An Incremental and Parallel Aerial Disaster Data Crawler

Andrew Hin Chong Yau*

Abstract
The state of the climate right now is causing the weather to become progressively volatile to the extent certain regions can no longer handle the intensity and frequency of natural disasters. As prevention is growing less viable, the knowledge to help mitigate disasters is an asset. The web is now the central wealth information in the world but most of it is disorganised and unusable. This paper analyses web crawling methods to organise and coalesce aerial disaster drone images to be practically used during disasters and potentially be used to gather other sets of data in the future. The product of this paper will hopefully be a methodology for a parallel and incremental web crawler.

Contents

I certify that all material in this dissertation which is not my own work has been identified.

Signature: Andrew Hin Chong Yau

* ahcy202@exeter.ac.uk

1. Introduction

The web is composed of nearly 2 billion websites[1, 2] with Google indexing 30-50 billion individual web pages from these sites [2]. In addition the web is growing at an exponential rate doubling every 9-12 months and the changing rate of web pages with small changes is approximately 40% weekly, with a change rate of 7% weekly for web pages changed by a third or more [3]. This indicates that an extensive amount of useful data spanning almost every field in the modern world, is present and accumulating.

However, the potential for the majority of this data to benefit society is left untapped, as it is mostly unorganised and dispersed (data is supplied by individuals or groups) in websites made by different parties. The best way to begin coalescing this quantity of data is using specialised web crawlers that search for websites throughout the web with data related to a topic or field, and subsequently downloading it utilising a specialised web scraper. The data can then be organised and used accordingly. This paper will survey different web crawler and web scraper implementation methods, pertaining to incremental and parallel web crawler types for application towards disaster relief management.

According to the Centre for Research on the Epidemiology of Disasters (CRED) [4], 7,348 natural disasters (natural hazards resulting in significant deaths, homelessness and economic loss etc.) were recorded globally by the Emergency Events Database (EM-DAT) spanning 2000-2019, whilst between 1980-1999, 4,212 natural disasters were recorded. Therefore, indicating a 75% increase in natural disasters within 20 years as a result of a gradual 3°C temperature increase of the global climate. In conjunction, the Global Assessment Report on Disaster Risk Reduction's analysis of the data [5] has projected a linear increase in disasters over time, predicting an annual increase from approximately 400 in 2015 to 560 by 2030 which is a 40% increase.

CRED [4] has concluded that the increased disaster frequency may render climate change adaptation and current national and local strategies for disaster risk reduction obsolete in many countries. Therefore, the bottleneck in disaster prevention generates a need to optimise humanitarian and disaster relief actions in preparation to mitigate the effects of the inevitable disasters when they occur.

This project aims to create a web crawler that efficiently gathers raw aerial drone data of disaster sites available within the world wide web, specifically supporting two facets of humanitarian and disaster relief management fields[6]:

- Disaster Preparation: Gathered drone images can be utilised as data sets to train AI specialised in enhancing real-time disaster support operations. Examples include AI that can identify where victims are most like to be located, locations which are still risk factors or accessibility of different forms of transport, allowing on-site operators to make the most informed responses possible.

- Disaster Response: Up-to-date data is continuously and swiftly gathered due to parallel and incremental features, allowing aerial images taken by 3rd parties of an on-going disaster to be acquired and utilised by on-site operators as soon as possible, dramatically increasing the data coverage of the disaster area from different viewpoints.

2. Literature Review

A web crawler [7] at it's core, is an automated web object (page) retrieval system applied to the web which is effectively structured as a graph consisting of nodes (web pages) and edges (URLs). The crawler downloads the web pages by following the hyperlinks found in individual pages, starting with a few seed URLs (seed URL queue), and then reaching out to the other URLs (added to queue) extracted from the seed URLS. The page fetching and subsequent hyperlink extraction processes are essentially expanding a node in graph search. This entire process is repeated until the crawler decides to stop, or specified end conditions are met. Different web crawlers are then defined by the implementation methodologies [7, 8] used, such as what web traversal algorithm is used, what technology is used or how the URL queue is processed.

2.1. Efficient Web Crawling Algorithms

The Breadth First Search Algorithm[9] starts at the root URL and searches all the neighboring URLs at the same level, before traversing the next level in the tree. This algorithm is suitable for situations

where the objective is found on the shallower parts in a deeper tree. Performance drastically drops when the branches are abundant in a game tree. The Depth First Search Algorithm starts at the root URL and traverses depth through the entire tree using the backtracking principle to ensure every leaf of the tree is accessed. It's well suited for gathering all the data within individual small domains. This is limited by trees with a large branches as the algorithm may end up in an infinite loop, however this may be minimised by setting a maximum depth that should be traversed. The Page Rank Algorithm downloads the web pages based on page rank. This ranking may be determined according to a URL's popularity etc. Important pages can be extracted within a short time period without any severe detriments of limitations. The crawling of large sites with a large number of pending pages first may cause significant pages existing in small websites to only be crawled after an bad amount of time.

2.2. Incremental Methodology

An incremental crawler [7] continuously crawls the web, with a priority on maintaining the freshness of the local collection of URLS and improving the quality of the local collection which is revisited periodically according to a calculated priority. During its continuous crawl, low quality pages in the local collection may be purged to make space for new pages which may contain higher quality pages.

Generally, an incremental crawler is split into three main modules (Crawl Module, Ranking Module and Update Module). All URLs records all discovered URLs, and Coll URLs records the URLs that are/will be in the Collection. Coll URLs contains all the URLs which are already crawled and is implemented as a priority-queue in which URLs are kept according to their priority score. The Ranking Module chooses URLS to be added to Coll URLs after considering All URLs and the Collection to make the refinement decision. It must also schedule a replacement task when a page not present in Coll URLs, is given a higher rank than a page within Coll URLs. This new URL is placed in Coll URLs with highest priority, so that the Update Module can crawl the page immediately.

The Update Module maintains the Collection freshness by using the page's estimated change frequency and its importance as factors to calculate where the URL is placed in the Queue. URLs closer to the head of the queue have a higher refresh rate. Since the crawling policy of incremental crawlers are best first search, the majority of the web is left untouched. Therefore, the crawler falls into a circumstance where there may be a multitude of web pages the Ranking Module would rank high in importance throughout the web, that are never found which is sub optimal. Other limitations include a lower average relevancy of results to a specified topic in comparison to specialised crawlers, and frequent updates to it's target web pages will cause it to suffer from a higher network (bandwidth) load in comparison to other crawler types. Countering these issues should be done by creating a hybrid web crawler of different types.

2.2.1. Specialised Crawler Implementations

Shijia Xi[10] predicts recently modified pages and newly added pages with no actual crawling, reducing crawling overhead. Their implementation revolves around identifying feasible and stable ”structure patterns” (constant structures such as UI) within certain websites using a hybrid clustering method of K-means combined with agglomerative hierarchical clustering, to be processed by an unsupervised algorithm to generate the website's template. This template should be capable of enhancing the ease and precision of the web scraper as the locations of new data can be recognised and directly extracted.

The modified probability prediction of pages is initially calculated using the traditional method (gathering historical data and calculating the modification frequency) and applying the number of times the page has been updated to the Poisson Distribution, since it is approximately followed by the modifying process of individual web pages [10, 11]. These probabilities contribute to each pages' corresponding structure pattern's modified probability as statistically, each pattern has a decently stable modified probability.

The modified probability of initial and new pages then take into consideration how long a page has not been modified (according to the number of other crawling tasks executed), the structure pattern's modifying probability and it's transitive modifying probability (direct sub-pages of a certain page inherit the parent's modified frequency). Pages with a higher modified probability have a higher update priority.

This methodology has been shown to possess a satisfactory performance on small-medium sizes websites but there is a linear decrease in coverage rate as the scale of a covered website increases, indicating that an increasing quantity of page modifications will be missed as they occur. Overall, it has been successful on multiple fronts such as bandwidth saving by a large margin and increasing data extraction efficiency. This method may also have a high versatility due to the fact that it focuses on predictions and affects data extraction, which may theoretically grant an increase in performance when combined with methods such page ranking or URL pattern etc.

Zejian Shi[12] has implemented a specialised web crawler for crawling news using a Bloom filter in order to identify repeating URLs discovered during execution to prevent a duplicate URL from being added to the list of all URLs and being reprocessed efficiently. The Bloom filter [13, 14] is a binary long series mapping function which uses 1 or more independent and uniformly distributed hash functions applied on a seed. The web URL in this case and all bits in the Bloom filter hash map are initially 0. The set of random numbers generated by the hash functions are then operated on by the modulus of the length of the Bloom filter. The resulting integers are then mapped to their corresponding indexes in the Bloom filter and changed to one. The sets of hashed numbers act as an information fingerprint for every individual URL which becomes more unique with increasing quantities of hash functions used. The possibility of a web page being a duplicate can be identified, if all of the values of a number set's indexes in the Bloom filter is 1 which would then require cross checking the URL with the database as the exact same hash set from the same web URL may have been generated. The existence of a single 0 indicates that a web page has not been processed previously.

This limits the maximum number of optimal URL identifications to the length of the Bloom filter subtracted by the number of hash functions used (maximum quantity of 0s in the Bloom filter that can be changed). Zejian Shi[12]'s implementation of this only had a Bloom filter length of 625 bits which is quite insignificant for web crawlers due to the size of the web, supplying the collection with approximately 500 URLs. However, as the core implementation is incremental these 500 URLs should be enough to rank and constantly update without searching for any new URLS, leaving the Bloom filter obsolete. Overall, using a Bloom filter presents no detriments but it provides a simple efficiency benefit despite being somewhat limited.

Zejian Shi[12] and Shijia Xi's implementations[10] contrast as Shijia Xi has a general implementation which can be applied to all websites, in addition to suppling versatility and a far more substantial increase in performance than Zejian Shi[12] despite the higher complexity. Meanwhile, Zejian Shi's implementation[12] is specialised towards news web pages which would benefit from generalisation and has a low complexity. The fact that these features are applied onto differing modules, indicates that combining the two methods may be possible.

2.2.2. Search Engine Implementations

Hadrien Bullot[11] has introduced a data-mining approach centred around the actual practical usage of search engine web crawlers. URL usage is monitored by recording user data such as the URLs clicks and the query or category searches in the search engine's database within a specified time frame. This ensures that URLs that are more likely to be accessed by users will be discovered first by the web crawler. This reduces the 'embarrassment metric' optimisation mathematical model which refers to the potential of the queried information found being irrelevant to the topic. However, this should inversely generate an issue with bias towards the majority of users, leaving outliers within the user-base such as topic specialists and professionals where their web pages of interest are less likely to be updated and found. Furthermore new web pages that may be of interest would not be discovered and potentially render the information obsolete when discovered.

An update score module is created to calculate how the data contributes to the ranking score of web page and updates all of the URLs in the collection. The frequency this is updated at is highly dependent on numerous factors including the size of the URL collection and available resources, due to the fact that low frequency updates have a high risk of overloading the database, along with being incredibly inefficient and unviable. In addition, a high update frequency such as 12 hours may prevent important data such as news or announcements to be inaccessible within a suitable time frame.

Wei Liu[15] has proposed a sample-guided approach, using a query related graph, transforming the incremental crawling task to a graph traversal process (web is essentially a graph). The web pages

of a small quantity of random samples within the collection are updated and the standard deviation from their historical versions are calculated. Keywords from the sample with the highest standard deviations are subsequently generated and queried to the query-related graph model, expanding the nodes of related URLs. The random sampling is executed with a random walking approach where the number of traversals between each point in the graph is random and probabilistic rejection based techniques applied to the samples, further enhancing them. This would result in an increase in efficiency in terms of speed and bandwidth as web pages that are likely to have less modifications are not searched unnecessarily.

A query related graph is an undirected graph where every record has a unique vertex, and at least one undirected edge record connected to it where both records are members or results of a specific query. The graph complexity of a database is determined by the query capability (number of attributes and their range of values) of the query interface and number of records in the database. Therefore, a graph with a high attribute quantity and value range can return more results with a single query since every vertex possesses more neighbours that are likely to be fresh.

Experiments run over 4 different databases consisting of 2 sets (job and research) of 2 databases and 3 queryable attributes resulted in the final cumulative coverage rate of all of the simulations exceeding 96%, the incremental coverage rate (percentage of new pages) of job database runs exceeding 80% and the research database exceeding 64%. These results display that the crawler has a very reliable performance and the majority of new data pages can be acquired.

Hadrien Bullot[11] and Wei Liu's implementations[15] are web search approaches differing through the target data they seek to acquire, as Bullot focuses on user popularity and experience but may be limited by temporal or resource management limitations, whilst Liu focuses solely on the acquisition of as many fresh web pages as possible but is far more reliable with less overhead as there are no clear external dynamic factors (fluctuating web page popularity) influencing the ranking of different topics that require updating or consideration.

Conceptually, fragments of both methods may be used in conjunction where analysing the data gathered of user queries as they may show trends in popular topics during different times of the year, particularly different seasons [16] as typically the weather conditions such as evaporation rates or water levels vary in different seasons which would cause different natural disasters to become more prevalent at different seasons. Therefore, if a trend is found it may be beneficial to regularly increase the importance of trending disasters where the query related graph only contains queries associated with the corresponding natural disasters.

TABLE I. Comparison of Incremental Crawler Implementations

Implementaion	Shijia Xi[10]	Zejian Shi[12]	Hadrien Bullot[11]	Wei Liu[15]
Target Module	Update Module	Crawl Module	Ranking Module	Ranking Module
Application	Specialised Crawling	Specialised Crawling	Web Search	Web Search
Optimisation Approach	Page refresh rates and reduce web scraping overhead by ignoring ”structure patterns”	Rapid duplicate URL identification during execution enabling process skip and collection URL list exclusion	Rank pages according to actual page access rate by users.	Rank pages according to a query generated by identifying keywords from random websites and crawling the web pages in it's collection corresponding to the key words

2.3. Parallel Methodology

The exponential growth of the web causes crawling it's entirety a nigh impossible feat with a single instance of a web crawler. Parallelisation [7] is the methodology enabling the possibility of covering and managing the entire web at a reasonable rate and is it's most prominent benefit (Scalability). The

core of parallelisation lies in running multiple web crawler processes simultaneously with each process performing the simple operations of a single crawler process, such as downloading a specified page from the web and storing them in the local collection.

Other benefits include network load reduction and dispersion as dividing the crawling locations of each crawler process reduces the network load on individual websites/areas, in addition dispersing the processes themselves to geographically distant locations to download pages enables large scale crawls on geographically adjacent pages that would be impossible over a single network. Parallel crawlers may either be connected over a LAN (Intra-site parallel crawler) or distant geographically (Intra-site parallel distributed crawler)

There are several limitations on parallel crawlers such as the high volatility of data as certain web pages are frequently modified causing the the URL to possess multiple variations, generally no inter process communication redundant and the unstructured nature of gathered data.

Tithi Dhar [17] proposed an intra-site parallel crawler that employs multi-threaded server architecture with client crawlers. Redundant data is avoided as this crawler is run in sessions where downloaded pages are not re-downloaded, as after a URL mediator function extracts and interprets required URLs from the depository, the links are sent to the URL queue where a DNS resolver function that converts the host name of the URL to an IP address. This address is then transferred to a URL mapped function to check if the IP (domain) has already been downloaded, saving the URL-IP and sending the IP back to the URL queue to be queued for crawling. If an IP is duplicated, the IP is discarded. The C-Proc function is the main function for downloading pages with a depth first search algorithm where the main domain obtained from the main queue is downloaded, and sub pages under the same domain are immediately extracted, leaving external pages to be cross checked by the URL IP queue.

2.3.1. Virtualised Crawler Implementations

A. Guerriero [8] proposed a dynamic URL assignment approach for their parallel web crawler, employing multiprocessing via cloud/grid computing technology and dynamic clustering. Therefore, this implementation is a distributed crawler and the suggested architecture consists of a Broker, a Dealer and the Crawlers. The Broker is tasked with selecting URLs from the database which are conveyed to the Dealer, that distributes the URLs in a fashion optimised for load balancing with a dynamic fuzzy clustering method. The crawlers gather new URLs from their scheduled pages and relay them back to the dealer to analyse and store in the database. Crawlers tackle recurring web pages by checking the local cache catalogue that stores URLs that reappear with the highest frequency which is kept fresh by replacing the elements with the lowest recurrence rate.

Tests without the local cache catalogue displayed that 'URLs to be seen' and processed both increased at a linear rate over time, however contrasting this, the 'URLs to be seen' test including the local cache catalogue showed an inverse exponential trend which sharply dropped at a linear rate a the end whilst the 'processed URLs' increased at a linear rate, sharply increasing the gradient simultaneously with the URLs to be seen dropping. This depiction exhibits how less 'URLs to be seen' exist over time since the URLs found from the 'processed URLs' are duplicates with increasing chances, until there are no longer any new URLs resulting in the sharp drop. It can subsequently be inferred that without the local cache catalogue, the crawler would be executed infinitely until interrupted manually or naturally such running out of storage space.

Wani Rohit Bhaginath [18] implemented a very similar crawler to A. Guerriero [8] as they are both distributed crawlers using cloud machines (virtualisation) connected with high speed networks as their parallel multiprocessing technology, along with the usage of the cluster method to assign URLs to each virtual machine. However, differing from A. Guerriero [8], K-means was used to generate clusters for each virtual machine, and the pages were also ranked according to the quantity of pages pointing to the specified URL indicating that the web page data and links are likely to be more desirable. Data redundancy prevention was done using a dynamic assignment framework with a center coordinator (one virtual machine becomes the coordinator for all other machines) in conjunction with a static assignment framework (no central coordinator to countering the risk of single point failure from center coordinator), allowing URLs to be distributed without redundancy.

The usage of virtualization is a beneficial resource due to a multitude of factors such as the ability

to consolidate the system with shared memory, an increase in CPU usage and an overall reduction to crawling CPU time. Resource and load distribution is also simpler to implement and is more efficient.

TABLE II. Comparison of Parallel Crawler Implementations

Implementation	Tithi Dhar [17]	A. Guerriero [8]	Wani Rohit Bhaginath [18]
Parallelisation	Multithreading	Multiprocessing	Multiprocessing
Crawler Algorithm	Depth First Search	Breadth First Search	Best First Search
Optimisation Approach	In-depth parallel implementation avoiding redundant pages	Low cost, maintaining the load balancing and robustness of system through the dynamic URL assignment method and efficiently increasing crawling rate	Low cost web crawler using virtualization with balanced load distribution through a K-means clustering algorithm that assigning requests to the machines according to cluster availability

Yunsoo Lee [19], has proposed an automatic data augmentation method using the web scraping crawling for Deep Neural Networks in order to expand a set of training data through the crawler. Deep Neural Network based classification algorithms today have the capacity to significantly enhance the classification performance, the the extent of exceeding human visual limitations.

Their implementation of the web scraper and crawler allowed them to augment their training sets of vehicle data by percentages ranging from 56% to 74%, utilising a large quantity of vehicle images, the robustly trained the vehicle classification model with images of vehicles under conditions factoring background, brightness and point of view. Source URL of target images were subsequently obtained from Google images and since the data set is very small, imaging processing techniques were applied to the images to augment them. As they were vehicles horizontal flipping, Gaussian noising, brightness conversion, and sharpening processes were suitable methods.

In the Web Scraping Crawling and Data Augmentation Algorithm, after crawling for and obtaining relevant images, imaging processing techniques were immediately applied again. Their final obtained data sets were able to obtain a 92.1% vehicle cross validation accuracy at their peak which was shown to be 13.35% higher using only simply augmented data and 23.58% higher using original data.

Y. Kalmukov [20] presented a method to use web crawlers and image processing techniques to extract, index and store metadata from images. They implemented a three tier architecture where the user interface, business logic and data are split into independent non-overlapping layers. The crawler consists of the web crawler and content analysis modules to traverse and identify web page contents, the image analysis and processing module to identify whether images meet the requirements for extraction and the crawling strategy planner that dictates the web pages to traverse and their order according to the data gathered from the two analysis modules.

3. Implementation and Management Methodology

The management methodology that I will using is the Kanban framework for agile development [21] where the Trello Kanban Board software is utilised in order to schedule my tasks and objectives. Development will be test driven so the test results will be generated prior to implementation when required.

3.1. Project Risks and Ethical/Legal Issues with Mitigation Strategies

Web crawling and scraping is legal under the condition that the acquisition and usage of mined data does not harm any individuals or website's business and operations. In addition, the data must be publicly available (privileges are not required to access the data) and cannot contain sensitive personal data, unless explicit permission is given [22, 23]. These issues will be mitigated by ensuring the mined data is not used for commercial benefit and executing the crawler in incognito mode, ensuring no sessions or cookies that provide privileges on a website are active.

Relevant legal infringements to my project [23] include the Computer Misuse Act 1990 [24], covering Denial of Service attacks caused by extensive or high frequency calls to a single website and

TABLE III. Kanban Board Sectors

Primary Tasks	Secondary Tasks	Specification	Implementation	Testing
Records the functional requirements that require implementation	Records non-functional requirements that would be nice to have but are not compulsory	Feature title Contains a detailed description and/or checklists of the feature to be implemented	Contains a detailed description and/or checklists of the feature to be implemented	Testing conditions and success criteria

Intellectual Property, covering the possibility of mining data that was gathered or recorded with a unique methodology, granting it the capacity to be copyrighted or licensed [25]. The possibility of DoS attacks can be mitigated by applying restrictions on the number of URLs within the CollURLs that fall under a single domain, in addition to utilising a high maximum number of CollURLs, causing more time to be consumed mining different URLs before refreshing a URL (call frequency reduction). Intellectual Property infringement will be avoided by ensuring only the raw unencrypted drone data and its corresponding temporal data is acquired, preventing any unique relationships within the data to be exploitable or identifiable.

Some web resources enforce restrictions or outright block web crawling and scraping prevention mechanisms by various organizations or there may be clear instructions on websites to automated tools to avoid specific site areas or the site completely. This is the website owner's right in it's entirety therefore, it is ethically sound to ensure those boundaries are respected and if any websites are discovered, they will be blacklisted from the web crawler.

4. Specification

I will be initially using the Beautiful Soup 3rd party web scraping library to start parsing images without implementing my own web scraper straight away and using a 3rd party database MySQL in order to store drone images. The web crawler will be a hybrid web crawler with parallel features, benefiting from it's scalebility in order to quickly and broadly search for new relevant URLs and increase the efficiency of the overall web crawler. The incremental implementation is the core component of this web crawler as aerial drone images of natural disasters are not posted or updated regularly. Therefore, maintaining the freshness of relevant discovered web pages should be prioritised whilst searching for old images with free resources.

The proposed project can be recognised as a success under the condition that the aerial drone images can be acquired within a reasonable time frame, and the majority of the obtained images are relevant. The storage efficiency may be evaluated by calculating the presence of any redundant data. The scalability granted by the parallel processing should also be variable, in order to expedite the possibility of evaluating it's contribution to discovering new URLs that may receive a high ranking from the web crawler and page coverage. In addition, it should have a reduced network load from using more crawler processes.

5. Conclusion

It has become apparent that the implementation of web crawlers can be very liberal and the most efficient implementation or combination of implementations is highly dependent on the target data being gathered. All of the methodologies and implementations analysed in Table 1 and Table 2 have a high quantity of commonalities with each other, but web crawlers have a large variety of components that can completely modify the efficiency and functionality such as filters, machine learning etc.

[1] O. Djuraskovic, "How many websites are there? – the growth of the web (1990–2022)," Sep 2022. [Online]. Available: https://firstsiteguide.com/how-many-websites/#:~:text=question%20people%20ask%3A-,How%20many%20websites%20are%20there%3F,over%201.98%20billion%20websites%20online.

TABLE IV. Project Task Plan

Optimal Completion Dates	Functional Requirements	Success Criteria
23rd January 2023	Basic Web Crawler	URLs can be extracted from web pages and stored in a text file.
30th January 2023	Breadth First Search Image Extraction with Web Crawler	Target images can be extracted from a web page and downloaded.
13rd February 2023	Parallelisation of Web Crawler	The average web coverage rate should have increased from the old results.
20th February 2023	Create Database for Aerial Images	Aerial Images can be viewed and edited in the database
6th March 2023	Incremental Ranking Module at regular intervals according to web page deviation from historical data and modification frequency	Output ranking scores and the historical data and modification frequency they were derived from from each URL and compare URLs. URLs with highest data overall should have the highest rank.
13rd March 2023	Incremental Web Crawler	Record and output the total number of updates and ranking values of all URLS. The URL with the highest ranking value should have the highest refresh quantity. New URLs should also be extracted immediately.
27th March 2023	Parallel and Incremental Web Crawler	Compare performance with and without parallelisation and the coverage rate should have increased.
3rd April 2023	Comment/Docstring and Final Testing and Optimisations	All tests pass and doctoring folder compiles.
17th April 2023	Finish Report	Finish report and prepare anything for demonstration.
Optimal Completion Dates	Non-Functional Requirements	Success Criteria
N/A	Bloom Filter	A web page is applied to the Web Crawler twice and a message should be printed indicating that the filter let the 2nd web page get cross checked with the collection of URLs
N/A	Attempt to Create a Specialised Aerial Web Scraper with Deep Neural Networks	The Web Scraper can identify and download aerial drone data automatically
N/A	Attempt to Create Database UI as a query related graph and enable query searches with natural disaster query attributes	Images and/or URLs of a specified natural disaster that was queried can be sorted and viewed.
N/A	Crawler process status, management and records	Images and/or URLs of a specified natural disaster that was queried can be sorted and viewed and records of all operations with may be useful for the final report.

[2] D. Georgiev, “How many websites are there in 2022? [updated guide],” Oct 2022. [Online]. Available: https://techjury.net/blog/how-many-websites-are-there/

[3] R. Kumar, A. Jain, and C. Agrawal, “Survey of web crawling algorithms,” 2016. [Online]. Available: https://dx.doi.org/10.5281/zenodo.3674658

[4] N. Yaghmaei, *Human cost of disasters: An overview of the last 20 years, 2000-2019*. United Nations Office for Disaster Risk Reduction, 2020.

[5] U. N. O. for Disaster Risk Reduction, “Global assessment report on disaster risk reduction 2022: Our world at risk: Transforming governance for a resilient future.” United Nations Office for Disaster Risk Reduction, 7bis Avenue de la Paix, CH1211 Geneva 2, Switzerland, Tech. Rep., 2022.

[6] W. Sun, P. Bocchini, and B. D. Davison, “Applications of artificial intelligence for disaster management,” *Natural Hazards*, vol. 103, no. 3, p. 2631–2689, 2020.

[7] S. Sharma and P. Gupta, “The anatomy of web crawlers,” in *International Conference on Computing, Communication Automation*, 2015, pp. 849–853.

[8] A. Guerriero, F. Ragni, and C. Martines, “A dynamic url assignment method for parallel web crawler,” in *2010 IEEE International Conference on Computational Intelligence for Measurement Systems and Applications*, 2010, pp. 119–123.

[9] A. Garg, K. Gupta, and A. Singh, “Survey of web crawler algorithms,” *International Journal of Advanced Research in Computer Science; Vol 8, No 5 (2017): May-June 2017; 426-428 ; 0976-5697 ; 10.26483/ijarcs.v8i5*, 6 2017. [Online]. Available: http://www.ijarcs.info/index.php/Ijarcs/article/view/3321

[10] S. Xi, F. Sun, and J. Wang, “A cognitive crawler using structure pattern for incremental crawling and content extraction,” in *9th IEEE International Conference on Cognitive Informatics (ICCI’10)*, 2010, pp. 238–244.

[11] H. Bullot, S. Gupta, and M. Mohania, “A data-mining approach for optimizing performance of an incremental crawler,” in *Proceedings IEEE/WIC International Conference on Web Intelligence (WI 2003)*, 2003, pp. 610–615.

[12] Z. Shi, M. Shi, and W. Lin, “The implementation of crawling news page based on incremental web crawler,” in *2016 4th Intl Conf on Applied Computing and Information Technology/3rd Intl Conf on Computational Science/Intelligence and Applied Informatics/1st Intl Conf on Big Data, Cloud Computing, Data Science Engineering (ACIT-CSII-BCD)*, 2016, pp. 348–351.

[13] A. Kumar, Jun 2022. [Online]. Available: https://www.geeksforgeeks.org/bloom-filters-introduction-and-python-implementation/

[14] W. Hui-chang, R. Shu-hua, and T. Qi-jie, “The implementation of a web crawler url filter algorithm based on caching,” in *2009 Second International Workshop on Computer Science and Engineering*, vol. 2, 2009, pp. 453–456.

[15] W. Liu, J. Xiao, and J. Yang, “A sample-guided approach to incremental structured web database crawling,” in *The 2010 IEEE International Conference on Information and Automation*, 2010, pp. 890–895.

[16] N. Oceanic and A. Administration, Mar 2022. [Online]. Available: https://www.noaa.gov/education/resource-collections/climate/changing-seasons#:~:text=Seasonal%20effects,for%20humans%20and%20other%20organisms.

[17] T. Dhar, S. Mazumder, S. Dhar, S. Karak, and D. Chatterjee, “An approach to design and implement parallel web crawler,” in *2021 International Conference on Smart Generation Computing, Communication and Networking (SMART GENCON)*, 2021, pp. 1–4.

[18] W. R. Bhaginath, S. Shingade, and M. Shirole, “Virtualized dynamic url assignment web crawling model,” in *2014 International Conference on Advances in Engineering Technology Research (ICAETR - 2014)*, 2014, pp. 1–7.

[19] Y. Lee and S.-J. Kang, “Web scraping crawling-based automatic data augmentation for deep neural networks-based vehicle classifications,” in *2019 IEEE International Conference on Consumer Electronics (ICCE)*, 2019, pp. 1–2.

[20] Y. Kalmukov and I. Valova, “Design and development of an automated web crawler used for building image databases,” in *2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, 2019, pp. 1553–1558.

[21] C. Drumond, “Agile project management,” 2022. [Online]. Available: https://www.atlassian.com/agile/project-management

[22] S. Upadhyay, V. Pant, S. Bhasin, and M. K. Pattanshetti, “Articulating the construction of a web scraper for massive data extraction,” in *2017 Second International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, 2017, pp. 1–4.

[23] C. Dilmegani, “Watch-outs for legal and ethical web scraping in 2022,” *research.aimultiple.com*, Jan 2022. [Online]. Available: https://research.aimultiple.com/web-scraping-ethics/

[24] T. C. P. Service, “Computer misuse act | the crown prosecution service,” Feb 2020. [Online]. Available: https://www.cps.gov.uk/legal-guidance/computer-misuse-act

[25] L. M. A. R. D. M. W. Group, “Intellectual property,” 2022. [Online]. Available: https://datamanagement.hms.harvard.edu/evaluate/intellectual-property#:~:text=Intellectual%20Property%20and%20Copyright,can%20be%20copyrighted%20or%20licensed.