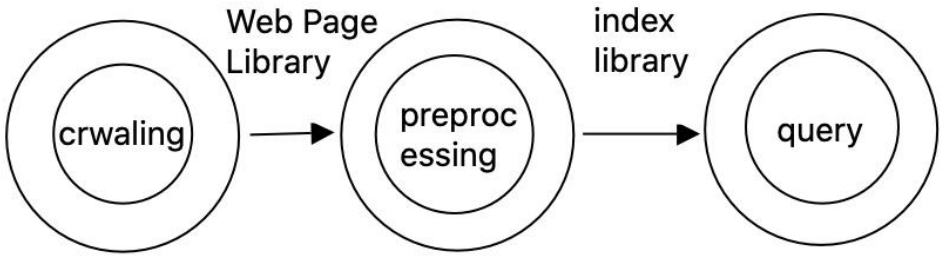
System Structure of Search Engine(Coo)

Abstract

This report includes three parts, and will explain step by step how to design and implement a search engine. In the first part, you will first learn the working principle of the search engine and understand its architecture, and then explain how to implement the first part of the search engine, the web crawler module, which completes the web page collection function. In the second part of the series, we will introduce the preprocessing module, that is, how to process the collected web pages, and organize, segmentation, and indexing are all in this part. In the third part of the series, the implementation of the information query service will be introduced, mainly the establishment of the query interface, the return of query results, and the implementation of snapshots.

The overall structure of Coo and the process of data transmission.

In fact, the three parts of the search engine are independent of each other, and the three parts work separately. The main relationship is reflected in the data obtained in the previous part. The relationship between the three is shown in the following figure:
Figure 1. Search engine three-stage workflow



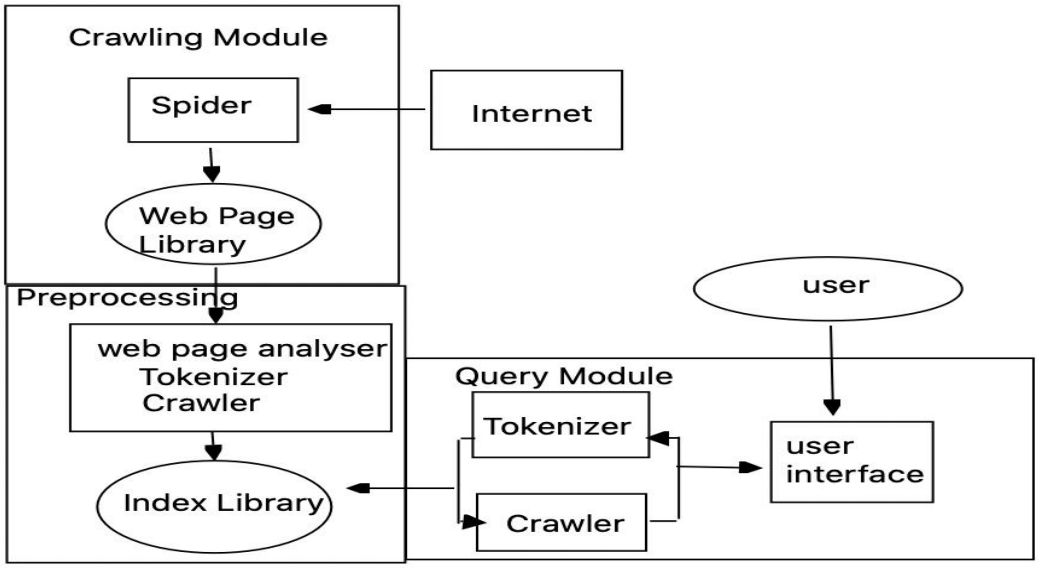
The top-down approach describes the search engine execution process:
The user submits the query word or phrase P through the browser, and the search engine returns a list of matching web page information (L) according to the user query;
The above process involves two questions, how to match the user's query and where does the web page information list come from, and according to what sort? The user's query P is cut into small phrases <p1, p2… pn> by the tokenizer. According to an inverted index maintained by the system, a certain word pi can be queried in which web pages have appeared, matching those pages where <p1, p2… pn> appears can be used as the initial result. Furthermore, the returned initial web page set can calculate the relevance of the query term to obtain the page rank, that is, Page Rank , You can get the final list of pages according to the ranking order of the pages;

Assuming that the formulas for the tokenizer and page ranking are both established, where does the inverted index and the original page set come from? In the introduction of the previous data flow of the original web page set, it can be known that the crawler spider crawls the web pages and saves them locally, and the inverted index, that is, the

phrase-to-page mapping table is based on the forward index. The latter is the web page-to-phrase mapping table obtained after analyzing the content of the web page and segmenting the content, and the inverted index can be obtained by inverting the positive index;

What does web page analyzer do? Because the original web page collected by the crawler contains a lot of information, such as html forms and some spam information such as advertisements, the web page analysis removes this information, and extracts the text information in it as subsequent basic data.

After the above analysis, we can get the overall structure of the search engine as follows:



The crawler crawls a large number of web pages from the Internet and stores them as the original web page library. Then the web page analyzer extracts the topic content from the web page and sends it to the tokenizer for word segmentation. The obtained results use the indexer to establish forward and backward indexes. The index database is obtained. When the user queries, the input query phrase is cut by the tokenizer and the query is performed in the index database by the searcher. The obtained result is returned to the user.

No matter the size of the search engine, its main structure is composed of these parts, and there is no big difference. The quality of the search engine is mainly determined by the internal implementation of each part.

With the above-mentioned overall understanding of search engines, let's learn the specific design and implementation of the crawler module in Coo.

The concrete implementation of Spider

Web collector Gather

The web page collector obtains the web page data corresponding to the URL through a URL key word. Its implementation mainly uses the URLConnection class in Java to open the network connection of the corresponding page of the URL, and then reads the data in the I / O stream. The buffer of data improves the efficiency of data

reading and the readLine () line reading function.

Web processing

The collected single web page needs to be processed in two different ways. One is to put it into the web page library as the raw data for subsequent processing; the other is to analyze the URL link and extract it into the URL pool to wait for the corresponding webpage. Web pages need to be saved in a certain format for batch processing of future data. The web page library consists of several records, each record contains a piece of web page data information, and the storage of records is added sequentially; The header is composed of several attributes, including: version number, date, IP address, data length, arranged according to the attribute name and attribute value, with a colon in the middle, and each attribute occupies a line; Data is web page data. It should be noted that the reason for adding the date of data collection is that since the content of many websites is dynamically changed, such as the homepage content of some large portals, this means that if it is not the web page data that is crawled that day, the data may be out of date Problem, so you need to add date information to identify it.

URL extraction is divided into two steps. The first step is URL identification. The second step is to organize the URL. The two steps are mainly because the links of some websites use relative paths. If they are not organized, errors will occur. URL recognition is mainly through regular expression matching. The process first sets a string as the matching string pattern, and then after compiling in the pattern, you can use the Matcher class to match the corresponding string. The implementation code is as follows:

According to String pattern1 = "(\\S)*(news.yahoo.com) [-A-Za-z0-9/]+ [-A-Za-z 0-9/_]";this regular expression can match the entire tag where the URL is located, so after the loop gets the entire tag, you need to further extract the real URL, we can intercept the content between the first two quotes in the tag to get this content. After that, we can get a preliminary URL collection that belongs to the web page.

Next, we perform the second step, the URL sorting, that is, filtering and integrating the URL collection in the entire page obtained previously.Since we can easily obtain the URL of the current web page. On the other hand, among the comprehensive URLs included in the page, there are some web pages such as video or image web pages that we do not want to crawl or are not important. Here we mainly perform a simple processing for these elements on the page by adding a boolean express like page.hasText(). For example, when the connection contains expressions such as "jpg", the priority of the link can be lowered, so that crawling of advertisement links can be avoided to a certain extent.

Dispatcher

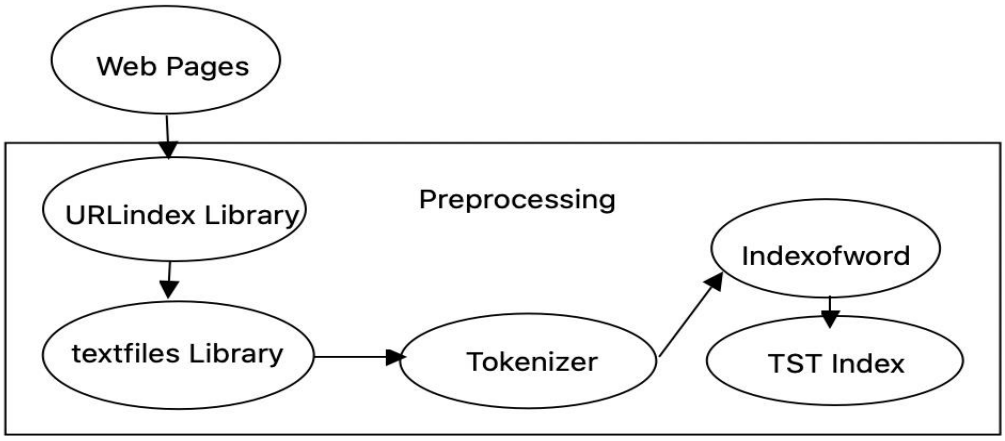
The allocator manages URLs, which is responsible for saving the URL pool and distributing new URLs after Gather gets a certain web page, and also avoids repeated collection of web pages. The allocator is coded in the singleton pattern in the design pattern and is responsible for providing Gather with the new URL. The singleton pattern is particularly important because it involves subsequent multithreading rewrites.

Duplicate collection refers to a web page that physically exists and is repeatedly

accessed by Gather without updating. This results in a waste of resources. The main reason is that there is no clear record of the URLs that have been accessed and cannot be discerned. Therefore, Dispatcher maintains two lists, "visited tables" and "unvisited tables". After the page corresponding to each URL is crawled, the URL is placed in the visited table, and the URL extracted from the page is placed in the unvisited table. When Gather requests the URL from Dispatcher, first verify whether the URL in the visited table. Spider launches multiple Gather threads.Now that there are hundreds of millions of web pages in the Internet, it is obviously inefficient for a single Gather to collect web pages, so we need to use a multi-threaded method to improve efficiency. Gather's function is to collect web pages. We can use the Spider class to start multiple Gather threads to achieve the purpose of multi-threading. code show as below:

After the thread is started, the web collector starts the operation of the job, and after one job is completed, it applies to the Dispatcher for the next job. Because of the multi-threaded Gather, in order to avoid thread unsafeness, it is necessary to perform exclusive access to the Dispatcher. The synchronized keyword is added to its function to achieve thread-safe access.

Preprocessing Moudle



In the first process, the index of the web page needs to be established first, so through the index, we can easily obtain the page information corresponding to a URL from the original web library. After that, we process the web page data. For a web page, we first need to extract the text information of the web page, then segment the text information, and then build indexes and inverted indexes based on the word segmentation. In this way, the pre-processing of the web page is all completed.

Due to the large amount of data, these indexed web page information needs a way to save, Coo uses an Excel database to save these information. That is we use “./Index/indexURL.csv”. The contents of the table are as follows: version, time, url, content, location. The URL is the URL corresponding to a record, because after the index database is established, we use the URL to determine the required webpage; Location represent the webpage library name (ie, the folder and relative position of all the web pages(save as txt format)). The amount of data on the web page is generally large. It is

not very practical to put the entire content of the web page into the database. Therefore, we put the MD5 summary of the web page content into the content property, which is equivalent to a check Code, in actual application, when we obtain a certain page information according to the URL, we can make the MD5 summary of the obtained page and then match the value in the content. If it is the same, the page is successfully obtained. If it is not the same, the page is explained there is some problem.

You may be a little confused about the MD5 digest algorithm. What is it? What's the use of this? Message Digest Algorithm MD5 is a hash function widely used in the field of computer security to provide message integrity protection. The typical application of MD5 is to generate a 128-bit binary message digest (Message-Digest), that is, a 32-bit hexadecimal number string, to prevent tampering. For us, for example, through MD5 calculation,the summary of the data of a web page is 000992914CFE6CD1 A959C31C076F49EA8.If we arbitrarily change the data in this page, after the calculation, the summary will change. We can consider the MD5 summary of the information as Information of the fingerprint information. Therefore, storing the summary can verify whether the web page information obtained afterwards is consistent with the original web page.

Text information extraction

Before the text information extraction, we first need a simple tool class, which can take out the content in the database and go to the original web page set to obtain the web page information. It uses the URL from the database library content of the web page corresponding to the URL are obtained in the database, and then the data content of the web page can be read according to content.

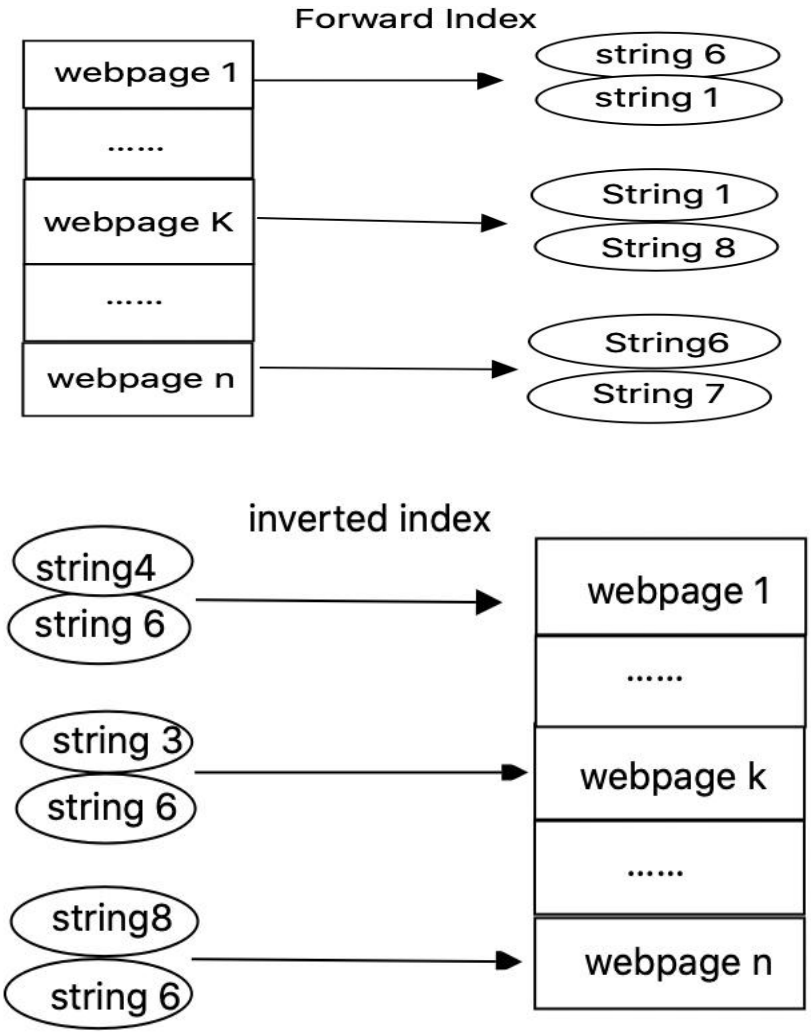
Text extraction

For the data of a single web page, we can proceed to the next step. The first thing to do is to extract the content of the text to remove the tag content in the web page. This step is mainly performed by regular expressions(Here we use Regex).

Refers to cutting a character sequence into individual words, so that the computer can automatically recognize. There are three main methods of word segmentation: the first is based on string matching, the second is based on semantic understanding, and the third is based on statistics. Since the second and third implementations require a lot of data to support them, we use a method based on string matching.

TST index

Here, we explain the last two steps of the preprocessing module, the establishment of the index and the establishment of the inverted index. With the segmentation results, we can obtain a positive index, that is, a certain web page and its corresponding segmentation results. As shown below:



At the beginning, we built an index web library, which can be used to directly locate the data corresponding to the URL in the original web library through the URL; and for the current forward index, we can get the web page by the URL of a web page Segmentation information. Obtaining a positive index does not seem to be of practical help to our upcoming query operation, because the query service obtains web page information through keywords, while the forward index cannot reversely check the web page information through the word segmentation results. In fact, the purpose of establishing a forward index is to establish an inverted index through a flip operation. The so-called inversion is the mapping method of the webpage-word segmentation result in the forward index. The inverted index corresponding to Figure 2 is shown in Figure 3 above.

For webpage i, get its segmentation list List;For each phrase in the List, check whether the phrase is included in the inverted index. If not, insert the phrase into the index of the inverted index and add the webpage i to its index value. If the inverted index already contains This phrase directly adds webpage i to its index value;
If there are still pages that have not been analyzed, go to 1; otherwise, end

The algorithm for establishing inverted index is not difficult to implement, mainly the choice of data structure. In Coo the forward index in hash map and use TST to do inverted index. The key of the forward index in the mapping is the string corresponding to the web page URL , While the inverted index uses the segmentation phrase, the value in the mapping, the former is a list of segmentations, and the latter is a string list of URLs. Here you can use an optimization, create two tables separately, and store the segmentation list and URL list according to the label. In this way, the value in the index can use the integer variable list to save space.