Improving the Web Indexing Quality through A Website-Search Engine Coactions

Rehab F. Hassan
Computer Science Department
University of Technology
Baghdad- Iraq

Sara Hussain
Computer Science Department
University of Technology
Baghdad, Iraq
Email: suror {at} yahoo.com

Abstract—**Rapidly needing for indexing the contents of web pages to update the search engine's indexes in the case of any potential modifications of its contents, represents a time and resource consuming operation which finally affects a new content or a content modification to be easily reflected in the search engines results. This paper proposes an approach called Multilateral Web Indexing Model (MWIM), which aims to describe a solution to the previously mentioned issues by establishing a more tied collaboration between websites and search engines. The collaboration consists in exposing to search engines auto generated metadata about the structure of a website and the contents of each web page. The Document Object Module (DOM) tree has used as an efficient tool for representing the web page content, while the XML Path language (Xpath) has used to provides a powerful syntax to address specific elements of DOM tree and extract metadata from HTML web page. Consequently, the search engine does not need any more to retrieve the web page mark-up code and perform the data extraction because this stage is performed directly on the website's platform when a web page was created or modified. Where every site had reliable metadata along of files, the work of a search engine would be made a lot easier and less time-consuming and resources and bandwidth, where the percentage of savings is 66.49%.**

Keywords--Web Indexing; Metadata; Search Engine; Website: Sitemap.

I. INTRODUCTION

The World Wide Web (commonly known as the web or WWW) is an architectural framework for accessing information spread out over millions of machines all over the Internet [1, 2, 3]. The information is generated by multiple sources and is carefully organized in the form of files and web pages, which, when grouped together to form a single entity, become a website.

A website represents a centrally managed group of web pages, containing text, images and all types of multi-media files presented to the attention of the Internet users in an aesthetic and easily accessible way [4]. The web pages have poorly characterized in terms of both metadata and structure descriptions. Metadata is data that describes data, for web page it may provides a significant amount of information about the web without examining the content of it and serves a variety of purposes[5]. With information being shared worldwide, there was a need for individuals to find information in an orderly and efficient manner. Thus began the development of search engines [1]. Search engines are the primary tools people use to find information on the web. The appearance of any website on the search engine result pages is also related to some kind of indexing. As soon as your website is noticed by the search engine crawler, its contents are scanned and entered into their database of already scanned sites i.e. their index [6]. It will be so powerful and useful if all websites contain by itself its own metadata files that are accessible by any search engine where these metadata files are extracted and available for any search engine. i.e. it do not need to be extracted by each search engine.

II. WEBSITE

A website, indicated by the homepage URL, denotes a set of pages that form a complicated directed graph with ‘‘page” nodes and ‘‘link” edges. In general, a website can be regarded as a set of URLs sharing the same domain name (i.e. the website indicates a host that contains pages whose URLs share the same hostname) [7].

III. SITEMAP

A sitemap is a list of URLs of a website accessible to crawlers or users. XML Sitemaps are written only for web spiders .The work of these joint efforts is now under the

auspices of Sitemaps.org. XML Sitemap Protocol is an XML file that lists URLs for a site along with additional metadata about each URL (when it was last updated, how often it usually changes, and how important it is, relative to other URLs in the site) so that search engines can more intelligently crawl the site. The premise of using XML Sitemap Protocol was that it would help search engines index content faster while it providing ways to improve the existing crawling algorithms. Using XML Sitemap Protocol does not guarantee anything in terms of better page rankings. Furthermore, use of XML Sitemap Protocol is not mandatory for all sites. The XML Sitemap protocol format consists of XML tags as shown in figure (1) [8].

```
<?xml version='1.0' encoding='UTF-8'?>
<urlset
xmlns="http://www.sitemaps.org/schemas/sitemap/0.9">
   <url>
        <loc>http://mydomain.com/</loc>
       <lastmod>2010-01-01</lastmod>
       <changefreq>weekly</changefreq>
        <priority>0.5</priority>
   </url>
</urlset>
```

Figure(1) Example of XML Sitemap

IV. META DATA

The term metadata, purportedly first used in 1969 is often called 'data about data' [5] or 'information about information'. The term 'meta' derives from the Greek word denoting a nature of a 'higher order' or more 'fundamental kind', or 'above', 'beyond', and 'of something in a different context' [9]. A metadata record consists of a number of pre-defined elements representing specific attributes of a resource, and each element can have one or more values. These elements could be the content, quality, condition, origin, and other characteristics of data or other pieces of information which presents the best way to share information about the data without having to provide the actual data. In other words, metadata is structured information that describes, explains, locates, or otherwise makes it easier to retrieve, use, or manage an information resource also states that metadata is the key to ensuring that (digital) resources will survive and continue to be accessible into the future[10]. Metadata standards for describing Internet resources have appeared (i.e. representation of Web-related metadata), such as meta tags, the Dublin Core Metadata Element Set (DCMES), and the Resource Description Framework (RDF) [11].

V. SEARCH ENGINE

Web Search Engine is a software program that collects data taken from the content of files available on the Web and puts them in an index or database that Web users can search in a variety of ways [5].The typical design of search engines is a "cascade", in which a Web crawler creates a collection which is indexed and searched. Most of the designs of search engines consider the Web crawler as just a first stage in Web search, with little feedback from the ranking algorithms to the crawling process. This is a cascade model, in which operations are executed in strict order: first crawling, then indexing, and then searching [12].

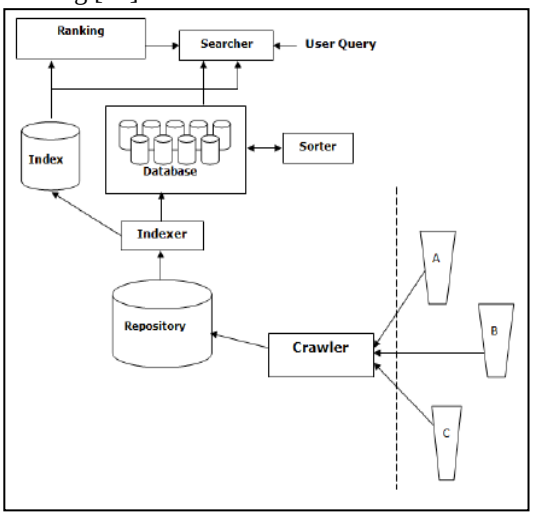

Figure (2) The Structure of Search Engine

VI. WEB DATA EXTRACTION

Web Data Extraction systems are a broad class of software applications targeting at extracting information from Web sources like Web pages: for instance, if the source is a HTML web page, the extracted information could consist of elements in the page as well as the full-text of the page itself. Eventually, extracted data might be post-processed, converted in the most convenient structured format and stored for further usage [13]. Web Data Extraction is closely related to web indexing, which indexes information on the web using a web crawler and is a universal technique adopted by most search engines [14]. The representation of a web page by using a labeled ordered rooted tree is usually referred as DOM. The general idea behind the Document Object Model is that HTML web pages are represented by means of plain text, which contains HTML tags, so as free-text. HTML tags may be nested one into another, forming a hierarchical structure. This hierarchy is captured in the DOM by the document tree, whose nodes represent HTML tags. The document tree (DOM tree) has been successfully exploited for Web Data Extraction purposes in a number of techniques, such as addressing elements in the document tree: XPath. The XML Path Language (XPath) provides with a powerful syntax to address specific elements of a DOM Tree in a simple manner [13].

VII. PROPOSED MWIM APPROACH

This paper proposes a different approach have been called **Multilateral Web Indexing Model(MWIM)** for a search engine to retrieve the relevant data from a website by

establishing a tied collaboration between the search engine and the websites served by it. The proposed solution a chive by construction new structured information for the web-related metadata and new extension of XML sitemap. By making the website have the ability to auto-generating metadata that describing the website contents and structure that needed by the search engine to index the website in its database, which give the website a faster reflection in the search results. This is done by a server-side component called **Search Engine Agent**, which generating metadata for a web pages content at the request of the website's back end. Figure(3) illustrate an overview of the data path in the proposed approach and all the involved processes in a MWIM. **MWIM** characterized by:

- Distributed, where the indexing is distributed on many website servers instead of search engine's one;
- Multilateral, where the website and the search engine are collaborate to improve the indexing process;
- Full Text Index;
- Containing XML Sitemap with the same properties and new elements;
- Standard Stemmer for English language and stop word list including 1300 word.

VIII. THE GENERATED METADATA TYPES

MWIM approach is designed to generate different metadata files for each website where the CE file will be generated for each web page, while a single sitemap file will be generate for the whole website.

1. Content Extract Metadata

Basically the proposed Content Extract binary file can be considered as a forward index of the document's content to which. The structure of the whole information included in this file can be classified as three main parts (Document properties, Textual content and Non-textual content), as presented in figure (4).

A. Document Properties

Consists of the information that describes the web page document to which the generate Content Extract belongs to. This information is organized as a property list called Document Properties. This property list represented as a pairs of particular information called property value, and a unique name called property key which is used to identify each property. The basic set of properties is title, url, base-url, author, description, keywords, retrieval-date and language.

Figure (3) an overview of the data path and the involved processes in a MWIM

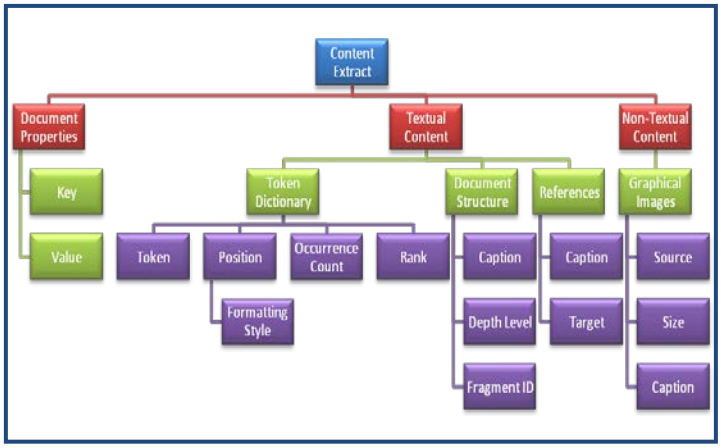

Figure (4) an overview of the information contained in CE

B. Textual Content

This part contains the information that describes the textual content of the document. This is divided in to three categories:

- **Token Dictionary**: a tokenized representation of the document text along with additional information (Position, Occurrence Count, Rank and Formatting styles). The tokens

of a Token Dictionary are pre-filtered to eliminate the stop-words, to improve the compactness of the Content Extract file.

- **Document Structure**: the index of a document as defined by its headings.
- **References:** a list of references (links) to the same or other document(s).

C. Non-Textual Content

Contain information which describes non-textual elements. Currently, only the Graphical Image List are considered.

2. **The Proposed XML Sitemap**

The proposed Sitemap represents an extended version of the original XML Sitemaps 0.9 protocol introduced by Google. This extension provides support for exposing information regarding to the Content Extract corresponding to a Sitemap entry such as: the location, last update time and the hash code of the source document. The extension is achieved by allowing new elements belonging to the XML namespace "urn:mwimc/content-extract-manifest" to be descendants of the "url" element defined in the original protocol. The new elements added to the proposed XML Sitemap are:

- **"loc" element:** The location of the Content Extract file is required for the search engine to be able to retrieve the corresponding Content Extract file of a Sitemap entry.
- **"lastupd" element:** The search engine identifies the updated Content Extract files by comparing the actual modification date and time of each Sitemap Entry with the corresponding values from the cached version of the Sitemap.
- **"fingerprint" element:** For a client side tool which automatically generates the metadata for static websites, the MD5 hash code of the source files can be specified to allow the identification of the web pages which were modified from the previous update procedure.

Figure (5) show an exemplification of a XML Sitemap extended with Content Extract Manifest:

```
<?xml version="1.0" encoding="utf-8"?>
<urlset
xmlns="http://www.sitemaps.org/schemas/sitemap/0.9"
xmlns:ce="urn:mwimc/content-extract-manifest">
<url>
<loc>
http://www.samplesite.com/index
</loc>
<ce:loc>
    http://www.samplesite.com/index.ce
</ce:loc>
<ce:lastupd>
    2013-7-29T16:18:12
</ce:lastupd>
<ce:fingerprint>
    721F46B7E21B49D47CCEFAF6BBF4729D320A3AF5
</ce:fingerprint>
</url>
</urlset>
```

Figure (5) Example of Proposed Sitemap

IX. PROTO-MWIM SDK

The MWIM Software Development Toolkit (SDK) is a set of development tools and libraries to implement the Proto-MWIM that includes:

1. Search Engine Agent

The Search Engine Agent (SEA) prototype consists of a set of component libraries built on .NET Framework 4. The routine which interoperates with these libraries should be resided in the back-end of a website built on ASP.NET technology, or an client side application and it could be called the "caller" routine. SEA features a modular architecture, to facilitate eventual extensions of its functionality, currently containing three modules:

A. The HTML Scraper Module

This module retrieves Content Extract by scraping a HTML markup code. The scraping procedure represents the parsing of the document. This is achieved by HTML Agility Pack, an open-source HTML parser (.Net library) which scans the HTML page and provides a DOM Tree which then can be used to fetch required values from the page's content by using

Scraping Algorithm

Input	DOM Tree
Output	Document Properties, Textual content except token

Begin

```
Titlement ← /html/head/title
Metaelement ← /html/head/meta[@name != '' and @content != '']
Linkelement ← "//a[@href !="
headelement ← "//h1|//h2|//h3|//h4|//h5|//h6"
imgelement ← "//img[@src != '']"
   for each Node(element)in DOM Tree DO
    if element= Titlement or Metaelement or Linkelement or
headelement or imgelement then
       CE file ← element(content)
    end if
   end for
End
```

XPath query as illustrate in scraping algorithm.

Then, a depth-first tree traversal algorithm is used for traversal DOM tree to collect all text in document with its rank and formatting style as present in HTML DOM Traversal Algorithm then send them to next process (tokenization).

HTML DOM Traversal Algorithm

Input Parameters	
Element	The HTML node instance to process. Its descendant elements will be also processed.
Style	The formatting style of the parent of the specified node. Initially the value is empty.
Rank	The rank level of the parent node. Initially the value is empty.
Output Parameters	
Text, Rank, Style	Collect the Text from all elements with its rank and style

```
Begin
   newStyle←style
   ! check the kind of the current node
      if isTextElement(element) then
         tokenize(getElementText(element), style, rank)

      ! Collect rank value:
      else if
         getElementName(element) = "h1", "h2", "h3", "h4",
                        "h5", "h6" then  newRank ← 5
      else if getElementName(element) = "a" then newRank ←
6
       .
       .
       .
      ! Collect style information:
      else if getElementName(element) = "b", "strong" then
         newStyle.bold = true
      else if getElementName(element) = "i", "em" then
         newStyle.italic = true
       .
       .
       .
  end if
  ! Call recursively the same procedure for the children
elements
  ! of the current element.
  for allchild in element
    recall HTML DOM Traversal(child, newStyle, max(rank,
newRank))
  end for
End
```

B. The Core Module

The Core Module represents the base component library of the Search Engine Agent which exposes the following functionality to the caller:

- Provides access to the Content Extract information, either by the deserialization of a previously generated file or by creating a blank Content Extract instance.
- Provides text tokenization services, including stop-word removal, stemming, and aggregation of the results in the token dictionary.
- Performs serialization.

The Text Tokenization represents the most important functions of a search engine indexer which used for keeping information about each word occurred in an individual web page. It consists of the three processes (Tokenizer, Token Filter and Aggregator):

A. The Tokenizer: is a component that splits the supplied text chunk into tokens as present in the Tokenizer Algorithm.

B. Token Filtering: The Token Filtering can be performed by a separate module or by a custom Token Filtering implementation in the Search Engine Agent caller, since the Core Module exposes an interface to allow full control over this procedure. Additionally, a built-in Token Filter called Configurable Token Filter supports defining the filtering process via XML-based configuration file. The filtering parameters specified in this configuration file consist of token length limits, non-alpha characters allowed in its body, stop-words and stemming. To Adding Support for New Natural Languages, new Token Filter Configuration files may be added to the "Filters" sub-directory from the application's directory. MWIM employs the Porter stemming algorithm to improve system accuracy by reducing the large number of words morphological variants and list of (1300) stop words is used that have no meaning or irrelevant.

C. Aggregator: This process aggregates tokens in Content Extract's Token Dictionary along of its associated information (rank, formatting style and position).

The Tokenizer Algorithm	
Input parameters	*Text, style, rank*
Output Parameter	List of Tokens

```
Begin
    token ← incompleteToken
    for index = 0 to length(text) - 1
      char ← text(index)
      if letter(char)  then token += char
      if isDigit(char) and length(token) = 0 then discard(char)
       else
         token += char
       endif
      if end_word_symbol (char) then
         move to next token
                aggregate(token, position, style, rank)
        position += 1
      endif
endfor
    if char(length(text)-1)<> end_word_symbol then
          incompleteToken← LastToken
    else      incompleteToken← null
End.
```

C. The Reporting Module

The Reporting Module exports the information contained in Content Extract binary file in a human-readable form, trough a stand-alone HTML document.

2. The Proto-MWIM Toolkit

Is a software application which bundles a series of MWIM related tools for user to applied SEA functions. It use a parallel computing for performing tasks concurrently, to take advantage of all logical CPUs available on a machine. This is improving the performance of this tool by reducing the time required for the finalization of the procedure.

X. MWIM EXPERIMENTATION AND EVALUATION

MWIM experiment consists of collecting statistics report resulted from scraping 22.602 different HTML document samples using Proto-MWIM Toolkit. Each record from the Statistics Table contains the following fields (URL, Original Size, CE size ,Parsing Time ,Processing Time ,Overall Time ,CE serialization Time, CE Deserialization Time). This test will focus to evidence two of the key benefits which contribute on a large scale to the overall efficiency of the concept:

1. The network traffic savings.
2. Less computational power required by the search engine servers.

The amount of savings will be established by deterring the difference between the size of the markup code and the size of Content Extract metadata file for the test website.

Considering $\mathbf{S_M}$ as the sum of the "Original Size" fields of all records from the Statistics Table resulted from this experiment, respectively $\mathbf{S_C}$ as the sum of the "CE Size" fields of the same records, the amount of savings will be: $\mathbf{\Delta_S=S_M-S_C}$. The values of SM and SC have been shown to be of **306,773,324** bytes (**292.6 MB**) respectively **102,806,177** bytes (**98** MB). According to these values, $\mathbf{\Delta_S}$ become **203,967,147** (**194.5** MB). Thus, for a full indexing of the test website, the search engine would have retrieved 194.5 MB less in the MWIM approach than using the traditional method. The percentage of savings determined using: **(100/Sm)*Sc**, resulted to be of **66.49%**, the average size of a Content Extract metadata file being of one third of the original HTML markup size.

The average time spent for the Document Parsing Stage ($\mathbf{T_P}$), Information Collection Stage ($\mathbf{T_{IC}}$), and Content Extract Serialization Stage ($\mathbf{T_S}$) were established by determining the average value of the "Parsing Time", "Processing Time" respectively "CE Serialization Time" fields of all Statistics Table records. Their values shown to be of: **5.02** ms for $\mathbf{T_P}$, **194.78** ms for $\mathbf{T_{IC}}$, respectively **150.97** ms for $\mathbf{T_S}$. The average time spent for all three stages was of **350.77** ms per document. Because the value of $\mathbf{T_{IC}}$ was far greater than initially expected, the experiment was re-ran without using a token filter, becoming **12.94** ms. It seems that a proportion of **97.9%** of the duration of this stage was spent for the natural language processing tasks. A faster dictionary lookup algorithm, alternative to the intrinsic .NET Framework's one and a memorization technique for the stemming procedure may be considered as an eventual future work for improving the performance of the current MWIM implementation.

On the search engine servers, this approach replaces the web crawling and natural language processing stages with the Content Extract Deserialization stage. The **average time** spent in this stage ($\mathbf{T_D}$) was of **5.47** ms, being established by calculating the average value of the "CE Deserialization" field from all records of the Statistics Table.

XI. CONCLUSION

Several conclusions can be considered here to establish the main properties, advantages and disadvantages of the proposed approach:

- **Less bandwidth consumption**: Based on a binary format optimized for size compactness and containing only the data extraction results, the CE files are smaller in size than the original HTML code. Due to these two aspects drastic savings for the network traffic are made.
- **Less computing power consumption**: As the web indexing computational requirements are partially distributed on the hardware of the web hosting servers, the computational power for indexing the MWIM enabled websites is significantly reduced.
- **Faster reflection of content modification in search results**: due to the reassigning the parsing and data extraction to the web site back ends, the routine of the search engine data extraction procedure resumes only in retrieving and aggregating the pre-processed (extracted) data. This causes faster data updating cycles for index in search engine.
- **More accurate search results**: Due to the fact that the content is submitted selectively to the search engines, it can be filtered by the websites of any irrelevant portions and keeping only the portions which relates to the main subject.
- **Describable Content** - Since the Content Extracts are metadata containers, which provides additional information and function of the elements can be stored also. As an example, a search engine may be aware that the content of a web page describes a software product, being able to identify its name, version, license type, and size. This may bring the opportunity to perform categorized searches.

XII. SUGGESTIONS FOR FUTURE WORK

For this concept to work properly, it is necessary to establish implemented in the future to make the project optimal:

- Establish a service called **Multilateral Web Indexing Client Dispatcher** (**MWICD**) to an up-to-date indexing updating process in search engine. Which provides a list of all MWIM enabled websites and broadcasting the content

modification notifications from the Search Engine Agents to the search engines servers. Once the search engine receives a content modification notification, it retrieves the Sitemap of that website and compares it to the cached version to determine which web pages, documents or multimedia resources were added, modified or removed in order to re-index these elements by retrieving and further processing the data contained in the Content Extract files. Another function of MWICD is to host Search Engine Agent updates, which needs to auto-update in order to quickly deliver security updates or eventual new futures.

- Generated metadata for a broad variety of source file formats such as (.pdf, .doc, .ppt, .xsl, ..etc) and multimedia resources contained by each web page. Also, for another languages such as Arabic Language.
- Additional website information regarding to the taxonomy of their contents, representing another enhancement factor for categorized searches and also for aggregators.

REFERENCES

[1] Liu B.,Web Data Mining, Springer-Verlag, 2007.

[2] Rana R. K., Tyagi N., A Novel Architecture of Ontology-based Semantic Web Crawler, International Journal of Computer Applications (0975 – 8887), Volume 44– No18, April 2012.

[3] Behrouz A. Forouzan, Data Communications and Networking, Fourth Edition, McGraw-Hil, 2007.

[4] Estiévenart F., François A., Henrard J. and Hainaut J., A tool-supported method to extract data and schema from web sites, Website evolution, Fifth IEEE international workshop, 2003.

[5] Baca M., Introduction to Metadata, Second Edition, J. Paul Getty Trust, 2008.

[6] Mishra A. A., Kamat CH., Migration of Search Engine Process into the Cloud, International Journal of Computer Applications,Vol.19, No.1, April- 2011.

[7] Lin SH., Chu K., Chiu CH., Automatic sitemaps generation: Exploring website structures using block extraction and hyperlink analysis, Expert Systems with Applications,38, pages 3944–3958,Elsevier Ltd, 2011.

[8] Jerkovic J. I., SEO Warrior, O'Reilly Media, Inc.,2010.

[9] Litwin L., Ross M., Geoinformation Metadata in INSPIRE and SDI: Understanding. Editing. Publishing, Springer, 2011.

[10] Olfat H., Automatic Spatial Metadata Updating and Enrichment, PH.D Thesis , Department of Infrastructure Engineering, School of Engineering, The University of Melbourne, Victoria, Australia, January-2013.

[11] Chang CH., Kayed M., Girgis M.R., Shaalan K., A Survey of Web Information Extraction Systems, IEEE Transactions on Knowledge and Data Engineering, 2006.

[12] Zou J., Le D., Thoma G.R., Combining DOM Tree and Geometric Layout Analysis for Online Medical Journal Article Segmentation, ACM, Chapel Hill, North Carolina, USA, 2006.

[13] Ferrara E., Demeo P., Fiumara G., Baumgartner R., Web Data Extraction, Applications and Techniques: A Survey, ACM Computing Surveys, Pages 1-54, July 2012.

[14] Devika K., Surendran S., An Overview of Web Data Extraction Techniques, International Journal of Scientific Engineering