**WEB CRAWLER IN PYTHON**

With the advent of the big data era, the need for network information has skyrocketed. Many companies collect external data from the Internet for a variety of reasons, including analysing competition, summarizing news stories, tracking trends in specific markets, and gathering daily stock prices to build predictive models. As a result, web crawlers are becoming increasingly important. Web crawlers automatically browse or collect information from the Internet based on predefined rules.

Basic workflow of general web crawlers

The basic workflow of a general web crawler is as follows:

* Get the initial URL. The initial URL is an entry point for the web crawler, which links to the web page that needs to be crawled;
* While crawling the web page, we need to fetch the HTML content of the page, then parse it to get the URLs of all the pages linked to this page.
* Put these URLs into a queue;
* Loop through the queue, read the URLs from the queue one by one, for each URL, crawl the corresponding web page, then repeat the above crawling process;
* Check whether the stop condition is met. If the stop condition is not set, the crawler will keep crawling until it cannot get a new URL.

##### **1. FIRST, YOU NEED TO IMPORT THE LIBRARIES YOU NEED TO USE.**

import requests

import lxml

from bs4

import BeautifulSoup

##### **2. CREATE AND ACCESS URL**

Create a URL address that needs to be crawled, then create the header information, and then send a network request to wait for a response.

**url = "https://www.rottentomatoes.com/top/bestofrt/"**

**f = requests.get(url)**

When requesting access to the content of a webpage, sometimes you will find that a 403 error will appear. This is because the server has rejected your access. This is the anti-crawler setting used by the webpage to prevent malicious collection of information. At this time, you can access it by simulating the browser header information.

**url = "https://www.rottentomatoes.com/top/bestofrt/"**

**headers = {**

**'User-Agent': 'Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/63.0.3239.132 Safari/537.36 QIHU 360SE'**

**}**

**f = requests.get(url, headers = headers)**

##### **3. PARSE WEBPAGE**

Create a BeautifulSoup object and specify the parser as lxml.

**soup = BeautifulSoup(f.content,'lxml')**

##### **4. EXTRACT INFORMATION**

The BeautifulSoup library has three methods to find elements:

findall() :find all nodes

find() :find a single node

select() :finds according to the selector CSS Selector

We need to get the name and link of the top100 movies. We noticed that the name of the movie needed is under. After extracting the page content using BeautifulSoup, we can use the find method to extract the relevant information.

movies = soup.find('table',{'class':'table'}).find\_all('a')