Crawler Design

-Nisha Fathima T (19pt14)
-Varna Sureshbabu (19pt32)

Web Crawler

- Web crawler is a system for downloading, storing, and analyzing web pages.
- It performs the task of organizing web pages that allow users to easily find information. This is done by collecting a few web pages and following links to gather new content.
- Web crawlers are used for a variety of purposes, but they are commonly used as a key component of search engines which compile a collection of web pages, index them, and allow users to search the index for pages that match their queries.

Use cases of web crawler

- **Search engine indexing:**
  - Search engines uses web crawlers to collect web pages and generate a local index. For example, Google uses Googlebot web crawler.
- **Web archiving**:
  - We can use web crawlers for collecting web-based information and storing it for future use.
  - The US Library of Congress and the EU web archive often use crawlers for this purpose.
- **Web monitoring:**
  - Web crawlers can monitor the internet for copyright and trademark violations.
  - For example, Digimarc uses crawlers to identify and report pirated activities.

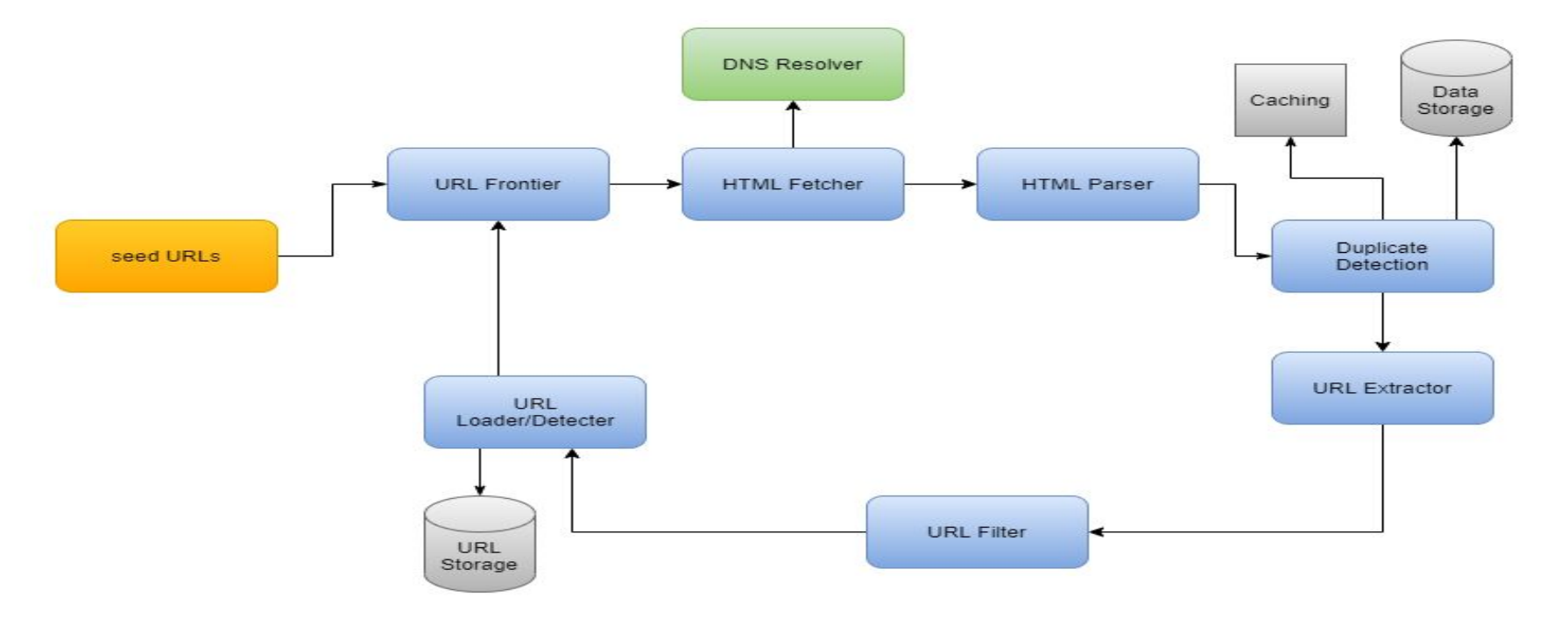
Requirements of the System

- A simple design of a web crawler should have following functionalities:
- Given a set of URLs, visit the URL and store the web page.
- Now, extract URLs in these web pages.
- Append new URLs extracted to the list of URLs to be visited.
- Repeat the process.

System Analytics and Characteristics

- System should be scalable because there are billions of web pages that need to be crawled and analyzed.
- Web crawler should be robust to handle various challenges like poorly formatted HTML, unresponsive servers, crashes, and malicious links.
- Web crawler should avoid making too many requests to a website within a short time interval because this can lead to DDoS attacks.
- System needs to be extensible so that it can be flexible to any future changes.
- For example, the ability to crawl images or music files may be required in the future.
- Web crawler should be manageable and reconfigurable i.e. it should have a good interface for monitoring crawl like statistics about hosts and pages, crawler speed, and sizes of the main data sets.

High Level Design



Individual Components

- **Seed URLs**
  - Initial web addresses or links that are used as starting points in web crawling
  - They are the first URLs that the program visits to initiate data collection process
  - Choosing the right seed URL can impact the number of web pages crawled
  - Depends on various factor like
    - Geographical location
    - Categories like education, entertainment, sports, food etc.

Individual Components

- **URL Frontier**
  - also known as a URL queue or URL frontier queue
  - data structure used in web crawling
  - to manage and prioritize the URLs that a web crawler or scraper needs to visit and process.
- **HTML Fetcher**
  - Downloads webpages corresponding to the URL given by the URL Frontier
  - It retrieves actual web page content that needs to be analysed and stored

Individual Components

- **DNS Resolver**
  - Before a web page can be downloaded, URL must be translated to IP address
  - HTML fetcher will initiate download process by calling DNS Resolver
  - After conversion, it is used to access the web page
- **HTML Parser**
  - Done to ensure integrity of the data to be stored
  - Parser will check for poorly formatted HTML or malware

Individual Components

- **Duplicate Detection**
  - Storing duplicate content multiple times can lead to inefficiencies and slowdown in storage system
  - To overcome that, MD5 hashing can be used
- **Data Storage**
  - They need to be stored in a storage system
  - If we want to use the content for offline analysis, we can compress and store data with a low-cost cloud storage provider

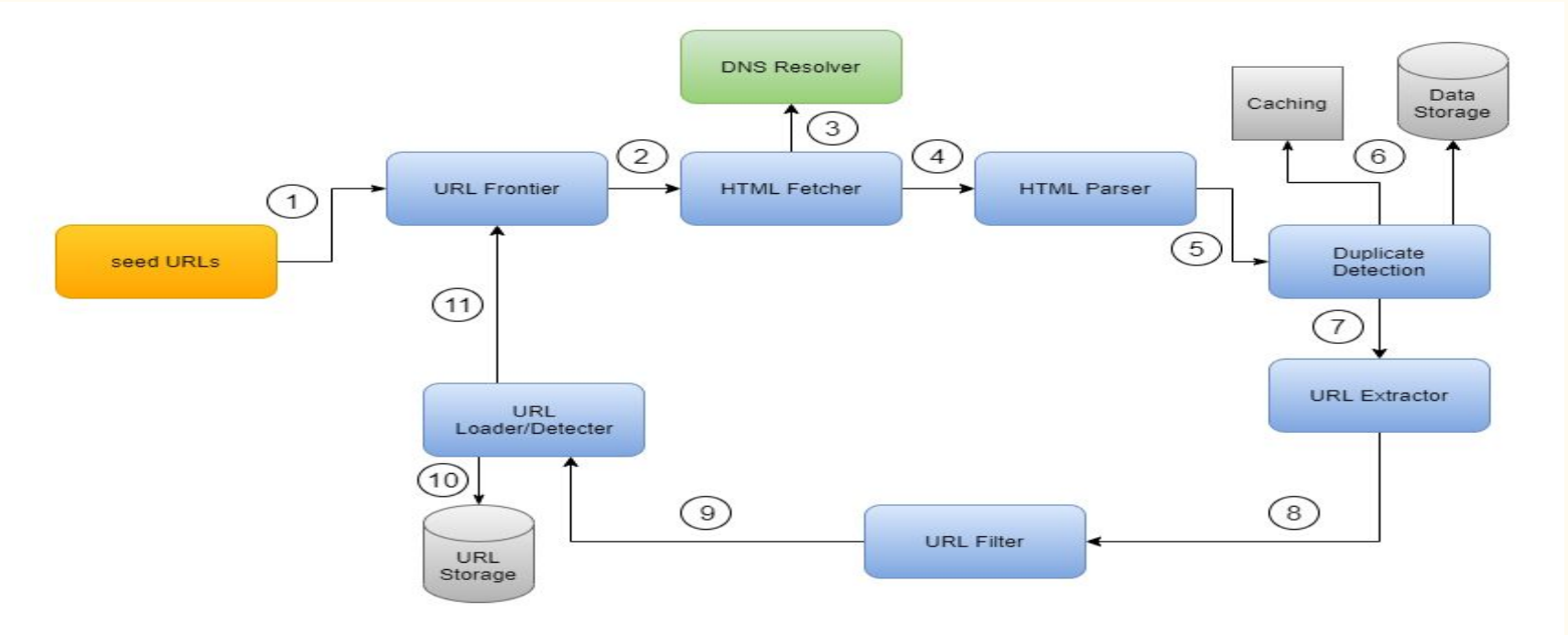
Individual Components

- **Caching**
  - To improve the efficiency of web crawler
  - To store recently processed URLs
- **URL Extractor**
  - Extract links from HTML pages
  - Enables system to discover new content

Individual Components

- **URL Filter**
  - Filter out unwanted content types, faulty links, and URLs from unsafe sites
  - This ensure system only collects high quality relevant content
- **URL Detector**
  - filter out URLs that are already visited
  - Bloom filters and hash tables can be used
- **URL Storage**
  - Keep track of visited URLs

Workflow of web crawler system



Steps

- At start, worker thread sends seed URLs to the URL Frontier
- HTML Fetcher module - fetch URLs from the URL Frontier
- HTML Fetcher calls DNS resolver
- HTML Parser parses the HTML page and analyzes content of the page
- The content is validated and then passed to the duplicate detection component to check for duplicity
- The duplicate detection component checks the cache, and if the content is not found there, it checks the data storage to see if the content is already stored there.

Steps

- If content is not in the storage, web page is sent to the Link Extractor, which extracts any outgoing links from the page.
- The extracted URLs are passed to the URL filter, which filters out unwanted URLs such as file extensions of no interest or blacklisted sites.
- After links are filtered, they are passed to the URL Detector component.
- URL Detector checks if a URL has already been processed and stored. If it has, no further action is needed. If URL has not been processed before, it is added to the URL Frontier to be crawled in a future work cycle.

Thank you