ADVANCED AND EFFECTIVE WEB SEARCHING APPROACHES USING SEGMENT BASED INDEXING TECHNIQUES

Nadeem Akram N¹, B Nithya², Dr. V. Ilango³

¹Department of MCA, New Horizon College of Engineering, Bangalore
²Senior Assistant Professor, Department of MCA, New Horizon College of Engineering, Bangalore
³Professor, Department of MCA, New Horizon College of Engineering, Bangalore

ABSTRACT:

The most important tool in this current generation is the web phenomena (World Wide Web) when we are requesting for a resource at any point of time. We often tend those resources that never relates to our request, not only it is unnecessary but also it creates a sense of confusion among the end-users where they request for something else but they end-up with resource delivered not serving the purpose at all. Since most-often we get access to web, it is important that if proper mechanisms are implemented, so we can come-up with an advanced version web accessing platform whose aim would be to serve the purpose i.e. “serve what is requested”. This paper will discuss on some advanced approaches for accessing the web through advanced web-query-system and also to optimize the web-crawling approach with advanced algorithms. Advanced and efficient web searching methods using segment based indexing techniques are depicted in this paper.

Keywords: Automated Web Crawler, Segment Based Indexing, Trap-Key Procedure, Query-In-Position Advanced Resource Display

[1] INTRODUCTION

Worldwide web is one of the most emerging concepts in the 21st century as most often as an end user we may require a lot of data that is spread out in a number of categories. Every user has a definite set of request for different kinds of resources that
may involve browsing, including various type of resources, tutorials, connecting people, transporting data, view results, customer feed-back and also future predictions using given set of parameters[4]. It always has been seen that we don’t get what we request rather the resource which is relative to our request is delivered to us that may not serve the purpose at all. So, it is very important that when we query a search engine, we should be careful on what words we enter in the query-box such that it delivers the required resource to us but not the relative resource [7].

Assume that we hit a Google query such as “machine” then different search results would be displayed ranging from:

- machine wiki
- machine by Ronald grad
- machine-The man eater
- machine liker
- machine learning and so on.

Here our intention is only to find “what is the purpose of machine”. When we look at the above query results we can figure-out that the result “machine wiki” was relative to what we requested. The rest of the query had no link with what we requested rather it literally redirected us to elsewhere from machine by Ronald grad to machine learning.

As we can see, this is the major problem of the existing search engines such as Google, and if we measure the time required to search a resource in the web via Google 1/3 of actual time is spent on classifying about whether the resulted resource has some connection with our requirements.

It is a serious problem wherein we are looking for a critical resource and we are redirected to some other resource especially with the same title but the content is diverse from what we request [5-9]. When this is the case then a query result about 1000000 in 0.5 seconds would be nothing as even one of them doesn’t provide the basic necessity of what is required rather they deliver what is available[7]. This paper is to discuss about some advanced searching techniques using optimized version of web crawlers and also on the order of how resource is listed out.

[2] HOW SEARCH ENGINE WORKS

A search Engine can be defined as a software program whose purpose is to accept the user request which is usually text and connect to the crawler which performs most of the work in processing a query [5]. It runs down to a visit to all the servers located in order to find relevant web pages that contain the links. Then it returns a set of web pages from a trillion of web pages.

Once the required resource is located by the crawler by means of an algorithm it is processed to find out the hierarchy on what order the matched pages are displayed [4]. The user then selects the required link based on the content of interest to it. To
better understand about what happens behind the scene during querying process, we should understand the working of the web-crawler.

[3] WEB CRAWLER’S APPROACH

A web crawler or a bot or also a spider is a sophisticated application program whose purpose is to accept the user query and find the matching pattern through-out the web universe starting from the URL of the system where the query is entered and leaping towards nearest server and then advancing through-out the web.

A web crawler typically crawls over the web and visits all the pages containing the entered query and it marks the pages visited by indexing them [3]. Indexing in a web crawler refers to storing/markng the page visited that enables it connect to it directly for the next ever entered query[6] contains the matched pattern.

Each time when a query is entered, the spider first checks whether the entered query has already been indexed or not. Then it sees to it if any updates are made to the page since last visit[9], if so then it is indexed i.e. the updated page is entered into index.

This process continues as far as every query is concerned and the spider keeps on expanding the web universe until it finds all the pages in the web containing the designated query word, as each visit is updated [6].

The architecture of web crawler depicts the flow of querying and search process. The web crawler works on this simple algorithm and even in less than a second it attempts to visit as many pages as it can. This is the traditional approach of web crawler, which is inefficient as it may consider even those pages which are out of interest to the end user.

The traditional algorithm of web crawler has the following drawbacks [6-9].

- It returns all the pages containing the text.

![Figure 1. Web Crawler Architecture](image-url)
90% of the returned pages may not be of interest to the end user.
Since the search is based on text, the text may have different meanings in different context.
It may only return the relative resource since it literally doesn’hui understand the needs of needs.

To address on all these issues, we can implement the web-crawler process using an advanced crawling approach.

[4] ADVANCED SEGMENT BASED INDEXING ALGORITHM FOR WEB CRAWLERS

The purpose of this algorithm is to over-come the drawbacks of the existing web-crawlers approach and also to provide an efficient means of web-crawler search mechanism and to ensure that when the end-user hits the search query he gets to see only the resources of his/her interest and the actual resources of what is requested (not the relative resources).

4.1 Segment Based Indexing Algorithm – Objectives

- To enable the Web crawler to link to the required pages only
- To provide the facility of Refine Query mechanism
- To filter the end-user query with optimal search words [8]
- To enable the End-user to understand how to input the right query to get access into actual resources
- To provide a segment based query approach with different levels
- To provide multiple query windows for acquiring right query [4]

4.2 How Segment Based Indexing Algorithm Works?

The segment-based indexing algorithm provides an efficient algorithmic procedure to the existing web crawlers approach. It works as follows.

1. When the user hits the query, the query is sent to the buffer.
2. It starts scanning the web-pages based on the entered text [9].
3. At the back-end the query is tuned to get a clear picture of what type of resource is required.
4. Here the query processing is broken into segments.
5. As the first query hits, it is refined to the place where the category of the query is determined.
6. Then another text-box is displayed to determine the type of resources required.
7. When this is entered, the tuned query is passed.
8. In some cases of complex query, a three-level hierarchical segment is displayed [8].
9. Now tuned query is given as an input to crawler.
10. The crawler will begin to index only those pages and it conforms to the given query in exact way [5].

The above process ensures that the crawler will link and load only the pages which will satisfy all the 2-level or 3-level search criteria. This ensures that the end-user will be presented with only what he requires, which will reduce the number of pages linked for per query request [6]. This process provides a better-view for serving what is being requested from the End-user.

Figure: 2. Flow of Control for Segment Based Indexing

Now the mechanism for the web crawler is being described, the issues which are a matter of concern are:

- Where the crawler will begin its searching when it is looking for web-pages?
- What would be the order of displaying the search results?

In order to address the first issue, we have the following process: The usual working of the search request when-ever a query is hit; it starts from the URL path itself to locate the requested resource. This follows the spiders’ path i.e. it starts scanning the nearest local server, if it finds the match then it indexes them, but even when it is found it continues to link to all the connected servers.
The above process continues until not all but the most of the resource that contains the web page is displayed [8]. Every time when the search is made, it sets onto all but the most of the resource that contains the web page is displayed.

![Figure: 3. Web Spider](image)

The drawbacks of the above process are:

- Multiple resources may be indexed with different id containing the identical resources.
- Whenever the page is indexed, there is no proper identifier for the page accepts keyword, so all the pages may be tagged with some relative name.
- There is no proper mechanism to associate each resource so that each has its own identity when locating the resources.
- There is no mechanism to classify the exact domain, perspective of the resource.
- A sub quest is to be maintained to categorize the resource which is not present in the current approach.
- Every web page must have a unique resource key to describe the exact resource.

To address these issues there is a need for an efficient Resource Identification Process (RIP).

[5] PROGRESSIVE RESOURCE LOCATOR USING TRAP-KEY APPROACH

This approach is intended to resolve the issues pertaining to resource identification mechanism. It provides an efficient way. Whenever a resource is indexed it is trapped with the approach in a way that the resource has its own unique independent identity. Hence every time a resource of this type is queried an exact resource is delivered to the end user based on this algorithm.

5.1 Trapkey Approach

The Trapkey Approach [7-9] works as follows:

- When the resource is indexed for the first time, a unique key is generated for this resource.
- When the key is generated, the crawler scans the page and associates a three-level category for the page.
• The first level associates to the **exact topic of the resource** i.e. actual content of the page. For example: “India and the kargil war”.
• The second level is a **directory host** that will contain the domain of the resource. i.e. **“India and wars”** for the example mentioned.
• The third level is a **container host** that will include the category of the resource i.e. **“wars”** for the above example.
• When a query is given at first time it is tuned using segment-indexed approach.
• Now the tuned query is given to the crawler where the above discussed algorithm them is employed.
• The crawler starts with bottom up approach and it first matches the query with level three criteria.
• If it finds a match, then it drags the query to level two.
• If the level two matches, then the crawler compares the query with level one.
• The crawler then **traps** all the actual pages containing that resource based on the search key.

The resource is indexed and the necessary resources are delivered to the end-user. The above process provides an efficient approach to categorize the resource based on the contents. It also enables the crawler to locate the exact resource and not the relative resource.

5.2 Page-Rank Algorithm

The second question to be answered is in what order does the matched resource is displayed. The recent trend employed in the page order of display is **Page-Rank Algorithm** by Larry Page. It is described as follows.

\[
PR(A) = (1-d) + d \left( \frac{PR(T1)}{C(T1)} + ... + \frac{PR(Tn)}{C(Tn)} \right)
\]

Where
• \(PR(A)\) is the PageRank of page A,
• \(PR(Ti)\) is the PageRank of pages Ti which link to page A,
• \(C(Ti)\) is the number of outbound links on page Ti and
• \(d\) is a damping factor which can be set between 0 and 1.

As we see that Page-Rank does not rank web sites as a whole, but is determined for each page individually. Further, the Page-Rank of page A is recursively defined by the Page-Rank’s of those pages which link to page A[7-8]. The Page-Rank of pages Ti which link to page A does not influence the Page-Rank of page A uniformly. Within the Page-Rank algorithm, the Page-Rank of a page T is always weighted by the number of
outbound links $C(T)$ on page $T$. This means that the more outbound links a page $T$ has, the less will page $A$ benefit from a link to it on page $T$.

5.3 Control Flow for Trap-Key Approach

This approach is an eminent approach but it ranks the page based on think number of in-bound and out-bound links, rather ordering them based on the content of the page for a given resource. So, whenever a request to resource is made the set of resulting pages will be displayed based on their popularity not on the actual content that it possesses. This has the following draw-backs.

- The most popular page will be displayed at the top not the page that is requested.
- Some local page site will never be highlighted though they have some fair contents.
- The end user most at least browse a hand-full of Pages to find the right content.
- More time will be wasted by visiting alternate sites.
- The top sites may sometimes redirect to some other pages which never serves the purpose [9].

With the objective of delivering right set of web-pages from a huge number of popular pages, it is necessary to have resource ordering from a different perspective for ensuring that not the popular but the right page is displayed on top for a given query.

Figure: 4. Trap-Key Approach
To obtain the objective which is mentioned, a strategically ordering algorithm is essential. This can be achieved by applying the query-in position approach.


The objective of this process is to locate the right page for the right request query such that the pages displayed will be in an order. So, the top-level page will contain the actual content and the page following that will contain the resources which are the next-best to the first entry. The workflow of the Query-in-position approach is given as follows.

- The given query is fine-tuned using the above 2 approaches.
- The crawler then begins the hunt for the requested pages.
- Since the requested query is a text and all the searching is done is based on the text the text may be located in any portion of the web page.
- The Query-in-position approach defines associate a priority with each page. The priority is based on where the text is located in the web page [8].

There are 3 cases to set the priority.

Case 1: If the query text is the title of the web page then set the priority of the page as 1.

Case 2: If the query text is the sub-topic of the web page set the priority of the page as 2.

Case 3: If the query text is located elsewhere in the web page i.e. in the paragraph then there are 2 possibilities.

- If the text is in the first half of the paragraph. Then set priority as 3.
- If the text is in bottom half of the paragraph, then set the priority as 4.

If there is a collision with respect to priority between 2 different pages the page with most close refection to the requested query will be favoured [7]. The above process is repeated as the crawler goes on indexing the page and the web pages will be displayed relative to their priority. This process ensures that the page with the close proximity to the required content will be favoured first [6].

The control flow of query-in-position approach is given below based on the cases mentioned earlier. The page levels will be displayed based on the priority assigned.
[7] CONCLUSION

This proposed method is intended to display the drawbacks of the existing web-crawler algorithm. This is also to define a new approach that could overcome the existing issues to represent different resource storage approaches and also to determine the order in which the web page is to be displayed. By implementing these methods and techniques the current approaches can be optimized to produce high throughput. By insuring on proper and accurate algorithms and their implementation we can have a search engine which delivers the relevant information which is required.
REFERENCES