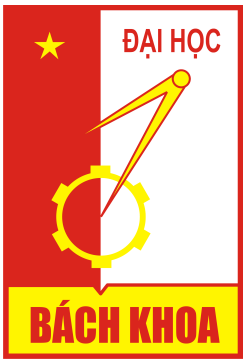
Graduation Research 1 Report

Web Crawler

Nguyen Ngoc Lam - 20162316
Instructor: Cao Tan Dung

XPath, CSS selector, Web crawler, Scrapy and Data cleaning



School of Information and Communication
Hanoi University of Science and Technology
Saturday 11th July, 2020

Contents

1 Theoretical Background **2**
- 1.1 XPath and CSS selector . . . 3
  - 1.1.1 XPath . . . 3
  - 1.1.2 CSS selector . . . 3
- 1.2 Data Crawling and Web Crawler . . . 4
  - 1.2.1 Data Crawling . . . 4
  - 1.2.2 Web Crawler . . . 4
  - 1.2.3 Scrapy . . . 5

2 Preparation **6**
- 2.1 Problem Understanding . . . 7
  - 2.1.1 Crawling Plan . . . 7
  - 2.1.2 Tools Used . . . 8

3 Web Crawler by Using Scrapy on Python **10**
- 3.1 Example problem: github page . . . 11
  - 3.1.1 Goals . . . 11
  - 3.1.2 Website classification . . . 11
  - 3.1.3 Result . . . 11
- 3.2 First website: travelling blog . . . 11
  - 3.2.1 Goals . . . 11
  - 3.2.2 Website classification . . . 11
  - 3.2.3 Result . . . 11
- 3.3 Second website: tripadvisor page . . . 11
  - 3.3.1 Goals . . . 11
  - 3.3.2 Website classification . . . 12
  - 3.3.3 Result . . . 12

Chapter 1

Theoretical Background

1.1 XPath and CSS selector

To crawl data from a website, we need to specify exactly where[1] in the website[2] that we need to extract information from. To do that, we can use one of these two tools

1.1.1 XPath

XPath or XML Path Language is a query language for selecting nodes from an XML document. The XPath language is based on a tree representation of the XML document. It provides the ability to navigate around the tree, selecting nodes by a variety of criteria.

1. Syntax:
    - the system is similar to the Unix file system
    - select node by *nodename*
    - select from the root node by /
    - select from everywhere in the document from the current node by //
    - select the current node by .
    - select the parent of the node by ..
    - select attribute by @
    - predicates are used to find a specific node or a node that contains a specific value and predicates are always embedded in square brackets [].
2. Usage:
    - XPath has been adopted by a number of XML processing libraries and tools
    - W3C has made it one of their official standard
    - It is also supported by most data crawling libraries like scrapy, apache nutch and cheerio

1.1.2 CSS selector

As CSS allow authors to not have to repeat the style for every single HTML tag but instead store them in a separated file, CSS deploy a system of selectors to declare which part of the markup a style applies to by matching tags and attributes in the markup itself. When crawl the data from a webpage, we can use the system of selectors to select the node that we need.

[1] the HTML tag
[2] the HTML document

1. Syntax:
    - select by class name by using $.class$
    - select by id by using $\#id$
    - select element by using $elementname$
    - select child node by $>$
    - select node by attribute using $[attribute = value]$
    - select the i^{th} child of a node by : $nth - child(i)$
2. Usage:
    - CSS selector is widely used in internet.
    - W3C has made it one of their official standard
    - Due to CSS popular, it is also supported by most data crawling libraries

1.2 Data Crawling and Web Crawler

1.2.1 Data Crawling

This is one of the most important steps on building a successful machine learning model. Data crawling helps us get the input data for the model as while as the data for training the model. A good dataset can make a big different in the training process later on.

1.2.2 Web Crawler

Web crawler is an Internet bot that systematically browses the World Wide Web, typically for the purpose of Web indexing[3]

Steps

1. A Web crawler starts with a list of URLs to visit, called the seeds
2. It then identifies all the hyperlinks in the pages and adds them to the list of URLs to visit, called the crawl frontier
3. URLs from the frontier are recursively visited according to a set of policies
4. information gathered from the crawler then can be stored at an archive is known as the repository

[3]also known as web spidering

Policies

The behavior of a Web crawler is the outcome of a combination of policies:

- A selection policy which states the pages to download as the internet is extremely huge and it is near impossible to crawl all the website
- A re-visit policy which states when to check for changes to the pages as most pages are changing frequently, like adding a new item to the list of restaurant.
- A politeness policy that states how to avoid overloading Web sites as most server can only handle a limited amount of load and the crawler is not the only one use the bandwidth
- A parallelization policy that states how to coordinate distributed web crawlers

Potential problems

- Unwanted security breaches if not careful
- Most sites has a measure to prevent overload its server. If our crawler is not "polite" enough we could be ban for calling to the server as it automatically blocks the IP address.
- Data can be old if you not update it.

Examples

- The core of most search engines like google or bing is a web crawler.
- There are also a lot of open-sourced crawler like frontera, seeks, apache nutch and php-crawler

1.2.3 Scrapy

"Scrapy is an open source and collaborative framework for extracting the data you need from websites. In a fast, simple, yet extensible way."[4] Scrapy project architecture is built around "spiders", which are self-contained crawlers that are given a set of instructions. Following the spirit of other don't repeat yourself frameworks, it makes it easier to build and scale large crawling projects by allowing developers to reuse their code. Scrapy also provides a web-crawling shell, which can be used by developers to test their assumptions on a site's behavior. Others components of a scrapy project are selectors, items, item loader, link extractor and a system of requests and responses.

[4] official website

Chapter 2

Preparation

2.1 Problem Understanding

Scope

In this problem, we will focus on english site about Vietnam tourism, where to stay, what to eat or what to do.

Goal

To build a database about tourism in Vietnam which will be used later as a data for a smart tourism application[1]

Expectation

To build a crawler that can smartly crawl data without human interference. Before when you want to crawl a website you need to know the structure of it, and because each website has a different structure, we need to specify where to get the information. This can be achieved by clustering the web pages into different categories based on their structure. However, as I was doing the crawler, the structure of the website might be similar due to the use of framework in the making of the web page, it is still a lot of differences, especially in the naming system and how the class was name. Since our selectors depend on at least the id or the class name to select the nodes, we cannot achieved it right now.

Actual Website used for the research

- https://github.com/vietanhdev?tab=repositories: to understand how scrapy works, how the selectors select the node and which selector is better to deal with which type of website[2]
- https://worldtravelfamily.com/vietnam-travel-blog/: a travel blog with a simple structure with everything inside a big div tag
- https://www.tripadvisor.com/Tourism-g293921-Vietnam-Vacations.html: its structure is clearly divided into node and child nodes

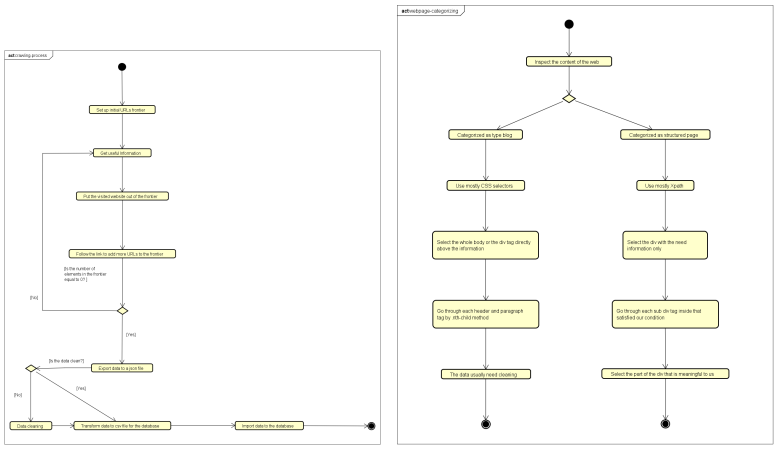
2.1.1 Crawling Plan

1. Divide the website into two large categories: blog-like website with all the tags containing are a sub tags of a big div tag or are a sub tag of the body tag; or well-define structure and using additional framework to write the website

[1] as chat bot or an recommendation system

[2] everything inside a big div tag (like some blogs) or its structure is clearly divided into node and child nodes

2. For each category, we build a general crawler to get the data that we want
3. Inspect each website to see exactly what do we need and write a selector to that
4. Generalize for each type of information we need, for example, the tripadvisor site can be divided into smaller categories like hotels, restaurant and thing to do since inside 1 website, there is usually 1 naming system for classes and id only
5. Skim through the result, get the information we really need
6. Transform it to a table-like file type like csv or excel or put them to a database



(a) Detail figure about the crawling process

(b) The process of webpage categorizing

Figure 2.1: Crawling plan

2.1.2 Tools Used

- Crawler library; Scrapy (written in python)
- Additional libraries[3]:
  - json: to read and write json file
  - BeautifulSoup: to pull data from html or xml document
  - pandas: data manipulation tools
- Database management system: postgresql[4]

[3] all of which are python libraries

[4] although it is not the fastest DBMS in the market but it works well with python

- Postgresql GUI tools: pgadmin

Chapter 3

Web Crawler by Using Scrapy on Python

3.1 Example problem: github page

3.1.1 Goals

The goal of this problem is to familiarize with how scrapy works

3.1.2 Website classification

Github website is a structured website thus it better for our crawler to use the xpath selector. But to familiarize with how scrapy works, I choose to use the css selector.

3.1.3 Result

The end result is a json file containing title of the repository, Url to that repository, and last modified date of all the public repository in Github. You can find out the result at

3.2 First website: travelling blog

3.2.1 Goals

The goal of this problem is to get the user experiences when travelling in Vietnam via their travelling blog.

3.2.2 Website classification

This is a blog website with simple structure. It only contains a big div for css to make the website more beautiful. The information is stored on p tags with the titles is on the h3 and h4 tags.

3.2.3 Result

The end result is a csv file containing title of the experiences, places that he or she pass in Vietnam, and and the paragraph about the experiences. You can find out the result at

3.3 Second website: tripadvisor page

3.3.1 Goals

The goal of this problem is to get the user rating of each places they visited when travelling in Vietnam via a community rating system on tripadvisor.

3.3.2 Website classification

This is a well-defined structure website. This type of website usually built using a open-source framework like node.js. It has multiple div tags and inside those tags there are a lot of other tag. Each location is also on a div tag of their own. The information (the rating) is stored on the attribute on an a tag inside each individual div tags.

3.3.3 Result

The result is a detail database on places and rating of that places on 6 famous tourist cities in Vietnam Hanoi, Ho Chi Minh city, Danang, Hue, Hoian, and Sapa. This information is stored on the csv with each line containing the name of the city, the type of the places, the name of the place and its rating. You can find out the result at