International Journal of
Technology and Systems
(IJTS)

An Ontology Based Web Crawler with a Near-Duplicate Detection System to Improve the Performance of a Web Crawler

Ngulamu Daines Walowe, Dr. Michael Kimwele and Dr. Ann Kibe

International Journal of Technology and Systems

ISSN 2518-881X (Online)

Vol.9, Issue 5, No.2, pp 15 – 28, 2024

www.iprjb.org

An Ontology Based Web Crawler with a Near-Duplicate Detection System to Improve the Performance of a Web Crawler

[1*]Ngulamu Daines Walowe
Master's Student: Jomo Kenyatta University of Agriculture and Technology

[2]Dr. Michael Kimwele and [3]Dr. Ann Kibe
Jomo Kenyatta University of Agriculture and Technology

Article History

Received 1st July 2024

Received in Revised Form 3rd August 2024

Accepted 30th September 2024

How to cite in APA format:

Walowe, N., Kimwele, M., & Kibe, A. (2024). An Ontology Based Web Crawler with a Near-Duplicate Detection System to Improve the Performance of a Web Crawler. *International Journal of Technology and Systems*, *9*(5), 15–28. https://doi.org/10.47604/ijts.2984

Abstract

Purpose: The aim of the study is to examine how an ontology-based web crawler with a near-duplicate detection system improves the performance of a web crawler.

Methodology: The experiment was carried out using secondary data from a sample web site which was used since crawling is an endless process. Using these two approaches, the ontology web crawler would search for relevant searches according to the search query of the user while the near-duplicate detection system would eliminate redundant data.

Findings: It was observed that ontology web crawler performed better and faster than a normal crawler. It takes less execution time to search the web than other web crawlers. This is due to the fact that web documents are being filtered by the ontology web crawler such that only relevant web documents are retrieved according to the search query of the user. The relevant documents are further filtered by a near-duplicate detection system by removing web pages that are duplicates of each other and also remove near-duplicate web documents. This further reduces the number of web pages retrieved by the web crawler. This model saves on storage space because of the reduced number of web pages retrieved as it takes care of irrelevant and redundant web pages searched.

Unique Contribution to Theory, Practice and Policy: The study recommends that the model can be improved to be dynamic by adding new relations that is the crawler should search for web pages related to the search even if they don't contain the keywords searched. Domains and concepts should be added when visiting new web pages. Standardization of weights needs to be done because as of now experts assign weights to terms according to the area of expertise and knowledge.

Keywords: *Ontology Based Web Crawler, Near-Duplicate Detection System, Performance*

INTRODUCTION

A web crawler is a program that creates entries for a search engine after visiting web sites and reading their web pages (Lawankar and Mangrulkar, 2016). All computers are connected by virtual networks or mesh known as the internet. There are various communication systems contained in the internet including academic, business, private and public networks. Various communication media are used to connect these communication systems such as, fiber optic cables, telephone lines, microwave and satellite. There is no single person or organization that owns the internet therefore information contained in the internet is extremely difficult to manage (Saini, 2016).

According to Pranav and Chauhan (2015), the objective of a crawler is to gather as many useful pages as quickly and as efficiently as possible. A copy of all the visited pages is create by a web crawler for later processing by the search engine that will index the downloaded pages to provide fast searches.

According to Komal and Dixit (2016), the internet has become the largest unstructured database for accessing information over the documents. Due to this, there are issues related to the World Wide Web that makes crawling difficult. The world wide web has grown from a thousand to a billion in recent years. It is also changing as time also changes. Due to its explosion in size web crawlers are important for locating information (Komal and Dixit, 2016). A large number of web pages are also constantly being added every day and information is constantly changing. The nature of the information on the web also gets changed (Udapure et al. 2014).

There is also the issue of mirrored and near-duplicate pages when a crawler searches for information on the web. According to Singh and Imtiyaz (2017), the detection of duplicate records is tested due to the rapid growth in data volumes and the need to integrate data from various heterogeneous sources. There is an increase of duplicate web pages on the internet due to lack of a standard mechanism to guarantee the non-existence of a webpage before hosting them (Arun and Sumesh, 2015). Duplicate web pages and near-duplicate web pages they affect the quality of the crawled content. They waste the user’s time, impact on the crawler’s storage affect page ranking and create additional overhead on search engines (Subramanyam et al. 2016).

Architecture of a Web Crawler

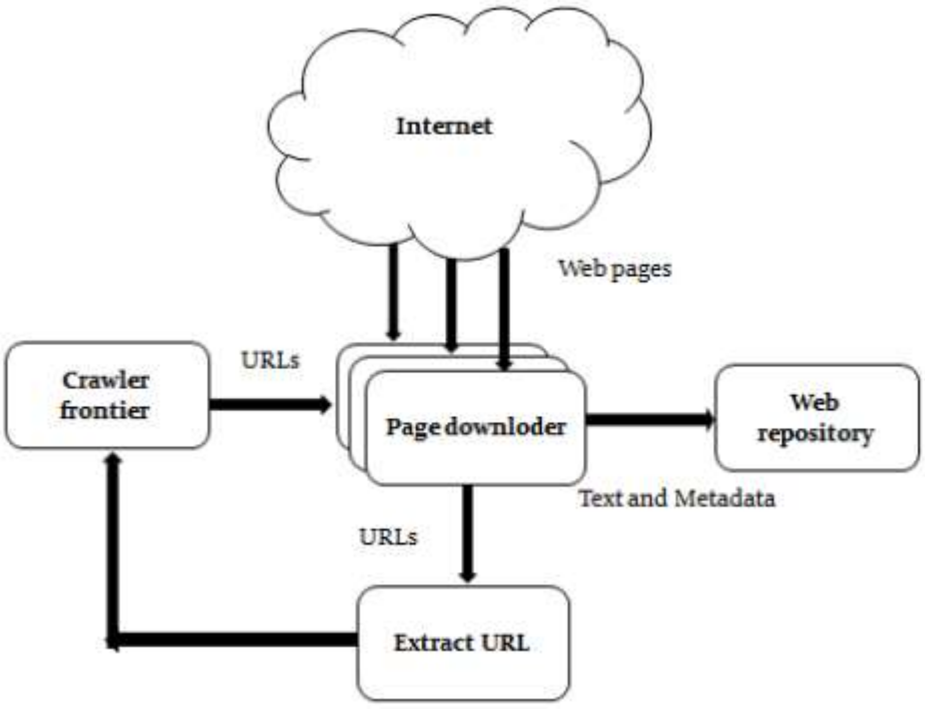


Figure 1: Architecture of a Web Crawler

Source: Amudha (2017)

Problem Statement

Web crawlers have faced several limitations regarding retrieval of data from the World Wide Web. One of the challenges of web crawlers is being unable to get relevant and quality information according to the search query of the user from the web. This is due to the large volume of the World Wide Web. Information is provided by millions of web content providers and web pages are constantly being added and content of web pages keeps changing (Suryawanshi and Patil, 2015).

Due to the large volume of the internet, users must either browse through a hierarchy of concepts to find the information they need or submit a query to a search engine to wade through thousands of results most of which are irrelevant (Suganiya and Haripriya, 2015).

There is an increase of duplicate web pages on the internet due to lack of a standard mechanism to guarantee the non-existence of a webpage before hosting them (Anun and Sumesh, 2015). Duplicate web pages and near-duplicate web pages affect the quality of the crawled content. They waste the user’s time, impact on the crawler’s storage affect page ranking and create additional overhead on search engines (Subramanyam et al. 2016).

The problem addressed by this thesis was the retrieval of irrelevant and redundant information from the web when a user enters a search query.

Evolution of Web Crawlers

Web crawlers could initially just collect data, modern day crawlers are capable of monitoring vulnerabilities and accessibility in web application. Web crawlers have been around since the web was just the size of 100,000 web pages (Singla, 2015). The first web crawlers were RBSE spider developed by NASA in 1994, then came the WebCrawler developed by Brian Pinkerton in 1994 and the third was Archive.org known as way back machine (Williams, 2015). Web crawlers have evolved in the last decade from spiders to bots to multitasking and multipurpose web crawlers. The last decade has introduced distributed crawlers in 2003, which could crawl the world wide web pages in seconds, the Heritrix crawler developed in 2004 which was specifically intended to support focused and broad crawls. Ajax crawler was developed in 2007 which crawled through rich internet application and used Breadth first algorithm to index the web pages. Googlebot-mobile and Bingbot mobile were developed in 2014 due to advancement in internet technology and ease of accessibility via mobile phones (Singla, 2015).

Conceptual Web Crawler

According to Sharma and Devi (2016), web crawling is a tedious process because of the large volume of the web as the crawler crawls billions of web pages every day. The rate of change of web pages is high and also web pages are being added, removed or changed every day.

The search crawl results are normally wasteful as most of the pages retrieved or downloaded do not match the search query of interest (Suganiya and Haripriya, 2015). Web pages downloaded may also be mirrored or near-duplicates reducing the quality of the search and wasting the user's time (Nirmalrani et al. 2015). Due to this there is a need to retrieve relevant information according to the search query and at the same time be able to retrieve non-redundant data.

The proposed web crawler was a combination of the ontology-based web crawler and a near-duplicate detection system. The ontology-based web crawler would ensure that relevant web pages are retrieved according to the search query of the user and a near-duplicate detection system would ensure retrieval of non-redundant information from the web reducing overhead on the search engine and saving the user's time in sorting of documents. Out of the relevant pages retrieved by the ontology crawler, the near-duplicate detection system would ensure that there are no duplicates or near-duplicate in the crawled web pages.

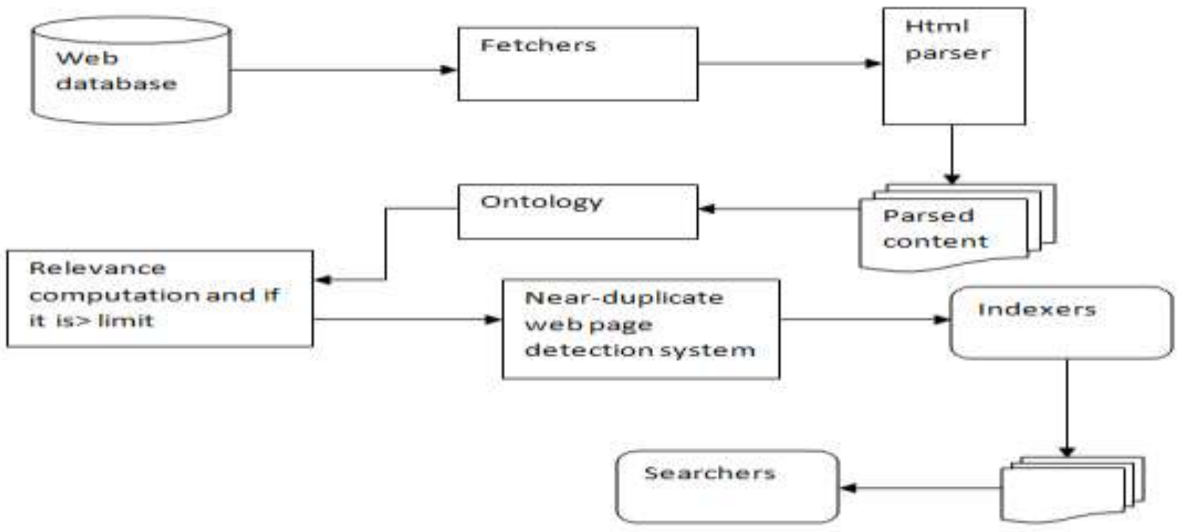

Web database
Fetchers
Html parser
Parsed content
Ontology
Relevance computation and if it is> limit
Near-duplicate web page detection system
Indexers
Searchers

Figure 2: Conceptual web crawler

Source: Kumar and Kumar (2015)

METHODOLOGY

This chapter outlines the research methodology and specific approach that was adopted from (Kumar and Kumar, 2015) to investigate the appropriate software for checking web pages and duplicate web pages.

Steps for this approach:

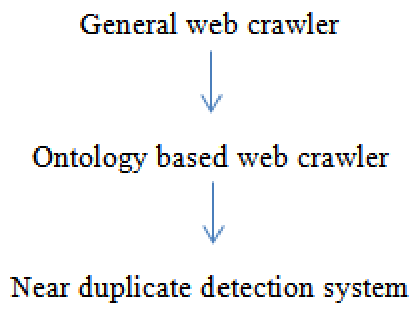


Figure 3: An Illustration of How the Ontology Based Web Crawler With a Near Duplicate Detection System Was Conducted.

A general crawler is a crawler that crawls one page at a time until all the pages have been indexed. An ontology crawler is guided by an ontology describing the domain of interest and focuses on returning pages that are relevant to the topic ontology. A near duplicate detection system is a system that searches for web pages that are similar to each other or duplicates of each other and discards them.

Research Design

The researcher adopted a descriptive survey design. This was preferred since it had the aspect of investigating possible relationships between one or two variables. Descriptive research involves gathering data and later on tabulates, organizes, depicts and describes the data (Glass & Hopkins, 1984). Visual aids such as graphs and charts are used to aid the reader in understanding data distribution. Some tests were conducted for the purpose of testing the proposed approach.

Parameters

The relevance score was computed with reference taken from Kumar and Kumar (2015). The algorithm is, If SIMILAR_P <RANGE where "SIMILAR_P" is a relevant web page; if it is less than "RANGE" that is the limit set by the researcher, then web page is discarded and if SIMILAR_P >RANGE, then the web page is downloaded. The cutoff point that was set by the researcher was "3" which was used to check the relevance of the web document. Weight was assigned to all key words and phrases by the researcher in the ontology. Terms which were more specific were assigned more weight and terms which were more common and were in more than one domain had less weight. For example, the word "internet" is a common term and is assigned the weight "1" and "website" is a more specific term and assigned weight "15". The term is calculated according to the number of times it appears in a web document. The number of occurrence of a term is termed as frequency. The frequency of a term is multiplied by the weight assigned to it. The value obtained will be cross checked against the cutoff point. If the value obtained is greater than the cutoff point, then the document is relevant and therefore indexed and if the value obtained is less than the cutoff point, then the document is irrelevant and is therefore discarded.

The introduction of a near-duplicate web page detection system is with reference from Arun and Sumesh (2015), in the ontology web crawler will ensure that indexed relevant web documents are non-redundant. A threshold was set for the comparison of web documents which was "80". Web documents were compared against each other to ensure that they are not duplicates or near-duplicates of each other. This was done by using a hybrid mechanism. Filtering was done and it used the number of sentences on the web page. The difference in number of sentences were to be less than or equal to the sentence threshold. The documents were then calculated bit by bit and the simHash value obtained from the calculation was compared with the set sentence threshold. If the calculated difference was less than or equal to the sentence threshold then the web document was termed as a duplicate and was discarded and if the calculated difference was greater than the threshold, then the document was termed as a non-duplicate document and indexed.

Pseudo code for relevance:

```
The web page is checked for legitimacy that is, which markup language
The web page is added to line once markup language is characterized
The web page is parsed
Contents of the web page are compared to ontology
Threshold is set to determine the relevance of the document
If web page is less than the set threshold then it is discarded else
The web page is downloaded
```

Pseudo code for duplicate detection:

```
Terms are assigned weights
Frequency are assigned to terms
A TDW list is computed
Input web pages is compared with existing web pages
Bit by bit comparison is done to the input web pages
If the specified threshold is satisfied then web pages is marked as a near-duplicate
```

Flow chart for proposed model

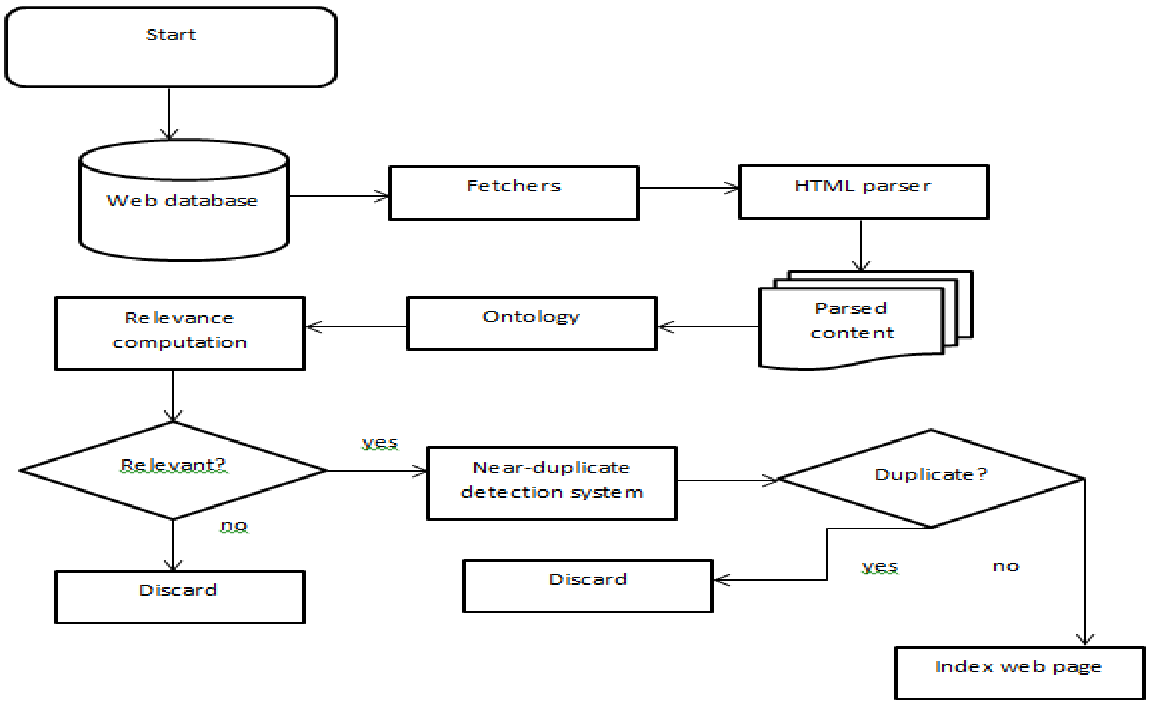


Figure 4: Flow Chart for Proposed Model

The researcher presented a case of how the suggested web crawler computes the relevancy of the web given in reference for the keyword "web design". Web design was the root class in the ontology. It has subclasses developer, design and web. The developer class is further divided into subclasses Front-end, back-end and full stack. Design has subclasses layout, color and graphics. Web has subclasses search engine, websites and browser. Search engine is further divided to search, URLs, index and crawler. Websites is divided into page and database.

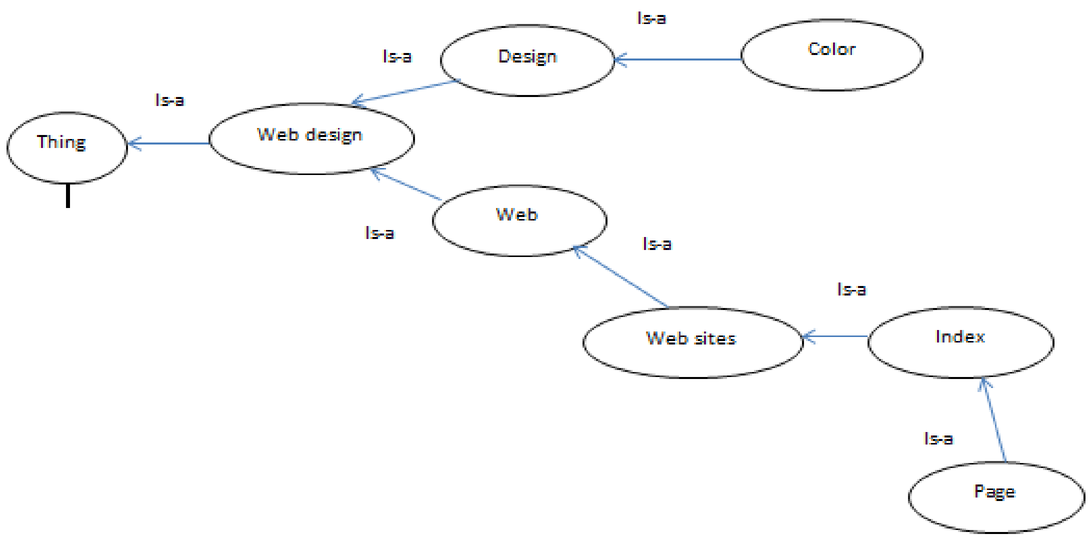
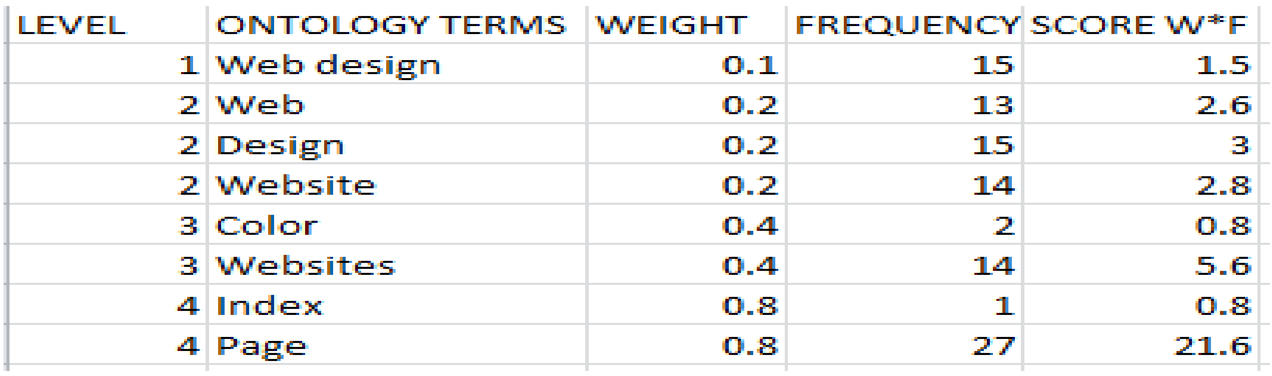


Figure 5: Graphical Representation of the Ontology for Web Design

Weight is assigned to different terms in the ontology based on how important it is in that particular domain. Experts assign weight depending on their field of expertise.



LEVEL	ONTOLOGY TERMS	WEIGHT	FREQUENCY	SCORE W*F
1	Web design	0.1	15	1.5
2	Web	0.2	13	2.6
2	Design	0.2	15	3
2	Website	0.2	14	2.8
3	Color	0.4	2	0.8
3	Websites	0.4	14	5.6
4	Index	0.8	1	0.8
4	Page	0.8	27	21.6

Figure 6: Weight Table

Here, relevance of a page is SIMILAR_P=∑38.7

The LIMIT set by the researcher was "3" therefore the web page is SIMILAR_P>LIMIT

The crawler concludes that the web page is relevant and starts downloading it.

The user was able to search for a web page after it was indexed.

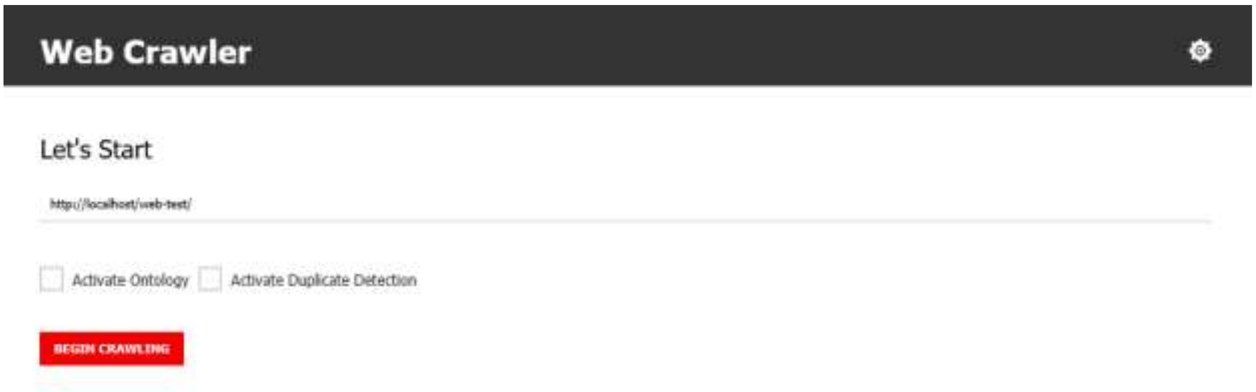
The content of web pages are then checked bit by bit if they are similar or not. If they pass the threshold set, that is 60% and above then the web pages are near-duplicates. If the similarity check is less than 60%, then the web pages are not similar.

Experimental Setup

The experiment was conducted on a windows machine running AMD E1-1200 APU CPU 1.40 GHz processor with 2GB of RAM. Xampp software was installed and used to develop the crawler. Simulation was done using 102 pages from a sample website. The URL for the sample website is "http://localhost/web-test/"

RESULTS

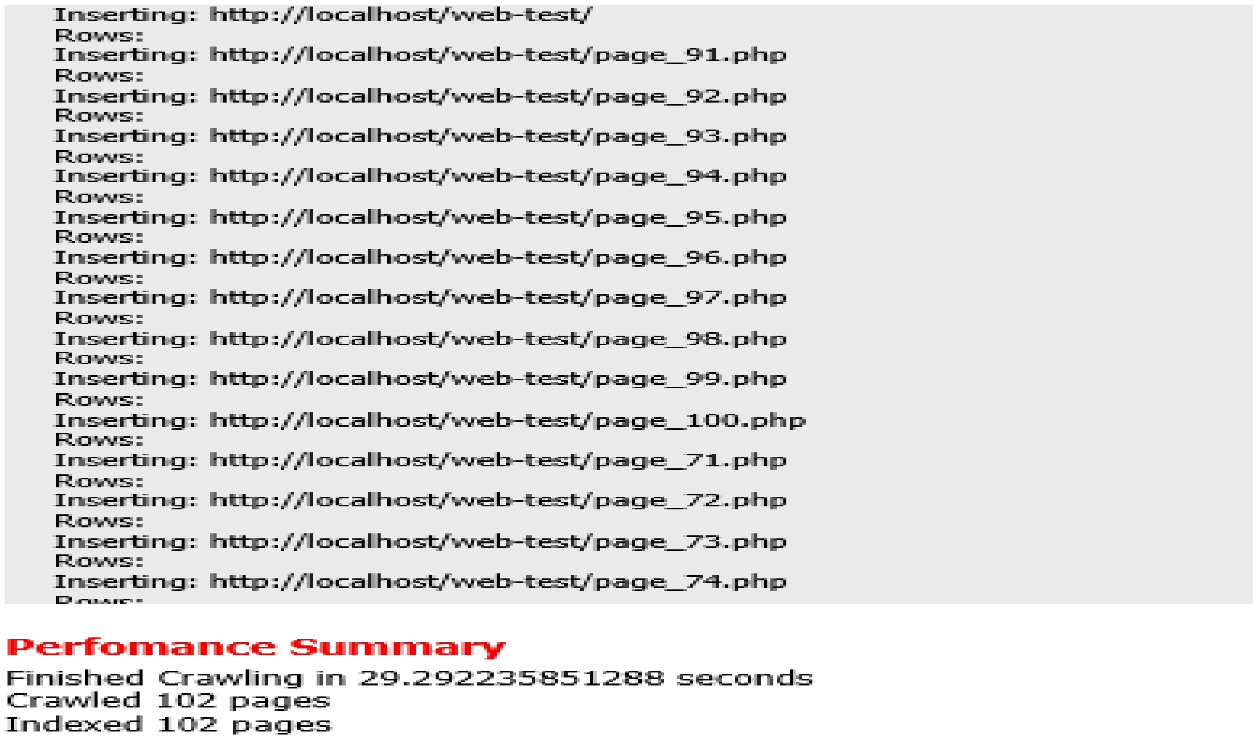
Experimental results



http://localhost/web-test/

Figure 7: Interface for Web Page Indexing

The researcher started by entering a URL "http://localhost/web-test/" of the sample website to be indexed. Relevance and duplicate detection is not being checked at this point.



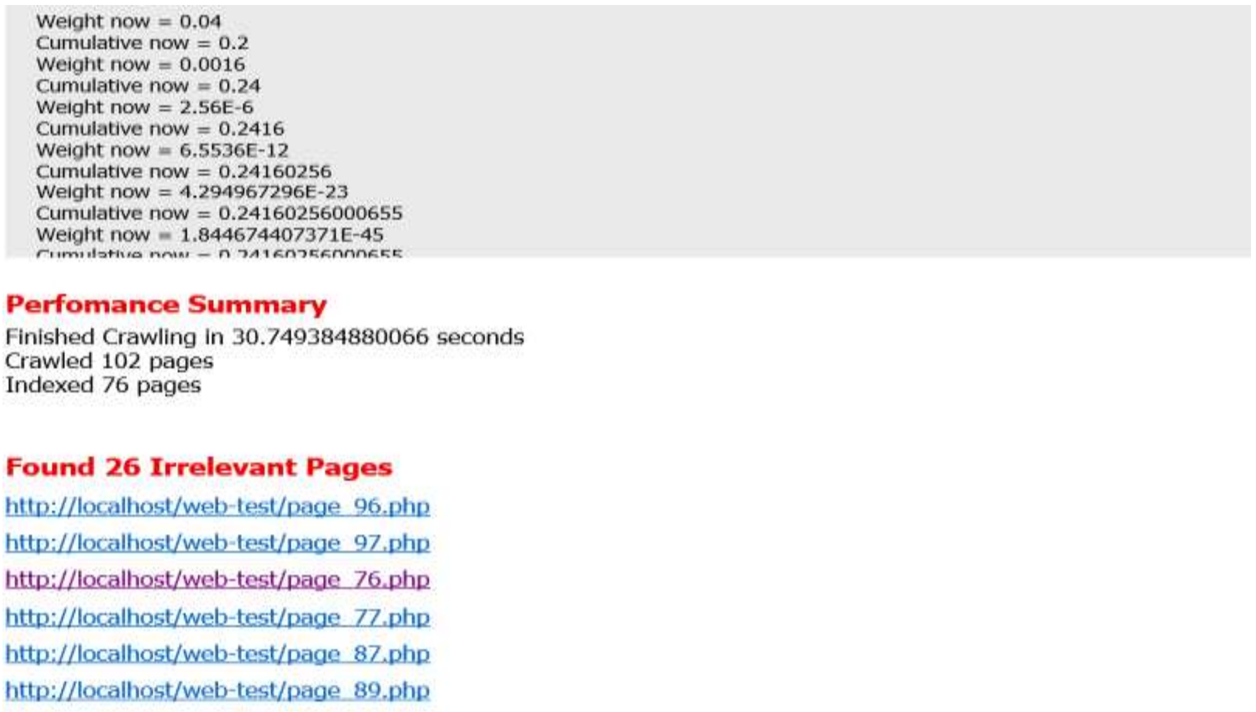
```
Inserting: http://localhost/web-test/
Rows:
Inserting: http://localhost/web-test/page_91.php
Rows:
Inserting: http://localhost/web-test/page_92.php
Rows:
Inserting: http://localhost/web-test/page_93.php
Rows:
Inserting: http://localhost/web-test/page_94.php
Rows:
Inserting: http://localhost/web-test/page_95.php
Rows:
Inserting: http://localhost/web-test/page_96.php
Rows:
Inserting: http://localhost/web-test/page_97.php
Rows:
Inserting: http://localhost/web-test/page_98.php
Rows:
Inserting: http://localhost/web-test/page_99.php
Rows:
Inserting: http://localhost/web-test/page_100.php
Rows:
Inserting: http://localhost/web-test/page_71.php
Rows:
Inserting: http://localhost/web-test/page_72.php
Rows:
Inserting: http://localhost/web-test/page_73.php
Rows:
Inserting: http://localhost/web-test/page_74.php
Rows:
```

Perfomance Summary

```
Finished Crawling in 29.292235851288 seconds
Crawled 102 pages
Indexed 102 pages
```

Figure 8: Indexed Web Pages without Ontology or Duplicate Detection

All 102 web pages are being indexed because relevance and duplicate detection are not being considered at this point.



```
Weight now = 0.04
Cumulative now = 0.2
Weight now = 0.0016
Cumulative now = 0.24
Weight now = 2.56E-6
Cumulative now = 0.2416
Weight now = 6.5536E-12
Cumulative now = 0.24160256
Weight now = 4.294967296E-23
Cumulative now = 0.24160256000655
Weight now = 1.844674407371E-45
Cumulative now = 0.24160256000655
```

Perfomance Summary

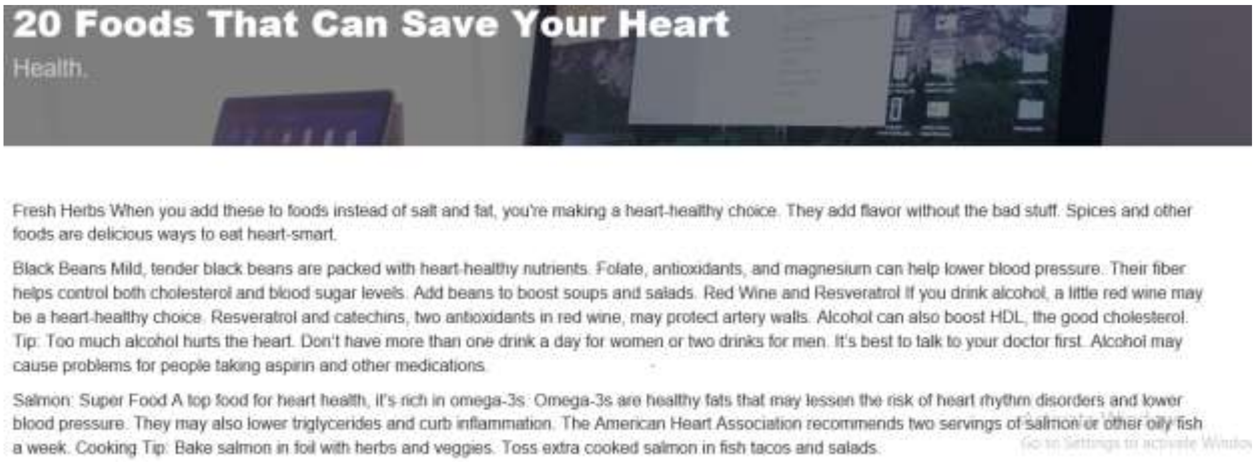
```
Finished Crawling in 30.749384880066 seconds
Crawled 102 pages
Indexed 76 pages
```

Found 26 Irrelevant Pages

http://localhost/web-test/page_96.php
http://localhost/web-test/page_97.php
http://localhost/web-test/page_76.php
http://localhost/web-test/page_77.php
http://localhost/web-test/page_87.php
http://localhost/web-test/page_89.php

Figure 9: Relevance Computation

When ontology is used, the crawler indexes 76 pages which are relevant pages as the crawler has found 26 irrelevant web pages. A relevant web page is a web page where the content matches the ontology. Here the domain of the ontology is "web design". Therefore 26web pages did not match the ontology that is, the content was not about web design.



20 Foods That Can Save Your Heart

Health.

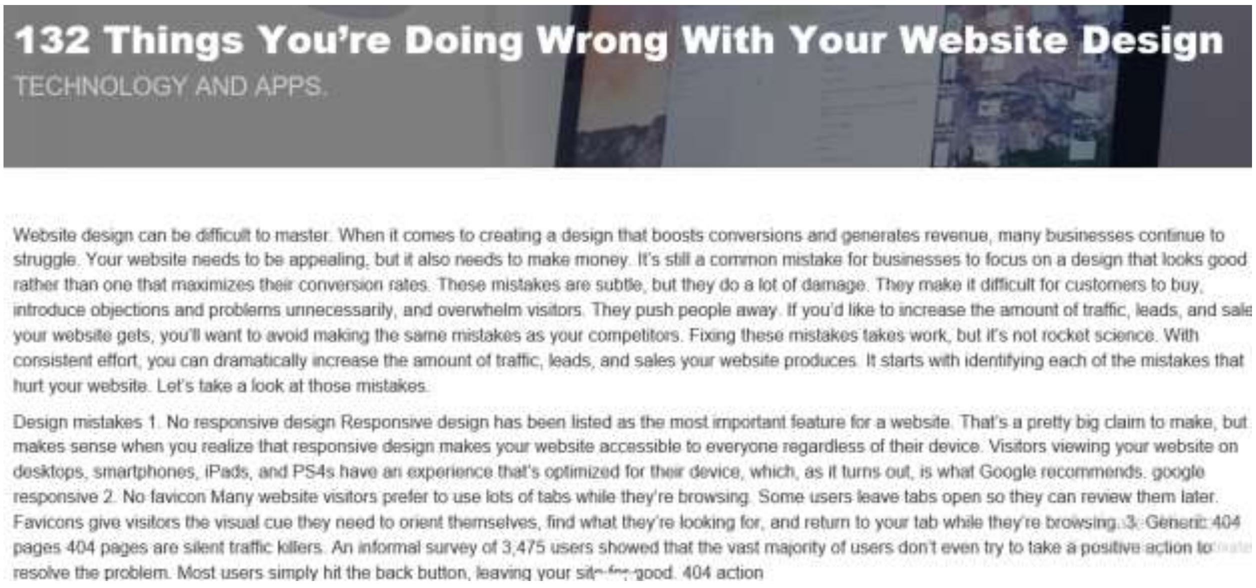
Fresh Herbs When you add these to foods instead of salt and fat, you're making a heart-healthy choice. They add flavor without the bad stuff. Spices and other foods are delicious ways to eat heart-smart.

Black Beans Mild, tender black beans are packed with heart-healthy nutrients. Folate, antioxidants, and magnesium can help lower blood pressure. Their fiber helps control both cholesterol and blood sugar levels. Add beans to boost soups and salads. Red Wine and Resveratrol If you drink alcohol, a little red wine may be a heart-healthy choice. Resveratrol and catechins, two antioxidants in red wine, may protect artery walls. Alcohol can also boost HDL, the good cholesterol. Tip: Too much alcohol hurts the heart. Don't have more than one drink a day for women or two drinks for men. It's best to talk to your doctor first. Alcohol may cause problems for people taking aspirin and other medications.

Salmon: Super Food A top food for heart health, it's rich in omega-3s. Omega-3s are healthy fats that may lessen the risk of heart rhythm disorders and lower blood pressure. They may also lower triglycerides and curb inflammation. The American Heart Association recommends two servings of salmon or other oily fish a week. Cooking Tip: Bake salmon in foil with herbs and veggies. Toss extra cooked salmon in fish tacos and salads.

Figure 10: Sample of Web Page 76

The content of this web page is about food that can save your heart. This content does not match the ontology that is "web design" therefore it is identified as an irrelevant web page.



132 Things You're Doing Wrong With Your Website Design

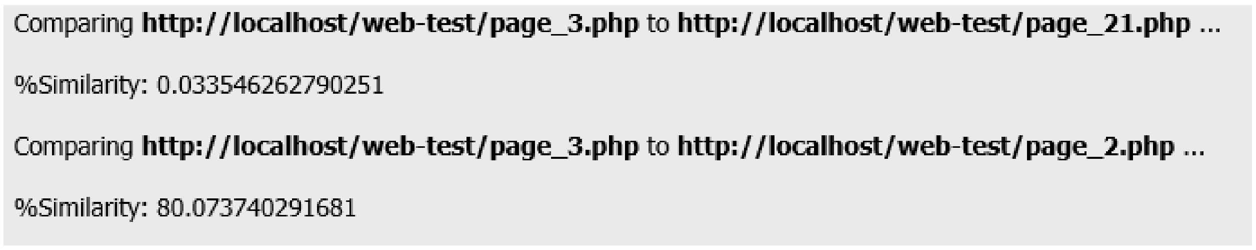
TECHNOLOGY AND APPS

Website design can be difficult to master. When it comes to creating a design that boosts conversions and generates revenue, many businesses continue to struggle. Your website needs to be appealing, but it also needs to make money. It's still a common mistake for businesses to focus on a design that looks good rather than one that maximizes their conversion rates. These mistakes are subtle, but they do a lot of damage. They make it difficult for customers to buy, introduce objections and problems unnecessarily, and overwhelm visitors. They push people away. If you'd like to increase the amount of traffic, leads, and sale your website gets, you'll want to avoid making the same mistakes as your competitors. Fixing these mistakes takes work, but it's not rocket science. With consistent effort, you can dramatically increase the amount of traffic, leads, and sales your website produces. It starts with identifying each of the mistakes that hurt your website. Let's take a look at those mistakes.

Design mistakes 1. No responsive design Responsive design has been listed as the most important feature for a website. That's a pretty big claim to make, but makes sense when you realize that responsive design makes your website accessible to everyone regardless of their device. Visitors viewing your website on desktops, smartphones, iPads, and PS4s have an experience that's optimized for their device, which, as it turns out, is what Google recommends. google responsive 2. No favicon Many website visitors prefer to use lots of tabs while they're browsing. Some users leave tabs open so they can review them later. Favicons give visitors the visual cue they need to orient themselves, find what they're looking for, and return to your tab while they're browsing. 3. Generic 404 pages 404 pages are silent traffic killers. An informal survey of 3,475 users showed that the vast majority of users don't even try to take a positive action to resolve the problem. Most users simply hit the back button, leaving your site for good. 404 action

Figure 11: Sample Web Page 75

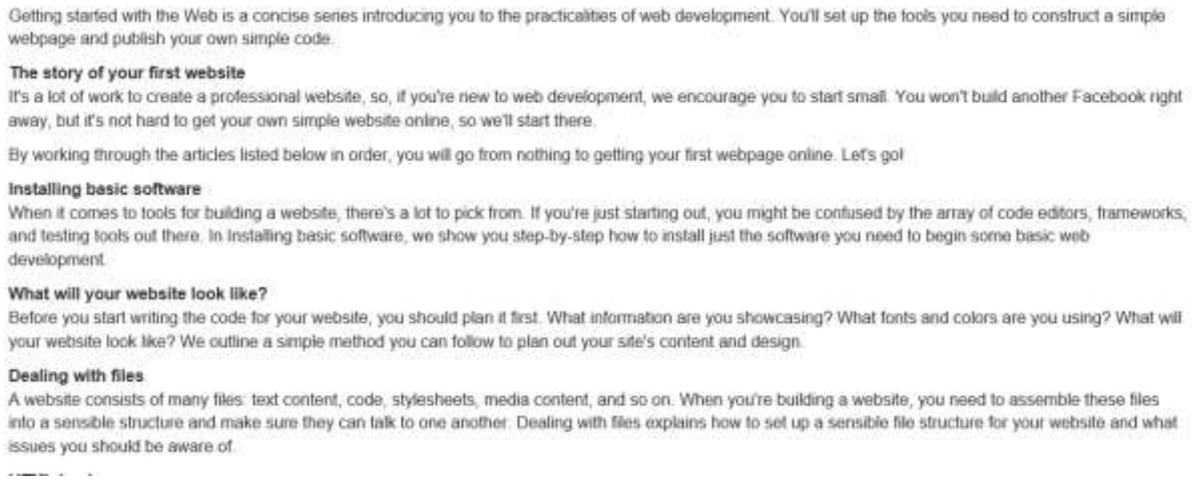
The contents of this web page are about "what you are doing wrong with your website". Therefore this web page is identified as a relevant web page as it matches the ontology "web design"



```
Comparing http://localhost/web-test/page_3.php to http://localhost/web-test/page_21.php ...

%Similarity: 0.033546262790251

Comparing http://localhost/web-test/page_3.php to http://localhost/web-test/page_2.php ...

%Similarity: 80.073740291681
```

Figure 12: Similarity Comparison

Web page 3 is compared to web page 21 and the similarity is 0.03%. This means that the content of the web pages is not similar. Web page 3 is also compared to web page 2 and the similarity is 80%. This means that web page 3 and web page 2 are duplicates or near-duplicates of each other. Similarity is defined as a part of text that has been modified or altered.


Getting started with the Web is a concise series introducing you to the practicalities of web development. You'll set up the tools you need to construct a simple webpage and publish your own simple code.

The story of your first website

It's a lot of work to create a professional website, so, if you're new to web development, we encourage you to start small. You won't build another Facebook right away, but it's not hard to get your own simple website online, so we'll start there.

By working through the articles listed below in order, you will go from nothing to getting your first webpage online. Let's go!

Installing basic software

When it comes to tools for building a website, there's a lot to pick from. If you're just starting out, you might be confused by the array of code editors, frameworks, and testing tools out there. In Installing basic software, we show you step-by-step how to install just the software you need to begin some basic web development.

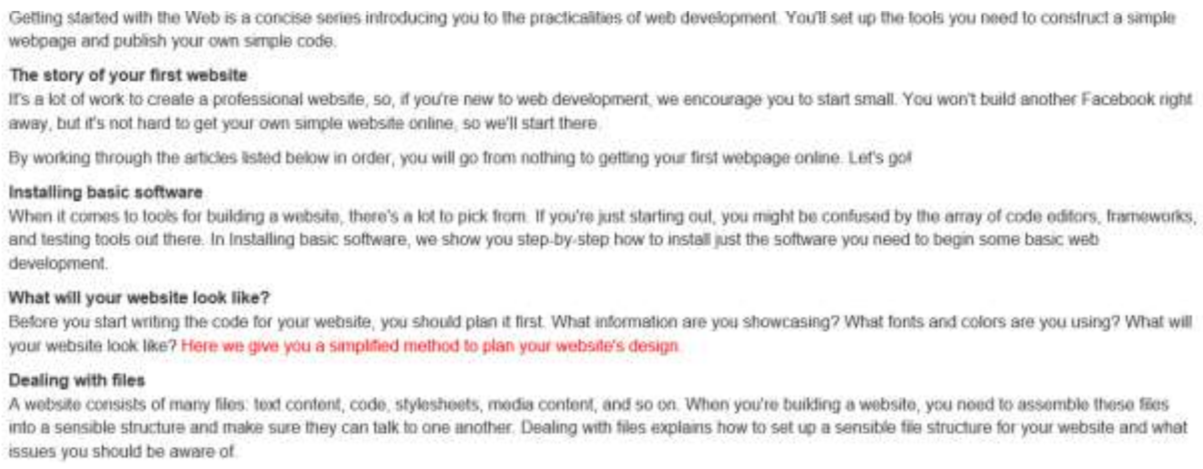
What will your website look like?

Before you start writing the code for your website, you should plan it first. What information are you showcasing? What fonts and colors are you using? What will your website look like? We outline a simple method you can follow to plan out your site's content and design.

Dealing with files

A website consists of many files: text content, code, stylesheets, media content, and so on. When you're building a website, you need to assemble these files into a sensible structure and make sure they can talk to one another. Dealing with files explains how to set up a sensible file structure for your website and what issues you should be aware of.

Figure 13: Sample of Web Page 2


Getting started with the Web is a concise series introducing you to the practicalities of web development. You'll set up the tools you need to construct a simple webpage and publish your own simple code.

The story of your first website

It's a lot of work to create a professional website, so, if you're new to web development, we encourage you to start small. You won't build another Facebook right away, but it's not hard to get your own simple website online, so we'll start there.

By working through the articles listed below in order, you will go from nothing to getting your first webpage online. Let's go!

Installing basic software

When it comes to tools for building a website, there's a lot to pick from. If you're just starting out, you might be confused by the array of code editors, frameworks, and testing tools out there. In Installing basic software, we show you step-by-step how to install just the software you need to begin some basic web development.

What will your website look like?

Before you start writing the code for your website, you should plan it first. What information are you showcasing? What fonts and colors are you using? What will your website look like? Here we give you a simplified method to plan your website's design

Dealing with files

A website consists of many files: text content, code, stylesheets, media content, and so on. When you're building a website, you need to assemble these files into a sensible structure and make sure they can talk to one another. Dealing with files explains how to set up a sensible file structure for your website and what issues you should be aware of.

Figure 14: Sample of Web Page 3

The text “Here we give you a simplified method to plan your website design” is the part of text that is modified from “we outline a simple method that you can follow to plan out your sites content and design” making these two web pages duplicates of each other

Experimental Results for Performance Measure

Table 1: A Table Showing Experimental Results

	No. of web pages retrieved	**Time elapsed**
Normal crawler	102 web pages	0.079 seconds
Crawler with ontology	76 web pages	0.031 seconds
Crawler with near-duplicate detection	42 web pages	0.009 seconds
Crawler with ontology and near-duplicate detection	38 web pages	0.005 seconds

The number of web pages and the time elapsed keep reducing when the crawler uses ontology as with near-duplicate detection and with both.

An Ontology Web Crawler with a Near-Duplicate Detection System

The proposed framework was adopted from two frameworks; ontology based web crawler and Near-Duplicate Web Page Detection by Enhanced TDW and simHash Technique.

The feature borrowed from the ontology web crawler was the use of ontology. In our case the domain of the ontology was “web design”. The featured borrowed from the near-duplicate detection system was the use of TDW and the simHash technique.

Data Description

The sample size was 102 web pages that were in a test website. The web pages included a combination of relevant, irrelevant and duplicate web pages. The focused domain was on “web design”. The criteria used to test performance were time and number of web pages retrieved.

Difference between Proposed Framework and Others

Similar approaches have been used to improve the performance of a web crawler and they have one of two things in common. One is the retrieval of relevant web pages and the other is they remove duplicate and near-duplicate web pages. The approaches that retrieved relevant web pages, did not take into account that the retrieved relevant documents could be duplicated or near-duplicated of each other. The approaches that removes redundant data did not retrieve relevant searches. The proposed approach has combined the use of ontology with a near-duplicate detection system that retrieved relevant information according to the search query of the user and also removes redundant information from the search.

Discussion

After evaluating the performance of our proposed approach, it was observed it performed better and faster than a normal crawler. When it imitated the behavior of a normal crawler, it indexed and crawled all 102 web pages including irrelevant and duplicate web pages. Irrelevant web pages included pages about food and travelling around the world. The time elapsed when crawling with a normal crawler was 0.079 seconds. When ontology was used, 76 web pages were retrieved and the time elapsed was 0.031 seconds. When near duplicate detection was used, 42 web pages were retrieved and the time elapsed was 0.009 seconds. When both ontology and near-duplicate detection were used, 38 web pages were retrieved and the time elapsed was 0.005 seconds. The number of web pages and time elapsed reduced when ontology was used. The same effect was seen when duplicate detection was used. The number of web pages reduced even further when both ontology and duplicate detection were used.

SUMMARY, CONCLUSION AND AREAS OF FURTHER RESEARCH

Summary

An ontology web crawler with a near-duplicate detection system will take web search to a whole new level. The proposed approach takes less execution time to search the web than other web crawlers. This is due to the fact that web documents are being filtered by the ontology web crawler such that only relevant web documents are retrieved according to the search query of the user. The relevant documents are further filtered by a near-duplicate detection system by removing web pages that are duplicates of each other and also remove near-duplicate web documents. This further reduces the number of web pages retrieved by the web crawler. This process takes less execution time to search the web because it focuses on crawling web pages that are relevant to a given topic ontology and non-duplicate data. The proposed model will save on storage space because of the reduced number of web pages retrieved as it takes care of irrelevant and redundant web pages searched. The proposed model will also improve page ranking because the search results will match the search query of the user because all the irrelevant and redundant web pages will have been discarded and removed from the search results. The time spent by the user searching for relevant information was reduced greatly because there were fewer documents to choose from.

Conclusion

Due to every growing World Wide Web, web pages are constantly updated, deleted or changed. This poses a challenge on web crawlers as they are only able to download a fraction of the web pages on the World Wide Web. This means that even when web crawlers download pages they may be irrelevant and do not match the search query. The crawled results also include mirrored or near-duplicate web pages. The proposed web crawler will crawl relevant web pages that match the search query at the same time get rid of redundant data. This approach will reduce computational time, manage storage space, improve page ranking and reduce user's time in searching for relevant information.

Areas of Further Research

Despite the fact that the proposed model is efficient, it would need improvement in some areas. The ontology remains static; it can be improved to be dynamic by adding new relations that is the crawler should search for web pages related to the search even if they don't contain the keywords searched. Domains and concepts should be added when visiting new web pages. Standardization of weights needs to be done because as of now experts assign weights to terms according to the area of expertise and knowledge.

REFERENCES

Arun Pr, Sumesh Ms. (2015). Near-duplicate web page detection by enhanced TDW and simHash technique. *International Conference on Computing and Network Communications.*

Komal , Dr. Ashutosh Dixit. (2016). Design Issues in Web Crawlers and Review of Parallel Crawlers

Lawankar, A., & Mangrulkar, N. (2016, February). A review on techniques for optimizing web crawler results. In 2016 World Conference on Futuristic Trends in Research and Innovation for Social Welfare (Startup Conclave) (pp. 1-4). IEEE.

Pranav, A., & Chauhan, S. (2015). Efficient focused web crawling approach for search engine. International Journal of Computer Science and Mobile Computing, 4(5), 545-551.

Saini, A. K., & KumarKhurana, V. (2016, March). ICT based communication systems as enabler for technology transfer. In 2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom) (pp. 90-99). IEEE.

Sharma, G., Sharma, S., & Singla, H. (2016). Evolution of web crawler its challenges. Int J Comput Technol Appl, 9, 53-57.

Sharma, N., & Devi, V. S. (2016). An Efficient SmartCrawler for Harvesting Web Interfaces of a Two-Stage Crawler. i-Manager's Journal on Information Technology, 5(4), 20.

Suryawanshi, P., & Patil, D. V. (2015). An Overview of Approaches Used In Focused Crawlers. International Research Journal of Engineering and Technology (IRJET) Volume, 2.

Udapure, T. V., Kale, R. D., & Dharmik, R. C. (2014). Study of web crawler and its different types. IOSR Journal of Computer Engineering, 16(1), 01-05.