**Software Design Document**

**Crawling a website**

Date: 21st November,2022

Written By: Monisha Malla

**Introduction**

Web crawling is the process of indexing data on web pages by using a program or automated script. These automated scripts or programs are known by multiple names, including web crawler, spider, spider bot, and often shortened to crawler.

Web crawlers copy pages for processing by a search engine, which indexes the downloaded pages so that users can search more efficiently. The goal of a crawler is to learn what webpages are about. This enables users to retrieve any information on one or more pages when it’s needed.

**System Overview**

A web crawler provides the following features:

**Scalability:** The web is growing exponentially day by day. There are billions of web pages to be crawled and analyzed. Therefore, web crawling should be extremely efficient using parallelization.

**Robustness:** The internet is full of traps such as bad HTML, unresponsive servers, crashes, malicious links, etc. The crawler must be able to handle all these cases.

**Politeness:** The web crawler should not make too many requests to a website within a short time interval as it may unintentionally lead to DDoS on several websites.

**Extensibility:** The system should be adaptable and flexible to any changes that we might encounter in the future, like if we want to crawl images or music files.

**Manageability and Reconfigurability:** An appropriate interface is needed to monitor the crawl, including the crawler's speed, statistics about hosts and pages, and the sizes of the main data sets.

**Design Considerations**

Many factors need to be considered while evaluating the design considerations-

**Estimation of scale and constraints**

We should estimate the scale by taking many assumptions by discussing them with the interviewer.

Assuming 1 billion web pages to be crawled every month, we can estimate the average QPS to be: QPS (Query per Second) = 1,000,000,000 / 30 days / 24 hours / 3600 seconds 400 pages per second.

We can assume that the peak value of queries per second is 2 times the average, i.e., 800 pages per second. Now, assuming that, on average, a web page is 500 Kb in size, we can estimate storage required per month: 1-billion-page 500k = 500 TB storage per month.

Also, assuming data are stored for five years, 500 TB \* 12 months\* 5 years = 30 PB.

**Challenges faced**

* Non-Uniform Structures.
* Maintain Database Freshness.
* Bandwidth and Impact on Web Servers.
* Absence of Context.
* The Rise of Anti-Scraping Tools.

**Goals and Guidelines**

The web crawler or web spider, is a computer program that's used to search and automatically index website content and other information over the internet. These programs, or bots, are most commonly used to create entries for a search engine index.

**Requirements of the System**

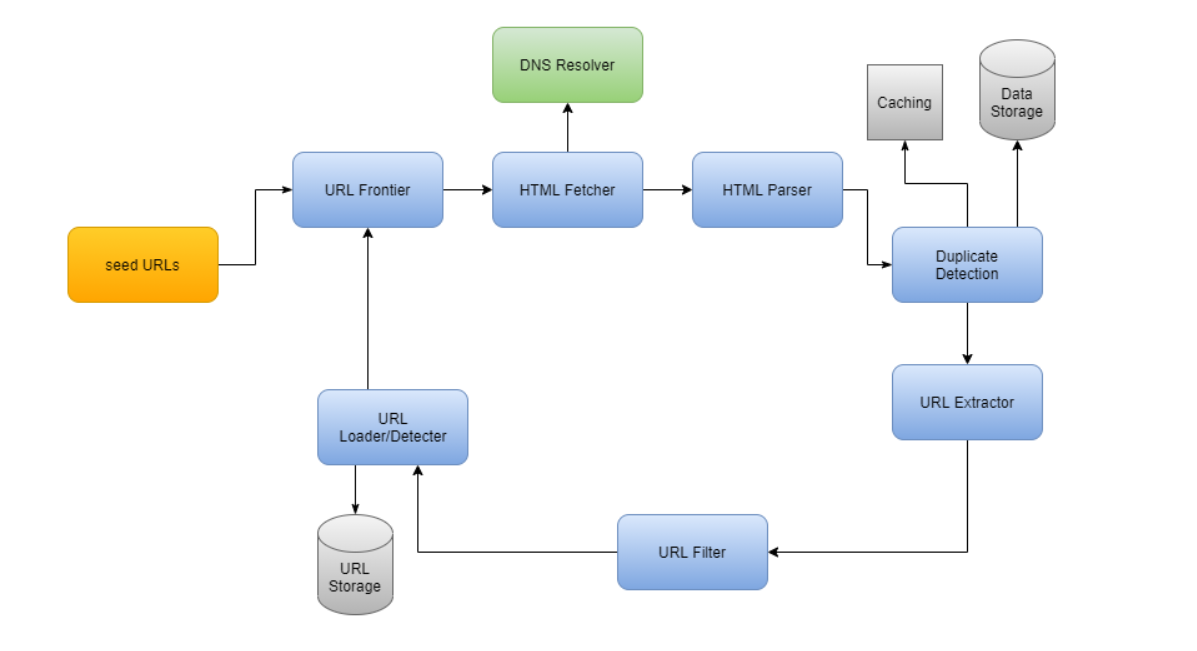
A simple web crawler should have the following functionalities-

* Given a set of URLs, visit the URL and store the page
* Now, extract the URLs in these web pages
* Append the new URLs extracted to the list to be visited
* Repeat the process

**Software Requirements**

* Python
* MySQL
* Visual Studio or Spyder or PyCharm
* Windows or Mac OS

**Detailed System Design**



Let us explore the individual components of the system-

**Seed URLs:** We need to feed seed URLs to the web crawler as a starting point for the crawl method. A good way to pick seed URLs is to use any website's domain name to crawl all web pages. To make our system more efficient, we need to be creative in choosing the URL to get started with to crawl as many web pages as possible.

The selection of it also depends on the use case. It can be chosen in many ways, such as geographical location, categories (such as entertainment, education, sports, food), content type, etc.

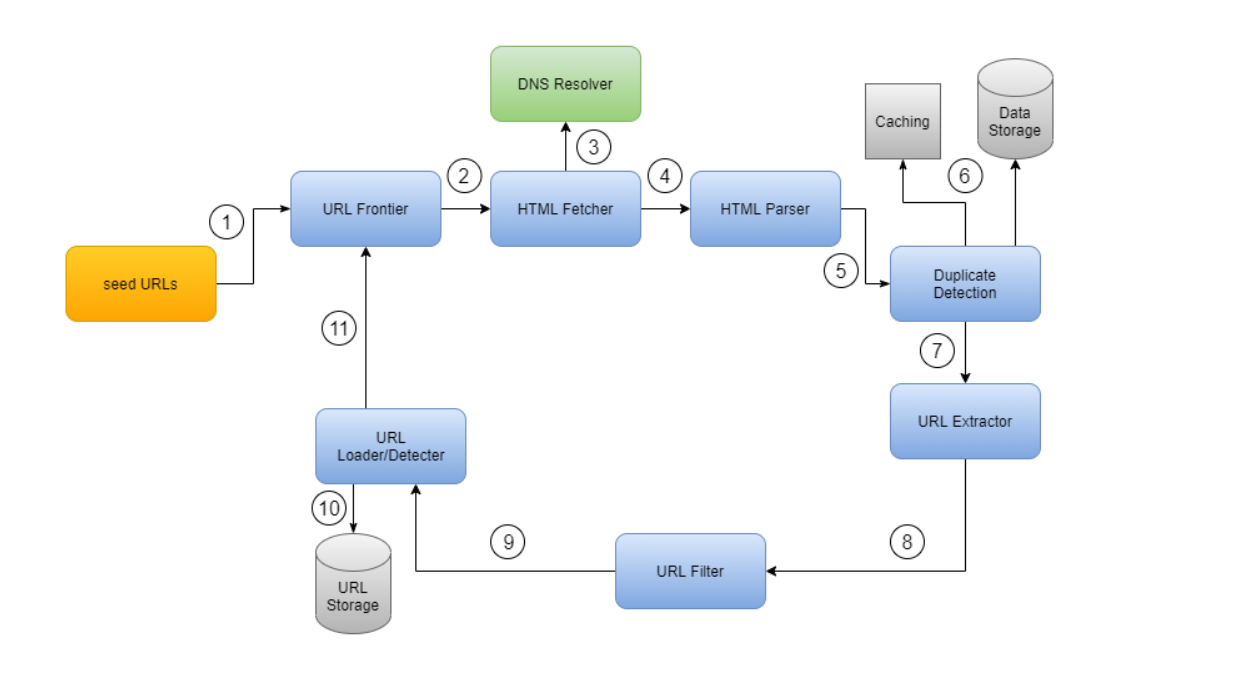
**URL Frontier:** The component that stores URLs to be downloaded is called the URL Frontier. We can crawl by performing a breadth-first traversal of the web, starting from the seed URLs. This is achieved by implementing the frontier as a FIFO.

**HTML Fetcher:** The purpose of an HTML fetcher is to download the web page corresponding to a given URL given by the URL Frontier using the appropriate network protocol like HTTP/HTTPS.

**DNS Resolver:** A URL needs to be translated into an IP address to download a web page. To get the corresponding IP address for the URL, the HTML Downloader calls the DNS Resolver.

**HTML Parser:** The downloaded web pages must be parsed, analyzed, and validated to protect the storage from any problem caused due to bad HTML, malware, etc.

**Workflow of Web Crawler System**



**Use cases of Web Crawler**

A crawler is used for many purposes:

**Search engine indexing:** One of the best use cases of web crawlers is in search engines. A crawler collects web pages to generate a local index for search engines. For example, Googlebot is the web crawler behind the Google search engine.

**Web archiving:** This is the compilation of web-based information to store data for future use. Many national libraries, for instance, run crawlers to archive web pages. The US Library of Congress and the EU web archive are notable examples.

**Web mining:** An incredible opportunity for data mining is created by the exponential growth of the web. Web mining allows the internet to discover valuable information.

**Web monitoring:** The crawlers help to monitor copyright and trademark violations over the Internet. Digimarc, for instance, uses crawlers to discover pirated activities and reports.

**Conclusion**

The evolution of these web crawlers is studied. A web crawler is a way for the search engines and other users to regularly ensure that their databases are up to date. Web crawlers are a central part of search engines, and details on their algorithms and architecture are kept as business secrets. When crawler designs are published, there is often an important lack of detail that prevents others from reproducing the work. There are also emerging concerns about “Search Engine Spamming”, which prevent major search engines from publishing their ranking algorithms.