Automatic Semantic Header Generator

Bipin C. DESAI[*], Sami S. HADDAD, Abdelbaset ALI.
Department of Computer Science,
Concordia University,
Montreal, H3G 1M8, CANADA

February 28, 2000

Abstract

As the mounds of information and the number of Internet users grow, the problem of indexing and retrieving of electronic information resources becomes more critical. The existing search systems tend to generate misses and false hits due to the fact that they attempt to match the specified search terms without proper context in the target information resource. In environments that contain many different types of data, content indexing requires type-specific processing to extract indexing information effectively. The COncordia INdexing and DIscovery (Cindi) system is a system devised to support the registration of indexing meta-data for information resources and provide a convenient system for search and discovery. The Semantic Header, containing the semantic contents of information resources stored in the Cindi system, provides a useful tool to facilitate the searching for documents based on a number of commonly used criteria. This paper presents an automatic tool for the extraction and storage of some of the meta-information in a Semantic Header and the classification scheme used for generating the subject headings.

1 Introduction

Rapid growth in data volume, user base and data diversity render Internet-accessible information increasingly difficult to use effectively. The number of information sources, both public and private, available on the Internet are increasing almost exponentially. They include text, computer programs, books, electronic journals, newspapers, organisational, local and national directories of various types, sound and voice recordings, images, video clips, scientific data, and private information services such as price lists and quotations, databases of products and services, and speciality newsletters [7]. There is a need for an automated search system that allows easy search for and access to relevant resources available on the Internet which in turn requires proper indexing of the available information. The semantics of the resource are exploited in the current system to extract and summarise the relevant meta-information(Semantic Headers [6, 8]) to support its discovery. Specialised databases

[*] Author for communication: bcdesai@ideas.concordia.ca

maintain archives of these Semantic Headers(SH) which could be searched by another component of Cindi which features cooperating distributed expert systems and helps users in locating pertinent documents.

The Cindi system provides mechanisms to register, search and manage the SHs, with the help of easy to use graphical user interfaces. Cindi avoids problems caused by differences in semantics and representation as well as incomplete and incorrect data cataloguing by using a standardized subject heading hierarchy. This meta-information could be entered by the primary resource provider with the help of an Automatic Semantic Header Generator (ASHG) described in this paper. ASHG is a software that assists the authors of documents to semi-automatically generate many of the fields of the SH and hence assist them in the registration of their documents in the Cindi system. One of the main tasks of ASHG is to classify a document under a list of subject headings as described herein. As the author is required to verify and complete the ASHG generated Semantic Header entry, the potential for its accuracy is high.

The paper is organized as follows: in section 2, we introduce the Cindi system. An overview of information retrieval and the algorithms used are examined in section 3. Section 4 covers our approach to the building of the thesaurus used in ASHG system and section 5 describes its components. Following this, we give the results of our tests to generate the SH on a set of documents prepared in the HTML, LaTeX, RTF and plain text format and our conclusions.

2 The Cindi system

Attempts to provide easy search of relevant documents has lead to a number of systems [5, 13, 15, 19, 29, 32, 36, 37, 38]. However, the problem with many of these is that their selectivity of documents is often poor [7]. The chances of getting inappropriate documents and missing relevant information because of poor choice of search terms are great. Hence, there is a need for the development of a system which allows easy search for and access to resources available on the Internet. Using a standard index structure and building an expert system based bibliographic system using standardised control definitions and terms can alleviate the problem and provide fast, efficient and easy access to the Web documents. For cataloguing and searching, Cindi uses a meta-data description called SH[6, 8] to describe an information resource. The SH includes those elements that are most often used in the search for an information resource. Since the majority of searches begin with a title, name of the authors (70%), subject and sub-subject (50%) [14], Cindi requires the entry for these elements in the SH. Similarly, the abstract and annotations are relevant in deciding whether or not a resource is useful, so they are included too[7, 27]. The components of the SH are: Title, Alt-title, Language, Character Set, Keyword, Identifier, Date, Version, Classification, Coverage, System Requirements, Genre, Source and Reference, Cost, Abstract, Annotations and User ID, Password.

Preparing the primary source's SH requires identifying it as to its subject, title, author, keywords, abstract, etc. These problems are addressed by Cindi, which provides a mechanism to register, manage and search the bibliographic information.

The overall Cindi system uses knowledge bases and expert sub-systems to help the user in the registering and search processes. The index generation and maintenance subsystem uses Cindi's thesaurus to help the provider of the resource select the most-appropriate standard terms for items such as subject, sub-subject and keywords. Similarly, another expert sub-system is used to help the user in the search for appropriate information resources [6].

The SH information entered by the provider of the resource using a graphical interface is relayed from the user's workstation by a client process to the database server process at one of the nodes of a distributed database system (SHDDB). The node is chosen based on its proximity to the workstation or on the subject of the index record. From the point of view of the users of the system, the underlying database may be considered to be a monolithic system. In reality, it would be distributed and replicated allowing for reliable and failure-tolerant operations. The interface hides the distributed and replicated nature of the database. On receipt of the information, the server verifies the correctness and authenticity of the information and on finding everything in order, sends an acknowledgment to the client. The server node is responsible for locating the partitions of the SHDDB where the entry should be stored and forwards the replicated information to appropriate nodes. The various sites of the database work in a cooperating mode to maintain consistency of the replicated portion. The replicated nature of the database also ensures distribution of load and ensures continued access to the bibliography when one or more sites are temporarily nonfunctional.

Cindi search sub-system guides the user in entering the various search items in a graphical interface similar to the one used by the index entry system. Once the user has entered a search request, the client process communicates with the nearest SHDDB catalogue to determine the appropriate site of the SHDDB database. Subsequently, the client process communicates with this database and retrieves one or more SHs. The result of the query could then be collected and sent to the user's workstation. The contents of these headers are displayed, on demand, to the user who may decide to access one or more of the actual resources.

3 Information Retrieval and ASHG

Information Retrieval (IR) is concerned with the representation, storage, organisation and accessing of information. Indexing is the basis for retrieving documents that are relevant to the user's need [16]. The main concern in IR is how to select significant words and phrases from a document that best describe it [11, 17]. Luhn[17] used frequency counts of words in the document text to determine which words were sufficiently significant to represent the document. The use of statistical information about distributions of words in documents was further exploited by Maron and Kuhn [18] and Stiles [28] who obtained statistical associations between keywords.

Automatic summarisation of full documents generates a condensed version of the document and generates coherent output[4]. Luhn[17] assumes that frequency data can be used in extracting words and sentences that represent a document. The document's representation used by Rijsbergen [22] consisted simply of a list of class names, each name representing a class of words occurring in the total input text.

The above approaches use keyword searches and statistical techniques to retrieve relevant documents (e.g., [2, 12, 24, 30]). Statistical techniques take advantage of large document collections to automatically identify words that are useful indexing terms. However, word-based techniques have several limitations due to synonyms, polysemys and anaphoras. Furthermore, most keywords are believed to be ambiguous and are often poorly represented by small collections of individual terms [25]. It is therefore widely believed that the keyword approach is not adequate for text content representation in information retrieval. By extension, the identification of text content by weighted term sets may also be unacceptable[3].

Since the frequency criteria are not very reliable, additional criteria should be used such as contextual inference (the word location or the presence of cue words), and syntactic coherence criteria [1, 9, 10, 11, 17, 20, 21, 35].

The available experimental evidence indicates that the use of abstracts in addition to titles brings substantial advantages in retrieval effectiveness[23]. This is one of the main reasons why the abstract is included in the SH. Building an accurate representation of a document, which would increase precision, is one of Cindi's main concerns. Our approach integrates the features of these systems as described below. Since some of the available experimental evidence indicates that the use of abstracts in addition to titles brings substantial advantages in retrieval effectiveness [23], we assign high weights to the terms located in the abstract and title fields. In addition to assigning term weights, our system used the term frequency of occurrence addressed by Luhn.

Our system looks for a match between a set of different weighted terms generated from the document and a set of controlled terms. The highest weighted subject headings associated with the matched controlled terms will be selected. Luhn's automatic abstracting idea is used in generating an abstract for a document in case one is not found and our file recognition system borrows from the one used by Harvest.

4 ASHG's Thesarus

ACM[34], INSPEC[31] and Library of Congress Subject Headings (LCSH)[33] were the main building blocks of Cindi's three level Subject Hierarchy which currently is limited to the domains of Computer Science and Electrical Engineering. ASHG's computer science subject hierarchy uses ACM's subject hierarchy as the starting point, and electrical engineering subject hierarchy is based on that of INSPEC's. We have exploited LCSH's subject headings relations to refine both hierarchies. LCSH contained relations between subject headings such as BT (Broader Term), NT (Narrow Term), UF (Used For), and RT (Related To). In order to augment ACM and INSPEC subject hierarchies, a search for an ACM or INSPEC subject heading was made in LCSH. If a match was found, the narrow terms found in LCSH under the matched subject were added to the list of subjects or terms under the ACM or INSPEC's matched subject heading. This augmentation produced a hierarchy composed of five or six levels. Since Cindi's subject hierarchy was limited to only three levels, the following rules illustrated in Figure 1, were applied to merge these subject headings. The (*Level_0*) subject is *Computer Science* or Electrical Engineering. Some of the subject headings found in the *Level_1* and *Level_2* augmented subject hierarchies were concatenated to form the Cindi's *Level_1* subject heading. The same rule was applied on subject headings at *Level_3* and *Level_4* to yield Cindi's *Level_2* subject heading. The *Level_5* and *Level_6* subjects were used as controlled terms associated with Cindi's *Level_2* subject headings.

The resulting subject hierarchy has three levels and a set of control terms associated with the lowest level subject headings.

The reason behind the Control Term Subject association is to extract or classify the primary source under a number of subject headings by comparing the significant list of words contained in the document with the list of control terms. An association between the control terms and their corresponding subject headings is created.

Each control term has three lists of subject headings attached to it. The control terms

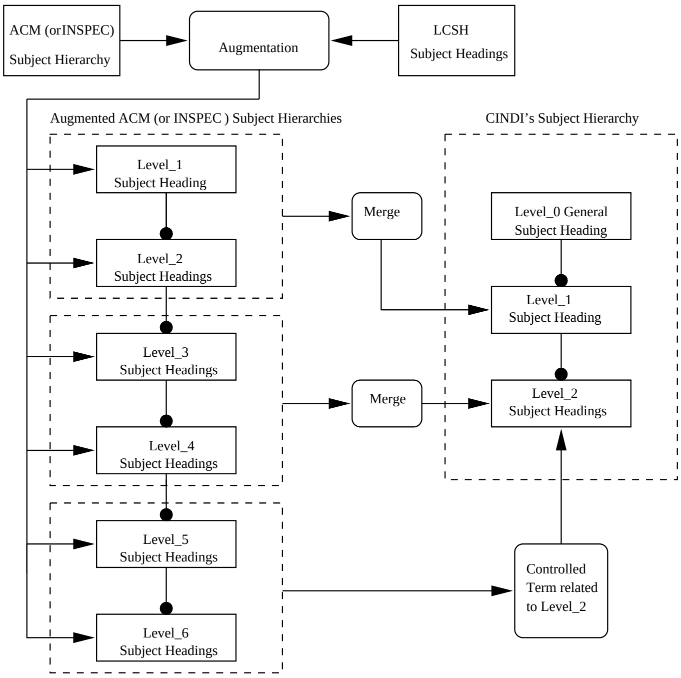

Figure 1: Transforming ACM (or INSPEC) Subject Hierarchy into Cindi's Subject Hierarchy

are based on the terms found in ASHG's subject hierarchy and the additional terms that are associated with *Level_2* subject headings. For each subject heading and the additional controlled terms, we use their constituent English none noise words as their corresponding control terms. For example, the control term *compute* will be associated with *Computer Science* general subject heading. Similarly, the control term *hardware* will be associated with *Hardware integrated circuits* and *Hardware performance and reliability* level_1 subject headings and *Hardware Simulation Design Aids* level_2 subject heading. Each controlled term is associated with one or more subject headings.

Mapping ASHG's subject heading terms into control terms involves: removing noise (stop) words; stemming the remaining words to find the the root and associating the root with the corresponding subject heading.

5 ASHG Implementation

In this section, we present the implementation details of the Automatic Semantic Header Generator (ASHG) of the Cindi system. This is an important step in providing the author of a document a draft SH with an initial set of subject classifications and a number of components of the SH for the document. The ASHG scheme takes into account both the occurrence frequency and positional weight of keywords found in the document. Based on the document's keywords, ASHG assigns a list of subject headings by matching those keywords with the controlled terms found in the controlled term subject association. The ASHG also extracts some of the meta-information from a document such as title, abstract, keywords, dates, author, author's information, size and type. The major steps followed by ASHG consist of:

1. *Document Type Recognition*: In order to apply the correct ASHG to a document, the type of the document has to be recognised. The system currently understands HyperText Markup Language (HTML), Latex, RTF and plain text documents.

2. *Applying ASHG's Extractor*: The summariser corresponding to the type of document is applied to the input document.

3. *ASHG's Document Classification*: The document is assigned subject headings. It involves:

   (a) *Word stemming*: The system applies the stemming process described below, to map the words found in the extracted fields onto a base root word.

   (b) *A Look up into the Controlled Term Subject dictionary.*

4. *SH Validation*: The generated SH is presented to the user for modification and validation

ASHG uses the syntax of documents in HTML, LaTex, RTF or text to extract the document's meta-information. ASHG extracts summary information, such as the title, keywords, dates of creation, author, author's information, abstract and size. In tagged documents, the author might explicitly tag some of the fields to be extracted. In case these fields are not explicitly tagged, ASHG attempts to extract them using heuristics. However, if the explicit keywords were not found in the document, then words found in the title, abstract and other tagged words would be used to extract an implicit list of keywords.

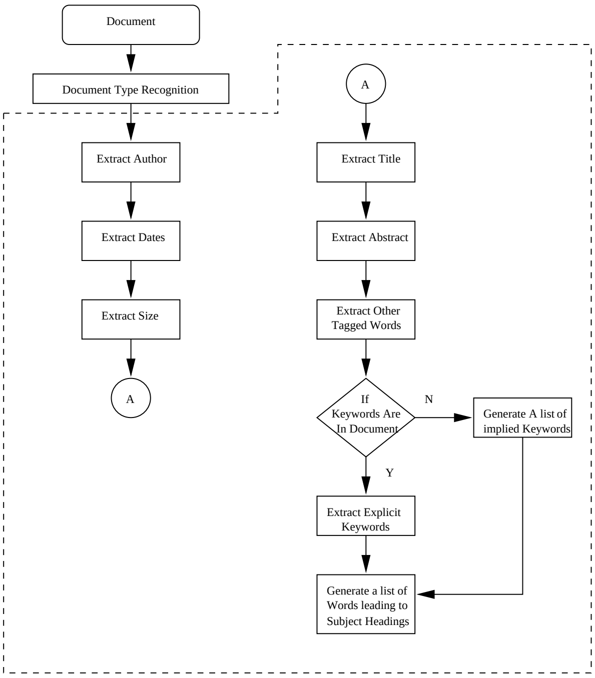
Document
Document Type Recognition
A
Extract Author
Extract Dates
Extract Size
A
Extract Title
Extract Abstract
Extract Other Tagged Words
If Keywords Are In Document
N
Generate A list of implied Keywords
Y
Extract Explicit Keywords
Generate a list of Words leading to Subject Headings

Figure 2: ASHG's extraction steps

Let us consider the method of extracting an abstract from a document. If it fails to find a tagged abstract, it applies an automatic abstracting method. This method, which is similar to Luhn's scheme, attempts to extract a section or paragraph that is headed by introduction. Based on the number of significant root words in the sentence, a numerical measure is developed for a sentence. The automatic abstracting would consider only sentences with the highest numerical measure. If this fails, the extractor extracts the first paragraph and applies the automatic abstracting method to it.

Perhaps one of the most challenging tasks in information extraction is to extract and manipulate information found in plain text documents. Since these documents do not contain tags or mark-up elements, the *TEXT_extractor* relies heavily on heuristics in extracting the title, explicitly stated keywords, author(s), dates (Created, Expiry), size of the file, and the abstract.

1. **Extracting the author from a plain text document:** The *TEXT_extractor* looks for a pattern such as *written by, edited by* or *revised by.* If it finds one of them, it extracts the text following it and stores it as the author's SH field.
2. **Extracting dates from a plain text document:** The *TEXT_extractor* uses the *stat* and *GM-time* commands on the document file to extract the date of creation.
3. **Extracting the size of the plain text document:** Using the *stat* unix command, the size of the file can be extracted.
4. **Extracting the title from a plain text document:** When presented with a plain text document, the *TEXT_extractor* extracts the first sentence from the document. This sentence is used as the document's title. If it fails, it generates a list of sentences by extracting all sentences found in the first, second and last paragraph and by extracting the first sentence of all other paragraphs. Each sentence is divided into its constituent words. After dropping all English Noise or Stop words, the remaining words are stemmed[39]. Each sentence is given a weight according to the frequencies occurrences' sum of the stemmed words found in the sentence. The *TEXT_extractor* selects the highest weighted sentence as the document's title.
5. **Extracting the abstract from plain text document:** The *TEXT_extractor* looks for the pattern, *abstract*, and extracts the first paragraph following it. If it fails to construct an abstract, *TEXT_extractor* applies the automatic abstracting method on the sentences found in the first, second and last paragraph and on the first sentence of all other paragraphs. The sentences are divided into their constituent words. Dropping all English Noise words, the remaining words are stemmed. The extracted sentences are weighted according to the frequency occurrence of the stemmed words. The *TEXT_extractor* will construct the document's abstract by extracting the highest weighted sentences.
6. **Extracting other words from a plain text document:** The *TEXT_extractor* extracts the words found in the first two paragraphs, the last paragraph and in the first sentence of each other paragraph. After removing the English Noise words, a list of stemmed words is derived. The derived words will be used in the generation of an implicit list of keywords and the generation of a list of significant words used in the document's classification scheme.
7. **Extracting explicitly stated keywords from a plain text document:** The *TEXT_extractor* extracts the text following the word *keyword* as the document's keywords, until the *TEXT_extractor* reaches an *introduction* heading or a new paragraph.

Currently, ASHG supports *HTML, Latex, RTF* and *Text* documents; however, if the document is not any of these types, ASHG applies the *UNKNOWN_extractor.* It extracts the *size* of the document and the *creation date.* It is up to the document's author or provider to enter the remaining SH's information.

Generating an implicit list of keywords and words used in Document classification

ASHG generates an implicit list of keywords in case explicit keywords were not found in the document, the system derives a list of words from the words found in the title, abstract, and other tagged fields. This list of derived words will also be used in classifying the document. However, if the keywords were explicitly stated in the document, then ASHG will augment them with a list of words from the words found in the title, abstract, and other tagged fields.

Generating both lists of words relies on the stemming process that will map the words into their root words, the stemmed word frequency of occurrence and the word location in the document. Because the terms are not equally useful for content representation, it is important to introduce a term weighting system that assigns high weights for important terms and low weight for the less important terms [26]. The weight assignment uses the following scheme:

If the keywords are explicitly included in the document, they convey some important concepts and hence are assigned the highest weight of five. Usually, words found in the abstract are the second most important words, and are assigned a weight of four. The words in the title, are assigned a weight of three. The word appearing in the other tagged fields, are assigned a weight of two.

Each numeric weight is a class by itself defining the words' location. The range of class weight generated will be between two and 14, depending on the postions where a word appears.

For each class, we set the maximum class frequency to be the frequency of occurrence of a term found most often in that class. For instance, if, in class four, we had three terms having two, four and six as frequencies, the system would select six as the maximum class four frequency. The words' frequencies are compared with their corresponding maximum class frequency. For low weighted classes such as two and three, significant terms have the maximum class frequencies. Thus, limiting the number of significant terms. However, all terms found in class eight and more are significant regardless of their frequency of occurrence.

Term Weight	Term Frequency
2	Maximum Class 2 frequency
3	Maximum Class 3 frequency
4	Greater or equal to Maximum Class 4 frequency minus 1
5	Greater or equal to Maximum Class 5 frequency minus 1
6	Greater or equal to Maximum Class 6 frequency minus 2
7	Greater or equal to Maximum Class 7 frequency minus 3
8 or more	All

Table 1: Weight and Frequency numbers used in extracting terms

Two lists of words are generated. The first one containing only the root words or control terms found in Cindi's thesaurus. This list of control terms is used in the document's subject classification scheme. The second list contains the most significant root words not found in Cindi's thesaurus. If no keywords were found in the document, ASHG extracts words having a term weight more than four and their corresponding frequencies of occurrence is the same as the ones tabulated. These words are the document's keywords. In generating a list of control terms used to classify the document, terms having weight of two or more are extracted. The extracted words have the frequencies of occurrence as tabulated in Table 1.

ASHG's Document Subject Headings Classification scheme

An important step in constructing the draft SH is to automatically assign subject headings to the documents. The title, explicitly stated keywords, and abstract are not enough by themselves to convey the ideas or subjects of the document. Since the author tries to convey or to summarise his ideas in the previously mentioned fields, there is a need to use all none noise words found in those fields. To assign the subject headings, ASHG uses the resulting list of significant words generated from the previous section and the control term to subject association. The subject heading classification scheme relies on passing weights from the significant terms to their associated subjects, and selecting the highest weighted subject headings. The following algorithm is used to construct the three levels of subject headings:

1. For each term found in both Cindi's control terms and the generated list of words, the system traces the control term's attached list of subjects (list of *level0, level1 and level2*) headings, and adds the subject headings to their corresponding list of possible subject headings.
2. Weights are also assigned to the subject hierarchies. The weight for a subject is given according to where the term matching its controlled term was found. A subject heading having a term or set of terms occurring in both title and abstract, for instance, gets a weight of seven. The matched terms' weights are passed to their subject headings.
3. The system extracts *Level_2, Level_1* and *Level_0* subject headings having the highest weights from the three lists of possible subject headings.
4. After building the three lists for the three level subject headings, the system selects the subjects using the bottom-up scheme:
   (a) Having selected the highest weighted *level_2* subject headings, the system derives their *level_1* parent subject headings.
   (b) An intersection is made between the derived *level_1* subject headings and the list of the highest weighted *level_1* subject headings. The common *level_1* subjects are the document's *level_1* subject headings.
   (c) The system uses the same procedure in selecting *level_0* subject headings.

Once the process of extracting the meta-information is terminated, the SH is displayed for the source provider to modify, add or remove some of the attributes. Once the provider finishes, the semantic header can be registered in the Cindi database.

6 Analysis of ASHG's Results and Conclusions

The experiments described here are designed to test the accuracy of the generated index and the subject headings classification results. After applying the ASHG on a set of documents, the generated index fields such as title, keywords, abstract and author are compared with those that are found in the document.

The experiments were conducted on a number of documents[40]. These documents dealt with computer science and electrical engineering subjects. Each of these documents was rendered manually in the four formats. ASHG was able to extract all the explicitly stated fields such as title, abstract, keywords, and author's information with a hundred percent accuracy. If the abstract was not explicitly stated, ASHG was able to automatically generate an abstract that would describe the paper. However, ASHG's implicit keyword extraction generated a list of words which included some words that were insignificant. These insignificant words in turn lead to the diversion in subject classification.

The ASHG's automatic subject headings classification results are compared with the INSPEC's classification and with what the papers' authors would regard as good subject classifications and poor ones. For the former we consulted the authors about the subject heading generated by ASHG system for their documents. The results are tabulated in Table 2. which shows a greater than 50% of acceptable subject headings. Some of the ASHG's subject classifications had different words than INSPEC's even though they described the same subject. That was due to the fact that our computer science subject classification was built from ACM and not from INSPEC.

Document Type	Avg. Number of Subject Headings Generated	Avg. Number of Acceptable Subject Headings	Percent of Inspec Heading Discovered
HTML	4.9	66.1%	74%
LaTex	4.4	63%	80%
RTF	4.8	60.6%	65%
Text	5.9	57.0%	80%

Table 2: Summary of ASHG's tests

ASHG's was able to generate between 65% and 80% of the subject heading that were generated by professional catalogers. However, since ASHG produced, on the average more classifications, the accuracy was lower at about 22%. Since our system was only based on the frequency and location of words in a document to determine the document's keywords and subject classification, it missed the importance of the word senses and the relationship between words in a sentence. The simplistic system did not capture the concepts behind the documents, or the ideas that the author was trying to convey. Our results support the idea that word frequency and location are not enough in information retrieval. However, since the ASHG's result will be used as a starting point by the author, he/she has the opportunity to correct the errors and include fields of the SH not given before registering it. Further work is required in refining the subject classification to reduce the number of poor classsifcations.

In conclusion, we believe that resolving word senses and determining the relationships that those words have to one another will have the greatest impact on refining the ASHG's subject classification scheme. Therefore, we plan to pursue semantic level language processing in the future.

References

[1] Baxendale P. B., *Man made Index for Technical Literature - An Experiment*, IBM Journal of Research and Development, 2:4, pp. 354-361, 1958.

[2] Belkin N., Croft W. B., *Retrieval techniques*, Annual review of information science and technology (ARIST), 22, pp. 109-145, 1987.

[3] Blair D. C. , *Language representation in Information Retrieval*, Elsevier Science publishers, New York, 1990.

[4] Brandow R. , Mitze K., Rau L. F., *Automatic condensation of electronic publications by sentence selection*, Information Processing and management, Vol. 31, No. 5., pp. 675-685, 1995.

[5] De Bra, P., Houben, G-J., & Kornatzky, Y., *Search in the World-Wide Web*,

http://www.win.tue.nl/help/doc/demo.ps

[6] Desai B. C., *Cover page aka Semantic Header*,

http://www.cs.concordia.ca/~faculty/bcdesai/semantic-header.html, July 1994, revised version, August 1994.

[7] Desai B. C., *The Semantic Header Indexing and Searching on the internet*, Department of Computer Science, Concordia University. Montreal, Canada, February 1995.

http://www.cs.concordia.ca/~faculty/bcdesai/cindi-system-1.1.html

[8] Desai, Bipin C., *Supporting Discovery in Virtual Libraries*, Journal of the American Society of Information Science(JASIS), 48-3, pp. 190-204, 1997.

[9] Earl L. L., *Experiments in Automatic Extracting and Indexing*, Information Storage and Retrieval, 6:4, pp. 313-334, October 1970.

[10] Edmundson H. P. and Wyllys R. E., *Automatic Abstracting and Indexing Survey and Recommendations*, Communications of ACM, 4:5, pp. 226-234, May 1961.

[11] Edmundson H. P., *Problems in Automatic Abstracting*, Communications of the ACM, 7:4, pp. 259-263, April 1964.

[12] Fung R. and Del Favero B. , *Applying Bayesian Networks to Information Retrieval*, Communications of the ACM, Vol 38, No. 3, pp. 42-57, March 1995.

[13] Fletcher, J. 1993., Jumpstation,

http://www.stir.ac.uk/jsbin/js

[14] Katz, W. A., *Introduction to Reference Work*, Vol. 1-2 McGraw-Hill, New York, NY.

[15] Koster, M., *ALIWEB(Archie Like Indexing the WEB)*,

http://web.nexor.co.uk/aliweb/doc/aliweb.html

[16] Lewis D. D. , Jones K. , *Natural Language processing for information Retrieval*, Communications of the ACM, Vol 39, pp. 92-101, January 1996.

[17] Luhn, H. P., *The automatic creation of literature abstracts*, IBM Journal of Research and Development, 2, pp. 159-165, 1958.

[18] Maron, M. E. and Kuhns, J. L., *On relevance, probabilistic indexing and information retrieval*, Journal of the ACM, 7, pp. 216-244, 1960.

[19] McBryan, Oliver A., *World Wide Web Worm*,

http://www.cs.colorado.edu/home/mcbryan/WWWW.html

[20] Paice C. D., *Automatic Generation of Literature Abstracts - An Approach Based on the identification of self indicating phrases, in information retrieval research*, R.N. Oddy, S.E. Robertson, C.J. van Rijsbergen and P.W. Williams, editors, Butterworths, London, pp. 172-191, 1981.

[21] Paice C. D., *Constructing Literature Abstracts by Computer: Techniques and Prospects*, Information Processing and Management, 26:1, pp. 171-186, 1990.

[22] Rijsbergen C. J. van, *Information Retrieval*, second edition, Butterworths, pp. 17-22, 1979.

[23] Salton G. and Lesk M. E., *Computer Evaluation of Indexing and text processing*, Journal of ACM, Vol 25, No. 1, pp. 8-36, 1968.

[24] Salton G., *Automatic Text Processing: The Transformation, Analysis, and Retrieval of information by Computer*, Addison-Welsey, Reading, MA., 1989.

[25] Salton G., Allen J. , Buckley O. , *Automatic Structuring and Retrieval of Large Text Files*, Department of Computer Science, Cornell University. 1992.

[26] Salton G., Allan J. , Buckley C., and Singhal A. , *Automatic Analysis, Theme Generation, and Summarization of Machine-Readable Texts*, Science, Vol264, pp. 1421-1426, June 1994.

[27] Shayan N., *CINDI: Concordia INdexing and DIscovery system*, Department of Computer Science, Concordia University, Montreal, Canada, 1997.

[28] Stiles, H. F., *The association factor in information retrieval*, Journal of the ACM, 8, pp. 271-279, 1961.

[29] Thau, R., *SiteIndex Transducer*,

http://www.ai.mit.edu/tools/site-index.html

[30] Turtle H. R. and Croft, W. B., *Efficient Probabilistic Inference for Text Retrieval*, In Proceedings of RIAO 91. pp. 644-661, 1991.

[31] Computer and Control Abstracts, Produced by INSPEC, No. 10, October 1997.

[32] *Experimental Search Engine Meta-Index*,

http://www.ncsa.uiuc.edu/SDG/Software/Mosaic/Demo/metaindex.html

[33] Library of Congress Subject Headings, September 1996.

[34] http://www.acm.org/class/1998/ccs98.txt.

[35] Rush J. E., Salvador R., and Zamora A., *Automatic Abstracting and Indexing-Production of Indicative Abstracts By Application of Contextual Inference and Syntactic Coherence Criteria*, Journal of the ASIS, 22:4,pp. 260-274, July-August 1964.

[36] Search WWW document full text,
http://rbse.jsc.nasa.gov/eichmann/urlsearch.html

[37] WebCrawler,
http://www.biotech.washington.edu/WebCrawler/WebQuery.html

[38] World Wide Web Catalog,
http://cuiwww.unige.ch/cgi-bin/w3catalog

[39] http://web.soi.city.ac.uk/research/cisr/okapi/stem.html

[40] http://www.cs.concordia.ca/~faculty/bcdesai/cindi/listofpapers.html