is shown below:

1) Create a method to matching text in HTML file
2) Create a reader to read the HTML file line by line. If the line has <a href=*>, get the content after the equal sign, if not, return to step 2.
3) Match the content after the equal sign with http protocol's header. If matched, go to step 4; if not, return to step 2.
4) If the content does not have ".jpg", ".swf" and other type of files, go to step 5; if not, return to step 2.
5) The current content is a link of webpage, add it to the bottom of UnChecked URL database, and return to step 2.

In order to avoid the crawling process goes into an infinite loop. I make the URL to be the primary key, and set a update cycle. If the cycle is too short, the burden of the system is too heavy. If the cycle is too long, the modification of the checked link would not be found. So, the length of the update cycle is a problem worthy to be discussed.

A large part of web information is generated dynamically by program or script accessing database. The part of information is hard to collect, because of the

nondeterminacy. So, this system will only collect the information from static webpage.

4. The implementation of design

When I start to do the details design, I find to make a web crawler is a hard work. For example, the method to match the text in HTML file is not that easy, and there are a lot of errors in the matching result. So, I find an open source HTML parser to help me finish this design. It is the jsoup, a Java HTML parser. Jsoup is a Java library for working with real-world HTML. It provides a very convenient API for extracting and manipulating data, using the best of DOM, CSS, and jquery-like methods.

This design is used to collection news from websites and shows them to users. So, I analyze several popular websites, and get to know their URL format. For example, I find the URL of sina.com.cn has this format: sports.sina.com.cn/g/laliga/2013-12-01/08136911563.shtml. The first section says this link is sports news. “g/” means this is a global one. “laliga” means this news is in a special subject for Real Madrid. And the link shows the date of news and it ID. Using the result of analysis of the link format, and with the HTML parser, I modify the algorithm for each website and subject which the data will be collected from. For example, I provide a function to return then sports news to users from sina.com.cn. The initial URL is the “http://www.sina.com.cn/”. And the crawler will match the links which have a “sports” in the first section of the URLs.

To make the result quickly shown to users, and to make sure the users get latest news, I design a quick look function. In this function, users will get the latest headline news

from the front page of websites. The links will be shown to users as buttons with a title of short summary. Users can click the button, and the browser will open the page. And the links will not be stored in database, because the links may be already in the database.

Result

I use Java to build this News Collector. By using “JFrame”, I make it a desktop application. It has mainly three kinds of class, the class to show the function menu and result, the class to crawler data and store the data to database, and the class to achieve quick look function. By now, I finish the data crawling function of two websites, and the quick look function of one of them. The main menu and the quick look function are shown below.

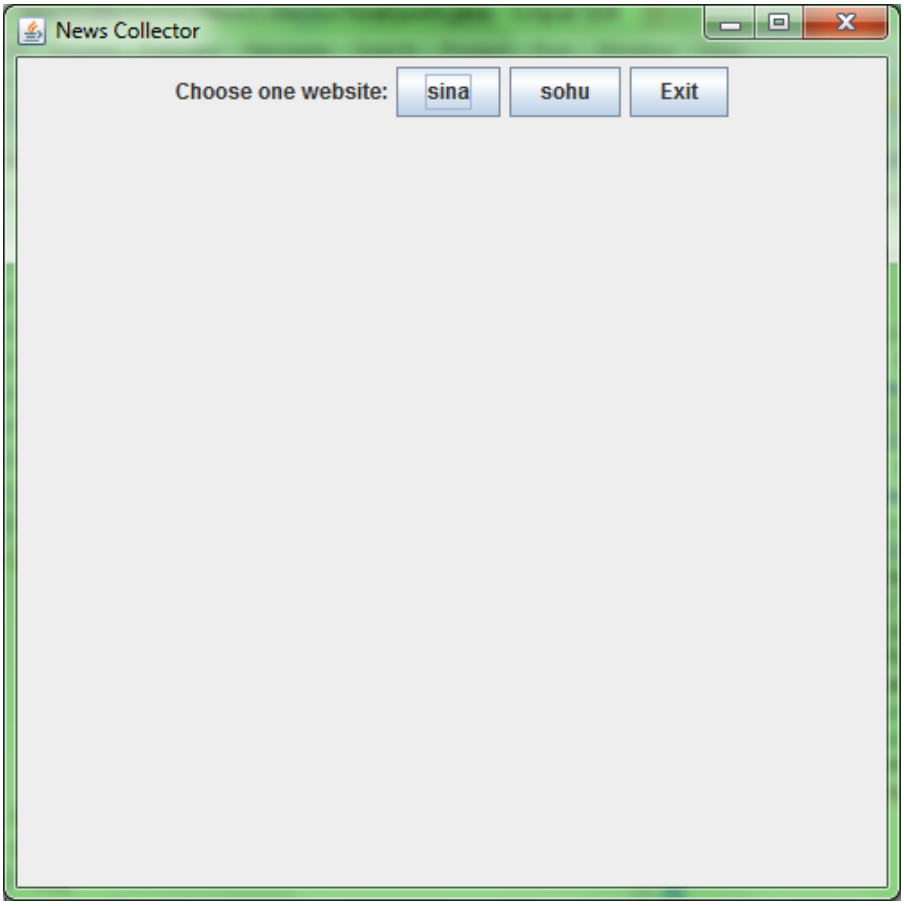

figure 1: the main menu of News Collector

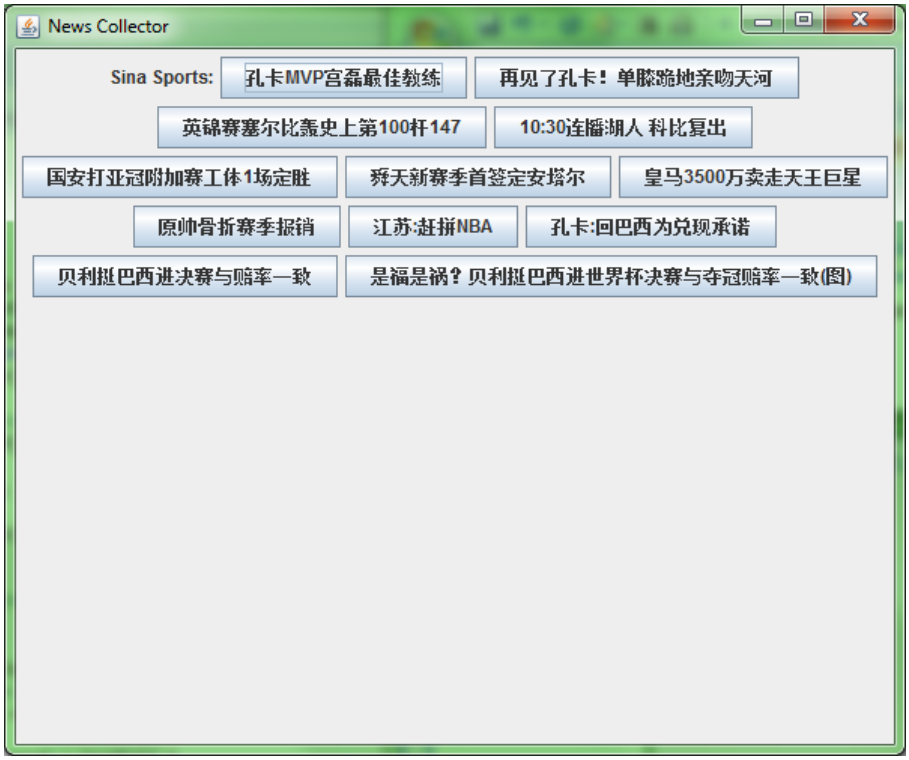

figure 2: the quick look function

Future Work

According to my design proposal, I have some work to do in the future. Achieving more data crawling from more websites is a task to make the design have more content to show to the users. So, I need to analyze more websites, get to know their URL format to modify the algorithm to each of them. And I have to modify the UI for this design. The current version is just used to test the function and used to do the demo.

Reference

Alpert, J & Hajaj, N (2008). We knew the web was big. http://googleblog.blogspot.corn/2008/07/we-knew-web-was-big.html, 2008-07-25

Brin S & Page L(1998). The Anatomy of a Large-scale Hypertextual web Search Engine. *Computer Networks*, 1998, 30:107-117

Ghemawat S, Gobioff H & Leung Shun-Tak. The Google File System. *Proceedings of the 19th ACM Symposium Oil-Operating Systems Principles*. 2003: 20-43

Heydon A & Najork M. Mercator: A scalable, extensible Web crawler. *World wide web*, 1999, 2(4): 219-229

Burner M. Crawling towards Eternity: Building an Archive of the World Wide Web. *Web Techniques Magazine*, 1997, 2(5): 125-130

Boldi P, Codenotti B. Santini M. UbiCrawler: A Scalable Fully Distributed Web Crawler. *Software: Practice & Experience*. 2004, 34: 711-726

Lee Hsin-Tsang, Leonard D. IRLbot: Scalingt06 BillionPages and Beyond. *Proceedings of the 17th International World Wide Web Conference*. ACM Press, 2008: 427-436

Maze. http://e.pku.edu.cn, 2005-05-06