Multilanguage Library Search Systems

by

Abdellatif Habes

A thesis submitted to the

Faculty of Science,

University of Glasgow

For the degree of Master of Science

Department of Computing Science

University of Glasgow

July, 1990

This work is dedicated
to my father, my wife, my children
and to the memory of my mother

Acknowledgments

I am greatly indebted to my supervisor, Professor Dennis Gilles, who was a continuous source of inspiration and advice. He provided me, over the years, with the idea and supply of the relevant material to this work. His keen interest and comments were continually renewed until the end of this project.

I also thank my friends Khaled and Djamel for the advice they provided and the interest they showed for the present work. I am grateful to Jak Deschner who read this thesis and made many useful comments. Thanks also to Abdenour, Gebre and Saeed for the memorable moments we spent together in this department.

Last but not the least, I would like to thank my wife Fatiha, my daughter Ibtissam and my son Sophiane for their patience for putting up with my being away from them.

My gratitude and my respect must go to the Algerian Government for the financial support and the opportunity it gave me to study for this postgraduate degree.

Declaration

This thesis is entirely on my original work and no part is done in collaboration. Where the work of others is used, explicit reference is made in the text. No part of this thesis has been or is being submitted for a degree at any other university.

Abstract

This thesis presents an application of information retrieval in a multilingual environment, by computerising the documents stored in a library. These documents are either in English/French or in Arabic. Some approaches to processing them are discussed; the possibilities to mix the documents are studied and a solution is developed. English and French documents are grouped together and a method for their retrieval is developed. Arabic documents are not considered in their original scripts but are romanised. The proposed method for searching the Arabic documents is based on the reduction of the words into root-pattern pairs which is handled by a morphological study of the Arabic words. The Arabic documents are totally separated from English/French documents.

Finally, a user interface and the implementation of the system are developed to realise the aim of this project. The user interface gives possibilities for the user of the library to search documents in either English/French or Arabic. File structures are presented in the context of information retrieval and an evaluation of the system in terms of efficiency is studied.

TABLE OF CONTENTS

Chapter 1: INTRODUCTION 1

1·1 GENERALITIES 1

1·2 EXISTING TECHNIQUES 2

1·2·1 Romanisation 3

1·2·2 Original scripts 3

1·3 FORMULATION OF THE PROBLEM 5

1·4 PROPOSED SOLUTION 6

1·5 OUTLINE OF THE THESIS 7

Chapter 2: SURVEY OF THE RELEVANT WORK 9

2·1 INTRODUCTION 9

2·2 LIBRARY SYSTEM AND AUTOMATION 9

2·2·1 Acquisitions 11

2·2·2 Cataloging 12

2·2·3 Serials 13

2·2·4 Circulation 13

2·3 INFORMATION RETRIEVAL 14

2·4 SEARCH STRATEGIES 15

2·4·1 Method of position keywords 17

2·4·2 Method of postings 17

2·4·3 KWIC index 18

2·4·4 KWOC index 20

2·4·5 Search-key 20

2·4·6 Boolean search 21

2·5 ARABIC 22

2·5·1 Arabic characters 22

2·5·2 Internal representation of Arabic characters 23

2·5·3 Display and direction layout 23

2·5·4 Output of an Arabic text 24

2·5·5 Writing a mixed text 25

2·5·6 Deleting and inserting a character 25

2·5·6·1 Deleting a character 25

2·5·6·2 Inserting a character 26

2·6 EXISTING MULTILANGUAGE LIBRARY SYSTEMS 28

2·7 SYNTHESIS 28

Chapter 3: MULTILANGUAGE SEARCHING 30

3·1 INTRODUCTION 30

3·2 SEARCHING ENGLISH/FRENCH DOCUMENTS 31
3·2·1 Unit record file (URF) 33
3·2·1·1 First approach 34
3·2·1·2 Second approach 35
3·2·2 Definitions 37
3·2·2·1 Dictionary stop list (DSL) 37
3·2·2·2 Keyword 37
3·2·2·3 Index file (IXF) 38
3·2·2·4 Inverted file (IVF) 38
3·2·2·5 Position keyword (PSK) 38
3·2·3 Dictionaries 39
3·2·3·1 Creation of dictionaries 41
3·2·3·2 Updating the dictionaries 42
3·2·4 Search approach 43
3·2·4·1 Text input 45
3·2·4·2 Elimination of noise words 45
3·2·4·3 IXF, IVF files 46
3·2·4·4 Intersection of sets 46
3·2·4·5 Output documents 47
3·2·4·6 End of search 47
3·3 SEARCHING ARABIC DOCUMENTS 48
3·3·1 Definitions 49
3·3·1·1 Arabic text 50
3·3·1·2 Root and pattern 50
3·3·1·3 Arabic stop list dictionary (ASL) 51
3·3·1·4 Keyword 52
3·2·2 Reducible words 52
3·2·2·1 Method of reduction 52
3·2·2·2 Cases of triliteral roots 53
3·2·2·3 Organization of table ROOT 55
3·2·2·4 Index files 57
3·3·3 Irreducible words 60
3·3·4 Search documents 61
3·3·4·1 Text input 61
3·3·4·2 Text decomposition 62
3·3·4·3 Separation of reducible from irreducible words 62
3·3·4·4 Sets of document numbers 62
3·3·4·5 Retrieval documents 63
3·3·4·6 Output documents 65
3·3·4·7 End of search 65
3·4 FORMS OF THE ARABIC TEXT 65
3·4·1 Pure Arabic characters 65

3·4·2 Arabic romanization 66
3·4·3 Correspondence Arabic scripts/Roman characters 66
3·5 MODEL OF MULTILANGUAGE SEARCHING 67
3·5·1 Mixed languages (English/French and Arabic) 67
3·5·2 Separated languages (English/French and Arabic) 68
3·6 CONCLUSION 68

Chapter 4: MANIPULATION OF THE QUERY SYSTEM 69
4·1 INTRODUCTION 69
4·2 DESCRIPTION OF THE QUERY SYSTEM 70
4·2·1 Level 0 71
4·2·2 Level 1 71
4·2·3 Level 2 72
4·2·4 Level 3 73
4·3 SCREEN LAYOUT 73
4·4 MANIPULATION OF THE QUERY SYSTEM 74
4·4·1 English/French documents 76
4·4·1 Arabic documents 79
4·5 CONCLUSION 82

Chapter 5: SYSTEM DESIGN 83
5·1 INTRODUCTION 83
5·2 DATA STRUCTURES 84
5·2·1 English/French documents 84
5·2·1·1 URF file 84
5·2·1·2 DSL file 84
5·2·1·3 IXF and IVF files 85
5·2·2 Arabic documents 87
5·2·2·1 ROOT, PATTERN and TWO-ROOT tables 87
5·2·2·2 ASL dictionary 88
5·2·2·3 IXIR and IVIR files 88
5·2·2·4 IXR and IXP files 88
5·3 UPDATING SYSTEM(USY) 91
5·4 CONCLUSION 92

Chapter 6: SYSTEM EVALUATION 93
6·1 INTRODUCTION 93
6·2 EFFICIENCY ANALYSIS 94
6·2·1 Case of retrieval English/French documents 94
6·2·2 Case of retrieval Arabic documents 96
6·2·2·1 First organization 97
6·2·2·2 Second organization 98
6·2·2·3 Third organization 99

6·2·2·4 Evaluation 99
6·2·2·5 Discussion and examples 101
6·3 CONCLUSION 105

Chapter 7: CONCLUSIONS 107
7·1 RESULTS 107
7·2 IMPLEMENTATION 109
7·3 FURTHER WORK 109

Appendix A: SYSTEM IMPLEMENTATION 111

Appendix B: COMPLEMENT ON ARABIC 120

References 123

Chapter 1

INTRODUCTION

1·1 GENERALITIES

Every human being is asking for knowledge at any time. Domains of this knowledge are very wide; one has an interest in music or in phenomena of the nature; another is fascinated by oriental culture or by computers etc. Sources of knowledge are plentiful, e.g. discussion, correspondence, travel, contacts or materials such as books, journals, maps, tape recordings to name a few. These materials are either in the native language of the interested person, or are in foreign languages. Suppose the desired texts are in a library. One of the aims of every library is to facilitate rapid and efficient bibliographic searches, often by computerising their catalogues and indexes. It would be highly desirable if these catalogues and indexes could be searched irrespective of the natural language used. Hence, computerising the documents of the library despite the number of languages used being countless is an important and difficult task of any library.

There are only ten major scripts in use, which account for 99 percent of the world's present book production as listed by [WELLI 78], the six most productive being Roman (English, French, German, Spanish, etc), Cyrillic (Russian, Bulgarian, etc), Japanese, Chinese, Devanagari (Sanskrit, Hindi, etc), and Arabic (Arabic, Farsi, Urdu, etc).

How to organise the automation of the library which contains materials in many languages is the main question I wish to address. I shall not deal with the acquisition and circulation control of documents. My interest is the searching of documents and this study is focused around this area in a multilingual environment. A rich discussion on cataloging and classification of non-Western material is presented in [AMANM 80].

1·2 EXISTING TECHNIQUES

Languages may be divided into two classes, those using Roman scripts and those using non-Roman scripts. Therefore, for searching or cataloging multi-lingual documents, some problems of language arise to the users and the library staff. Documents have to be well processed in order to be accessed by a user. The solution is either to romanise the non-Roman scripts or to provide systems which manipulate non-Roman scripts.

1 · 2 · 1 Romanisation

Romanisation is a method which converts a word that is written in non-Roman script into a word that sounds like the original but is written in the Roman script. Transliteration and phonetic transcription are the only examples of romanisation well described by [MILLE 82]. This solution is applied by a lot of libraries throughout western countries where most of the non-Roman material is romanised (titles, authors). This romanised material is then introduced to the computer to be processed in a manner similar to Roman material. All the documents are then mixed together or separated depending on the organisation adopted by the library manager; thereafter, usual search strategies are used to retrieve documents.

Disadvantages appear when such romanisation is used. The library users must romanise by hand non-Roman material when there is not a guide for romanisation, whereas most of them would much prefer to use catalogues in the original script. The failure of the reverse process, that of changing scripts into non-Roman scripts, is obvious; thus the system is not universal. For example English, or French names and titles are not easily Cyrillicised, or Arabised.

However, romanisation is a good solution for libraries which have a small number of non-Roman materials or cannot provide material and software which manipulate non-Roman scripts.

1 · 2 · 2 Original scripts

An alternative solution is to provide access in the original script for non-Roman materials. Using a keyboard which handles script characters and a CRT to display those characters is the first part of the solution. Keyboards can be multiscript allowing many scripts to be manipulated at the same time, or bilingual with reference to Roman scripts with the keyboard having Roman script characters and a class of non-Roman script characters such as Roman-Arabic, or Roman-Cyrillic. Input is in the original script as displayed on the stem being processed. Names are input in their original script. In a paper, Pierre A MacKay [MACKA 86] thinks that computer typesetting provides a solution for Arabic and similar scripts such as he proposed in [MACKA 83] with the utilisation of $\TeX$.

Cataloging and retrieving non-Roman materials is the second part of the solution. Software has to be developed that is able to handle the storage and manipulation of non-Roman materials. The search strategies for retrieving Roman materials can be adapted, when ever possible. Thereafter, processing documents becomes simpler for library staff and the results are more accurate.

Some hardware products already exist on the market. Macintosh and IBM developed word processing of several non-Roman scripts. Vendors such as APTEC and TECHNOCRAT developed packages (namely 'Arabdbase' and 'Linguafile') for database management systems in Arabic and English materials. The languages are not mixed.

A multilingual environment within Roman scripts is not deeply studied in this work. Indeed, there are some difficulties in mixing these languages; for example the same keyboard is used for English, French, and German. Titles of documents in these three languages all contain keywords and noise words; the list of noise words can be mixed such as *the, an, of, le, les, dans, die, der.* Differences in pronunciation or in grammar are not a handicap for the creation of a database containing documents in these languages be it mixed or separated.

1·3 FORMULATION OF THE PROBLEM

As presented earlier in section 1·1, this study is focused on searching documents in a multilingual environment. There are difficulties in developing solutions for general environments, so that the study is restricted to the particular behaviour of the languages English, French and Arabic although it is hoped that the ideas can be extended to the other languages too. The reasons for choosing the above languages in our study are:

- There are differences between Arabic and Western languages in character shapes, direction of writing and language structure; thus Software systems developed for Western languages are not readily applicable for use with Arabic.

- There is a need for tools and operations for the manipulation of materials in Arabic scripts, in particular for the development of a method for searching Arabic materials involving romanisation, Arabic morphological analysis and the ideas of information retrieval.

- The development of a method for searching English/French materials provides a study of processing two different languages with the same script.

Finally, this would be benefit for the Algerian University Libraries where most of the materials are in Arabic, French, and English, in that order.

1·4 PROPOSED SOLUTION

The main interest of the study which concerns the materials in these three languages can be divided into three parts. The Romanisation of Arabic characters and their correspondence with Roman characters, a morphological study of the Arabic words by reducing them into root pattern pairs where the pattern is the measure of the word, and the organisation of files to handle Arabic materials forms the first part. This representation of the Arabic word facilitates the creation of inverted files necessary to search the documents. The conception of this part is developed by the application of some Arabic rules. Therefore, the developed programs can handle either Arabic scripts when appropriate software exists or romanised Arabic scripts which are tested in this project.

Secondly, a study of English/French documents is developed with the aim of jointly processing documents written in these two Roman scripts. It is found that mixing the two languages is the best solution. The techniques used are inverted files method and Boolean search.

Finally, we attempt to combine systems in all three languages. We find that, whatever the representation of Arabic materials, they are best separated from the English/French materials. A multilanguage search system is constructed with an acceptable user interface, the results of the search documents are satisfactory although the time due to the reduction of words into root-pattern pairs can be appreciable. Although romanised materials can be gathered with Roman materials, the separation gives more accuracy because of the special study of Arabic words.

1·5 OUTLINE OF THE THESIS

The thesis is organised in seven chapters which describe the aim of this work and its context, the difficulties, the proposed solutions and their evaluation. Appendix A is reserved for the implementation of the system and Appendix B illustrates the Arabic characters, their associated binary codes and their correspondence with the ASCII characters.

Chapter 1: Introduction - presents some basic solutions relative to a multilingual environment in a library and the problems due to the non-Roman materials. Different solutions for a multiscript system such as romanisation, keyboard and display for non-Roman scripts are discussed. Finally, a presentation of the studied case is outlined.

Chapter 2: Survey of relevant work - outlines theories and studies on automation, library systems, acquisition cataloging, serials and circulation control. Information retrieval and search strategies are introduced. Useful details on Arabic language in the context of the project are given. Some existing multilanguage systems are presented.

Chapter 3: Multilanguage searching - is the major chapter dealing with searching Arabic, English and French documents. It provides a solution to manipulate these documents. The studies of English/French material and Arabic material are separated. A method to handle English/French materials is outlined. An Arabic morphological analysis is presented and a technique for the reduction of Arabic words into root-pattern pairs is discussed. This reduction of the words facilitates the creation of files to represent and search Arabic materials.

Chapter 4: Manipulation of the query system - proposes a user interface for the search documents. Layout of screen is outlined. The different levels of the search are presented; a guide to perform a search is discussed with examples throughout.

Chapter 5: System design - introduces the files and data structures of the English/French and Arabic materials. All necessary files and dictionaries are detailed. The updating system is also introduced.

Chapter 6: System evaluation - analyses the efficiency of the proposed solution. Formulae which compute the number of operations from the checking of existing documents to the relevance are described for each language. Number of operations due to the reduction of the Arabic words are also discussed. Examples are provided and results are compared.

Chapter 7: Conclusions - has some concluding remarks and some comparisons of results of the implemented systems. Optimisation of the system and further work is also discussed.

Appendix A: System implementation - contains the basic algorithms which implement the proposed solution. The methodology of such algorithms is illustrated by some examples when it is necessary.

Appendix B: Arabic representation - provides representation of the Arabic characters, the associated binary code and the correspondence with Roman characters.

Chapter 2

SURVEY OF RELEVANT WORK

2 · 1 INTRODUCTION

This chapter outlines the origin, description, different problems encountered, and results of the library systems and automation. Information retrieval is also described, especially the elements required in this project such as file structures and searching strategies. General notions on Arabic structure are given. Finally, multilanguage searching is discussed and illustrated with some existing systems.

2 · 2 LIBRARY SYSTEM AND AUTOMATION

Modern libraries try to simplify the tasks of librarians and users through automation. Hence, knowledge is stored in a library with a computer system allowing its retrieval. Much work has been done in the area of library automation: as S. R. Salmon [SALMO 75] defines «*Library automation is the use of automatic and semiautomatic data processing machines to perform such traditional libraries activities as acquisitions, cataloging, and circulation*». This subject is quite different from the field of information retrieval.

Library automation projects started in the early 1960s, the most notable being the National Library of Medecine (the MEDLARS project), the University of California at San Diego on serials (periodicals) control, the Southern Illinois University in Carbonade for circulation systems, computerised book catalogs at the University of Toronto. Most of these computerised library automation projects of the mid-1960s, including those based on MARC, were off-line, batch-processing systems. An example of processing documents at Durham University in an off-line environment is studied in detail by R.N. Oddy [ODDYR 71.]

The improvement of the interface between user and system, between man and machine permitted such automated systems to be online. The user can maintain a dialogue with the computer using keyboard, and cathode-ray tube (CRT) displays. In response to a prompt, the computer program requests further information, the user supplies it, and the computer proceeds with the conversation depending on the answers to its questions. As a result of such improvements, these off-line batch processing systems were superseded by online systems.

Automation is a crucial problem to librarians; however, it needs money, material, and time. Automation can only be accepted if it changes the environment of the library by helping librarians and users; by, for example, reduction of manual tasks, providing more information and fast answers to the user. As R.T Kimber [KIMBE 74] said « *Research* *effort in library automation should be directed towards improving existing services, and developing new services, rather than just mechanizing a status quo* ». Computation and analysis is crucial to controlling the automated system, facilitating the evaluation of the automated system. The relationship between user and library may be improved when knowledge of, for example,which kinds of books are borrowed from the library, and which kinds of books that bought for the library.

P.A Thomas [THOMA 73] discusses some routines necessary in library systems by showing the different tasks that the librarians have to handle: as he said « *A library operates by acquiring books, processing them, and making them available to its users. The routine which enables it to do this are based on bibliographic descriptions of individual books* ». The terms *acquisition, cataloging, circulation* and *serial systems* are often used in library systems. Acquisition concerns the obtaining of new documents by the library, depending on some criteria. Cataloging is the production of library catalogues detailing the current stock. Serial publications (periodicals) are also taken into account. Circulation makes facilities available to the user to access catalogues and documents. Some other functions such as accounting, ordering and accessioning systems are introduced but only the library staff has to use them. These considerations are developed in the following sections.

2 · 2 · 1 Acquisitions

Acquisition is an important part of an automated library. Tasks and routines have to be well structured and easily updated to facilitate the management of the process of buying books. Ordering of documents takes into account the statistics detailing the utilisation of documents in the library and the needs of the users. Therefore, lists are prepared depending on the author, country, subject, language or other criteria.

Acquisition systems depend on the type of material and the orders handled, the type of payment and the type of language. [SALMO 75] gives details on this matter.

2·2·2 Cataloging

Cataloging is a fundamental task for any library automation system. A small flaw in conception of the catalog can have bad consequences for the future of the library. A number of theories and systems have been developed for cataloging. These include Library of Congress (LC) and British Library (BL) which, using the Anglo-American Cataloging Rules (AACR), have normalised some routines that are used all over the world. However, each library can develop its own routines for cataloging in order to ease the search for information required most by users. Catalogs created by author, title, subject or by code are the most frequently used indexes in libraries. Classification is also handled by the standard codification of the documents. Many libraries use the Dewey codification. J. Horner [HORNE 70] contributed many details on this subject, and J. Friedman & A Jeffreys [FRIED 67] give examples of cataloging and classification in British University Libraries. M. A. Aman [AMANM 80] presents the cataloging and classification in Asian, African and Arabic countries where transliteration is sometimes necessary.

2·2·3 Serials

Serials publications are periodical documents where their acquisition and cataloging could be slightly different to that for books. They have their own code, ISSN (International Standard Serial Number), or Coden, which is defined as a five character code designating the title of a specific publication. Details on this code are given in [AMERI 60]. The full title is often used to manipulate the serials which do not have a code. An in depth study held by A.D. Osborn [OSBOR 73] gives many details on serials, illustrated by several examples of University Libraries. Serials are in generally printed out in readable form and placed at various sites for consultation.

2·2·4 Circulation

This is one of the easiest phase of library automation. It describes the relation between the borrowers and the documents. The library staff are the only ones who can control this circulation; they are able to manipulate files related to the borrower, files related to the loans and other information in the catalogue: for example, searching for a document by its identification number. The information stored on borrowers is name, address, telephone, identification number, category borrower (postgraduate, undergraduate), books borrowed and their status. The main role of circulation is to satisfy every user by giving him all the information about the status of documents in the library, especially those on loan with date due for return and its consequences (fines to pay, blacklist), time of loan, physical storage, missing documents, new documents, how to reserve a document which is on loan, how to renew loans of documents, etc. These operations are easily computerised and used every time a transaction is carried out between the user and the library. Several approaches are handled in [KIMBE 74] and [SALMO 75]. Actually, most of the online systems have sophisticated circulation controls which simplify enormously a wide part of the management of the library. For example, Westlake & Clarke [WESTL 87] described these functions of the GEAC system which is used at Glasgow University Library.

2 · 3 INFORMATION RETRIEVAL

The definition of information retrieval is complex. Performing methods to retrieve relevant documents from a huge set of documents is one of the aims of information retrieval. In the case of a transaction between a user who formulates a quěry or a question and a set of documents, results are given by the relevance which is paramount: as C.J. Rijsbergen [RIJSB 79] said « *the notion of relevance is the centre of the information retrieval*». Therefore one of the main tasks of information retrieval is to find the best strategy to calculate this relevance. Vickery [VICKE 70], Lancaster [LANCA 79], Heaps [HEAPS 78], Rijsbergen [RIJSB 79], Salton [SALTO 89], Doyle [DOYLE 75] to name but a few discuss this matter with several examples and experiences. Ashford & Willet [ASHFO 88] analyse the retrieval functions of several online systems. Throughout these papers, the main emphasis is on the topics of text processing, information structures and the evaluation of the effectiveness of retrieval strategies.

Some ideas used in this project are outlined with more detail in chapters 3 & 5. Most of the principles of information retrieval are adapted in the present work, especially for foreign languages such as

Arabic which is totally different from Latin languages. Finally, this notion of searching relevant documents, which is an important part of information retrieval as detailed above, will be further studied by analysing several search strategies.

2·4 SEARCH STRATEGIES

Most of the search methods depend on the organisation of the information files. The notion of *keywords* is often used; hence, the inverted files method provides the solution in a lot of cases of retrieving documents. These two notions are very close to the intuition of a user who requires documents; a user who has in mind a title or author can remember it by only one or two words for example. If he scans a list of potentially relevant documents, he can retrieve very easily the relevant ones. Therefore, a search method based on inverted files adopts this idea.

The approach consists of the creation of a sequential file containing catalogue titles (authors), and a file which contains all the keywords of such catalogue with each keyword referring to the titles (authors) which contain it. Finally, some search methods discussed in this section are described in more details in chapter 3. Here, we justify their choice for the present project.

[ROWLE 87] gives a number of points on searches which are very relevant in any search method. They are as follows:

- Identification and clear exposition of user's request.
- Translation of the request into the language of the system.
- Comparison of the codification of the request with description of the documentation, in the Database.
- Selection of documents that match the search criteria.

- Receipt and browsing of these documents, or information about them by the searcher.

Before I develop some search strategies, it is useful to know what searching methods are provided in some existing library systems. In a study of 48 online catalogs, J.R. Matthews [MATTH 85] gives a profile for each one. The analysis of such profiles allow us to construct some statistics which could aid the choice of search strategy, menu display and other information useful to this project. Table 2·1 illustrates some of these profiles which show the opportunities of the online systems. Operations numbered from 1 to 14 illustrate the kinds of search or kinds of display available; for example search by author and search by title are held by all the 48 online systems; combination Author/title is performed by 32 ; 12 expand the search results by publisher; practically all have a 'HELP' function to help the user in his search and allow a brief and full record to be displayed.

No	Operation	Yes	No
1	Author	48	0
2	Title	48	0
3	Author/Title	32	16
4	Subject Heading	41	7
5	ISSN	37	11
6	ISBN	37	11
7	Abstract of document	5	43
8	Full text of document	2	46
9	Publisher	12	36
10	Year of publication	24	24
11	Language of publication	18	30
12	Use a 'HELP' function	40	8
13	Brief record	40	8
14	Full record	47	1

Table 2·1: Statistic of some profiles

The user preferences are the searching by title, keywords, and author in that order.

2 • 4 • 1 Method of position keywords

The notion of a *stop-list* is introduced. A dictionary contains the noise words such as pronouns, adverbs, connectives and adjectives. This set of words comprises the stop-list. The same notion may be used for the Arabic Language. This method is based on the "document number" which identifies the document, the "terms" or " keywords" of the document, the "position" of the terms in the document. An algorithm presented by Heaps [HEAPS 78] is adapted in the solutions presented in chapter 3. Inverted files and their associated directory can easily created, especially when sorts are performed by the use of packages. Because there is a dictionary which contains the position of each keyword in any title, this method is powerful in searching for an exact match.

2 • 4 • 2 Method of postings

This method also uses the notion of keywords and inverted files. However, the notion of position keyword is replaced by the notion of postings. P. F. Burton and J.H. Petrie [BURTO 84] describe this notion: «*The entries in the inverted lists are often linked to their respective records in the main file through a postings file which stores the record numbers corresponding to entries in the inverted lists*». This method, which facilitates both the boolean search and the combination author/title search, is also detailed in chapter 3.

2 · 4 · 3 KWIC index

The KWIC (Keyword in context) index method which is based upon the indexed keywords of title documents is very useful for the elaboration of a catalogue. The noise words are not taken into account. As described by J.E. Rowley [ROWLE 85] « *A KWIC index is the most basic of natural language indexes, and is widely used in in-house applications*». Every keyword, arranged in alphabetical order is printed "in context" together with the remainder of the title which contains that keyword. Entry words may be aligned in a centre column or in a left hand column as illustrated by figure 2·1. A brief source reference to lead the user to the document is on the remainder of the the line. For example, consider the following documents

14	H A	Programming in pascal	Edit1	1980
21	A B	Online information retrieval systems	Edit2	1987
24	A B	Information and processing type	Edit3	1982
30	C D	online public access catalogs	Edit4	1981

The KWIC index associated to these documents is

Online public	**Access** Catalogs Edit4 30
Online Public Access	**Catalogs** Edit4 30
Online	**Information** Retrieval Systems Edit2 21
	Information and Processing Type Edit3 24
	Online Information Retrieval Systems Edit2 21
	Online Public Access Catalogs Edit4 30
Programming in	**Pascal** Edit1 14
Information and	**Processing** Type Edit3 24
	Programming in Pascal Edit1 14
Online	**Public** Access Catalogs Edit4 30
Online Information	**Retrieval** Systems Edit2 21
Online information retrieval	**Systems** Edit2 21
Information and Processing	**Type** Edit3 24

Figure 2·1: Example of KWIC index

The KWIC index format is very restrictive. Longer titles will be truncated and only brief source references are included. KWIC indexes are unacceptable for some users because of the awkward and alphabetical arrangement of keywords. Other disadvantages appear when titles contain homonyms, or do not have keywords as is the case in some documents on humanity and social science. For this last case, Habes & Zerouali [HABES 78] propose gathering those titles around a "special" keyword as illustrated in figure 2·2 for the following titles:

learn to drive	67
les conclusions de l'annee 89	58

When mixing these titles with some of those given in figure 2·1 and 'Other' being the special keyword, the KWIC index is

Online	**Information** Retrieval Systems	21
	Information and Processing Type	24
	Online Information Systems	21
	Other learn to drive	67
	Other les conclusions de l'annee 89	58
Programming in	**Pascal**	4

Figure 2·2: Particular case of KWIC index

The use of a KWIC index is carried out by the following steps:

Step1. Enter the title
Step2. Extract the keywords from the title
Step3. Check the keywords in the KWIC index
Step4. Display all titles which contain the entered keywords
Step5. Check the existence of the given title from the displayed titles

The term KWIT index (keyword in title) is often used. The KWIC index is useful for a batch system, and does not require the construction of a separate dictionary of keywords.

2 · 4 · 4 KWOC index

KWOC, or keyword out of context indexes, is also a title index and is constructed in exactly the same way as a KWIC index. However, the display of the index entries differs. The keywords are extracted from the titles and displayed in alphabetical order as a heading; this makes the search by KWOC index easier than the one by KWIC index. Under each keyword the complete title and source reference is given. Finally, the creation of a KWOC index from a large number of documents is quickly and cheaply handled. Figure 2·3 illustrates an example of KWOC index supplied from the same titles used for KWIC index in section 2·4·3.

Access	Online Public Access Catalogs	Edit4	30
Catalogs	Online Public Access Catalogs	Edit4	30
Information	Online Information Retrieval Systems	Edit2	21
Information	Information and Processing Type	Edit3	24
Online	Online Information retrieval Systems	Edit2	21
Online	Online Public Access Catalogs	Edit4	30
Pascal	Programming in Pascal	Edit1	14
Processing	information and Processing Type	Edit3	24
Programming	Programming in Pascal	Edit1	14
Public	Online Public Access catalogs	Edit4	30
Retrieval	Online Information Retrieval Systems	Edit2	21
Systems	Online Information Retrieval Systems	Edit2	21
Type	Information and Processing Type	Edit3	24

Figure 2·3: Example of KWOC index

2 · 4 · 5 Search-key

The search-key is another approach to indexing. According to [MATTH 85], this technique defines a specific index for a title or an author by using a specific formula. For example the search-key for an author (A) index might be AAAA, AAA, A. Thus the search-key for Charles, Antoine L could be CHAR, ANT, L. The search-key formula is sometimes referred to in numbers, e.g. , 4, 3, 1, which correspond to the number of characters used per name. A possible author/title (A,T) search-key might be (AAAA,TTTT). For example an author/title (A,T) search-key for "Faster Evans *Olympic games 1988*" could be FAST, OLYM; this last search-key author/title is used by Glasgow University Library. An example of a title search-key (TTT,TT,TT,T), *Computer-based information services science* is COM, BA, IN, S.

2 · 4 · 6 Boolean search

The boolean search may be used in conjunction with either a controlled-vocabulary or a free- text search. It uses the operators AND, OR, NOT, IF...THEN. The information given in the query is compared with the stored documents. For example, when a formulation of a set of keywords linked by the above operators is entered, the results is the relevant documents for which the formulation is true.

Several methods on search strategies not outlined here may be found in [RIJSB 79], [SALTO 89] and [LANCA 79]; [HENRY 80] and [HARTE 86] detail an online searching context. However, some other techniques such as signature files or compression, as described by Faloustos [FALOU 85a] and [FALOU 85b], might be used for searching documents in relation to any language. The above search strategies could be adapted for searching Arabic documents.

2·5 ARABIC

Because retrieval of Arabic documents is one of the purposes of this work, information about this language is given. Analysis are performed to show the possibilities in including this language with English and French.

Arabic is a complicated language which needs special processing for the characters and the writing from right-to-left as enumerated by M. G. Khayat [KHAYA 86] and J. D. Becker [BECKER 87]. There is no notion of upper case or lower case. Arabic words are morphologically derived from a shorter list of roots. Recently, El-Sadany & Hashish [SADAY 89] develop an Arabic morphological system capable of dealing with Arabic words. A verb may be conjugated in three tenses (past, present and imperative). Different types of diacritic marks are used. The romanization of Arabic is described in [CATAl 70]. There is no notion of upper case and lower case characters.

2·5·1 Arabic characters

There are 28 consonantal characters and several additional characters which may have one, two, three or four shapes each. Such characters are classified by shape into four classes and are presented in Appendix B. It is not possible to write full Arabic text without having additional letters and special characters. The languages Urdu, Farsi, and Maley are non-Arabic languages which use Arabic alphabet. The classification by shape of the main 28 characters is as follows:

One shape: There are 6 characters with invariant shape. However, 4 of them cannot be linked from their left to another character. They are free characters.

Two shapes: There 17 characters. The full-letter shape can be used for isolated and end of word occurrences. The short letter shape can be used for beginning and middle of word occurrences.

Three shapes: There is only one character which has three shapes. The shapes for beginning and middle of word are identical whereas those for end of and isolated word are different.

Four shapes: There are 3 characters where the shapes for beginning, middle, end and isolated word are different.

2 · 5 · 2 Internal representation of Arabic characters

The Arabic alphabet is represented numerically by a standard code for communication interchange elaborated by the Arab Organisation for Standardization and Metrology (ASMO), in a manner similar to the ASCII code of the English Alphabet. Details on this are described in [ARABL 85]. This binary code is presented in Appendix B with a comparison and a correspondence of the English ASCII characters.

2 · 5 · 3 Display and direction layout

A paper by M. M. Aman [AMANM 84] outlines many propositions and devices discussed by the Arab countries to computerize the Arabic language. An example of keyboard layout is well defined by Macintosh in [APPLE. 87] for an Arabic word processing. However, Professor Hyder at the Department of Computing Science, University of Montreal, Canada introduces a method (Method of Hyder) to handle the output of Arabic words which requires that when a key is pressed, the letter is not printed immediately but is stored in the terminal's buffer. When the next character or space is typed the terminal is then in a position to calculate both the correct shape of the first character and the other code(s) needed to perform it. The second letter of the word is displayed when the third letter is keyed, and so the process is continued until the word is completed.

2 · 5 · 4 Output of an Arabic text

The principle of writing an Arabic text is based upon the writing of the word and the method of Hyder. The words of the text are separated by blanks or not depending on the shape of the last character of a word. Let wl, w2, w3, w4, ,w5, and w6 be the words of the given text of figure 2·4; the arrow is the direction of the text. The width of the screen is assumed to be 80 characters. While the words are constructed in a file, the number of the characters keyed is computed and compared to 80.

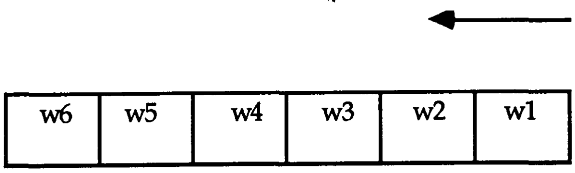


Figure 2·4: Direction of an Arabic text

There is no notion of syllables in Arabic so a word cannot be written in two successive lines.

2 · 5 · 5 Writing a mixed text

Multilingual word processing, as described by J.D. Becker [BECKER 84] outlines the complexities and advantages which arise in computerising many scripts. Approaches to write mixed texts such as English/Arabic or Arabic/English are discussed.

2 · 5 · 6 Deleting and inserting a character

Deleting or inserting characters in the Arabic language is more difficult than in English/French language due to the different character shapes. Such shapes depend on the position of a character in a word. Some cases of deleting and inserting charäcters are described.

2 · 5 · 6 · 1 Deleting a character

a) The character to be deleted is at the beginning of the word. Suppose that is C1 in figure 2·5

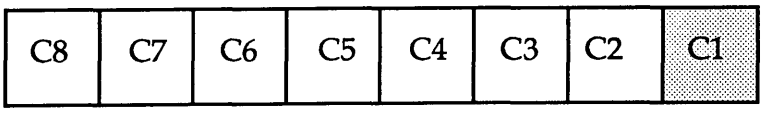


Figure 2·5: First case of deleting a character

If C1 is a free character, the array is shifted from left to right, and C2 becomes the first, its shape not changing. Otherwise, it is deleted, and C2 becomes the first with the shape of beginning word.

b) The character to be deleted is at the middle of the word. Suppose that it is C5 in figure 2·6. Hence, after examination of the shape of C5, C4 and C6, the character is deleted and the correct new word is displayed.

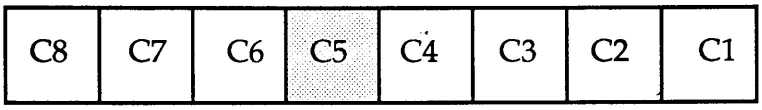


Figure 2·6: Second case of deleting a character

c) The character to be deleted is at the end of the word. Suppose that it is C8 in figure 2·7. C8 is deleted and C7 takes the form of an end word as illustrated in figure 2·7.

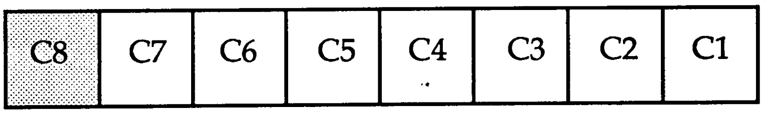


Figure 2·7: Third case of deleting a character

2 · 5 · 6 · 1 Inserting a character

The study of inserting a character is similar the one of deleting.

Indeed, the same considerations on shapes are taken into account. Inserting a character consists of shifting some others to the left and linking the new character to the right side, as shown in figure 2·8 which illustrates the three possibilities of inserting. Let C1C2C3C4C5C6C7C8 be the given word and C0 the character to be inserted. The three grey squares with an arrow show the place where the character C0 might be inserted. When the insertion is at the beginning of the word, C1 becomes middle of word and C0 becomes beginning of word. If the insertion of C0 is at the middle, it becomes middle of word. Finally, when C0 is inserted at the end, it becomes end of word and C8 becomes middle of word.

Several word processing packages which manipulate Arabic are available from IBM, Apple, Rocket Field, APTEC, Technocrat. [BRICS 88] describes work done on computerisation of Arabic, its future and its consequences for libraries. Finally, reduction of Arabic words into root-pattern pairs is widely studied in section 3·3 of chapter 3.

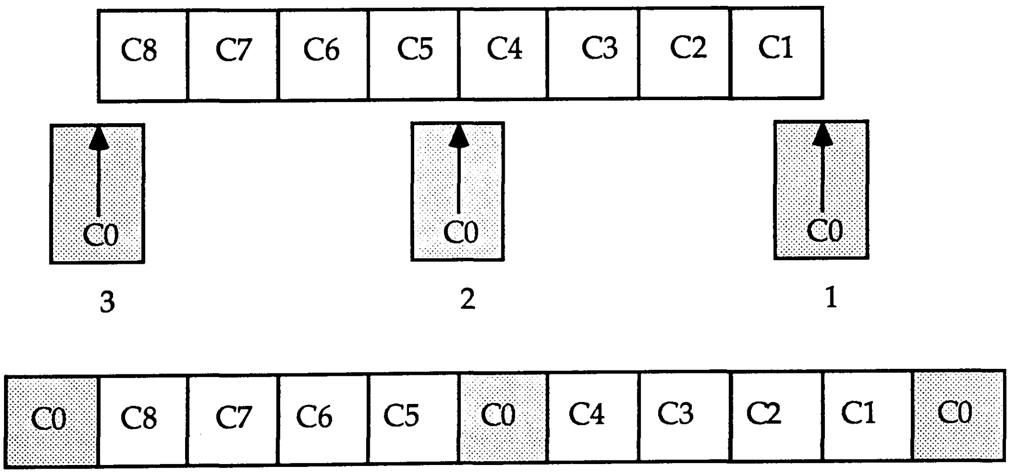


Figure 2·8: The three cases of inserting a character

2·7 EXISTING MULTILANGUAGE LIBRARY SYSTEMS

Some systems which have multilingual capacity already exist such as DOBIS/LIBIS system which can handle at least a dozen languages as disparate as English, Arabic, and Chinese; an example of such a system used in Saudi Arabia, is described by Deemer [DEEME 82]. Some others such as VUBIS system, permit input and output in all standard Western European characters; the GEAC system, using the transliteration of non-roman characters, handles also Arabic documents. An in-house system is provided by the National Scientific and Technical Information Center (NSTIC) of the Kuwait Institute for Science Research (KISR), allowing the manipulation of applications in both English and Arabic, as described by [KHALI 83]. The last in-house system to refer to is the one which belongs to the National Library of Wales, offering research facilities to the University of Aberystwyth. The screen dialogue when readers and/or staff search the database is in either Welsh or English. In fact all screen dialogues in any part of the system may be in either Welsh or English, although the data being entered or retrieved may be Welsh, English, French, German, etc. Finally, the LINGUAFILE system is a package which allows the creation and manipulation of a database in both English and Arabic; some characteristics of this package are described in [BRICS 88].

2·8 SYNTHESIS

Several studies and methods related to the purpose of our project are presented in this chapter. The principles in managing a library have been described, including acquisitions, cataloging, and circulation. Several search methods have been described, some of which are discarded due to their ineffectiveness and the rigidity they present. Automatic classification of the words into keywords, partial match, file structures such as inverted files, boolean search are some of the interesting tools that can be used with success to manipulate a set of mixed documents. Indeed, this study of Arabic leads us to develop a searching system where Arabic and English/French documents might be together.

Chapter 3

MULTILANGUAGE SEARCHING

3·1 INTRODUCTION

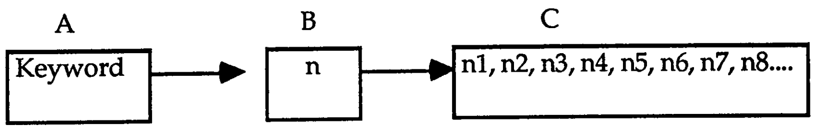
In an experiment on automatic processing of foreign languages, Salton [SALTO 71] concludes that : «... *the methods are evaluated, and it is shown that the effectiveness of the mixed language processing is approximately equivalent to that of the standard process operating within a single language...*». This strategy has a similar approach to the one presented in this thesis. The latter treats both English and French as a single language since they share the same alphabet. The difference in morphology and in language ambiguities does not prevent language merger. In contrast, the Arabic language uses a completely different cryptology which makes its merger with the above languages difficult to achieve. In this chapter, the different file organisations that can be used for document storage and retrieval are presented. Methods of searching English/French/Arabic documents by author, title, keywords, subject, combination author/title,... etc are discussed.

3 · 2 SEARCHING ENGLISH/FRENCH DOCUMENTS

As mentioned in the previous section, the two languages, English and French are unified. Only one file containing the documents (books, periodicals, conference...) is used. Documents are stored using an inverted file data organisation.

The method of inverted files is chosen to retrieve documents needed by the user. Heaps [HEAPS 78] and Lancaster [LANCA 79] introduce three main files: Unit record file, Index file and Inverted file of postings.

The method consists of the creation of entries from textual items numbered from 1 to t (t is the total number of documents). All the noise words are discarded and the rest of the words (keywords) are sorted in alphabetical order. In front of each keyword there is a number which indicates the number of items containing this keyword. Finally, an inverted index contains, for each keyword, a list of the items to which it belongs. Figure 3·1 shows such considerations for a given keyword.



1<= n1, n2, n3, n4,...<t

n1, n2, n3, ... are the document numbers

Figure 3·1: Relation of a keyword with the document numbers

Figure 3·1 gives the main parts (A, B and C) of the tools of the inverted file method which are keyword or entry, posting, and inverted file. [HEAPS 78] finds that the method «*may be regarded as equivalent to a document term matrix of rows and columns in which the rows correspond to document items and the columns correspond to terms*». Hence the *i*th row and *j*th column is set to 1, or 0, according to whether the *i*th document contains, or does not contain, the *j*th term as given in figure 3·2

$$
\begin{array}{c|cccccccccc|}
 & t1 & t2 & t3 & t4 & t5 & ..t23.. & t99.. & t145.. & t1678 & .... \\
1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & .. \\
2 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & .. \\
3 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & .. \\
4 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & .. \\
5 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & .. \\
.. & .. & .. & .. & .. & .. & .. & .. & .. & .. & .. \\
23 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & .. \\
... & .. & .. & .. & .. & .. & .. & .. & .. & .. & .. \\
450 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & .. \\
.... & .. & .. & .. & .. & .. & .. & .. & .. & .. & .. \\
1234 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & .. \\
..... & .. & .. & .. & .. & .. & .. & .. & .. & .. & .. \\
t & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & .. \\
\end{array}
$$

t1, t2, t3, t4, ...are the keywords
t is the total number of documents

Figure 3·2: Matrix representation of inverted file method

Different Boolean operations (AND, OR and NOT) can be used to search for documents. For example the intersection of column 1 and column 4 of figure 3·2 shows that the terms t1 and t4 are in the items 1, 3. This analogy with the matrix representation gives more precision of the tasks of the inverted file method. For a given text (title or a set of words), it suffices to omit the insignificant words and to consider the rest as entries to the different developed dictionaries and then to check whether the entered text exist. The definitions of the three main files given in section 3·2·1, section 3·2·2·3 and section 3·2·2·4 belong to [LANCA 79].

3 • 2 • 1 Unit record file (URF)

«The URF file or linear list is a file organised by document number. It stores the bibliographic information about the documents in the database». Searching documents is the main interest in the present work. The fields are fixed. However, a particular care is given to the field "author" because of the large number of authors who can match the same document. Figure 3·3 describes the records of such file and the total characters associated with each field.

The choice of fixed fields is better than variable fields; most of the fields have either controlled length or unchanged length. They are as follows:

Document Number, Number of Authors, Edition, Year, Call Number, Status, Type (Book, Periodical, Conference....), Language. The field publisher could be truncated when it is necessary (the publisher is also well-known). Subjects and abstracts are defined by the library manager.

1	2	3	4	5	6	7	8	9	10	11	12	13	14
Document Number	Number Authors	First Author	Title	Edition	Publisher	Year	Subject	Abstract	Call Number	Status	Borrower	type	Language
7(D)	1(D)	25(C)	70(C)	1(D)	20(C)	4(D)	40(C)	80(C)	15(A/N)	15(C)	10(D)	15(C)	3(C)

C: Character
D: Digit
A/N: Alpha-Numeric

Figure 3·3: Fields of URF file

A maximum length is given to the field title to cover all possible titles and avoid using tags. However, with such a representation, fields of short titles are completed by blanks. Finally, the only field which is not controlled is the field author; some solutions are listed below:

3 · 2 · 1 · 1 First approach

The record document described above shows two special fields: number of authors and first author; the number of authors being between 1 and 8 is the exact number of persons who participated in the writing (or elaboration) of the document. While URF is created, authors related to the same document and to the document number are put in a transit file (TPF). These files are used alternatively. The advantage of such a structure is to present exclusively the main author in the record of

URF. Such representation has a main interest in the case of a search by author.

Another file named 'transit linked authors' (TLA) and containing the authors linked to the main author (for a given document) is added; it is used as a complement of information needed. Suppose, the following documents of figure 3·4 are to be stored, only some brief records are related to the authors. Figure 3·5 shows the organisation.of the TPF and TLA files.

Doc Number	Author(s)
1400	X, Y, F
1401	A
....	....
3211	D, A, R, X
3212	F, H, R, E, W, Q
....	....
10098	Z, A, Y

Figure 3·4: Brief records

3 · 2 · 1 · 2 Second approach

The second solution limits the number of authors. Only the two first authors are taken into account. Most documents written collaboratively involve only two authors. The field length is a set of characters between 20 and 50. The field number of authors is not necessary. This solution is chosen to test my system with a field length of 20 characters.

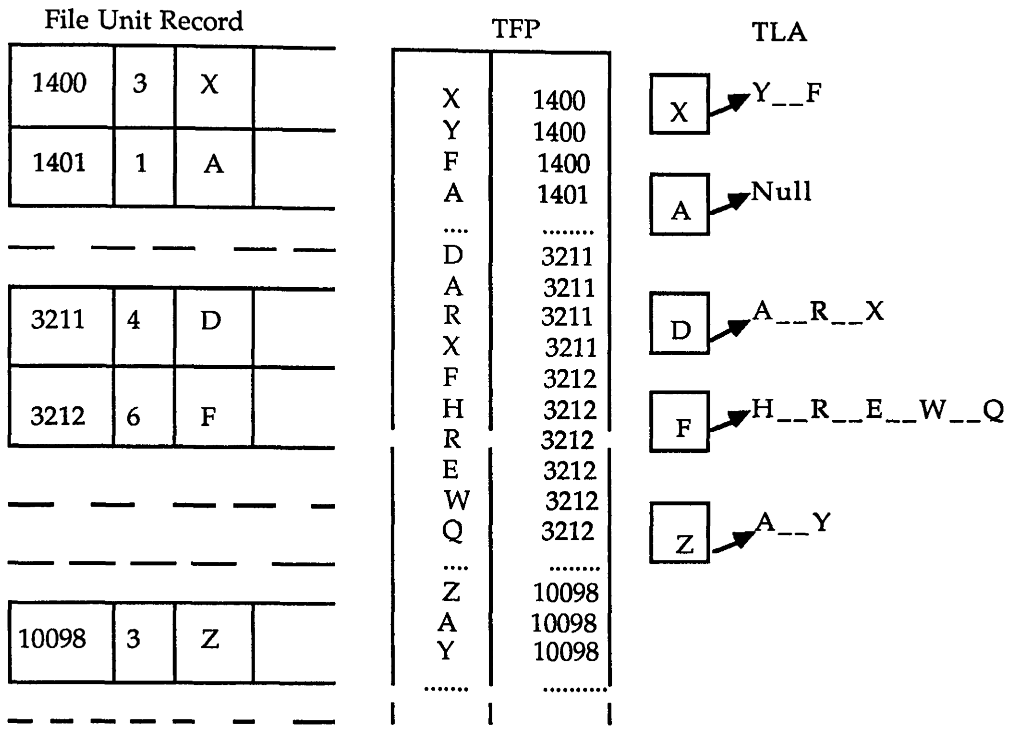


Figure 3·5: Transit files TPF and TLA

The call number is an internal identification of the document; it gives the physical storage of the document inside the library.

The status indicates whether the document is in the library, on loan or missing.

The document-number is the identification of the document; these numbers are defined in chronological order due to the following reasons:

- Facilitates the access to the file URF when the search of documents starts.

- The library manager knows exactly at any time the volume of documents.

3·2·2 Definitions

Some dictionaries and files are created to emphasise the structure of the multilanguage system and to construct the search of documents.

3·2·2·1 Dictionary stop list (DSL)

DSL is a dictionary which contains the noise words such as: *a, an, also, conclusion, from, in, some the,* .... for English; *l, le, la, les, ce, dans, pour, se,* ... for French. The number of such words is finite for each language. However, these words are mixed together, sorted, and stored in DSL.

3·2·2·2 Keyword

All words from titles, subjects and abstracts (which are not noise words) as well as the names of authors are considered as keywords. In fact, "positive" keywords are keywords listed from the documents of the URF file and are stored in the IXF file (see section 3·2·2·3).

3 • 2 • 2 • 3 Index file (IXF)

«It is an alphabetical list of the searchable terms (or author) in the database». The postings are associated with each term, they represent the number of documents to which the index term is assigned.

3 • 2 • 2 • 4 Inverted file (IVF)

«The inverted file stores, for each term (or author) appearing in the index file, a list of all the documents numbers to which this term applies; that is, the list of numbers of all the documents to which this index term has been assigned ». In the case of the terms issued from titles, their position in the inverted file is given.

3 • 2 • 2 • 5 Position of keyword (PSK)

[HEAPS 78] introduced this notion to facilitate the exact match while searching documents using inverted file structure. The PSK is defined as the rank of word after suppression of all noise words in a given text (title, subject or abstract). For example, in the text 'the components of the information retrieval systems', there are 7 words. Among them, 3 noise words: 'the', 'of' and 'the'. The keywords are, in order with respect to their position: components(1), information(2), retrieval(3), systems(4).

With these definitions, it is now possible to construct a model of searching documents in English/French. However, some modifications have been performed in order to use my system. They are as follow:

- all dictionaries are sorted in alphabetical order.

- The PSK dictionary is not taken into account in order to minimise the searching time.

- Different entries described by W. M. Henry & al [HENRY 80] are applied; either entry in a pure dictionary (dictionaries are separated in author dictionary, keyword dictionary, and abstract dictionary), or entry in a mixed dictionary (one dictionary which contains keywords from titles , subjects, abstracts and authors). The advantage of the mixed dictionary is to facilitate the search in case of combination of author and title. The implementation of the system can use either the pure dictionary or the mixed dictionary.

- All the words of the author are considered as entries in the dictionaries; this is to facilitate the search for the user who has knowledge of only one part of the author's name.

- The subject is a set of 'keywords' defined by the library manager. The number of documents retrieved using such a search is large.

- Each specimen (e.g., duplicate document) is considered as a separate document and has its own record.

3 · 2 · 3 Dictionaries

Salton [SALTO 89] describes several search methods by using inverted file, search key, to name a few of these, only inverted file has been implemented in my system. Search by title/keyword/author are the only ones considered in my work. Other useful dictionaries are described below. For instance, let us examine the following documents of figure 3·6.

101	1	knuth, donald	the art of programming
102	1	stephen, harter	online information retrieval
.....	..	..............., ...	....
110	2	bruno, frappat	l'economie du tiers monde
111	1	donald, alain	le sport en afrique
112	1	maxim, stephen	la recherche documentaire
....	...	................., ...	....
1121	1	stephen, salton	library automation systems
1122	1	fritz, lancaster	information retrieval systems

Figure 3·6: Example of documents

Using a mixed dictionary, figure 3·7 shows the different entries (IXF file), IVF and URF files associated with the documents of figure 3·6. Each index term (keyword) with its posting and the set of documents associated with this keyword are shown.

The eventual search of a document is based on Boolean search by taking the intersection of the index terms of the user's query (entered text). While the intersection is not empty, all relevant documents are retrieved. Otherwise, index terms and their postings are output to the user, and a search continues normally by using separate keywords on his request.

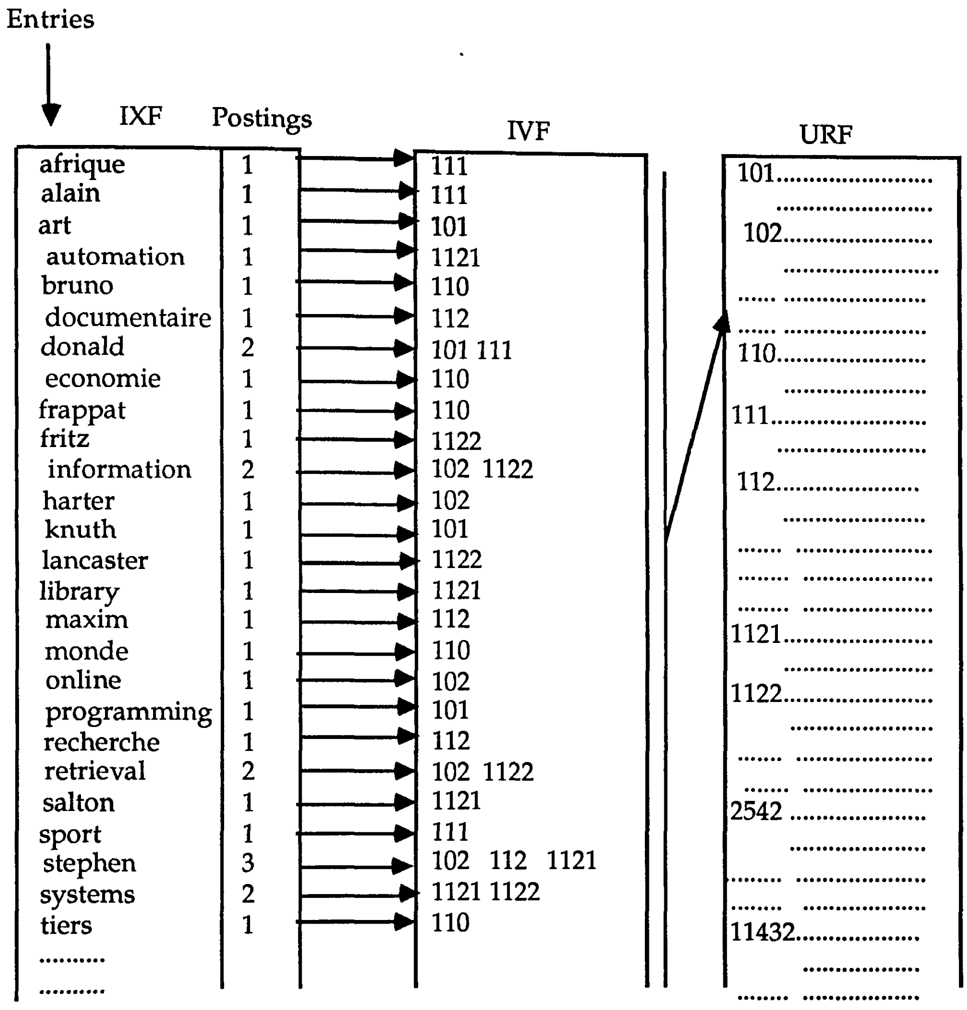


Figure 3·7: Entries in a mixed dictionary

3 · 2 · 3 · 1 Creation of dictionaries

The pure or mixed dictionaries are created from the URF file; the DSL dictionary is used when it is necessary. Each word from any title, subject, abstract, or author is compared to the noise words. Figure 3·8 shows the different dictionaries in a phase of their elaboration.

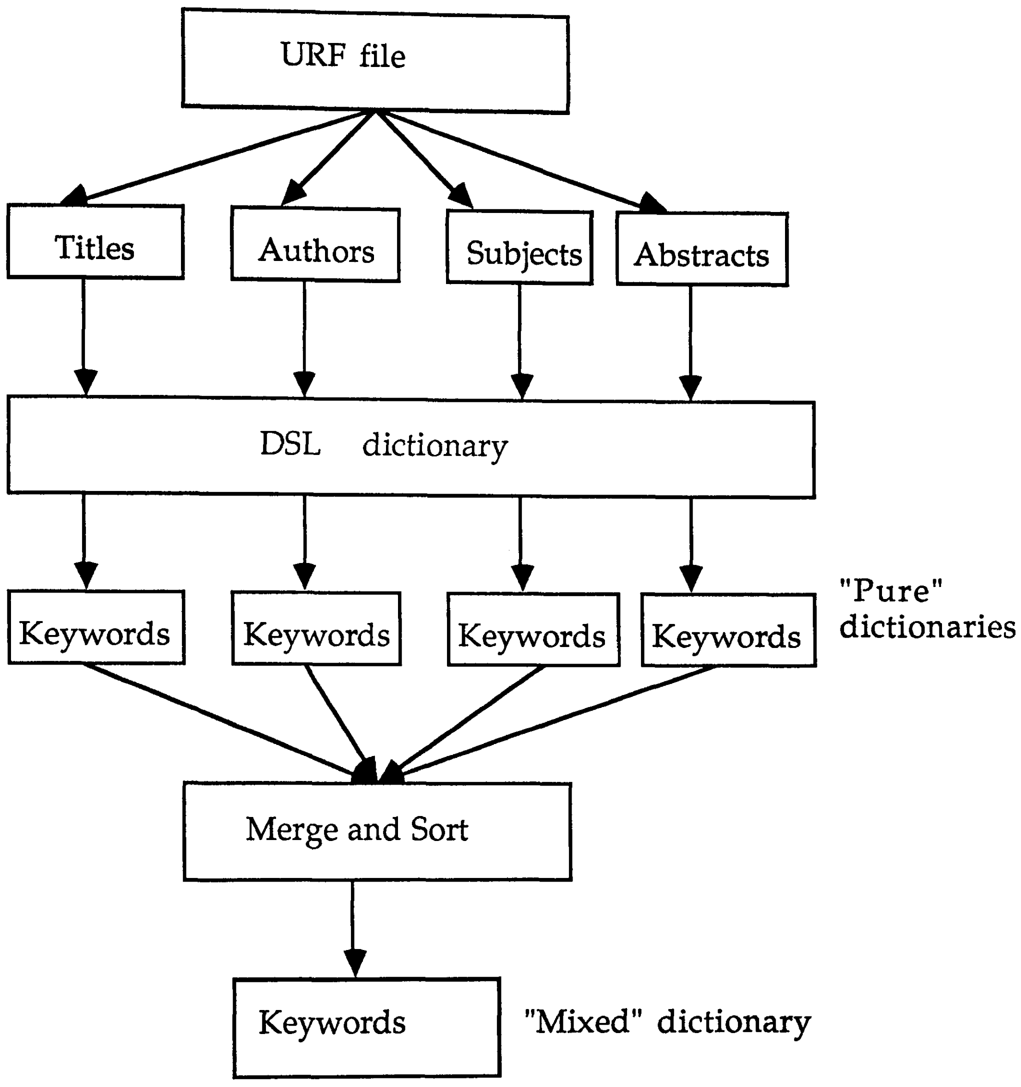


Figure 3·8: Pure and mixed dictionaries

3·2·3·2 Updating the dictionaries

The URF file is updated whenever the document is either borrowed or lost. In the case of acquisition of new documents, new records are created and, therefore, the URF file is extended. New document numbers, titles, authors, subjects and abstracts are added. Old dictionaries become outdated and these need to be updated in order to satisfy the users in their search and to bring them information about the new documents. The dictionaries being sorted in alphabetical order are updated or recreated at regular intervals, such as every week. This might depend either on the arrival of new documents or the library management organisation.

Since the updating is too time consuming for pure dictionaries, a mixed dictionary is preferable. There are two methods for creating the dictionary:

- The URF file is read, record by record. Keywords from titles, authors, subjects and abstracts are merged together in a file. Then the file is sorted.

- Again the URF file is read record by record and the pure dictionaries are created; thereafter, they are merged and the result sorted.

The first solution is preferred because there is only one dictionary to create and to update. However, only titles and authors are considered in my system and the mixed dictionary is recreated every time in the case of new documents. Chapter 5 outlines these updating mechanisms in more detail and deals with the consequences of the different choices described above.

3 · 2 · 4 Search approach

Figure 3·9 shows the different phases of the search approach.

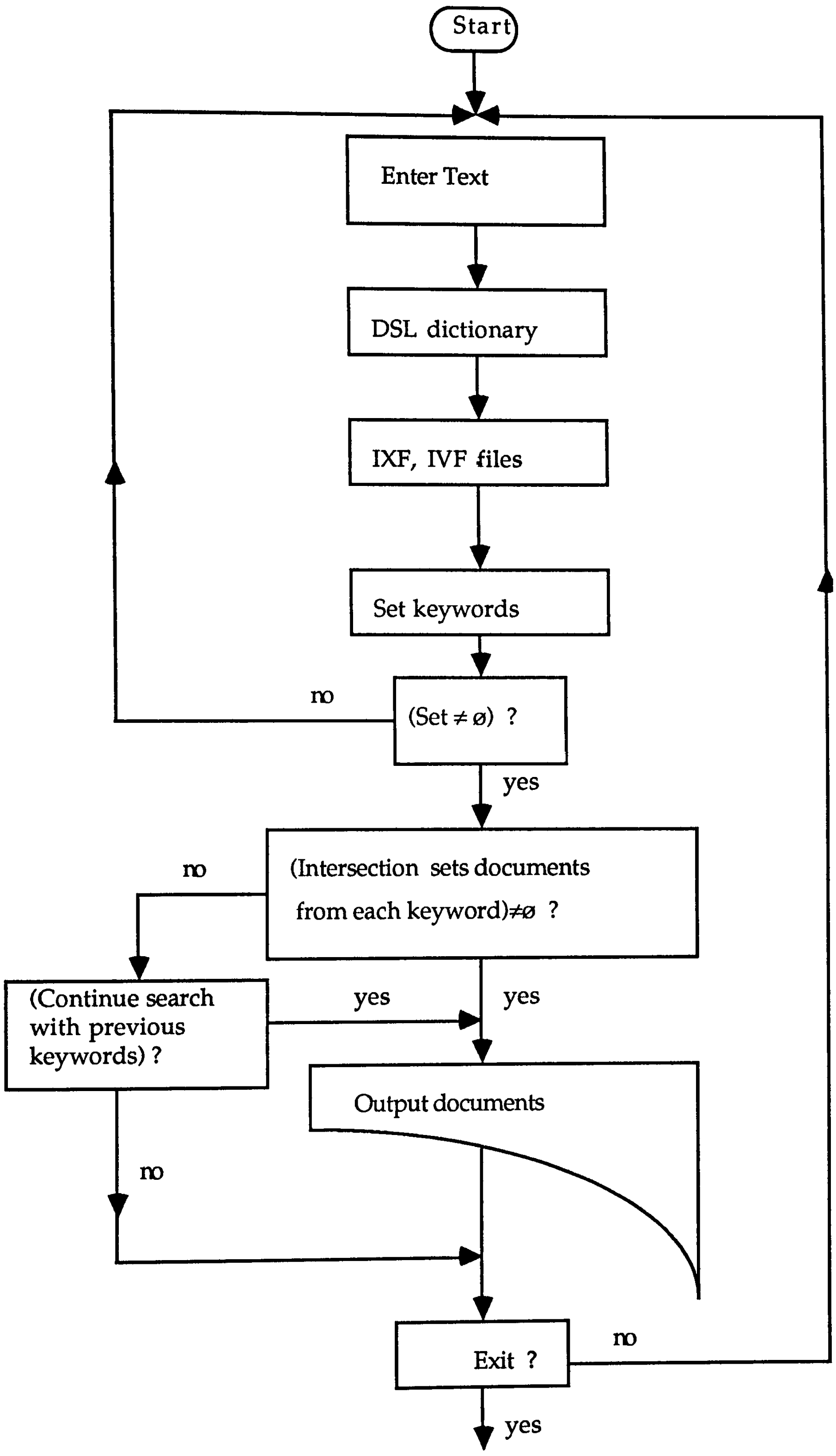


Figure 3·9: Different phases of the search.

3 · 2 · 4 · 1 Text Input

The text emphasizes the type of search (title, author, combination author/title, subject or abstract). It is a set of characters which is keyed by the user. The text is either in English and/or in French.

3 · 2 · 4 · 2 Elimination of noise words

Once the text is input, it is decomposed into words which are compared with those of the DSL dictionary. Consequently, all the noise words are eliminated from the text and the remaining words are keywords. This is called the first filtration. Let there be *e* keywords remaining. Section 3·2·4·3 describes the last filtration of keywords. However, when *e* is zero, the user has to key another text to continue the search.

No study concerning singular/plural, masculine/feminine, roots and grammar of the words has been considered. This was considered to be outside the scope of the project. However, the notion of term truncation studied by Stephen P. Harter [HARTE 86], H. S. Heaps [HEAPS 78] and W. M. Henry & al [HENRY 80] allows us to cover this subject matter.

Some authors'names contain noise words such as names of organisations (e.g. organisation Mondiale de la Sante), or some French names (e.g. Jean de La Fontaine, La Bruyere, Guy de Maupassant) where 'de' and 'la' are noise words. These noise words are discarded.

3 · 2 · 4 · 3 IXF, IVF files

The *e* keywords are compared to the index terms of file IXF to prove their existence in the URF file. Hence, the set of keywords (the 'positive keywords') is defined and the number of elements is e' ($e' \leq e$). Thereafter, knowing the postings, each keyword is linked to its associated set of document numbers. The IVF file is used for this last operation Figure 3·10 shows an example of such sets.

Set_i is a list of document numbers to which the keyword K_i applies ($0<i<k$, k being the number of keywords). Set_i depends on the original text (title, author, author/title, subject or abstract)

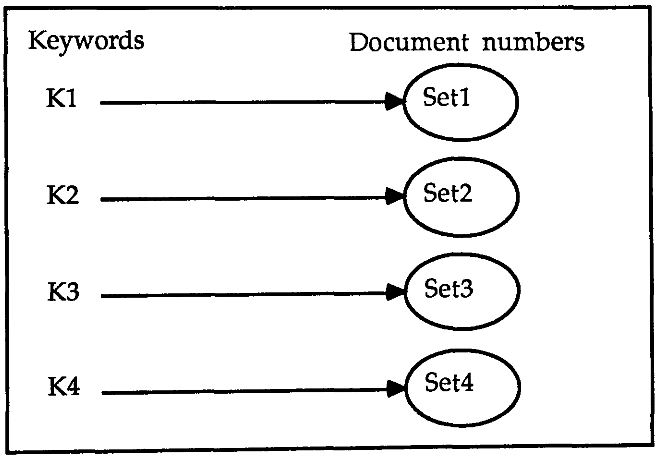


Figure 3·10: link keywords/sets document numbers

3 · 2 · 4 · 4 Intersection of sets

Once the pairs (K_i, Set_i) have been defined, the search starts. Two types of document matching are presented :

Exact or partial match: The exact match or partial match for retrieving relevant documents which contain all keywords of the given text is held (e.g. when $e'=e$ and the intersection of the sets Set_i is not empty). Note that the position of the keyword is not taken into account. For example, take the text 'information retrieval', the retrieved documents are:

- The computer in information retrieval
- Information retrieval
- Information retrieval systems
- Retrieval systems and information processing

Keyword search: Either the intersection of the sets Set_i is empty or $e'<e$, whereupon a keyword search is applied. The keywords and the associated postings are output to the user who can decide about an eventual search.

3 · 2 · 4 · 5 Output documents

After analysing the online catalogs introduced by J R. Matthews [MATTH 85], most of the systems allow to output either a brief record or a full record, depending on the user's needs. In the case of a brief record, after a search is completed the documents are output to the user with the following information: title, author, publisher, status. For a full record, more information is given: year, language, subject, abstract.

3 · 2 · 4 · 6 End of search

The user has the alternative either to stop the search at anytime and quit from the system, or to carry out another search (more details are given in chapter 4).

The search system is able to handle other languages having the Latin alphabet (Italian, Spanish... etc). The searches allowed by my system are title, author, author/title and keywords. Finally, similar considerations and definitions are used in section 3·3.

3 · 3 SEARCHING ARABIC DOCUMENTS

Searching for Arabic documents seems a difficult task compared to the search of English/French documents. Ali Nabil & al [NABIL 84] present a comparison between Arabic and English languages in a study of a compression technique for Arabic text. The words (nouns, verbs, noise words, suffixes, infixes, prefixes) are classified and codified with special rules due to the complexities of the language. However, the introduced codes are confirmed to be insufficient to cover all Arabic text. Hence, the technique is not efficient to be applied for retrieving documents.

Another method such as 'signature files' presented by C. Faloustos [FALOU 85a], [FALOU 85b] is able to handle any Arabic text; one reason is the availability of the binary code for Arabic characters. Each word of an Arabic title is hashed into a bit pattern of length *f*. These patterns or word signatures are concatenated to form the document signature. Searching is performed in the obvious way. For example, on a single word query the signature of the search word is extracted and all the document signatures are searched. Those that contain the signature of the search word are retrieved. In order to improve the performance, noise words may be ignored. However, for each kind of search, a signature file is used. That means, titles, authors, subjects, abstracts or keywords have their own signatures. This may result in a large number of collisions. The method is slower than inversion for a large file.

Finally, the decomposition of Arabic words into the pair, root and pattern, and their storage in the form of inverted files can be used to produce a method for the search of Arabic documents. The existence of roots (i.e a group of letters) for most words is a major characteristic in the Arabic language. Each of these words can be reduced to its root. There are four kind of roots. According to [NABIL 84], there are 7597 triliteral roots (63%), 4081 quadriradical roots (34%) and 300 pentaradical roots (2.5%). There are 50 biliteral roots (0.5%). The total number of roots is 12028; a special study of reduction words in triliteral roots is described in section 3·3·2. However, the words which cannot be reduced in any of the existing roots are irreducible words and are considered to be roots themselves. Most of the arabised foreign words belong to this category.

The method used for searching for English/French documents that is inverted files method described in section 3·3 is adapted to the Arabic documents. Indeed, all the reducible Arabic words will be replaced by their root-pattern pair which represent a keyword and then the inverted files method is applied by using postings and document numbers. This method is also used in case of search by keywords and more details are developed throughout this section.

3 · 3 · 1 Definitions

Words in the Arabic language are nouns, verbs, adverbs, adjectives, pronouns, prepositions... etc. This is quite similar to the type of Latin languages. However, some differences exist:

- The noise words (or tools) can be stand alone or linked to words For example 'al' which means 'the' in English is always linked to the word.
- The great majority of nouns and adjectives may have up to four different forms, namely: singular/plural and masculine/feminine.

The number of files and dictionaries created for Arabic documents is greater than those associated with English/French documents. However, the structure of some files are similar to those defined in section 3·2·1. These are the Unit record file (URF), TPF, and TLA files. Definitions and conceptions of title, author, subject and abstract are maintained.

3 · 3 · 1 · 1 Arabic text

An Arabic text is a set of words not necessarily separated by spaces. For instance, the shape of the last character of an Arabic word may be sufficient to indicate the end of the word. However, this study assumes that words are separated by spaces in the stored form of the text in order to identify all tokens. This facilitates the decomposition of the entered text.

3 · 3 · 1 · 2 Root and pattern

Root: Before I develop the notion of the root in Arabic, it is interesting to describe the one in English and then compare them. In English, the root is at the beginning, the middle or the end and where the words have all the same root and a close meaning. Thus, the words 'compute', 'computer','computing', 'computation' have the same root 'compute'. This concept is also used in Arabic but the characters of the root are not grouped together at the beginning, the middle or the end of the word. The order of the characters of a root is unchanged for all the words which have the same root whereas the rest of characters are different. For example the words 'kitab' (a book), 'kutub' (books), 'kataba' (to write), 'maktaba' (a library) have the same root which is 'ktb'. A root is a set of the main (origin) characters of a word. It cannot be reduced to another word.

Pattern: A pattern is a string of Arabic characters containing the basic verb "F'AL" and any prefix, infix and/or suffix attached to it. Every Arabic morphological form is built according to a pattern and a root. Hence, any word in the language can be reduced to a pattern and a root. Al-Fedaghi [FEDAG 88] proposes a reduction technique to produce the roots and the patterns of words.

Two tables ROOT and PATTERN are used in this study. The first is a set of all triliteral roots in Arabic, and the second is a set of all patterns of this group of roots.

3·3·1·3 Arabic stop list dictionary (ASL)

The ASL dictionary contains the stand alone tools (prepositions, adverbs, adjectives... etc) and the tool-tool links. The number of noise words is finite. Some of them are linked to nouns (possible keywords) and are difficult to isolate (for example 'be' which is a romanised Arabic noise word and means 'with' in English). Such tools are considered to be part of the word.

3·3·1·4 Keyword

The conception of keywords is slightly different. Two classes of words (which are not noise words) are introduced. Those which are reducible into root-pattern (class one) and the others which are irreducible (class two). The previous definitions of keyword, index file, inverted file (see section 3·2) apply to the second class. The position in the text of the keyword is not considered.

3·3·2 Reducible words

Since the choice of inverted files method appears to be adequate for the purpose of this research, keywords have to be defined. The introduction of reducible words is necessary and useful. It facilitates the manipulation of Arabic documents. The major part of words are triliteral. A thorough study of such words is presented in this section.

3·3·2·1 Method of reduction

Arabic characters are read right to left. The form of a given word can be formally described by the following notation: F1=Z(R3)W(R2)(R1)YX , where R1, R2 and R3 are the root characters and X, Y, W and Z are those characters which are added to the morphological extension. 'YX' is a prefix, 'W' is an infix and 'Z' is a suffix.

The morphological pattern of the above form is F2=Z(L)W('A)(F)YX where "F", "'A" and "L" are the Arabic letters to replace those of the root in the morphological pattern. Let "l" the length of the word.

The string (R3)(R2)(R1) (read from right to left) is the root of the word and the expression denoted by F2 is the pattern. The expression F1 shows that the root characters do not have a fixed place in an Arabic word. They are mixed together with the other characters

The root-pattern pairs of a word is obtained as follows: every combination of three characters is compared to the elements of the table ROOT. Once it is found to be a root R, the expression F2 is used to generate the corresponding pattern P which in turn is compared with the elements of table PATTERN. Hence, the pair (R, P) determines W. Examples are shown in section 3·3·2·2. A reducible word may have more than one root-pattern pair (R, P).

3 • 3 • 2 • 2 Cases of triliteral roots

Knowing the importance of the triliteral roots in Arabic, a complete study is necessary. Some roots lose characters and are degenerated and others have unchanged characters. Three groups of such roots depending on the Arabic words are:

Case 1: Some letters in a triliteral of a word may be changed when writing the word. The triliteral roots do not lose any character. The number of combination to be tested is $c= l(l-1)(l-2)/6$. Let us call this case S1. For example, let us consider the word $W=C_8C_7C_6C_5C_4C_3C_2C_1$. If $R = C_6C_3C_1$ is a root, the pattern is $P=C_8C_7(L)C_5C_4('A)C_2(F)$ (R and P are respectively looked up in tables ROOT and PATTERN).

Case 2: Some triliteral roots (which are about 500 in number) lose their last letter when forming an Arabic word because it is identical to the second letter. A table TWO ROOT containing all the first two characters of such roots is provided. This new table is extended to contain the first two characters of all the triliteral roots. This is to accelerate the decomposition of an Arabic word (see section 3·3·2·3). The number of combination to be tested is $c= l$. This case is referred to as S2. For example, let us consider the word W= $C_7C_6C_5C_4C_3C_2C_1$. If the substring C_4C_3 is in table TWO ROOT, the root of W is R= $C_4C_4C_3$ and the pattern is P= $C_7C_6C_5$(L)('A)(F)C_2C_1. (R and P are respectively looked up in tables ROOT and PATTERN).

Case 3: Some triliteral roots (about 2000 in number) lose one of their three letters during the morphological inflection, it is often a vowel. Table VOWL contains the 10 vowels and table LOSTV contains the corresponding roots. The table LOSTV is included in table ROOT (see section 3·3·2·3). The number of combinations to be tested is $c= 5(l-1)l(l+1)$. Let us call this case S3. For example, let us consider the word W= $C_7C_6C_5C_4C_3C_2C_1$, v is a vowel added to W at the iteration k, W becomes W'= $C_7C_6vC_5C_4C_3C_2C_1$. If R= vC_5C_4 is a root then the pattern is C_7C_6(L)('A)(F)$C_3C_2C_1$ (R and P are respectively checked in tables ROOT and PATTERN).

Cases S1, S2, and S3, are developed in detail and applied to every reducible word W of an Arabic text (see chapter 4). W belongs to one of the cases (case 1, case 2 or case 3). Error handling is done when necessary. The tools (noise words) not reducible, are eliminated before the decomposition of the text.

3·3·2·3 Organisation of table ROOT

The reduction of words depends on the cases described above. The table ROOT can be organised and accessed in many different ways. These are as follows:

First organisation: Table ROOT contains all the triliteral roots. Figure 3·11 shows the reduction of a word into root-pattern pairs with such an organisation. All the possible combinations of three characters are tested in table ROOT. The latter is accessed about *c* times (see section 3·3·2·2). However, the study of the first two characters of some of the combinations is enough to prove they cannot be roots.

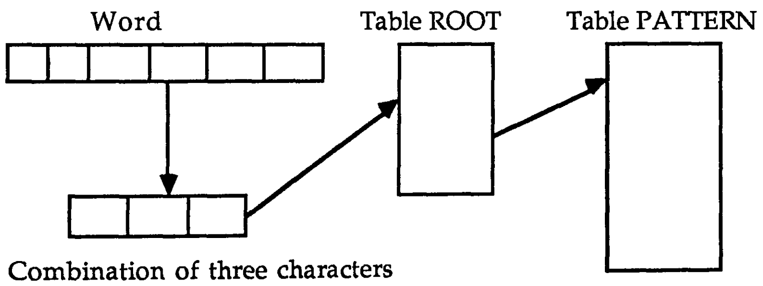


Figure 3·11: determination of pairs (R, P) of word, first version

Second organisation: Table ROOT contains all triliteral roots. A new table, TWO ROOT is used as a filter containing all the first two characters of the elements of table ROOT. For a given word, the first two characters are tested in table TWO ROOT, thereafter an eventual checking of the combination is made in table ROOT. Figure 3·12 shows the process. Table TWO ROOT is accessed about *c* times while table ROOT is accessed about *c* - *e* times, *e* being the number of the combinations which cannot be roots after analysis of their first two characters.

Third organisation: In this organisaton, table ROOT is divided into tables ROOT1, ROOT2 and ROOT3 according to the three root types. Each table has its own size and entry. Figure 3·13 shows such organisation.

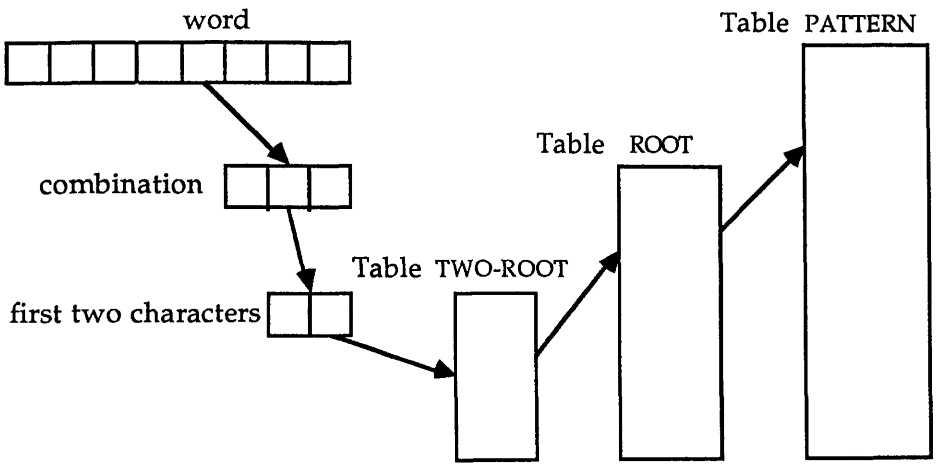


Figure 3·12: Determination of the pair (R, P), second version

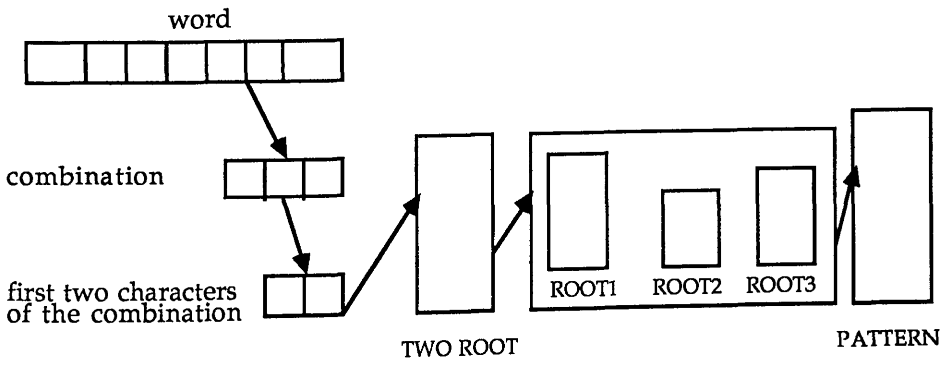


Figure 3·13: Determination of pairs (R, P), third version

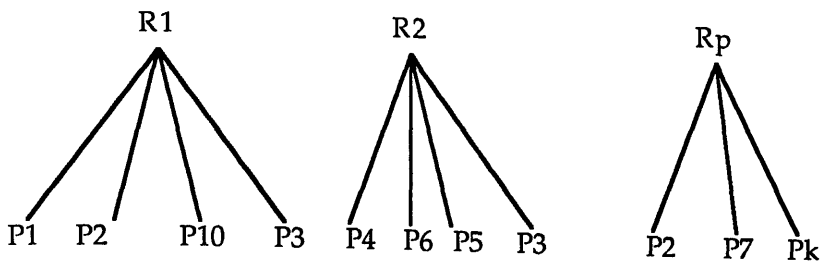
The second organisation is chosen for the purpose of this study. Indeed with this choice, the scanning of table ROOT is reduced and the roots defined only by two characters as described in case 2 of section 3·3·2·2 are combined with the elements of table TWO ROOT. Chapter 6 introduces a section about the consumed time associated when each organisation is used.

3 · 3 · 2 · 4 Index files

The keyword is a reducible word. It is defined as a pair root-pattern pair (R, P). New index files are created from the Arabic URF file; definitions of sections 3·3·2·2 and 3·3·2·3 are used.

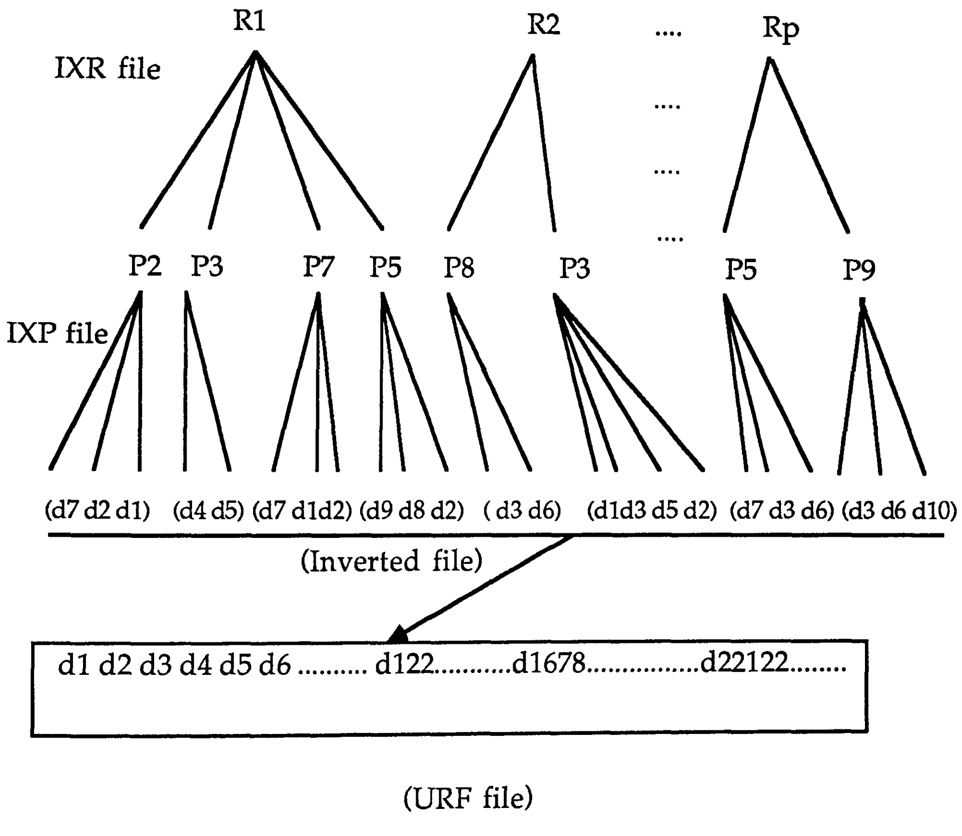
Index file roots (IXR): The IXR file contains the alphabetical list of triliteral roots existing in URF file (roots from titles, subjects and abstracts) and the number of postings associated with each root, that is, a count representing the number of patterns in relation to that root. All keywords with the same root are gathered around this root. Figure 3·14 shows this approach.

Index file patterns (IXP): The IXP file stores, for each pair (R, P) of the keyword appearing in the IXR file, a list of all document numbers to which this pair applies; that is, the list of numbers of all documents to which this pair has been assigned. This concept is shown in figure 3·15 where for example, the keyword (R1,P7) is in the documents d7, d1 and d2.



R1, R2, ...Rp are the roots.
P1, P2,.............Pk are the patterns

Figure 3·14: Relation between the roots and the patterns


R1
R2
....
Rp
IXR file
P2 P3 P7 P5 P8 P3 P5 P9
IXP file
(d7 d2 d1) (d4 d5) (d7 d1d2) (d9 d8 d2) (d3 d6) (d1d3 d5 d2) (d7 d3 d6) (d3 d6 d10)
(Inverted file)
d1 d2 d3 d4 d5 d6 .... d122...........d1678................d22122........
(URF file)

R1, R2, ... Rp are the roots
P1, P2, P2,.... are the patterns
d1, d2, d3,.... are the document numbers
The couple (Ri, Pk) is a keyword (1<i<n, 1<k<m)

Figure 3·15: Relation pairs (R, P) and document numbers

Notions of pure dictionaries and mixed dictionaries as defined in section 3·2 can be used. Figure 3 ·16 shows this.

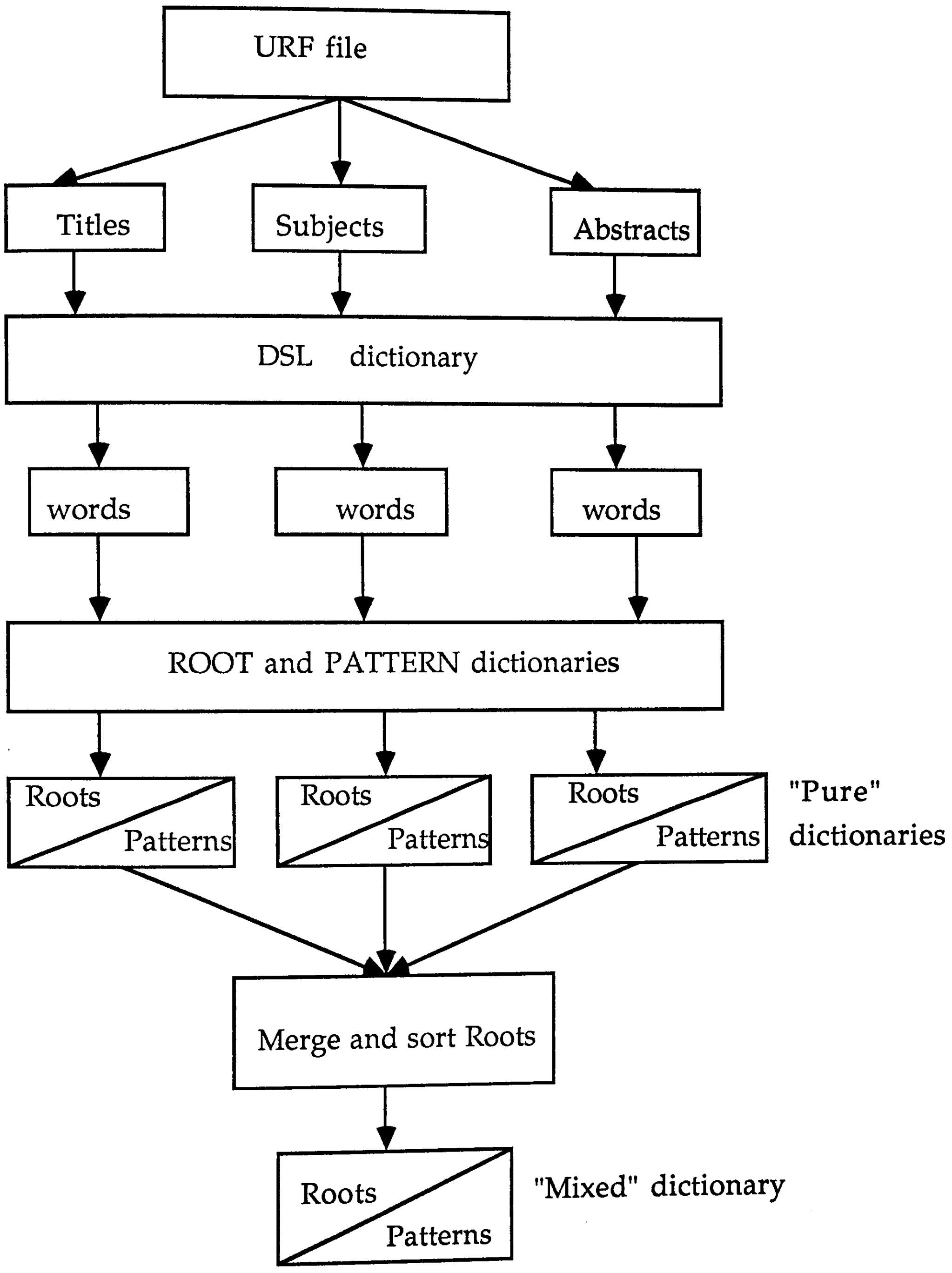


Figure 3·16: Pure and mixed dictionaries for Arabic

The mixed dictionary is also used for Arabic words. This is to facilitate the search for documents, especially the combination author-title.

The complete study of triliteral roots can be adapted to the quadriradical and pentaradical roots (respectively 34% and 2.5% of the total number of roots). However, the number of combinations for a given word is of the order $l(l-1)(l-2)(l-4)/24$ and $l(l-1)(l-2)(l-3)(l-4)/120$. respectively. The reduction of the words having these roots is an expensive operation and uses much memory space.

Because most of the words appearing in titles, subjects, and abstracts are in general generated from triliteral roots, only these roots are considered in the system described in this thesis. All other words are considered to be irreducible words.

3 · 3 · 3 Irreducible words

There is no root for such words as defined in section 3·3. They are classified as class two, relatively to the reducible words, and are not noise words. The words of figure 3·16 which are not decomposed into root-pattern pairs are necessarily irreducible words, also the notion of mixed dictionary is maintained in this case.

The arabised foreign words (either technical words or author names and international words) are irreducible. Examples of this are 'bank', 'computer', 'Algol', 'Fortran', 'Issac Newton', etc.

The Arabic author names are considered irreducible words; the eventual noise words in such names are discarded.

Quadriradical, pentaradical and biliteral roots have not been examined. Words which belong to these roots are also considered irreducible.

Consequently, these irreducible words are keywords. Most of the considerations about keywords, defined in section 3·2, are valid here. Two files are created for this purpose: the IXIR index file of irreducible words and the IVIR inverted file of irreducible words (using the Arabic URF file).

IXIR file contains the mixed keywords (from titles, subjects, authors and abstracts) and the associated postings. IVIR file holds the document numbers relative to each keyword. These two tables are vital. They are necessary for a complete search of an Arabic text.

3 · 3 · 4 Search documents

The section 3·3·3 has prepared all the tools necessary for the search of Arabic documents. Some concepts described in section 3·3·2 are used here. The following phases guarantee the different steps of the search.

3 · 3 · 4 · 1 Text input

The text is either a title, an author, a combination author-title, a subject or an abstract. It is a set of characters keyed in by the user.

3 · 3 · 4 · 2 Text decomposition

The text is decomposed into words which are looked up in the noise word dictionary. Then all the noise words found are eliminated from the text. The remaining words are reducible and/or irreducible words. Let *p* be the number of such words. In the case when *p* equals zero, the user has to enter more text to continue the search.

No special study of singular/plural and masculine/feminine is carried out. These forms are totally different. The nouns may appear in the four forms. For example, the plural of **walad** (boy) is **awled** (boys); the root is the same for singular and plural noun.

3 · 3 · 4 · 3 Separation of reducible from irreducible words

The text, having been emptied of the noise words, the rest of the words are divided into two sets, *set1* and *set2*, with respective sizes *r* and *n*. The simplest way is to check whether any word W is reducible into a root-pattern pair. If W is in ['S1', 'S2', 'S3'] (see section 3·3·3), it is reducible and is put in *set1*; otherwise W is put in *set2*.

set1 contains the pairs (R, P) of the reducible words of the text, where *set2* contains the irreducible words of the same text. *set1* and/or *set2* may be empty.

3 · 3 · 4 · 4 Sets of document numbers

The files IXR, IXP of section 3·3·2·3 and the files IXIR and IVIR of section 3·3·3 are used to determine whether the required documents exist in the library. The elements of *set1* are compared to those of files IXR and IXP. Each existing pair (R, P) or element is linked to its associated set of document numbers. Red_i is a list of document numbers to which the keyword defined by the pair $(R, P)_i$ applies ($0< i<r$)

The elements of *set2* are compared to the index terms of file IXIR. Once the existence of the keywords K_j is proved, the IVIR file allows the storage of the list of document numbers to which the keyword Kj applies in IRed_j $(0<j<n)$. The keyed text contains an average of 10 words (including the noise words) or more, depending on a users' needs. This gives a valuation of r and n. Figure 3·17 shows an example of sets Red_i and IRed_j.

3 · 2 · 4 · 5 Retrieval documents

After the text is analysed, the answer to the user's query is ready. The definitions of exact or partial match and keyword search used are similar to those of section 3·2·4·4.

Exact or partial match: The exact match or partial match search is complicated due to the different classes of words. To retrieve documents containing these words, one of the following cases can happen (INT means intersection):

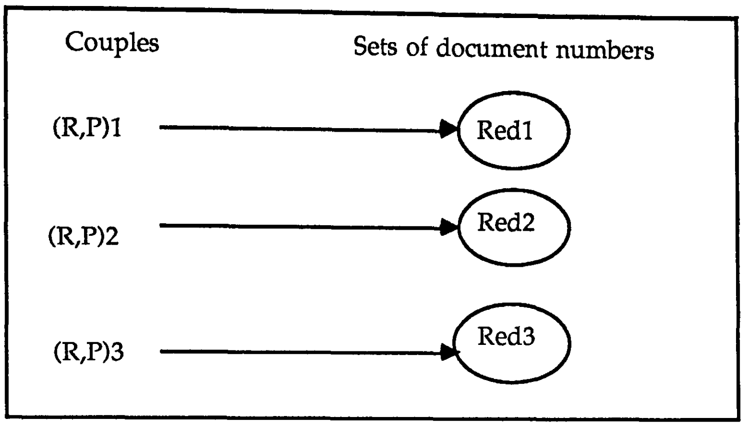
Case A: These conditions are to be fulfilled:
a) $p\neq0$;
b) $r+n=p$;
c) $set1\neq\emptyset$ and $set2\neq\emptyset$;
d) NEW1=INT (Red_i) $(0<i\leq r)$; NEW2=INT (IRed_j) $(0<j\leq n)$
INT(NEW1, NEW2)$\neq\emptyset$;

Case B: If r=0 or n=0, then the conditions become:

a) $p \neq 0$;

b) $r = p$ (or $n = p$);

c) $\mathrm{INT}(\mathrm{Red}_i) \neq \emptyset$ $(0<i \leq r)$ (or $(\mathrm{INT}(\mathrm{IRed}_j \neq \emptyset)$ $(0<j \leq n))$;



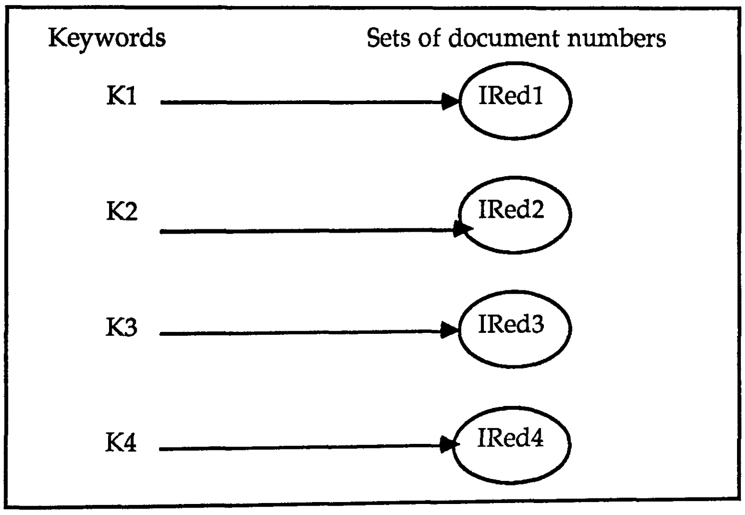


Figure 3-17: The sets Red_i and IRed_j

Keyword search: On the outside of Case A and Case B, a keyword search is made. The keywords and the associated postings are output to the user who can then decide about an eventual search.

3 · 3 · 4 · 6 Output documents

Either a brief record or a full record of documents are output, depending on the user's needs as in section 3·2·4·5.

3 · 3 · 4 · 7 End of search

The user can either stop the search at anytime and leave the system or try another search.

The described approach of searching Arabic documents has been implemented and tested for a set of documents. However, the only facilities that have been made available are: search by title, author, or combination of both and keywords.

3 · 4 FORMS OF THE ARABIC TEXT

The search method of Arabic text defined in section 3·3 has been developed in respect to the Arabic rules: right-left writing of the text, the existence of only the lower case characters...etc. However, the method can be tested either with pure Arabic characters or with romanized characters.

3 · 4 · 1 Pure Arabic characters

With the availability of such Arabic tools like software manipulation of words, text, files... etc, and keyboard (or similar), the search method of section 3·3 is applicable and the results can be presented in Arabic language. However, the program in this study associated to the search is not in Arabic. This is due to the unavailability of the above named tools in a form which could be used on the data for our program.

3 · 4 · 2 Arabic romanisation

The romanisation of the Arabic language is applied in many libraries in the countries where Latin based languages are used. Such romanisation depends on the language (English, French, Italian, etc). Cataloging service [CATAL 70] gives the essential of the Arabic romanisation. This transliteration is used in this thesis.

All the Arabic files and texts are romanised and processed. The word being romanised, the characters of the eventual root are recognisable. The spelling of words is important. The words are close to those obtained after romanising as shown by the same example of section 3·3·1·2 where 'ktb' is the root of the following romanised words 'kitab' (a book), 'kutub' (books), 'al-kitab' (the book), 'kataba' (to write). However, there are more characters in romanised text than in the original.

3 · 4 · 3 Correspondence Arabic scripts/Roman characters

An alternative representation of the Arabic language to test the search method of section 3·3 is the use of the Arabic characters binary representation. The Arab League [ARABL 85] has elaborated a standard binary code for Arabic characters which is approved by ASMO (Arab Organisation for Standardization and Metrology). This representation is used for the internal storage of the Arabic characters.

A correspondence between the binary codes of the Arabic and ASCII characters can be established. Hence, all Arabic files, texts, words can be transcribed. A table of such correspondence is shown in Appendix B. This transliteration is not automatic and takes a long time to perform. The spelling of Arabic words is totally inaccurate and does not bring any help. There is a big difference between an Arabic word and its transcription. However, the use of this correspondence demonstrates that the system can work with the Arab League standard for the representation of Arabic characters

3·5 MODEL OF MULTILANGUAGE SEARCHING

The studied methods of section 3·3 and section 3·4 are analyzed in order to mix or separate the different languages. What solution is best?

3·5·1 Mixed languages (English/French and Arabic)

As defined in the previous studies, the English and French documents are mixed. The search handles texts either written in both languages or simply one of them. The introduction of the Arabic documents make the situation more difficult due to the differences in languages. However, in the case of the romanisation of Arabic documents, it is possible to gather the three languages together. The manipulation is identical for each of them. The different files (URF, DSL...etc) can be either mixed or separated.

3·5·2 Separated languages (English/French and Arabic)

In the case of pure Arabic characters, it is better to separate Arabic documents from Latin documents, as a result of which, the search has to be independent and switched from one language to the other.

3·6 CONCLUSION

Ideas of how to handle a multilanguage search documents have been presented in this chapter. Many files and dictionaries have been introduced and described, to decide a better way for retrieving mixed documents. An approach of searching Arabic text is developed and compared to that defined for English/French text. Finally, the method of separating Arabic documents from English/French documents is adopted. This is described in more detail in chapters 4 & 5.

Chapter 4

MANIPULATION OF THE QUERY SYSTEM

4·1 INTRODUCTION

This chapter describes the way documents are retrieved in our system. All the files maintained in chapter 3 are created and stored in secondary memory. Some imaginary and real documents are tested. The system contains two separate parts. The query system (QSY) used for searching for documents is the subject of this chapter. The updating system (USY) which allows the updating of files and dictionaries is described in chapter 5. QSY is available to the user whereas USY is reserved only for library staff use. The entire system is illustrated in figure 4·1 where the user can only work with the query system and the library staff may use all the system. The QSY helps the user in guiding him throughout the search.

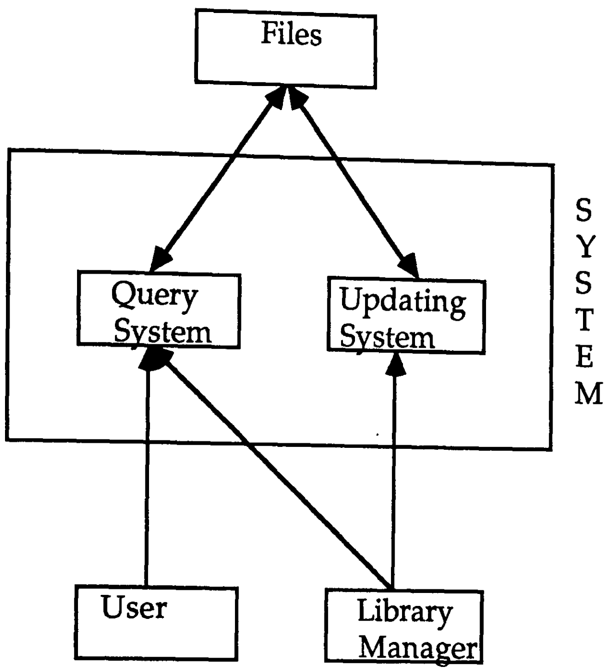


Figure 4·1: Architecture system

4·2 DESCRIPTION OF THE QUERY SYSTEM

The design of a query system is one of the aims of this work. The system should provide the user with the facility to search for the required documents, which are structured logically into files. A set of commands are provided to help him. A menu is displayed at each step to guide the user in his search.

A prototype system has been constructed in order to test the solutions adopted in section 3·2·4 and section 3·3·4. An overview of the query system is outlined. The method to be followed is described and a short example session with the system is held. The query system is a set of queries developed with the intention to help the user in his search. The system has been designed so that the queries are simple and adapted to every user, especially to novice users.

There are 4 levels in the query system depending on the entered text. Level 0 is the root of the query system at which all sessions begin. The conception of levels 1, 2 and 3 is identical in the case of Arabic or English/French documents. It is not necessary to describe the query system separately for each language. These levels are apparent to the user who can follow his search step by step.

Finally, the descriptions of the four levels show also how to manipulate a part of the proposed system. Figure 4·2 illustrates the four levels of the query system.

4 · 2 · 1 Level 0

This is the most important level. It is in this level that the user chooses the language of his search. The nature of the language is decided by keying either 'a' for Arabic documents, 'e' for English/French documents, or 'q' to end the session and to leave the system.

4 · 2 · 2 Level 1

This level consists of entering the searched text. The appropriate command is 'e'. The existence of the text is checked in order to decide whether there is an exact match or not.

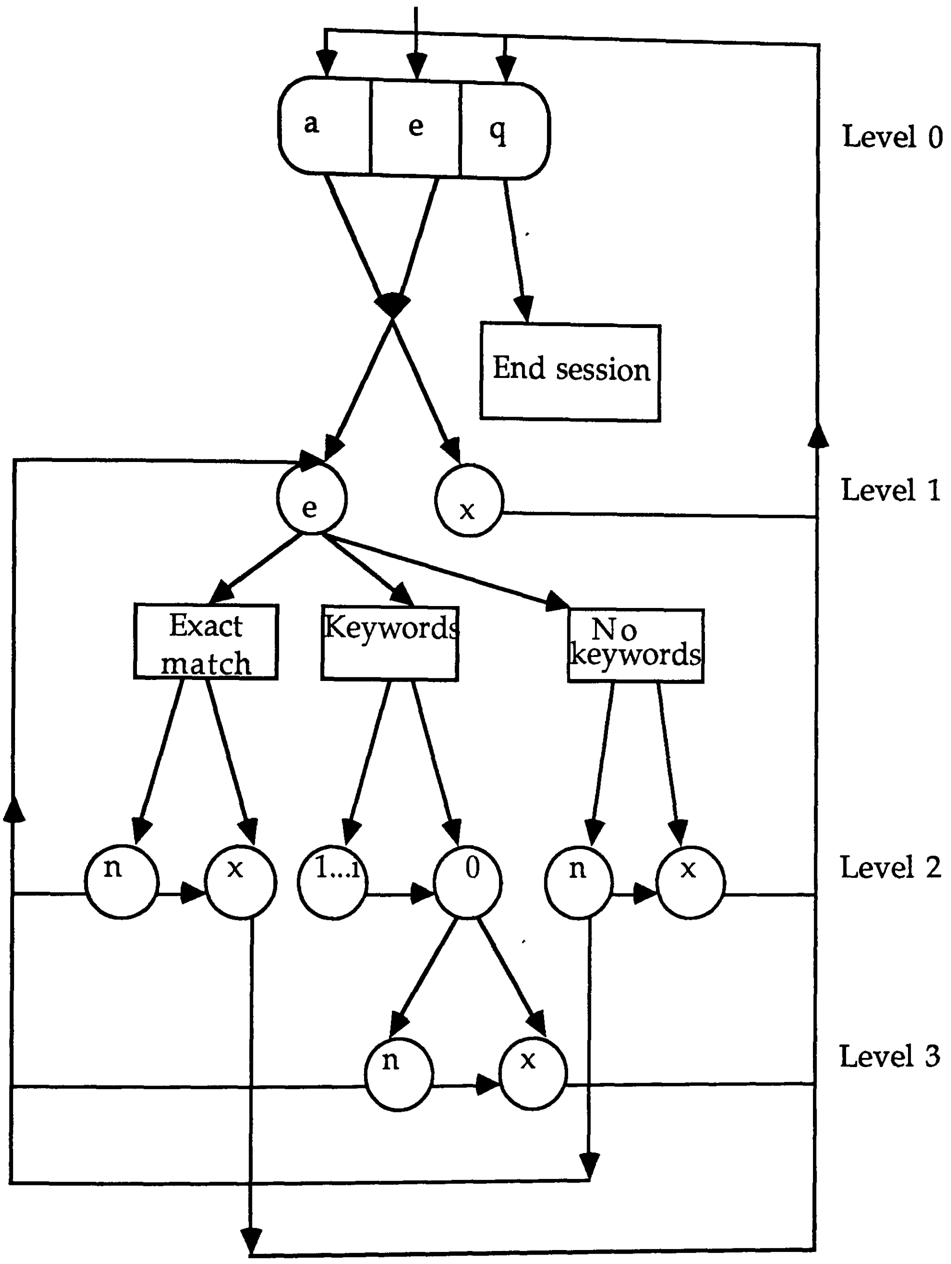


Figure 4·2: The Query system

4 · 2 · 3 Level 2

In the case of a positive answer from level 1, either a brief record or a full record is output to the user. Thereafter, the user can either renew a search using the command 'n', or move back to the main menu using the command 'x'. However, if no exact or partial match is found, the user continues the search from the found keywords in the text. These keywords numbered from 1 to *i* with their associated posting are output to the user who has the choice to get more information. When there is no keyword, the user can either renew a search or stop the search and return to level 0.

4·2·4 Level 3

When the user leaves the allowed alternative of searching by keywords in level 2, he has the possibility to either renew the search or exit and move back to the main menu.

4·3 SCREEN LAYOUT

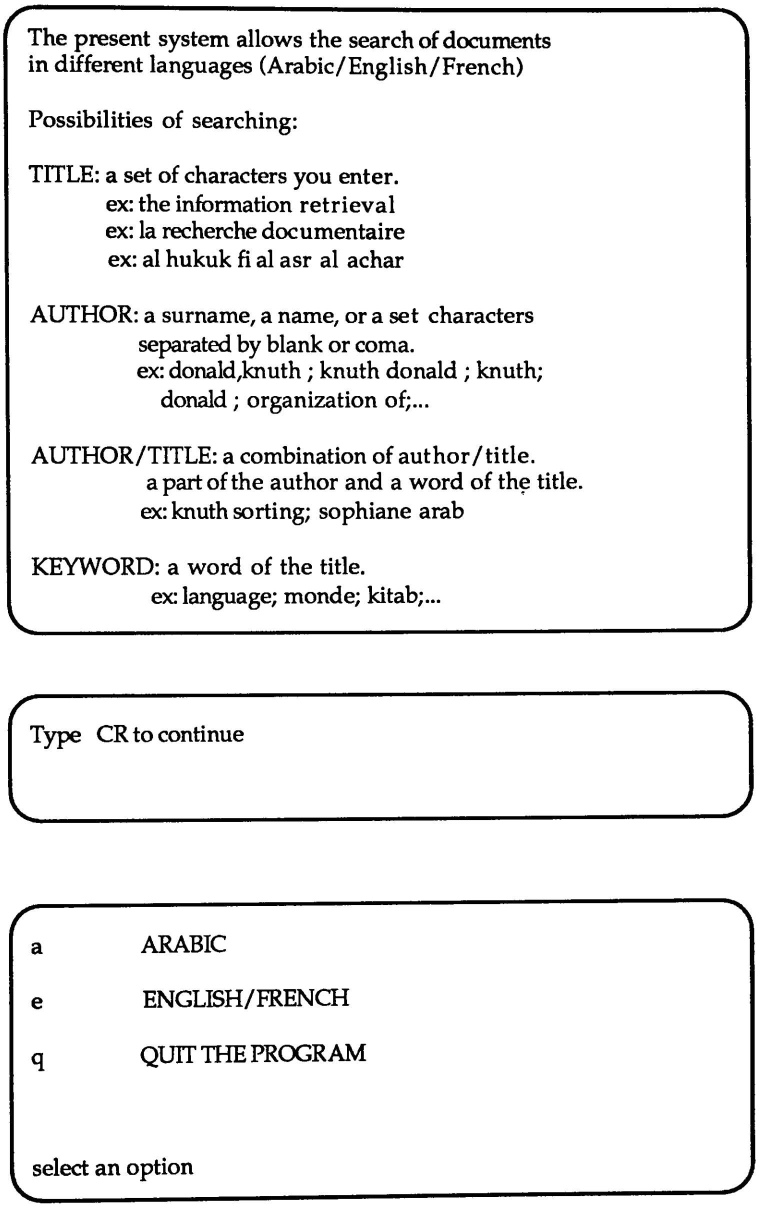
The design of the screen is based on some guidelines of Shneiderman [SHNEI 87]. It is presented as simply as possible with a clear menu as described. The screen contains the menu in a first page of information about the allowed possibilities of searching; such information is written in upper case and lower case letters to be easily memorised by a novice user, for example TITLE to indicate a search by title, a brief explanation of the title with examples is given. However, some simple commands marked by lower case characters are given to the user. A command is designated by only a character followed by a short phrase which shows to the user its action when it is keyed. For example, after the command 'a', is written 'ARABIC'. This indicates that when the user keys this character, he is interested in Arabic documents.

Any command is displayed on the left of the screen and its symbol is in relation with its action; for example at the start when the user has to choose the language, he keys 'e' for English/French documents (the relation is between the character 'e' and the beginning of the string English/French), the same symbol is used again after the choice of the language for enter title/author/keyword(s). The rest of the commands are 'q' for QUIT THE PROGRAM, 'x' for exit, 'n' for new search. Digits are also used in the case of search by keywords; their utilisation depends on the keywords.

The same commands are used for all the languages. However, when Arabic is manipulated in its original script, the commands (also single letters) are displayed on the right of the screen with a short explanation. Finally, the commands are drawn in a simple manner to answer to every user, the maximum of information is given at each step, this is on a first step to test the opportunities of searching documents in several languages. A subsequent step consisting of the amelioration of the actual screen design by introducing the original scripts of Arabic and by giving more possibilities to the user may be considered in future work.

4·4 MANIPULATION OF THE QUERY SYSTEM

The different levels of section 4·3·1 are taken into account to describe the user interface. Some news about the system is output to the user, showing him the searching possibilities allowed. Thereafter, a menu is displayed and after this moment, the user has only to follow and utilise the commands to reach the required information. Throughout the examples, figures are drawn to illustrate the main steps of retrieval documents. The user reads the menu and keys CR. Thereafter, he has the choice to enter 'a', 'e', or 'q'. Figure 4·3 shows this phase. The Key CR is pressed after every command.



```
The present system allows the search of documents
in different languages (Arabic/English/French)

Possibilities of searching:

TITLE: a set of characters you enter.
         ex: the information retrieval
         ex: la recherche documentaire
         ex: al hukuk fi al asr al achar

AUTHOR: a surname, a name, or a set characters
             separated by blank or coma.
             ex: donald,knuth ; knuth donald ; knuth;
                donald ; organization of;...

AUTHOR/TITLE: a combination of author/title.
              a part of the author and a word of the title.
             ex: knuth sorting; sophiane arab

KEYWORD: a word of the title.
              ex: language; monde; kitab;...
```

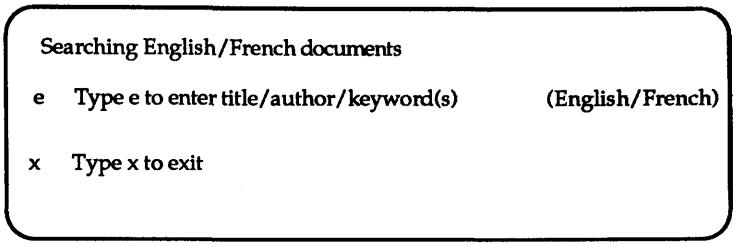
```
Type  CR to continue
```

```
a          ARABIC

e          ENGLISH/FRENCH

q          QUIT THE PROGRAM


select an option
```

Figure 4·3: Introductory menu.

4·4·1 English/French documents

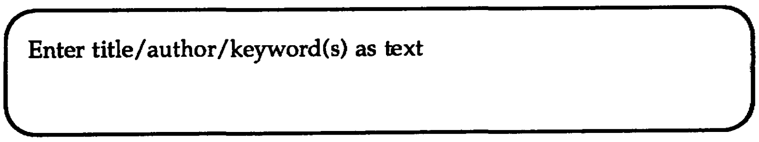
The user keys 'e', figure 4·4 shows how to continue.



```
Searching English/French documents
e   Type e to enter title/author/keyword(s)          (English/French)

x   Type x to exit
```

Figure 4·4: Level 1

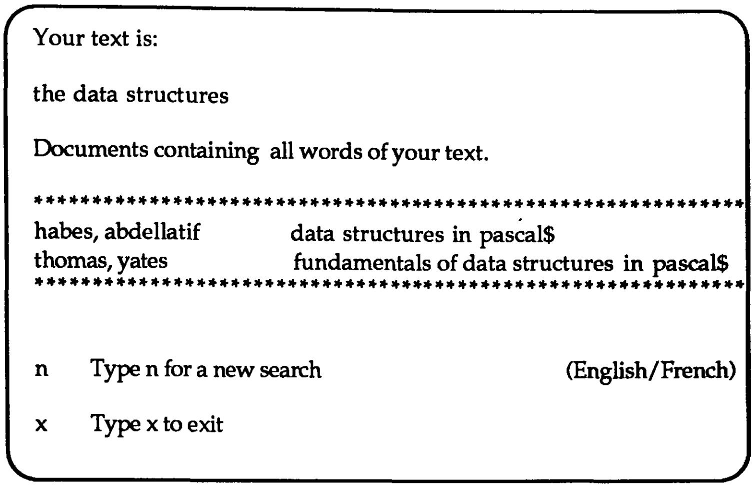
The user now keys 'e', then figure 4·5 exemplifying the next step.



```
Enter title/author/keyword(s) as text
```

Figure 4·5: Level 1 (next)

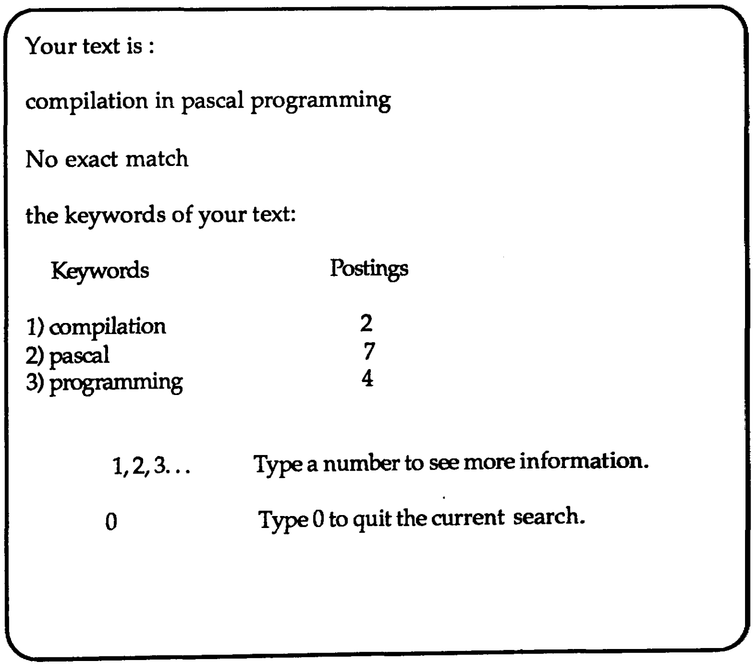
The title: 'the data structures' is entered, figure 4·6 illustrates the results.



```
Your text is:

the data structures

Documents containing  all words of your text.

****************************************************************
habes, abdellatif          data structures in pascal$
thomas, yates              fundamentals of data structures in pascal$
****************************************************************


n     Type n for a new search                        (English/French)

x     Type x to exit
```

Figure 4·6: Results of the query, level 1 and level 2

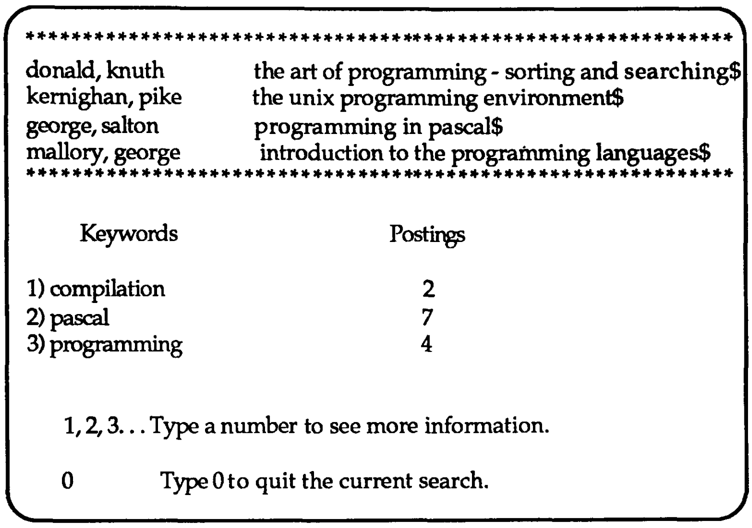
At this stage, the user has the choice to try another search or to exit. To appreciate the others cases of a search, 'n' is keyed. Figure 4·5 is displayed and the title: 'compilation in pascal programming' is entered. Figure 4·7 shows the results.



```
Your text is :

compilation in pascal programming

No exact match

the keywords of your text:

  Keywords                 Postings

1) compilation                2
2) pascal                     7
3) programming                4


        1,2,3...     Type a number to see more information.

        0            Type 0 to quit the current search.
```

Figure 4·7: Results of the query, level 1 and level 2

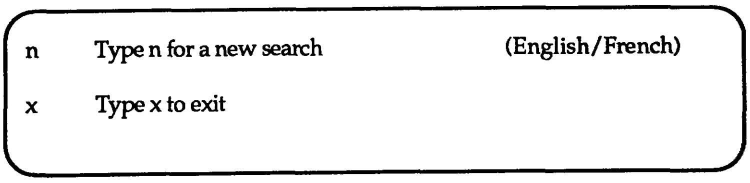
The keywords found in the text are output to the user. Each keyword with its posting is numbered. To illustrate this case, digit '3' is keyed. The results are in figure 4·8.



```
************************************************************
donald, knuth        the art of programming - sorting and searching$
kernighan, pike      the unix programming environment$
george, salton       programming in pascal$
mallory, george       introduction to the programming languages$
************************************************************

     Keywords                      Postings

1) compilation                        2
2) pascal                             7
3) programming                        4


   1, 2, 3... Type a number to see more information.

   0          Type 0 to quit the current search.
```

Figure 4·8: Results of the query, level 2

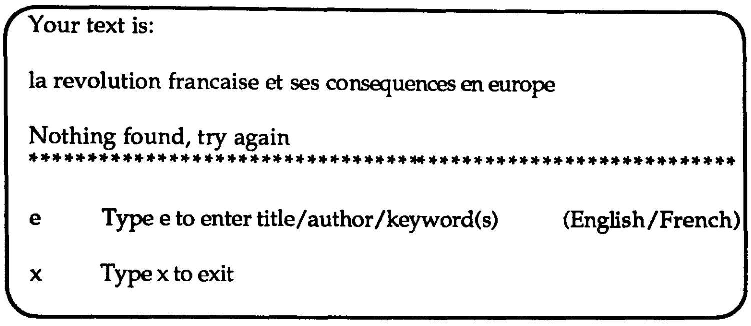
Figure 4·8 outlines all the titles which contain the keyword 'programming'. Thereafter, all the entries are output again, the user can search by another keyword. When the user keys '0', he leaves the current search and figure 4·9 is output.



```
n      Type n for a new search                (English/French)

x      Type x to exit
```

Figure 4·9: Level 3

Let us now see the case where there is no indexed keyword in the entered text. Suppose the French title: 'la revolution francaise et ses consequences en europe' (the French revolution and its consequences in Europe) is keyed, figure 4·10 presents the results.



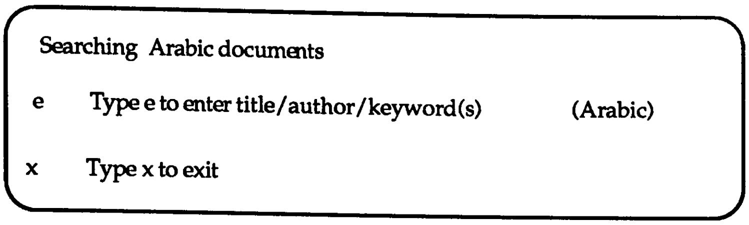
```
Your text is:

la revolution francaise et ses consequences en europe

Nothing found, try again
****************************************************************

e       Type e to enter title/author/keyword(s)       (English/French)

x       Type x to exit
```

Figure 4·10: Results of the query

If the user presses 'x', then he goes to the main menu of figure 4·3 (only the first two parts of the figure are displayed).

4 · 4 · 2 Arabic documents

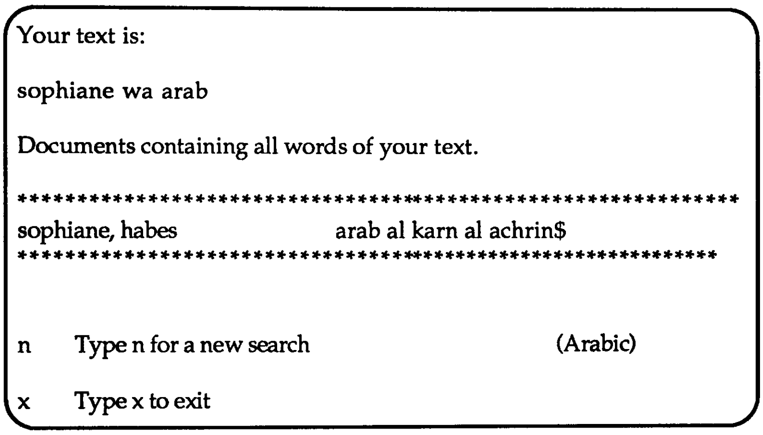
The mechanisms of the search outlined for English/French documents are now repeated for Arabic documents. Examples of searching are illustrated by figures. The user keys 'a' to search the Arabic documents. Figure 4·11 shows what he has to do.



```
Searching  Arabic documents

e      Type e to enter title/author/keyword(s)           (Arabic)

x      Type x to exit
```

Figure 4·11: Starting

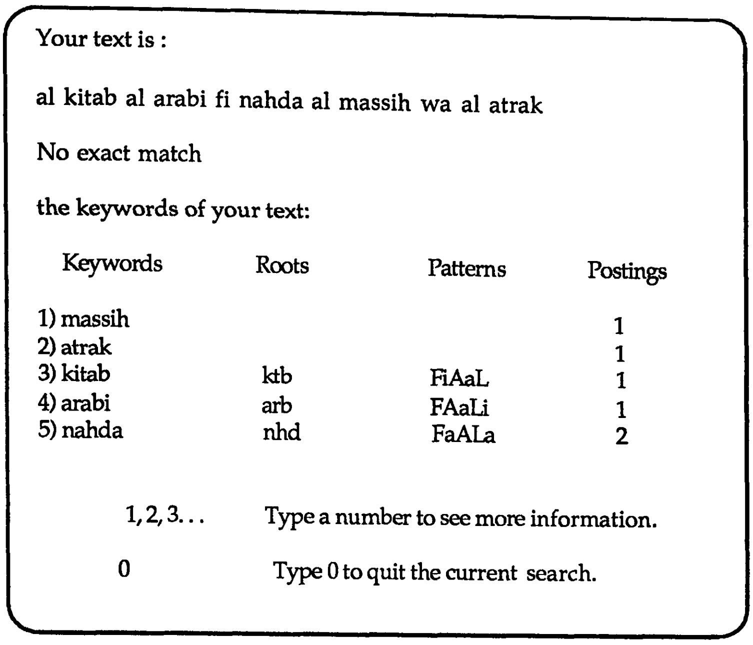
The following example concerns a combination author/title. The entered text is: 'sophiane wa arab'. Results are output in figure 4·12.



```
Your text is:

sophiane wa arab

Documents containing all words of your text.

***********************************************************
sophiane, habes                    arab al karn al achrin$
***********************************************************


n     Type n for a new search                         (Arabic)

x     Type x to exit
```

Figure 4·12: Results of the query

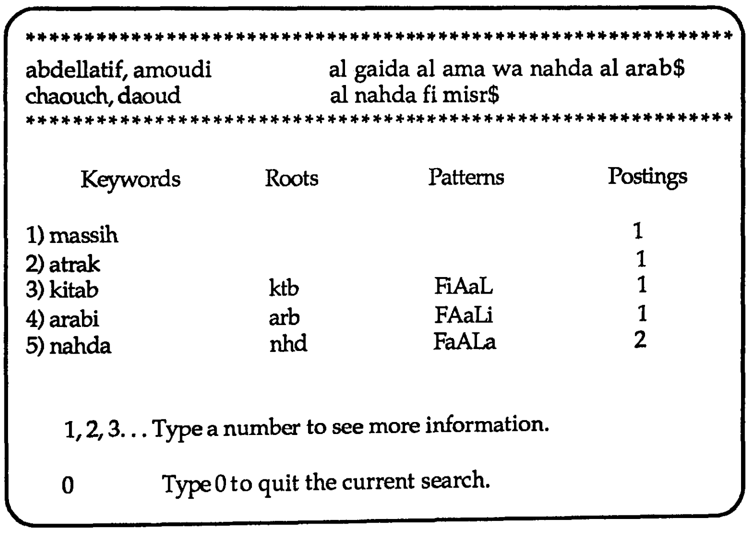
The last example describes the case where there is no exact match. Let 'al kitab al arabi fi nahda al massih wa al atrak' (the Arabic book in period of the Christianism renascence and Turkish civilisation) be the new title. Figure 4·13 illustrates the results of the query.



```
Your text is :

al kitab al arabi fi nahda al massih wa al atrak

No exact match

the keywords of your text:

   Keywords        Roots          Patterns       Postings

1) massih                                           1
2) atrak                                            1
3) kitab           ktb            FiAaL             1
4) arabi           arb            FAaLi             1
5) nahda           nhd            FaALa             2


        1,2,3...    Type a number to see more information.

        0           Type 0 to quit the current search.
```

Figure 4·13: Results of the query

Suppose digit '5' is keyed, figure 4·14 illustrates the results.



```
*****************************************************************
abdellatif, amoudi           al gaida al ama wa nahda al arab$
chaouch, daoud               al nahda fi misr$
*****************************************************************

     Keywords        Roots          Patterns       Postings

1) massih                                             1
2) atrak                                              1
3) kitab             ktb            FiAaL             1
4) arabi             arb            FAaLi             1
5) nahda             nhd            FaALa             2


   1,2,3... Type a number to see more information.

   0        Type 0 to quit the current search.
```

Figure 4·14: Results of the query

There are two new columns roots and patterns (which do not exist for English/French documents). They give the root-pattern pair associated with each reducible word.

Finally, to leave the system, the user keys 'q'.

4·5 CONCLUSION

The query system has been tested with some document records and the results are quite satisfactory. However, the prototype query system has to be rerun each time it is used by a new user. This is a disadvantage which could be avoided in the future by optimising the implementation of the different routines.

The time taken between prompt and response from the various commands is short and acceptable. The system allows the user to re-enter a command in case it is badly keyed.

There is no need for a lexical analysis of the texts. They are a set of words with concordable characters (the letters a-z, the digits 0-9) or optionally concordable characters (for example the hyphen) as defined by F.N. Teskey [TESKE 82]. When a bad text (text which contains extra characters such as /<>*%£,...etc) is entered, the user is invited to renew the search.

Chapter 5

SYSTEM DESIGN

5·1 INTRODUCTION

This chapter describes the different structures adopted for the files and dictionaries outlined in chapter 3. The method of inverted files with a boolean search is adopted for the retrieval of documents. All the written programs are performed around this structure. However, some files and dictionaries have sequential structures. The structure of linked lists is introduced when necessarry to reduce the memory space. The updating system (USY) introduced in section 4·1 of chapter 4 is described too.

For more detail on file structures, [RIJSB 79] and [LYNCH 74] gives a summary of this on describing many file structures relative to information retrieval. Horrowitz & Sahni [HOROW 86] provide a lot of information on data structures. The previous chapter concludes on differentiating between English/French documents and Arabic documents. The files associated with the two classes of language are then separated.

5·2 DATA STRUCTURES

Files and dictionaries are now described with the notion of data structures. Descriptions have to handle English/French and Arabic documents. Detailed descriptions are given for the English/French language. The one for Arabic is somewhat different; similarities with English/French are only pinpointed.

5·2·1 English/French documents

5·2·1·1 URF file

It is a sequential file organised on lexicographic order on the value of the document number field. The fixed fields 'publisher', 'code', 'location', 'type', 'language', and 'status' of the document are packed arrays of characters. Figure 5·1 shows such structure. The field 'status' is updated each time the document is borrowed by a user. New documents may be added every week depending on the organisation of the staff manager.

5·2·1·2 DSL file

It is a sequential file. The noise words are packed arrays of characters. This is to facilitate their elimination from entered texts at the start of the search. The file is not updated because the number of such noise words is fixed.

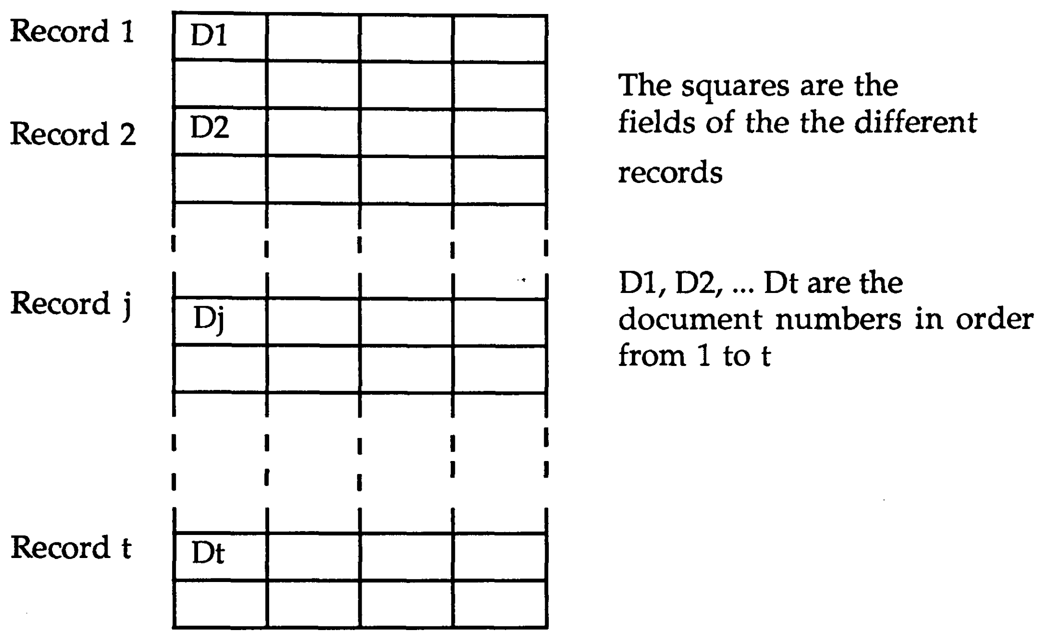

Record 1
Record 2
Record j
Record t
D1
D2
Dj
Dt
The squares are the fields of the the different records
D1, D2, ... Dt are the document numbers in order from 1 to t

Figure 5·1: The structure of URF file

5·2·1·3 IXF and IVF files

They are constructed from fields title and author. The IXF file, which is a sequential file, contains the keywords (entries) in the form of packed arrays of characters, followed by their postings. The IVF is a set of linked lists of document numbers associated with each entry. Figure 5·2 shows the relationship between these two files. The two files are updated every time the library receives new documents. Indeed, either new keywords and postings are to be stored in the IXF file and their relationship with the documents has to be created in the IVF file, or postings of some keywords which already exist are updated in the IXF file and some linked lists have to be updated. The main disadvantage of such structures is that when there are new documents, these two files are practically recreated from scratch.

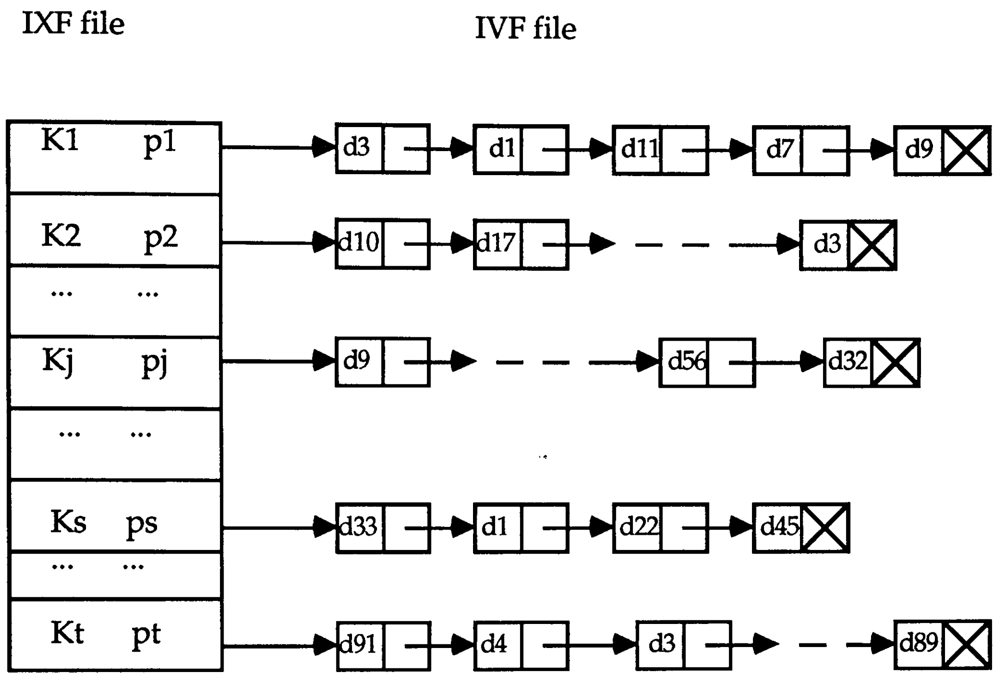


Figure 5·2: IXF and IVF file structures

K1, K2, ..., Kp are the keywords (entries) formed from the documents of the URF file. p1, p2, ..., are in this case the associated postings related with each of the keywords and d1, d2, ..., are the document numbers.

The previously outlined files TPF and TLA are not used in the proposed solution. Their full description is not necessary here. However, the TPF file is sequential whereas the TLA file is a set of linked lists.

5·2·2 Arabic documents

The number of files is greater in the case of this language than in the case of the previous languages. However, descriptions of the files URF, TPF, TLA are identical to those presented in section 5·2·1. The updating of the URF file is the same as that decided for the English/French documents.

5·2·2·1 ROOT, PATTERN and TWO - ROOT tables

Table ROOT which contains all Arabic roots is a sequential file. Each root is a packed array of characters, that facilitates further checking of roots and increases the speed of the search. Table PATTERN which contains all Arabic patterns is also a sequential file; their elements are packed arrays of characters. Table TWO-ROOT which contains all the first two characters of the Arabic roots is a sequential file. Figure 5·3 illustrates such files. The three files are not updated; they are invariant data.

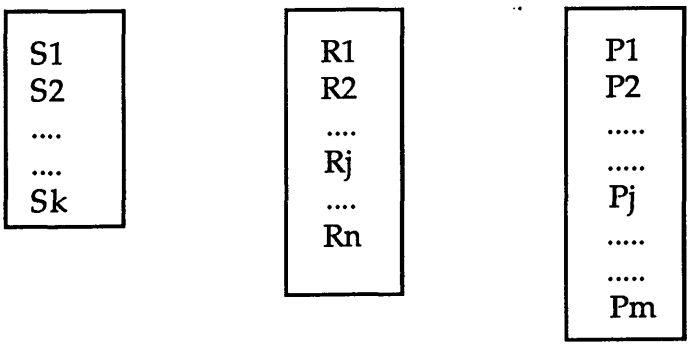


Figure 5·3: Tables TWO-ROOT, ROOT and PATTERN

S1, S2, ..., Sk are the elements of table TWO-ROOT, R1, R2,..., Rn are the roots and P1, P2, ..., Pm are the patterns. The different tables ROOT1, ROOT2, ROOT3 discussed, in chapter 3 but not used in our solution have the structure of table ROOT.

5·2·2·2 ASL dictionary

ASL (Arabic stop list dictionary) which contains the Arabic noise words and the linked noise word is a sequential file. It is a static file which does not need to be updated because the number of elements is constant.

5·2·2·3 IXIR and IVIR files

The words which are irreducible are keywords. Hence, the structure of the IXIR (Index file of irreducible words) file is identical to the structure of the IXF file defined in section 5·2·1. The IVIR (Inverted file of irreducible words) file has the structure of the IVF file discussed in section 5·2·1. The updating of the two files is similar to that discussed in section 5·2·1. However, it depends also on the updating of the files IXR and IXP which is outlined in this section.

5·2·2·4 IXR and IXP files

The last files to be described are the IXR (Index file root) file and the IXP (Index file pattern) file. The IXR file, which contains all the triliteral roots (entries), is a sequential file. The IXP file is a set of linked lists, where each element of the list is in turn a linked list. The reason for having such a structure is that all reducible words which have the same root are gathered around this root; all documents which contain the same root-pattern pair (keyword) are grouped round this pair. Figure 5·4 shows this structure. R1, R2, ..., Rn are the roots; p1, p2, ..., pj are the patterns associated with each root. Ri and d1, d2, ..., dl (l is in [1,t]) are the document numbers related to the root-pattern pair (Ri,pk) (i is in [1,n] and k is in [1,m]).

While there is acquisition of new documents, the two files have to be updated because of the creation of reducible and irreducible words. Reducible words are decomposed into root-pattern pairs. Roots which are new are added to the IXR file and new linked lists are created in the IXP file. Roots which already exist have their associated linked lists updated in the IXP file.

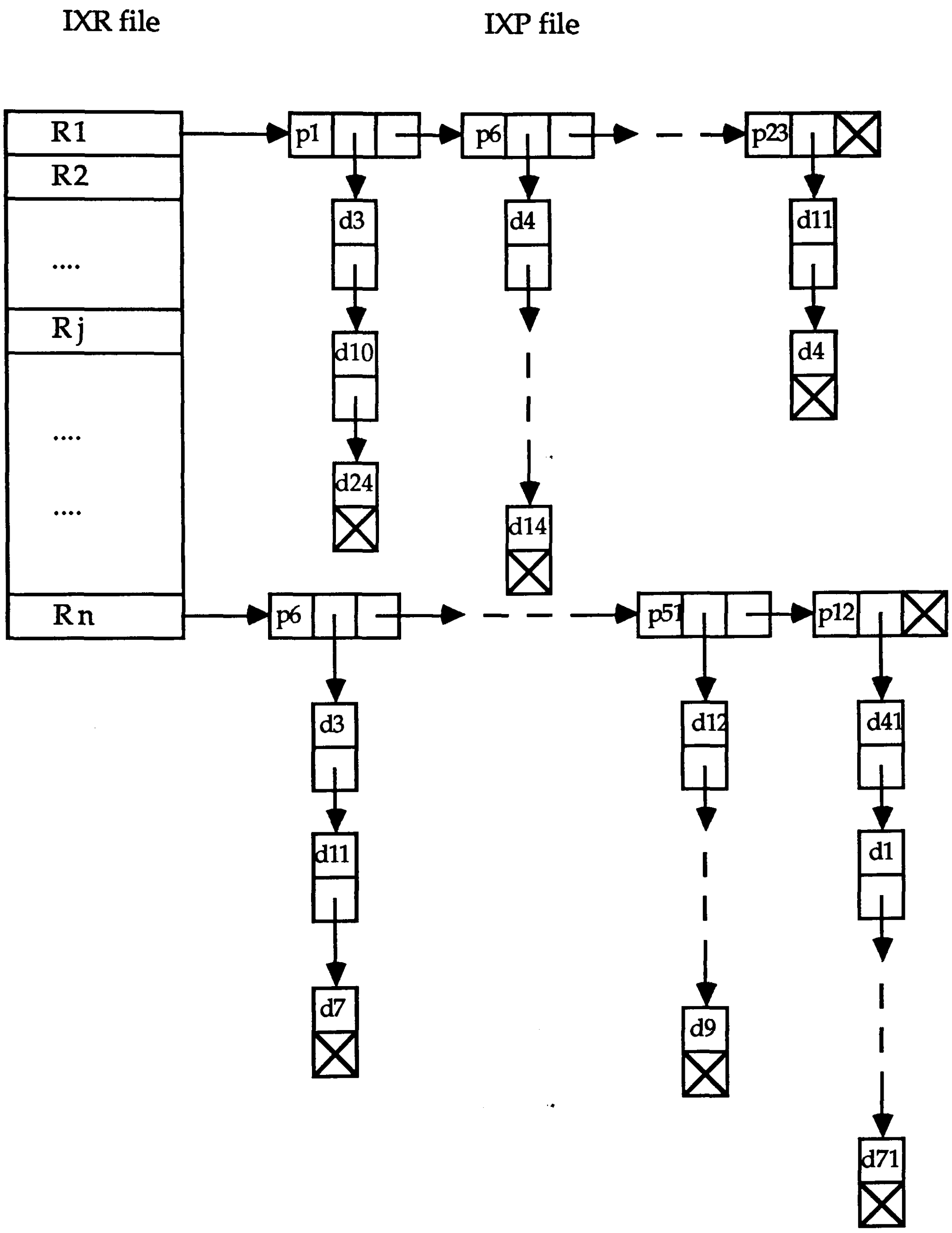


Figure 5·4: Structure of the IXR and IXP files and their relations

5·3 UPDATING SYSTEM

Its main role is to give the user more detailed information by computerising new documents. They are added to those which already exist. The most important files, such as the URF's file, are updated when necessary.

The updating system assumes the different files and dictionaries have already been created, and its task is to update them when necessary. As discussed previously, this part is reserved to the library staff. All the required data is prepared in advance and added to the main files. However, there is one disadvantage in updating the 'status' and 'borrower' fields of documents required by users. This operation is performed every time a transaction is held between a user and the library. The operation is time consuming due to the sequential organisation of the URF's file and when the number of documents is high.

Figure 5·5 shows the main files to be updated. The separation between the languages is still apparent. The fields 'status' and 'borrower' fields of the URF's file are updated every day. The IXF, IVF, IXR, IXP, IXIR and IVIR files are reconstructed every week when new documents are added to the URF's file. However, some other parameters are updated such as the number of documents, the number of roots, patterns, the size of file TWO-ROOT.

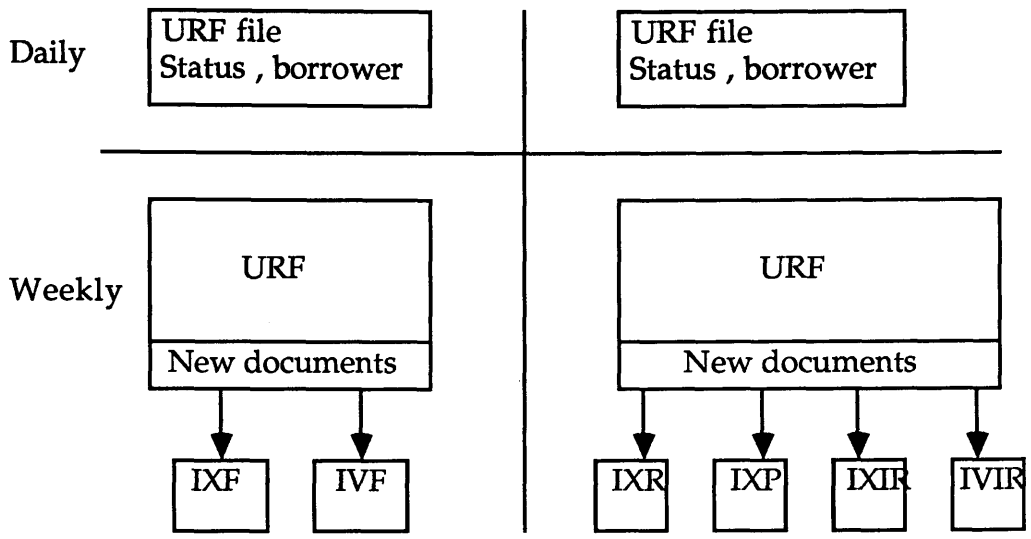


Figure 5·5: Updating of the main files

5·4 CONCLUSION

Most of the files are sequential due to the conception of the inverted file method. The implementation is coded in Pascal. Thereafter, some documents are stored and manipulated for tests with success. The structure of the program is detailed in Appendix A and a listing of the code is available.

Chapter 6

SYSTEM EVALUATION

6·1 INTRODUCTION

This section outlines the time for execution when the query system is called for a given text. Evaluations are discussed depending on the language of the documents. The time of execution of the searched texts is computed for the different languages. Because most of the files are ordered, a binary search method is used to check the existence of an element in a given file. For a table of size *l*, the number of comparisons is $Log_2 l$ [KNUTH 73].

6 · 2 EFFICIENCY ANALYSIS

The number of operations due to the search documents is done separately for each language.

6 · 2 · 1 Case of retrieval English/French documents

Let

d be the number of documents.

s the number of noise words.

k the number of indexed keywords.

w the number of words of the entered text.

Then the search for a document has three steps:

(1) To eliminate the noise words, a binary search through the file of noise words with each of the w words in the entered text.

number of operations = $w\log_2 s$

(2) To determine the documents of the entered keywords, a binary search through the file of keywords for each of the w' keywords in the entered text ($0 \leq w' \leq w$).

number of operations = $w'\log_2 k$

(3) To retrieve the relevant documents, a sequential search in the file of documents for each possibility.

number of operations per search = $(d+1)/2$

The first two steps are performed for each entered text. The total number of these operations will be denoted by T so that

$$T = w\log_2 s + w'\log_2 k \qquad (1)$$

If there is an exact match, then let v be the number of documents which contain all the w' keywords ($1 \le v < d$). Hence, the total number of operations is

$$E = wLog_2 s + w'Log_2 k + v(d+1)/2 \qquad (2)$$

If there is no exact match, then let there be references to g documents by w' keywords. The total number of operations is therefore

$$K = wLog_2 s + w'Log_2 k + g(d+1)/2 \qquad (3)$$

Finally, if the entered text contains only noise words and/or w" words which are not indexed keywords ($0 \le w'' \le w$), then, the total number of operations is

$$N = w\log_2 s + w''\log_2 k \qquad (4)$$

The numbers E, K and N are an average of the execution number of operations for an entered text. Note that these formulae give the number of operations for retrieval only, they do not include the operations for the creation of the tampon files or transfer from file to file. Table 6·1 gives values to T, E, K, N for various file sizes based on assumptions on the size of entered text and the number of keywords it contains. Information about tested texts are given by vectors (w, w', w") of columns 4, 5 and 6. The computations necessary to check the existence of documents (T) is not large as it is shown by column 9 of table 6·1, it is between 30 and 150 for a number of documents between 40 and 40000 and a number of keywords between 100 and 80000. However, the number of operations for retrieving relevant documents is very short for small number of documents and high for a large number as shown by columns 10 and 11 of table 6·1. This is due to the sequential access of the URF file.

1	2	3	4	5	6	7	8	9	10	11	12
d	s	k	w	w'	w"	v	g	T	E	K	N
40	100	147	3	2	0	2	0	34	75		
40	100	147	7	4	0	1	0	75	96		
40	100	147	7	4	0	5	0	75	180		
40	100	147	7	4	0	0	7	75		220	
40	100	147	7	0	4	0	0	47			75
40	100	147	5	0	0	0	0	33			33
40	100	147	7	3	0	7	0	68	210		
400	300	1300	7	4	0	4	0	99	900		
400	300	1300	4	2	0	15	0	54	3100		
400	300	1300	7	0	4	0	0	58			99
400	300	1300	5	3	0	5	0	72	1100		
4000	500	10000	7	4	0	10	0	120	20000		
4000	500	10000	7	0	4	0	0	63			120
4000	500	10000	7	4	0	2	0	120	4100		
4000	500	10000	5	2	0	50	0	71	100000		
4000	500	10000	7	3	0	0	100	100		20000	
4000	500	10000	6	0	4	0	0	54			110
40000	1000	80000	7	4	0	25	0	130	500000		
40000	1000	80000	7	0	5	0	0	70			150
40000	1000	80000	7	3	0	40	0	120	800000		
40000	1000	80000	7	4	0	100	0	130	2000000		

Table 6·1: Examples of number operations

6 • 2 • 2 Case of retrieval Arabic documents

The number of operations due to the cases and organisations presented respectively in section 3·3·2·2 and section 3·3·2·3 of chapter 3 is first computed in this section before analysing the one of search documents. The combinations to be tested are respectively *$l(l-1)(l-2)/6$,* *l-1, 5l(l-1)(l+1)* for the different cases (*l* is the length of a word). However, in the last case, all combinations which do not contain a vowel are discarded. The number of operations needed to reduce words into a root-pattern pair is detailed for each organisation.

6·2·2·1 First organisation

Let

n and m be the sizes of tables ROOT and PATTERN

c the number of combinations of three characters of a word depending on each case of roots

k the number of combinations which are in table ROOT $0 \leq k < c$

Then the reduction of a word has two steps.

(1) To eliminate the bad combinations, a binary search through the file of roots with each of c combinations is necessary. The number of operations is $c\log_2 n$.

(2) To determine the patterns associated with the qualified roots, a binary search through the file of patterns with each of k roots is held. The number of operations is $k\log_2 m$.

These two steps are performed for any Arabic word different from a noise word. Therefore, the number of operations is:

$$f = cLog_2 n + kLog_2 m \qquad (5)$$

From case to case, the number of comparisons is accumulated. The principle is that when combinations do not belong to the first case, they are checked in the second case and so on until the last case. It is not obvious to decide easily the adherence of any combination. However, when a root-pattern pair is determinated with one case (case1, case2 or case3), it is not necessary to check with the other(s). These remarks are valid for all the three organisations. Thereafter, h being given as $0 \leq h < c$, the number of operations for each case is :

$$f1 = cLog_2 n + hLog_2 m \quad (6)$$

$$f2 = f1 + cLog_2 n + hLog_2 m \quad (7)$$

$$f3 = f2 + cLog_2 n + hLog_2 m \quad (8)$$

6·2·2·2 Second organisation

In this organisation, a new table, namely TWO-ROOT, of size t is introduced and taken into account with tables ROOT and PATTERN. It has the role of a filter. A combination of three characters is a root if and only if its first two characters are in file TWO-ROOT. Thereafter, a binary search through this new file for each c combinations is held. The number of operations is $clog_2 t$.

Then formula (5) becomes

$$g = cLog_2 t + kLog_2 n + hLog_2 m \quad (9)$$

Therefore, for each case (case1, case2, or case3) the values of k and h are $0 \leq k < c$, $0 \leq h < k$ and the number of operations are:

$$g1 = cLog_2 t + kLog_2 n + hLog_2 m \qquad (10)$$

$$g2 = g1 + cLog_2 t + kLog_2 n + hLog_2 m \qquad (11)$$

$$g3 = g2 + cLog_2 t + kLog_2 n + hLog_2 m \qquad (12)$$

6 • 2 • 2 • 3 Third organisation

Each case (case1, case2 or case3) has its own table of roots named ROOT1, ROOT2, and ROOT3 with respectively a size allocation of n1, n2 and n3. The formula (9) is used with the same definition of k and h. However, in case1: n=n1, in case2: n=n2 and in case3: n=n3 and the number of operations for each case are in that order:

$$h1 = cLog_2 t + kLog_2 n1 + hLog_2 m \qquad (13)$$

$$h2 = h1 + cLog_2 t + kLog_2 n2 + hLog_2 m \qquad (14)$$

$$h3 = h2 + cLog_2 t + kLog_2 n3 + hLog_2 m \qquad (15)$$

6 • 2 • 2 • 4 Evaluation

The number of operations in the case of the retrieval of Arabic documents is more complex than the one computed for English/French documents. Indeed, this is due to the extra tasks of decomposing the Arabic words into root-pattern pairs. To complete the computation of the search time, let

w be the number of words in the entered text.

s the number of noise words.

a the number of indexed roots.

p the number of indexed patterns.

k the number of indexed irreducible words.

d the number of documents.

h the number of words of the keyed text devoid of the noise words.

Then the search for a document has five steps:

(1) To eliminate the noise words, a binary search through the file of noise words for each of the w words in the entered text. The number of operations is $wlog_2 s$.

(2) Reduction of the words of the entered text into root-pattern pairs. The formulae (10),(12) of the second organisation are used. Note that the values of formulae (10) and (11) are slightly similar.

number of operations = $\Sigma g1 + \Sigma g3$

(3) Check the existence of the constructed root-pattern pairs in the files of indexed roots and indexed patterns. A binary search is used.

number of operations = $r\log_2(a*p)$ $(0 \le r \le h)$

(4) Check the existence of the irreducible words of the entered text using a binary search.

number of operations = $i\mathrm{Log}_2 k$ $(0 \le i \le k)$

(5) To retrieve the relevant documents, a sequential search in the file of documents for each possibility.

number of operations per search = $(d+1)/2$

The first four steps are performed for each entered text. The total number of these operations will be denoted by T so that

$$T = wLog_2 s + \Sigma g1 + \Sigma g3 + rLog_2(a^*p) + iLog_2 k \qquad (16)$$

The same discussion is held such as presented in section 6·2·1 for English/French documents. Thereafter, if there is an exact match, then let v the number of documents which contain all the h words ($1 \leq v < d$). Hence, the number of operations is

$$E = wLog_2 s + \Sigma g1 + \Sigma g3 + rLog_2(a^*p) + iLog_2 k + v(d+1)/2 \qquad (17)$$

If there is not exact match, then let there be references to g documents by h keywords. The total number of operations is therefore

$$K = wLog_2 s + \Sigma g1 + \Sigma g3 + rLog_2(a^*p) + iLog_2 k + g(d+1)/2 \qquad (18)$$

Finally, when there is neither indexed roots (r', $0 \leq r' \leq h$) nor indexed irreducible words (i', $0 \leq i' \leq h$), or the keyed text contains only noise words, then, the number of operations is

$$N = wLog_2 s + \Sigma g1 + \Sigma g3 + r'Log_2(a^*p) + i'Log_2 k \qquad (19)$$

The numbers E, K and N are an average of the execution number of operations for an entered text.

6·2·2·5 Discussion and examples

A number of operations have been computed for some entered texts. The second organisation is chosen. Values of parameters used in this section are given and detailed formulae are applied to determine an average of the search time in the number of operations for Arabic documents. The basic information of the tested texts are: total number of words, noise words, reducible words and irreducible words. With this representation, the vectors (7, 3, 2, 2), (6, 3, 1, 2), (8, 4, 4, 0) represent the entered titles.

Justification of the choice of the second organisation: Values are given to the different parameters of formulae (6) and (8) in case of the first organisation (the values of formulae (6) and (7) are slightly similar), to parameters of formulae (10) and (12) in case of the second organisation and to parameters of formulae (13) and (15) in case of the third organisation (the values of formulae (13) and (14) are slightly similar). The formulae compute the number of operations required to reduce the words of the above titles, the reducible words are supposed to belong either to case 1 or to case 3. The values of sizes of the different are as follows:

t=400, n=7600, m=100, n1=5100, n2=500 and n3=2000.

Table 6·2 contains the different values of the the parameters l, k and h used in the formulae of the three organisations.

word	l	k	h
1	5	2	2
2	4	1	1
3	7	10	5
4	6	5	1

word	l	k	h
1	9	30	10
2	6	8	4
3	8	14	10

word	l	k	h
1	5	5	2
2	10	35	15
3	7	10	5
4	11	40	20

Table 6·2: values of parameters of l, k and h

Table 6·3, table 6·4 and table 6·5 illustrated the computation of the number of operations associated to each organisation.

word	f1	f3
1	142	-
2	-	3874
3	-	21691
4	264	-
total	25971	

word	f1	f3
1	-	46476
2	284	-
3	-	32553
total	79313	

word	f1	f3
1	271	-
2	1646	-
3	484	-
4	-	85218
total	87619	

Table 6·3: Number of operations due to reduction of words - First organisation

word	g1	g3
1	125	-
2	-	2612
3	-	14683
4	244	-
total	17664	

word	g1	g3
1	-	31157
2	302	-
3	-	22029
total	53902	

word	g1	g3
1	250	-
2	1588	-
3	464	-
4	-	57698
total	60000	

Table 6·4: Number of operations due to reduction of words - Second organisation

word	h1	h3
1	124	-
2	-	2610
3	-	14664
4	241	-
total	17639	

word	h1	h3
1	-	31513
2	298	-
3	-	22002
total	53813	

word	h1	h3
1	247	-
2	1667	-
3	459	-
4	-	57621
total	59994	

Table 6·5: Number of operations due to reduction of words - Third organisation

The number of operations of the first organisation are too high when compared to those of the other organisations, so that this organisation is not discarded. The differences between the number of operations computed with the second and the third organisation are so small that the two organisation could be adopted. Because of the number of files used in the second organisation is smaller than the ones used in the third organisation, the choice of the second organisation is to be preferred. For instance, in table 6·4, the first word of the first title is reduced into a root pattern pair after 125 comparisons; the third word is found to be irreducible after 14683 comparisons. The total number of comparisons to split the first title into reducible words and irreducible words is 17664. If a word does not belong to one case (case 1, case 2 or case 3), it is irreducible and the number of operations computed to this is equal at least to the value of g3.

Total number of operations due to the relevant documents - Values are given to the vector (d, s, a, p, k) and the parameters v and g to compute the time of execution for searching for the entered titles. Formulae (17) and (18) are used. Table 6·6 gives values of T, E, K for various file sizes and assumptions of the size of entered text and the number of indexed roots, indexed patterns and indexed irreducible words. The same keyed titles are tested for different values of the vector (d, s, a, p). Column 3 of table 6·6 shows that increasing the number of roots does not affect the number of comparisons of column 6 needed to reduce words into root-pattern pairs.

1	2	3	4	5	6	7	8
d	s	a	p	k	T	E	K
13	6	9	12	16	17664	18660	
13	6	9	12	16	53902		54000
13	6	9	12	16	60000	61000	
130	60	90	90	160	17664	18660	
130	60	90	90	160	53902		54900
130	60	90	90	160	60000	61000	
1300	500	1000	100	1600	17664	18660	
1300	500	1000	100	1600	53902		56000
1300	500	1000	100	1600	60000	61000	
40000	750	7000	100	16000	17664	190000	
40000	750	7000	100	16000	53902		130000
40000	750	7000	100	16000	60000	315000	

Table 6·6: Examples of number operations

The number of documents has not a major effect on computation (including the search for relevant documents) when it is around 10000 as shown in columns 7 and 8 compared to column 6. However, the number of operations becomes large for over 10000 documents due to the sequential access of the URF file and also the values of v and g.

6·3 CONCLUSION

The execution time in the number of operations is split into two: run time and search time. The run time is the time due to operations while data is input and computerised with respect to the data structures described in chapter 5. This time is relatively high, especially because of the large number of operations required to reduce Arabic words into root-pattern pairs. The search time is acceptable and reasonable. However, because more files are scanned in Arabic than in English/French, the search time is slightly higher in the case for searching Arabic documents.

Chapter 7

CONCLUSIONS

7·1 RESULTS

A method for retrieving mixed documents has been presented, based on Roman and Arabic scripts only. The best solution was to separate English/French from Arabic documents. Files and dictionaries are created for this purpose. The same methods for searching for documents are adopted for both scripts. A particular case study of Arabic which concerns its representation is added; Arabic text is romanised; a morphological study of Arabic has been performed which gave a real boost to the performance of the sytem. Moreover, the presented solution for handling Arabic documents can also support these documents in their original script.

The searching method developed was tested with some documents, reduced only to output 'authors' and 'titles' in the case of relevant documents. A small and simple user interface is proposed for the manipulation of the system and for the search for documents. An evaluation of the system in terms of number of operations carried out has been done, and statistics are presented of the performance of such a system. It appears that searching for Arabic documents takes much more time than searching for English/French documents; this is due to the adopted method of reduction of the Arabic words into root-pattern pairs on the one hand and, the structure of some files on the other.

The main difference between my system and the currently available systems which use the Arabic romanisation is in the way files, in which documents, are stored. While those systems use single unified file to store all documents including the Arabic ones (which are romanised), documents in my system are stored in two separate files: one for the Arabic documents and the other for English/French documents; this is to ease the acceess to Arabic documents which have a totally different structure than the other languages. My system also handles searches by keywords in a different manner than the other systems. For example if a user is searching a title which cannot be found, then a list of keywords is diplayed inviting him to go further in his investigation if he wishes to do so. This option is not at all offered to the users in the other systems, which use pattern matching only.

Finally, the choice of the reduction of Arabic words into root-pattern pairs seems the best way to control and to represent the different files which are needed for our purpose; it gives the opportunity to apply some information retrieval techniques.

7·2 IMPLEMENTATION

The presented implementation of the system was done only to test the different ideas proposed for the search of documents in a multilingual environment. The principal method introduced for searching for documents was the inverted files method, because of its simplicity and efficiency. Routines are written for this purpose, to implement all the required files and also to manipulate them. The implementation was done in three steps:

First step: The files which contains the documents are input.

Second step: The dictionaries and inverted files are constructed from the records associated with the documents; only 'title' and 'author' fields are used for the different tests.

Third step: The elaboration of the user interface for the manipulation of the prepared files.

7·3 FURTHER WORK

Multilingual document processing constitues a wide area of search, only a specific point of which has been considered in some detail. The author is aware however, that some improvement of the proposed solutions may still be needed for the following topics:

- Extend the proposed implementation allowing it to handle genuine Arabic scripts. The morphological method developed to decompose Arabic words into root-pattern pairs can be used in this case.

- Add searching by subject or by abstract in the system. Depending on the importance of the library, these two possibilities can be added to the existing ones such as searching by title and by author.

- Allow the user, who is searching for documents in one language, to get information about documents in another language covering the same subject area. This is to give more possibilities for users who are polyglot. The 'second' search is therefore done by subject which is the only link between documents written in different languages. However, the subject must be coded and unique for all these kinds of documents.

Appendix A

SYSTEM IMPLEMENTATION

A·1 INTRODUCTION

The implementation of the system is handled by routines written in pascal. The searching of documents is very similar for the three languages, so that I will deal only with the ones in English/French. Some special techniques for the Arabic language, not needed for the other two languages are also described. The main algorithms of the implementation are written in pseudo-code. Because many files and dictionaries are sequential files, the binary search algorithm is often used but not outlined in this appendix. However, to illustrate the algorithms and facilitate their comprehension, examples of documents, sorted and condensed keywords are given.

A · 2 ALGORITHMS

It is necessary, once again, to illustrate the record fields of the URF's file described in chapter 3. Such fields are document number, authors, title, edition, publisher, year, subject, abstract, call number, status, borrower type document and language.

A · 2 · 1 Determination of the keywords.

This operation is carried out by decomposing all the titles (respectively subjects or abstracts) into words. Thereafter, each word is compared with the stop list. If any exists there it is a noise word, otherwise it is a keyword as illustrated in figure 3·6 of section 3·2·3·1. However, in Arabic, after the elimination of noise words, some words are reduced into root -pattern pairs and others are irreducible as illustrated in figure 3·14 of section 3·3·2·4. Each keyword (reducible and irreducible) is linked to its origin document number. Authors are also decomposed. This operation splits each field author into a set of 'words'. For example let

/345/Bernard Houghton, John Convey/ online information retrieval systems/ (document number, authors and title) be the document. There is no noise word and the keywords are as follows:

online	345
information	345
retrieval	345
systems	345
Bernard	345
Houghton	345
John	345
Convey	345

For an Arabic document such as:

/1232/lotfe al khaled/al ketab al arabe anda al atrak wa fi urube/ which means 'The Arabic book while Turkish period and in Europe', the words 'al' , 'wa', 'fi' and 'anda' are noise words. There reducible words) and irreducible words are as follows:

Reducible words	root patten	
ketab	(ktb, FeAaL)	1232
arabe	(arb, FAaLe)	1232

Irreducible words	
atrak	1232
urubu	1232
lotfe	1232
khaled	1232

While the MARC system is not used in my solution, tags are introduced when it is convenient. All titles in the URF's file are ended by the character '$'. A word is a set of characters as defined in section 4·4 of chapter 5. Thereafter, two words are separated either by a blank or a hyphen. The following algorithm shows the decomposition of the titles into keywords.

```
while not end title do
      begin
        isolate word;
        {The construction of the word is obtained character by
        character tested with blanks and hyphens}
        if not end title then
            if word is not in stop list
            {The isolated word is checked
            whether it is a noise word}
            then
                begin
                save word;
                save doc number
                {The keyword and its origin
                doc number are stored in a file}
                end
      end;
```

Algorithm A·1: Decomposition of titles into keywords

The same algorithm is adapted to decompose authors into 'words' when they are separated by blanks, hyphens, comma or dots. Algorithm A·1 can also decompose Arabic titles by eliminating the noise words. However, a complementary algorithm for reducing Arabic words into root-pattern pairs is held in section 1·2·4. Finally, algorithm A·1 is also used to decompose any text entered by the user.

A · 2 · 2 Sorting

The file of keywords is then sorted into alphabetical order. Existing algorithms are used, especially those presented by Knuth [KNUTH 73]. Figure A·1 illustrates the sorting of keywords

Before sorting		After sorting	
sorting	123	database	161
....	.....	database	987
retrieval	145	....	....
....	.....	online	8867
database	161	....	....
....	.....	retrieval	145
sorting	876	retrieval	12000
....	.....	....	....
database	987	sorting	123
....	.....	sorting	876
online	8867	sorting	11110
....	.....	....	....
sorting	11110		
....	.....		
retrieval	12000		
....	....		

Figure A·1: Example of a sorting keywords

A · 2 · 3 Condensing

After the sorting process, there is redundancy of keywords as shown in figure A·1. The repetitive keywords must be condensed to only one keyword linked to a list of documents. This is the aim of algorithm A·2. The condensing of the above sorted keywords is shown in figure A·2 where the terms database, online, retrieval and sorting are leader keywords. Condensing takes into account the structures defined in section 5·2·1·3 and section 5·2·2·3 of chapter 5.

database	161	987	
....	....		
online	8867		
....	....		
retrieval	145	12000	
....	....		
sorting	123	876	11100
....	....		

Figure A·2: Example of a condensing keywords

```
while not end of keyword file do
begin
        keyword is leader keyword
        while next keyword is identical to leader keyword or not end of file do
           begin
              compute the number of keywords;
              save doc numbers
           end
{for each leader keyword}
        create linked list of doc numbers;
        link leader keyword to that list
end;
```

Algorithm A·2: Condensing keywords

A particular process may be added for condensing Arabic keywords. For the irreducible words, algorithm A·2 can be applied without modification. A reducible word is defined as a pair of words (r, p) where r is a root and p is a pattern; an example is given in section 3·1·2. Thereafter, the sorted roots are condensed. Each leader root is linked to its associated list of patterns where in turn each pattern is linked to its list of document numbers as described in section 5·2·2·4 of chapter 5. Algorithm A·3, which is a variant of algorithm A·2 describes this case.

```
while not end of file of roots do
begin
        root is leader root;
            while next root is identical to leader root or not end of file do
            begin
                compute the number of roots;
                save patterns
                save doc numbers
            end
{for  each leader root}
            create linked list of patterns;
            for each pattern do
                begin
                    create linked list of doc numbers;
                    link the pattern to that list
                end;
            link list patterns to leader root
end;
```

Algorithm A·3: Condensing reducible words

A · 2 · 4 Reducing Arabic words

This part is the core of the project as described in chapter 3. It gives the basic ideas to handle and to manipulate Arabic words. Algorithms are written, depending on the type of roots of words outlined in section 3·3·2·2 of chapter 3. The words do not include the noise words. Algorithm A·4 handles the first case of a root. The following abbreviations are used to simplify the writing of algorithms; comb: combination, cp: construction of pattern,

```
for all comb of three char do
        begin
           if comb in ROOT
           then
              begin
              save comb
              construct pattern (cp)
              if cp in PATTERN
              then save pair (comb,cp)
              end;
        end;
```

Algorithm A·4: Reduction of a word, first case

Algorithm A·5 handles the second case of reduction.

```
for all comb of two consecutive char do
        begin
           if comb in  TWO ROOT
           then
              begin
              duplicate second char of comb
              save  the new comb of three char
              construct pattern (cp) with new comb
              if cp in PATTERN
              then save pair (new comb,cp)
              end;
        end;
```

Algorithm A·5: Reduction of word, second case

Algorithm A·6 handles the third case of reduction. The number of operations is too high due to all the added 'vowels'.

```
for all vowel do
begin
for all position  of word do
begin
        insert vowel in word;
        save new word
        for all comb of three char of new  word do
        begin
           if comb in ROOT then
              begin
                 save comb
                 construct pattern (cp)
                 if cp in PATTERN
                 then save pair (comb,cp)
              end;
        end;
end;
```

Algorithm A·6: Reduction of word, third case

The routines associated with these last 3 algorithms are gathered in a single procedure. A word belongs either to the first, the second, the third case or it is irreducible.

A · 2 · 5 Intersection of sets

When the search starts, documents which contain the same keywords have to be output. The intersection of sets of document numbers associated with these keywords must be done. The intersection of linked lists is done using existing algorithms developed in [HOROW 86]. These are the main algorithms of the implementation.

Finally, some programming techniques are used, especially the creation of records and files. The URF's file are records where the updating is carried out by adding new records or modifying some particular fields. There is no need to develop difficult algorithms. It suffices to add new records at the end or to scan records to update a given field.

A · 2 · 6 General organisation of computations

The computations can be divided into two components:

First component: Establish the files of documents to permit document retrieval. The steps are:

1) input English/French records (EFrec)
2) input Arabic records (Arec)
3) apply algorithm A·1 to EFrec

4) apply algorithm A·1 to Arec

5) apply algorithm A·2 to results of step 3

6) apply algorithms A·4, A·5, A·6 to results of step 4

7) apply algorithm A·3 to results of step 6

This creates the files containing all the data required to retrieve documents.

Second component: Search for an individual document.

Repeat for each query:

1) choose the language (or maintain language)

2) enter searched text

3) display results of the query

The written programs permit a session to consist initially of an optional creation run, followed by a series of search runs.

Appendix B

COMPLEMENT ON ARABIC

This appendix contains some information about the Arabic language, particularly its alphabet. A table of six columns is given and details are as follows:

Column 1 provides the Arabic alphabet which has 28 characters. Arabic uses also 8 vowelisation characters and 8 compounded characters. These characters facilitate the writing and the comprehension of Arabic.

Column 2 shows the number of shapes that an Arabic character can have. The four free characters are indicated by the asterisk attached to the digit 1.

Column 3 contains the values of the Arabic romanisation developed by [CATAL 70]. Some characters are followed by a dot, whereas in reality the dot is under the character. However, the role of the dot is to indicate the similarity in pronunciation of some characters, such as the characters associated with 's' and 's.' , 'd' and 'd.' , 't' and 't.'

Column 4 gives each binary code corresponding to the Arabic character, developed by [ARABL 85]; it is a 7-bits code defined in the same manner as the Roman Alphabet ASCII. The added characters not presented in the table have also a binary code (digits, separators, vowelization characters, compounded characters).

Column 5 presents, for each Arabic character, the associated English pronunciation. The tested routines related to the romanised Arabic documents are based on this correspondence, which is simplified when necessary.

Column 6 contains the ASCII characters (e.g of a Macintosh keyboard) which correspond to the binary code of column 4. The choice of the distribution of the Arabic keys depends on the shape characteristic of each Arabic character. Practically, all characters which have the same number of shapes are put in the same area of the keyboard; for example, the characters which have only one shape are grouped in the same line and the associated Roman keys in Macintosh keyboard are Z, X, C, V, B, N, M.

Arabic	Shape	Value	Binary code	English	ASCII char
ا	2	alif	1000111	alef	H
ب	2	b	1001000	ba'a	F
ت	2	t	1001010	ta'a	J
ث	2	th	1001011	tha	E
ج	2	j	1001100	jeem	[
ح	2	h.	1001101	ha'a	P
خ	2	kh	1001110	kha'a	O
د	1*	d	1001111	dal	V
ذ	1*	dh	1010000	thal	C
ر	1*	r	1011001	ra	N
ز	1*	z	1010010	zain	B
س	2	s	1010011	seen	S
ش	2	sh	1010100	sheen	A
ص	2	s.	1010101	sad	W
ض	2	d.	1010110	dad	Q
ط	1	t.	1010111	tah	X
ظ	1	z.	1011000	dhah	Z
ع	4	‹	1011001	ain	U
غ	4	gh	1011010	ghain	Y
ف	2	f	1100001	fa	T
ق	2	q	1100010	qaf	R
ك	2	k	1100011	caf	;
ل	2	l	1100100	lam	G
م	2	m	1100101	meem	L
ن	2	n	1100110	noon	K
ه	4	h	1100111	ha	I
و	1*	w	1101000	waw	M
ي	3	y	1101001	ya	D

Table of Arabic alphabet

References

[AMANM 80] M.M. AMAN *Cataloging and Classification of Non-Western Material: Concerns, Issues and Practice* Neal-Shuman Book ORYX PRESS (1980).

[AMANM 84] M.M. AMAN *Use of Arabic in Computerized Information Interchange* Journal of the Society for Information Science Vol 35 (4), pp 204-10 (1984).

[AMERI 60] AMERICAN SOCIETY FOR TESTING AND MATERIALS *CODEN for periodical titles* ASTM Data Series DS 23B - S2 (1960).

[APPLE 87] APPLE DEVELOPERS'GROUP *Arabic Macintosh System Software Version 1.1.* (1987).

[ARABL 85] ARAB ORGANIZATION FOR STANDARDIZATION AND METROLOGY *Arab Standard Specifications* ASMO 449 (1985).

[ASHFO 88] J. ASHFORD, P. WILLET *Text Retrieval and Document Databases.* Chartwell-Bratt (1988).

[BECKE 84] J.D. BECKER *Multilingual Word Processing* Scientific American Vol 251 (1), pp 82-93 (July 1984).

[BECKE 87] J.D. BECKER *Arabic Word Processing* Communication of ACM Vol 30 (7), pp 600-10 (July 1987).

[BRICS 88] THE BRITISH COMPUTER SOCIETY *Arabisation* British Informatics Society Ltd (1988).

[BURTO 84] P.F. BURTON & J.H. PETRIE *Introducing Microcomputers: a Guide for Libraries* Van Nostrand Reinhold (UK) (1984).

[CATAL 70] CATALOGING SERVICE *Arabic Romanization* Library of Congress Bulletin 91 (1970).

[DEEME 82] S.S. DEEMER *Online in Saudi Arabia* Information Technology and Libraries Vol 1(1), pp 37-41 (March 1982).

[DOYLE 75] L.B. DOYLE *Information Retrieval and Processing* Melville Publishing Company (1975).

[FALOU 85a] C. FALOUSTOS *Access Methods for Text* ACM Computing Surveys Vol 17 (1), pp 49-74 (March 1985).

[FALOU 85b] C. FALOUSTOS *Signatures files: Design and Performance Comparison of some Signature Extraction Methods* ACM Sigmod Record Vol 4 (4), pp 63-82 (Dec 1985).

[FEDAG 88] S.S. Al-FEDAGHI & H.B. Al-SADOUN *Morphological Compression of Arabic Text* Electrical and Computer Engineering Department Kuwait University (1988).

[FRIED 67] J. FRIEDMAN & A. JEFFREYS *Cataloging and Classification in British University Libraries: A survey of Practices and Procedures* Postgraduate School of Librarianship University of Sheffield (1967).

[HABES 78] A. HABES & M.R. ZEROUALI *Conception d'une Base de Donnees pour la Gestion de Stock Reparti de Periodiques dans le Reseau ALGIST* These d'Ingeniorat en Informatique Universite d'Alger (1978).

[HARTE 86] S.P. HARTER *Online Information Retrieval: Concepts, Principles, and Techniques* Academic Press, Inc (1986).

[HEAPS 78] H.S. HEAPS *Information retrieval: Computational and Theoretical Aspects* Academic Press (1978).

[HENRY 80] W. M. HENRY, J.A. LEIGH, L.A. TEDD and P.W. WILLIAMS *Online Searching: an Introduction* Butterworths (1980).

[HORNE 70] J. HORNER *Cataloging* Association of Assistant Libraries (1970).

[HOROW 86] E. HOROWITZ & S. SAHNI *Fundamentals of data Structures in Pascal* 2nd Ed Computer Science Press (1986).

[KHALI 83] F.A. KHALID *Automation in a Special Library in Kuwait* Information Technology and libraries Vol 2 (1), pp 351-363 (Dec 1983).

[KHAYA 86] M.G. KHAYAT *Printing Arabic Text Using Dot Matrix Printers* Software - Practice and Experience Vol 16 (2), pp 165-172 (Feb 1986).

[KIMBE 74] R.T. KIMBER *Automation in Libraries* 2nd Ed Pergamon Press (1974).

[KNUTH 73] D.E KNUTH *The Art of Programming - Sorting and Searching* Vol 3 Addison Wesley (1973).

[LANCA 79] F.W. LANCASTER *Information Retrieval Systems* 2nd Ed John Wiley & Sons (1979).

[LYNCH 74] M.F. LYNCH *Computer-Based Information Services in Science and Technology-principles and Techniques* Peter Peregrinus Ltd (1974).

[MACKA 83] P.A. MACKAY *T_EX for Arabic Script* International Conference on Computing and the Humanities (6th: 1983: Raleigh) Computer Science Press, pp 391-400 (1983).

[MACKA 86] P.A. MACKAY *Typesetting Problem Scripts* Byte 11 (2), pp 201-18 (1986).

[MATTH 85] J.R. MATTHEWS *Public Access to Online Catalogs* 2nd Ed Neal-Shuman Publishers, Inc (1985).

[MILLE 82] R.B. MILLER *Nonroman Scripts and Computer Terminal Developments* Information Technologies and Libraries Vol 1 (2), pp 143-148 (June 1982).

[NABIL 84] A.NABIL, N. HEGAZI and E. ABED *A Morphology- Based Data Compression Technique for Arabic Text* 1st African Conference on Computer Communications Tunis, 21-23 May (1984).

[ODDYR 71] R.N. ODDY *Computer Processing of Library Files at Durham University* Durham University Library (1971).

[OSBOR 73] A.D. OSBORN *Serial Publications* 2nd Ed, revised. American Library Association (1973).

[ROWLE 85] J.E. ROWLEY *Computers for Libraries* 2nd Ed Clive Bingley (1985).

[ROWLE 87] J.E. ROWLEY *Organizing Knowledge* Gower (1987).

[RIJSB 79] C.J. van RIJSBERGEN *Information Retrieval* 2nd Ed Butterworths (1979).

[SADAY 89] T.A. EL-SADANY & M.A. HASHISH *An Arabic Morphological System* IBM Systems Journal Vol 28 (4), pp 600-12 (1989).

[SALMO 75] S.R. SALMON *Library Automation Systems* Marcel Dekker, Inc (1975).

[SALTO 71] G. SALTON *The SMART Retrieval System - Experiments in Automatic Document Processing* Prentice-Hall, Inc (1971).

[SALTO 89] G. SALTON *Automatic Text Processing: The transformation, Analysis, and Retrieval of Information by Computer* Addison Wesley (1989).

[SHNEI 87] B. SHNEIDERMAN *Designing the user Interface: Strategies for Effective Human-Computer Interaction* Addison Wesley Publishing Company (1987).

[TESKE 82] F.N. TESKEY *Principles of Text Processing* Ellis Horwood Ltd (1982).

[THOMA 73] P.A. THOMAS *Bibliographic Information in Library Systems* Aslib Occasional Publication No. 13 (1973).

[VICKE 70] B.C. VICKERY *Techniques of Information Retrieval* Butterworths (1970).

[WELLI 78] H.H. WELLISH *Multiscript and Multilingual Bibliographic Control: Alternatives to Romanization* Library Resources & Technical Services Vol 22 (2), pp 179-190 (Spring 1978).

[WESTL 87] D.R. WESTLAKE & J.E. CLARKE *GEAC A Guide for Librarians and Systems Managers* Gower (1987).