ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

SCHOOL OF INFORMATION STUDIES FOR AFRICA

THE APPLICATION OF INFORMATION RETRIEVAL TECHNIQUES TO AMHARIC DOCUMENTS ON THE WEB

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENT FOR THE DEGREE OF MASTER OF SCIENCE IN INFORMATION SCIENCE

BY

SABA AMSALU TESERRA

July, 2001

ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

SCHOOL OF INFORMATION STUDIES FOR AFRICA

THE APPLICATION OF INFORMATION RETRIEVAL TECHNIQUES TO AMHARIC DOCUMENTS ON THE WEB

BY

SABA AMSALU

Name and Signature of Members of the Examining Board

Ato Getachew Jemenah, Chairman, Examining Board Getachew Jemaneh

Ato Dereje Teferi, Advisor

Ato Million Meshesha, Advisor Million

Dr. Kamal Bechkoum, External Examiner

DEDICATION

To **God** who Gave me Life

And

To my Family

ACKNOWLEDGMENT

I offer my special thanks to my Advisors, Ato Dereje Teferi for his invaluable advice and Ato Million Meshesha for his ongoing advice and support in making this thesis a success.

A distinctive thanks goes to my sister Dr. Ribka Amsalu and to my brother Dr. Solomon Amsalu who have given me their continuous support to complete my study. I also extend my deepest gratitude to Dr. Belayneh Abate who has provided me moral, financial and material support during my stay at SISA.

I would like to express my thanks to my family and friends who have been encouraging me during my study.

I want to mention my great thanks to Ato Berhanu Hailemariam for supporting me in going along with Visual Basic. I also appreciate W/ro Yemisirach Alemayehu for providing me Internet access and Ato Kibru Yisfa for providing me ColdFusion diskette.

Finally, I want to offer my many thanks to all SISA staff who have contributed in one way or another for the completion of this thesis and especially I am grateful to Dr. Nega Alemayehu, for his invaluable comments and advice. I also thank Ato Tesfaye Biru, Ato Workeshet Lamenew and W/ro Weiynishet Abdella for their great assistance.

Saba Amsalu

ABSTRACT

The World Wide Web is an escalating mass of interconnected data that stretches from computer to computer across the world. Information retrieval systems on the Web provide users with relevant information without human intervention, saving time, labor and money.

The Web contains documents of diverse content in different languages. Making those documents accessible to users has become a difficult task with the fast growth of the Web. Hence developing information retrieval systems to cope with inherent features of Web data has been a research area of the time in information science.

In this study an attempt is made to explore the possibilities of applying some information retrieval techniques for Amharic documents on the Web. To back the research, literature review on related works has been made. Different information retrieval techniques and algorithms used on other languages have been reviewed to determine the possibilities of applying them to Amharic documents on the Web.

A database that stores Amharic Web page data, suffix list and index files has been designed. Web page submission form was developed to allow the submission of Web page data into the database. Designing an Amharic query input interface was also part of the research.

Automatic indexing and searching techniques have been applied on a collection of 313 Web pages of Amharic documents taken from Walta Information Center news publications.

Word and stem inverted index options were explored. An Amharic search interface was then created to handle Amharic data on the Web using ColdFusion Studio and ColdFusion Server 4.0 on Windows NT 4.0 Operating System and Internet Information Server (IIS).

The searching algorithm that was implemented is Extended Boolean model, which is a Boolean model with a vector functionality that allowed to rank retrieved documents.

To measure the performance of the prototype system, retrieval experiments have been conducted for twenty-two queries and an average recall-precision graph is drawn. Using terms with suffixes and prefixes removed resulted in a better performance than using words.

Finally, conclusions are drawn based on the test results obtained and recommendations are made as to what further researches could be done for the development of Amharic information retrieval systems on the Web.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER ONE

INTRODUCTION

1.1 BACKGROUND

Most people are faced with a need for information at some time or another. One might ask friends or colleagues for help, but if that is not satisfactory, a more formal search may be initiated in a library or information center.

However, it is not an easy task to search for relevant information in a high dimensional document space where there is vast amount of documents with complex writing styles. The availability of modern information retrieval systems has greatly improved the access to many stored collections (Baeza-Yates & Ribeiro-Neto, 1999).

1.1.1 Information Retrieval

Information retrieval deals with the representation, storage, organization of, and access to information items (Salton & McGrill, 1983). According to Strzalkowski (1999), the task of information retrieval is to extract relevant documents from a collection of documents in response to user queries.

Information is stored in the form of documents in computer information system for it to be made accessible. It is not always necessary for the whole of the document to be stored as part of the system collection. Document representatives are extracted either manually

or automatically and organized in a way that they can be subjected to information retrieval techniques ([a]Rijsbergen, 1996).

Theoretically there is no constraint on the type and structure of the information items to be stored and retrieved with information retrieval (IR) systems. In practice, though, most large-scale IR systems are still mostly processing textual information ([b]Gloor, 1997).

After the document collection is created, the searching mechanism is selected. When the size of the document collection from which information is to be retrieved is small and when the collection is highly volatile or the index space cannot be afforded, sequential searching of the full text can be applied ([b]Rijsbergen, 1996). However in cases where the collection size is large, techniques that allow faster retrieval are used.

Many of the large commercial online systems today base their searching and retrieval on the concept of term indexing (Harman, 2000). An index term is a pre-selected term which can be used to refer to the contents of a document and is basically obtained from the document itself (ibid.). Indexing can be made either manually or automatically. Manual indexing however, is tedious and time consuming making it highly desirable to use automatic indexing, especially in large collections ([a]Gloor, 1997).

1.1.2 Information Retrieval on the World Wide Web (WWW)

The Internet is one of the largest publicly available databases of documents and it has been testing ground for most retrieval techniques. With the explosive growth of documents on the Internet, a number of services have arisen to help users search and

retrieve documents from servers around the world. This is also becoming important in large Intranets, where there is a need to extract information for decision making (Martin, 1999).

Despite the existence of invaluable information out there, searching the Web is challenging for several reasons. For one thing the nature of the data on the Internet is difficult to manage and for another the users' approach in finding their information need is a problem (Baeza-Yates & Ribeiro-Neto, 1999). According to Baeza-Yates & Ribeiro-Neto, data on the Internet has the characteristics listed below that have made it a challenge for information retrieval:

- *Distributed data:* Due to the intrinsic nature of the Web, data spans over many computers and platforms.
- *Volatile data:* High percentage of volatile data exists on the Web due to the Internet dynamics. New computers and data can be added and removed easily.
- *Large volume:* Exponential growth of the Web content and difficulty to cope with this growth
- *Unstructured and redundant data* exists on the Web
- *Quality of data:* In most cases there is no editorial process performed on documents uploaded. Hence, data can be false, out of date, poorly written and with errors (typological, grammatical, etc)
- *Heterogeneous data:* In addition to having to deal with multimedia and with multiple formats, there are also documents written in different languages and, what is worse is the need to deal with documents written in languages having different alphabets.

Some of the problems such as variety of data type and poor quality of data are not solvable simply by software improvements. In fact some of the problems will not change and should not change, as in the case of language diversity.

Users on the other hand face problems such as inability to express their information needs in a query and usually have difficulty to interpret the answer provided by the system, especially due to the amount of data retrieved at once.

1.2 STATEMENT OF THE PROBLEM AND JUSTIFICATION

People are shifting towards electronic publishing, in view of the additional capabilities attained by publishing on the Web over the print (Poulter, 1997).

While publishing on the Web, it is possible to include multimedia elements, and link to information in many locations. Updating and distributing information is also very easy and fast while reducing communication costs, and it is possible to create virtual documents from other sources and applications (Microsoft® Encarta® Encyclopedia 99).

Organizations can also achieve enterprise-wide connectivity by creating internal networks called Intranets based on Internet networking standards and Web Technology. Companies are using Intranets in several ways. Customer files, product inventories, policy manuals, financial information, legal issues, telephone directories and other important information are made available to employees who need them (Laudon, 1998).

Extranets (private Intranets accessible to outsiders) are especially useful for linking organizations with customers or business partners, and for providing product availability, pricing, and shipment of data (ibid.).

Owing to the existence of millions of Web sites and their increase in tremendous amount daily, locating information on the Web has been a critical issue. To alleviate the complexity of the process of following many links to get to documents, much effort has been made and is being made (Poulter, 1997).

Several companies have created directories of Web sites and their addresses, providing search tools for locating information on specified topics. Web page owners, editors of sites or interested groups, who want their pages to be available online, submit short descriptions and corresponding URL (Unified Resource Locator) of Web sites to these directories (SEW, 2001).

With the release of WWW browsers in 1991 and with the increase in HTTP (Hypertext Transfer Protocol) resources, in 1994, the World Wide Web search engines of today began to appear. Search engines crawl the Web and create their listings automatically, providing the opportunity to get updated information on changes to Web sites (Schwartz, 1998).

Most retrieval systems are based in the United States and they focus on documents written in English. However, there are retrieval systems specialized in different countries and in different languages, which are designed to query and retrieve documents in the specific languages (Baeza-Yates & Ribeiro-Neto, 1999).

Though the bulk of information on the Web is in the English language, today there are also documents written in different languages such as French, Chinese, Arabic, etc (ibid.). The variety of languages used to post documents on the Web demands a search tool for each language or a retrieval system having a cross-lingual or multilingual search capabilities.

Amharic, the official language of the federal government and a language used most widely in Ethiopia, is one of the languages used on the Web. Amharic is one of the languages that have their own alphabet.

In terms of native speakers Amharic is approximately the fifth most widely spoken language in the world (IU, 2000). The native speakers are predominantly found in Ethiopia where they are 1/3 (18,000,000) of the population. Because it has been used as an official language of the country for over a century and has been taught in schools, many other Ethiopians speak it as a second language. Unlike most African Languages, Amharic has been a writing language for at least 500 years. And it has a fairly sizable written literature as compared to other Languages in the continent (Furzey, 1996).

Since the introduction of Internet service in Ethiopia, in 1997, more than 60 private organizations and government bodies have created their own Web sites hosted by Ethiopian Telecommunication Corporation Internet Service alone (informal discussion with a Computer Science graduate and involved in designing Web pages at Ethiopian Telecommunication Corporation). Some of these Web sites contain Amharic documents. Ethiopians in the Diaspora are also important elements that do publish Amharic

documents that have invaluable information. There are Web sites that collect information on Web sites related to Ethiopia (AC, 2001). Some of the pages have Amharic language multimedia program and others have documents in Amharic. Local government and private news agencies are nowadays publishing several articles daily. Two of the government news agencies that do publish Amharic news daily are Walta Information Center and Ethiopian News Agency.

However, there is no retrieval system designed to access Amharic documents on the Web. Obtaining these documents is possible only if you have the addresses of the sites or got a link from other sites.

Besides, currently the final draft of a proposal demanding the need for a national information policy has been presented to the federal government (Lishan, 1999). The government is currently making efforts for its realization. Included in the proposal is that their should exist a national database, archives, and regional databases that somehow will be integrated. These databases are expected to give valuable information to academia, research institutes, local NGOs, business sectors and others. Indigenous knowledge like historical facts, legislation, research works, cultural heritages, government proclamations, and other facts about and in Ethiopia that are documented in Amharic could be made available to Ethiopians within, to Ethiopians in the Diaspora and to whoever may need them.

In this initiative it is also indicated that there is a need to localize the contents of the Internet for the society to make use of knowledge available on the Internet.

On top of that wide area networks currently affluent in Addis Ababa and across some regional cities like Dire Dawa, Bahirdar, etc. and the availability of Amharic software are encouraging the use of distributed information sources and hence the need for a means of having access to them.

Furzey (1996) says, communicating in the Geez alphabet is a key factor in promoting information sharing and developing a successful national networking in Ethiopia. This is because, for most people English is a second or third language to communicate easily. According to Furzey, the inability to create and sort databases in Sabean alphabets has hampered data collection.

Furzey further emphasizes that the development of user friendly Amharic interface and the ability to download information in Amharic for on-line connectivity is a priority for the promotion of electronic communication in the country.

Furzey finally recommends the need for a national research effort to develop country specific software and online-interface for communication and information retrieval.

Thus, the aim of this research is to investigate the application of some information retrieval techniques to access Amharic documents on the Web.

1.3 OBJECTIVES

1.3.1 General Objective

The general objective of this study is to apply selected information retrieval techniques on Amharic documents on the Web with the aim of exploring the possibilities of retrieving relevant information in response to user queries.

1.3.2 Specific Objectives

The specific objectives of this study are to:

1. Review the different information retrieval techniques and methods for storing, indexing and searching documents on the Web.
2. Make a review on the Amharic language to identify properties that are relevant for exploring searching and indexing.
3. Collect sample documents available on the Web for making experiments on searching and retrieval.
4. Create a database for storing of Web page data.
5. Develop a submit form to allow users (Web page owners) to submit their Web pages.
6. Select the appropriate retrieval techniques to be applied to design the prototype system.
7. Develop an automatic indexer to index the contents of Web pages in Web sites.
8. Design an interface that will accept query terms in Amharic and display description of Amharic documents retrieved with a link to their URL.

9. Conduct experiments by formulating queries and searching for relevant documents to determine the effectiveness of the system.
10. Based on the results of the experiment, make a relevance judgment for determining recall and precision level of the system.

1.4 METHODS

The methods employed to achieve the above stated objectives of the research are:

1.4.1 Literature Review

Literature review has been made to understand about the Amharic language with respect to its features relevant for exploring searching and indexing and to investigate the information retrieval practices and indexing techniques that are used in other languages, which helped in selecting those appropriate to retrieve Amharic documents on the Web.

Reference is made particularly to journal articles, research works, books and Manual of Amharic font.

1.4.2 Development Environment and Programming Tools

A database containing Web pages of Amharic news articles is created using Microsoft Access 97. Microsoft Access was selected for building the database because it is easy to make use of and meets the expectations of this study. The articles are then subjected to be processed and indexed by an indexer developed using Visual Basic 6.0. Visual Basic

is preferred because it has facilities that are easy to use for accessing data and hence has been found efficient for developing the indexer.

ColdFusion studio version 4.0 is used in creating document input form, the search interface and output forms. Windows NT 4.0 operating system with Internet Information Server (IIS) and ColdFusion Server are used on the server side. Internet Explorer 4.0 is used to browse the Web.

ColdFusion is a Web application construction kit, that has capabilities of extending the standard HTML files with high level formatting functions, conditional operators and database commands (Forta, et al, 1998). It has been used in this study for its features that enable to construct dynamic forms to access databases and to insert data into databases.

1.4.3 Experimentation

313 Web pages of Amharic documents containing news published by Walta Information Center are taken as a sample for this study. The articles consist of social, economic and political news. Walta Information Center publishes post their news articles on the Web daily. These publications were easily accessible and hence the researcher decided to use them. News publications were selected among other Amharic publications on the Web for the reason that it is news that is published in a relatively higher amount and the contents are updated daily.

The title, content and URL of these Web pages is submitted to the database. Documents in the database are indexed using words and teems with prefixes and suffixes removed.

An Amharic query input interface that allows entering query terms in Amharic is created. Experiments to retrieve relevant documents are made and outputs are collected. Finally, retrieval effectiveness of the system is measured against recall and precision. To draw the average recall-precision graph a smoothing algorithm applied by Hmeidi, Kanaan & Evens (1997) is used.

1.5 SCOPE AND LIMITATION OF THE STUDY

The experiment in this study is conducted on a collection of Web pages containing only news articles written in Amharic using VG2 font.

Due to lack of time ways of updating the database of Web page data is not dealt with. The ability to incorporate different characters that serve the same purpose is not included in the prototype and case sensitivity is not considered either.

1.6 ORGANIZATION OF THE THESIS

This study is organized into five chapters. The first chapter consists of the background of the problem and justification for conducting the research. The objectives behind this study are also discussed. The methods used in conducting the research are presented and the scope and limitation of the study is also defined in the first chapter.

The second chapter of this study comprises literature review on the Amharic language and its properties that are relevant for information retrieval from documents written in the language. A review of researches are made on Web searching practices and Automatic indexing researches as applied to other languages.

In chapter three, information retrieval techniques that are highly used and pertain to Web searching are dealt with. The creation of database containing Web page data, techniques applied in developing Amharic indexer using words and stems, and the design of the prototype retrieval system along with the experimental results are discussed in chapter four. Chapter five contains conclusions drawn as a result of the test results and recommendations made for further research.

CHAPTER TWO

LITERATURE REVIEW

2.1 INTRODUCTION

As indicated in the previous chapter the purpose of this study is to apply information retrieval techniques to Amharic documents on the Web.

Research on information processing on Amharic documents has been made in several areas owing to the need to harness the potentials of computer technology for managing documents published in Amharic. Some of the researches carried in this area are: The application of OCR techniques on computer printout, type written and handwritten documents (Worku, 1997; Ermias, 1998; Dereje, 1999; Million, 2000 and Nigussie, 2000); Amharic word parsing (Abiyot, 2000) and stemming algorithm for Amharic (Nega, 1999).

In this chapter, features of Amharic words pertaining to the purpose of this research and the Amharic writing system are discussed.

Reviewing information retrieval researches conducted in Arabic and Chinese is also the focus of this chapter. Web search practices in English languages are also dealt with.

2.2 THE AMHARIC LANGUAGE

The Amharic language is one of the 70 or more languages spoken in Ethiopia. Most of the languages in Ethiopia belong to the Semitic and Cushitic branches of the Afro-Asiatic family. In Ethiopia, the Orthodox Church liturgy, Geez, is the origin of the Semitic cluster of languages in the country. Amharic is one of the Semitic languages spoken by more than half the population of Ethiopia (Keller, 1999).

Being a Semitic language, Amharic has morphology similar to other Semitic languages like Arabic and Hebrew. The writing system of Amharic was taken from Geez, which was the language of literature in Ethiopia until the middle of the 19th century. Geez in turn took its script from the ancient South Arabian languages (Bender, 1976).

There are several characteristics that set off Semitic languages from others. The major once are: the presence of roots and vowel patterns, the formation of derived verbs by certain prefixes widespread in Semitic, and basic vocabulary (Ibid.).

2.2.1 Amharic Word Shapes

The Amharic language has a root-pattern structure. The root is the element that carries the meaning in a word and provides the base for inflection. The root is sometimes called *radical* and is made of consonants. The pattern in a word is a set of vowels that are inserted among the consonants of the root (Microsoft® Encarta® Encyclopedia 99).

In Amharic the verb roots provide large number of verbal derivations and most nouns are derived from verbs (Abiyot, 2000). For example from the radical ቆጠረ 'qtr' (counted) the derived nouns, adjectives and different conjugations of the verb appear as:

qotar-i	ቆጣሪ	'one who counts'
qutir	ቁጥር	'number'
yätäqotärä	የተቆጠረ	'counted'
asqotärä	አስቆጠረ	'he made somebody to count something'
qotatärä	ቆጣጠረ	'he counts something again and again'
tä-qotärä	ተቆጠረ	'it has got counted'
aquatärä	አቋጠረ	'he counted something with somebody'
aqotatärä	አቆጣጠረ	'he helped in the counting process of something'
mäqutär	መቁጠር	'to count'
masqotär	ማስቆጠር	'to make counted'
⋮		
etc.		

In the different forms of the words above, the core meaning remains to be the same due to the common root they share. Many information retrieval systems use algorithms (including standard algorithms, and algorithms built for a specific domain such as medical English) to replace all indexed words with their root forms (Harman, 2000).

Cotterell (1964) categorizes Amharic words into five broad classes as follows:

1. Nominals: noun –like words such as

   bet (house), tïllïk (large), ïrsu (he)

2. Verbals: verb-like words such as

   hedä (he-went)

   sämto (he- having-heard)

3. Words that share characteristics with nominals and verbs in their morphological features

   mämtat (coming) a verbo-nominal

   yätäffa (he-who-was-lost) a nomino-verbal

4. Positional complexes

   wädä bet (towards house)

5. Other word classes

   Responsives-

mïn	(what?)	lämïn	(why?)
wädet or yät	(where?)	man	(who?)
ïndet	(how?)	sïnt	(how many?)

All languages have a way of expressing relations like 'in, on, before, behind, inside, outside, from, near, against.... (Bender,1976). English has independent words that

precede the nouns they relate to. Other ways of expressing these relations are using prefixes, suffixes or internal modifications.

Amharic has prefixes and post positions which occur with prefixes and sometimes without (ibid.). For example most Amharic equivalents of the English 'inside, over, near, until, before, after' consist of two parts: a prepositional prefix and a postposition placed after the noun like the example below,

Ke.....behuala (after) in the middle can be a noun or a verb

ከ-ምሳ በኋላ

Ke- misa behuala to mean 'after lunch'

Amharic uses prepositional prefixes with both nouns and verbs.

Ye-(of)

Ke-(from, since, if)

Le-/li-(for, to)

Be-/bi-(at, by, if)

'ye-' is also used with nouns to make a possessive.

e.g.

ሰው sew (man)

የሰው yesew (of a man, a man's)

የሰው· ቤት· yesew bet (a man's house)

2.2.2 The Amharic Writing System

Amharic alphabet consists of more than 300 symbols that consist of letters (core symbols, special symbols and labialized symbols), numerals, punctuation marks and space. The Amharic script does not have symbols for zero, negative, decimal point and mathematical operators and hence borrows numerals from Arabic and operators from Latin script (Worku, 1997 as quoted by Dereje, 1999).

Some punctuation marks used in English are also used in addition to those in Amharic depending on preferences of individuals, which for specific marks vary from writer to writer (Beletu, 1982).

The core symbols have a main form and six other orders. The seven forms are just different combinations of a consonant with different vowels. The six non-basic symbols are formed by slight modification of the shape of the basic form almost uniformly for all core symbols. The core and six other orders for the consonants 'h, m, b, and k' are illustrated below (Visual Geez for Windows 3.x and Windows 9x Applications, 1995-1997):

ሀ	ሁ	ሂ	ሃ	ሄ	ህ	ሆ
h	hu	hi	ha	he	H	ho

መ	ሙ	ሚ	ማ	ሜ	ም	ሞ
m	mu	mi	ma	me	M	mo
በ	ቡ	ቢ	ባ	ቤ	ብ	ቦ
b	bu	bi	ba	be	B	bo
ከ	ኩ	ኪ	ካ	ኬ	ክ	ኮ
k	ku	ki	ka	ke	K	ko

The full set of the Amharic symbols including numbers and punctuation marks (as taken by Beletu (1982), from Merse Hazen Wolde Kerkos's work) are attached as Appendix I.

The Amharic writing system does not have distinction between capital and lower case letters, i.e. all letters have the same case anywhere in a text. Unlike Arabic and Hebrew, Amharic is written from left to right (Beletu,1982).

2.2.3 Amharic Computer Fonts

There are several Amharic fonts that have been developed over the years. Some of these fonts are (Agafari, Washra, Visual Geez, Power Geez, Alpas, etc (Adebabay, 1999

as quoted by Million, 2000). What makes Amharic coding different from English is that the number of characters in Amharic is more than 256 and the 8 bit character set for English is not sufficient.

Worku (1997) discusses two approaches that have been used to solve this problem. One, by assigning ASCII values to a minimum number of characters, made possible by using common extensions to create the rest of the characters from base characters. For example to represent all 5^{th} order characters only two extensions are needed to be extended to the base characters, thus saving 31 characters. Second, by splitting the characters into two different font files, one file contains the most frequently occurring and the second the rest.

Thus there is no standard Amharic code yet. This has created a problem for Amharic text users as it is not possible to access documents written by different bodies in different fonts. This is also a big problem for posting and having access to Amharic documents on the Web. As Worku (1997) quotes Abas (1993), a consortium created by a number of Amharic software companies has taken an initiative to setup an international standard code based on 16 bit for Amharic, which is expected to solve the Amharic character representation and standardization problem. Currently the Ethiopian Computer Standards Association (ECoSA), according to the ECoSA secretary has already defined the Ethiopic character set and presented it to Quality and Standards Association (QSA) for confirmation. ECoSA has also projects held on character encoding that attempts to work in coordination with UNICODE and in creating a keyboard layout for Ethiopic characters.

2.3 REVIEW OF INFORMATION RETRIEVAL WORKS

Experimentation with retrieval systems in the English language has been conducted for long. Limited research works have also been made in some other languages like Arabic and Chinese (Al-Karashi & Evens, 1994). Two researches that were conducted on Arabic and one in Chinese are presented below.

2.3.1 Comparing Words, Stems, and Roots as Index Terms in an Arabic Information Retrieval System

A system for Arabic information was designed by Al-Kharashi and Evens (1994) to investigate the application of indexing and retrieval techniques on Arabic documents. In this system called Micro-AIRS, a microcomputer system for Arabic information retrieval, a series of experiments were conducted using 29 queries on 355 Arabic bibliographic records of computer and information science.

Micro-AIRS structure has three main components:

- ***User interface:*** consisting of a permanent menu and pull-down menus listed vertically.
- ***Command processor*** : that accepts a user request, validates it, and processes it.
- ***Database handler***: responsible for accessing and updating the data file.

a. Processing the Arabic Language

Arabic is a Semitic language and hence has a grammatical system based on a root and pattern structure. Arabic words are classified into three main categories: nouns, verbs and particles. All verbs and many nouns are derived from the root verbs.

b. Indexing process

The frequency of index terms was not used as a measure for selecting significant terms for two reasons:

1. As quoted by Al-Kharashi and Evens, Luhn's (1958) law that states that the frequency of occurrence of a word in an article furnishes a useful measurement of word significance has not been verified with Arabic text, and it is not realistic to use it as a solid base for indexing Arabic text.
2. The type of collection used to test the system consisted of a short title with no abstract except for few records and it is seldom to find a word more than once or twice in the same document.

For the purpose of creating queries and making relevance judgement experts were required. A subset that covers a specific area was chosen for this reason.

Spell checking was performed to clear spelling errors. The indexing process accepts plain text and extracts all words from every extractable field in all database records (category, title and abstract). The length of any extracted term is limited to 25 bytes; if above that it was truncated. The index list is then sorted.

c. Creation of the word-stem-root dictionary

A word-stem-root dictionary was created using the document collection. The dictionary was used during indexing and retrieval to identify the stem or root of a given word.

d. Experimentation and evaluation of the Micro-AIRS system

Graduate students in the area made 60 queries. 10 were removed because they were repetition. The 355 database records were divided into three sets. Three people were then giving queries and judging for relevance of the documents retrieved. The judgements of the three sets were then grouped together in a relevance judgement matrix. Out of the 50 queries only the 29 queries were found to have one or more relevant documents. Hence, only the 29 queries were used for the relevance judgement.

Experiments made to compare *words*, *stems*, and *roots* as index terms in the retrieval system showed that root and stem retrieval methods were superior over the word retrieval methods. The root works as well as or better than the stem at low recall levels, and better at high recall levels.

To rank the retrieved documents, the Cosine, Dice and Jaccard coefficients were used. The ranking process showed that all three similarity coefficients produced exactly the same ranking for all queries.

2.3.2 Design and Implementation of Automatic Indexing for Information Retrieval with Arabic Documents

Hmeidi, Kanaan and Evan (1997) made another experiment on a collection of 242

abstracts of Arabic documents whose contents were computer science and information systems. In the study both manual and automatic indexing techniques were implemented to select content bearing words from the document collection. A series of experiments were conducted on both manual and automatic indexing environments using words, stems ad roots as index terms.

a. Data description

Machine readable abstracts for Arabic documents were created. These abstracts contained records with categories, titles, authors, sources, and abstracts.

60 queries were later developed by Computer Science students. The abstracts were divided into six sets, each with 40 abstracts. Experiments were finally conducted with the queries.

b. Indexing process

To implement automatic indexing, the frequency of each word in each document in the collection was determined and the total collection frequency for each word was calculated by summing the frequencies of the word across all the documents in the collection.

Upper and lower thresholds were then selected to remove non content bearing terms after observing the words in decreasing order of their collection frequencies. The high frequency and low frequency words were removed and the middle frequency words were used as index terms.

The most highly frequent word had a frequency value of 1,105 and the lowest frequency was 1. The upper and lower thresholds were selected to be 240 and 3 respectively.

In this experiment conducted by Hmeidi, Kanaan and Evan (1997) the number of running words in the Arabic corpus was 48,538 and the number of distinct words was 9,443, making the token to type ratio 5.14.

Hmeidi, Kanaan and Evan (1997) quote Yahya (1989) and state the number of running words per number of distinct words for texts of different sizes in Arabic and English were measured and found as presented in the table below:

Length of Text	Arabic Distinct Words	Arabic Ratio	English Distinct Words	English Ratio
100	84	1.190	69	1.449
200	149	1.342	124	1.613
400	281	1.423	165	2.424
800	507	1.578	328	2.439
1600	902	1.774	621	2.576
3200	1537	2.082	871	3.674
6400	2715	2.357	1361	4.702
12800	4895	2.615	2337	5.477
16000	5775	2.771	2699	5.928
20000	6956	2.875	3154	6.341

Table 2.1 Token to Type Ratio on Arabic and English Documents

According to Hmeidi, Kanaan and Evan (1997) the number of individual words in Arabic appears less often than English words for three reasons. First, verbs and many derived nouns are formed from roots. Person, number, gender, and tense are expressed by affixes. Second, subject and direct object pronouns are often combined with the verb. Third, the definite article "the" and the word "and" as well as several common prepositions are combined with the word following them with no word spaces.

c. Stem and root indexing

To increase the possibility of retrieving potentially relevant items, the Arabic words were stemmed and experiments were conducted with stems as index terms. Since Arabic has a root pattern morphology, roots of words were also extracted. As in the case of full words the stem and root words were ordered by their collection frequencies, and the high and low frequency words were removed from the index list.

d. Experimental results

The results showed that automatic indexing gives better results than manual when stems are used as index terms at recall level above 3.0 and when roots are used at a recall level above 5.0. Automatic indexing is found better than manual indexing in cases where words are used as index terms. In manual indexing environment, using roots gives better results than using stems or words. In automatic indexing environment, using roots gives better results than using words. When comparing between the root and stem indexing in the automatic indexing environment, the differences were insignificant.

2.3.3 Experiments with Automatic Indexing and a Relational Thesaurus in a Chinese Information Retrieval System.

A series of experiments were carried out by Wan et al. (1997) with an interactive Chinese information retrieval system named CIRS and an interactive relational thesaurus, where they found that thesaurus can improve retrieval effectiveness and automatic indexing techniques developed for English do work for Chinese.

Chinese differs from English in the opposite direction to the difference Arabic makes with English. Chinese is more synthetic than English. Changes in gender, case, and number are usually unmarked in the surface structure, so that individual word frequencies are higher and inverse document frequencies are lower. Words in written text consist of one, two ,three or four characters or sometimes more but there are no word spaces. Hence, words cannot be identified in a manner in English.

In this research A Chinese word-segmentation system built by Lin in 1995 was used. Automatic and manual indexing experiments were conducted on CIRS, an experimental retrieval system that was implemented on Microsoft Chinese Windows environment.

a. The system has the following major components:

- *User interface: allows query construction, execute the queries and display retrieved documents.*
- *Inference engine: The inference engine performs searching using automatic or manual indexing.*

- *File management subsystem:* provides data access to dBase III databases in order to support query construction and document display.

b. Experimental procedure

A collection of 555 abstracts in the area of computer and information science published by the Science and Technology Information Center, Republic of China was developed. Three graduate students, native speakers of Chinese, in Computer Science department of Illinois Institute of Technology formulated 76 queries, out of which 30 were taken and 46 were removed for being redundant and some out of the scope of the collection. An expert who has M.S. degree in computer Science made the relevance judgment.

c. Construction of the relational thesaurus

As quoted by Wan et. al. (1997), the relations defined by Casagrande (1967) to level relationships between words was used to construct a relational thesaurus. Those relationships are : attributive, function, spatial, operational, comparison, exemplification, class inclusion, synonym, antonym, provenience and constituent.

c. Indexing process

A professional who has M.S. degree in computers science carried out the manual indexing process. Automatic indexing was facilitated by using a Thesaurus of Science and Technology, a dictionary that contains approximately 49,270 terms. Keywords automatically obtained from abstracts were chosen as index terms if they appear in this collection.

e. Experimental Results

The experimental results showed that the relational thesaurus improves effectiveness in both the automatic and manual indexing environments; and automatic indexing works at least as well as manual indexing.

An experiment in English using relational thesaurus was made by O'Connor in 1980 as discussed by Wan et al. (1997) and the result showed that better results were obtained with relational thesaurus than without.

2.4 SEARCHING THE WEB

Since its introduction in 1989 the World Wide Web has been growing in a very fast rate that there are documents that can be counted in terabytes and images, audio and video are also available (Baeza-Yates & Ribeiro-Neto 1999).

There are basically two ways of searching documents on the Web. These alternatives are (Lager, 1996):

- to use search engines that index a portion of the Web documents as a full-text database
- to use Web directories that classify selected Web documents by subject

2.4.1 Search Engines

Search engines are computer programs that compile lists of documents on the Web and the contents of those documents (Microsoft® Encarta® Encyclopedia 99). Search

engines respond to a user entry, or *query,* by searching the lists and displaying a list of documents that match the search query. A number of search engines include the abstract of the text of Web pages in their lists, but others include only the titles or addresses (Universal Resource Locators, or URLs) of Web pages. Some search engines occur apart from the WWW, indexing documents on Intranets or local area networks (ibid.).

Search engines create their listings automatically by crawling the Web. If people change their Web pages search engines eventually find these changes (SEW, 2001). Search engines use programs called spiders (also called robot, softbot, wanderer, crawler, and fish) to search the WWW for new documents.

A Spider is a computer program that automatically monitors Web pages on the World Wide Web (Microsoft® Encarta® Encyclopedia 99.). Most Web pages include at least one link to another Web page, and some include more. A spider takes advantage of this structure by starting at one Web page and working its way out by following every link on a Web page and then following every link provided by the new Web pages.

Spiders save the URL, of every Web page they visit. These spiders are used by search engines to build indexes of Web pages that users can access to search for information on a particular topic. Spiders often also store the title and partial or complete text of a Web page so users can do more detailed searches. Some spiders store only URLs of Web pages that have not been listed yet in order to update lists or provide lists of new Web pages. Some spiders make note of URLs that are no longer valid in order to correct lists (ibid.).

2.4.2 Search Directories

Unlike search engines, a directory depends on humans for its listings. Web page owners submit short descriptions representing their Web site to the directory, or editors of sites or interested groups submit sites in a special area. Changing the Web pages does not change the listing. A typical example of a search directory is Yahoo (SEW, 2001).

The largest directory Yahoo has close to one million pages classified and specialized directories in other languages such as Danish, French, Korean, Norwegian, ... (Baeza-Yates & Ribeiro-Neto, 1999).

The Web coverage provided by directories is usually small, however, the answers returned are found to be more relevant (ibid.).

2.4.3 The Anatomy of a Large-Scale Hypertextual Web Search Engine

Page and Brin (2000) from the Computer Science Department of Stanford University, U.S.A. have presented the Anatomy of Google as described below.

Google is a large-scale search engine with a full text and hyperlink database of at least 24 million pages. Google is designed to scale well to extremely large sets of data considering both the rate of growth of the Web and technological changes.

According to Page and Brin (2000), search engines have grown from academic domain to the commercial. Most of them migrating in to being advertising oriented. Google aims at brining more development into the academic realm. Google aims to support research activities in large-scale Web data and create an environment where researchers can come in.

The goal of designing Google was to increase search quality by avoiding junk results and increasing precision. Google makes use of link structure and anchor text.

Google has two features that enable it to produce high precision results.

1. It uses the link structure of the Web to calculate a quality ranking for each page (PageRank). Assuming page A has pages T1...Tn that point to it PageRank of page A is calculated as:

   PR(A)=(1-d) +d(PR(T1)/C(T1) +...+PR(Tn)/C(Tn))

   where PR(A) is PageRank of page A

   d is damping factor (the probability that the "random surfer" which is given a random page and keeps on clicking on links, but eventually gets bored and requests another random page).

   d can be set from 0 to1 (in Google it is 0.85),

   PR(T1) ... PR(Tn) are PageRanks of pages that point to page A

   C(T1)...C(Tn) are number of links going out of the pages C(T1)...C(Tn)

2. It utilizes links to improve search results

   Most search engines associate the text of a link with the page that the link is on. Google in addition associates the link with the page it points to. Its advantages are:

   - Anchors often provide more accurate description of pages than the pages themselves. Anchor propagation mostly provides better quality results
   - Anchors may exist for documents that cannot be indexed by a text based search engine (images, programs and databases). Hence they enable to access pages that have not been crawled.

Other features of Google are:

- It has location information for all hits and so makes use of proximity in search
- It keeps visual presentation details such as font size of words, bolder and larger fonts are given a higher weight.
- Full raw HTML of pages is available in a repository

Web crawling in Google is performed by several distributed crawlers. There is a URLserver that sends lists of URLs to be fetched to the crawlers. The Web pages fetched are sent to the storeserver where they are compressed and stored into a repository. Every Web page is given a docID whenever a URL is parsed of a Web page. The indexer reads the repository, decompresses the documents and converts each document into a set of word occurrences called hits. The hits record the word, its position in document, approximation of the font size and capitalization. The indexer also parses all the links in every Web page and store information about them (where each link points

from and to the text of the link) in an anchors file. The URLresolver reads the anchors files and converts relative URLs into absolute URLs and then into docIDs.

Google answers most queries in 1-10 seconds. Further, Google employees a number of techniques to improve search quality including PageRank and proximity searching. Google has a complete architecture for gathering Web pages, indexing them, and performing search queries over them.

2.4.4 Boolean Searching on the Internet

Cohen (2001) has made a review on Boolean Searching on the Internet. Much of the searching on the Internet is based on the principles of Boolean searching which refers to the logical relationship of terms.

The Boolean logic consists of three logical operators: OR, AND & NOT. OR logic is commonly used to search for synonymous terms or concepts. The OR logic collates the results obtained for at least one of the terms in the query. Hence as the number of terms in the query increases the number of documents retrieved increases.

While using the AND operator, records in which both of the search terms are present are retrieved. The more terms in the query the fewer the records are retrieved using the AND operator.

There are few Internet search engines that use the proximity operator NEAR. NEAR is a more strict AND in that it also determines how close the terms should be found within the

document. Alta Vista (one of the popular search engines) uses the NEAR operator with a closeness of 10 words; which means there cannot be more than ten words between the AND separated search terms in the document.

The NOT operator retrieves records in which one of the terms is present excluding those documents containing the second term (the term after the NOT operator).

In Internet search engines the above logical operators are used in three ways:

1. Full Boolean logic with the use of the logical operators.

Some search engines offer the option to do full Boolean searching. In this method it is possible to use any combination of logical operators in a query. Brackets are used to force the order of processing.

E.g. (cats OR felines) AND behavior

This could be a query to know about the behavior of cats.

2. Implied Boolean logic with keyword searching

In implied Boolean logic the space between keywords in the query is by default taken as either OR logic or AND logic. Many well known search engines, some being: AltaVista, Excite, Infoseek, and MetaCrawler default to OR. Google, AOL.COM, Lycos, and North Light default to AND. Implied Boolean logic is so common in Web searching that it may be considered a de facto standard.

E.g. information retrieval

Search engines that default to AND will retrieve documents with both the terms in the query above.

3. Predetermined language in a user fill-in template

Some search engines have a menu from which you choose the logical operator (often the choices are *Any of these words/Should contain all words*).

CHAPTER THREE

INFORMATION RETRIEVAL TECHNIQUES

3.1 INTRODUCTION

Techniques for storing, maintaining, and retrieving information from English documents have been studied, implemented and tested for more than a couple of decades. However, whether those techniques are appropriate to all other languages is not sure. Experiments have been done to adopt them for languages like Arabic and Chinese by including the specific features of the languages.

This chapter discusses the basic information retrieval processes and text processing and indexing techniques, and different IR models available to develop an information retrieval system.

3.2 THE RETRIEVAL PROCESS

An information retrieval system is a system that helps to retrieve documents or texts with information content that is relevant to a user's need (Jones & Willett, 1997). The following figure shows the framework of a text retrieval system as described by [b]Rijsbergen (1996).

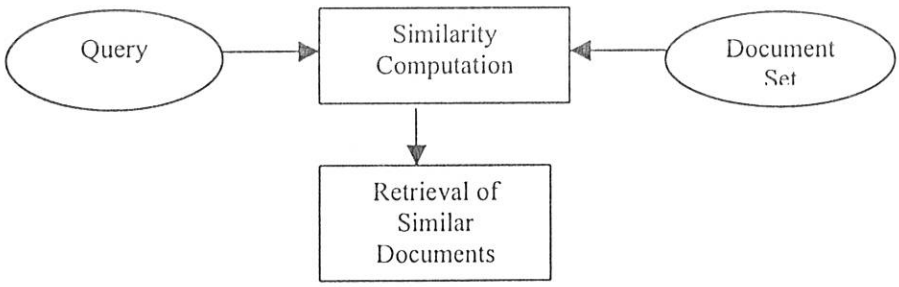


Figure. 3.1 The Concept of Text Retrieval

As shown in the above diagram, the information need of the user has to be translated into a query that can be processed by information retrieval (IR) systems. A query is a set of key words describing the summary of the information need of the user (ibid.).

Similarity computation is then made between the query and the documents in the document collection. Documents in the set that have certain degree of similarity with the query and might be relevant to the user are finally retrieved (ibid.).

In cases where the full text document set is large, an index file containing terms that represent each document in the collection is built. Since not all terms are equally important and not all documents are equally relevant index terms may be weighted and the retrieved documents may be ranked before they are sent to the user. In some cases the user pinpoints some of the documents as useful and initiates a user feedback cycle. The system then uses the user preferred documents to change the query formulation (Baeza-Yates & Ribeiro-Neto, 1999).

There are several approaches in processing documents and subjecting them to be retrieved. The choice of which techniques to use for a given system depends on the type of information available and the output expected from the system apart from other issues

such as language features. Important algorithms that are used in most information retrieval systems are discussed below.

3.3 AUTOMATIC INDEXING OF DOCUMENT CONTENTS

To search documents for a query, there are basically two options (Baeza-Yates & Ribeiro-Neto, 1999). The first one is to scan the whole text of documents and find for the query term(s) in the documents. Sequential scanning of the whole text is preferred when the text collection is not large and when the text is volatile. However, this is practically very difficult in situations where the size of the documents in a document space is large.

The second option is to build a data structure over the text. This technique speeds up searching since you don't need to scan the entire text sequentially. When the text collection is large and semi-static, this method is preferable.

An index term is a keyword or group of related words which has usually the semantics of a noun (Salton & McGrill, 1983). The index terms could be taken from a complete document text, abstract, title only or perhaps a list of words from the Web document ([a]Rijsbergen, 1996). Lycos indexes only the top 100 words of Web pages, and Infoseek indexes every word in the Web pages (Lager, 1996).

3.3.1 Systems Based on Inverted Files

The preeminent indexing technique for most applications these days have become inverted files (Baeza-Yates & Ribeiro-Neto, 1999). An inverted file contains index terms and list of occurrences. Each of the occurrences (document reference numbers) uniquely

identify a document for which the given term is representative. Hence searching for documents for a given query involves searching for the query terms in the inverted file to obtain the corresponding occurrences. Finally, the identified documents are selected from the document file (Salton & McGrill, 1983).

3.3.2 Automatically Producing Index Terms

Producing index terms involves two basic tasks. First, documents are preprocessed to identify terms that are capable of representing the content of each stored document. Second, weights are assigned to each term, reflecting its presumed importance for purposes of content identification (Salton & McGrill, 1983).

3.3.2.1 Preprocessing documents

Before documents can be indexed, they are preprocessed to avoid characters or parts of words or words that could reduce the retrieval effectiveness of the system. Some of the most important text operations that are considered during indexing are:

a. Lexical Analysis of the text:

Lexical analysis is the process of converting a sequence of characters in a text to sequence of words (Baeza-Yates & Ribeiro-Neto, 1999). Characters that go with words but are not part of the words are eliminated. These characters are: digits, hyphens, and punctuation marks.

Yet, it is not always ideal to simply remove those characters. Numbers are not good index terms unless there is a surrounding context. For instance it is vague if one uses **1999** as a query term. But if **death rate in Ethiopia in 1999** is taken then 1999 can be specified in the query. On the other hand, numbers may also exist together with letters as 10a.m., 300B.C., which may be important index terms. In Amharic there are similar forms like for example **1990** እ.አ.አ., which means '1999 G.C.'.

Punctuation marks are in some cases important elements of words, when they exist within or between words, in the English language. Similar forms exist in Amharic.

- Hyphens: some words can appear in both hyphenated and unhyphenated versions. The treatment of hyphens is critical to retrieval of certain words that exist more as hyphenated. In Amharic some writers use hyphens for words like, ፀረ-ሰው ልብ-ወለድ ሠርቶ-አደር ሥርዓተ-ነጥብ

- Periods: periods can appear as a part of a word, such as computer file names (paper.htm),. In Amharic periods are also used in acronyms as in:

  1999 እ.አ.አ. a short form of እንደ አውሮፓ አቆጣጠር (to mean G.C.)

  ዶ.ር Dr.

- Slashes, underscores: these can appear as parts of words in Amharic in cases like:

  ት/ቤት to mean 'school' in short.

  ወ/ሮ (Mrs.)

  Usually . (dot) and / (slash) are used interchangeably to express short forms, like ዶ/ር for ዶ.ር

  Underscore is used as in between intervals for example

ከ60_75 ኪሎ ግራም which means between 60 and 75 Kg.

Problems similar to the ones above are usually solved by having general rules and specifying the exceptions case by case. But still there is no clear solution for this problem. Some Web search engines are opting for avoiding text operations altogether (Baeza-Yates & Ribeiro-Neto, 1999).

b. Elimination of Stopwords

Most automatic indexing techniques work with a stop list that prevents certain high-frequency or "fluff' words from being indexed (Harman, 2000).

The assumption is that, words that are high-frequency function words are weak discriminators and cannot be used to identify documents. Such words are called stopwords or negative dictionary (Salton & McGrill, 1983). According to Baeza-Yates & Ribeiro-Neto (1999) in English these words comprise 40-50% of text words.

Articles, prepositions, and conjunctions are common candidates of stopwords. Some verbs, adverbs and adjectives could also be treated as stopwords. Elimination of these words is necessary as they result in the retrieval of non-relevant documents. In addition the size of the index is reduced with their removal ([a]Rijsbergen, 1996).

As discussed in chapter two, the structure of words in Amharic is different from English. In Amharic the definite article 'the' is used for masculine singular and masculine and feminine and plural nouns as a suffix -u, (ibid.).

house ቤት houses ቤቶች

the house	ቤቱ	the houses	ቤቶቹ
women	ሴቶች		
the women	ሴቶቹ		

In nouns ending with a vowel the suffix is spelt as ው (w)

cup	ሲኒ
the cup	ሲኒው

The definite article for feminine singular is also a suffix with three different forms

cow	ላም
the cow	ላሚቱ, ላሙዋ or ላሚትዋ

The indefinite articles (a and an) are not usually used in Amharic unless to avoid ambiguity, where in certain cases the numeral 'one' (አንድ) is used. One such example is

ወንበር አምጣ	could mean bring a chair or bring chairs
አንድ ወንበር አምጣ	can only mean bring a chair

In Amharic prepositions in most cases occur as prefixes and in some cases as infixes (Dawkins, 1960). Simple Prepositions exist prefixed to nouns, pronouns and adjectives.

ለአባቱ	for his father
በጂማ ተቀመጠ	he stayed at Jima
በእርሳስ ጻፍኩ	I wrote with a pencil
በመኪና ሄዱ	they went by car
ከአገሬ መጣሁ	I came from my country
የአሽከሩ ሚስት	the wife of the servant
ከመኝታ ቤት ነው	it is in the bedroom

Infixed prepositions between verbs and objective nouns are used in Amharic.

ንገረኝ	tell me
ንገርልኝ	tell for me
ይሰራበታል	he works with it
ዳኛው ፈረደበት	the judge judged against him
ተነሳበት	he rose against him.

There are conjunctions in Amharic that occur as a separate word. Common once are:

እና	and	በሁዋላ	after
ቢሆንም	however	አለበለዚያ	unless
ግን	but	ምክንያቱም	because
ወይም	or	በፊት	before

እና May also be occur as a suffix dropping እ alternatively. For instance one can write

ሴቶች እና ህጻናት women and children or

ሴቶችና ህጻናት

The above features of word forms have to be given consideration while trying to remove non content bearing words from Amharic text.

There are two most known approaches of excluding non content bearing words from being included as index terms. One commonly used approach to building a stop list is to use one of the many lists generated in the past referred as controlled vocabulary. These lists contain many of the words that always have a high frequency, and also may contain "fluff,' words that may not have a high frequency (Harman, 2000).

The second approach is the one suggested by Luhn (1958). Luhn states that the frequency of occurrence of a word in an article furnishes a useful measurement of word significance ([a]Rijsbergen, 1996). To construct the stopwords, the word frequency listing of the text to be indexed is produced. Upper and lower cut-off points are then decided to cut the high and low frequency words with no importance in identifying the content of the document (s) from being index terms.

Some Web search engines, however do not remove stopwords from index structure believing their removal will decrease recall (Baeza-Yates & Ribeiro-Neto, 1999). In phrases like **'to be or not to be'** after elimination of stopwords the term **'be'** remains, which does not represent the phrase.

c. Stemming

There is a problem in retrieval usually as the word specified in a query of a user may not be found in the relevant document but the variants of the word are in a relevant document. To overcome this problem the substitution of the words with their stems (a portion of a word which is left after the removal of its affixes) is one method adopted (Harman, 2000).

Many information retrieval systems also use stemming to replace all indexed words with their root forms (ibid.).

Stemming is done for two principal reasons (ibid.):

- the reduction in index storage required and

- the increase in performance due to the use of word variants.

The simple prepositions discussed in section *b* above and other suffixes in Amharic can be removed by stemming.

d. Construction of thesaurus

Thesaurus is constructed for the expansion of the original query with related terms. The thesaurus provides a map of a given field of knowledge, indicating the relationship of concepts, which in information retrieval help to index documents under all of the related terms (Foskett, 1997).

3.3.2.2 Weighting

Distinct index terms have varying relevance when used to describe document contents (Baeza-Yates & Ribeiro-Neto, 1999). The relevance of a term is then captured by assigning numerical weights, or values representing its importance. Term weighting is essential to systems that perform statistical or probabilistic ranking.

There are various models of assigning weights to terms in documents and in queries:

a. Term Frequency Weights (tf)

Term frequency is obtained by counting the occurrence of a term in a document (Salton & McGrill, 1983). Total Collection frequency of a term is calculated as:

$$TOTFREQ_k = \Sigma\ FREQ_{ik} \text{ (for i ranging from 1 to n)}$$

WHERE $FREQ_{ik}$ = the frequency of term k in document i

n = number of documents in the collection

This weighting method has been found misleading in that documents vary in their size. It is important to normalize ranking for length because otherwise long documents often rank higher than short documents, even though the query terms may be more concentrated in the short documents (ibid.).

b. Inverse Document Frequency Weights (IDF)

In determining IDF weights, the frequency of occurrence of terms is weighted by the number of documents in the collection that contains the term. If the term occurs in many documents it is weak document identifier and hence given a low weight.

To calculate the IDF weight

$$IDF = \log_2(n) - \log_2(DOCFREQ_K) + 1$$

WHERE n = total number of documents in the collection

$DOCFREQ_K$ = number of documents that contain term *k*

The IDF weight is based on the premises that terms that uniquely identify a document are not found in every other document or in many other documents. But how frequently that term exists in a document is not taken into consideration (Salton & McGrill, 1983).

c. The composite Measure (tf*idf)

The composite measure of term *k* in a document *i* would increase as the frequency of the term in the document ($FREQ_{iK}$) increases and decreases as the number of documents in the collection that contain the term *k* ($DOCFREQ_K$) increases.

$$WEIGHT_{iK} = FREQ_{ik} * [\log_2(n)-\log_2(DOCFREQ_K)+1]$$

WHERE $FREQ_{ik}$ = the frequency of term k in document i

n = total number of documents in the collection

$DOCFREQ_K$ = number of documents that contain term *k*

The composite measure assigns a high weight to terms occurring in only a few documents. The composite weight gives importance both to the distribution of a term in the documents in the collection and the frequency of occurrence of the term in the individual documents (Salton & McGrill, 1983).

d. The Signal – Noise Ratio

The information content of a term can be measured as an inverse function of the probability of occurrence of the word in a given text. The higher the probability of occurrence of a word in a text, the lower its information content (Salton & McGrill, 1983).

$$INFORMATION = -\log_2 p$$

WHERE p = probability of occurrence of the word

When a document is characterized by *t* terms, each occurring with a probability p_k, the average information gained by using one of the terms is given by Shannon's formula as

AVERAGE INFORMATION = $-\Sigma\ p_k \log_2 p_k$ (*k* ranging from 1 to *t*)

The average information is maximum when probabilities of the terms are all equal to 1/t for t distinct terms.

The noise for index term k for a collection of *n* documents is calculated as

NOISE = Σ ($FREQ_{ik}$ /$TOTFREQ_k$)* ($TOTFREQ_k$ /$FREQ_{ik}$) (for i ranging from 1 to n)

e. The Term Discrimination Value

The term discrimination value is the resolving power of a term with respect to a document (Salton & McGrill, 1983). The similarity of all pairs of documents Di and Dj in a document collection D can be measured as

AVERAGE SIMILARITY =K Σ ΣSIMILAR (Di,Dj)

(for i and j ranging from 1 to n)

WHERE K= a constant

n is total number of documents

When all the documents are identical the AVERAGE SIMILARITY becomes 1

The space density is computed more efficiently by creating an artificial average document (Dc) as the *centroid*, where the terms are assumed to exhibit average frequency characteristics.

AVGSIM = K ΣSIMILAR (Dc,Di) (for i ranging from 1 to n)

The discrimination value $DISCVALUE_k$ for each term *k* can be calculated as

$DISCVALUE_k = (AVGSIM)_k - AVGSIM$

Terms can then be ranked in decreasing order of the discrimination value. The terms at the top are then highly specific and those at the end are more general.

- Good discriminators have positive $DISCVALUE_k$
- Indifferent discriminators have close to zero $DISCVALUE_k$
- Poor discriminators have negative $DISCVALUE_k$

Weights can then be calculated as

$WEIGHT_{iK} = FREQ_{ik} * DISCVALUE_k$

On the Web the term frequency and inverse document frequency weights are highly used. Another approach is to consider the font sizes and boldness of text in the pages. Bigger and bolder text are considered more important (Page & Brin, 2000).

Some new ranking methods rank pages with out considering the weights of terms. This is by using hyperlink information. The number of hyperlinks that point to a page provides a measure of its popularity and quality. Pages that have many outgoing links to popular pages are also considered of a high quality (Baeza-Yates & Ribeiro-Neto, 1999).

3.4 INFORMATION RETRIEVAL MODELS

Basically there are three classic models in information retrieval: Boolean, vector and probabilistic (Baeza-Yates & Ribeiro-Neto, 1999).

3.4.1 The Boolean Model

The Boolean model is a model based on set theory and Boolean algebra. The queries are represented as Boolean expressions that have precise semantics. Boolean model is adopted by many of commercial bibliographic systems owing to its inherent simplicity and neat formalism (Baeza-Yates & Ribeiro-Neto, 1999).

The query in a Boolean model is made of terms linked by the logical operators AND, OR and NOT. Thus, the presence or absence of a query term in a document only matters. The index term weights are assumed to be binary (0,1) (ibid.).

An obvious way to implement the Boolean search is through the inverted file. To satisfy a query, the set operation corresponding to the logical connectives is performed.

For the term Ki in the index

Term	Documents
K1	D1, D2, D3, D4
K2	D1, D2
K3	D1, D2, D3
K4	D1

For a query Q= (K1 AND K2) OR (K3 AND (NOT K4))

(K1 AND K2) = (D1, D2, D3, D4) INTERSECTION (D1, D2)

= (D1,D2)

(K3 AND (NOT K4)) = (D1, D2, D3) - (D1)

= (D2, D3)

(K1 AND K2) OR (K3 AND (NOT K4))= (D1,D2) UNION (D2,D3)

= {D1, D2, D3}

The result that satisfies the query is {D1, D2, D3}. Each element is true for the query. And hence the three documents are retrieved.

The Boolean model has inherent limitations that lessen its merit for text searching.

Boolean retrieval results in a simple partition of the document collection into discrete subsets; those that match the query and those that do not meet. Thus, the Boolean model retrieval strategy is on the premises that a document having the term(s) in a query is assumed to be relevant without taking any weighting (grading) which decreases retrieval performance. In addition, there is no means to consider the relative importance of the different components of the query and also no partial much to the query conditions. Implying if term *k1* exists in a document and the query *q1* is *k1* AND *k2*, the document is considered totally irrelevant (Jones & Willet, 1997).

A slight modification of the Boolean search is one which only allows the AND logic but takes the actual number of terms the query has in common with a document This number is known as the *co-ordination level* and the searching strategy is called simple matching ([b]Rijsbergen, 1996). The documents are then partially ranked (there can exist more than one documents at any level) by the co-ordination level.

For the query Q above Q= K1 AND K2 AND K3 the following ranking is obtained.

3 D1, D2

2 D3

1 D4

The simple matching search strategy avoids the need for the user to formulate a complicated Boolean query and gives an additional opportunity for getting ranked documents.

Another alternative is to extend the Boolean model with the functionality of partial matching and term weighting. The extended Boolean model was introduced in 1983 by Salton, Fox and Wu to avoid the critique on the Boolean model (Baeza-Yates & Ribeiro-Neto, 1999). For a query

$Q_{or}=k_1 \vee k_2 \vee \ldots \vee k_m$

$Sim(q_{or},d_j) =((x_1^p+x_2^p+\ldots+x_m^p)/m)^{1/p}$

And for a query

$Q_{and}=k_1 \wedge k_2 \wedge \ldots \wedge k_m$

$Sim(q_{and},d_j)=1-(((1-x_1)^p+(1-x_2)^p+\ldots+(1-x_m)^p)/m)^{1/p}$

Where x_i stands for the weight $w_{i,d}$ associated to the pair $[k_i,d_j]$.

$1 \leq p \geq \infty$ is a newly introduced parameter whose value must be specified at query time. For p=1 conjunctive and disjunctive queries are evaluated by sum of term-document weights as done by vector-based formulas, which compare the inner product. Hence, with p=1 the extended Boolean model behaves like the Vector model (ibid.).

When p=1

$Sim(q_{or},d_j)= Sim(q_{and},d_j)=(x_1+\ldots+x_m)/m$

When $p=\infty$

$$Sim(q_{or},d_j)= \max(x_i)$$

$$Sim(q_{and},d_j)= \min(x_i)$$

Boolean query processing is at the heart of most commercially available retrieval systems (Takkinen, 1996). The Boolean search strategy is integrated with most search engines on the Web today because it is fast and can therefore be used online. Some of the search engines that integrate this strategy are: Lycos, Infoseek, Alta Vista, Excite, and Magellan (Lager, 1996).

The extended Boolean strategy is used for the system in this research due to its simplicity for users with *match on all words* or *match on any words* and because the Boolean search is known to be fast way of searching especially for document collections of large size like those on the Web.

3.4.2 Vector Model (Vector Space Model)

The vector model regards index terms as the coordinates of a multidimensional information space (Salton & McGrill, 1983). The vector model introduces the possibility of partial matching. Terms are assigned non-binary weights in queries and in documents. The degree of similarity between the documents in the document space and the user query are computed. The output will then be a ranked set of documents in order of decreasing the degree of similarity.

For $w_{i,q}$ being the weight associated with the pair $[k_i,q]$. The query vector q is defined as

$$q=(w_{1,q}, w_{2,q},\ldots,w_{t,q}).$$

The vector of a document dj is represented

$$dj=(w_{1,j}, w_{2,j},\ldots,w_{t,j})$$

where t= total number of index terms in the system

Therefore a document d_j and a user query q are represented as t-dimensional vectors as

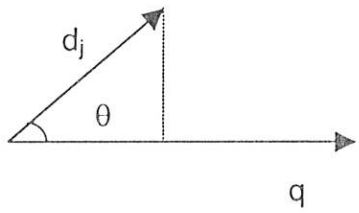


Figure. 3.3 Cosine of θ Taken as *sim(d_j,q)*

The degree of similarity between a document and a query q is evaluated as

$$\text{Sim}(d_j,q) = d_j \bullet q / |d_j| x |q|$$

$$= \Sigma\ w_{i,j} \times w_{i,q} / (\sqrt{(\Sigma\ w^2_{i,j})} \times \sqrt{(\Sigma\ w^2_{i,q})})$$

Index term weight can be calculated in any ways discussed in subsections above.

The main advantages of the vector model are (Baeza-Yates & Ribeiro-Neto 1999):

- its term weighting scheme allows better retrieval performance,
- its partial matching ability allows to retrieve documents that approximately match the query
- its ranking formula allows to sort documents according to their degree of similarity to the query.

The vector space model, however, has some shortcomings like the need for many query terms for discriminating among documents, which is very easily done by the use of AND

in the Boolean model. Synonymic and phrasal relationships which are easily handled by OR and AND in Boolean model are difficult to specify in vector model (Salton & McGrill, 1983).

3.4.3 Probabilistic Model

The probabilistic model attempts to solve the IR problem within a probabilistic framework. The whole idea is, given a user query there is a set of documents which contains exactly the relevant documents and no other. An interaction with the user is initiated to improve the probabilistic description of the ideal answer set of relevant documents. The process goes on in such a way that the user judges the documents retrieved for relevance and the system uses the user feed back to refine the ideal answer set. This process is repeated until the ideal answer set becomes close to the real answer set (Baeza-Yates & Ribeiro-Neto, 1999).

Conditional probabilities are used in calculating term weights.

Pt = P(term t is present in document | document is relevant); and

Qt = P(term t is present in document | document is non relevant)

The following presents the frequency table used to estimate conditional probabilities (Salton & McGrill, 1983).

Term t	Relevant	Non-relevant	Total
Present	R	n-r	N
Absent	R-r	N-n-(R-r)	N-n
Total	R	N-R	N

Table 3.2 Frequency Table to Estimate Conditional Probabilities

Where N is the number of documents in the collection

R is the number of relevant documents for query q

r is the number of relevant documents for query q having term t

n is he number of relevant documents in the collection having term t

The probability that term t is present in a relevant document and the probability that term t is present in a non relevant document are estimated by (r/R) and (n-r)(N-R) respectively.

The probabilistic model makes an initial guess to separate relevant and non-relevant documents and assumes all terms in a document have a binary weight. I.e. there is no consideration of their frequency (Baeza-Yates & Ribeiro-Neto, 1999).

CHAPTER FOUR

EXPERIMENTATION

4.1 INTRODUCTION

In this chapter, the creation of database, Web page submission form design, search interface design and automatic indexing design and implementation for Amharic Web documents are presented. The properties of the Amharic language and the information retrieval concepts discussed in chapters two and three are taken into consideration in the implementation.

To store the content and address of the Web pages, a database is created and the contents of the Web documents are indexed. A form that allows Web page submission and interfaces for query input and retrieval of documents are also designed. The effectiveness of the prototype system is finally, tested against recall and precision.

In the prototype system developed, two inverted index files are generated. One of the files contains full words as index terms, while the other contains index terms with prefixes and suffixes removed from them.

4.2 DESCRIPTION OF THE PROTOTYPE SYSTEM

The prototype Amharic information retrieval system designed has the components described in Figure. 4.1. The different components are designed to meet the requirements that are identified necessary for the system.

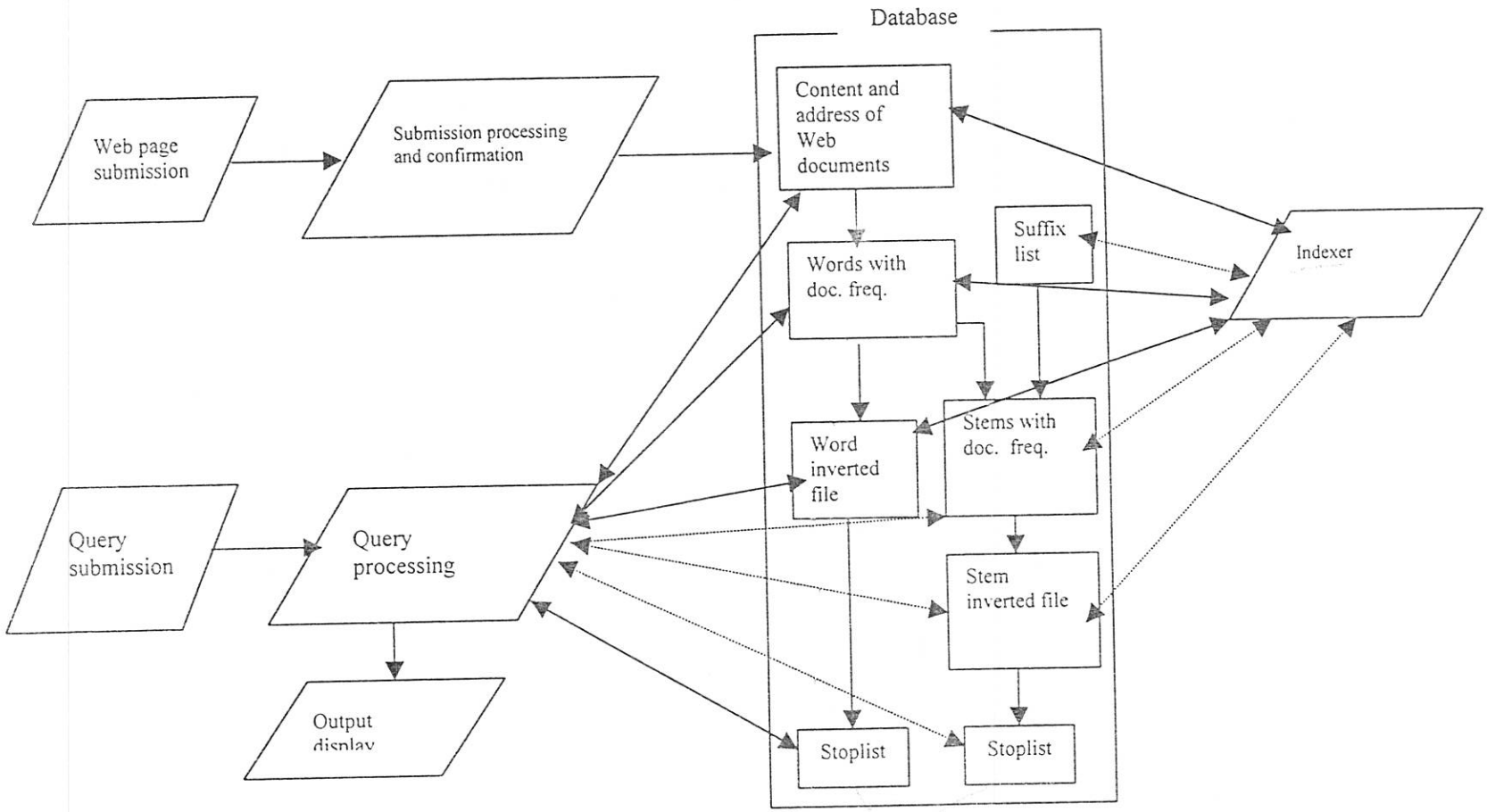


Figure. 4.1 General Description of the Prototype IR System Components

Web Page owners submit their Web pages to a database where they can be indexed and made available for searching. In addition, the database consists of a suffix list that contains a total of 69 concatenated and individual suffixes. Word and stem inverted indexes are generated from the above files. Lists of stopwords are obtained by a procedure discussed in section 4.5.3 from the two inverted indexes.

A query input interface that allows extended Boolean searching with *all words* or *any of the words* in the query is designed. A ranked output of documents containing the title of each document retrieved with a link to the Web page, and the top 150 characters of the document is displayed after processing a query.

The detail of the prototype system design is discussed in the following sections.

4.3 DATABASE CREATION

A database has been created using Microsoft Access 97. The database consists of files that contain data that are relevant for the application of information retrieval techniques that were integrated with the system.

Bibliographic information of the Web documents includes the title, the URL and the abstract (for the experiment in this study the news article itself is taken). An ID (DId) is assigned to the record the moment a Web page is submitted. The data description of this file is the one below:

tblarticle

Attribute	Data type
DId	*Autonumber*
Title	*Text*
Article	*Memo*
URL	*Text*

Table 4.1 Data Description of Web Pages Submitted

DId	Article	URL	Title
186	አድዋ ህሳስ 1/1993/ዋኢማ/ የሻዕቢያ መንግሥት በርካ እናቶችንና ሕፃናትን ከተለያዩ ከተሞች ሕዳር 28 ስብስባ ዓላ በተባለ በረሐማ ሥፍራ እንዳሳራቸውና	http://telecom/news/01049314.htm	ሻዕቢያ በእንዳኪማንዶ ማሳያ እናቶች፣ እመጫትና ሕፃናትን እያስቃየ
187	ወልዲያ ህሳስ 1/1993/ዋኢማ/ ከአንዲት ወጣት 300 ብር በመቀበል ሕገ - ወጥ ውርጃ በመፈፀም ሕይወቷ እንዲያልፍ አድርጓል ያለውን ተጠርጣ በቁጥጥር ሥር	http://telecom/news/01049315.htm	የሕገ - ወጥ ውርጃ በማካሄድ ለአንዲት ወጣት ሕይወት መጥፋት ምክንያት

Table 4.2 Sample Data in 'tblarticle'

The suffix list consists of lists of suffixes, concatenated and non-concatenated (basic suffixes). Amharic has suffixes that can be attached to verbs to indicate tense, person, number, gender, position and polarity. There are also suffixes to nouns that are either plural markers, possessive pronouns, object markers or conjunctions with attributes of gender, number and person (Abiyot, 2000).

A number of inflected words are also formed by concatenating the suffixes. The list of the basic suffixes and concatenated suffixes are obtained from those used by Abiyot in designing Amharic word parser. However, infixes that exist in Amharic words and

some prefixes and suffixes that rarely exist are not considered. The data description of the suffix list is:

tblsuf

Attribute	Data type
suffix	*Text*

Table 4.3 Data Description of the Suffix List

Suffix
ና
ናና
ያች
ም፡
ም፡ና
ም፡ናም
ም፡ናን

Table 4.4 Sample Data in 'tblsuf'

Prepositional prefixes that are frequently used with both nouns and verbs (የ, በ, ለ, ከ) are also removed. As they are few in number it was not necessary to store them in a file.

The other files that exist in the database are generated from the above mentioned database files. A file 'tblwords' with the data description shown below is designed.

tblwords

Attribute	Data type
Dld	*Text*
Word	*Text*
Freq	*Number (long integer)*

Table 4.5 Data Description of Terms with Their Document Frequency

Did	Word	Freq
1	ኢንፎርሜሽን	1
1	ማዕከል	1
2	ተቋርጦ	4

Table 4.6 Sample Data in 'tblwords'

The data in 'tblwords' is obtained from 'tblarticle'. The document frequency (Freq) in 'tblwords' is necessary for the calculation of term weights that would be used for ranking during retrieval. In fact 'tblwords' was built primarily for this reason. *Freq* is the frequency of the term in the document with reference number *Dld*. The terms are taken from the Web documents is 'tblarticle' and the frequencies of the terms are obtained by counting the number of times each term exists in a document.

The inverted file 'tblindex' is built from 'tblwords'. The index file contains terms with the list of their occurrences. The frequency 'TotFreq' is the frequency of each term in the total collection. The total collection frequency is calculated to identify terms of high frequency in the collection. This helped in cutting off the high frequency terms

that need to be removed from the index file being recognized as words with no content. Those terms have been included in the list of stopwords, 'tblstop'.

tblindex

Attribute	Data type
DId	*Memo*
Word	*Text*
TotFreq	*Number (long integer)*

Table 4.7 Data Description of Word Inverted File

Word	Frequency	DId
የልማት	73	341,339,335,335,327,326,324,321,316,316,314,284,284,
የኢትዮጵያ	72	341,341,339,339,337,332,321,312,290,289,286,277,265,
ማንበር	72	373,341,339,322,312,287,286,272,265,265,261,260,255,

Table 4.8 Sample Data in 'tblindex'

tblstop

Attribute	Data type
Word	*Text*

Table 4.9 Data Description of Word Stoplist

Word
እየ
ላይ
ብር
ማዕከል
ቤተ
ሸ
ኢንፎርሜሽን
እና

Table 4.10 Sample Data in 'tblstop'

Suffixes and prefixes are removed from terms in 'tblwords' and those terms are placed in 'tblstem'. The frequency of each term in each document is also included, which is later used in determining term weights for ranking. The data description is:

tblstem

Attribute	Data type
DId	*Text*
Stem	*Text*
Freq	*Number (long integer)*

Table 4.11 Data Description of Stems with Their Document Frequency

stem	sfreq	sdid
ግዕከል	1	1
ትምህርት	14	2
ተጽርጠ	4	2

Table 4.12 Sample Data in 'tblstem'

From 'tblstem' is generated the inverted file 'tstem'. The data description of tstem is:

tstem

Attribute	Data type
stDId	*Memo*
Stem	*Text*
StTotFreq	*Number (long integer)*

Table 4.13 Data Description of Stem Inverted File

stem	tsfreq	tsdid
ካቲ:ች	72	310,225,209,181,171,97,96,95,94,93,92,91,90,89,88,87,8
ቢሮ	71	366,342,329,327,323,320,318,314,285,282,279,269,267,2
ድጋፍ	71	371,369,341,340,339,325,318,314,287,286,280,279,266,2

Table 4.14 Sample Data in 'tstem'

After determining non content bearing terms from the inverted file, a list of stopwords 'tblstopstem' has been generated.

tblstopstem

Attribute	Data type
Stem	*Text*

Table 4.15 Data Description of Stem Stoplist

stop
ላይ
እኖ
ብር
.ኮ
ማዕከል
ሸ.
ዋል
እንፎርማሺሸ

Table 4.16 Sample Data in 'tblstopstem'

The IDs in the inverted files have a data type memo. This is because they consist of the list of occurrences of the terms separated by a comma. The list may be large for terms found distributed in many documents.

4.4 WEB PAGE SUBMISSION

Web pages are submitted into 'tblarticle' discussed in section 4.3 by Web site owners that desire to make their Web pages available to information seekers.

A Web page submission form has been developed using ColdFusion 4.0 studio to allow the submission of these Web pages. On the interface are input boxes that request to enter the *Title*, *abstract*, and *URL* of the Web page. The page submitted is instantly assigned an ID. The ID (*Did)* in 'tblarticle' has an AutoNumber data type and hence adds its value as the page is submitted. The Web page submission form 'Submitform.cfm' looks like the one below:

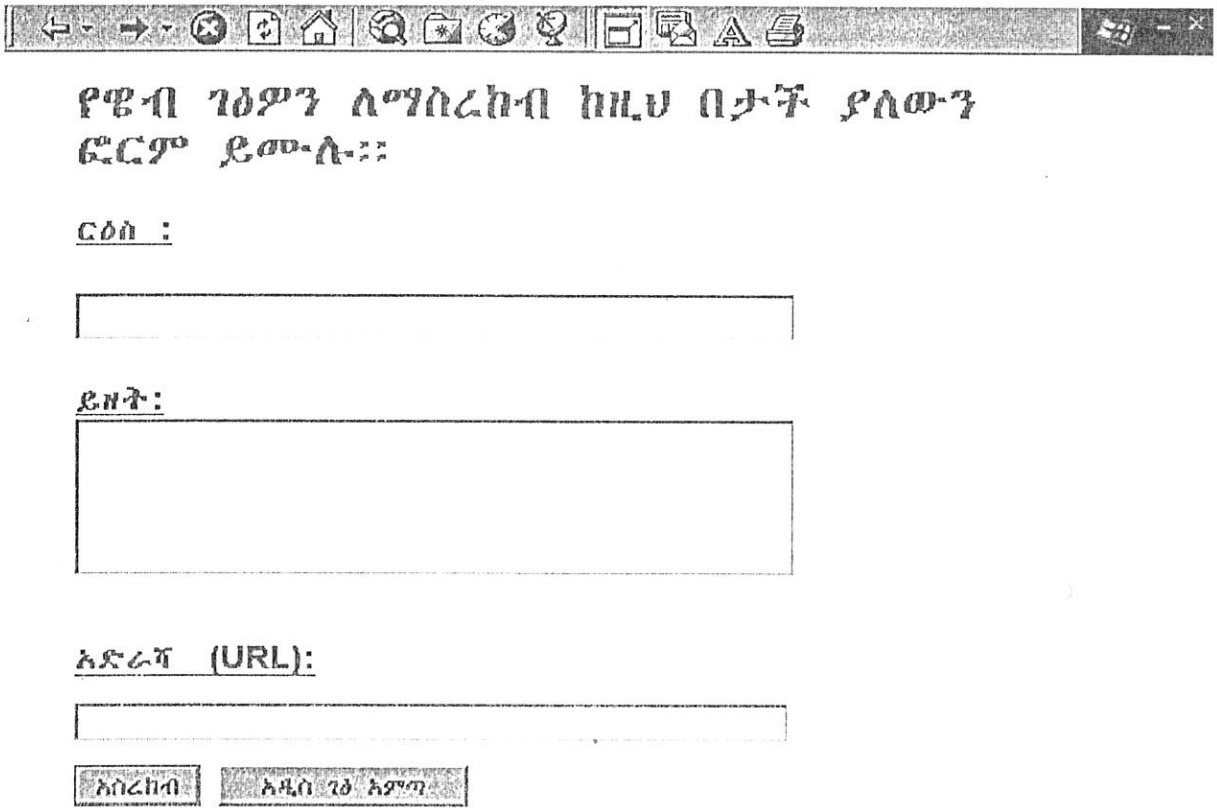


Figure. 4.2 Web Page Submit Form

The title (ርዕስ) and abstract (ይዘት) are to be input in Amharic while the URL is in English. With the press of the 'አስረክብ' (submit) button the value of the input boxes is passed to a second form 'Submitprocess.cfm' where the submission to the database actually takes place. እዲስ ገፅ አምጣ is equivalent to *reset* in English.

Server-side form validation is performed to control that the form be filled completely. A Null value is not accepted as all the three attributes are important. Hence, for example a message "Title is required!" is sent to the user if the user tries to put a null value for 'Title'.

```
<u><FONT FACE="VG2 MAIN"> R:S</FONT>:</u><br>
<input name="Title" size="40" maxlength="50" STYLE="FONT-FAMILY: VG2 main" >
<input type = "hidden" name="Title_required" value = "Title is required!">
```

The 'submitprocess.cfm' form accepts values from the interface and inserts them to the corresponding attributes in 'tblarticle'.

```
<cfquery datasource="source">
insert into document (Title, Article, URL)
values ('#Title#', '#Article#', '#URL#')
</cfquery>
```

A confirmation page like the one below is then displayed to confirm the submission of the page.

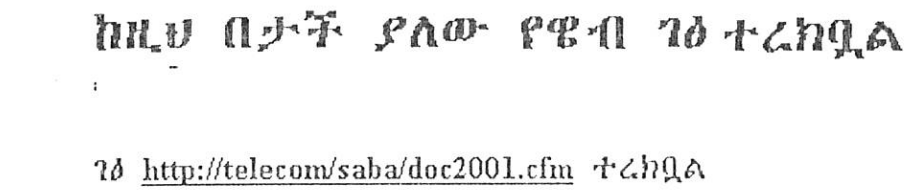


Figure. 4.3 Web Page Submission Confirmation

4.5 AUTOMATIC INDEXING

Scanning every document in a large collection like the Web is not advisable in modern information retrieval. Hence in this study indexing the Web documents (representatives of the documents) has been taken as a better option. The documents are indexed to produce index terms with the reference numbers of the documents they exist in.

For the purpose of this experiment, contents of Web pages containing only news articles in Amharic were inserted in the database.

The documents have been preprocessed to prepare them for the application of information retrieval techniques as discussed below.

4.5.1 Document Preprocessing

Web pages that contain news in Amharic published by Walta Information Center are taken as a test document collection. The pages included for the experiment are 313 in number. The Actual Web pages are available in a Web site, where they can be accessed by the user in response to their queries. It is these documents that are submitted to a database where they are indexed.

4.5.1.1 Selected document part

The nature of the Web documents collected is that they have titles and the article itself. To obtain index terms that could represent the news articles, a choice of what part to index had to be made. Alternatives available were to take:

- the title,
- certain number of words or sentences from the body of the article, or
- the entire article.

Among the alternatives indicated above, the entire article was preferred for the reason that it is better representative of the content, and the articles are of a size mostly around half a page, a few full page and very few past a page which was an encouragement that it won't be taking much space to index or to store the entire article.

4.5.1.2 Identification of index terms

Lexical analysis has been made to identify words in the Amharic text collections. Characters that exist with words but do not take part in the semantics of the words have been removed in this process.

Punctuation marks were removed at this stage. In written languages, punctuation marks are used to clarify meaning, to help convey emphases and breathing pauses, to indicate sentence structure, and to enhance readability (Microsoft® Encarta® Encyclopedia 99). But these marks do not have a meaning that is related to the contents of documents. Hence in this research they are removed with the

exceptions of hyphen, period and slash for the reason discussed later in this section (-,. &/).

Punctuation marks vary from language to language and preferences for specific marks vary from writer to writer. In Amharic there are punctuation marks that are of Amharic and some that are borrowed from Latin (Beletu, 1982). Writers may only use some of those punctuation marks as they prefer. The punctuation marks dealt with in this research are those that exist in the VG2 Amharic writing software, because they are the only ones writers using this software can make use of.

Those punctuation marks are :: : ፣ ፤ - ? / . « »

:: (equivalent to period in English)

: (used to separate words, but not used by most writers of modern Amharic and is not used in the news articles collected for this research)

፣ (equivalent to comma in English)

፤ (equivalent to semi-colon in English)

- (equivalent to hyphen in English)

? (equivalent to question mark in English)

/ (equivalent to slash in English, usually used between letters in Acronyms, in telephone numbers, in identification numbers and the like)

. (used between letters in acronyms)

« (equivalent to double quotes in English, used before the quoted

part)

» (equivalent to double quotes in English, used after the quoted part)

For the removal of punctuation marks that exist at the end of words (: ; ? ») the following code is used:

```
If Mid(article, n - 1, 1) = "½" Or Mid(article, n - 1, 1) = "¿" Or Mid(article, n - 1, 1) = "'" Or Mid(article, n - 1, 1) = ")" Then

    article = Mid(article, n + 2)
Else
    article = Mid(article, n + 1)
```

The punctuation mark (::) which is made of two :s and exist at the end of words is removed using the code below:

```
If Mid(article, n - 1, 1) = "Ý" Then

    article = Mid(article, n + 3)
Else
    article = Mid(article, n + 1)
```

The one that could exist at the beginning of words («) is removed as follows:

```
If Mid(article, n + 1, 1) = "«" Then
        article = Mid(article, n + 2)
Else
        article = Mid(article, n + 1)
```

Not all punctuation marks in the documents are removed. The punctuation marks /(slash),. (period) and – (hyphen) are left as part of the index terms. These marks occur between letters or between words and to leave them will allow the retrieval of words that contain them. In case of hyphenated words, the words are included into the index file as a single term. However, in some cases writers insert space before and after the hyphen. Those are considered as separate words. So only those with no such spaces are considered as one term.

Numbers were not removed hoping they may increase recall and because they were not found disadvantageous as numbers in the collection were not many.

4.5.2 Algorithm Of Automatic Indexer

The index terms were placed in two files in light of determining the document and collection frequencies of each term. These two frequencies are important in calculating the weight of the terms in order to rank the documents during retrieval and in identifying non content bearing terms respectively. The code fragment used in developing these two files from the 'tblarticle' relation containing the articles is as shown below:

Start

Set the workspace and the database containing the Articles

```
Set Ws = DBEngine.Workspaces(0)
Set Db = Ws.OpenDatabase(App.Path & "\news.mdb")
```

Set a record set from 'tblArticle'

```
sqltext = "select * from tblArticle"
Set Rs = Db.OpenRecordset(sqltext)
```

Move to the beginning of the first record and assign the value of the Article field and the value of the DId field to the variables *article* and *DId* respectively

```
Rs.MoveFirst
While Not Rs.EOF
        Article = Rs!Article
        DId = Rs!DId
```

Start reading from the beginning of the article. Remove any punctuation marks (in this case only «) before the first word in the article and assign the remaining part of the article to the variable *article*.

```
If Mid(article, n + 1, 1) = "«" Then
        article = Mid(article, n + 2)
Else
        article = Mid(article, n + 1)
```

Start reading from the beginning of the article until you get a space and remove punctuation marks at the end of the word and put the string read into an array.

If the word exists in the array previously only increase the value of the frequency of the word by 1.

```
If Mid(article, n - 1, 1) = "½" Or Mid(article, n - 1, 1) = "¿" Or Mid(article, n - 1, 1) = "'" Or Mid(article, n - 1, 1)
= ")" Then

    Loop Until Arrword(j - 1) = Mid(Article, 1, n - 2) Or j = i


     If Arrword(j - 1) = Mid(Article, 1, n - 2) Then
       ArrFreq(j - 1) = ArrFreq(j - 1) + 1

Else
        Arrword(i) = Mid(Article, 1, n - 2)
        ArrFreq(i) = 1
```

Remove the punctuation mark (::) if at the end of the word

```
        If Mid(article, n - 1, 1) = "Ý" Then

          article = Mid(article, n + 3)
        Else
          article = Mid(article, n + 1)
        End If
```

If no punctuation marks are found at the end of the first word in the article, read until you get a space and simply put the string read into an array.

If the word exists in the array previously only increase the value of the frequency of the word by 1.

```
Loop Until Arrword(j - 1) = Mid(Article, 1, n - 1) Or j = i


If Arrword(j - 1) = Mid(Article, 1, n - 1) Then
        ArrFreq(j - 1) = ArrFreq(j - 1) + 1
```

```
Else
        Arrword(i) = Mid(Article, 1, n - 1)
        ArrFreq(i) = 1
```

Remove the punctuation mark («) before the first word in the remaining part of the article using the same code that is used to remove («) above and assign this value (the remaining article) to the variable *article*.

Repeat the above procedures until the end of the file.

Insert the values of the arrays *Arrword(i)* and *ArrFreq(i)* into the attributes *word* and *Freq* in 'tblwords'.

```
.AddNew
!word = Arrword(n)
!frequency = IIf((ArrFreq(n)) = "", 0, ArrFreq(n))
!DId = DId
.Update
```

From the 'tblwords' relation put the first value of the *word* attribute to the *word* attribute in the relation 'tblindex' and insert its frequency and document ID to the *Totfreq* and *Did* attributes respectively.

```
.AddNew
!word = Rss!word
!frequency = Rss!frequency
!DId = Rss!DId
.Update
```

Read the second value of the attribute *word* in the 'tblwords' relation and compare it with the value (values later on) already in the word attribute of the 'tblindex' relation.

If the values are the same, add the frequency of this term to the one in the 'tblindex' relation and append its *DId* after a comma to the *DId* in the 'tblindex' relation. Continue doing this until it is the end of the 'tblwords' file.

```
    !frequency = Rss!frequency + !frequency
!DId = Rss!DId & "," & !DId
```

The 'Tblstem' relation is built from the 'tblwords' relation after removing the prefixes and suffixes from the index terms.

If the length of the index term is greater than 2, search for the longest match of suffix from the suffix list "tblsuf".

```
sqtext = "select * from tblsuf"
   Set RsSuf = Db.OpenRecordset(sqtext)
For q = 3 To m
            RsSuf.MoveFirst
        While Not RsSuf.EOF
          If Mid(Term, q, m) = RsSuf!suf Then
            Stem = Mid(Term, 1, q - 1)
Else
           RsSuf.MoveNext
          End If
```

Remove the prefixes (ף, ה, ל, ה) if found at the beginning of the index terms.

```
If Mid(Stem, 1, 1) = "k" Or Mid(Stem, 1, 1) = "y" Or Mid(Stem, 1, 1) = "b" Or Mid(Stem, 1, 1) = "l"
Then
        Stem = Mid(Stem, 2, q)
Else
        Stem = Stem
```

The inverted file 'tstem' is generated from 'tblstem' the same way 'tblindex' is generated from 'tblwords'.

4.5.3 Identification of Non Content Bearing Words

As discussed in chapter three of this paper, non content bearing words can be removed from documents by two common methods: by removing high and low frequency terms from the document collection or by using a negative dictionary that has been prepared in advance for that language.

The first alternative was considered for this research because there is no negative dictionary for Amharic words that was developed previously that is accessible and developing it is out of the scope of this research work.

Hence, documents in the document collection were indexed and the index terms sorted in the order of decreasing frequency of occurrence.

The token to type ratio (number of running words divided by the number of distinct words) obtained is as shown below:

Term	Length of text	Distinct words	Ratio
Word	48058	13949	3.44
Stem	48058	9941	4.83

Table 4.17 Token to Type Ratio of Words and Stems

This ratio is not big when compared with that of English that is discussed in chapter two. Inspection of terms sorted in descending order showed a maximum frequency of 497 which goes down to a minimum of 1. After observing non content bearing words, the upper threshold was decided to be a frequency of 161. Low frequency words obtained during indexing were many and a significant number of them were with content. The highest frequency for suffix and prefix stripped terms was 555. Terms with a frequency of 169 and above were considered as non content bearing words.

Some of the words with a high frequency are:

Word	Frequency
እኖ	497
ላይ	388
ብር	307
ማዕከል	289
ቤተ	278
ሸ	270
ኢንፎርሜሽን	269
እና	243
ፌን	226
ዓመት	219

Table 4.18 Sample Data of High Frequency Words

During this indexing process two problems were encountered in using the Amharic software. These problems are:

1. The character ፡ (¬ in English) and the character ሪ (¶ in English) were not copied from the articles to the words in 'tblwords' relation. If there existed a word with any of these letters, it was taken after removing the character.

2. The second problem was that the character ሥ (| in English) was not read as a string value by visual Basic and it displays an error message. This problem was also encountered when the character was found in query terms while searching. So to handle this case during indexing all the ሥ characters in the articles in 'tblarticle' relation were substituted by another character (ስ) which serves the same purpose (some characters in Amharic have alternative characters).

4.6 SEARCHING FOR INFORMATION

Documents being available somewhere the users or information seekers are provided with a tool to search and find the information they need. For this purpose a query input interface that looks like the one below was designed.

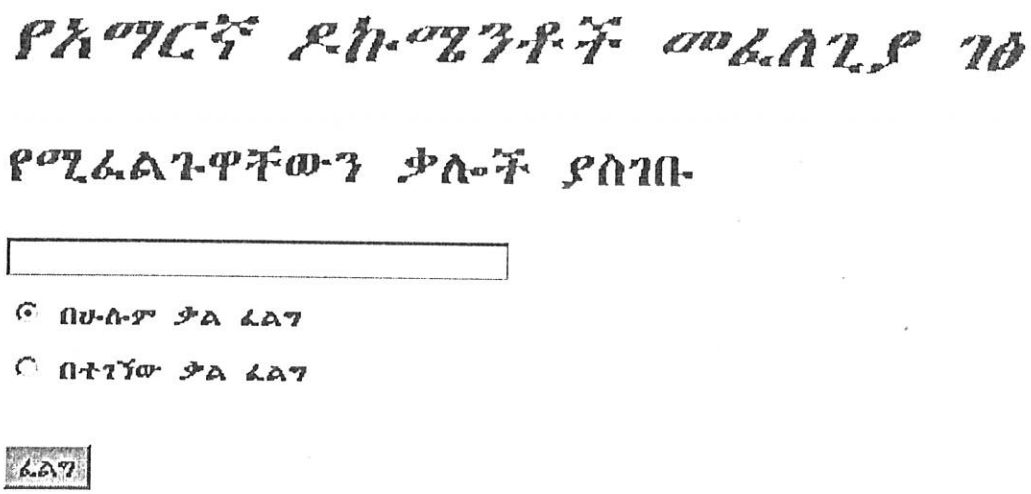


Figure 4.4 Query Input Interface

Query terms will be entered in the input box and there are two options offered for extended Boolean searching. One can choose to search with all terms in the query (AND) or has the alternative to search with any of the terms in the query (OR). By default search is performed with all the terms in the query.

The following code shows how query terms are received and validated.

```
<CFFORM action = "SearchOutputInterface.cfm" method = "post" >
<input NAME="keyterm" TYPE="text" STYLE="font-family: VG2 main">
<input type = "hidden" name="keyterm_required" value = "Key term is required!">
<br><br><input type="submit" value="ĩLG" STYLE="font-family: VG2 main">
```

```
</CFFORM>
```

Null value is not valid hence the message "*Key term is required!*" is sent to the user if the user tries to search a null value.

Each of the query terms are searched in the 'tblindex' relation, a word at a time.

```
<!--- Retrieve document ids corresponding to the query term --->

<cfquery name="GetResults" DATASOURCE="data" cachedwithin="#createtimespan(0,0,8,0)#">
select tblindex.DId as ind, tblindex.word as term
from tblindex
<!--- The cfloop in the where statement helps to make a search for each word in the input keyterms and
retrieve documents for each term using the AND operator--->
WHERE (0=1
<cfloop list="#keyterm#" index="thisword" delimiters=" ">
or (word like '#thisword#')
</cfloop>)
and word not in (select stop
                 from tblstop

                 WHERE (0=1
                 <cfloop list="#keyterm#" index="thisword" delimiters=" ">
                 or (stop like '#thisword#')
                 </cfloop>))
</cfquery>
```

The cfloop in the where statement treats the query terms as elements of a list and makes a search for each element in the list using the OR operator. The document identifiers retrieved corresponding to a single term could be a list of document IDs or a single value. Because a single term for instance could be identifier of many documents like:

Term1 d3,d6,d8,d19,d37,d62,d91

Or a term may only be identifier of a single document like

Term2 d9

If a term in the query is a stop word, the subquery identifies it and is excluded from the query.

Each element in the lists obtained in the query 'getresults' above is then compared with the document IDs in the 'tblarticle' relation.

When searching with all words, only documents containing all the terms in the query are considered as the possible relevant documents.

```
<!--- Retrieve document title and URL and a part of the article for the SELECT box--->
<cfquery name="final" datasource="data" cachedwithin="#createtimespan(0,0,8,0)#">

select tblwords.DId as sdid, sum(tblwords.frequency) as freq, ( select url
                                                                from tblarticle
                                                                where  tblarticle.did=tblwords.did) as ur,
                                                                ( select { fn left( article,150)}
                                                                 from tblarticle
                                                                where  tblarticle.did=tblwords.did) as art,
                                                                ( select { fn left( title,50)}
                                                                from tblarticle
                                                                where  tblarticle.did=tblwords.did) as title1
from tblwords
<!--- The cfloop in the where statement helps to make a search for each word in the input keyterms and
retrieve documents for each term using the AND operator--->
WHERE (0=1
<cfloop list="#keyterm#" index="thisword" delimiters=" ">
or (word like '#thisword#')
</cfloop>)
and tblwords.did in
                    (select did
                    from tblarticle
                    where (0=0
                            <cfloop query="getresults">
                    and ({ fn convert( tblarticle.did,sql_varchar)} like '#getresults.ind#' or (0=1
                    <cfloop list="#getresults.ind#" index="thisw" delimiters="," >
                    or { fn convert( tblarticle.did,sql_varchar)} like '#thisw#'
                    </cfloop>))
                    </cfloop>))
group by tblwords.did
order by sum(tblwords.frequency) DESC
</cfquery>
```

Because document.DId is an AutoNumber it has to be changed into a string for comparison with the elements in the lists that have a memo data type.

```
{ fn convert( tblarticle.DId,sql_varchar)}
```

The retrieved documents are ranked in order of extended Boolean ranking discussed in chapter three (taking a p value of 1). The term weighting scheme considered is the

term frequency weight. It was found reasonable to use the term frequency weight because the Web page documents that are used for the experiments in this research in particular are of similar size and since high frequency function words are already eliminated as stopwords.

The search output is then displayed in a screen in the form of the one below.

ለ 'ነጋዴ ሴቶች ማኅበር ' 3 ዶክመንቶች ተገኝተዋል::

የጅማ ከተማ ነጋዴ ሴቶች ማኅበር አቋቋሙ

ጅማ ሕዳር 1/1993/ዋኢማ/ የጅማ ከተማ ነጋዴ ሴቶች ራሳቸውን በኢኮኖሚ ለማጠናከር የሚያስችል ማኅበር አቋቋሙ፤ 94 ሴቶች ኑሯቸውን አሻሽሉ:: በጅማ ከተማ በንግድ እና በዕደ - ጥበብ ሥራ የተሠማሩ 150 ሴቶች ካለፈው

ግሉውያን ነጋዴዎች ከኢትዮጵያውያን ጋር ተባብረው ለመሥራት ፈቃደኛ ናቸው ተባለ

አዲስ አበባ የካቲት 2/1993/ዋኢማ/ የግክ የንግድ ማኅበረሰብ ከኢትዮጵያዊያን ጋር ተባብሮ ለመሥራት ፍላጎት እንዳለው መግለፁን በትርቡ አገቱን ጎብኝተው የተመለሱ ኢትዮጵያዊያን አረጋገጡ:: ከኢትዮጵያ የንግድ ማኅበ

በማኅበር የተደራጁ ደላሎች ችግር ያጋጠማቸውን አባላት እስከ መርዳት ደረሱ

ጎንደር የካቲት 8/1993/ዋኢማ/ በተናጥል ይንቀሳቀሱ የነበሩ የጎንደር ከተማ ደላሎች /ኮሚሽን ሠራተኞች/ በማኅበር በመደራጀታቸው ራሣቸውን ችለው ችግር ላጋጠማቸው የማኅበሩ አባላት ድጎማ ማድረጋቸውን አስወቁ:: አባላቱ የማኅ

The output screen contains:

- The number of documents found for the query
- The titles of the documents are retrieved with a link to the actual document address (URL).
- A part from the article (first 150 characters) is retrieved and displayed.

The maximum number of results per page is limited to 10. With the click of the *NEXT10* (የሚቀጥሉት 10) and *PREVIOUS10* (የቀደሙት 10) buttons the rest of the search results can be obtained.

```
select title, { fn left( article,150)} as art
        .
        .
        .

<cfoutput maxrows = #maxrows# startrow = #start# query = final>
<br><li><font size="+1"><A href="#url#"><b><font face="VG2
Main">#title#</font></b></A></font>
<br><dd><font size="-1"><font face="VG2 Main">#art# </font><br></font>


<cfset maxrows =10>
<cfparam name="start" default="1">
        .
        .
        .
<cfset  prevstart = start - maxrows>
<cfset  nextstart = start + maxrows >
<cfif prevstart GTE 1 >

<cfoutput>
<form action="searchoutputinterface.cfm" method="post">
<input type="hidden" name="start" value ="#prevstart#">
        .
        .
        .

<input type= "submit" value="previous  #maxrows#">
</form></CFOUTPUT></CFIF>
<cfif nextstart LTE final.recordcount >
<cfoutput>

<form action="searchoutputinterface.cfm" method="post">
<input type="hidden" name="start" value ="#nextstart#">



<input type= "submit" value="next  #maxrows#" style="HEIGHT: 24px; WIDTH: 99px">
</form></CFOUTPUT></CFIF></dd></li>
</body>
</html>
```

As shown in the *getresults* and *final* queries the output of search results are placed in memory for 10 minutes to allow the display of 10 records at a time.

Prefixes and longest suffix are removed from the query terms using the following code:

```
<cfquery name="stem" DATASOURCE="data">
select suf
from tblsuf
</cfquery>

<cfset sr="">
<cfset st=valuelist(stem.suf)>
<cfloop list="#keyterm#" index="stword" delimiters=" ">
<cfset d =0>
<cfset leng=#len(#stword#)#>

<cfif  #leng# gt 2>
<cfloop index="d" FROM="3" TO="#leng#" STEP="1">
<cfif #listcontains(st, "#mid(stword,d,leng)#")# gt 0>
<cfset ss=#mid(stword,1,d-1)#>
<cfset ln=#len(#ss#)#>
<cfset prefix="y,k,l,b">
<cfif #listcontains(prefix, "#mid(ss,1,1)#")# gt 0>
<cfset sps=#mid(ss,2,ln)#>
<cfbreak>
<cfelse>
<cfset sps=#ss#>
</CFIF>

<cfelse>
<cfset ss=#stword#>
<cfset ln=#len(#ss#)#>
<cfset prefix="y,k,l,b">
<cfif #listcontains(prefix, "#mid(ss,1,1)#")# gt 0>
<cfset sps=#mid(ss,2,ln)#>
<cfelse>
<cfset sps=#ss#>
</CFIF>
</CFIF> </cfloop>
<cfelse>
<cfset sps=#stword#>
</cfif>
<cfset sr=listappend(sr,"#sps#")></cfloop>
```

Experiments carried out to retrieve relevant documents using both word and stem indexes with *all words* and *any of the words* is discussed below. The results obtained with *any of the words* is analyzed and average recall and precision values are presented.

4.7 TEST RESULTS

To determine the performance of the system, queries were formulated for items that are known to exist. Two journalists (Ato Goraw Salilew, Editor & Ato Amanuel Abrha, Reporter) that work at Walta Information Center made 41 queries and decided the relevant documents. The output obtained from 22 (21 having more than one relevant documents and 1 with one relevant document) queries is presented below. The 41 queries and the relevant documents corresponding to each one of them is attached as appendix II.

Table 4.19 contains the twenty-two queries used for the experiment, the number of relevant documents available in the collection corresponding to each query , the total number of documents retrieved for each query (Ret.) and the number of relevant documents retrieved for each query (Rel.). The experiments are conducted by using word and stem index terms and operators of AND and OR.

From the results on the table the recall and precision values obtained for each query are calculated.

Recall is the fraction of relevant document which has been retrieved (Baeza-Yates & Ribeiro-Neto, 1999). I.e.

$$\text{Recall} = \frac{\text{Number of relevant documents retrieved}}{\text{Number of relevant documents in the collection}}$$

Precision is the fraction of the retrieved documents which is relevant (ibid.). I.e.

$$\text{Precision} = \frac{\text{Number of relevant documents retrieved}}{\text{Number of documents retrieved}}$$

No.	Query	Rel. doc.	Stem				Word			
			AND		OR		AND		OR	
			Rel.	Ret.	Rel.	Ret.	Rel.	Ret.	Rel.	Ret.
1	የውሃ ጉድጓዶች ቁፋሮ	7	1	7	7	14	1	6	6	13
2	የንፁህ መጠጥ ውሃ ፕሮጀክቶች	7	0	0	5	29	0	0	4	27
3	በሰደድ እሳት የደኖች መጥፋት	3	0	0	2	7	0	0	1	6
4	የኤድስን ስርጭት መግታት	9	6	7	6	27	2	7	2	8
5	ጠለፋና አስገድዶ መድፈር	4	2	2	2	5	2	2	2	5
6	የስልክ አገልግሎት	4	1	20	1	46	1	15	1	38
7	የወረዳና የቀበሌ አባላት ምርጫ	5	3	3	3	7	1	5	1	6
8	ጎጂ ልማዶች	4	3	8	3	8	2	5	2	6
9	የመስኖ ልማት ፕሮጀክቶች	4	0	3	2	15	0	3	1	7
10	ኢትዮጵያውያን ስደተኞች በሱዳን	2	1	1	2	27	0	0	2	25
11	የቅድመ ጋብቻ የኤድስ ምርመራ	8	3	13	3	14	2	3	3	14
12	ሴቶችና ልማት	8	3	15	4	18	1	2	3	14

13	የማጅራት ገትር ክትባት	3	3	4	3	14	0	3	0	14
14	የትራፊክ አደጋ በአዲስ አበባ	1	0	16	0	24	0	1	0	6
15	የትምህርት በራዲዮ ማሰራጫ	2	2	2	2	18	1	2	1	7
16	በስደት ያሉ ኢትዮጵያውያን	2	0	11	0	22	0	11	0	11
17	ከኤርትራ የተመለሱ ኢትዮጵያውያን	8	2	4	2	13	2	4	2	8
18	ቫዕቢያ ያሰራችው ኢትዮጵያውያን	4	1	3	1	4	1	2	1	3
19	አዲስ የተገነቡ የጤና ተቋማት በኢትዮጵያ	2	0	5	1	21	0	0	0	18
20	የመንግስት ገንዘብ መዝባሪዎች	4	1	12	1	16	1	10	1	12
21	የመንግስት ገንዘብ ምዝበራ	4	1	12	1	14	1	11	1	12
22	ነጋዴ ሴቶች ማህበር	2	0	15	1	24	0	9	0	15

Table 4.19 Test Results for 22 Queries

The query results (with the OR operator) were analyzed to draw the average recall-precision graph after using the smoothing algorithm adapted from Keen (1972) and used by Hmeidi, Kanaan and Evens (1997). The smoothing algorithm is applied as shown below:

a. Distribute the recall values into 10 equal sub-intervals in the interval 0-1
b. Assign the largest precision value beginning in that interval to that interval
c. Assign the largest precision value found in the table to the first interval
d. Beginning from the 10^{th} interval, remove all sawtooth lines by assigning the current interval's precision value to the next interval, if its precision value is lower than the current one.
e. To make sure that the precision vale will drop gradually from a certain precision value to zero value, assign any interval with a zero precision to half of the precision value of the previous interval.

Recall	Stem	Word
0.1	0.5000	0.4615
0.2	0.4286	0.4000
0.3	0.4000	0.3333
0.4	0.3750	0.3333
0.5	0.2857	0.2143
0.6	0.2143	0.1481
0.7	0.1071	0.0800
0.8	0.0536	0.0400
0.9	0.0268	0.0200
1.0	0.0134	0.0100

Table 4.20 Average Recall-Precision Values After Zero-Smoothing

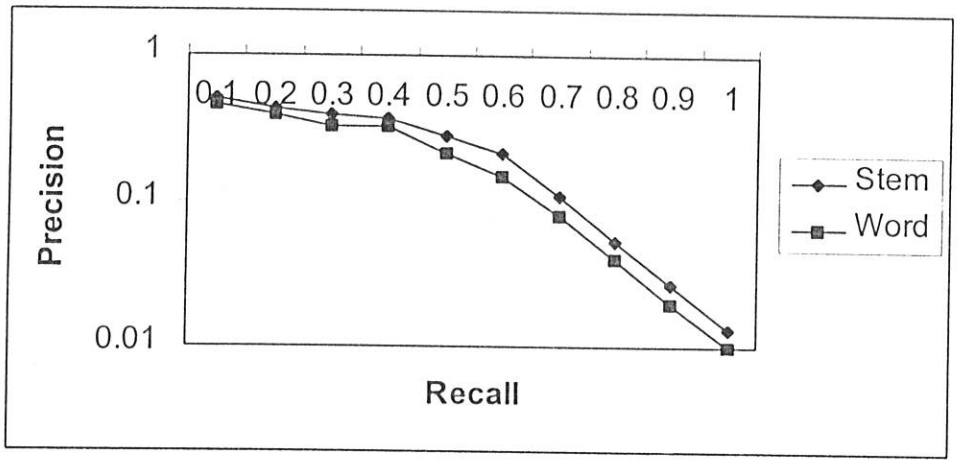


Figure 4.4 Average Recall-Precision Graph After Zero-Smoothing

As can be seen from the graph stems gave a better performance than words.

Submission of Web page data was experimented by submitting ten pages http://telecom/news/0802931.htm – http://telecom/news/08029310.htm into the database. The rest of the pages in the database were entered with out using the submit form to economize time. Though lately it was found that using the form is not any slower.

Before a page can be submitted any ሥ character in the parts of the Web page to be submitted has to be avoided for the reason discussed in the previous sections.

To exhibit the process the submission of http://telecom/news/08029310.htm is demonstrated below:

The title, body of article and the URL of the page are filled in the form below. With the press of the አስረክብ button the contents are submitted into the database and an Id is instantly assigned to the record.

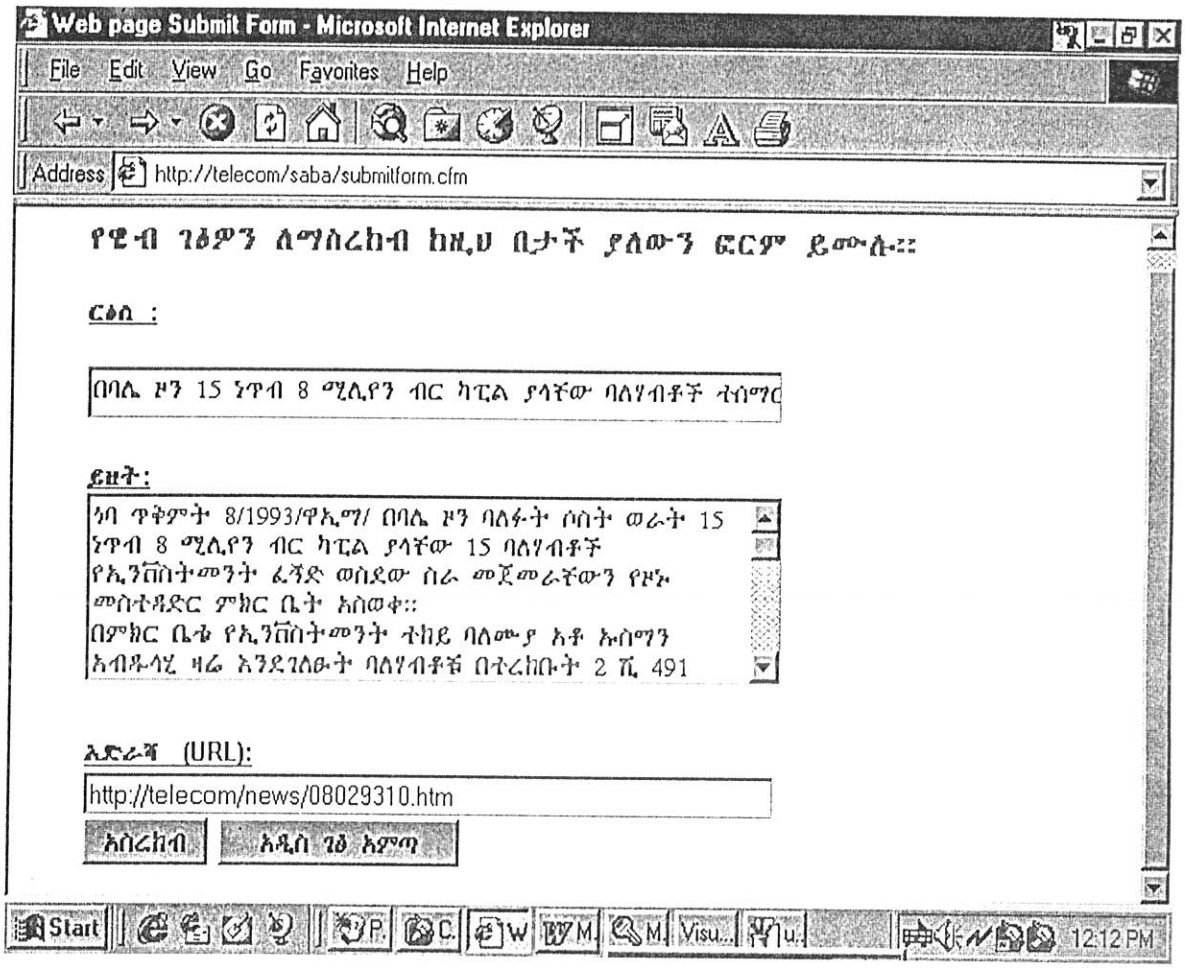


Figure 4.5 Submission of http://telecom/news/08029310.htm

To confirm the submission the message screen below was displayed.

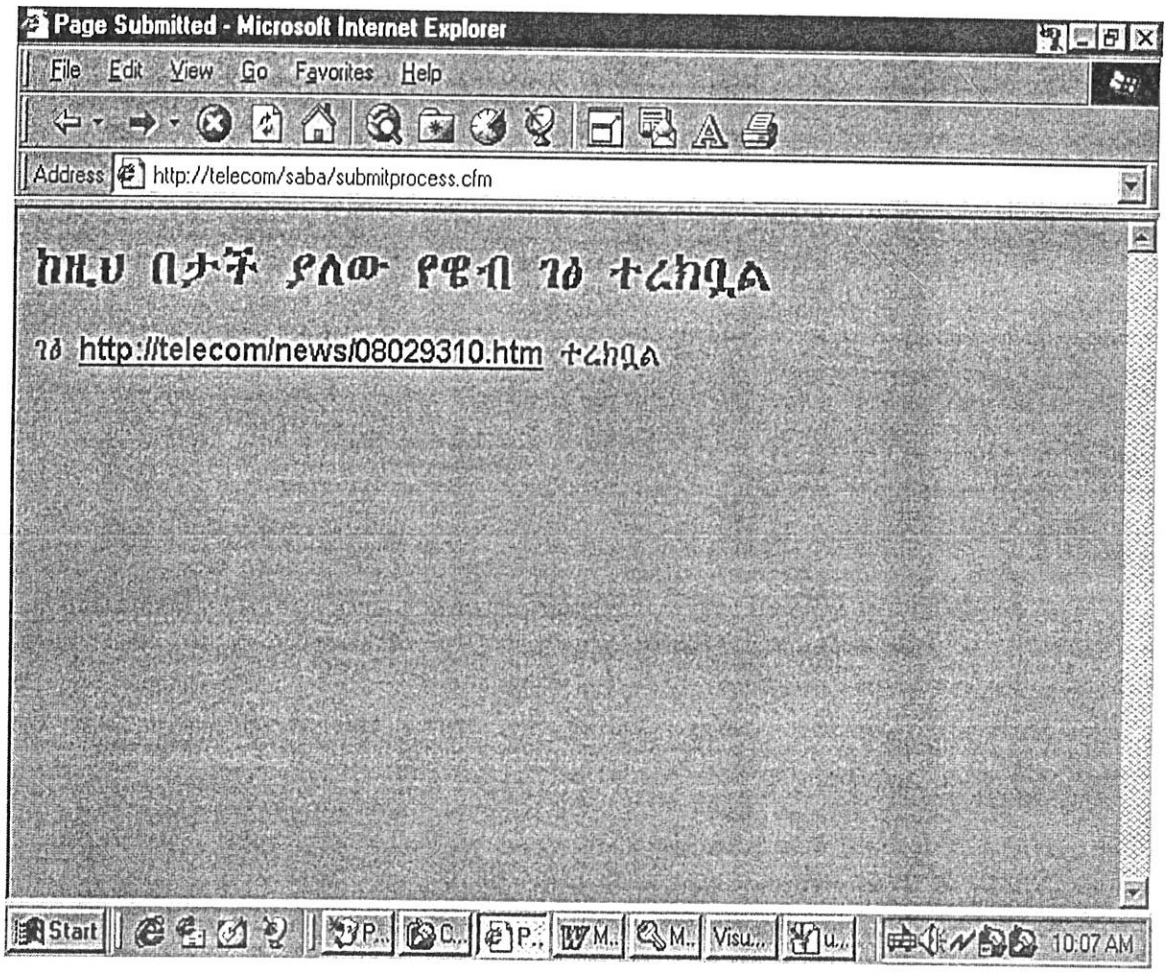


Figure 4.6 Confirmation of http://telecom/news/08029310.htm Submission

In the database the submitted data appears like the one below:

Did	Article	URL	Title
374	8/1993/ዋኢማ/ በባሌ ዞን ባለፉት ሶስት ወራት ሚሊየን ብር ካፒል ያላቸው 15 ባለሃብቶች ንት ፈንድ ወስደው ስራ መጀመራቸውን የዞኑ ምክር ቤት አስወቀ።	om/news/0802931	5 ነጥብ 8 ሚሊየን ያላቸው ባለሃብቶች ል

Table 4.21 Submitted Data for http://telecom/news/08029310.htm

Discussion

In designing the search and output interfaces and during searching some characters in Amharic needed to be handled differently. The basic problems encountered were:

1. In Amharic there are some letters that have the same value. For instance, ሠ and ሰ mean the same letter (sä) but there are common places where one or either of them are regularly used. The letter ሰ for instance is not usually used in the word ተሠማሩ (meaning 'they got engaged') instead ሠ is used. ሰ is often used in the word ሰበረ (he broke) and not ሠ. However, it does not mean it is wrong to write ተሰማሩ or ሠበረ. The searching in this experiment considers these two letters as different characters. Hence unless users reformulate their queries keeping in mind letters of the same meaning they may miss what is available. Look at the query ለ 'ነጋዴ ሴቶች ማኅበር' 3 ዶክመንቶች ተገኝተዋል። and its outputs in section 4.5 above. And look at the query below and the documents retrieved.

ለ 'ነጋዴ ሴቶች ማህበር ' 3 ዶክመንቶች ተገኝተዋል።

የጅማ ከተማ ነጋዴ ሴቶች ማኅበር አቋቋሙ

ጅማ ሕዳር 1/1993/ዋኢማ/ የጅማ ከተማ ነጋዴ ሴቶች ራሳቸውን በኢኮኖሚ ለማጠናከር የሚያስችል ማኅበር አቋቋሙ፤ 94 ሴቶች ኑሯቸውን አሻሻሉ። በጅማ ከተማ በንግድ እና በዕደ - ጥበብ ሥራ የተሠማሩ 150 ሴቶች ካለፈው

የጫት ኬላ ሠራተኞች ለለውዝ ምርት ቀረጥ ይጠይቃሉ ተባለ

ቆሮ የካቲት 2/1993/ዋኢማ/ በምዕራብ ሐረርጌ ዞን በሶስት የጫት ኬላዎች የተመደቡ ሠራተኞች የለውዝ ምርት በመያዝ ቀረጥ ይጠይቁናል ሲሉ ከባቢሌና ጉርሱም ወረዳዎች የለውዝ ምርት ገዝተው ለመሐል ሐገር የሚያቀርቡ አን

በሚዛን ተፈ ባዕድ ነገር የተቀላቀለበት ቅቤ በስፋት እየተሸጠ ነው ሕ በዚህ ሳምንት ብቻ አምስት ሻጮች ተይዘዋል

ሚዛን የካቲት 8/1993/ዋኢማ/ በቤንቺ ማጂ ዞን ሚዛን ተፈ ከተማ በሚገኙ ሱቆችና የገበያ ሥፍራዎች ከባዕድ ነገሮች ጋር የተቀየጠ ቅቤ በስፋት እየተሸጠ መሆኑን ሸማቾች አመለከቱ፤ የከተማው ፖሊስ ከባዕድ ነገሮች ጋር የተ

The only difference in the queries is the letters ጎ and ሀ in ሚጎበር and ሚሀበር words respectively. But the retrieved documents are different except for one. Remember the two words have the same meaning.

2. The same word could be spelled differently but still have the same meaning. For instance one can write ደላሎች or ደላላዎች to mean 'brokers' but they are treated as different words during searching. This problem might be solved by using roots or stems as index terms.

3. When the letter ሥ (| in English) exists in the query an error message is displayed. In fact all characters with ASCII values above 127 were a problem until the option to 'automatically convert special characters to their entity name' in ColdFusion studio was turned off.

4. In English the words smoking, Smoking or SMOKING have the same semantics. But this is not the case in Amharic. Since the Amharic font used in this research uses the English keyboard, even though there is no upper or lower case in Amharic, words like ስሞ & ሰሞ (Sm & sm) which do not have the same meaning are considered the same.

5. Some characters like " and # used to write text in Amharic are used by ColdFusion scripts and hence had to be excluded from text written on the Web pages.

CHAPTER FIVE

CONCLUSION AND RECOMMENDATION

5.1 CONCLUSION

Information retrieval has grown beyond its narrow use in the libraries. The introduction of the World Wide Web changed the perspective of information retrieval. The Web is the richest universal source of information that has ever existed. So being, it has allowed sharing of ideas and information in a very large scale. To retrieve relevant information from this huge source is not however an easy task. To satisfy his/her information needs one may navigate the space of Web links. However, since the hyperspace is vast such a navigation effort is usually inefficient.

The data on the Web is unstructured, volatile, with redundancy, and heterogeneous containing text and multimedia data in different languages. For naive users, the problem of finding relevant documents becomes even harder. To specify what is in the mind of the information seeker into a query is not an easy task. Even if the user is able to create the query successfully the answer may be thousands of Web pages and hence difficult to interpret.

There are search tools that are designed to more or less handle these problems, more widely for documents written in the English language and few for other languages like Arabic and Chinese.

Studies on how to retrieve information from Amharic documents on the Web have been necessary as there are documents written in Amharic on the Web but are not accessible by the search systems for other languages that are available on the Web today.

In this research a test collection of 313 Amharic Web pages, that covered news are collected and their contents are entered into a database. An interface that allows the submission of Amharic Web pages into the database where they can be indexed is designed and tested. The operation is found successful.

The Web documents entered into the database are then indexed. Among the parts of the Web pages the body of the page (the whole of it) of each of the documents is selected to be indexed. During the preparation of the text for indexing, punctuation marks are removed and the rest of the text is indexed taking a word as a set of characters between spaces.

Suffixes and prefixes are removed from index terms with the intention of improving performance. Hence, two inverted files, one for words and the other for stems are built.

The ratio of running words to distinct words in the collection that has 48058 running words is 3.44 for words and 4.83 for stems. This value is low when compared with the values obtained for English texts of smaller sizes in the research discussed in chapter two of this research. This indicates that in the Amharic documents the same words are not used repeatedly.

After sorting the index terms extracted in descending order of their total frequencies (frequencies in the entire collection), terms with a high frequency are cut off from being index terms. Terms with low frequency are not removed because in the Amharic documents individual terms are many and a significant number of them are with content.

The term weighting scheme used in this research is term frequency weight. Because the Web page documents that are used for the experiments in this research in particular are of similar size and since high frequency function words are already eliminated as stopwords the term frequency is selected as a reasonable weighting scheme.

Extended Boolean search with the operators OR and AND is the search facility that is incorporated with the system. Twenty-two queries are prepared to test the system against recall and precision. The queries are experimented by using the inverted indexes for words and for stems. A higher performance is obtained from stems than from words.

During searching, the need to handle alternative letters for the same letter in Amharic writing system is observed. The system must handle this in a way the English small and upper case letters are taken to mean the same.

In addition there is a character giving an error message when found in query terms and when found in documents during indexing and page submission. This specific character (ሥ) may not be a problem for other fonts. However because the Amharic font VG2 and many others use the English ASCII code, characters that are known as special characters in English but are used as standard characters in Amharic will possibly create problems.

There are also other characters that Visual Basic did not include in index terms even if they are part of the words in the documents indexed. Characters that have other use in ColdFusion Studio (like # & ") are not smoothly handled (had to be avoided from being entered as part of text on the interfaces designed).

5.2 RECOMMENDATIONS

In this study an attempt is made to access Amharic documents on the Web. Experiments have been conducted as to the possibilities and difficulties of using some of the retrieval techniques to handle Amharic documents on the Web. A lot is there to be done in the future by researchers in improving the scope of accessing available information for Amharic readers. The following recommendations are made for further research.

1. The use of thesaurus to increase the performance of the systems must be tested.
2. An indexing algorithm that makes use of root words should be developed to investigate in case better results may be attained.
3. Ways of handling all Amharic characters should be devised.
4. A standard Amharic code that could allow every Amharic software user to access documents in Amharic anywhere should be developed.
5. A system that handles alternative letters in Amharic has to be designed.
6. Research on how to crawl the Web and obtain Amharic documents should be made for the sake of finding up to date information and for the sake of avoiding the need for Web page owners to submit their Web pages.

7. Ranking schemes that consider links on pages, and weights based on font sizes and boldness of text on pages should be tested.
8. Ways of updating database of Web page data have to be devised.

REFERENCES

Abiyot Bayou. (2000). *Design and Development of Word Parser for Amharic Language*. (Masters Thesis). School of Information Studies for Africa. Addis Ababa University. Addis Ababa

AC. (2001). Ethiopia Sites on the Internet. African Cradle (AC). http://www.africacradle.com./ethiopia.html

[a]Gloor, P. (1997). Design-1.3 Automatic Indexing. *Elements of Hypermedia Design: Techniques for Navigation & Visualization in Cyberspace.* http://www.birkhauser.com/hypermedia/cyb7.html

Al-Kharashi, I. & Evenw, W. (1994). Comparing Words, Stems, and Roots as Index Terms in an Arabic Information Retrieval System. *Journal of the American Society of Information Science* 45(8):548-560

[a]Rijsbergen, V. (1996). Automatic Text Analysis. Information Retrieval. http://www.dcs.gla.ac.uk/keith/chapter.5/ch.5.html

Baeza-Yates, R. & Ribeiro-Neto, B. (1999). *Modern Information Retrieval.* England. Addison Wesley Longman Limited.

Beletu Reda.(1982). *A Graphemic Analysis of the Writing System of Amharic* Department of Linguistics. Addis Ababa University. Addis Ababa

Bender, M., et al. (1976). *Language in Ethiopia*. Great Britain. Oxford University Press

[b]Gloor, P. (1997). Short Introduction to Information Retrieval. *Elements of Hypermedia Design: Techniques for Navigation & Visualization in Cyberspace.* http://www.birkhauser.com/hypermedia/cyb3.html

[b]Rijsbergen, V. (1996). Searching Strategy. Information Retrieval. http://www.dcs.gla.ac.uk/keith/chapter.2/ch.2.html

Chandrashecar, B. (2001). Introduction to Information Retrieval. http://memebers.tripod.com/chandrashekarb/intro.html

Cohn, L. (2001). Boolean Searching on the Internet. http://library.albany.edu/Internet/boolean.html

Cotterell, F.(1964). Amharic Word Classes. *Journal of Ethiopian Studies*: *2(1)*

Dawkins, C. (1960). *The Fundamentals of Amharic.* Addis Ababa Sudan Interior Mission.

Dereje Teferi (1999) *Optical Recognition of Typewritten Amharic Text.* (Masters Thesis). School of Information Studies for Africa. Addis Ababa University. Addis Ababa

Ermias Abebe (1998). *Recognition of Formatted Amharic Text Using Optical Character Recognition.* (Masters Thesis). School of Information Studies for Africa. Addis Ababa University. Addis Ababa

Forta, B. et al (1998). The ColdFusion 4.0 Web Application Kit. Que. U.S.A.

Foskett, D. (1997). Thesaurus. *Readings in Information Retrieval.* U.S.A. Morgan Kaufman Publishers, Inc

Furzey, J. (1996). *Empowering Socio-Economic Development in Africa Utilizing Information Technology: A Critical Examination of the Social, Economic, Technical and Policy Issues, with Respect to the Expansion or Initiation of Information and Communication Infrastructure in Ethiopia.* African Information Society Initiative (AISI). Case Study

Harman, D. (2000). Automatic Indexing. National Institute of Standards and Technology. http://www.itl.nist.gov/iaui/894-02/works/pubs/ir4873.html

Hmeidi, I., Kanaan, G. & Evans, M. (1997). Design and Implementation of Automatic Indexing for Information Retrieval with Arabic Documents. *Journal of The American Society of Information Science.* 48(10):867-881

Hypertext. *Microsoft® Encarta® Encyclopedia 99.* © 1993-1998 Microsoft Corporation.

IU.(2000). Main Languages Discussed in Lectures. Indian University (IU) http://www.indiana.edu/~gasser/L103/languages.html

Jones, K & Willet P. (1997). Overall Introduction. *Readings in Information Retrieval.* U.S.A. Morgan Kaufman Publishers, Inc

Keller, E. Ethiopia. *Microsoft® Encarta® Encyclopedia 99.* © 1993-1998 Microsoft Corporation.

Lager, M. (1996). Spinning a Web Search. http://www.library.edu./untangle/lager.html

Laudon, K. & Laudon L. (1998). *Management Information Systems: New Approach to Organizations and Technology*. New Jersey. Prentice Hall, Inc.

Lishan Adam (1999). Information and Communication Technologies in Ethiopia: Past, Present and Future Potential for Social and Economic Development in Ethiopian Information *Technology professional Association Workshop*, 2 March 1999

Martin, P. (1999). Information Retrieval. http://agents.www.media.mit.edu/groups/agents/publiications/newt-

thesis/subsection2-7_4_1.html

Million Meshesha (2000). *A Generalized Approach to Optical Character Recognition of Amharic Texts.* (Masters Thesis). School of Information Studies for Africa. Addis Ababa University. Addis Ababa

Nega Alemayehu (1999). *Stemming Amharic Text for Information Retrieval.* (Ph.D. Thesis) Department of Information Studies. University of Sheffield. England

Nigussie Tadesse (2000). *Handwritten Amharic Text Recognition Applied to the Processing of Bank Checks.* (Masters Thesis). School of Information Studies for Africa. Addis Ababa University. Addis Ababa

Page, L. & Brin, S (2000). The Anatomy of a Large-Scale Hypertextual Web Search Engine. Stanford University. http:/www7.scu.edu.au/programme/fullpapers/1921/com1921.htn

Poulter, A. (1997) The Design of World Wide Web Search Engine: A Critical Review. *Program. 31(2)*: 131-145

Punctuation. *Microsoft® Encarta® Encyclopedia 99.* © 1993-1998. Microsoft Corporation.

Salton, G. & McGrill, M. (1983). *Introduction to Modern Information retrieval.* U.S.A. McGraw-Hill, Inc.

Schwartz, C. (1998). Web Search Engines. *Journal of the American Society of Information Science. 49(11):* 973-982

Search Engine. *Microsoft® Encarta® Encyclopedia 99.* © 1993-1998. Microsoft Corporation.

SEW.(2001). How Search Engines Work. Search Engine Watch (SEW)

Internet.com Corp. http://searchenginewatch.com/

Spider. *Microsoft® Encarta® Encyclopedia 99.* © 1993-1998. Microsoft Corporation

Strzalkowski, T. (1997). Robust Text Processing in Automated Information Retrieval. *Readings in Information Retrieval.* U.S.A. Morgan Kaufman Publishers, Inc

Takkinen, J. (1996). Information Retrieval and Information Filtering (IRIF), *Spring 1996: Introduction to Course.* Http://www.ida.liu.se/~jahta/IRIF/IRIF_introduktion.html

Visual Geez for Windows 3.x and Windows 9x Applications. 1995-1997. Custor Computing

Wan, T., Evans, M., Wan, Y. & Pao, Y. (1997). Experiments with Automatic Indexing and a Relational Thesaurus in a Chinese Information Retrieval System. *Journal of The American Society of Information Science.* 48(12):1086-1096

Web Site. *Microsoft® Encarta® Encyclopedia 99.* © 1993-1998 Microsoft Corporation.

Worku Alemu (1997). *The Application of OCR Techniques to the Amharic Script.* (Masters Thesis). School of Information Studies for Africa. Addis Ababa University. Addis Ababa

APPENDICES

APPENDIX I

FULL AMHARIC CHARACTER SET

Order							Labialized				
1st	2nd	3rd	4th	5th	6th	7th					
ሀ	ሁ	ሂ	ሃ	ሄ	ህ	ሆ					
ለ	ሉ	ሊ	ላ	ሌ	ል	ሎ			ሏ		
ሐ	ሑ	ሒ	ሓ	ሔ	ሕ	ሖ			ሗ		
መ	ሙ	ሚ	ማ	ሜ	ም	ሞ					
ሠ	ሡ	ሢ	ሣ	ሤ	ሥ	ሦ					
ረ	ሩ	ሪ	ራ	ሬ	ር	ሮ			ሯ		
ሰ	ሱ	ሲ	ሳ	ሴ	ስ	ሶ			ሷ		
ሸ	ሹ	ሺ	ሻ	ሼ	ሽ	ሾ			ሿ		
ቀ	ቁ	ቂ	ቃ	ቄ	ቅ	ቆ	ቈ	ቍ	ቋ	ቊ	ቌ
በ	ቡ	ቢ	ባ	ቤ	ብ	ቦ			ቧ		
ተ	ቱ	ቲ	ታ	ቴ	ት	ቶ			ቷ		
ቸ	ቹ	ቺ	ቻ	ቼ	ች	ቾ			ቿ		
ኀ	ኁ	ኂ	ኃ	ኄ	ኅ	ኆ	ኈ	ኍ	ኋ	ኊ	ኌ
ነ	ኑ	ኒ	ና	ኔ	ን	ኖ			ኗ		
ኘ	ኙ	ኚ	ኛ	ኜ	ኝ	ኞ			ኟ		
አ	ኡ	ኢ	ኣ	ኤ	እ	ኦ					
ከ	ኩ	ኪ	ካ	ኬ	ክ	ኮ	ኰ	ኵ	ኳ	ኲ	ኴ
ኸ	ኹ	ኺ	ኻ	ኼ	ኽ	ኾ					
ወ	ዉ	ዊ	ዋ	ዌ	ው	ዎ					
ዐ	ዑ	ዒ	ዓ	ዔ	ዕ	ዖ					
ዘ	ዙ	ዚ	ዛ	ዜ	ዝ	ዞ			ዟ		
ዠ	ዡ	ዢ	ዣ	ዤ	ዥ	ዦ			ዧ		
የ	ዩ	ዪ	ያ	ዬ	ይ	ዮ					
ደ	ዱ	ዲ	ዳ	ዴ	ድ	ዶ			ዷ		
ጀ	ጁ	ጂ	ጃ	ጄ	ጅ	ጆ					
ገ	ጉ	ጊ	ጋ	ጌ	ግ	ጎ	ጐ	ጕ	ጓ	ጒ	ጔ
ጠ	ጡ	ጢ	ጣ	ጤ	ጥ	ጦ			ጧ		
ጨ	ጩ	ጪ	ጫ	ጬ	ጭ	ጮ			ጯ		
ጸ	ጹ	ጺ	ጻ	ጼ	ጽ	ጾ					
ፀ	ፁ	ፂ	ፃ	ፄ	ፅ	ፆ			ፏ		
ፈ	ፉ	ፊ	ፋ	ፌ	ፍ	ፎ			ፗ		
ፐ	ፑ	ፒ	ፓ	ፔ	ፕ	ፖ					

ቨ	ቩ	ቪ	ቫ	ቬ	ቭ	ቮ

Numerals

1	፩	6	፮	20	፳	70	፸
2	፪	7	፯	30	፴	80	፹
3	፫	8	፰	40	፵	90	፺
4	፬	9	፱	50	፶	100	፻
5	፭	10	፲	60	፷	1000	፼

The Punctuation Marks of Amharic

	Punctuation marks	Names	Allographs
1	⟨ : ⟩	/nEt'Ib or hulEt nEt'Ib/	/ ፡ /mainly in printing
2	⟨ ፥ ⟩	/nEt'Ela sErEz/	/፦/ and / ፥ /
3	⟨ ፧ ⟩	/dirrb sErEz/	
4	⟨ :: ⟩	/mulu or arattnEt'Ib/	
5	⟨ ! ⟩	/tImIrtE ankIro/	
6	⟨ ¡ ⟩	/tImIrtE sIlak'k'/	
7	⟨ ? ⟩	/tImIrtE t'Iyyak'e/	
8	⟨ · ⟩	/yIzEt or and nEt'Ib/	/-/; /·/; /።/ &/.⁄
9	⟨ — ⟩	/C'IrEt or sErEz/	
10	⟨ - ⟩	/nIus C'IrEt/	
11	⟨ ... ⟩	/nEt'Ebt't'ab/	
12	⟨ () ⟩	/k'InnIf/	/(/ and /( )/
13	⟨ " " ⟩	/tImIrtE t'Ik'k's/	/« »/and/“ ”/
14	⟨ " ⟩[25]	/IdEllaINNaw/	/»/ and /“/ /”/
15	⟨ ' ⟩[25]	/C'IrEt/ or apostrophy	
16	⟨ ፧- ⟩[25]	/mEzEgaIrawi nEt'Ib/	/ ፧-/ and / ፧/

[25] with the exception of those that have numbers at the top, the symbols are all taken from MErse HazEn WEldE K'erk'os's, "yamarINNa sEwasIw" p. 207.

APPENDIX II

Queries

No.	Query	Relevant Doc
1	ጎጂ ልማዶች	161,174,268,225
2	ማዳበሪያና ምርጥ ዘር መጠቀም	66
3	በድርቅ የተጎዱ ክልሎች	95
4	በምግብ ሰብል ራስን ለመቻል የሚደረግ ጥረት	56
5	በሻዕቢያ ወረራ የተፈናቀሉ አርሶ አደሮችን ማቋቋም	33
6	በስደድ እሳት የደኖች መጥፋት	3,97,262
7	በስደት ያሉ ኢትዮጵያውያን	57,155
8	በተባበሩት መንግስታት የሰላም ማስከበር ወታደሮች	80
9	ህገ ወጥ ወርቅ አምራቾች	18
10	ከኤርትራ የተመለሱ ኢትዮጵያውያን	5,10,33,69,186,256, 259,339
11	ጠለፋና አስገድዶ መድፈር	8,64,107,179
12	ኑጋዴ ሴቶች ማህበር	5,246
13	ሻዕቢያ ያሰራችው ኢትዮጵያውያን	10,186,241,259
14	ሻዕቢያ በውትድርና አገልግሎት ጦር ሜዳ ያዘመታቸው መምህራን	6
15	ሴቶችና ልማት	125,133,145,166,17 0,246,247,279
16	ኢትዮጵያውያን ስደተኞች በሱዳን	57,155
17	አዲስ የተገነቡ የጤና ተቋማት በኢትዮጵያ	9,24
18	የፖሊዮ በሽታ ክትባት	252
19	የማጅራት ገትር ክትባት	131,144,165
20	የደላሎች ማህበር	72
21	የደም ተትማጥ ለመከላከል የተደረገ ጥረት	73
22	የመንግስት ገንዘብ መገበሪዎች	14,51,110,153
23	የመንግስት ገንዘብ ምገበራ	14,51,110,153
24	የመስኖ ልማት ፕሮጀክቶች	26,87.227,284

25	የንፁህ መጠጥ ውሃ ፕሮጀክቶች	1,24,199,212,276,277,329
26	የቅድመ ጋብቻ የኤድስ ምርመራ	41,104,108,142,188,218,245,264
27	የጠጠር መንገድ ግንባታ	89
28	የሴቶችና ህጻናት ልማት ድርጅት	126
29	የስልክ አገልግሎት	12,50,185,325
30	የትራፊክ አደጋ በአዲስ አበባ	39
31	የተያዙ የኮንትሮባንድ እቃዎች	65
32	የተፈጥሮ ደንና ዱር እንስሳት ጥበቃ	4
33	የትምህርት ቤቶች የቤተመከራ አገልግሎት	2
34	የትምህርት በራዲዮ ማሰራጫ	2,285
35	የውሃ ጉድጓዶች ቁፋሮ	1,24,199,212,276,277,329
36	የወረዳና የቀበሌ አባላት ምርጫ	49,94,181,209,225
37	የኢትዮጵያ ማህበራዊ ተህድሶና ልማት ፈንድ	122
38	የኢትዮጵያ መከላከያ ሰራዊት ጽጥታ ማስከበር	69
39	የኢትዮጵያና የስዊድን መንግስት የልማት ትብብር ስምምነት	77
40	የኤድስን ስርጭት መግታት	7,28,96,118,124,173,183,188,222
41	የአካባቢ ልማት ፕሮግራም	74

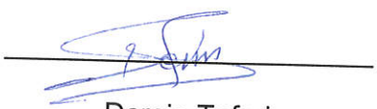
DECLARATION

I, the undersigned, declare that this thesis is my original work and has not been presented for a degree in any other university, and that all sources of material used for the thesis have been duly acknowledged.

Saba Amsalu Teserra

July, 2001

The thesis has been submitted for examination with our approval as university advisors.



Dereje Teferi

Million Meshesha

July, 2001