Large Scale Web Crawling and Distributed Search Engines: Techniques, Challenges, Current Trends, and Future Prospects

Asadullah Al Galib, Md Humaion Kabir Mehedi, Ehsanur Rahman Rhythm, Annajiat Alim Rasel
Department of Computer Science and Engineering
BRAC University
Dhaka, Bangladesh
{asadullah.al.galib, humaion.kabir.mehedi, ehsanur.rahman.rhythm}@g.bracu.ac.bd, annajiat@bracu.ac.bd

***Abstract*—The heart of any substantial search engine is a crawler. A crawler is simply a program that collects web pages by following links from one web page to the next. Due to our complete dependence on search engines for finding information and insights into every aspect of human endeavors, from finding cat videos to the deep mysteries of the universe, we tend to overlook the enormous complexities of today's search engines powered by the web crawlers to index and aggregate everything found on the internet. The sheer scale and technological innovation that enabled the vast body of knowledge on the internet to be indexed and easily accessible upon queries is constantly evolving. In this paper, we will look at the current state of the massive apparatus of crawling the internet, specifically focusing on deep web crawling given the explosion of information behind an interface that cannot be simply extracted from raw text. We will also explore the feasibility and future of distributed search engines and the way forward of finding information in the age of large language models like ChatGPT or Bard. Finally, we will present the design of an asynchronous crawler that can be used to extract information from any specific domain into a structural format.**

***Index Terms*—web crawling, crawler, distributed systems, search engines, deep web crawling, common crawl, asynchronous crawler**

I. Introduction

The technology behind web crawling has come a long way since Google was first introduced in the late 1990s. The initial design of Google was a centralized one that took into consideration the rate of growth of the internet and various scalability issues. According to the original PageRank paper [1], Google employed a distributed web crawler architecture to download content from hundreds of millions of web pages. At that time the internet could be considered to be in its nascent stage compared to today's explosion of data. The technology behind web crawlers grew ever more complex to accommodate billions of websites that exist today [2], in hundreds of languages. Everything has changed from fairly simple HTML web pages to complex web applications and animated sites that provide a gamified experience to users of the internet. The underlying crawlers, whether collecting raw text from web pages for any well-known and established search engines or aggregating structured information for domain-specific tasks, now have to deal with the enormous variety of the web in terms of site structure, types of data, and many more.

Even though crawling and related technologies have gone through tremendous innovations in the past decade, the ever-expanding and fluid nature of the internet means crawling tools and techniques need to stay on top of the current trends of the web to collect and aggregate information. The regular web, part of the internet that is accessible through various search engines is what most of us are familiar with in our day-to-day life. But due to the sudden explosion of various web technologies, most of the data on the internet now sits behind various search forms that can only be accessed using search queries. For crawlers, this adds another layer of complexities where simple link traversing from web pages' extracted content is no longer enough to meet the information demand of current internet users. Along with the current technological advancements in the field of large-scale web crawling, we will also focus on deep web crawling where crawlers need to interact with various web forms to access information.

We will look at how distributed system concepts are utilized in operating a massive search engine where major components of a search engine, namely crawling, indexing, sorting, efficiently storing, and finally the ranking of search results are distributed and scaled across multiple nodes and how all of these components interact together to transform raw web page content to actionable insights for the users. We will also take a look at the concepts of decentralized peer-to-peer search engines and meta-search engines. We will dive deep into Common Crawl, an open-source crawler that provides public copies of the entire web for research and analysis. In order to take into consideration of emerging, prominent, and disruptive AI conversational agents like ChatGPT and Bard and understand their implications on how people search for information on the web, we will explore how these agents can upend the status quo of current search engines. Finally, we are proposing a new asynchronous web crawling technique that can be employed by small to medium organizations for domain-specific crawling needs that require data to be extracted from web pages with similar content architectures and stored in specific formats. This crawler can be used to crawl certain websites either once or at regular intervals to serve the specific data needs of any organization without relying on others.

The *Introduction* part describes the motivation and overall targets of what the paper will accomplish. In the *Related Work* section, we will explore previous survey works related to web crawling techniques and various architectures of distributed search engines. In the next section, *Crawling*, we will take a detailed look at various crawling techniques currently in use as well as different scaling issues that crawlers need to deal with. In the *Distributed Search Engine* section, we will go through various distributed system concepts used to build and operate a search engine. We will explore various open-source search engines such as Apache Lucene, ElasticSearch, and others. We will also touch upon decentralized or P2P search engines along with meta-search engines. In the section, *AI Conversational Agents*, we will discuss the implications of ChatGPT and other conversational AI agents for the domain of search engines and how we acquire information from the web. The section, *Domain-Specific Asynchronous Crawler* proposes a new asynchronous crawler suitable for crawling sites of any domain that share a common architecture. Finally, we will conclude with the utility and prospects of crawling in the age of AI and generated content.

II. Related Work

Related Work

III. Crawling

Crawling

A. Techniques

B. Scaling

C. Common Crawl

D. Challenges & Prospects

IV. Distributed Search Engine

Distributed Search Engine

A. Architectures

B. Open-source Search Engines

C. Decentralized, P2P Search Engines

D. Meta-Search Engines

E. Challenges & Prospects

V. AI Conversational Agents

AI Conversational Agents

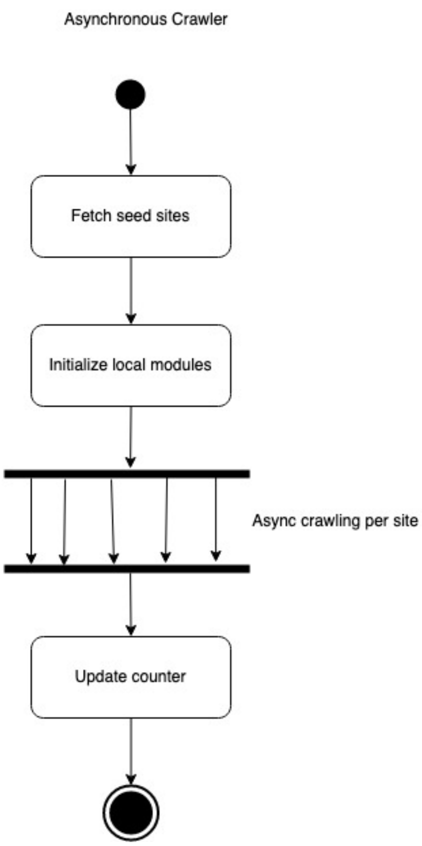

Fig. 1. Asynchronous Crawler - Orchestrator

A. Idea

B. Emerging Tools

C. ChatGPT

D. Bard

E. Challenges to Traditional Search Engines

F. Societal & Ethical Impacts

G. Future Prospects

VI. Domain-Specific Asynchronous Crawler

Domain-Specific Asynchronous Crawler

A. Architecture

Figures 1, 2, 3, and 4 show different stages of the asynchronous crawler architecture, namely the orchestrator, single site handler, single page handler, and single item handler.

B. Usage & Scopes

VII. Conclusion

Conclusion

References

[1] lastname, f., lastname f. (). test. DOI: https://doi.org/10.1016/S0169-7552(98)00110-X

[2] https://www.statista.com/chart/19058/number-of-websites-online/

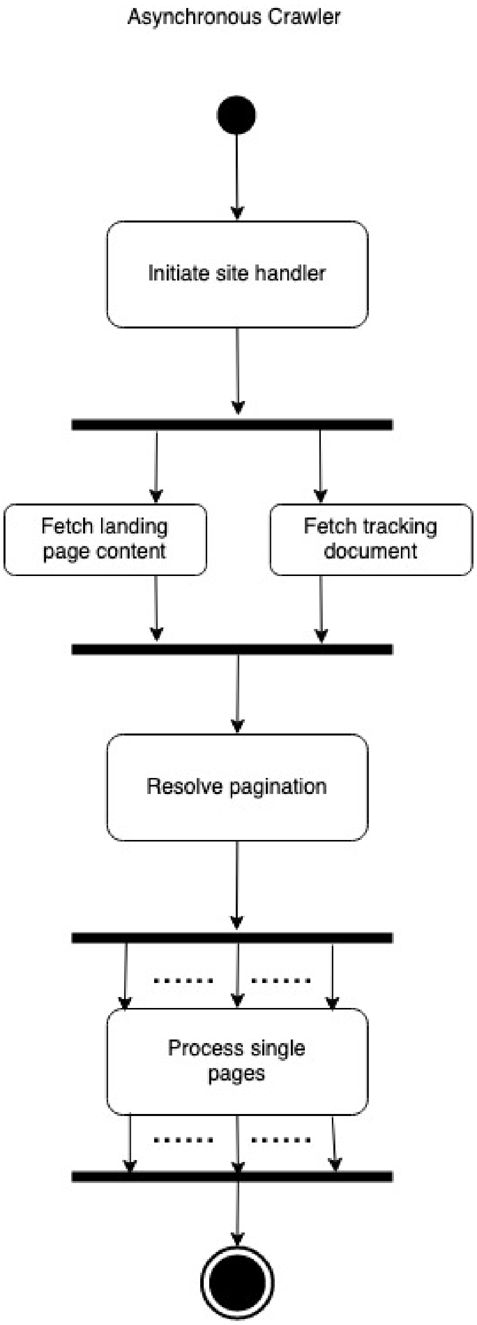

Fig. 2. Asynchronous Crawler - Single Site Handler

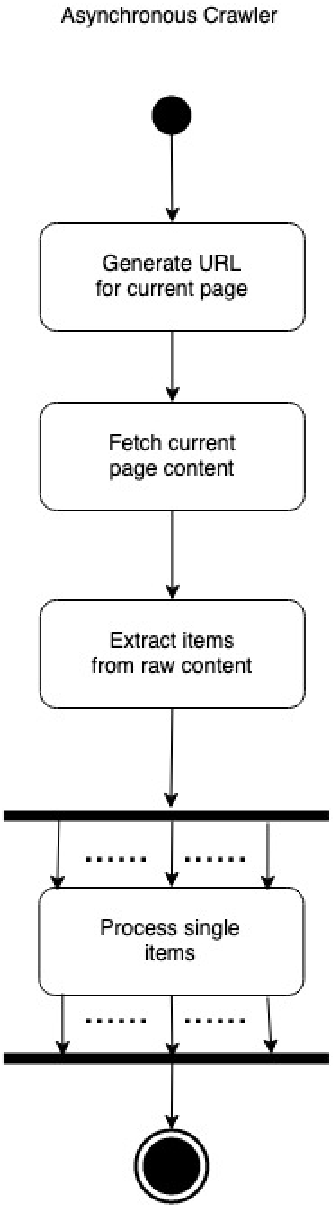
Asynchronous Crawler
Generate URL for current page
Fetch current page content
Extract items from raw content
Process single items
......
......
......
......

Fig. 3. Asynchronous Crawler - Single Page Handler

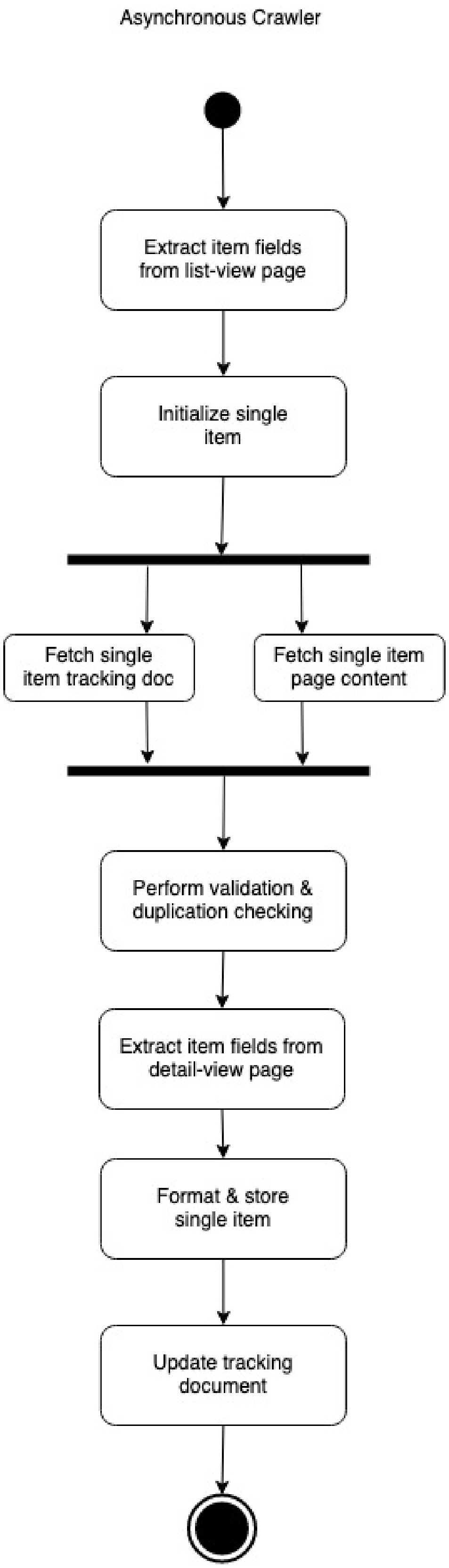
Asynchronous Crawler
Extract item fields from list-view page
Initialize single item
Fetch single item tracking doc
Fetch single item page content
Perform validation & duplication checking
Extract item fields from detail-view page
Format & store single item
Update tracking document

Fig. 4. Asynchronous Crawler - Single Item Handler