KNOWLEDGE-DIRECTED INTELLIGENT INFORMATION RETRIEVAL FOR RESEARCH FUNDING

Sanjith Hansraj, B. Sc (Hons)

KNOWLEDGE-DIRECTED INTELLIGENT INFORMATION RETRIEVAL FOR RESEARCH FUNDING

by

Sanjith Hansraj

Submitted in fulfillment of the requirements for the degree of

MASTER OF SCIENCE

in

COMPUTER SCIENCE

in the

Department of Computer Science and Information Systems

University of Natal (PMB)

2001

Abstract

Researchers have always found difficulty in attaining funding from the National Research Foundation (NRF) for new research interests. The field of Artificial Intelligence (AI) holds the promise of improving the matching of research proposals to funding sources in the area of Intelligent Information Retrieval (IIR). IIR is a fairly new AI technique that has evolved from the traditional IR systems to solve real-world problems. Typically, an IIR system contains three main components, namely, a knowledge base, an inference engine and a user-interface. Due to its inferential capabilities, IIR has been found to be applicable to domains for which traditional techniques, such as the use of databases, have not been well suited. This applicability has led it to become a viable AI technique from both, a research and an application perspective.

This dissertation concentrates on researching and implementing an IIR system in LPA Prolog, that we call FUND, to assist in the matching of research proposals of prospective researchers to funding sources within the National Research Foundation (NRF). FUND's reasoning strategy for its inference engine is backward chaining that carries out a depth-first search over its knowledge representation structure, namely, a semantic network. The distance constraint of the Constrained Spreading Activation (CSA) technique is incorporated within the search strategy to help prune non-relevant returns by FUND.

The evolution of IIR from IR was covered in detail. Various reasoning strategies and knowledge representation schemes were reviewed to find the combination that best suited the problem domain and programming language chosen. FUND accommodated a depth 4, depth 5 and an exhaustive search algorithm. FUND's effectiveness was tested, in relation to the different searches with respect to their precision and recall ability and in comparison to other similar systems. FUND's performance in providing researchers with better funding advice in the South African situation proved to be favourably comparable to other similar systems elsewhere.

Keywords/phrases: *Intelligent Information Retrieval, Expert Systems, Knowledge Representation, Semantic Networks, Searching, Constrained Spreading Activation.*

Preface

This study represents original work by the author and has not been submitted in any form for any degree or diploma to any other institution. Where use has been made of the work of others, it has been duly acknowledged in the text.

The experimental work described in this dissertation was carried out in the Department of Computer Studies, ML Sultan Technikon, Durban, South Africa, under the supervision of Professor Peter R. Warren, Head of Department of School of Maths, Stats and Information Technology, University of Natal, Pietermaritzburg, South Africa.

Acknowledgements

This research study is the product of the efforts and assistance of a number of people, and my heartfelt thanks and gratitude go to all of them. In particular, the following people deserve a special mention:

Firstly, I would like to thank my supervisor, Prof. P. R. Warren, for always being available to give invaluable advice, for encouraging me, for providing his expertise and spending untold hours proof-reading and assisting me with my work.

Thank you to my colleagues at ML Sultan Technikon, and in particular, members of the Research department for their assistance.

The financial support of the National Research Foundation (NRF) is gratefully acknowledged.

Special thanks go to my family, in particular, my wife Ishara who has been my inspiration and, my daughter Tiana, for their undying love, support, understanding and sacrifices.

Finally, I would like to thank the Almighty for giving me the strength and conviction to complete this dissertation.

KNOWLEDGE-DIRECTED INTELLIGENT INFORMATION RETRIEVAL FOR RESEARCH FUNDING

Contents

Chapter Two: An Overview of Information Retrieval (IR) and Intelligent Information Retrieval (IIR)

Chapter Three: The Current Techniques of Intelligent Information Retrieval Systems (IIRSs)

Chapter Four: The Reasoning Strategies and Knowledge Representation Structures of IIR

Chapter Five: IIR Systems

Chapter Six: Design of FUND

Chapter Seven: Implementation and Evaluation of FUND

Chapter One

Introduction

1.1 The Dichotomy

The central thrust of the National Research Foundation (NRF) rationale is to strengthen and enhance research initiatives at the historically disadvantaged institutions (HDIs) of South Africa (Westhuysen 1999: 3). This development is based on identified research thrusts that an institution seeks to develop and establish into centres of excellence. The problem however, is that these thrusts are usually generated by existing researchers at these institutions and consequently, most new research initiatives that require funding are either streamlined into existing research programmes or abandoned.

There is little room for new and innovative research ideas to be pursued since the novice researcher requiring funding must apply for it via an approved activity area at their institution or apply for a freestanding bursary (Westhuysen 1999: 6). The nett result is that fresh and new research ideas are stifled primarily because funding opportunities are difficult to come by. Furthermore, the very limited freestanding funding opportunities are usually reserved for established researchers and are seldom awarded to novice researchers. For more meaningful research development to take place, it is vital that we acknowledge this problem and be committed to finding ways of broadening the research scope at all levels.

Our suggestion is that we explore avenues that will increase the accessibility of funding opportunities to budding researchers in order that they may pursue personal research

interests that stimulate them. Although the NRF does fund a wide scope of research interests, our concern is that novice researchers at these institutions are generally not given an equal opportunity to pursue their own research interests as opposed to researchers within approved research activity areas that currently prevail at those institutions, partially because they do not know what funding is available to them. If the researcher had a greater scope to choose from, this may well lead to a greater throughput in research development. This mechanism will ensure "cross-pollination" of research ideas and can take place at all levels of participation. Furthermore, exposure to established research initiatives will assist in consolidating knowledge gained and the existing expertise will supplement the broadening of the research base at these institutions.

To facilitate the accessibility of funding opportunities, we believe that a solution lies in the matching of prospective researcher proposals at *any institution* to *any related* funding opportunity that exists within the NRF, independent of institutional dependencies.

The question now becomes, how does one link up several million rands of research funding available by the National Research Foundation (NRF) with several thousand prospective researchers *effectively*? This matching process is a problem for both the researcher and the grantee. The researchers want the money, based on their research interest, but are not always sure where and how to apply for it (Cohen & Kjeldsen 1987: 255). The grantees, on the other hand, want to guarantee that the grant is going to be used by the most appropriate recipient. Herein lies the dichotomy, the researchers needing the money and the donors are willing, but getting the two groups together to forge the best partnership has been difficult.

The essence of the problem therefore, has been that the task of matching a prospective researcher with a relevant grant is jeopardized by established institutional dependencies. There must be some way of resolving the problem with relative ease and efficiency, and the field of Artificial Intelligence (AI) holds the promise that we can alleviate the problem of providing more effective research advice, devoid of any institutional dependencies, by means of an automated system - and this thesis pursues this promise to see where it leads.

Currently, the task of matching a prospective researcher to an appropriate funding source is handled by, an institution-based, human expert. When GRANT (Cohen & Kjeldsen 1987: 255; Kjeldsen & Cohen 1987: 73) was being developed to solve a similar problem in the United States (US), Cohen and Kjeldsen found that the human intermediary could not accommodate all the ramifications involved in this process and was in need of assistance. A similar situation seems to exist in South Africa and this was confirmed via informal interviews with a resident technikon research funding-advisor who indicated a general unhappiness concerning the current process. Once again it seems worthwhile to look to AI to see if automated techniques can alleviate this problem in South Africa in the same way that GRANT does in the US.

Usually, a researcher seeking funding advice is first notified by the funding-advisor of all the approved activity areas of that particular institution. Failure to find an appropriate activity area into which his/her proposal will fit, results in the researcher applying for one of the limited freestanding grants.

The probability of gaining funding via an approved activity area is significantly higher than attaining a freestanding bursary. For this reason, the researcher usually restates his/her research proposal, usually in a completely new direction than his/her original proposal, in order that it may be included within an approved activity area. Consequently, the process is reversed and the researcher is slotted within one of the existing activity areas for which he/she is now eligible to apply for a grant.

As mentioned earlier, the current process is also problematic in that prospective researchers seldom receive grants for their *own* areas of research expertise outside the ambit of the approved activity areas. The task of having a new research direction be accommodated by an institution is a time-consuming and arduous task that usually requires proof of a team of researchers interested in that particular direction, before it is approved (Westhuysen 1999: 9). The research assistant interviewed revealed that the added drawbacks associated with

this process, in conjunction with the red tape involved, usually dissuades the novice researcher in pursuing this option.

Furthermore, funding advisors usually rely on their *memory* or experiential learning to offer relatively quick advice to researchers seeking funding advice. The same expert, who listens to a research proposal and, in under five minutes, suggests several appropriate funding sources, may require a few hours to translate their ideas into keywords, run a conventional database program, and wade through the results - most of which would be inappropriate in terms of the actual project. However, while a human expert can be much faster than a clumsy database program, he/she has to rely on memory - and, as pointed out in (Cohen & Kjeldsen 1987: 255 - 257), human memory, although fast for familiar terms, tends to be fallible and unreliable at times.

The reality of the situation therefore, is that only a small percentage of funded researchers actually do research in *their* chosen area of research. Consequently, the passion, fervor and commitment of being a self-motivated researcher is somewhat diminished. In addition, the inherent desire to accomplish meaningful research work is significantly suppressed. It is therefore not surprising to discover that most research work is being done predominantly in pursuit of some formal qualification rather than for pure altruistic reasons. Although some of these findings might be anecdotal, they are sufficient reasons to investigate whether AI can assist us to alleviate some of them.

In order to forge the best partnership, we need an effective information retrieval system in place to match prospective research *proposals* with available funding sources in those research areas that are directly, or *closely related*, to the research proposals. Although this is difficult task, we envisage that our attempt will broaden the researcher's choices and extend beyond the scope of those approved activity areas pertaining solely to their institution.

With regard to the current matching process, the major advantage of human memory is that it encodes the *meaning* of concepts by association with other concepts. Bearing in mind that the funding agencies are likely to fund not only research restricted by their themes or topics of interest, but also *related* research; the use of semantic memory within an expert IIR system will potentially, be able to find funding sources that standard keyword search methods (as carried out by most database-like systems) would miss (Kjeldsen & Cohen 1987: 76). Our research will investigate the effectiveness, in terms of accuracy and efficacy (as regards precision and recall), of the use of semantic memory in building an intelligent information retrieval system that attempts to emulate this human characteristic.

We will propose and demonstrate a system that will rely on a semantic network, of related research topics and concepts, to aid in finding likely funding sources within the NRF for prospective researchers with an established proposal. If the system cannot find a theme or programme to fund research on a specific topic, then it must find themes and programmes that would support work on related topics. We expect that the system will find these agencies as quickly as a human funding advisor relying on his or her memory, but keep many more alternatives permanently accessible. The hypothesis of this research therefore is, that such a system will be sufficiently accurate, as regards precision and recall, to find suitable funding possibilities within the NRF for prospective researchers. This research effort investigates whether this hypothesis is true or false.

1.2 Research Issues and Objectives

The hypothesis of this research effort is that a knowledge-directed IIR system, using a semantic network as its knowledge representation structure and a complementary reasoning strategy, will be sufficiently accurate in matching research proposals to NRF funding sources. It is widely accepted that a precision and recall rate of above 60% (Croft 1987: 198) is an acceptable degree of success for an IIR system.

In order to achieve this we will:

- Review IR and IIR in general in order to identify the area of research that shows the greatest promise to help us achieve our main aim.
- Review the current techniques used to match grantees to prospective researchers.
- Review current IIR techniques.
- Survey literature on semantic networks for the knowledge base structure as well as the Constrained Spreading Activation (CSA) as the primary search technique that forms the main focus of the dissertation.
- Choose a test domain within the National Research Foundation (NRF) – concentrating on "directed themes" only – where our system could be prototyped to incorporate terms and phrases within these subject areas.
- Design and implement the system in a suitable AI programming language.
- Build the knowledge base using a semantic network of nodes representing phrases or terms related to the "directed themes" supported by the NRF.
- Implement the inference mechanism, using base rules and multiple inheritance rules. (The Constrained Spreading Activation technique will be used as the search method for the queries to the system).
- Develop a menu-driven front-end user interface to support the system and make it more user-friendly.
- Conduct experimental tests on the system to diagnose its performance with regard to recall, precision and fallout rate.
- Draw conclusions from the experimental results by reporting on the success (or failure) of the system in comparison to other IIR techniques.

For the purpose of this and subsequent Chapters, the prototype we are developing will be called FUND. FUND will be built using Prolog (LPA Prolog for Windows) mainly because of its built-in inference mechanism that is pivotal to the search technique employed over the semantic network (section 6.2).

In outline, we first review material concerning Information Retrieval and in particular, the field of Intelligent Information Retrieval. We also survey the relevant psychological and AI literature on semantic memory, describe the architecture of our system, FUND and compare it with others doing the same task. We then discuss experiments and respective results on FUND. We measure its performance in several ways and show how it can be improved. We also discuss the feasibility of other similar systems.

1.3 Significance of the Research

The significance of the research can be observed from two perspectives. Firstly, from a practical perspective, an automated system that matches prospective researchers with appropriate funding sources could assist a human expert, or *anyone* else, by providing *relevant* research advice. The system does not require any experiential knowledge, special skills, or aptitude on behalf of the user. Furthermore, the system should be more accurate, precise and faster in providing the user with an exhaustive set of choices, many of which a human expert would usually miss (Cohen and Kjeldsen 1987: 257; Kjeldsen and Cohen 1987: 76).

Secondly, from a theoretical perspective, this research effort attempts to investigate the validity of an AI school of thought that claims that machines could, in some small way, replicate how human beings store and retrieve information. By trying to replicate semantic memory of humans with the aid of a semantic network, our research effort will attempt to partially emulate the ability of the human mind to store and retrieve information. Granted that it may not imitate *exactly* how the human mind reasons in a haphazard fashion, but it would help us understand how associative reasoning contributes to human recall ability. We will try to provide experimental evidence to support or refute this AI perspective (Chapter Seven).

In the earlier Chapters we will provide a theoretical foundation for our system by reviewing similar systems that attempt to mimic human reasoning capabilities and highlight those features that are pivotal to such systems. We hope to discover whether these AI approaches are justified or not by building and testing a prototype IIR system. If the results obtained from the testing of the system prove to be promising, we could use the prototype as a springboard for future research efforts that may result in a real-life practical application that institutions, as well as the NRF, could use to solve some of the problems outlined earlier. In addition, the same technique could be easily adapted to suit other similar problem spaces.

1.4 Research Methods and Methodology

A literature survey will be conducted to sift out the most promising technique in AI to develop a prototype IIR system to solve the practical dichotomy, outlined earlier, in part. The prototype will be tested and the results will be evaluated with respect to its effectiveness in comparison with other standard techniques. Our justification will be found in the theoretical review of the subject area in general, and an investigation of other practical applications that have been developed to solve similar problems. This methodology is in keeping with a scientific approach as outlined by Glass (1995: 3 - 7). The various research methods that will be employed to accomplish our principal aim and the sub-goals of the dissertation are enunciated below.

Literature on IR and IIR (journal publications, conference proceedings, WWW resources, etc.) will be surveyed to cover:

- A general overview of IR and IIR, highlighting:
    - the origins of IR and IIR (section 2.1);
    - the major components of IIR (section 2.3).

- The current techniques employed by modern IIR systems (Chapter Three).
- The techniques of IIR reasoning and knowledge representation techniques (section 4.1).
- The advantages and shortcomings of existing IIR systems (section 5.3).
- The psychology of Associationist's theories and semantic memory (section 5.5).
- Applications incorporating semantic networks (section 5.4 and section 5.6).
- Searching methodologies, and in particular, the Constrained Spreading Activation techniques (section 6.5).

NRF literature will be reviewed and research assistants will be interviewed to investigate and evaluate the current techniques employed by the NRF to match prospective researchers to funding sources (Westhuysen 1999: 26-49). Informal interviews with the technikon research funding-assistant will be discussed to reveal the nature and scope of the problem with respect to the shortcomings of the current process, particularly with regard to institutional dependencies (section 6.1).

AI material will be reviewed to find the most appropriate software to be used for the prototype (section 6.2). The design and implementation of the practical solution will be built in the programming language chosen (sections 6.3, 6.4, 6.5 and 6.6).

Literature concerning the Constrained Spreading Activation technique will be reviewed and investigated whether it could be adapted as the primary search technique to be implemented (section 6.5). The implementation of the depth-first algorithm from Bratko (1994: 262) will be altered to incorporate a depth-limitation parameter to determine whether it will be capable of pruning the search by eradicating distant and irrelevant links.

The method that will be employed for testing the prototype (FUND) will involve using two test subjects, namely, a research funding assistant and a masters student, that will make numerous random queries to FUND. The testing domain will be restricted to incorporate only the "directed-themes" of the NRF. The results of the testing will be tabulated and,

statistical inferences will be drawn as regards the accuracy (precision, recall and fallout rate) of the prototype. Mathematical formulae derived from previous research in IIR will be used to calculate these measurables (section 7.2). User value judgments will be used to determine the relevancy of results from each of the queries (section 7.4).

The analysis of the test results will provide us with evidence to substantiate or refute our original hypothesis. A comparative analysis between depth-4, depth-5 and exhaustive depth-first searches will be made to justify our claims. If the testing results are encouraging, we will be able to provide suggestions on how to improve the system as well as suggest future research options that will be worth pursuing (section 8.3).

1.5 Scope and Delimitations of the Research

The scope of the entire project has been limited to a prototype (FUND) that will be sufficient to demonstrate the viability of the approach we adopted. If the results from testing the prototype prove to be promising, we will be able to use the prototype as a springboard for future research efforts. Issues concerning the problems associated with the scaling up of the prototype, to become a commercially viable project, will be discussed in the final Chapter.

The scope of the knowledge base for the prototype will be restricted to incorporate only the four "directed themes" identified by the NRF for technikons, viz., *Competitive Industry, Improved Quality of Life, Sustainable Environment, and Effective SET education and Awareness* (Westhuysen 1999: 26 - 54). These, in turn, adequately reflect the ten programmes that are contained within these themes, viz.,

Competitive Industry	*Primary resource beneficiation*
	Manufacturing advancement
	Information and infrastructure systems

Improved Quality of Life	*Rural and urban development*
	Food production and food security
Sustainable Environment	*Inland resources*
	Marine and coastal resources
Effective SET education and Awareness	*Innovation and change in education*
	Preparation and development of educators
	Public understanding of SET

The keywords and phrases that will be used to build the hierarchical knowledge base that will form the semantic network (SN) for the prototype will be derived from the NRF material on each of the above programmes (Westhuysen 1999: 26 -54). The knowledge base however, will not contain an exhaustive set of terminology pertaining to these programmes and this will be revealed during testing (section 7.4).

As regards the scope and delimitation of the inference engine, we will implement a suitable AI search technique for FUND. If the results from testing are encouraging, this will be sufficient to justify the viability of using an AI technique to solve part of our original problem. Since the primary focus of the dissertation is to demonstrate the success of the inference engine over the knowledge representation structure, a simple menu-driven user-interface will suffice. Recall that our main goal is to build a *prototype* only and, any favourable results will be indicative that it has the potential to be improved on during future research.

1.6 Overview of the Dissertation Structure

Chapter One gives a brief description of the problem domain with respect to matching researchers to funding sources via the NRF and suggested a solution to resolve part of the problem from the field of AI in the form of an automated solution. Chapter Two investigates how the field of Information Retrieval (IR) can assist in alleviating the dichotomy, and we present an overview of IR and its evolution into Intelligent Information Retrieval (IIR), in particular the expert system approach, that demonstrates the greatest promise to resolve the problem. In Chapter Three we will review the current techniques employed by modern IIR systems, other than the expert system approach. Chapter Four will elaborate on the two issues central to any expert information retrieval system, viz., the reasoning strategies as well as the knowledge representation schemes. Chapter Five will review "intelligence" in IIR systems and provide an examination of issues relating to the advantages and disadvantages of IIR systems by investigating the successes and failures of existing IIR systems. In this Chapter we will also survey the various knowledge representation structures and concentrate on the area of research that has demonstrated the greatest promise, namely, Semantic Networks. Chapter Six will present an overview of current techniques employed by the National Research Foundation (NRF) for matching grants and prospective researchers, as well as demonstrate how the reasoning strategy is influenced by the choice of programming language for the prototype. This Chapter will also provide a detailed design of our solution to the problem. Chapter Seven will contain a summary of the implementation of the project as well as provide a closer inspection of whether the AI searching technique viz. Constrained Spreading Activation can be adopted. This Chapter will also give a description of any tests performed, test results, statistical calculations and a discussion of the results. Following this, conclusions will be documented and prospective future research initiatives will be suggested in Chapter Eight.

Chapter Two

An Overview of Information Retrieval (IR) and Intelligent Information Retrieval (IIR)

The introductory Chapter gave us a brief description of the scope and nature of the problem domain by highlighting the difficulties faced in matching prospective researchers to relevant funding sources of the NRF. The field of AI was investigated for assistance to partially resolve this problem. A need for an automated solution was established and it was suggested that Information Retrieval (IR), and in particular Intelligent Information Retrieval (IIR) could assist in this regard. This Chapter develops this theme and we will now show, in principle, how this can be achieved.

We begin this Chapter by presenting an overview of the field of IR and its evolution into IIR, as well as its current status, in order that we may extract the necessary tools to attack our problem. We will also give some insight into some of the techniques that will be needed in order to search for the funding data. The main focus of this Chapter will be to justify why an automated IIR technique is better suited to solve our problem than any of the other conventional techniques. As a result, we will investigate an area within the field of AI, namely expert systems, that have been successfully used to solve other similar problems. Consequently, we will review the roles of the various people involved within an IIR system, and the basic components of an expert IIR system, namely a knowledge base, an inference engine and a user-interface, that will form the building blocks of our system, FUND.

2.1 The Origins of IR and IIR

Since the 1940s the problem of information storage and retrieval has attracted increasing attention. It is simply stated that: "we have vast amounts of information to which accurate and speedy access is becoming ever more difficult" (Van Rijsbergen 1983: 4). One effect of this is that *relevant* information gets ignored since it is usually frequently missed, which in turn leads to much duplication of work and effort. It is this notion of "*relevance*" that is at the centre of all IR systems and therefore, it is not surprising to learn that it is this measure that determines the success or failure of a system.

With the advent of computers, a great deal of thought has been given to using them to provide rapid and intelligent retrieval systems. The purpose of an automated strategy is to retrieve as much as possible of the relevant information, at the same time retrieving as few of the non-relevant ones as possible. In libraries and many other institutions, some of the more mundane tasks such as cataloguing and general administration have successfully been taken over by computers. However, the problem of *effective* retrieval, even through the WWW search engines, remains to a large extent, unresolved.

In the next two sub-sections, we will give a brief outline of the evolution of IR into IIR and demonstrate how relevancy gets incorporated into the process

2.1.1 IR in Perspective

Before we trace the origins of IR systems, it is worthwhile firstly to differentiate between Data Retrieval (DR) and IR. With DR systems, we are concerned with an exact matching system that is complete and deterministic in its deductions, such as database systems. With regards to IR, the emphasis is on partial matching of the request and possibly a selection of the best matching ones thereafter. The inference mechanism of IR is of an inductive nature

that usually uses some probabilistic model that usually results in an incomplete matching. The chief distinction between DR and IR is that IR produces more *relevant* results than DR, albeit obliquely.

We will now briefly sketch the evolution of IR from DR systems and highlight the major strides made in this area as well as their shortcomings. In addition, we will highlight the relevancy issue and show that the degree of relevancy of IR systems themselves is relatively low.

IR is a branch of computer and information science that has gradually grown in importance over the more than 40-year period of its development. Since the 1950s in particular, there has been tremendous interest in text processing, indexing of information, searching, retrieval, and presentation of results to users. Luhn (1957) was one of the pioneers in this field who used frequency counts of words in the document text to determine which words were sufficiently significant to represent or characterise the document in the computer. Thus, a list of what might be called "keywords" was derived for each document. In addition, the frequency of occurrence of these words in the body of the text could also be used to indicate a degree of significance. This provided a simple weighting scheme for the "keywords" in each list and made available a document representative in the form of a "weighted keyword description" (Wong &Yao 1989:40).

The use of statistical information concerning the distribution of words in documents, were further exploited by Maron, Kuhns (1960) and Stiles (1961), both of whom obtained statistical associations between keywords. These associations provided a basis for the construction of a thesaurus as an aid to retrieval. Much of this early research was brought together with the publication of the 1964 Washington Symposium on Statistical Association Methods for Mechanised Documentation (Stevens, Guiliano & Heilprin 1964).

Spark Jones (1971: 559) carried out this work using measures of association between keywords based on the frequency of co-occurrence i.e., the frequency with which any two

keywords occur together in the same document. She demonstrated that such related words could be used effectively to improve recall, that is, to increase the proportion of the *relevant* documents that are retrieved.

The development of information structures, or data representation techniques, that covers specifically a logical organisation of information for the purpose of IR, was the next logical step in the evolution of IR. The primary reason for the slowness of development in this area of IR is that for a long time no one realised that computers would not give an acceptable retrieval time with a large document set unless some logical structure was imposed on it. This was largely due to the scantiness of experimental evidence to back this. The earlier experiments usually adopted a serial file organisation that, although it was efficient when sufficiently large number of queries was processed simultaneously in a batch mode, proved inadequate if each query required a short real-time response. The popular organisation to be adopted instead was the inverted file, but this also proved to be restrictive (Salton 1972).

More recently experiments have attempted to demonstrate the superiority of clustered files for real-time retrieval. The organisation of these files was produced by an automatic classification system. Good (1958) and Fairthorne (1961: 7) were among the first to suggest that automatic classification might prove useful in IR. Experiments on a small scale proved to be very successful (Doyle 1965: 473 - 489). However, large-scale experimentation showed this technique to be counterproductive.

Mathematical theories have been developed and applied, with on-going interest in the use of Boolean algebra, probability and artificial intelligence (AI). Data organisation, coding, and compression have been investigated and applied in a variety of contexts. However, there is still no accepted integrating theory for dealing with data and information, let alone with knowledge, which is the focus of many so-called "knowledge-based systems."

Fox (1989: 2 - 5) points out that for the past 40 years researchers have been trying to develop information retrieval (IR) systems, particularly using statistical information for the identification of content-bearing portions of document texts. Most of those IR systems in common use have been constructed to work with queries that are implicitly or explicitly expressed as Boolean logic expressions – including many of the web-enabled IR systems. However, in spite of its ability to process structured queries, the Boolean retrieval model has been criticised for its inability to provide ranked results, as all retrieved documents are considered equally important. A Boolean request is also apt to retrieve either too many or too few documents. Most of these drawbacks stem from the exact matching strategy adopted by the Boolean retrieval model (Wong and Yao 1989: 41 - 49).

By adopting a partial matching strategy, the vector space model is able to rank documents according to their similarity values with respect to a query. These similarity values were believed to reflect the degree of relevance of each document from the user's point of view. It has been pointed out in (Raghavan & Wong 1986: 280) that some earlier work with vector space model did not fully explain the various concepts involved. The misunderstandings of the interactions among these concepts may have led to some inconsistent usage of the model. In fact, some of the fundamental issues have not yet been completely resolved. One of the main criticisms of the standard vector space model is the pairwise orthogonality assumption (Salton & McGill 1983). Some attempts have been made to remove such a strict assumption but there is still a lack of vigorous justifications for using linear similarity functions in these approaches. Nevertheless, the vector space model has contributed a great deal to our understanding of the basic concepts in IR.

During the late 1970s and early 1980s there was hope that probabilistic methods would provide a sound theoretical basis for retrieval investigations. The conventional probabilistic model offers a different approach to IR. In essence, it is an adaptive model based on Bayes' decision theory (Croft 1981: 451 - 457).

In contrast to the Boolean and vector space models, the query is not formulated directly by the user. Instead, a discriminant (decision) function representing the information request is constructed by the system through an inductive learning process (relevance feedback). Although the probabilistic model is theoretically sound, due to the problem of large dimensionality, one is often forced to make some rather restrictive assumptions on an *n*th order probability distribution of index terms. The independence model (Croft 1981) is simple to use but its validity is questionable. The tree dependence model (Croft 1981) is also not very useful because it is difficult to estimate accurately the pairwise probability distributions with a small number of samples. To some extent, one can remedy this situation by enlarging the sample size. However, this makes the model impractical because the user would have to inspect a large number of documents. For these reasons, the conventional probabilistic model has gradually fallen into disfavour despite its promising beginning.

Thus far, we have sketched the evolution of IR and highlighted many of the important advances in this field as well as revealed the many shortcomings of these approaches. However, none of these techniques worked very well because their outputs did not possess a high degree of relevancy in relation to the queries. In the section that follows, we will outline how the current trends in IR have shifted to give rise to IIR that incorporates an "intelligence" component in the hope of improving the degree of relevancy.

2.1.2 Motivation for the Development of IIR

In the 1990s, it became apparent that IR systems should be somewhat "intelligent" if they are to be more effective and more adaptable to heterogeneous communities of users. More recently, researchers devoted most of their efforts in pursuit of an automated intelligent IR system that address the issues of what makes a system "intelligent."

The purpose of automatic retrieval is to allow the user to access information without understanding how to formulate queries or knowing about the structure of the knowledge base. In the sense that the knowledge base itself can supply this knowledge and use it to perform retrievals, it is "intelligent" (Aldous 1996: 139 - 142). To enable a knowledge base to do this, information about its content and structure must be included in the knowledge base itself and perhaps in an intelligent interface. In particular, the knowledge base, and its interface, must contain information that enables decisions to be made about which data are to be returned in response to information supplied by the user. This task represents the first of Sparck Jones' (1991: 560) *generic roles* for artificial intelligence in IR.

In her earlier work, Sparck Jones (1983) describes an intelligent information retrieval system as "a system with a knowledge base (KB) and inferential capabilities that can be used to establish connections between a request and a set of documents". She was among the first people to suggest the use of *inferential links* within a knowledge base to assist in retrieval of information.

Van Rijsbergen (1983) also emphasises the role of inference in his attempt to define the IIR process. He defines IIR in terms of plausible inference; that is, given a document representation D and a request R, IIR is the process of establishing a probability for D → R. If D accurately represents the meaning of the document and R is also an accurate representation of the user's information need, the documents for which this implication holds would be *relevant*.

In a real system, of course, both D and R will contain errors and we cannot retrieve only those documents for which the implication strictly holds. In this situation, retrieved documents will have a high probability of being irrelevant. Since errors in D and R are an inherent part of communication and representing "meaning", this raises the question about what levels of performance are truly possible with given representations; that is, the optimal

performance of a retrieval system may be much less than the theoretical best performance of 100% recall and precision.

Often, the first question that is usually asked when a system supporting "intelligent" information retrieval is proposed, "Why can't you just use a database?" While making "enemies" with the database community would be neither productive nor have a real point, proponents of what is traditionally considered "knowledge-based" approaches must have ready answers to show what the advantages are in pursuing one or the other.

The basic answer to this question lies in the fundamental trade-off in knowledge representation between expressability and tractability (Black 1986). A database system is a form of knowledge representation that is low on the expressive scale and high on the tractability scale, and this is their *raison d'etre*. One must understand when choosing to use a database that there is information – indeed, knowledge – which simply is too difficult to be expressed. One example is the inherent inability of a conventional database to make inferences during a search. This capability usually has to be explicitly included by additional code that transforms the conventional database. On the same note, one must understand when using a more expressive representation that response for certain queries may be quite slow – indeed, the possibility that there will be no response does exist in some systems.

A simple example of the kind of useful knowledge that is not represented within a database is the case of finding articles in a library. Consider the scenario where a user is looking for seminal papers on intelligent access to digital libraries, and finds the following entry in the database:

ARTICLE-01405::

TITLE: Knowledge Representation for Intelligent Information Retrieval
AUTHOR: Person – 11234
PUBLISHED IN: Proceedings-54382

The user decides to pick up a copy of this paper in the library, and needs to know where to find it. The next step now requires some simple reasoning on the part of the user: *since the article is published in the proceedings, if I find the proceedings I can find the article.* The user then looks up Proceedings-54382:

Proceedings-54382::

TITLE: Workshop on Intelligent access to Digital libraries

LOCATION: /dl/data/proceedings/54382

This information is within the users head and not in the database, that is, the inferential aspect associated with this reasoning is not encapsulated within the database. Rather than requiring users make this inference, a more expressive knowledge representation system would allow that piece of knowledge to be represented as a rule, like:

If ?x published_in ?y and ?y location ?z then ?x location ?z

If such a rule were represented, the entry for the article would appear with the proper location as a result of the initial query.

This is a very simple example for the purpose of illustration, and it may well be within the capability of most users to come up with it themselves, but the point is rather the fact that inferences are difficult to represent within databases. A user must *know* that an article's location is the same as the location of the thing it is published in. A more expressive knowledge representation system allows the modeller to add that knowledge to the system so that the user does not have to know it. This is the essence of intelligent assistance (Van Rijsbergen 1986: 196).

There is clearly a performance issue introduced, as necessarily specified in the expressiveness/ tractability trade-off. The cost of performing inference for every entry processed may be fairly high. Some systems provide the capacity to attach rules only to

specific entries. for example in the above case we might say the rule for inferring location should only be used for proceedings and books.

The amount of information available is increasing exponentially, and the need for more intelligence in the representation will soon increase over the need for rapid retrieval – the more information there is in the library (not to mention the unstructured information on the WWW), the more knowledge will be needed to find things. If users are expected to have that knowledge in order to find the information they want, many will not be able to find it. The survival of systems like the digital library technology, will be dependent on making as much of the information accessible with as little as possible required on the part of the user. The trade-off must be balanced; however, too much expressiveness will have the same negative effect as too little, due to increased performance problems.

A number of IR experiments have led to the conclusion that search strategies based on different retrieval models and different representation of document contents retrieve different relevant documents even though their average levels of performance may be very similar. In a recent article by Michael Gordon and Praveen Prathak (1999: 141 - 147), the retrieval effectiveness of popular search engines on the WWW was carried out and an efficiency of only 40 – 60 % was calculated, which was by far unsatisfactory. When users browse, they tend to follow only a few links. Conversely, a retrieval algorithm may follow too many links with the consequence of both high retrieval costs and doubtful results.

Frisse (1980) proposed a method that seemed to satisfy users searching for convenient starting points for browsing hierarchically structured medical handbooks. Several recent approaches adopted Frisse's ideas. Tomek and Maurer (1992) used two variations on Frisse's methods to weight links. Guinam and Smeaton (1992) used the same method to generate dynamically planned guided tours that are then presented to the user as a result to a query. Although many of these techniques proved to be successful, most were criticised as being too domain-specific and were not robust enough to be considered for other application domains.

Another important trend that has increased its interest, particularly in the late 1990s, is that of designing systems that can be more easily used, or can be used by more people, not only those who possess expert knowledge of it. It is important to identify what capabilities, knowledge, experience, and other human characteristics have an effect on retrieval performance, and to learn how to extend and improve our systems based on this understanding. In later Chapters we explore one human trait, namely, "semantic memory", in detail and attempt to replicate this process in a practical application to partially solve our original dichotomy.

Finally, although the conventional IR systems proved to be successful as far as recall was concerned they lacked an important characteristic that is crucial in determining its degree of relevancy, that is, inferential capabilities. Therefore, the incorporation of an "intelligence" component within an IR system was the next logical step in the evolution of IR systems in order to make them more robust and adaptable with regards their performance. It was commonplace for IIR systems to possess inferential links between the individual data items within their "databases" that formed the glue that assisted during the search process. In the sections that follow we will investigate those concepts that underpin modern IIR systems and highlight their advantages whilst exposing their shortcomings in pursuit of ways to overcome them during the design of FUND.

2.2 Initial Concepts and Methods behind IIR

Most of the conventional information retrieval algorithms have been developed for searching in large linear text collections, and the retrieval results so far have been satisfactory. As pointed out in the previous section, conventional IR algorithms are not suited to retrieve non-linearly organised information. In the development of our IIR system, our focus will be on exploiting the links when specific content-related information is to be retrieved. We will present new IR algorithms that will make use of the semantic

content of the links involved. In our experiments we will be working with textual descriptors exclusively in the form of Prolog facts and rules.

The purpose of the semantic links is to point to similar, more detailed, or additional information on a specific topic (Frei & Schach 1991). To facilitate content-specific retrieval, nodes and semantic links are usually indexed. It is generally agreed that the more information is integrated into the retrieval process, the better the retrieval is likely to be. For this reason we anticipate that suitable semantically linked information available to the retrieval algorithm should lead to the desired effect. However, although indexing is an important aspect, we will not incorporate it into FUND, primarily because it is a prototype that will not need more than five link types. Indexing however, is a viable option that could be tackled by future research efforts (section 8.3)

Our focus will be on the nature of the semantically linked information and, how it could be used to improve the degree of relevancy of searches. The hope is that semantic links contain the necessary information to decide whether any nodes that are further away, should be visited by the retrieval algorithm, or not. The proposed automated navigation through the search is governed by the following considerations (Duda, Gaschnig & Hart 1978):

- The further away from the initial node the IR algorithm searches, the less likely it is to find suitable information.
- Links are only followed when they promise to point to nodes containing information relevant to the query.
- It may become optional to visit a specific node (e.g., because of given restrictions during the retrieval process) depending on the retrieval resources available.

In this way the descriptions of semantic links control the navigation process of the retrieval algorithm.

Before examining the design and implementation of FUND, it is important that we investigate the issues that individually make up a generic IIR system.

2.3 Basic Components of an Expert IIR System

As pointed out in the previous sections, the realisation of the need to incorporate and use knowledge (or intelligence) within IR systems has led researchers to look to artificial intelligence (AI) systems that also aim to incorporate and use knowledge, that is, intelligent knowledge-based systems, and in particular, one class of such systems – expert systems (Brooks 1987: 367). It is therefore worthwhile reviewing literature on expert systems and establishing just what an expert system is, what it does, and how it accomplishes this.

An intelligent knowledge-based system is usually understood to be a set of programs and data that contain human knowledge (Muller 1985). Spark Jones (1983) defines it as a system that uses inference to apply knowledge to perform some task. The term "knowledge" is used very loosely in this context. Hayes-Roth and Waterman (1983: 3 - 6) simply define it as something that incorporates facts, beliefs, and heuristic rules. The "intelligent" in intelligent knowledge-based systems, if it is defined at all, is usually given as a characteristic of a system that is behaving in such a way that were a human to be behaving in the same way in a similar situation, that behaviour would be described as intelligent (Brooks 1987: 368).

Definitions of expert systems given by various authors display a wide diversity of interpretation and emphasis. However, the formal definition approved by the British Computer Society's Committee of the specialist group on expert systems reads as follows:

> *"An expert system is regarded as the embodiment within a computer of a knowledge-based component from an expert skill in such a form that the system can offer intelligent advice or make an intelligent decision about a processing function.*

A desirable additional characteristic, which many would consider fundamental, is the capacity of the system, on demand, to justify its own line of reasoning in a manner directly intelligible to the enquirer. The style adopted to attain these characteristics is rule-based reasoning." (Brooks 1987: 369)

Thus, it is evident that an expert system is a program that behaves like an expert for some problem domain. It should be capable of *explaining* it decisions and the underlying reasoning. We must not forget, however, that often an expert system is expected to deal with uncertain and incomplete information (Bratko 1994: 344). In the next section of this Chapter, we will review some of the basic components of an expert intelligent information retrieval system that will form the building blocks of FUND.

2.3.1 The Main Structure of an Expert IIR System

To build an expert IIR system we have to, in general, develop the following functions:

- *Problem-solving function* capable of using domain-specific knowledge which may require dealing with uncertainty;
- *User-interaction function* that includes explanation of the system's intentions and decisions during and after the problem-solving process (Bratko 1994: 332).

Each of these functions can be very complicated, and can depend on the domain of application and practical requirements. Various intricate problems may arise in the design and implementation. This involves the representation of knowledge, that is the intelligible manner in which the knowledge is stored, and associated reasoning, that is the technique used to retrieve the data.

In this section, we will develop a framework of basic ideas and concepts needed for the building of FUND.

From much of the literature surveyed, we found that it was convenient to divide the development of an expert IIR system into three main modules, as illustrated below in fig.2.1 (Bratko 1994: 332):

I. ***A Knowledge Base (KB)*** – a declarative representation of the expertise, often in IF...THEN rules, semantic networks, frames, etc.;

II. ***An Inference Engine*** – the code at the core of the system that derives recommendations from the knowledge base and problem specific data in working storage.

III. ***A User Interface*** – the code that controls the dialogue between the user and the system.

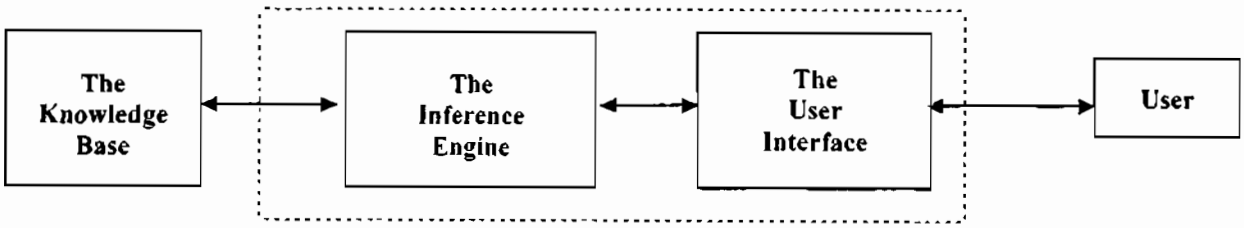

Fig. 2.1. The components of an expert IIR system

To understand expert system design, it is also necessary to understand the major role players who interact with the system (Dennis 1989):

I. ***Domain expert*** – the individual or individuals who currently are experts solving the problems that the system is intended to solve;

II. ***Knowledge engineer*** – the individual who encodes the expert's knowledge in a declarative form that can be used by the expert system;

III. ***User*** – the individual who will be consulting the system to get advice that would have been provided by the expert.

Later (section 5.2), we will highlight the role these individuals play in making an IIR system "intelligent".

Many expert systems are built with products called expert system *shells*. The shell is a piece of software that contains the user interface, a format for declarative knowledge in the knowledge base, and an inference engine. The knowledge engineer uses the shell to build a system for a particular problem domain.

Expert systems are also built with shells that are custom developed for particular applications. In this case there is another key individual:

IV. ***System engineer*** – the individual who builds the user interface, designs the declarative format of the knowledge base, and implements the inference engine.

Depending on the size of the project, the knowledge engineer and the system engineer might be the same person (Dennis 1989). For a custom built system, the design of the format of the knowledge base, and the coding of the domain knowledge are closely related. The format has a significant effect on the coding of the knowledge.

One of the major bottlenecks in building expert systems is the knowledge engineering process. The coding of the expertise into the declarative rule format can be a difficult and tedious task. One major advantage of a customised shell is that the format of the knowledge base can be designed to facilitate the knowledge engineering process.

The objective of this design process is to reduce the *semantic gap* (Dennis 1989). Semantic gap refers to the difference between the natural representation of some knowledge and the programmatic representation of the knowledge. For example, compare the semantic gap between a mathematical formula and its representation in both assembler and FORTRAN. Clearly, FORTRAN code (for formulae) has a smaller semantic gap and is therefore easier

to work with. Since the major bottleneck in expert system development is the building of the knowledge base, it stands to reason that the semantic gap between the expert's representation of the knowledge and the representation in the knowledge base should be minimised. It is this notion of semantic gaps that is largely responsible for the introduction of the subjectivity characteristic within most expert systems and will be addressed later (section 6.5.4) in more detail.

2.3.2 The Knowledge Base (KB)

The knowledge base comprises the knowledge that is specific to the domain of application, including such things as simple facts about the domain, rules that describe relations or phenomena in the domain, and possibly also methods, heuristics and ideas for solving problems in this domain (Bratko 1994: 333).

How are knowledge bases built? Typically, a knowledge engineer interviews a domain expert to elucidate expert knowledge, which is then translated into rules (Elaine & Kevin 1991: 515). The development of an expert system involves the frequent consultation of an expert (or a group of experts) with the aim of developing a knowledge base that can be manipulated. Thus, the first phase of the process consists of the formation of a database of domain specific knowledge. In the expert interrogation process, the formulation of "good" questions is paramount. Fortunately, experts often phrase problem-solving questions in terms of IF...THEN structures. For example (Robert 1990: 364),

"IF it looks like a duck, talks like a duck, ... then I classify it as a duck."

After the initial system is built, it must be iteratively refined until it approximates expert level performance. This process is expensive and time-consuming, so it is worthwhile to look for more automatic ways of constructing expert knowledge bases. While no totally automatic knowledge acquisition system exists, there are many programs that interact with

the domain experts to extract expert knowledge efficiently. These programs provide support for the following activities (Elaine & Kevin 1991: 553):

- Entering knowledge
- Maintaining knowledge base consistency
- Ensuring knowledge base consistency.

The knowledge that needs to be incorporated into an IIR system is diverse and extensive, involving many different kinds of knowledge including knowledge of the task (relating terms to problems), knowledge of the relationship between phrases and their descriptions, knowledge about the "database" and its access mechanisms, knowledge of the users, knowledge of the users problems, and knowledge of the subject domain underlying the problems (Brooks 1987: 368). Relatively little work has been done to explore the *nature* of the knowledge needed to carry out IR effectively.

Quite clearly, the issue of what kinds of knowledge and what individual pieces of knowledge are required needs to be addressed before any IIR system based on any kind of AI technique is developed.

The other consideration concerns the *type* of knowledge required by an IIR system. IR systems, we hope, will deal with quite broad subject domains, for example, medicine, rather than with very specific domains, such as diagnosis of blood disorders. Even if the IR system is restricted to problems and documents in a relatively narrow area, such as concrete or fluid mechanics, the system will require knowledge of a very large number of concepts, as well as overall knowledge of the subject area and its structure.

Thus, it is evident that the knowledge base is an important, if not the most important, component of an expert IIR system. It embodies the expert knowledge the system must use to solve the problems it is given, plus the added feature of having some structure that adequately resembles relationships between the data items.

Different types of knowledge may need to be incorporated, for example, knowledge about solving the problem, knowledge about entities in the domain and their relations, and facts about the problem domain. Factual information is data about the domain that does not vary whatever the particular circumstances being considered. Data specific to the set of conditions being considered at the time is held in "working memory", which may be either in the computer's main memory or in an external database structure and is usually discarded at the end of the session.

Human expert knowledge can be represented in the knowledge base of an expert system in a variety of ways. The three main categories of knowledge representation schemes (discussed in the Chapter Four) – frames, semantic networks, and production rules – have all been used to construct expert system knowledge bases (Brooks 1987: 368). The method most commonly associated with expert systems is that of production rules, but there is no requirement that this must be the case. Nor does the mere fact of using production rules imply that a system is necessarily an expert system. CASNET, for instance, employs a causal network representation, that is, a semantic network, and PIP uses frames (Pople 1982).

Some expert systems use a combination of production rules and some other form of representation, for example, PROSPECTOR has the ore model coded in an inference network, which is represented internally as a semantic network; general knowledge about the character of a particular class of deposits is represented by production rules. The main design criteria when selecting a knowledge representation scheme are (Levesque & Brachman 1985):

- Representation of knowledge in a form that is consistent with the domain being represented, and;
- Efficiency as far as use in the problem-solving task is concerned.

The complex decision as to which knowledge representation structure to use for FUND will be tackled later (Chapter Five).

2.3.3 The Inference Engine

The inference engine or reasoning mechanism is the system component that applies and manipulates the knowledge base. The form the inference engine takes is dependent on such factors as the structure of the knowledge base, which method of representation has been used, how large the knowledge base is, and whether a confidence measure is attached to the knowledge, but it is completely independent of the content of the knowledge base (Hayes-Roth, Waterman & Lenat 1983; Feigenbaum 1979; Buchanan, 1982).

The manner in which the inference mechanism is implemented varies considerably. Certain techniques are used with particular forms of representation, for example, use of decision trees or RETE match algorithms for production rules (Forgy 1982) or of Spreading Activation for semantic networks (Kidd 1985: 9 - 19). Interpretation, the process by which parts of the knowledge base are matched with information about the current problem in working memory, is also highly dependent on the representation formalism used.

In addition to interpretation, the inference engine must also control the search through the knowledge base for appropriate fragments to interpret. Production systems, for example, are usually controlled by one of two strategies: forward chaining (or data-driven or bottom-up control) and backward chaining (or goal-driven or top-down control). In forward chaining the system reasons from a set of data to an outcome or goal-state. Backward chaining involves working back from a goal or conclusion to see if the conditions that would make it true are satisfied. Both these reasoning mechanisms will be revisited in the Chapter Four (section 4.1) when we investigate which one is more suited to our problem domain.

A few systems (e.g., INTERNIST/CADUCEUS) use a mixed strategy; that is, they employ both forward and backward chaining. This is the way most humans often try to resolve problems – "working from both ends" in a haphazard fashion, however, the large amount of coding required does not adequately justify the improvement in the recall ability.

Another essential aspect of many expert systems is the need to reason with uncertainty. This can arise because of inherent imprecision in the rules or because of imprecision in the data values. We shall return to this aspect in Chapter Four (section 4.1.4) as well.

The question as to which reasoning strategy will best suit FUND will be tackled later (Chapter Six).

2.3.4 The User Interface

The user interface is the system component responsible for communication between the user and the system. With a few exceptions, most expert systems require information from the user in order to specify the problem to be solved and to provide an initial set of conditions or observed data. However, much research in the field of expert systems have focussed on the issues of knowledge representation and reasoning mechanisms. Relatively little effort has been expended to make the systems user-friendly. It has been commented ironically that, many existing expert systems seem designed solely for use by human experts (Brooks 1987). This presents a considerable barrier to the employment of expert systems as advisors to users who are not experts in the problem domain.

Most expert systems require the user to communicate with them using a rigidly constrained, system-oriented dialogue. This usually consists of an exhaustive set of yes/no or menu-styled closed questions (Kidd 1985), thereby avoiding all but the most minimal language processing. A few of the more highly developed user interfaces (e.g., the interface in PROSPECTOR) have natural language processors to parse and interpret user questions,

volunteered information, etc. Several expert systems require no direct user input at all. XCON (McDermott 1982: 39 - 88), for instance, takes its initial data from the customer's order form and requires no intervention or data from the user.

A few expert systems, such as PROSPECTOR, will allow the user to enter data at any point during the session. It is usually the case, however, that certain categories of data must be entered at specific points. The user often has limited possibilities for interrupting his/her dialogue with the system (Kidd 1985; Elaine & Kevin 1991: 548). If any essential piece of information is missing, the system may pause during the problem-solving process to request it from the user. The lack of flexibility means that users carrying out a task in real time may find the dialogue laborious and inefficient.

As far as FUND is concerned, a simple menu-driven user-interface (section 6.6) will suffice, as the main focus of our dissertation is not to develop a highly interactive user-interface, but rather to determine whether an AI reasoning technique could be used effectively over an AI knowledge representation structure to assist in alleviating our original problem.

2.3.5 Explanation Mechanisms

Many definitions of an expert system include the criterion that the system should be able to justify its conclusions. The ability of the system to offer some sort of explanation was felt to be necessary, since users might be expected to take some action based in part on the advice given by the system and would therefore need and require an understanding of the rational basis of the system's decisions (Buchanan 1982). It was thought that medical diagnosis systems, for example, would lack credibility if they were unable to justify their reasoning (Shortliffe 1976; Swartout 1977; Buchanan & Shortliffe 1984 a). On a more practical note, provision of an explanation facility was seen as an invaluable tool for the

debugging of both the system (by the system designer) and the knowledge base (by the human expert and/or knowledge engineer) (Buchanan & Shortliffe 1984 b).

The explanation mechanisms used in existing expert systems range from non-existent (e.g., DENDRAL, XCON) to ones that justify a decision by producing a list of the steps they have taken to reach that decision, usually a list of rules activated (e.g., MYCIN) (Elaine & Kevin 1991: 550). How complex the explanation mechanism of an expert system needs to be depends on the problem domain. Systems that require little or no user interaction will have rudimentary or non-existent explanation facilities.

Rule listing and trail audit facilities will obviously be available to the system developer but do not have to be displayed to the user. Systems that have high levels of user interaction and require active user participation will need sophisticated explanation facilities that can, at the very least, reconstruct a rational line of argument and display this in natural language text. Ideally such explanation systems should also be able to tailor the explanation to its intended audience, but this ideal is a long way off.

FUND will need to explain it's reasoning to the user as a means of justifying why certain alternatives were chosen and to demonstrate the context of the nodes returned.

Finally, it should now be clear that we could not proceed to build an IIR system without investigating the relative importance of the various components that make up the complete system as well as the roles of the individuals involved. In the next two Chapters, we investigate the current IIR techniques and various reasoning techniques involved within IIR system development as well as the knowledge representation schemes available so that we may be able to incorporate them into FUND's design.

2.4 Conclusion

In this Chapter, we provided an overview of the evolution of Information Retrieval (IR) from Data Retrieval (DR) to Intelligent Information Retrieval (IIR) to put into perspective our pursuit of an AI solution to our problem. Conventional DR systems were found lacking a high degree of relevancy between their queries and their outputs (section 2.1.1). The most significant problem associated with conventional IR systems was that of its inherent inability to adequately encapsulate inferences. Furthermore, IR systems were also found to possess a relatively low degree of relevancy (section 2.1.1).

IIR systems, in particular expert systems, are however, able to build in inferential capabilities within the knowledge base (section 2.2), and we suspect that this approach could be used in the development of our prototype. For this reason, we highlighted the most important concepts that make up a generic expert IIR system, namely, the knowledge base, the inference engine and the user interface. We also provided an overview of the people involved in the IIR process and discussed the importance of their contributions (section 2.3).

Thus far, we have established a need to decide on an appropriate knowledge representation structure and a complementary reasoning strategy for FUND. Before we begin to make these decisions, we need to review, in the next Chapter, the current state of IIR, and investigate whether any of the modern techniques, other than the expert systems approach, could assist us in resolving our original problem.

Chapter Three

The Current Techniques of Intelligent Information Retrieval Systems (IIRS)

In the previous Chapter we presented an overview of the field of Information Retrieval (IR) and traced its evolution to current day Intelligent Information Retrieval (IIR). The focus of this Chapter will be a review of the current techniques employed by IIRSs to determine if any of them could assist us in resolving our original difficulty of matching prospective researchers to *relevant* funding opportunities within the NRF.

We begin this Chapter by reviewing the varied techniques employed by modern IIRSs and focus on the Artificial Intelligence (AI) techniques, other than the expert systems approach, that are used to solve problems that resemble ours.

3.1 Overview of Information Retrieval Research Areas

Much of IR research over the past nine years has been primarily directed at the basic issues of text representation, query acquisition, and retrieval models that form the basis of all search engines (Croft 2002). These core areas have been extended into a number of different architectures, such as filtering information streams and searching distributed databases. It has been extended into different languages in both multilingual and cross-lingual systems. It has been extended into different modes of interaction, such as graphic-based visualisation techniques. Finally, it has been extended into different data types, such as images, speech, video, and music.

In the forthcoming subsections, we will briefly describe the current state of research in the most prominent areas mentioned above. Since relevance feedback and probabilistic models for IR have been around for the longest, it is fitting that we begin our investigation with the most recent developments in these areas.

3.1.1 Relevance Feedback Models in IR

In classical IR models, relevance feedback, document space modification, probabilistic techniques, and the Bayesian inference networks are among the techniques most relevant to our research.

Relevance feedback is a process where users identify relevant information in an initial list of retrieved results, and the system then creates a new query based on those sample relevant hits. Algorithms for automatic relevance feedback have been studied in IR for more than thirty years, and the research community considers them to be thoroughly tested and effective (Croft 2001).

Most of the relevance feedback experiments reported in the IR literature were based on small test collections of abstract-length documents. The central problems in relevance feedback are selecting "features" (words, phrases) from relevant documents and calculating weights for these features in the context of a new query. These problems are substantially more difficult in environments with large databases of full-text documents. In addition, people searching databases in real applications often use relevance feedback in different ways than anticipated by IR researchers. Feedback techniques were developed to improve an initial query and assumed that a few relevant documents would be provided. In many real interactions, however, users specify only a single relevant document. Sometimes that relevant document may not even be strongly related to the initial query, and the user is, in effect, browsing using feedback.

These factors mean that traditional feedback techniques can be unpredictable in operational settings. Research aimed at correcting this problem is underway and more operational systems using relevance feedback can be expected in the near future (Croft 2001). Relevance feedback techniques are also an important part of building profiles, with the main difference being the number of example relevant documents available.

3.1.2 Modern Probabilistic Models in IR

The probabilistic approach to retrieval was first presented in Maron and Kuhns (1960). Since then it has been elaborated in different ways, tested and applied (Sparck et al 2000: 779). Although there are currently several distinct versions of the probabilistic approach, there is still a well-established core theory that permeates them all.

In Probabilistic IR the goal is to estimate the *probability of relevance* of a given document to a user with respect to a given query. Probability assumptions about the distribution of elements in the representations within relevant and irrelevant documents are required. Using relevance feedback from a few documents, the model can be applied in order to estimate the probability of relevance for the remaining documents in a collection (Fuhr & Pfeifer 1994).

Fuhr and his co-workers (1994) presented three probabilistic models as an application of machine learning (section 3.2). First, the classical binary independence retrieval model implemented a *query-oriented* strategy. In the relevance feedback phase, given a query, relevance information was provided for a set of documents. In the application phase, this model can be applied to all documents in the collection, but only for the same initial query. The second *document-oriented* strategy collected relevance feedback data for a specific document from a set of queries. The parameters derived from these data can be used only for the same document, but for all queries submitted to the system. Neither of these strategies can be generalised to all documents and for all queries.

Fuhr et al (1994) proposed a third, *feature-oriented* strategy. In the previous two strategies, the concept of abstraction was adopted implicitly by regarding terms associated with the query or the document, instead of the query and the document. In this feature-oriented strategy, abstraction was accomplished by using features of terms (e.g., the number of query terms, length of the document text, the with-document frequency of a term, etc.) instead of terms themselves. The feature-oriented strategy provides a more general form of probabilistic learning and produces bigger learning samples for estimation; but the disadvantage is the heuristics required to define appropriate features for analysis. Later, Fuhr and his co-workers adopted more general-purpose statistical and machine learning algorithms, such as regression and decision-tree building algorithms.

Further extensions of the probabilistic models include the use of Bayesian classification and inference networks for IR and indexing (Chen 1994). Recently, several probabilistic inference network models using Bayesian networks and causal networks have been proposed (Syu & Lang 2000: 314). In an application described in (Fung & Favero 1995), the users were allowed to specify the topics of interest, the relationship among the topics, the document features that relate to the topics, and the relationships between the document features and the topics.

Although relevance feedback and probabilistic models exhibit interesting query or document refinement capabilities, their abstraction processes are based on either simple addition/removal of terms or probabilistic assumptions and principles. Their learning behaviours are very different from those developed in symbolic machine learning, which will be the focus of our next section.

3.2 Machine Learning for Information Retrieval (IR)

IR using probabilistic techniques has attracted significant attention on the part of researchers in information and computer science over the past few decades. More recently,

information science researchers have turned to other newer artificial intelligence (AI) based inductive learning techniques including neural networks, symbolic learning, and genetic algorithms (Chen 1994). These newer techniques, which are grounded on diverse paradigms, have provided opportunities for researchers to enhance the information processing and retrieval capabilities of current information storage and retrieval systems. Chen (1994) believes that "with proper user-system interactions, these methods can greatly complement the prevailing full-text, keyword-based, probabilistic, and knowledge-based techniques."

In the subsections that follow, we will review the probabilistic and knowledge-based techniques and the emerging machine learning methods developed in AI and their use in information science research.

3.2.1 Emergence of the Machine Learning Approach

For the past few decades, the availability of cheap and effective storage devices and information systems has prompted the rapid growth and proliferation of relational, graphical, and textual databases (Chen 1994). Information collection and storage efforts have become easier, but effort required to retrieve *relevant* information has become significantly greater, especially in large-scale databases. Information stored in these databases, as pointed out in Chapter Two (section 2.1.2), often has become voluminous, fragmented, and unstructured after years of intensive use. Often, it is usually only users with extensive subject area knowledge, system knowledge, and classification scheme knowledge that are able to manoeuvre and explore in these databases.

Most commercial IRSs still rely on conventional inverted index and Boolean querying techniques. Probabilistic retrieval techniques have been used to improve the retrieval performance of IRSs. The approach was based on two main parameters, the probability of relevance and the probability on irrelevance of a document. Despite various extensions, as

pointed out in section 3.1.1, probabilistic models still require the *independence assumption* for terms and it suffers from difficulty of estimating term occurrence parameters correctly.

Since the late 1980s, knowledge-based IR techniques were used extensively by information science researchers for IR. These techniques have attempted to capture searchers' and information specialists' domain knowledge and classification scheme knowledge, effective search strategies, and query refinement heuristics in document retrieval systems design. Despite their usefulness, systems of this type were considered *performance systems* (Simon 1991) – they only perform what they were programmed to do (i.e., they were without *learning* ability). Significant efforts were often required to acquire knowledge from domain experts and to maintain and update the knowledge base.

A newer paradigm, generally considered to be the *machine learning* approach, has attracted attention of researchers in AI, computer science, and other functional disciplines such as engineering, medicine, and business (Weiss & Kulikowski 1991). In contrast to *performance systems* which acquire knowledge from human experts, *machine learning systems* acquire knowledge automatically from examples, i.e., from source data. The most frequently used techniques include symbolic, inductive learning algorithms such as ID3 , multi-layered feed-forward neural networks such as Backpropagation networks, and evolution based genetic algorithms. We will investigate these techniques in greater detail in the next sub-section.

3.2.2 Learning Systems: An Overview

The symbolic machine learning technique, the resurgent neural networks approach, and evolution-based genetic algorithms provide vastly different methods of data analysis and knowledge discovery (Weiss & Kulikowski 1991). We provide below a brief overview of these three classes of techniques:

- ***Symbolic Learning:***

These techniques can be classified based on such underlying learning strategies as rote learning, learning by being told, learning by analogy, and learning from discovery. Among these techniques, a special case of inductive learning appears to be the most promising symbolic machine learning technique for knowledge discovery or data analysis. Among the numerous symbolic learning algorithms that have been developed, Quinlan's ID3 decision-tree building algorithm and its descendants are popular and powerful algorithms for inductive learning.

- ***Neural Networks:***

The foundation of the neural networks paradigm was laid in the 1950s and this approach has attracted significant attention in the past decade due to the development of more powerful hardware and neural algorithms (Rumelhart et al 1994). Neural networks, a connectionists learning approach, provides a convenient knowledge representation for IR applications in which nodes represent IR objects such as keywords, authors, and citations and bi-directional links represents their weighted associations (of relevance).

In symbolic machine learning, knowledge is represented in the form of symbolic descriptions of the learned concepts, e.g., production rules or concept hierarchies. In connectionist learning, on the other hand, knowledge is learned and remembered by a network of interconnected neurons, weighted synapses, and threshold logic units (Rumelhart et al 1994). Learning algorithms can be applied to adjust connection weights so that the network can predict or classify unknown examples correctly. The most widely used neural networks technique used for IR tasks is Backpropagation networks.

- ***Genetic Algorithms:***

  Genetic algorithms (GAs) were developed based on the principles of genetics (Fogel 1994). In such algorithms a population of individual (potential hits) undergoes a sequence of unary (mutation) and higher order (crossover) transformations.

  Our literature survey revealed several implementations of GAs in IR. Gordon (1988) presented a GA-based approach for document indexing. Competing document descriptions (keywords) are associated with a document and altered over time by using genetic mutation and crossover operators. In his design, a keyword represents a gene (a bit pattern), a document's list of keywords represents individuals (a bit string), and a collection of documents initially judged relevant by a user represents the initial population. Based on a Jaccard's score matching function (fitness measure), the initial population evolved through generations and eventually converged to an optimal (improved) population – a set of keywords that best described the documents. In his later work, Gordon (1991), adopted a similar approach to document clustering. His experiments showed that after genetically re-describing the subject description of documents, descriptions of documents found co-relevant to a set of queries will bunch together.

  Petry et al. (1993) applied genetic programming to a weighted IRS. In their research, a weighted Boolean query was modified in order to improve recall and precision. They found that the form of the fitness function has a significant effect on performance. Yang and his co-workers (1993) have developed adaptive retrieval models based on genetic algorithms and the vector space model using relevance feedback.

More recently, there have been several studies (Croft 2002) that compared the performances of these techniques for different applications as well as some systems that

used hybrid representations and learning techniques. For example, Kitano (1996) and Harp (1998) considered hybrid systems that utilised both GAs and neural networks with favourable results. Systems, such as COGIN (Green & Smith 1998) performed symbolic induction using genetic algorithms and SC-net, which is a fuzzy connectionist expert system. Other hybrid systems developed in recent years employ symbolic and neural net characteristics. For example, Touretzky and Hinton (1998) and Gallant (1998) proposed connectionist production systems, and Shastri (2001) developed connectionist semantic networks.

3.2.3 Concluding Remarks and Future Directions of Machine Learning

IR research has been advancing very rapidly over the past few decades. Researchers have experimented with techniques ranging from probabilistic models and the vector space model to the knowledge-based approach and the recent machine-learning approach (Chen 1994). At each stage, significant insights regarding how to design more useful and "intelligent" IR systems have been gained.

In the preceding subsections, we presented an extensive review of IR research that was mainly based on machine learning techniques. Connectionist modelling and learning, in particular, has attracted considerable attention due to its strong resemblance to some existing IR models and techniques. Symbolic machine learning and genetic algorithms, two popular candidates for adaptive learning in other applications, on the other hand, have been used only rarely. However, these newer techniques have exhibited promising inductive learning capabilities for selected IR applications.

From this review, we learn that the proper selection of knowledge representation and the adaptation of machine learning algorithms in the IR context are essential to the successful use of such techniques, and this will be further evidenced by the choice we make for the solution to our original dichotomy in Chapter Six.

Despite some of the initial successful applications of selected machine learning techniques for IR, there are numerous research directions that need to be pursued before we can develop a robust solution to "intelligent" information retrieval. These include:

- *Limitations of learning techniques for IR*

  The performance of the inductive learning techniques rely strongly on the examples provided (Weiss & Kulikowski 1991). In IR, these examples may include user-provided queries and documents collected during relevance feedback. In reality, user-provided relevance feedback information may be limited in quantity an noisy (i.e., contradictory or incorrect), which may have adverse effects for the IR or indexing tasks. Some learning techniques such as the neural networks approach have documented noise-resistant capability, but empirical evidence and research need to be performed to verify this characteristic in the context of IR and indexing.

  For large-scale real-life applications, neural networks and genetic algorithms may suffer from requiring extensive computation time and lack of interpretable results. Symbolic learning, on the other hand, efficiently produces simple production rules or decision tree representations. The effects of the representations on the cognition of searchers in the real-life retrieval environments (e.g., users' acceptance of the analytical results provided by an intelligent system) remain to be determined.

- *Applicability to the full-text retrieval environment*

  In addition to extensive IR research conducted in probabilistic models, knowledge-based systems and machine learning, significant efforts have also been made, by many commercial companies, in pursuit of more effective and "intelligent" IR systems. Most full-text retrieval software has been designed to handle large volumes of text by indexing every word (and its position). This allows users to perform proximity searches, morphological searches (using prefix, suffix and wildcards), and thesaurus searches at the expense of implementation details.

Most recently, machine-learning techniques have found greater prominence in areas related to the Internet and the World Wide Web (WWW) (Mladenic & Stefan 1999). In particular, they form the backbone of most "intelligent agents." These are systems that assist users by finding information or performing simpler tasks on their behalf, such as Web browsing by retrieving documents similar to already-requested documents (Mothe et al 1997). Two frequently used methods for developing intelligent agents based on machine learning techniques are *content-based* and *collaborative* approaches (Chrisment et al 1997). Both can help users find and retrieve *relevant* information from the Web.

In the sections that follow, we will investigate Web-related IR further.

3.3 The World Wide Web (WWW) and IR

The tremendous growth of the WWW and the increasing plethora of web search engines and Web sites have broadened the scope of IR research to include aspects of Web-based IR (Spink & Qin 2000: 205). Although the major focus of the IR community remains the annual TREC competition, a growing number of IR researchers are working with the problems inherent in designing real systems for real Web users. Many of the classic issues challenging IR researchers from the 1950s remain with sharper dimension on the Web.

Finding wanted information on the WWW can be a frustrating and disappointing experience (Wang et al 2000: 230). Web resources are significantly different from traditional resources available in libraries and in online databases because Web resources are networked, aggregated, heterogeneous, and available in multimedia formats. There is a vast array of digital data formats: text, hypertext, image, sound, video, animation, etc.

Though many Internet-enabled applications and services are available today, the primary use of the Internet (other than e-mail) is for IR. With such a diversity of content, and the

enormous volume of information on the Internet, retrieving *relevant* information is far from assured.

There are four different methods of locating information on the Web (Gordon & Pathak 1999: 142). First, you may go directly to a Web page simply by knowing its location. Second, the hypertext links emanating from a Web page provide built-in associations to other pages that its author considers providing related information. Third, "narrowcast" services can "push" pages at you that meet your particular user profile. Fourth, search engines allow users to state the kind of information they hope to find and then furnish information that hopefully relates to that description.

Studies of online searching behaviour have been documented for decades. The earliest end-user IRSs are online public access catalogues (OPACs), which are often free and available remotely (Wang et al 2000: 230). Researchers have devoted more than 20 years of effort to studying the searching behaviours of OPAC users. But, Borgman (1996) argues, "online catalogues continue to be difficult to use, because their design does not incorporate sufficient understanding of searching behaviour", and "we need to incorporate more knowledge of searching behaviour into the design of these systems." This discovery is of particular importance to the design of our solution, and we expect that by utilising a semantic network as our knowledge representation structure, we can replicate much of the users behaviour.

Earlier studies of IR focussed on systems and technologies, but a growing body of literature indicates a shift in research in this field (Wang et al 2000). Recently, more studies take user-oriented approaches such as sense-making, cognitive and behavioural approaches to investigate the complex nature of user's IR (Bates 1996; Borgman et al 1996).

The Web is increasingly an important channel for information dissemination and retrieval. Several studies of users and Web searching have focussed on search engines, interfaces, and successes (Chu & Rosenthal 1996; Ding and Marchionini 1996; Pollock and Hockley

1997; Schneiderman et al 1997). The findings have shown that various Web search engines function differently and suggest that users are not very successful at searching for information on the WWW. Wang and Pouchard (1997) analysed queries submitted to a university website and found that users had difficulties with the syntax and semantics of different search engines; more than 30% of the searches resulted in zero-hit outcomes. They conclude that improvement of search engines (automatic error correction and context-sensitive help) can eliminate some of the errors. In addition, the field of human-computer interaction (HCI) is also closely related to the study of user-Web interactions (Wang et al 2000).

In a comprehensive review, Bishop and Starr (1996: 361) concluded, "we need to understand more about which aspects of searching behaviour are universal and which are situation-specific, if we were to design information systems to serve an increasingly heterogeneous user population with increasingly diverse sets of information needs,"

3.4 Overview of Search Engines

It is fair to say that Internet-based information retrieval would collapse if search engines were not available; without search engines, searchers would be about as successful negotiating the Internet as someone trying to look up a phone number in an unsorted telephone directory (Gordon & Pathak 1999: 142). While word of mouth pointers to pages from friends, acquaintances and others are very useful, and the live hypertext links of the Web make it such a rich and convenient source of information, these means of negotiating the Internet do nothing for the user who does not even know where to begin looking: that is the job of search engines.

Search engines provide three chief facilities: (1) they gather together (conceptually) a set of web pages that form the universe from which a searcher can retrieve information. (2) They represent the pages in this universe in a fashion that attempts to capture their content.

(3) They allow searchers to issue queries, and they employ IR algorithms that attempt to find for them the most relevant pages from this universe. Search engines differ from each other along all of these dimensions.

A search engine can gather new pages for its universe in two ways (Gordon & Pathak 1999:143). First, individuals or companies who have created Web pages may directly contact the search engine to submit their new pages. Second, search engines employ Web "spiders", "crawlers", or "robots" and more recently "intelligent agents" or "knowbots", which traverse known Web pages, link by link, in search of new material.

Since computers cannot read and understand text, every page that a search engine might retrieve for a user must have its content represented in a way that a computer can process. There are two basic ways that web search engines can do this. First, the web page, or an abbreviated version of it, may be *indexed* – or represented – by the set of words or phrases it contains (excluding common "stop words" such as *the*, *of*, *for*, etc.). More sophisticated indexing techniques attempt to determine the *concepts* being used in a document, using statistical methods that correlate word and concept occurrences. The occurrence frequency of words (or phrases or concepts) is often maintained, as well as their location.

The second main way that Web pages are represented is by their position within a knowledge hierarchy developed and maintained by people. For instance, Yahoo, the best known subject directory, begin with fourteen very general subject categories, then continues to subdivide them until individual Web pages are identified.

Web-based IR also differs in some respects from more traditional IR. For instance, some search engines allow a user to restrict retrieval to certain domains (only *.com* sites) or specific domain names (such as *ibm.com*) or even to specify that all the pages he/she wants to see should have a *link* to ibm.com. Further, some search engines allow the user to specify, for example, that the pages he/she is interested in should contain a plug-in script, Java applet or embedded real audio file.

The matching algorithms that search engines employ also often embrace certain principles that apply to Web-based searching but not traditional IR. As an example, some search engines boost the relevance scores of pages that have many incoming links, the argument being that these are popular pages and so are the ones people will want to retrieve. Some search engines also reduce the relevance scores of certain pages for violating rules of fair play.

More recently, Crestani and Lee (2000: 587) developed and tested an IIR prototype system, called WebSCSA (Web search by Constrained Spreading Activation). They concluded that WebSCSA improves retrieval of *relevant* information on top of search engines and serves as an agent browser for the user. This area of research is very much in its infancy and research undertaken in Web IIR, particularly in the areas of Data Mining, Intelligent Agents and Multi-media retrieval will be dominant topics in this decade.

In short, search engines are indispensable for searching the Web, they employ a variety of advanced IR techniques, and there are some peculiar aspects of search engines that make searching the Web different than more conventional IR.

3.5 Networked Information Retrieval (NIR)

Networked information retrieval (NIR) deals with the issue of coupling different IR systems, each managing its own database of documents (Fuhr 1999: 101). This approach contrasts with centralised IR systems like, for example, Web search engines aiming at indexing almost any document publicly accessible through the Internet. Both strategies give the user the impression of a single large database that can be searched, but only NIR systems are able to protect the rights of the owners of the databases.

Major issues in NIR that are still receiving interest form researchers are the interoperability of different IR systems, resource discovery (i.e., given a query, select the relevant databases

to be searched) and data fusion (merging the answers from different databases into a single result list).

3.6 Natural Language Processing (NLP) in Information Retrieval (IR)

Natural language processing (NLP) techniques may hold a tremendous potential for overcoming the inadequacies of purely quantitative methods for text information retrieval, but the empirical evidence to support such predictions has thus far been inadequate (Carballo & Strzalkowski 2000: 155).

Recently, a renewed interesting using NLP techniques in IR, was sparked by the sudden prominence, as well as the perceived limitations, of existing IR technology in rapidly emerging commercial applications, including on the Internet.

In IR, a typical task is to fetch *relevant* documents from a large archive in response to a user's query and rank these documents according to relevance. Although many of the sophisticated search and matching methods are available, the fundamental problem remains to be an inadequate representation of content for both the documents and the queries.

NLP has always seemed to offer the key to building the ultimate IR system. Unfortunately, a direct application of NLP techniques to IR has met severe obstacles, chief among which was a lack of robustness and efficiency (Carballo & Strzalkowski 2000: 160). Furthermore, the difficulties did not rest with the linguistic processing itself but extended to the representation it produced: it was unclear how the complex structures could be effectively compared to determine relevance. A better approach was to use NLP to assist an IR system, whether Boolean, statistical, or probabilistic, in automatically selecting terms, words and phrases, which then could be used in representing documents for search purposes. This approach provides extra manoeuvrability for softening any inadequacies of the NLP software without incapacitating the entire system.

Although NLP techniques have not proven to be as successful as originally anticipated, researchers are pursuing other improvements such as information extraction techniques, etc. In addition, cross-language IR systems have also become increasingly prominent (Oard & Resnik 1999: 363). Research in these areas are very much at the fore in recent times and they present many challenges for the future.

3.7 Conclusion

In this Chapter, we presented a broad overview of the current IIR techniques employed by modern IIRSs. We learnt that despite the general utility and popularity of these tools, in many respects, their performance is mediocre (Croft & Belkin 2001). Most of these IR techniques address searches of full-text documents and their relevance factor is still low. There is no available evidence to show that these techniques will work on keyword searches, such as the one we require.

Furthermore, text search engines and agent-based filtering systems make mistakes that are obvious and aggravating to the users, and relevant documents are usually mixed with many others that are totally unrelated. These problems significantly lower the productivity and effectiveness of people using tools, whether in education, science, business, or government. We believe that the fundamental issue that underlies all of these problems is a lack of adequate models of the user and the domain. In order to achieve breakthroughs in retrieval and filtering accuracy, the tools need to be able to use more information about the context of the query, better models of the user, and more knowledge about the domain.

These modern techniques are not required at this stage for the problem at hand since the data from the NRF that we are dealing with is relatively structured. However, should the specifications of our problem domain change in the future, to accommodate aspects such as natural language handling, etc., we would be forced to pursue these newer approaches. For these reasons, we believe that an expert system approach, as described in Chapter Two

(section 2.3), is the most appropriate approach to pursue to resolve our original dichotomy. In the Chapters that follow, we will investigate what reasoning strategies and knowledge representation structures best suits our problem domain in pursuit of an expert system solution.

Chapter Four

The Reasoning Strategies and Knowledge Representation Structures of IIR

In the previous Chapters, we established a need for an automated solution in the form of an expert IIR system to partially solve our original problem. The central issues when developing an expert system is to decide on the most suitable reasoning strategy for the inference engine and, a compatible knowledge representation structure for the knowledge base. A decision as to which reasoning strategy or knowledge representation structure to adopt will not be made until we investigate the various reasoning strategies and knowledge representation structures currently available, which is the focus of this Chapter.

In this Chapter we give a brief overview of the two predominant reasoning strategies, namely backward chaining and forward chaining, usually used in expert IIR systems and, discuss their strengths and weaknesses. We will also highlight the various knowledge representation schemes, namely, rule-based representations, semantic networks and frames, currently being used for most IIR applications in our pursuit for one that best meets FUND's needs.

4.1 The Reasoning Techniques of IIR

As mentioned in section 2.3.3, once knowledge is represented in some form, we need a reasoning procedure to draw conclusions from the knowledge base. For if-then rules, that

form the cornerstones of most IIR systems, there are two broad categories of reasoning (Dennis 1989):

- Backward chaining (or goal-driven or top-down control), and
- Forward chaining (or data-driven or bottom-up control).

If-then rules form chains that go from left to right. The elements of the left-hand side of these chains are input information, while those on the right-hand side are derived information:

Input information → ... → derived information

These two kinds of information have a variety of names, depending on the context in which they are used. Input information can be called *data* (for example, measurement data) or *findings* or *manifestations* (Bratko 1994: 341). Derived information can be called *hypotheses* to be proved, or *causes* of manifestations, or *diagnoses*, or *explanations* that explain findings. So chains of inference steps connect various types of information, such as:

Data → ... → goals
Evidence → ... → hypotheses
Findings, observations → ... → explanations, diagnoses
Manifestations → ... → diagnoses, causes

There are many differences between the two reasoning strategies mentioned earlier, but the most significant is the direction of the search. We need to understand the differences between these basic reasoning mechanisms as well as the influence of uncertainty in IIR systems in order that we may develop the most suitable reasoning strategy for FUND.

4.1.1 Goal-driven Reasoning or Backward Chaining

Goal-driven reasoning, or backward chaining, is an efficient way to solve problems that can be modelled as "structured selection" problems. That is, the aim of the system is to pick the best choice from many enumerated possibilities. It is used as a method of control when there is limited number of goals that can be specified in advance (Hayes-Roth, Waterman & Lenat 1983). For example, an identification problem falls in this category. Diagnostic systems also fit this model, since the aim of the system is to pick the correct diagnosis.

Usually, the knowledge is structured in rules that describe how each of the possibilities might be selected. The rules break the problem into sub-problems – that is, it is a kind of divide-and-conquer approach.

4.1.2 Data-driven Reasoning or Forward Chaining

For many problems it is not possible to enumerate all of the possible answers beforehand and have the system select the correct one. For example, configuration problems fall into this category. These systems might put components in a computer, design circuit boards, or lay out office space. Since the inputs vary and can be combined in an almost infinite number of ways, the goal driven approach will not work.

Forward chaining is therefore used if there are a large number of possible conclusions or goal-states and no means of predicting in advance which is the most likely outcome.

The data driven approach, or forward chaining, uses rules similar to those used for backward chaining, however, the inference process is different. The system keeps track of the current state of problem solution and looks for rules which will move that state closer to a final solution.

Note that for data driven systems, the system must be initially populated with data, in contrast to the goal driven systems that gather data as it is required (Dennis 1989).

With regard to FUND, we need to determine which reasoning scheme is best suited to our problem domain and, to do this we need to compare both of them as well as investigate whether hybrid systems are possible. This will be accomplished in the next sub-section.

4.1.3 Forward Chaining vs. Backward Chaining

Both forward and backward chaining involve search, but they differ in the direction of the search. Backward chaining searches from goals to data, from diagnoses to findings, etc. In contrast, forward chaining searches from data to goals, from findings to explanations or diagnoses, etc.

An obvious question that arises is: Which is better, forward chaining or backward chaining – and why? The answer is very much dependent on the problem domain. If we want to check whether a particular hypothesis is true then it is more natural to chain backward, starting with the hypothesis in question. On the other hand, if there are many competing hypotheses, and there is no reason to start with one rather than the other, it may be better to chain forward.

In particular, forward chaining is more natural in monitoring tasks where the data are acquired continuously and the system has to detect whether an anomalous situation has arisen; a change in the input data can be propagated in the forward chaining fashion to see whether this change indicates some fault in the monitored process or a change in the performance level.

In choosing between forward or backward chaining, simply the shape of the rule network can also help (Bratko 1994: 342). If there are few data nodes (the left flank) of the network

and many goal nodes (right flank) then forward chaining looks more promising; if there are few goal nodes and many data nodes then vice versa.

Sometimes, a combination of chaining in both directions is required. In medicine, for example, some initial observations in the patient typically trigger doctor's reasoning in the forward direction to generate some initial hypothesised diagnosis. This initial hypothesis has to be confirmed or rejected by additional evidence, which is done in the backward chaining style.

Before we commit to a reasoning scheme for FUND, we need to also look at the element of uncertainty associated with this process and what influence it has. This will be done in the next sub-section.

4.1.4 Uncertainty

As mentioned earlier (section 2.3.3), often in structured selection problems the final answer is not known with complete certainty. The expert's rule might be vague, and the user might be unsure of answers to questions. This can be easily seen in medical diagnostic systems where the expert is unable to be definite about the relationship between symptoms and diseases. In fact, the doctor might offer multiple possible diagnoses.

For expert systems to work in the real world they must be able to deal with uncertainty. One of the simplest schemes is to associate a numeric value with each piece of information (Dennis 1989). The numeric value represents the certainty with which the information is known. There are numerous ways in which these numbers can be defined, and how they are combined during the inference process. Most prototypes however, tend to assume certainty of its data, and certainty factors are usually introduced when the system is developed as a real-life application system.

There are also many domains however, in which assuming certainty is neither possible nor realistic. An expert diagnostic system, for example, needs to reflect the uncertainty of the diagnosis and the way that additional pieces of evidence support or contradict particular hypotheses. This is rather like, finding out from a patient if he is allergic to penicillin *before* administering any medication to him containing it.

Inclusion of facilities for handling imprecise rules or data adds a layer of complexity to the system design, and as mentioned earlier, some expert systems, particularly during the early prototype stages, bypass this problem by assuming certainty of both knowledge and data. The question of whether to cater for uncertainty or not is very much dependent on its influence on the problem domain, that is, how critical is it in terms of the returns of the actual system. As far as FUND is concerned, it is not that influential on the output and therefore can be bypassed at this stage. However, catering for uncertainty within FUND is a viable option to pursue as a future research effort (section 8.3), particularly if the system is to be developed as a real life application.

4.1.5 FUND's Reasoning Strategy

The question now becomes, what reasoning strategy do we adopt for FUND. The answer as pointed out earlier (section 4.1.3), is largely dependent on the nature of the problem domain and is also influenced by the choice of the programming language to be used. Sometimes, the programming language we choose may have a default reasoning strategy that will suit our problem domain and hence, no extra coding will be required. Therefore, we will decide on the reasoning strategy (section 7.1) after choosing the programming language (section 6.2). An even more critical decision than the reasoning strategy for FUND is the decision of which knowledge representation scheme best suits the problem domain and this will be addressed in the next section.

4.2 Knowledge Representation Structures

As mentioned above, in conjunction with the dilemma of deciding which reasoning strategy to employ, we are faced with an even greater decision, namely, how best can we represent the data, that is, what knowledge representation scheme would best suit the problem domain.

Knowledge representation approaches may be sub-divided into *declarative* (meaning the representation of facts and assertions) and *procedural* (meaning the storing of actions and consequences) approaches (Robert 1990: 26 - 27). An alternative characterization of this dichotomy is *knowing what* (declarative) versus *knowing how* (procedural). At this time AI research is divided more or less equally between these approaches. In addition, the dichotomy between procedural and declarative representations is no longer considered as clear cut as it once was. The specifics of an application may dictate a preferred approach. A reasonable postulate is that versatile, intelligent systems may need to use both procedural and declarative representations.

Declarative schemes include logic and relational approaches. Relational models may lead to representations in the form of trees, graphs, or semantic networks. Logical representation schemes include the use of prepositional logic and, more importantly, predicate logic.

Procedural models and representational schemes store knowledge in *how* to do things. They may be characterized by formal grammars and usually implemented via *procedural* or *rule-based (production) systems.* These are often structured as IF (condition) THEN (action).

The difference between declarative and procedural schemes may be illustrated by a simple example. A table of logarithms is an *explicit enumeration* of this (numerical) domain knowledge and would be considered a *declarative representation.* On the other hand, a

stored sequence of *actions*, like those stored within calculators, indicating *how to compute* the logarithm of a number might be considered a *procedural representation*.

The preceding example illustrates the trade-offs in using declarative or procedural knowledge representations. Declarative representations are usually more expansive (and expensive), in the sense that enumeration may be redundant and inefficient. However, modification of declarative representations is usually quite easy; one merely adds or deletes the knowledge. Procedural representations, on the other hand, may be more compact, at the expense of flexibility. Practical representations may include both declarative and procedural elements. For example, in subsequent Chapters we consider a knowledge base for FUND consisting of a *list of facts* (declarative) and a related *set of rules* (procedural) to manipulate the fact list.

Having decided that FUND will incorporate both a declarative and procedural approach, we now look at what properties FUND's knowledge representation scheme should possess before we can decide on the exact implementation scheme. A good system for the representation of knowledge in a particular domain should ideally possess the following four properties (Elaine & Kevin 1991: 109 - 110):

- *Representational Adequacy* – the ability to represent all of the kinds of knowledge that are needed in that domain.
- *Inferential Adequacy* – the ability to manipulate the representational structures in such a way as to derive new structures corresponding to new knowledge inferred from old.
- *Inferential Efficiency* - the ability to incorporate into the knowledge structure additional information that can be used to focus the attention of the inference mechanisms in the most promising directions.
- *Acquisitional Efficiency* – the ability to acquire new information easily. The simplest case involves direct insertion, by a person, of new knowledge into the

database. Ideally, the program itself would be able to control knowledge acquisition.

Unfortunately, no single system that optimizes all of the capabilities for all kinds of knowledge has yet been found. As a result, multiple techniques for knowledge representation exist. As far as FUND is concerned, we will attempt to incorporate as many of these properties as possible.

As mentioned above, artificial intelligence systems employ a variety of formalisms for representing knowledge and reasoning with it. There are however, two basic approaches to this problem that reflect different views of IIR systems. At one extreme are purely *syntactic systems,* also called logic systems, in which no concern is given to the meaning of the knowledge that is being represented. Such systems have simple, uniform rules for manipulating the representation. They do not care what information the representation contains.

At the other extreme are purely *semantic systems*, also called slot-and-filler structures, in which there is no unified form. Every aspect of the representation corresponds to a different piece of information, and the inference rules are correspondingly complicated (Elaine & Kevin 1991).

Once again we need to make a decision as to which knowledge representation scheme best suits our problem domain. In order to accomplish this, we will look at the knowledge representation techniques that have been used successfully for similar problem domains. In the sections that follow, we will discuss one form of syntactic representation, viz., rule-based or production rule representations and two forms of semantic representations, viz., semantic networks and frames.

4.2.1 Rule-based Representation

The language of *if-then* rules, also called *production rules*, is by far the most popular formalism for representing knowledge. In general, such rules are conditional statements, but they can have various interpretations. Examples are:

- *If* precondition P *then* conclusion C
- *If* situation S *then* action A
- *If* conditions C1 and C2 hold *then* condition C does not hold.

If–then rules usually turn out to be a natural form of expressing knowledge, and have the following additional desirable features:

- *Modularity*: each rule defines a small, relatively independent piece of knowledge.
- *Incrementability*: new rules can be added to the knowledge base relatively independently of other rules.
- *Modifiability*: old rules can be changed relatively independently of other rules.
- Support system's *transparency*. By transparency of the system, we mean the system's ability to explain its decisions and solutions.

Production rule systems are primarily syntactic. The interpreters for these systems usually only use syntactic information (such as the form of the pattern on the left hand side, the position of the rule in the knowledge base, or the position of the matched object in short-term memory) to decide which rules to trigger. Again here we see the similarity between logic and production rules as ways of representing and using knowledge. But it is possible to build production-rule systems that have more semantics embedded in them (Elaine & Kevin 1991). For example in EMYCIN and other systems that provide explicit support for certainty factors, the semantics of certainty factors are used by the rule interpreter to guide its behaviour.

In general, syntactic representations are to knowledge representation what weak methods are to problem solving. They are, in principle, adequate for any problem. But for difficult problems, their generality often means that answers cannot be found quickly.

Stronger, more semantically oriented approaches make it possible to use knowledge more effectively to guide search. This does not mean that there is no place for weak or syntactic methods. Sometimes they are adequate, and their simplicity makes a formal analysis of programs that use them much more straightforward than a comparable analysis of a program based on semantic methods. But powerful programs depend on powerful knowledge, some of which is typically embedded in their problem solving procedures and some of which is embedded in their knowledge representation mechanisms.

In the next two sections, we look at two other forms for representing knowledge via slot-and-filler structures, namely, *semantic networks* and *frames*. These differ from rule-based representations in that they are directed to representing, in a structured way, large sets of facts. The sets of facts are structured and possibly compressed. These facts can be abstracted away when they can be reconstructed through inference.

4.2.2 Semantic Network Representation

"Semantic nets, as their name implies, are designed to capture semantic relationships among entities, and they are usually employed with a set of inference rules that have been specially designed to correctly handle the specific types of arcs present in the network" (Elaine & Kevin 1991: 298). Semantic networks are basically graphic depictions of knowledge that show hierarchical relationships between object and may be successfully used as a knowledge representation scheme for an expert IIR system (Efraim 1992, George & William 1997).

In essence, a semantic network consists of entities and relations between the entities. It is customary to represent a semantic network as a graph. There are various types of semantic networks with various conventions, but usually nodes of the graph correspond to entities, while relations are shown as links labeled by the names of relations.

A network of this kind can be very easily translated into facts, such as those found in Prolog and other AI programming languages. In addition to these facts, which are explicitly stated, some other facts can be inferred from the network. Ways of inferring other facts are built into a semantic network type representation as part of the representation. A typical built-in principle of inference is *inheritance* (Bratko 1994: 349).

We will revisit semantic networks in the next Chapter (section 5.6), as it demonstrates promising characteristics as a possible knowledge representation structure for FUND that will form the backbone of its design.

4.2.3 Frame-based Representation

In frame representation, facts are clustered around objects. "Objects" here means either a concrete physical object or a more abstract concept, such as a class of objects (Bratko 1994: 350). Good candidates for representation by frames are, for example, the typical meeting situation or game conflict situation, or any attribute-classification system. Such situations have, in general, some common stereotype structure that can be filled with details of a particular situation.

A *frame* is a data structure whose components are called *slots*. Slots have names and accommodate information of various kinds. So, in slots, we can find simple values, references to other frames or even procedures that can compute the slot value from other information. A slot may also be left unfilled. Unfilled slots can be filled through inference. As in semantic networks, the most common principle of inference is inheritance. When a

frame represents a class of objects and another frame represents a super-class of this class, then the class frame can inherit values from the super-class frame.

What about the situation when a frame has more than one parent frame, that is, when multiple-inheritance is encountered? An inherited slot value may potentially come from more than one parent frame and the question of which one to adopt arises. To resolve this conflict one has to include other strategies or tie-breaking rules (Bratko 1994: 354).

Frame-based systems are more highly structured than semantic nets, and they contain an even larger set of specialised inference rules, including those that implement a whole array of default inheritance rules, as well as other procedures such as consistency checking and conflict resolution. In the section that follows, we attempt to resolve the dilemma of which technique best suits our problem domain.

4.2.4 Logic vs. Slot-and-filler Structures

Slot-and-filler structures have proven very valuable in the efficient storage and retrieval of knowledge for AI programs. They are usually poor, however, when it comes to representing rule-like assertions of the form "If x, y and z, then conclude w." Predicate logic, on the other hand, does a reasonable job of representing such assertions, although general reasoning using these assertions is inefficient. Slot-and-filler representations are usually more semantic, meaning that their reasoning procedures are more varied, more efficient, and tied more closely to specific types of knowledge (Elaine & Kevin 1991).

Hayes (1973) and Nilsson (1980) showed that slot-and-filler structures could be translated into predicate logic. In practical terms, however, moving to logic means losing efficiency. Part of the inefficiency of general reasoning methods like resolution, can be overcome by intelligent indexing techniques, but the more heavily cross-indexed predicate logic clauses are, the more they come to resemble slot-and-filler structures.

On the other hand, it is difficult to express assertions more complex than inheritance in slot-and-filler structures. The most logical question now becomes: Is it possible to create a *hybrid* representational structure that combines the advantages of slot-and-filler structures with the advantages of predicate logic? One such system (CYC) that already exists, maintains both a logical (epistemological level) and frame-based (heuristic level) version of each fact to encode specific types of knowledge and inference aimed at common-sense reasoning (Lenat, Prakash & shepherd 1986: 65 - 85). Another system, called KRYPTON (Nilsson 1980), divides its knowledge into two distinct repositories, called the *TBox* and the *ABox*. The *TBox* is a slot-and-filler structure that contains terminological information. The *ABox* contains logical assertions. The atomic predicates used in *ABox* assertions refer to concepts defined in the *TBox*.

4.2.5 Summary of the Role of Knowledge

To sum up our treatment of knowledge representation in IIR systems, we look at the two roles that knowledge can play in those programs (Elaine & Kevin 1991: 302):

- It may define the search space and the criteria for determining a solution to a problem. This knowledge is called *essential knowledge*.
- It may improve the efficiency of a reasoning procedure by informing that procedure of the best places to look for a solution. This is called *heuristic knowledge*.

In formal tasks, such as theorem proving and game playing, there is only a small amount of essential knowledge and a large amount of heuristic knowledge needed. When tackling naturally occurring problems however, like medical diagnosis, natural language processing or even searching for funding sources, substantial bodies of both essential knowledge and heuristic knowledge are absolutely necessary. We therefore conclude that both *essential knowledge* and *heuristic knowledge* are required within FUND. Although the nature of the problem domain itself will, to a large extent, determine which knowledge representation

scheme to adopt, it is acceptable to assume that the knowledge representation structure chosen should contain a reasonable amount of inheritable and inferential capabilities built in.

4.2.6 FUND's Knowledge Representation Structure

As pointed out in the previous section, FUND's knowledge representation structure must possess a reasonable amount of essential knowledge as well as some heuristic knowledge, that is, it must possess a reasonable amount of inferential capabilities built-in. However, we need to investigate the nature of the problem domain itself and look at other similar IIR systems before we decide on the exact knowledge representation structure for FUND. Thus far, semantic networks show the greatest promise, but we need to investigate its efficacy in other existing systems before we implement it in FUND. This will be accomplished in the next Chapter.

4.3 Conclusion

In order for us to design FUND, we need to make two decisions. Firstly, we need to decide on an appropriate reasoning strategy, either forward-chaining or backward-chaining. Secondly, we need to decide on a knowledge representation scheme that is capable of making inferences. Both these decisions are influenced by the nature of the problem domain and the programming language we choose. These issues, coupled with the uncertainty element, can only be resolved once we have investigated other similar IIR systems. In the Chapter that follows, we will highlight the pros and cons of the knowledge representation structures of other existing IIR systems. Finally, we will review whether the knowledge representation scheme that demonstrates the greatest promise for our problem domain, namely semantic networks could assist us in alleviating our original problem.

Chapter Five

IIR Systems

In the previous Chapter we developed a need for an appropriate knowledge representation structure that possessed inferential capabilities and an appropriate reasoning strategy for FUND. In this Chapter we will begin by highlighting the advantages and shortcomings of existing techniques in IIR in the hope of extracting those features that will suit our problem domain. In particular, we hope to discover what makes an IIR system "intelligent" so that we may incorporate this characteristic within FUND. We will also examine the past and current state of affairs with regard to knowledge representation schemes in IIR and will review the one that demonstrates the greatest promise for our problem domain, namely semantic networks. Finally, we will culminate the Chapter by tracing the evolution of semantic networks from its beginnings in Associationist's Theories (George & William 1997) to its current status and highlight how it would best suit our problem domain.

5.1 Shortcomings of Existing Techniques

Researchers with an Artificial Intelligence background have often ignored some of the major results from IR research (Croft 1986: 205). Although some of the experiments and techniques that produced these results may have had serious flaws, research should at least be used as a basis for comparisons and to avoid repetition of mistakes. For this reason, it is worth outlining some of the shortfalls and most significant contributions made by past research (Croft 1987).

The definition of an expert system in Chapter Two (section 2.3) did not include the term *reason.* Unfortunately, human experts do not reach conclusions solely through the application of describable reasoning process. Many experts attribute their success in reaching correct conclusions to "gut feel" or "instinct." This implies a significant streak of luck, or, more likely, a subliminal reasoning process that is the result of significant experience and that defies apparent quantification. This helps to explain why most expert system applications have been restricted to a narrow range of problem areas that are quantified in a straightforward fashion. The apparent lack of success of expert systems in other areas, notably, stock market forecasting, detective (criminal investigation) work, and "spread betting", provides a challenge for the future.

One might expect the performance of expert systems, which could tirelessly and exhaustively consider every possibility associated with a problem domain, to outperform humans in a wide spectrum of applications. However, this is not the issue. The critical issue is not the volume of returns but rather the degree of relevancy. Expert system developers have discovered that knowledge acquisition can be slow, expensive and usually possess a low degree of relevancy primarily because of the lack of reasoning capabilities within the knowledge base. Furthermore, systems tend to be "brittle" in the sense that slight modifications in the application lead to unacceptable deviations in expert system performance.

It is not incidental that a human spends approximately 12 years past the age of 5 (or thereabouts) in formal schooling. Notwithstanding the possible lack of efficiency in this process, a significant amount of both *information* and *experience* (that is perhaps not as easily quantifiable) is gained over this time interval. In addition, most experts have a considerable amount of additional informal and formal education past this point. Therefore, we should not be surprised at the practical difficulty of representing expert behaviour (Robert 1990: 364).

A strong theoretical framework has been built up in IR around the probabilistic model of retrieval (Van Rijsbergen 1983; Bookstein 1985: 121). This model uses a representation of document and queries based on sets of weighted features or terms. The features and the weights are both derived by Natural Language Processing (NLP). Typically, this processing is very simple, identifying important single word stems and weights that reflect the frequency of occurrence of a stem in the document text and in the collection as a whole. More complex features, such as phrases, etc., have been incorporated into the same model and many web agents presently still use these techniques. However, research shows that these techniques often tend to reveal high volume of hits and not necessarily *relevant* links, primarily because of its semantic deficiency.

The Intelligent Intermediary Information Retrieval (I^3R) systems (Croft & Thompson 1987), such as IOTA described by Chiaramella and Defude, and CODER described by Fox (1987: 341 - 366), had as their central theme an expert intermediary system that assisted the user with query formulation or reformulation, as well as the search strategy selection and evaluation of retrieved documents. Research by Fox revealed certain open problems in this area, namely (Fox 1987):

- Conventional techniques involving test collections are simply not appropriate for the evaluation of the complex interactive systems that result from this approach.
- Difficulties related to the formalisation of the knowledge used by a human intermediary.
- Designing of efficient access structures to support multiple search strategies and representations.

With regard to IR and knowledge representation, two systems, viz. GRANT (Cohen & Kjeldsen 1987; Kjeldsen & Cohen 1987) and RUBRIC (Croft 1986; McCune, Tong, Dean

& Shapiro 1985) also revealed some difficulties, namely (Croft 1987):

- The acquisition of domain knowledge from the users and/or document texts was an extremely difficult problem that depended largely on developments in NLP.
- There was no clarity with regard to the superiority of representation schemes appropriate for text retrieval.
- In addition, there was no clarity as to the level of domain knowledge that is required for effective IR.

With regard to IR and NLP, projects such as ADRENAL (Croft & Lewis 1987), described by Croft and Lewis, performed detailed analysis of the request text, with user interaction providing additional domain knowledge. Shortfalls in this area include (Croft 1987: 250):

- Current systems make use of thesaurus knowledge, and general dictionaries. However, many documents will contain words that do not fall in either of these categories. It is not clear how much these unknown words will affect the NLP required for IR.
- The level of NLP required for effective document retrieval was problematic. The comparison here is between simple, syntax-based techniques, such as pattern matching of word categories in the text, and more complex techniques, such as conceptual analysis, that produce representations of the "deep structure" of the text.
- Lastly, problems related to the use of NLP to build knowledge bases.

Finally, the research under way in the area of IIR, although laden with pitfalls, has considerable potential. As new systems and ideas are developed however, it is important to insist, as much as possible, on a continuation of the IR traditions of formal evaluation models. Implementations of IIR systems, whether "intelligent" or not, should address the important issues of IR, which include the incorporation of inferential reasoning capabilities into the knowledge base, very much like those found within semantic networks. In the next

section, we expand on the introductory remarks made earlier concerning the real meaning of "intelligence" in IIR systems as well as its impact on IR systems.

5.2 Why "intelligent" Searching?

At this point, we need to justify why a straightforward IR system will not suffice for FUND, and what aspects reflect intelligence within an IIR system. To accomplish this, we need to look at what role "intelligence" plays in an IIR system and where it can be found.

Nicholas J. Belkin (1983), in his article entitled "Intelligent Information Retrieval: Whose intelligence?" attempts to show us that the naïve concept of Intelligent Information Retrieval (IIR), based on the idea of "intelligent agents," misses the essence of intelligence in the IIR system, and will inevitably lead to dysfunctional information retrieval. As a counter proposal, he suggests that true intelligence in IR resides in appropriate allocation of responsibility amongst *all* the "actors" in the IR system, and that IIR will be achieved through effective support of people in their various interactions with information.

IIR systems has been defined differently by various people but a consistent theme has been one of the machine (or program) doing something for the user, or the machine (or program) taking over some functions that previously had to be performed by humans. For instance, for Belkin, an IIR system was one for which the functions of a human intermediary were performed by a program interacting with the user. For Maes, on the other hand, IIR is performed by a computer program (an intelligent agent), which, acting on, (perhaps minimal or no explicit) instructions from the user, retrieves and presents information to the user without any other interaction.

Croft (1987) suggests that IIR is "good" IR, meaning that it is inappropriate to ascribe intelligence to a computer program, and also meaning that good IR is that in which the

programs (i.e., the representations, comparisons and interaction methods implemented in the system) results in effective performance.

In all of these constructions, there is some idea that the intelligence (or goodness) of an IIR system resides in the *built* system. Most people that have commented on IIR systems assumed that the IR system referred only to the *built* system. However, proponents such as Belkin, Vickery, Bates and Ingwersen argue that this assumption is erroneous and suggest that both the interaction with information and also how much, and what kind of support to offer users for such interaction, is the key to understanding intelligence in IR (Belkin et al 1983; Belkin, 1980).

Nevertheless while we tend to concur with Belkin et al, we feel that FUND should possess a degree of intelligence in both its knowledge representation structure as well as its reasoning strategy if it is required to produce more effective results than a straightforward IR system. Although interaction with the information and the user are both important, we feel that since FUND is a prototype only, this issue is only worth pursuing as a future research effort when a greater detail is given to the human-computer interaction (HCI), that is, the user interface. For now, we will be content with building in intelligence only into FUND's knowledge base and reasoning scheme. To do this, it is important that we review the pros and cons of past and existing IIR systems in general before we decide on how to incorporate a degree of intelligence within FUND's knowledge representation structure and reasoning strategy (section 6.3).

5.3 The Advantages and Disadvantages of IIR

From section 2.1, it is evident that the traditional information retrieval systems based on Boolean logic suffer from two inherent problems: (1) inaccurate or incomplete query representation, and (2) inconsistent indexing.

Conversely, IIR systems seek to address both these problems with many documented successes. Most of them have at the outset an advanced query reformulation procedure that moulds the request into a structured format that the system can handle and process – even in the case of uncertainty.

Spark Jones (1983) has defined an intelligent IR system as a computer system with inferential capabilities such that it can use prior knowledge to establish, by plausible reasoning, a connection between a user's probably ill-specified request and a candidate set of relevant documents. This definition, while providing a useful starting point, can be expanded to include notions developed with respect to intelligent interfaces for text retrieval systems and with respect to the information-seeking behaviour of individuals (Belkin, Seeger & Wersig 1983; Brooks, Daniel & Belkin 1985; Belkin 1980; Belkin, Oddy & Brooks 1982). These ideas can be combined to produce a view of what an intelligent IR system should be like and what it should be able to do.

An intelligent IR system is a computer system that carries out intelligent retrieval. Intelligent retrieval pre-conditions can be defined as follows (Brooks 1987: 369):

> Given: A particular individual (or user) who has come to the information service with a *problem* for whose resolution or better management is required.
>
> Given: That the user will be unable to specify the information he or she needs, since this involves describing exactly that which the user does not know.
>
> Given: Access to one or more "databases" of document descriptions.

This definition of intelligent pre-conditions retrieval involves the use by a computer system of stored *knowledge* of its "world" (text, users, topics, etc.) and of information about the *user* and his/her *problem* to *infer* which documents would enable that particular user to resolve or better manage his/her problem. This is very much in keeping with the issues raised by Belkin (1982 & 1983) in the previous section.

With regard to this definition, to achieve the last given requires that an IIR system should posses some of the following characteristics:

- Can take into account the characteristics of individual users.
- Can deal with users' problems at the level of problem description (and not require users to make specific requests that involve describing the texts needed).
- Can take into account such aspects as the state of the problem, the capabilities of the system, and the kind of response(s) required.
- Has a sufficiently complex human-computer interface to be able to interact adequately with the user.
- Is able to relate texts to their descriptions.
- Stores knowledge of its world (texts – their structure and contents, users and their problems, concepts, the subject domain of texts and problems) and make use of it in the retrieval process.

The definition of intelligent IR systems outlined above contains some rather crucial shifts from emphasis from traditional IR work, in which the main concern has been to improve retrieval effectiveness through the development of statistical techniques (Van Rijsbergen 1986). First there is the central role of the user and the need to give due consideration to the human-computer interface (Fox 1987; Croft 1986; Van Rijsbergen 1986). The next area of differentiation was to enhance the inherent inference capabilities of the knowledge base itself. Lastly, there was a shift in the form of the inference engine that was adapted to incorporate human traits, such as accommodating an Associationist's perspective (Minsky 1975). Although all of these areas are equally important they are particularly important as far as document retrieval is concerned, especially using Natural Language Processing (NLP). Document retrieval itself is too large an area and is highly problematic. Therefore, as far as our prototype FUND is concerned only the last two areas are pivotal to its success and, warrants our concern. However, the issue concerning the user and the role of the HCI is a worthy investigation for future research efforts (section 8.3).

In the section that follows, we will review those applications, in particular with regard to their knowledge representation structures and reasoning schemes, in the field of IIR that have been successful and highlight their contributions and shortcomings that may be crucial to FUND.

5.4 Successful Applications of IIR

From the late 1960's onwards, various teams of researchers in AI have been concerned with developing intelligent knowledge-based systems for dealing with a limited range of problems, which are sufficiently nontrivial to require a solution normally by a human expert. The problem areas were very small and highly specific, but, on the whole, had real-world applicability, for example, medical diagnosis, computer fault-finding, chemical structure elucidation, etc. These systems were dubbed the pioneers in IIR systems. In this section, we give a brief review of some of these systems to illustrate the range of problems tackled as well as their shortcomings and their successes.

The DENDRAL (Buchanan & Feigenbaum 1978; Lindsay, Buchanan, Feigenbaum & Lederberg 1980) project was one of the longest running projects. It was initiated in 1965, as a collaborative project by the Stamford Mass Spectrometry Laboratory and the Stamford Heuristic Programming (SHP) project. DENDRAL infers plausible molecular structures of an unknown compound using data from mass spectroscopy and nuclear magnetic resonance analysis in conjunction with heuristic knowledge similar to that used by human chemists performing the same task. Its generator could enumerate every plausible organic structure that satisfied the constraints apparent in the data. Exponential search was avoided by an attempt to eliminate implausible structures at an early stage. The systematic generation of all plausible structures for an unknown molecule meant that its performance was sometimes better than that of human experts, who may miss some candidate structures, and certainly a great deal faster.

DENDRAL embodies rules about the way in which molecules fragment during mass spectroscopy. The development of these rules was semi-automated by the use of another intelligent knowledge-based system, METADENDRAL (Buchanan & Feigenbaum 1978; Lindsay, Buchanan, Feigenbaum & Lederberg 1980), which was designed to infer sets of rules for the fragmentation of molecules during mass spectroscopy analysis, which could be later incorporated into DENDRAL. It represents an early attempt at producing systems to automate the knowledge acquisition process. The DENDRAL \ METADENDRAL system had been actively used by chemists since it effectively automated a task that humans found time-consuming, tedious, and difficult to perform well (Feigenbaum, Buchanan & Lederberg 1971; Lindsay, Buchanan, Feigenbaum & Lederberg 1980).

Several IIR expert systems have been developed to tackle problems in the field of medical diagnosis and treatment. The most famous of these systems, and possibly the most famous IIR system of all, is MYCIN (Shortliffe 1976; Buchanan & Shortliffe 1984 a & b). The MYCIN project that began in the 1970s at Stanford University was an attempt to devise a system that could assist medical practitioners in the diagnosis of blood infections and the development of appropriate treatment plans. MYCIN conducts a consultation with a physician-user about a patient case, using data provided by the physician to diagnose and devise a treatment plan. When a panel of experts evaluated the performance of MYCIN, in 90% of the cases submitted a majority of the judges said that the decisions made by the system were as good as or the same as the decisions they themselves would have made (Buchanan & Shortliffe 1984 a).

CASNET (Causal Association Network) (Weiss, Kulikowski & Safir 1978 a; Weiss, Kulikowski & Safir 1978 b; Kunz, Fallat, McClung, Osborn, Votteri, Nii, Aikins & Fagan 1978) is a medical diagnosis system developed at Rutgers University. Its main application was the diagnosis and treatment of glaucoma. Ophthalmologists have rated its performance as being close to expert. One interesting feature of CASNET is that it incorporates differing expert opinions that can be used to produce alternative diagnoses.

Other IIR expert systems applied to medical diagnosis and treatment include PUFF, which diagnoses pulmonary disorders (Kunz, Fallat, McClung, Osborn, Votteri, Nii, Aikins & Fagan 1978); PIP that diagnose renal diseases (Pauker, Gorry, Kassirer & Schwartz 1976); and DIGITALIS, which is concerned with treatment recommendations rather than diagnoses (Gorry, Silverman & Pauker 1978), inter alia.

The PROSPECTOR (Duda, Hart, Nilsson & Sutherland 1979; Gaschnig 1982) project began in 1976 at SRI (Stanford Research Institute) and was intended to assist geologists in evaluating the favorability of an exploration site for various types of ore deposits. The user provides the system with data about the rocks and minerals found near the site, as well as other observations. The program matches these data against its ore deposition model, requests any additional necessary information, and provides a summary of its findings and some conclusions or recommendations. The performance of PROSPECTOR was, of course, only as good as the model of ore deposition it embodied. Each ore deposition model was encoded as a separate data structure so that theoretically, a wide range of ore deposition models may be used.

One expert system, in daily use until recently, in an industrial environment was R1, or XCON. Developed at Carnegie-Mellon University in conjunction with Digital Equipment Corp. (DEC) Ltd. (McDermott 1981: 21 – 29; McDermott 1982: 39 - 88). This system was designed to assist in the configuration of VAX-11 series computers. XCON takes a computer purchase order, determines what (if anything), is missing or inappropriate and makes suitable substitutions; and successfully produces a set of diagrams showing the logical and spatial relations between the components ordered.

Probably the closest application to FUND would be the GRANT system developed by Cohen and Kjeldsen (1987: 255 - 268) at the University of Massachusetts in the US. They developed a semantic matching algorithm that finds matches between concepts based on semantic relations between their properties. This expert system was implemented for finding *likely* sources of funding for researchers based on semantic matches between

research proposals and descriptions of funding agencies' research priorities. GRANT was built primarily to explore the interpretation of "likelihood" in classification problems, problems where hypotheses are represented by typical cases, and are considered conditionally likely if data are a good "fit" or match to the hypotheses. In GRANT, the "hypotheses" are the descriptions of funding agencies, especially their research interests, and the data are the research topics in a grant proposal. The likelihood that an agency will fund a proposal depends on the degree of fit between their respective research interests. Syntactic or keyword matching is insufficient for this task for two reasons: Researchers and agencies may use different terms to describe their interests even though the terms may have the same or similar meanings, and interest in one research topic often implies interest in another slightly different but semantically related. The first problem was partly solved by using a large table of synonyms, but the second required knowledge of what the descriptions of research interests *mean*. GRANT searches in a large semantic network for agencies whose research interests are related to those of proposals. Its search is constrained by knowledge, under the guidance of a set of heuristic rules, about which semantic relations lead to agencies that are likely to match a proposal and thus fund the research.

Once GRANT has found an agency by this search process, it calculates a total degree of match between the agency and the proposal. A simple mechanism, based on tallying algorithms, compares all properties of the agency with all those of the proposal. Both common keywords *and* semantically related properties are included. The results are used to determine a ranking on all the agencies found during the search.

In summary, the classic IIR expert systems have shown that it is possible to design knowledge-based systems that can perform at performance levels approaching that to a human expert over a very limited domain. However, each such system represents an individual solution to a particular problem domain and problem type. There exists numerous other working IIR expert systems currently in operation, ranging from minimising wastage of raw materials – such as those used by hardware stores for making furniture – to elaborate systems that diagnose potential nuclear disasters in real-time.

From the literature surveyed in this and previous sections, we were able to extract those avenues that demonstrate the greatest promise in achieving our primary goals. We also discovered that document retrieval systems were too large an area and was highly problematic especially as regards natural language processing (NLP) (section 5.3) to assist us. Also, the inclusion of a highly interactive HCI is only required if the system is being developed as a real-life project.

A very important, if not the most important, realisation that we arrived at from this section was that our system, FUND, had to reflect the semantic association between the components of the knowledge base. Therefore, we anticipate that a *semantic network* will adequately handle the data within our problem domain as the knowledge representation structure for FUND because of its intelligent inferential capabilities, very much like GRANT does in the US. Furthermore, semantic networks allow for *inheritance* that is crucial to FUND. In the section that follows we will look at a school of thought in AI from which semantic networks originated, namely the Associationist's Theories, for a psychological justification of our approach to FUND.

5.5 The Associationist's Theories of Meaning

Associationist's theories define the meaning of an object in terms of a network of associations with other objects in a person's mind or a knowledge base (George & William 1987). Although symbols denote objects in a world, this denotation is mediated by our store of knowledge. When we perceive and reason about an object, that perception is first mapped into a concept in our minds. This concept is part of our entire knowledge of the world and is connected through appropriate relationships to other concepts. These relationships form an understanding of the properties and behavior of objects.

For example, through experience we associate the concept of *snow* with other concepts such as *cold*, *white*, *snowman*, *slippery*, and *ice*. Our understanding of snow and the truth of statements such as "snow is white" manifest itself out of this network of associations.

There is psychological evidence that in addition to our ability to associate concepts, humans also organize their knowledge hierarchically, with information kept at the highest appropriate levels of the taxonomy. Collins and Quillian (1968: 227 - 270) modeled human information storage and management using a semantic network. The structure of this hierarchy was derived from laboratory testing of human subjects. The subjects were asked questions about different properties of birds, such as, "Is a canary a bird?" or "Can a canary sing?" or "Can a canary fly?" As obvious as the answers to these questions may seem, reaction time studies indicated that it took longer for subjects to answer "Can a canary fly?" than it did to answer "Can a canary sing?"

Collins and Quillian (1968) explain this difference in response time by arguing that people store information at its most abstract level. Instead of trying to recall that canaries can fly, and robins can fly, and swallows can fly, all stored with the individual bird, humans remember that canaries are birds and that birds have (usually) the property of flying. Even more general properties such as eating, breathing, and moving are stored at the "animal" level, and so trying to recall whether a canary can breathe should take longer than recalling whether a canary can fly. This is, of course, because the humans must travel further up the hierarchy of memory structures to get the answer. This notion is called "inheritance" and is the basis of most human reasoning.

The fastest recall was for the traits specific to the bird, say, that it can sing or is yellow. Exception handling also seemed to be done at the most specific level. When subjects were asked whether an ostrich could fly, the answer was produced faster than when they were asked whether an ostrich could breathe. Thus the hierarchy ostrich → bird → animal seems not to be traversed to get the exception information: it is stored directly with ostrich. This knowledge organization has been formalized in inheritance systems.

Inheritance systems allow us to store information at the highest level of abstraction, which reduces the size of knowledge bases and helps prevent update inconsistencies (George, 1997). For example, if we were building a knowledge base about birds, we can define the traits common to all birds, such as flying or having feathers, for the general class *bird* and allow a particular species of bird to inherit these properties. This reduces the size of the knowledge base by requiring us to define these essential traits only once, rather than requiring their assertion for individuals. Inheritance also helps us to maintain the consistency of the knowledge base when adding new classes and individuals.

Assume that we are adding the species *robin* to an existing knowledge base of birds. When we assert that *robin* is a subclass of *songbird*; *robin* automatically inherits all of the common properties of both *songbirds* and *birds*. It is not up to the programmer to remember (or possibly forget) to add this information.

Graphs, by providing a means of explicitly representing relations using arcs and nodes, have proved to be an ideal vehicle for formalizing Associationist theories of knowledge. A *semantic network* (Stiles 1961: 271 - 279) represents knowledge as a graph, with the nodes corresponding to facts or concepts and the arcs to relations or associations between concepts. Both nodes and links are generally labeled.

For example, a general semantic network that defines the properties of *snow* and *ice*, in a manner consistent to human reasoning, appears in figure 5.1 below. Note, this form of notation used is consistent to human reasoning only and, a different set of notation will be adopted when semantic networks are being use in a computer science perspective.

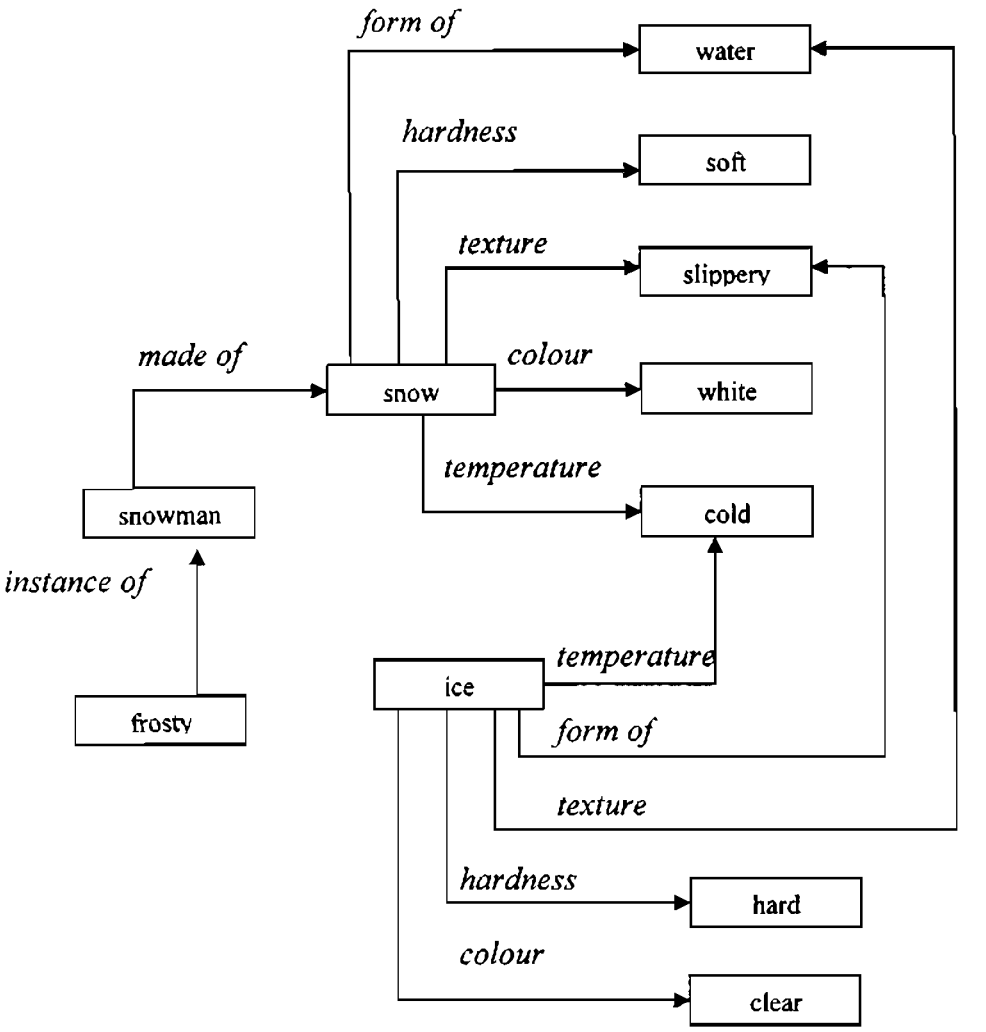

Fig. 5.1 Semantic network of snow and ice

The structure and symbols used in the generic semantic network of figure 5.1 differs from those used in a restrictive sense, as for example in figure 5.2, when it is used in a computer science perspective.

The network in figure 5.1could be used (with appropriate inference rules) to answer a range of questions about snow, ice, and snowmen. An inference could be made, by following the

appropriate links between related concepts. Semantic networks also implement *inheritance*; for example, *frosty* inherits all the properties of *snowmen*. The term "semantic network" encompasses a family of graph-related representations. These representations differ chiefly in the names that are allowed for nodes and links and the inferences that may be performed on these structures.

Semantic nets are sometimes referred to as *associative nets* because nodes are associated or related to other nodes. In fact, Quillian's original work (1968) modeled human memory as an associative net in which concepts were the nodes and links formed the connections between concepts.

According to this model, as one concept node is stimulated by reading words in a sentence, its links to other nodes are activated or triggered in a *spreading* pattern. If another node receives sufficient activations, the concept would be forced up into the conscious mind. For example, although you know thousands of words, you are only thinking of the specific words in this sentence as you read it. We must also bear in mind that one person's vocabulary and relational links may vastly differ from another one's and therefore, relying solely on one perspective would be profoundly disastrous where retrieval is concerned. In fact, it is this property that introduces the notion of subjectivity within an IIR system.

In the section that follows, we will elaborate on the importance of semantic networks from a computer science perspective and demonstrate how it could be used in FUND.

5.6 The Promise that Semantic Networks Show

As pointed out above, a *semantic network*, or *net*, is a classic Artificial Intelligence knowledge representation technique used for propositional information (Neil & Stillings 1987; Giarratano & Riley 1989: 72). A semantic net is often called a *propositional net*. A proposition is a statement that is either true or false, such as "all dogs are mammals" and "a

triangle has three sides." Propositions are a form of declarative knowledge because they state facts. In mathematics, a semantic net is a labeled, directed digraph. Because of these inherent properties of semantic networks, it makes it an excellent choice for a knowledge representation scheme for information retrieval.

Semantic nets were first developed for AI as a way of representing human memory and language understanding. As pointed out earlier, Quillian (1968) used semantic nets to analyse the meanings of words in sentences. Since then, semantic nets have been applied to many problems involving knowledge representation.

As was demonstrated in fig. 5.1, the structure of a generic semantic net is depicted graphically in terms of *nodes* and the *arcs* connecting them. Nodes are often referred as *objects* and arcs as *links* or *edges*. The links of a semantic net are used to express *relationships*.

Nodes are generally used to represent physical objects, concepts or situations. One of the most interesting and useful facts about a semantic network from a computer science perspective is that it can show *inheritance* (Efraim 1992). Since the semantic network is basically a hierarchy, the various characteristics of some nodes actually inherit the characteristics of others. As an example, consider the links in fig. 5.2 showing that Sam is a man and a man is, in turn, a human being. Here, Sam inherits the property of human being. We can ask the question, "Does Sam need food?" Because of the inheritance links, we can say that he needs food if human beings need food.

Figure 5.2 that follows illustrates a small semantic network with the structures and symbols traditionally used in computer science:

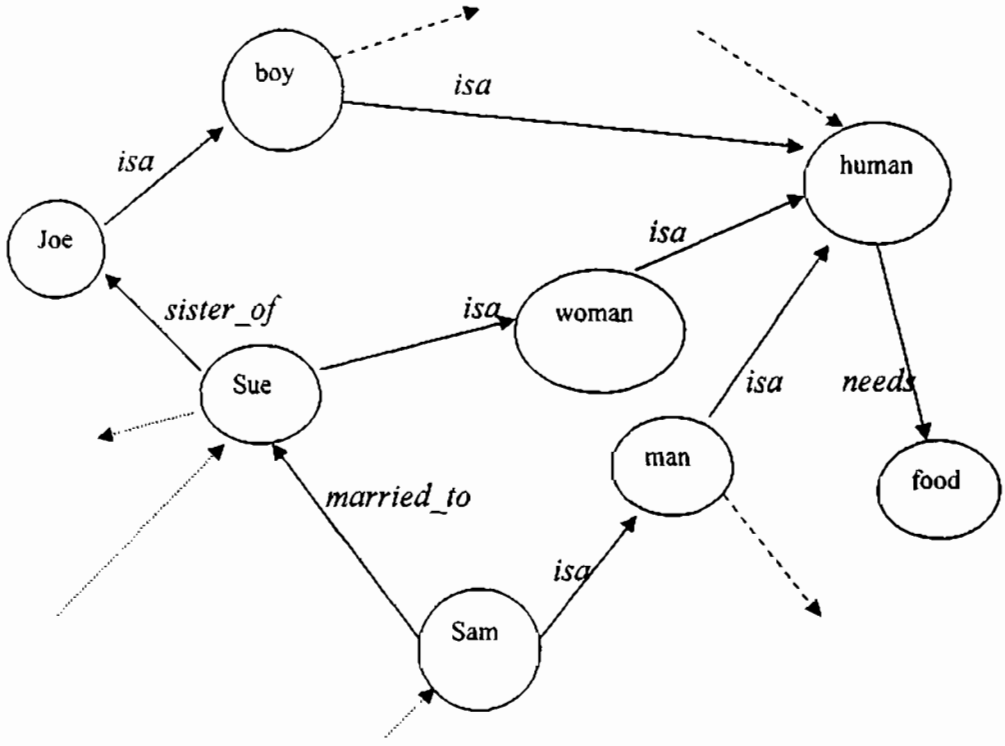

Fig. 5.2 Representation of knowledge in a semantic network

Relationships are of primary importance in semantic nets because they provide the basic structure for organising knowledge. Without relationships, knowledge is simply a collection of unrelated facts. With relationships, knowledge is a cohesive structure about which other knowledge can be inferred. It is this unique property of semantic networks that makes it an exceptional choice as the knowledge representation scheme for FUND, since inheritable relationships are pivotal to retrieving links that standard keyword searches would normally miss.

The major reasons for our choice of using semantic networks as our knowledge representation scheme for FUND are summarized below (Efraim 1992):

- ❖ The semantic network offers flexibility in adding new nodes and links to a definition as needed. The visual representation is easy to understand.
- ❖ The semantic network offers the economy of effort since a node can inherit characteristics from other nodes to which it has an **isa** or **ako** relationship.
- ❖ The semantic network functions in a manner similar to that of human information storage.
- ❖ Since nodes in semantic networks have the ability to inherit relationships from other nodes, a network can support the ability to reason and create definition statements between non-linked nodes.

Of course, the major setbacks or limitations of semantic networks (apart from subjectivity) include:

- ❖ No standard exists for the definition of nodes or relationships between and among nodes.
- ❖ The power of inheriting characteristics from one node to another offers potential difficulties with exceptions.
- ❖ The perception of the situation by the domain expert can place relevant facts at inappropriate points in the network.
- ❖ Procedural knowledge is difficult to represent in a semantic network, since sequence and time are not explicitly represented.
- ❖ The order of the links may differ for different contexts, especially where ambiguity creeps in.

Certain types of relationships have proven very useful in a wide variety of knowledge representations. Rather than defining new relationships for different problems, it is customary to use these customary types. The use of common types makes it easier for different people to understand an unfamiliar net (Giarratano & Riley 1989).

Two types of commonly used links are **IS-A** and **A-KIND-OF**, which are sometimes written as **isa** and **ako** (Winston 1984; Minsky 1975). The **isa** mean "is an instance of" and refers to a specific member of a class. A class is related to the mathematical concept of a set in that it refers to a collection of objects. While a set can have elements of any type, the objects in a class have some relation to one another. For example, it is possible to define a set consisting of

{3, eggs, blue, tyres, art}

However, members of this set have no common relationship. In contrast, the class consisting of planes, trains, and automobiles are related because they are all types of transportation.

The link **ako** is usually used to relate one class to another. The **ako** is not used to relate a specific individual because that is the function of **isa**. The **ako** relates an individual class to a parent class of classes of which the individual is a child class. From another viewpoint, the **ako** relates generic nodes to generic nodes while **isa** relates an instance or individual to a generic class.

Finally, we can justifiably say that a semantic network is the knowledge representation structure that best suits our problem domain and its inheritable characteristic will assist in introducing an inferential capability that is lacking in others.

5.7 Discussion and Conclusion

Before we render any concluding remarks from this Chapter, we need to summarise some of the salient points that have emerged with regard to our project from this and the previous Chapter. The first is that the knowledge representation scheme to be adopted for FUND

has to be inclusive of both essential knowledge and heuristic knowledge (section 4.2.5). Secondly, of the two slot-and-filler knowledge representation schemes discussed earlier (sections 4.2.2 and 4.2.3), namely semantic networks and frame-based schemes, it was found that frames contained much more information than was necessary for our project than semantic networks. Semantic networks possessed the inferential capabilities that logic systems lacked, and was sufficient to handle the data within FUND as a potential knowledge representation structure. We also discovered that FUND could not be moulded out of the traditional document retrieval systems because they were usually very large and, more often than not, incorporated a component of Natural Language Processing (NLP) that increased the complexity of the system (section 5.3). Finally, although the Human-Computer Interface (HCI) aspect of a real-world IIR system was found to be an essential component as far as the "intelligence" of the system was concerned, FUND will not incorporate this aspect as it is merely a prototype (section 5.3). However, this issue presents a worthy area of interest for any future research effort (section 8.3).

The essence of this Chapter was to choose an appropriate knowledge representation and a reasoning strategy for FUND. At this stage we are only able to decide on the knowledge representation structure. At the beginning of this Chapter it was established that the knowledge representation structure should possess an "intelligence" component that catered for the association of concepts so that inferences could be made. A semantic network possesses all of these characteristics, including inheritance and therefore, was chosen as the knowledge representation structure that best suited FUND.

With regard to the reasoning strategy to be adopted, no decision could be taken before we investigated which programming language would best suit the problem domain and the knowledge representation structure chosen. This can only be seen from the perspective of our solution of FUND whose design will be the focus of attention in the next Chapter. In particular, we hope that the nature of the problem domain and the choice of programming language would be able to shed more light on our choice of a reasoning strategy complementing the semantic network for FUND.

Chapter Six

Design of FUND

In the previous Chapter we settled on the knowledge representation structure for FUND, namely, a semantic network, but not the programming language and reasoning structure. These choices are closely coupled to the problem domain and therefore have to be discussed in relation to the detailed design of the project, which is the main focus of this Chapter.

We will begin the Chapter by revisiting the NRF problem and, outlining and justifying the reasons why we chose Prolog (LPA Prolog for Windows) as our programming language for FUND. We then present the finer details of the project in the form of our proposed solution to the problem. In keeping with the three-tier architecture for expert systems we discussed in Chapter Two (section 2.2.1), in this Chapter we will also present the various components that constitute FUND, namely, its knowledge base, its inference engine, and its user interface.

We will then depict a general search strategy for our reasoning scheme, namely Spreading Activation (SA), that is suited to semantic networks and have been successfully implemented in similar problem domains. Lastly, we investigate whether *Constrained* Spreading Activation (CSA) that was successfully implemented for GRANT (Cohen & Kjeldsen 1987), could assist in pruning the searches within FUND.

6.1 The NRF Problem Revisited

As mentioned in Chapter One (section 1.1), the fundamental problem lies with the task of matching prospective researchers to appropriate funding sources available from the NRF, which is currently being done by a human expert. Our task is to determine whether an AI search technique is a viable option to alleviate this problem by providing better funding advice to prospective researchers. To accomplish this we need to implement an appropriate search strategy over a semantic network in order to retrieve matches that normal keyword searches would usually miss. We do this by setting up a knowledge base of facts (essential knowledge) and relationships (heuristic knowledge) that would be semantically dependent, even if they were remotely related to each other. In this way, even if there are no direct matches between the researcher's proposal and the flat database of facts, there would be a possibility of finding matches that are *closely related* to the researcher's proposal. Therefore, by these inheritable inferences, the researcher could apply for a bursary that he or she otherwise would not have been able to.

Before we discuss the various components of the project itself, we need to elaborate on the features of Prolog that make it a viable programming language for FUND.

6.2 The Choice of Software to be Used

The choice of the programming language to use to write an expert system is dependent on a number of factors and also depends largely on the nature of the problem domain. To write an expert system, we must describe what we know about the domain as facts and the relationship between these facts. The computer has the job of both finding a procedure to solve the problem and, solving it. The computer primarily uses formal reasoning to accomplish this. As far as FUND is concerned, the choice of programming language is

further influenced by the choice of the knowledge representation structure, namely a semantic network.

Expert systems, in contrast with traditional systems, are more often developed using *data-driven* languages. These languages make no distinction between program and data, and are capable of symbolic processing. The data about the domain determines how the problem will be solved. Lisp and Prolog are examples of data-driven languages. Our project will be written using LPA Prolog (a commercial version). Apart from the inherent features of Prolog, one of the main reasons for choosing it as our programming language is that it is a natural choice for the implementation of a semantic network. Before we justify the other reasons for our choice we need to spell out what the requirements of the project are.

To build any expert system, certain language requirements are necessary (Townsend 1987):

- The language should support data-driven structures.
- The language must support symbolic processing. In other words, we should be able to use symbols, and not just numeric values, to represent objects. We must be able to do formal reasoning with the symbolic variables.
- The language must support list processing. We should be able to build list symbolic structures with hierarchical relationships if necessary.
- The language must support recursion.

From a review of many data-driven programming languages, we found that Prolog was best suited to our problem domain because of its inheritable and inferential capabilities. Prolog's greatest power is in its ability to infer (derive by reasoning) facts from other facts. This is a distinctly different process than numerical calculations. For example, a numerical processor can calculate the average of a set of input numbers. The average is something new, something not entered into the computer by the user. If this same user enters a hundred addresses, conventional procedural languages make it inconvenient to do anything more with the addresses. They can be sorted or retrieved in any order, but the addresses

still remain just that. The computer can tell you little about the relationship of the addresses; the computer can only see the addresses as numeric or string variables, not as symbolic objects that have relationships. In contrast, with Prolog the user can enter non-numeric data into a computer and the computer can deduce new facts (symbolic relationships) from the input data. The process is essentially a pattern-matching process using formal rules and is easy to accomplish.

Programming in Prolog is a completely different experience from using a procedural language. Programmers who have spent time learning procedural languages will have to go through an "unlearning" experience before they can begin to get proficient in Prolog. They tend to continually apply procedural concepts to their programs and have difficulty converting data into a form that adequately supports the problem-solving process. The advantage in learning the new language is that one will be able to apply the computer to solve new problems that are not easily adaptable to a solution using traditional languages. For example, the traversal of semantic networks is easier to accomplish in Prolog than in conventional procedural languages and these basic techniques are usually mastered by the programmer.

According to Lucas (1996), to write a Prolog expert system program there are essentially four steps, namely:

i. Define the goal.
ii. Define the domain.
iii. Define the objects in the domain and facts about these objects.
iv. Specify the rules about the facts and their relationships.

During the analysis phase of our project, we followed these four steps that culminated in a design for FUND as depicted in section 6.3.

Before we discuss the design of FUND however, we summarise in the subsections that follow why Prolog is a suitable language for the development of our expert system and highlight those features that are pivotal to FUND.

6.2.1 Rule-based

Any algorithm written in Prolog is a series of rules. The rule-based approach has certain advantages over the conventional approach to writing programs:

- Rules can work largely independently of each other (the whole system does not collapse if one is removed in the same way that a FORTRAN or C++ program might if one line were to be removed). The system will produce some sort of sense even when incorrect Conversely, with conventional programs the outcome is very unpredictable and will often lead to a program crash.
- Rules can be incrementally updated. More rules can be added at will, without any changes to the core programming aspects.
- Rules are the natural choice for semantically related data.
- A set of rules is a natural structure for expertise. Many human skills, perhaps most, can be accomplished by learning rules.

As we shall demonstrate later, the extensive use of rules will form the backbone of FUND.

6.2.2 Declarative

To a large extent, Prolog is declarative, which can make program design very much like program specification. This makes rules concise and amenable to verification by

inspection. Any assignment of variables is effected by *unification.* There are no explicit decisions or branches. There is virtually no execution logic to specify.

If/then rules are inherently encoded into the program. This style of specification is ideal for the development of a rule-base for an expert system that allows real-world rules to translate in a fairly direct way to program rules, very much like that which we wish to accomplish with FUND.

6.2.3 Explanations

A Prolog program can be made to explain its own reasoning in a very straightforward and simple manner. As we mentioned in Chapter Two (section 2.3.5), a very important characteristic of most expert systems is the ability to explain its reasoning process. This aspect is very important in relation to our system, since it is an advice-generating system that presents the user with various options and the responses must be justified by an explanation mechanism as to how each option was arrived at.

6.2.4 Backward and Forward Chaining

A focal point of this Chapter is to determine an appropriate basis reasoning strategy for FUND, that is, either forward chaining or backward chaining as discussed in Chapter Four (section 4.1) or a hybrid system. We established earlier that this decision was dependent on the programming language chosen, the knowledge representation scheme and the problem domain. Thus far, we have determined that Prolog has many of the necessary capabilities to be our programming language, but we now need to investigate what reasoning schemes it can accommodate that will best suit semantic networks and the problem domain.

The basic theorem-proving aspect of Prolog is backward chaining: this means that the Prolog system inherently starts with what has to be proved and attempts to build a proof tree for it. But, as Prolog is a complete programming language, there is nothing to prevent us coding our own inference engine in Prolog that forward chains. Recall from Chapter Two, an inference engine is simply a program that can manipulate a known set of facts and rules to produce (infer) new facts.

In forward chaining, we do not try to prove any particular goal, but deduce whatever we can from the known rules and facts. Either way, Prolog turns out to be a suitable language for writing inference engines.

In relation to FUND, the default backward chaining feature of Prolog is adequate to build our proof trees using the semantic network as the knowledge representation scheme for the problem domain.

6.2.5 Top-down Design

Prolog naturally encourages top-down design when writing programs. Horn clauses naturally break down problems into a hierarchy of sub-goals with local parameters. Hence, the natural design method is to start with the principal goal, break it down into sub-goals while identifying the variables needed at this level, and then repeat the process for each sub-goal. This is exactly the top-down design methodology of conventional software engineering. Thus, Prolog comes with a sound design methodology that facilitates the construction of expert system rule bases, and in particular, our IIR system. On the other hand, if the situation warrants a bottom-up design methodology, Prolog has the capability to allow the programmer to do just that.

As far as FUND is concerned, a top-down approach will be adopted during its design and this is will be evident in its code (Appendix A).

6.2.6 First-order Logic

The value of Prolog being based on first-order logic is that this is a mathematically sound vehicle for reasoning and modeling our problem area. It helps in formulating logically consistent rules, forcing us to think clearly by directly expressing our programs as logic, which Lucas (1996) considers to be one of the cornerstones of human intellect. This mathematical foundation of first-order logic is sufficient to justify FUND's reasoning strategies.

6.2.7 Searching in Prolog

Prolog essentially implements a depth-first search by default. In depth-first search, the search goes down as far as possible and then back up (Giarratano & Riley 1989). In Prolog, the search goes from left to right. Apart from the traditional depth-first search, the programmer can also implement a breadth-first search that proceed one level at a time before descending to the next lower level (again from left to right), as well as a best-first search algorithm (Bratko 1994: 321 - 324).

Nevertheless, with FUND, we found that the default depth-first search strategy was sufficient to return relevant hits, particularly because we introduced a depth-limitation constraint. A breadth-first search would also have sufficed, however, the extra code required was unwarranted. The only difference would have been in the order of the hits returned, which was inconsequential as regards the relevancy factor.

6.2.8 Backtracking

The description of Prolog's powerful pattern-matching capabilities in the next section (section 6.2.9) omits to explain what is one of the most important mechanisms available in Prolog. This mechanism is called *backtracking* (Black 1986).

When searching for an answer to a question, particularly if obtaining the answer involves the use of rules and the subsequent satisfaction of conjunctions of sub-goals, Prolog makes a series of choices as to which values it instantiates variables to. Basically, because of the top to bottom, left to right manner in which Prolog searches through the knowledge base, it will choose the first possible match that it discovers. This match might not, however, be the one that is required in order to provide a satisfactory match to the problem in hand. That is, Prolog might find that subsequent goals will fail as a result of having made the wrong choices at an earlier stage. Backtracking is the mechanism by which Prolog can go back and make other choices for the values of variables and then attempt to re-satisfy the subsequent goals. In the context of our project, this mechanism is invaluable in the pursuit of alternative semantically linked goals that may reveal a solution if none can be found for the original goal.

How does backtracking work in Prolog? Prolog will search the knowledge base from top to bottom and, if unable to find a suitable match, it must retrace its steps and "rethink" that which it has previously done. Prolog will backtrack to the last choice (instantiation) that it made. Prolog will then satisfy the remaining sub-goals to the right with the new instantiated set and the same procedure is repeated until a match is found or the knowledge base is exhausted.

How does Prolog manage to remember all the choices, which it has previously made in order to retrace its steps so quickly? It does so by placing a series of markers as it threads its way through a conjunction of goals and maintaining a stack of pointers to those markers. Each time it instantiates a variable to a particular value it leaves a marker which says, "I

made a choice here. I shall mark it so that, if need be, I can return and undo that choice in order to choose another value." On backtracking, Prolog will return to the last marker which it left and choose the next possible value for the variable which it instantiated at that point. It will then proceed to try to re-satisfy subsequent goals.

If it fails again, it will return to that marker and make yet another choice. It will continue to do this until either the subsequent goals succeed or there are no more choices to be made at the marked point. If the latter is true (that is, it has exhausted all of the choices to be made at a particular marked point) Prolog will go back to the marker previous to that point and continue in a similar manner. It is in this manner that Prolog manages to cover all the possible combinations of values of variables, which could possibly provide a solution to the problem, which it has been set by the user. The search is therefore exhaustive.

It may sometimes become necessary for the user to expect more than one solution to a single query. One such way is to use the semicolon (;) in a question-and-answer session or at appropriate points in the coding. When Prolog provides us with an answer to a query and we type in a semicolon, we are really saying to Prolog, "Go away and retrace your steps to see if there are any more possible answers to the question which I originally set you." Prolog will then backtrack to the nearest (most recent) clause marked, and will continue to search for another solution. Remember that variables that were previously bound in an attempt to satisfy the previous goal become unbound after the backtrack point (Robert 1990; Smith 1990).

This semi-colon will be coded into the Prolog "functions" of FUND to provide alternatives when backtracking (Appendix A). This is an important feature that appears to imitate the human mind to some degree. If whilst trying to solve a problem we arrive at a dead end, we usually backtrack to some earlier point of branching in our thought processes in pursuit of a solution in a new direction. This iterative nature of Prolog will prove to be most beneficial in FUND.

Automatic backtracking is a useful programming concept because it relieves the programmer of the burden of programming backtracking explicitly. On the other hand, uncontrolled backtracking may cause inefficiency in a program. Therefore, we sometimes want to control, or to prevent, backtracking (Bratko 1994: 125 - 139). Backtracking may be inhibited via the "cut" (!) predicate. The (!) symbol is interpreted as a predicate that always succeeds, i.e., always evaluates to ***TRUE***.

In realistic or significantly more complex Prolog descriptions, proper placements of the "cut" predicates often makes the difference between a program that works and one that doesn't. When a cut is encountered in the process of verifying a conjunction of sub-goals in a rule, the cut forces the Prolog unification mechanism to commit to all binding choices made from the past goal up to the cut. Thus, the cut may be viewed as a "fence" the unification mechanism may only cross once in unifying from left to right in an expression.

The "cut" yields efficiency by signifying that only one (not all) successful unification should be found. For example, suppose we have the rule involving the predicates ***r1***, ***a***, ***b***, and ***c***, of the form

r1 If a and b and ! and c.

The application of the "cut" operator following clause ***b*** indicates we are satisfied with the first solution of the sub-goals ***a***, ***b***. The "cut" operator, in conjunction with careful placement (order) of rules and facts in the database, is useful for controlling, and often limiting, the solution search (Robert 1990: 106). It is for this reason we will use the "cut" predicate extensively within FUND to control backtracking (Appendix A).

It is this powerful backtracking mechanism provided by Prolog that makes it such a useful problem-solving tool. Prolog will try every possible combination of choices in order to provide you with an answer to your query. Only when it has tried every possible combination and still failed will it be satisfied that no solution exists to your problem. It will then respond with the answer no. Backtracking and the ability to control backtracking

in Prolog has been one of the fundamental reasons for choosing Prolog as the language to use for our IIR project.

6.2.9 Unification

The process by which Prolog tries to match a sub-goal against facts and heads of other rules to prove the sub-goal is called *unification* (Townsend 1987). It is an inherent *pattern-matching* algorithm and an essential technique that Prolog uses to solve the original goal – it is coincidently also the heart of any search technique of an IIR system.

A simple term is said to unify with another term if they both have the same predicate, the same number of arguments, the arguments of the same domain type, and all of the sub-terms unify with each other. If a free variable is unified with another term, the free variable will be bound to the values of that term. Unification functions act much like parameter passing in procedural programming and can even do certain tests for equality or comparison.

A summary of the unification rules follow:

1. A free variable will unify with any term. As a result, the variable is bound to the term.
2. A constant can unify with itself or any free variable.
3. A variable can unify with any variable. After unifying, the two variables act as one.

6.2.10Recursion

Backtracking is an inherent strategy used in Prolog to search for the solution to a query. Due to its exhaustive approach, the programmer can often use knowledge of the problem domain to limit the extent of the search without jeopardizing the correctness of the evaluations. This is achieved by using the "cut" operator to delimit the search in acceptable ways (Kim 1991: 61 – 62).

Backtracking is therefore a *domain-independent* strategy for searching for a solution, given the facts and rules for a particular problem. It is used by Prolog regardless of the domain of application. For the search procedure to apply, however, there must be some knowledge of a problem encoded in the form of facts and rules.

A common reasoning technique used in AI and its applications is that of *recursion* in which a procedure is partly defined in terms of itself. This *domain-dependent* technique is especially useful when repetitive operations must be effected on a series of objects, but the number of objects is not known in advance, such as on trees and graph-like structures.

A *recursive rule* has two parts: the base case and an inductive part. The *base case* defines the terminating condition, while the *inductive part* specifies the iterative operation in terms of the rule itself. Prolog accommodates the numerous instantiations during recursion with the aid of a stack frame that usually is governed by the available memory of the computer being used. This recursive rule will be used extensively in FUND (Appendix A) to find all funding sources (within the specified depths). In addition, for the exhaustive search, it will return every possible hit. This recursive rule will be demonstrated during the design phase of this Chapter (section 6.5.2) and the implementation phase of the next Chapter (section 7.1).

6.2.11 Concluding Remarks on Prolog

To summarize, Prolog offers numerous advantages to the expert system builder. However, Prolog has a few disadvantages and it would not be fair to continue without mentioning these. Firstly, the order of the rules and facts is important to the meaning. Prolog is not truly non-procedural. Secondly, all rules must reside in the computer's memory. The number of rules the expert system can use is limited by the memory size of the computer. With most versions of Prolog (including LPA Prolog), there are methods by which you can use the disk as an extension of memory, but this alternative virtually ensures a slow program (Giarratano & Riley 1989).

Search control in Prolog is fixed (Elaine & Kevin 1991). Although it is possible to write Prolog code that uses search strategies other than depth-first with backtracking, it is difficult to do so. It is even more difficult to apply domain knowledge to constrain a search. However, a depth limitation constrain is easily accommodated by Prolog, as we will be evident in the next Chapter (section 7.1). Furthermore, Prolog does allow for rudimentary control of a search through a non-logical operator called "cut". As mentioned previously, a "cut" can be inserted into a rule to specify a point that may not be backtracked over.

More generally, the fact that Prolog programs must be composed of a restricted set of logical operators can be viewed as a limitation of the expressiveness of the language. But the other side of the coin is the introduction of the "cut" operator that makes it possible to build Prolog programs that produce efficient code, which is usually one of our requirements.

In the sections that follow, we elaborate on the design issues of our expert system, FUND, built in LPA Prolog and highlight those features that are pivotal to its success.

6.3 The Architecture of our Solution to FUND

Thus far, we have established that the knowledge representation structure for FUND will be a semantic network, and the reasoning scheme to be adopted will be the default backward chaining mechanism adopted by Prolog. We have also established that FUND will implement a search technique that resembles the depth-first search technique with backtracking that is also default to Prolog.

FUND's architecture, as depicted in fig. 6.1 below, includes a large semantic network of related research topics centered around the "directed" themes identified by the NRF, a menu-driven user interface for choosing the type of search technique to be employed as well as for specifying proposals and presenting results, and a built-in control structure (the inference engine) for finding appropriate themes and programmes for a given proposal.

The three-tier architecture for FUND depicted in fig. 5.1 below is consistent with the general architecture of an expert system depicted earlier, in fig. 2.1 (section 2.3.1)

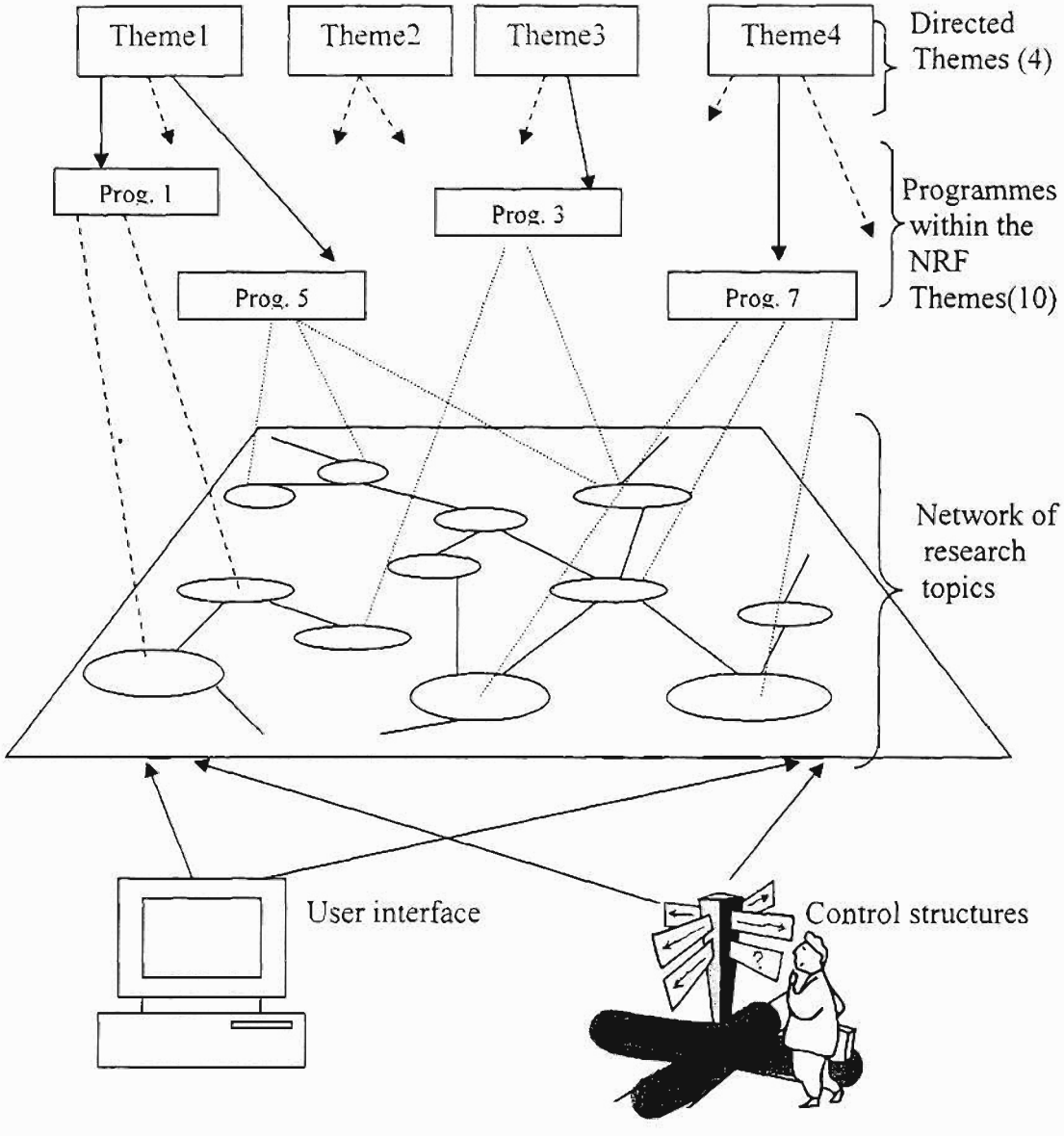

Fig. 6.1 General architecture of FUND

The semantic network is in effect an index to programmes and ultimately themes, since each programme is linked into the network at those nodes of the network that represent its research interests. Proposals, once elicited from researchers, are linked into the network in the same way.

In overview, the system works by Spreading Activation from a proposal through the network until one of the bottom level nodes is activated, followed by all terms that are directly related (that is, one link away) in the network, followed by *their* related topics, and so on, as activation spreads across relations in the network until one of the four themes is activated.

Ordinary Spreading Activation can quickly touch every topic in a network, which means that it can find pathways from any research proposal to any programme and ultimately any theme description. Since, most programmes found in this manner would not fund a given proposal, our task is to modify FUND's search algorithm to weed out distant and unrelated links. This could be accomplished by implementing a *Constrained Spreading Activation* algorithm.

This algorithm could be constrained by a set of rules to favour particular pathways through the network and to terminate search along other pathways where the semantic distance is too large. As far as FUND is concerned, we will only concentrate on the distance constraint as will be discussed later. Constraining the search to favour particular pathways could be tackled at a later stage as part of a future research effort in order to eliminate as many false positives as possible (section 8.3). The rules within FUND will lead the search to the "closest" programmes or themes related to the research proposal and prune distant and irrelevant links found by ordinary Spreading Activation. It may well happen that many links may lead to the same programmes.

Before we implement any constraints within FUND, we need to investigate Constrained Spreading Activation in general to determine which constraints will be best suited to

FUND. This will be accomplished later in this Chapter (section 6.5.4). First, however, we will examine the various components that constitute FUND, namely its knowledge base, inference engine and user-interface.

6.4 FUND's Knowledge Base

The knowledge base of FUND consists of 5 distinct levels, the highest of which are the four "directed themes" of the NRF funding programme for technikons, namely Competitive Industry, Improved Quality of Life, Sustainable Environment, and Effective SET Education and Awareness. The next level constitutes the various programmes within each of the themes listed above. Currently (1999 - 2000), there are 10 programmes within the "directed theme" branch of the NRF. These are listed below:

Directed Themes	***Programmes***
Competitive Industry	*Primary resource beneficiation*
	Manufacturing advancement
	Information and infrastructure systems
Improved Quality of Life	*Rural and urban development*
	Food production and food security
Sustainable Environment	*Inland resources*
	Marine and coastal resources
Effective SET Education and Awareness	*Innovation and change in education*
	Preparation and development of educators
	Public understanding of SET

The next level in the hierarchy belongs to the sub-themes of each of the programmes listed above. The fourth level, which contains the largest number of nodes, constitutes all those keywords that are relevant to the various programmes and sub-themes listed above. Some of the terms at the fourth level contain references to many more terms themselves – and this constitutes the fifth level of the hierarchy.

Fig. 6.2 below shows the hierarchy of FUND and the relationships between the nodes:

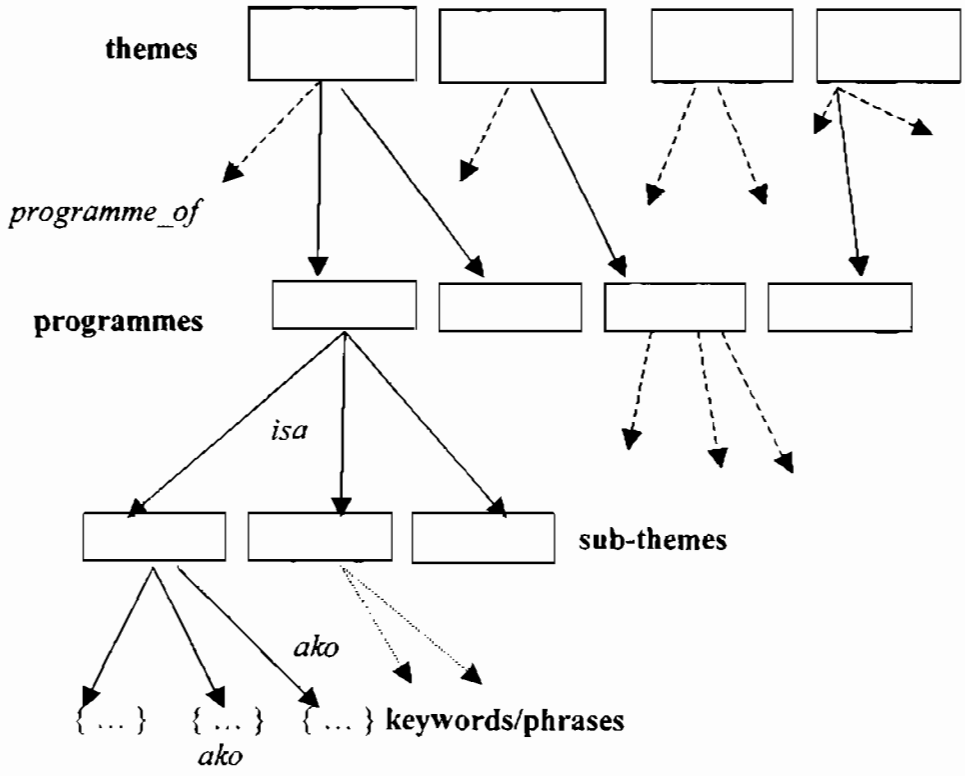

Fig. 6.2 FUND's hierarchy and inheritable links

All the terms of the hierarchy above have been intertwined within a semantic network of related concepts within Prolog. The glue that holds the network together is the three main network links, namely the **ako**, **isa** and **programme_of** relations. The **programme_of** relations link the programmes to the themes, the **isa** relations link the sub-themes to the programmes, and the **ako** relations link all the keywords to the sub-

themes. The lowest level is considered to be the most dense level because of its high degree of interconnectivity within that layer as well as higher layers.

Currently, there are ± 510 nodes in FUND that represent the research interests of the four directed themes within the NRF framework. Nodes may be easily added to the network by linking them to the other nodes using one (or more) of three distinct relations, namely, **ako**, **isa** and **programme_of**.

Because of the built-in inheritance mechanism of the semantic network, there is no need to drastically change the knowledge base to incorporate new nodes. For example, we can define a **gold** node by linking it to **metals** and **mining_optimisation** without any other changes to the knowledge base, as follows in fig. 6.3:

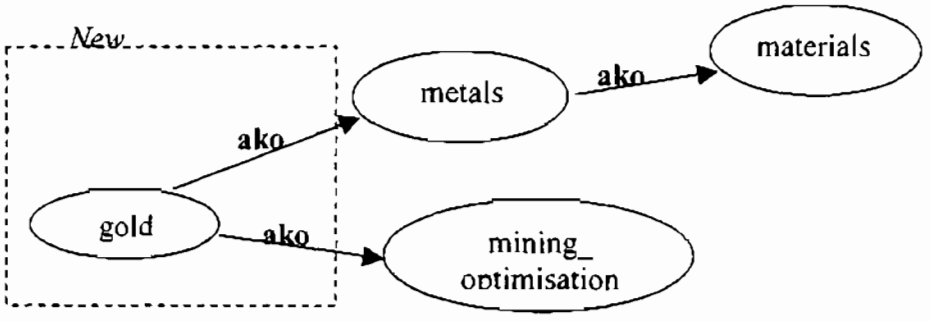

Fig. 6.3 Inclusion of a new node into the knowledge base

The following nodes need to be included into the existing knowledge base to reflect the changes above:

```
ako(gold, metals).
ako(gold, mining_optimisation).
```

Nodes are added only as needed to reference research topics. In the almost unlikely event that nodes need to be removed from the knowledge base, the system designer must be wary

of eradicating all "hanging" nodes as a consequence. However this seldom occurs as the initial knowledge base is usually carefully pre-planned and only relevant nodes are included. FUND's knowledge base is not an encyclopeadic collection of scientific, societal, geological, and educative terms, but a highly cross-referenced index of research terms, represented from the perspective of the NRF (Westhuysen 1999). The knowledge base is therefore relatively small as it serves mainly an illustrative purpose only.

6.5 FUND's Inference Engine

In the previous section, we outlined *how* the various nodes were linked together to form the complete semantic network of research related terms, and in doing so, formed the backbone of our system, namely, our knowledge base. In this section we highlight the design issues of how the search takes place through the network, that is, we concentrate on our inference engine. Bearing in mind that our search is initiated by entering a keyword or phrase from the research proposal, we need to demonstrate how the system is capable of "intelligently" searching through the semantic network to reveal links to programmes and ultimately themes that are closely related to the keyword or phrase entered.

We must not loose sight of the fact that the programming language, Prolog, itself has a *built-in* backward chaining inference engine that we could use (Dennis 1989; Black 1986; Chris & Gerald 1989; Claudia 1986). The Prolog rules that form the semantic network are used as the knowledge representation structure, and the Prolog inference engine is used to derive conclusions. Each rule has a goal and a number of sub-goals and, with the aid of backward chaining, the Prolog inference engine either proves (finds hits) or disproves (return nil hits) for each goal by traversing the network via the links or relationships. It is worthwhile at this stage to take a closer look at Prolog's built-in reasoning strategy.

6.5.1 Prolog's Reasoning Strategies

As mentioned earlier in this Chapter (section 6.2), the expressiveness of Prolog is due to three major features of the language: rule-based programming, built-in pattern matching, and backtracking execution. The rule-based programming allows the program code to be written in a form that is more declarative then procedural. This is made possible by the built-in pattern matching (unification) and backtracking that automatically provides for the flow of control in the program. Together these features make it possible to elegantly implement our expert system (Appendix A).

To determine the validity of a rule or query, Prolog must search through its knowledge base, and it does so by pursuing a systematic reasoning strategy known as *backtracking* (section 6.2.8). This involves a methodical search, through trial and error, of all possible combinations of facts and rules that might lead to a solution (Steven 1991). Backtracking is a way to explore a knowledge base in a coherent fashion. It is an automatic procedure that operates independently of the way in which the programmer represents a problem or specifies the rules of inference.

If Prolog determines that a solution cannot be reached by proceeding in a particular direction of search, it returns to the last choice or decision point and tries another route. Prolog will continue trying all routes until it finds the solution or determines that there is none. This is an important aspect of FUND since it must return an exhaustive list of options (if any exist).

Backtracking occurs when Prolog determines that the current line of inquiry will not yield a solution. For this reason, backtracking is unnecessary in the occasional instances when the only solution is discovered on the first attempt. With regard to FUND, this is applicable particularly in the event of the user entering a theme or existing programme as a proposal descriptor.

In addition to automatic backtracking, in our project we adopt a more pervasive reasoning technique in knowledge engineering, called *recursion* (Black 1986; Chris 1989), a technique whereby a procedure is defined partly in terms of its own structure (section 6.2.10). This method is especially useful when an operation must be performed repeatedly on a sequence of objects, but the length of the sequence is not known beforehand.

Before we look at the specific implementations of recursion (section 7.1), it is imperative that we examine the structure of the links that glue the nodes of the semantic network first.

6.5.2 Inheritable Links within FUND

The second highest order of the goals is represented in the form of **programme_of** relationships. These are many-to-one relationships. The next level is represented by **isa** links (also many-to-one type links), and the lower levels are depicted by **ako** relationships. These (**ako** links) are many-to-many relational links. By using the **ako** inheritance links, the lowest level of terms is attached to the network. Recall that this is the densest level because of the high degree of interconnectivity to this and higher levels.

The unification of variables using the **programme_of** and **isa** links is relatively straightforward since the maximum depth level is at most 3 between any two nodes and, both these links are susceptible to automatic backtracking. The **ako** relational links are inheritable and therefore cannot be treated in the same fashion because the maximum depth level is not obvious beforehand. Therefore, the **ako** links must be implemented using recursion (including a depth counter **D** used for pruning which is discussed later), and this is done using the **akko** recursive rule as follows:

```
akko(X, Q, D) :-
        ako(X, Q), write(Q), nl.
```

```
akko(X, Q, D) :-
        (DD is D - 1, DD > 0,
        ako(X, ZZ),  akko(ZZ, Q, DD)).
```

With the aid of this recursive rule, FUND will be able to search for hits at lower levels using the same links, namely **ako**. Next, we need to investigate whether Constrained Spreading Activation could be incorporated into FUND to assist in making the process more effective.

6.5.3 FUND's Search Methodology

The search for a solution proceeds systematically, by traversing each potential path in full before attempting the next, in a depth-first fashion, as opposed to a breadth-first fashion of layer by layer. The search therefore spans the semantic network from top to bottom and from left to right.

The major disadvantage of this methodology is that all the solutions that prove the goal will not necessarily be revealed in the preferred order of closest solutions first. This downfall however, is not counter-productive if we can constrain our search to a maximum depth that is predetermined in order that we ignore distant and unnecessary links. This pruning is vital since, by the very nature of semantic networks and the recursive rule above, the solution set could easily include every possible solution node, immaterial of the length of the solution paths.

If we choose to implement the constrained search technique, the choice of whether to use depth-first or breadth-first technique becomes inconsequential. Therefore, the strategy we adopted was the implicit technique used by Prolog. Prolog, as mentioned earlier, relies on a backward chaining, depth-first strategy. It begins with the hypothesized goal and moves backward to determine if there is indeed a logical consequence of the initial knowledge

base (Steven, 1991). The backward procedure is also called a "top down" strategy when the state tree is regarded in inverted form. In other words, the goal or hypothesized conclusion(s) is (are) regarded as the initial, topmost node and the facts or premises as leaves in the tree of knowledge. The search strategy then involves a descent through the levels of the hierarchy.

The pruning search technique that we wish to use is the Constrained Spreading Activation method. In the section that follows, we briefly explain the evolution of the general Constrained Spreading Activation search technique as well as some existing applications implementing this technique.

6.5.4 The Constrained Spreading Activation (CSA) Model

Spreading Activation (SA) is not new. The idea behind this heuristic method for implementing plausible inference processes were sketched by Fikes and Hendrix (1977: 235 - 246): In Spreading Activation a concept is identified in a semantic net and then adjacent concepts in the net are visited while solving a query. The significance of the traversal of the net depends on the net and the way the traversal is used to infer a solution to the problem.

Usually the node activation method used on the Spreading Activation model starts by placing a specified activation weight at some starting feature or at a node. The initial weight then spreads through the network along the links originating at the starting point (Salton & Buckley 1988: 147 - 160). Turtle and Croft (1991: 187 - 222) "attach" the query and its features to the network by coupling the query features with the corresponding node features. In this way, one might have several activation weights at the beginning of the spreading phase. This process of deploying activation weights at nodes however, is necessary for document retrieval and since FUND is concerned with keyword or phrase searches, it will not require weighted nodes.

The SA model has its roots in the field of Psychology, since it resulted from studies of the mechanisms of human memory (Rumelhart & Norman 1983). It was first used in the area of Artificial Intelligence, particularly in systems designed to imitate human thought and behaviour, but recently, it has also been used in other areas of Computer Science.

SA has often been associated with Semantic Networks (Quillian 1968), since it has been established as the preferred processing framework for Semantic Networks. This idea dates back to the work of Fahlman, Touretzky and van Roggen (1981) on the Marker Passing Model. We will take a somewhat different view. We will view SA as a general processing framework for any network of objects and their relationships, with the important difference that SA does not involve any learning at all.

The pure SA model consists of a network data structure upon which simple processing techniques are applied. The network data structure consists of nodes interconnected by links. Nodes may represent objects or features of objects, and are usually labeled with the names of the objects they represent. Links model relationships between objects or features of objects. Links may be labeled and/or weighted and usually have directions, reflecting on the relationship between the connected nodes.

SA techniques are iterative in nature. Every iteration consists of one or more pulses and a termination check. The processing is simply a sequence of such iterations that can be terminated either by the user, or by the system. Each pulse is made up of three stages: pre-adjustment, spreading, and post-adjustment. The first and the third phase are optional. They enable some form of control over the activation of the nodes in the network, like for example, some form of activation decay to be applied to the activated nodes, so that the retention of the activation from previous pulses can be avoided.

The spreading phase of the pulse consists of a flow of activation waves from one node to all other nodes connected to it. The activation input of a node can be calculated with the following simple formula (Crestani & Lee 2000: 585 - 605):

$$I_j = \sum_i O_i w_{ij}$$

where I_j is the total input of node j, O_i is the output of node i connected to j, and w_{ij} is a weight associated to the link, default of which is 1, connecting node i to node j.

The numerical values of the input and weight depend on the application being modeled by the network. For example, in Semantic Networks, the values of the weights are usually binary, i.e., either 0 or 1, or they could also reflect excitatory/inhibitory values (+1 and –1, respectively).

The output of the node, O_i, is usually its activation level. The output value is fired to all nodes connected to the active node. Usually, the same output is sent to each node. In this way, the activation spreads pulse after pulse.

After a pre-defined number of pulses, a termination check is carried out to determine whether a termination condition has been met. If so, the SA process halts. Otherwise, another series of pulses continues, followed by another termination check. This cycle goes on until the termination condition is met. The end result of the SA process is the activation level of each of the nodes in the network at termination time.

It has been proved experimentally that the pure SA falls short in several ways (Preece, 1981). The most salient fault is that, unless the activation is carefully controlled during the pre-adjustment and post-adjustment phases, the activation tends to quickly spread over the entire network. This would render the activation process meaningless. Secondly, the semantics of the network associations is not utilised by the pure SA model, so the

information provided by the types of associations is not used. Consequently, it is difficult to implement some form of inference based on the semantics of the associations themselves.

The shortcomings of the pure SA model can be partially overcome by implementing rules. so that the spreading of the activation could be used to implement some form of inference or heuristics. This new model is called *Constrained Spreading Activation (CSA)*. In fact, a common way of implementing rules is by way of placing constraints on the spreading of the activation. Some common constraints that have been implemented are the following (Cohen & Kjeldsen 1987; Crestani & Lee 2000):

- *Distance constraint*: This places a constraint on the distance to which the activation can spread. Nodes that are far away from the activated node, i.e., nodes with a very high semantic gap (section 2.3.1), should be less likely to be activated because of their longer distance in the number of links necessary to reach them. The rationale behind this rule is that the strength of the association between nodes decreases with increasing distance. This constraint, together with the semantics of the network associations, will prove to be most beneficial to our project as will be evident in the next Chapter.

- *Fan-out constraint*: The spread of the activation should stop when it reaches a node with a large fan-out, that is, a node with a large number of associations branching out. This is because a very large fan-out means that the node has very broad semantic meaning and thus the activation may flow to too many nodes, thus diluting the semantics of the spreading. This constraint is of particular importance. especially when synonyms are used or words with dual or multiple meanings are to be expressed in the network.

- *Path constraint*: Activation should spread along certain preferred paths, which are predetermined according to the particular application. This constraint can be modeled using the weights of the associations. For labeled associations.

preferences of activation flow can be set up according to the semantics as for example activation flow can be diverted to certain paths while being diverted from paths with different semantics.

- *Activation constraint*: This places a threshold value on the activation level of a single node. If the activation of the node is below the threshold value, activation would not spread from that node. It is possible that different threshold levels be set for different nodes according to their meaning and the application.

Since these constraints serve to control the activation spread, they can be introduced in the pre-adjustment and post-adjustment phases of the pure SA model. The distance, fan-out and path constraints take place during the pre-adjustment phase, while the activation constraint fires during the post-adjustment phase.

The *distance constraint* of the CSA model seems to be a sufficient constraint that FUND could utilise to prune its search. The other constraints are not necessary at the prototype stage and could be pursued as a future research option (section 8.3). In the Chapter that follows, we will investigate how this model can be implemented for searching the knowledge base for NRF programmes. However, before we look at the implementation of our prototype, it is worthwhile to review some related efforts in CSA that have been successful to better understand how this process could be refined within FUND.

6.5.5 Related Work on CSA

SA techniques have been used by a number of researchers in IR. The first works in this area date back to the 1980s. A detailed survey of the application of SA techniques can be found in Crestani (1997: 454); here however, we will only concentrate on the approaches that are more closely related to our approach.

Preece (1981) examined the SA approach to Associative Retrieval extensively, drawing the distinction between the various forms of SA, like for example, CSA. He argued that most of the classical approaches to IR could be explained in terms of different SA processing techniques on network representation of the document collection. This division between data structure and processing technique can be seen as a first attempt to conceptual modeling of IR applications. By combining different network data structures with different processing techniques, he showed how it was possible to implement the Boolean model, the Vector Space model and use various forms of weighting for Associative Retrieval. Moreover, he showed how, using relevance feedback, SA can be used for automatic classification and indexing, and for concept building.

As mentioned earlier (section 4.4), Cohen and Kjeldsen's GRANT system is one of the first systems to use CSA in IR (Cohen & Kjeldsen 1987). Recall in GRANT, knowledge about research proposal and potential funding agencies is organized using a Semantic Network. Research topics and agencies are interconnected using a wide variety of association links to form a dense network. A query expresses one or more research topic, or one or more funding agencies. The search is carried out by CSA on the network, identifying and using the types of constraints described in the previous section. In particular, GRANT makes large usage of path constraints in the form of "path endorsements", which consist of giving preference (positive endorsement) to some paths in the Semantic Network, while enabling the user to avoid (with a negative endorsement) some misleading paths.

This approach is possible and useful on a Semantic Network, where links connecting nodes have a type and a clear semantics. The use of path endorsements however, is not possible in our project since it is merely a prototype and does not posses a highly interactive HCI that accomplishes query reformulation. This issue, however, could be pursued as a future research effort (section 8.3). However, a study of the GRANT system has been beneficial in that FUND will implement the distance constraint in a similar fashion.

A prototype system similar in many aspects to GRANT is I^3R. The declared purpose of Croft, Lucia and Cohen (1988) and Croft, Lucia, Crigean and Willet (1989) in designing I^3R was mainly to study the possibility of retrieving documents by "plausible inference". I^3R is designed to act as a search intermediary. It accomplishes its task using domain knowledge to refine query descriptions, initiating the appropriate search strategies, assisting the users in evaluating the output, and reformulating queries.

I^3R can be considered as one of the best attempts to combine CSA model with the most sounded IR probabilistic techniques. However, it requires a well-constructed and consistent network of documents that will never be found on the Web. Web links do not have weights and it is therefore very difficult to determine certainty / evidence values and carry out inference processes.

Since then, much research has been carried out to incorporate CSA to search through hypertext, such as WebSCSA (Crestani & Lee 2000) that is currently at a prototypical stage. Therefore, since CSA has found widespread applicability as a viable IIR technique, it is worthwhile pursuing as a pruning mechanism within FUND. However, since FUND is only a prototype, we need only concern ourselves with the base level constraint of depth-limitation in CSA that will be sufficient to satisfy our original aim. We will investigate how this could be achieved in the next Chapter during the implementation phase of FUND.

Finally, as far as FUND's inference engine is concerned, we established that Prolog's default backward chaining, depth-first search technique will be used and that traditional Spreading Activation over the semantic network will not be sufficient to return relevant links by FUND. Therefore, we will investigate whether CSA, in particular, its depth constraint, could assist in pruning distant and non-relevant links during FUND's implementation in the next Chapter.

6.6 FUND's User Interface

Bearing in mind that the primary purpose of this research effort is the building of an IIR system to retrieve relevant links for rendering funding advice, we did not devote a great deal of effort on developing a highly interactive user-interface. Although we have established that an advanced HCI forms an integral part of a real-world IIR system, a rudimentary user-interface will suffice to meet FUND's original aim. Issues pertaining to an advanced HCI such as Natural Language Processing (NLP), query reformulation, etc. could be pursued at a later stage (section 8.3).

Nevertheless, we have included in FUND a menu-driven user-interface that will be adequate for our needs. FUND will be initiated by typing in the predicate "**menu**." that will pop up a window that will allow the user to make a choice from a menu of options pertaining to the type of search he or she requires. These options will include a **depth_4_search**, a **depth_5_search**, an optional **depth_4_or_5_search**, and an **exhaustive_depth_first_search**. The first two of these searches will accomplish depth-limited searches to a maximum depth of 4 and 5, respectively. The third search is an optional search that will accomplish a depth 5 search only if a depth 4 search fails. The last search is an exhaustive search that is primarily used as a means of comparison to determining the degree of relevancy of the other searches.

Once an option is chosen, the user will be prompted to enter a descriptive keyword or phrase for his/ her proposal. The result entered will be used for the search through the semantic network and all relevant links (from FUND's perspective) will be displayed onto the screen, clearly explaining it's reasoning for each programme or theme that is returned.

6.7 Conclusion

In this Chapter, we concentrated on the design issues relevant to our project. In particular, we highlighted those aspects of the Prolog programming language that were responsible for us choosing it as the programming language for FUND. The most important features, inter alia, included Prolog's powerful backtracking and pattern matching (unification) capabilities as well as its default backward chaining, depth-first search technique. Furthermore, this rule-based, declarative programming language provided an ideal vehicle for FUND to provide explanations for its choices. Also, in this Chapter, we briefly elaborated on the design of the semantic network in Prolog and the links between the nodes that form the glue for the knowledge base.

The default backward chaining, depth-first search technique of Prolog was chosen as the primary reasoning mechanism for FUND. The Spreading Activation search technique, usually adopted for systems that implement a semantic network as its knowledge representation structure, was found inadequate as far as returning relevant nodes only. As the need for pruning is an essential aspect of FUND's design, a review of the Constrained Spreading Activation search technique, in particular the distance constraint, demonstrated that this technique was most suited to pruning of non-relevant links.

The Chapter that follows will demonstrate whether this technique can be successfully adapted within FUND during its implementation phase. If the results when using a semantic network and Constrained Spreading Activation search technique prove to be encouraging, FUND can then be tested and evaluated with respect to its efficacy.

Chapter Seven

Implementation and Evaluation of FUND

From the preceding Chapters, we decided on the knowledge representation structure, namely, a semantic network, a programming language, namely Prolog, the reasoning strategy, namely, backward chaining depth-first search and a possible pruning strategy, namely, Constrained Spreading Activation (CSA). The central role of this Chapter will be to elaborate on the design issues introduced in Chapter Six and discuss the formal implementation of FUND with particular attention to the pruning strategy to be adopted. We will begin by accentuating how the traditional depth-first search in Prolog could be altered to accommodate iterative deepening (Bratko 1994) and thereafter a maximum depth search, to satisfy the *distance constraint* of CSA (section 6.5.4). FUND will then be tested with regard to its precision and recall ability. Finally, we will discuss and document an evaluation of the performance issues concerning FUND, that is, we will determine whether FUND produces more *relevant* results than a traditional depth-first search approach for the same queries (section 7.3).

7.1 Depth-first Search and Iterative Deepening

As was pointed out in Chapter Six, when the user enters a query term into FUND that satisfies the **programme_of** or **isa** relationship, there is no need for a deeper search since, the nodes at these levels are unique and therefore, a unique programme would be identified. The need for a deeper search comes into play when the query is deeper than the second level, in which case, an **ako** relationship is activated. Because of the transitive

nature of all **ako**-linked terms, the relationships are not unique and may not be at the same level, that is, there may be differing semantic distances between the goal nodes and solution nodes due to inheritance. For this reason, a depth-first search with iterative deepening is required.

Bratko (1994: 262 - 263) suggests that in order to find a solution path, **Sol**, from a given node, **N**, to some goal node, we must use the following algorithm:

- If **N** is a goal node, then **Sol = [N]**, or
- If there is a successor node, **N1**, of **N**, such that there is a path **Sol1** from **N1** to a goal node, then **Sol = [N | Sol1]**.

We start the development of the algorithm and its variation with this simple idea that translates into Prolog as:

```
akko(X, QQ) :-
            ako(X, QQ).
akko(X, QQ) :-
            ako(X, ZZ),
            akko(ZZ, QQ).
```

This program segment is in fact an implementation of the depth-first strategy. It is called "depth-first" because of the order in which the alternatives in the state space are explored. Whenever the depth-first algorithm is given a choice of continuing the search from several nodes it always decides to choose a deepest one. A "deepest" node is one that is farthest from the start node.

The depth-first search is most amenable to the recursive style of programming in Prolog. The reason for this is that Prolog itself, when executing goals, explores alternatives in the depth-first fashion as mentioned earlier (section 6.2.7). The traditional depth-first

algorithm works well in most cases, except for when state spaces contain cyclic paths between nodes. Consider the following example state space, in fig. 7.1 below, that contains a cyclic path namely, an arc from **d** to **h** and an arc from **h** to **d**.

The search in this case proceeds as follows: start at **a** and descend to **h** following the left-most branch of the graph. At this point, **h** has a successor, **d**. Therefore the execution will *not backtrack* from **h**, but *proceed* to **d** instead. Then the successor of **d**, **h** will be found, etc., resulting in cycling between **d** and **h**.

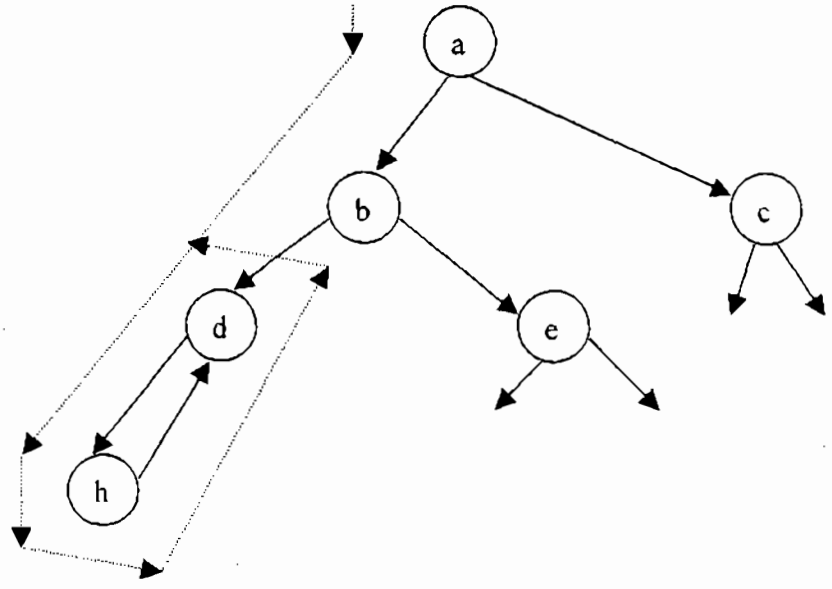

Fig. 7.1 Cyclic paths

An obvious improvement of our depth-first search is to add a cycle-detection mechanism. Accordingly, any node that is already in the path from the start node to the current node should not be considered again. With the cycle-detection mechanism, our depth-first search procedure will find solution paths in state spaces such as in fig. 7.1. Many state spaces are, however, infinite. In such a state space, the depth-first algorithm may miss a goal node, proceeding along an infinite branch of the graph. The program may then indefinitely explore this infinite part of the space never getting closer to a goal.

To avoid aimless infinite (non-cyclic) branches, Bratko (1994: 263) suggests adding another refinement to the basic depth-first search procedure: *limiting the depth of the search*. We then have the following arguments for the depth-first procedure:

```
akko(X, QQ, Maxdepth).
```

With this constraint, the search is not allowed to go in depth beyond **Maxdepth**. There are many reasons for limiting the depth of the search, the most obvious of which, is that of combinatorial explosion at greater depths, which we will re-visit in the next section. This constraint can be programmed by decreasing the depth limit at each recursive call, and not allowing this limit to become zero (or negative). The resulting predicate definition would now become:

```
akko(X, QQ, _) :-
          ako(X, QQ).
akko(X, QQ, Maxdepth) :-
          Maxdepth > 0,
          ako(X, ZZ),
          Max1 is Maxdepth - 1,
          akko(ZZ, QQ, Max1).
```

A difficulty with the depth-limited program above is that we have to guess a suitable limit in advance. If we set the limit too low, that is, less than any solution path, then the search will fail. If the limit is set too high, the search will become too complex (particularly if the lower levels are highly interconnected as in our state space).

To circumvent this difficulty, we can execute the depth-limited search iteratively, varying the depth limit: start with a very low depth limit and gradually increase the limit until a solution is found. This technique is called *iterative deepening* (Bratko 1994: 266). It can be implemented by modifying the program above in the following way. The **akko**

procedure can be called from another procedure which would, on each recursive call, increase the limit by 1. Furthermore, if the user wishes to query the knowledge base more than once, this new procedure can be invoked repetitively from another procedure using the **repeat** statement of Prolog.

Bratko (1994: 267) suggests a more elegant implementation based on a procedure

```
path(Node1, Node2, Path)
```

that generates, for the given first node, all possible acyclic paths of increasing length. This is exactly what is required by the iterative deepening approach: generate paths of increasing length until a path is generated that ends with a goal node.

A disadvantage of iterative deepening is the consequence of its main strength: on each iteration when the depth limit is increased, the paths previously computed have to be recomputed and extended to the new limit. The critics would argue, however that for every subsequent iteration, the search begins all over again, and all previous instantiation of variables will be lost, which would surely result in some loss of time. Albeit true, in typical search problems, this re-computation does not critically affect the overall computation time and, computing speeds for most IIR systems is inconsequential and regarded as trivial. This implies that the solution in the form of a *relevant* goal node hit far outweighs the almost insignificant time delay of the search.

This procedure is simple, and even if it does not do anything very clever, it does not waste much time or space. In comparison to other search strategies, such as breadth-first search, the main advantage of iterative deepening is that it requires relatively little memory space. At any point of execution, the space requirements are basically reduced to *one path* between the start node of the search and the current node. Paths are generated, checked for a hit and forgotten if none are found, which is in contrast to some other search procedures (like breadth-first search) that, during search, keep many candidate paths at the same time.

This solution is in fact quite useful in practice as long as the combinatorial complexity of the problem does not require the use of problem-specific heuristics. But, this is exactly the scenario we find ourselves in when the knowledge base of choice is a semantic network. How then, do we overcome this stumbling block?

Our solution has its basis from cognitive psychology and the findings of Cohen and Kjeldsen (1987) that conforms to the *distance* constraint that says that activation should cease at a distance of four links (i.e., five nodes) from the start node. Because our knowledge base for FUND is much smaller than that of Cohen and Kjeldsen's, we restricted our search depth to at most five links. This means that in the event no search result was found at a depth of four, then, one would be sought for at depth five. These heuristics were combined with the iterative deepening and max depth searches to form the basis of FUND's search algorithm below:

```
akko(X, QQ, Maxdepth) :-
      Maxdepth > 1,
      ako(X, ZZ),
      Max1 is Maxdepth - 1,
      akko(ZZ, QQ, Max1),write(ZZ), nl.

akko(XX, QQ, Maxdepth) :-
      Maxdepth > 0,
      ako(XX, QQ), write(QQ),nl, !.
```

Note that the order of the predicates has been reversed in order to trap the stopping case sooner and consequently, the **ako** links are displayed in reverse order. In addition, the cut (!) operator is used to prevent backtracking as soon as a hit occurs. This will prevent multiple returns for a single query to the **depth_4_or_depth_5_search**. The reason for this is, because there are only four top-most nodes and given the constraints that are adhered to and the fact that the order of the knowledge base items have been carefully

entered, the first hit for this search would be the best one. For the exclusive **depth_4_search** and **depth_5_search**, the cut operator will not be used so that the searches may produce multiple hits. The last search, namely the **exhaustive_depth_first_search** has been included as a menu option primarily for the purpose of testing and evaluation comparisons required for the next section.

7.2 Evaluation and Experimentation

Before we even begin to evaluate FUND, we need to investigate the *nature* and *process* of evaluation. Much effort and research has gone into solving the problem of evaluation of information retrieval systems. However, it is probably fair to say that most people active in the field of information storage and retrieval still feel that the problem is far from resolved. One may get an idea of the extent of the effort by looking at the volumes of numerous survey articles that have been published on the topic. Nevertheless, new approaches are constantly being published. For the benefit of this research it will be an impossible task to cover all work to date about evaluation, so, we shall attempt to explicate the conventional, most commonly used methods.

To put the problem of evaluation in perspective, we need to answer three questions: (1) Why evaluate? (2) What to evaluate? (3) How to evaluate? The answers to these questions pretty much cover the entire scope of evaluation (Van Rijsbergen 1983).

The answer to the first question is mainly a social and economic one. The social part is fairly intangible, but mainly relates to the desire to put a measure on the benefits (or disadvantages) to be derived from IR systems. We use the word "benefit" here in a wider sense than just the benefit accruing due to acquisition of relevant information. For example, what benefit will users obtain (or what harm will be done) by replacing the traditional sources of information by a fully automated and interactive retrieval system?

Studies to gauge this are ongoing but results are hard to interpret. For some kinds of retrieval systems the benefit may be more easily measured than for others.

As far as the economic answer to the first question is concerned, it relates to how much is it going to cost to use one of these systems, and coupled with this question, "is it worth it?" Even a simple statement of cost is difficult to make. The hardware costs may be easy to estimate, but the cost in terms of a personal effort are much harder to ascertain and usually speculative. Then, whether it is worth it or not, depends on the individual user.

It should now be apparent that in evaluating an IR system we are mainly concerned with providing data so that users can make a decision as to whether they want such a system (social answer) and whether it would be worth it (economic answer). Furthermore, these methods of evaluation are used in a comparative way to measure whether certain changes will lead to an improvement in performance. In other words, when a claim is made for, say a particular search strategy, the yardstick of evaluation can be applied to determine whether the claim is a valid one.

As far as FUND is concerned, it has enormous *social* and *economic* potential. Since it is an advice-rendering system that seeks to improve on the current process of matching researchers with prospective funding sources, its users may well benefit from pursuing research that is more relevant to their own interests. Secondly, from an economic perspective, FUND has the potential of being very cost effective as regards its hardware and software requirements are concerned if it were to be scaled up to a real-world application. Furthermore, if it were to be web-enabled, institutions as well as the NRF would save a lot of time, effort and resources.

The second question (what to evaluate?) boils down to what we can measure that will reflect the ability of the system to satisfy the user. In fact, as early as the 1960s, Cleverdon (1966) gave an answer to this. He listed six main measurable quantities:

- The *coverage* of the collection, that is, the extent to which the system includes relevant matter;
- The *time lag*, that is, the average time interval between the time the search request is made and the time an answer is given;
- The form of *presentation* of the output;
- The *effort* involved on the part of the user in obtaining answers to his search requests;
- The *recall* of the system, that is, the proportion of relevant material actually retrieved in answer to a search request;
- The *precision* of the system, that is, the proportion of the retrieved material that is actually relevant.

It is ***recall*** and ***precision*** that attempt to measure what is now known as the *effectiveness* of the retrieval system. In other words, it is a measure of the ability of the system to retrieve *relevant* information while at the same time holding back non-relevant ones. It is assumed that the more effective the system, the more it will satisfy the user. It is also assumed that precision and recall are sufficient for the measurement of effectiveness. With regard to the first four quantities, it is widely accepted that these are easily and readily assessed (Van Rijsbergen 1983). These quantities however, could be beneficial if the prototype were to be scaled up, as suggested in section 8.3.

Recently, there has been much debate as to whether precision and recall are in fact the appropriate quantities to use as measures of effectiveness. A popular alternative has been recall and *fall-out* (the proportion of non-relevant information retrieved). However, all the alternatives still require the determination of relevance in some way. Nevertheless, the advantages of basing relevancy on precision and recall are that they are:

- The most commonly used pair;
- Fairly well understood quantities.

As far as FUND is concerned, precision and recall will suffice as a measure of retrieval efficiency, particularly since similar systems, like GRANT (Cohen 1987), also use the same measures.

The final question (How to evaluate?) has a large technical answer. The technique for measuring retrieval effectiveness has been largely influenced by the particular retrieval strategy adopted and the form of its output. Usually, the measure of retrieval effectiveness of a particular strategy is calculated with the aid of a formula or formulae that take into consideration the various variables that influence the process and is analysed using a graph. However, before proceeding to the technical details relating to measurement of effectiveness of FUND, we need to examine more closely the concept of *relevance* that underlies it. We also examine the role of precision and recall in greater detail.

7.2.1 Relevance

Relevance is a *subjective* notion (Van Rijsbergen 1983). Different users may differ about the relevance or non-relevance of particular returns to a given query. However, the difference is not large enough to invalidate experiments that have been made with knowledge bases for which test questions with corresponding relevance assessments are available. These questions are usually elicited from bona fide users, that is, users in a particular discipline who have an information need. Relevance assessments will be made by a panel of experts in that discipline. So we now have the situation where a number of questions exist for which the correct responses are known. It is a general assumption in the field of IR that should a retrieval strategy fare well under a large number of *experimental* conditions then it is likely to perform well in an *operational* situation where relevance is not known in advance.

In the section that follows, we will tackle the role of precision and recall as a measure of retrieval effectiveness. In particular, we will investigate *how*, using formulae, these quantities will be used to determine the effectiveness of FUND.

7.2.2 Precision and Recall

Initially, we concentrate on measuring effectiveness by precision and recall; a similar analysis could be given for any pair of equivalent measures. Remember, precision refers to the ratio of the number of relevant nodes retrieved divided by the total number of nodes visited or retrieved; and, recall refers to the ratio of relevant nodes visited or retrieved divided by the total number of relevant nodes in the knowledge-base. It is helpful at this point to introduce the famous "contingency" table:

	Relevant	Non-relevant	
Retrieved	$A \cap B$	$A' \cap B$	B
Not retrieved	$A \cap B'$	$A' \cap B'$	B'
	A	A'	

Fig. 7.2 Contingency table

A large number of measures of effectiveness can be derived from this table. To list but a few:

$$\textbf{Precision} = \frac{|A \cap B|}{|B|}$$

$$\textbf{Recall} = \frac{|A \cap B|}{|A|}$$

$$\text{Fallout} = \frac{|A' \cap B|}{|B'|} \qquad \text{or} \qquad \text{Fallout} = 1.0 - \text{Precision}$$

(| | is the counting measure) (Van Rijsbergen 1983)

Since these quantities have been used extensively as a means of determining retrieval effectiveness of many IIR systems, we are confident that they will suffice for FUND as well.

Finally, having established the tools we are going to use to calculate the measure of retrieval effectiveness for FUND, we will now discuss the process involved in testing and evaluating it.

7.3 Testing and Evaluation of FUND

Given the nature of FUND, and the fact that it is only a prototype incorporating only a subset of NRF technikon programmes, it is not necessary to test our hypotheses with the same kind of evaluation methodology classically used in IR. Instead, a user study centred around some well defined tasks was regarded as a better evaluation strategy.

The results of the experimental evaluation on the effectiveness of FUND did however, incorporate measuring of precision and recall as its chief evaluation tool. The formulae established in section 7.2.2 will be used for the calculation and the results obtained by a similar IR system, namely, GRANT, which used the same formulae, will be used as a yardstick for our comparison.

The experimental evaluation of FUND was carried out to verify the following hypothesis:

- *The search and retrieval performed by FUND enables a user to find more relevant nodes (and ultimately funded themes within the NRF), including ones that are not directly related to the query, than a traditional depth-first search algorithm used for the same queries.*

To test this hypothesis, two users (namely, a Technikon research programme assistant and a prospective Masters student) were given the task of finding funding programmes and themes associated with various random topics by making appropriate keyword queries to FUND. The NRF brochure was given to both subjects and the relevant "directed" themes were highlighted from which they could draw a list of keywords for their queries. After each subject drew up their list of keywords, they were familiarised with FUND and LPA Prolog.

They were then each given an opportunity independently to query FUND with their choice of keywords. Each keyword was queried three times using the three searches afforded by FUND, namely, **`depth_4_search`**, **`depth_5_search`**, and **`exhaustive_depth_first_search`**. The first search was done to a maximum depth of 4 links, i.e., an interconnection of 5 nodes. The second search was done to a maximum depth of 5 links, i.e., an interconnection of 6 nodes. The last search was an exhaustive depth-first search for the entire knowledge base, devoid of any depth constraints. This search that is consistent with traditional searches, will be used as a yardstick measure for the other searches.

For each keyword searched (three times), once some relevant hit(s) were discovered, the user was asked to determine the relevancy of each hit with respect to the semantically linked terms in relation to the keyword or phrase queried. The user was then asked to document the results of each search in tabular form, each time denoting the total number of hits and the total number of relevant links. The decision regarding the relevancy of each hit

were made by the subjects themselves since we agree with Saracevic (1975: 321 - 343) and Hull (1993: 328 - 338) in considering relevance judgments as subjective.

The topics can be considered good examples of real user's information need since they were randomly chosen among a number of topics proposed by the subjects themselves. We are however, fully aware of the limitation of our method of evaluation. A discussion of the pitfalls of an evaluation procedure like the one we adopted is reported in Tague-Sutcliffe (1992: 467 - 490). However, because similar evaluation methods have been used for similar problem domains, such as those used for GRANT (Kjeldsen 1987), we are confident that the results tabulated below are quite meaningful and adequate in satisfying our original hypothesis.

Table 7.1 below contains the results of searches for 89 keyword phrases by both user subjects – 41 from the research assistant and 48 from the masters student.

	SEARCHES	depth_4_search		depth_5_search		Exhaustive_search	
No.	TERMS	No. of hits	No. of hits relevant	No. of Hits	No. of hits relevant	No. of hits	No. of hits relevant
1	ergonomics	1	1	1	1	1	1
2	optimization	11	8	15	9	20	11
3	separation	3	2	3	2	3	2
4	catalysis	2	2	2	2	2	2
5	chemicals	4	4	8	8	8	8
6	plastic	0	0	0	0	7	5
7	wood	3	3	4	3	10	6
8	process control	5	3	5	3	5	3
9	reliability	2	0	2	0	2	0
10	quality	3	2	4	2	4	2
11	materials	6	5	7	6	7	6
12	recycling	19	14	32	17	39	19
13	environment	0	0	4	4	6	4
14	environmental protection	7	4	30	22	50	24
15	plant	6	4	8	4	15	7
16	rapid_prototyping	1	1	1	1	1	1
17	ceramics	0	0	6	4	7	4

18	polymers	0	0	6	4	7	4
19	fire	11	2	20	6	27	8
20	bleach	1	1	4	2	10	4
21	biomedicine	1	1	1	1	1	1
22	multi-media	1	1	1	1	1	1
23	assessment	3	3	3	3	3	3
24	networks	2	2	2	2	2	2
25	forecasting	3	2	3	2	3	2
26	compression	1	1	1	1	1	1
27	measurement	1	1	3	3	3	3
28	conservation	19	12	22	13	24	14
29	economics	0	1	0	0	0	0
30	toxic_bloom	0	1	0	0	0	0
31	artificial_intelligence	1	1	1	1	1	1
32	logistic_systems	1	1	1	1	1	1
33	digital_coding	0	0	0	0	0	0
34	database_systems	1	1	1	1	1	1
35	crop_integration	1	1	1	1	1	1
36	livestock	1	1	1	1	1	1
37	farming_systems	1	1	1	1	1	1
38	soil_erosion	2	1	3	2	5	3
39	soil	2	1	4	2	4	2
40	farming_policies	0	0	3	2	12	5
41	constraints	0	0	1	1	12	8
42	food_production	3	1	8	3	19	7
43	research	3	2	12	8	12	8
44	adverse_climatic_conditions	4	3	4	3	4	3
45	geomorphology	0	0	0	0	0	0
46	additional_income	2	2	23	12	44	19
47	job_creation	2	2	20	11	40	15
48	horticultural_plants	1	1	1	1	1	1
49	transport	3	2	6	3	6	3
50	tranportation	0	0	0	0	0	0
51	preservatives	0	0	2	2	2	2
52	waste_and_sanitation	1	1	1	1	1	1
53	by-product_disposal	2	1	2	1	2	1
54	energy	1	0	1	0	1	0
55	energy-efficient	3	2	6	4	9	5
56	housing	1	1	1	1	1	1
57	affordable_housing	0	0	0	0	0	0
58	raw-materials	1	1	1	1	1	1
59	ecology	1	1	1	1	1	1
60	cephalopod	0	0	0	0	0	0
61	forestry	1	1	1	1	1	1
62	pollution	3	2	3	2	3	2
63	ecosystem	3	2	3	2	3	2
64	landscapes	2	2	2	2	2	2

65	rehabilitation	0	0	0	0	0	0
66	waste_products	1	1	1	1	1	1
67	management_of_waste	1	0	1	0	1	0
68	systems_management	1	1	1	1	1	1
69	population_dynamics	2	2	2	2	2	2
70	marine_reserves	11	6	19	9	24	9
71	habitat_degradation	1	1	1	1	1	1
72	habitat_restoration	0	0	0	0	0	0
73	coastal_living_resources	1	1	1	1	1	1
74	ocean_circulation	0	0	0	0	0	0
75	mariculture	1	1	1	1	1	1
76	stock_enhancement	2	1	2	1	2	1
77	stock_assessment	0	0	0	0	0	0
78	ecotourism	1	1	1	1	1	1
79	impacts_of_over-utilisation	0	0	0	0	0	0
80	tourist_expectations	0	0	0	0	0	0
81	cotep	1	1	1	1	1	1
82	initial_teacher_education	1	1	1	1	1	1
83	fde	1	1	1	1	1	1
84	in_service_training	3	3	3	3	3	3
85	staff_development	1	1	1	1	1	1
86	evaluation	1	0	3	2	3	2
87	scientific_literacy	1	1	1	1	1	1
88	appraisal	0	0	1	1	3	2
89	expertise_in_set	0	0	0	0	0	0
	TOTALS	**194**	**139**	**355**	**227**	**510**	**277**

Table 7.1 Subject responses from depth_4, depth_5 and exhaustive searches using FUND

7.4 Discussion of Results

It is worthwhile mentioning that FUND's knowledge base is susceptible to queries using keyword phrases that have not been catered for. From the 89 keyword phrases that were randomly chosen by the subjects, 14, that is, 16% did not appear anywhere in the knowledge base. These however, did not have any bearing with regard to the precision and recall calculations and could easily have been left out of table 6.1.

From the totals in table 6.1 we can calculate the precision and recall (or fallout rate) ability of FUND. Recall is the percentage of all the themes accepted by the subjects that FUND found. Precision is the percentage of themes found by FUND that were judged to be relevant. Fallout is the percentage of all the themes found by FUND but were judged non-relevant by the subjects:

$$\textbf{Precision} = \frac{\textbf{No. of themes judged relevant by FUND and relevant by subjects}}{\textbf{No. of themes judged relevant by FUND}}$$

$$\textbf{Recall} = \frac{\textbf{No. of themes judged good by FUND and relevant by subjects}}{\textbf{No. of themes judged relevant by subjects}}$$

$$\textbf{Fallout} = \frac{\textbf{No. of themes judged relevant by FUND, but non-relevant by subjects}}{\textbf{No. of themes judged relevant by FUND}}$$

$$= \textbf{1.0 - Precision}$$

The results of our calculations are documented in table 7.2 below:

	depth_4_search	Depth_5_search	exhaustive_search
PRECISION (%)	72	64	54
RECALL (%)	50	82	100
FALL-OUT (%)	28	36	46
Number of hits	194	355	510
Number of relevant hits	139	227	277
Number of false positives	55	128	233

Table 7.2 Statistics from depth_4, depth_5 and exhaustive_searches using FUND

At a maximum depth of 4 levels, FUND's precision is at 72% (and fallout rate is at 28%) finding only themes that are closely related to the user's query. As the depth is increased by one more level, the precision drops to 64% (and the fallout rate increases to 36%) that implies that deeper searches reveal a greater percentage of non-relevant themes. This is consistent with evaluation of GRANT (Kjeldsen & Cohen 1987) since a larger number of hits are returned at greater depths. The exhaustive (depth-first) search has a precision of only 54% that confirms our findings. Furthermore, searches at depth levels 4 and 5 revealed a number of associated programmes and themes that would not have been recalled by the experts.

Although the recall rate increases with deeper level searches, so too does the number of non-relevant associations tested as depicted by the increasing number of false positives. This implies that deeper searches reveal a number of very remotely associated nodes – many of which turn out to be non-relevant in relation to the initial query.

It was also discovered that **`depth_4_search`** was more productive in returning a number of relevant links if the keyword phrases were more subject specific, such as "marine_reserves", "conservation", etc. Generic terms such as "job_creation", "additional_income", "assessment", etc. proved to be more successful at depth levels 5 and higher. This is certainly in keeping with the real-life situation as pointed out in Chapter Five (section 5.5). This demands that an expert be very astute when it comes to making searches for funding sources if the research topic is very general or vague. However, the knowledge base of FUND contained too few generic terms to adequately substantiate the correlation hypothesised.

Lastly, it was discovered that FUND was very alert to return *most* themes that have been queried with the use of an ambiguous keyword phrase, such as "plant", that is usually forgotten by most human experts. Here again, we found that there were too few ambiguous keyword phrases within FUND to adequately test this.

7.5 Conclusion

The primary objective of this Chapter was to determine whether by implementing the distance constraint of the Constrained Spreading Activation technique within the reasoning strategy of FUND over the semantic network, would result in more *relevant* returns than a standard technique, such as the conventional depth-first search. From the testing and evaluation of FUND, we can confirm our original hypothesis (section 7.3) that FUND does produce more meaningful and relevant results than standard techniques.

However, although the results from testing of FUND have proved to be promising, we cannot claim that we have resolved our original dichotomy in full, that is, FUND did not solve the entire problem of adequately matching research proposals to appropriate funding sources. FUND has opened up avenues for new criticisms, such as how would the system cope when scaled up, will the recall and precision rates drop as the knowledge base increases, can the system be web-enabled without loss of efficacy, etc. These issues, amongst others, will be tackled in the concluding Chapter.

Nevertheless, FUND did accomplish our original aim to demonstrate that an AI technique could be used to assist in this matching process, especially from the point of view that the system can be used by anyone, and not necessarily by experts. In the concluding Chapter, we will summarise our findings concerning FUND and demonstrate what possibilities exist for further research (section 8.3).

Chapter Eight

Conclusion

8.1 Synopsis of this Dissertation

This Chapter will accentuate the major strides achieved by this dissertation by providing a synopsis of each of the preceding Chapters. In doing so, we will once again highlight the problem domain and discuss our solution from within the field of Artificial Intelligence (AI) to resolve this dichotomy. Although the performance results of our system, FUND has been comparably encouraging, we cannot profess to have totally resolved the dichotomy and, as such, we will offer some recommendations for future research efforts to resolve similar problems in the final section of this Chapter.

8.2 Achievements of this Research

For many years, researchers at various institutions have been approaching the National Research Foundation (NRF) for funding of various projects. Many have been successful at attaining funding, but a large percentage of researchers were not. We documented why this was so in our introduction from informal interviews conducted with a technikon research assistant, and demonstrated that the process of matching prospective researchers to appropriate funding sources was not as effective as it could be. From these interviews, it was also discovered that there were very few freestanding funding opportunities and the chances of any novice researcher getting such funding was unnecessarily difficult.

Usually, a researcher at a particular institution had limited opportunities of attaining funding for his/her particular area of research expertise and interest. More often than not, a researcher would be compelled to apply for research grants that were only directly related to any of the existing approved activity areas of that particular institution. Needless to say that these activity areas were very few and did not adequately represent the research interests of the NRF at large. Therefore, novice researchers had limited opportunities of pursuing research interests that were driven by inherent passion and an inner desire to do research. These individuals were usually veered into already established activity areas and this proved to be counter-productive in relation to the aims of the institution as well as the NRF.

The need for a system that would deliver better advice to prospective researchers who are seeking funding from the NRF was obvious. We were optimistic that the field of Artificial Intelligence (AI) could assist in this regard. Our approach was to investigate whether an automated Information Retrieval (IR) system could assist in providing researchers with more relevant funding advice than standard techniques used for similar problems and this, formed the crux of our research effort.

A review of the field of AI revealed that a solution was most plausible in the area of Expert Systems (ESs), and in particular, the area of Intelligent Information Retrieval (IIR). As pointed out in Chapter Five (section 5.6), an IIR system utilising a semantic network as the knowledge representation structure revealed the greatest promise in our pursuit of a solution to the problem.

In order to place research on IIR in context with IR, an overview of the field of AI with respect to Information Retrieval was conducted in Chapters Two, Three and Four. A literature survey was performed to examine the following crucial issues:

- The origins of IR and IIR (section 2.1)
- The basic components of an Expert IIR System (section 2.3)

- The current techniques employed in IIRSs (Chapter Three).
- The theory and techniques of IIR (section 4.1)
- The theory and techniques of Knowledge Representation Structures (section 4.2)

In addition, a literature survey (Westhuysen 1999) and informal interviews with a technikon research assistant were conducted regarding the current existing process used by the NRF for matching prospective researchers with funding sources at technikons, and examined with respect to its productivity and efficacy. The shortcomings of the current situation were documented in our introduction (section 1.1). This process was usually accomplished by a human expert who was restricted to choices from within the designated activity areas at a particular institution. Apart from this, the expert's precision and recall ability was solely dependent on his/her intellectual capabilities and experience. In Chapter One we highlighted in detail, the many reasons why this process was found wanting and made suggestions as to how it could be improved.

Following this, further literature surveyed in search of a solution to this problem revealed that a straightforward database system would not have been any more effective than the current manual approach. The primary reason for this was because of the lack of semantic associativity within a database system. It was extremely difficult, if not impossible, to explicitly include code within a database system to adequately reflect semantic associativity.

However, there has been a considerable amount of research effort devoted to solve similar problems by implementing an expert retrieval system. A review of the various techniques in implementing these expert systems, as documented in Chapter Five (section 5.1), revealed that our system, FUND, had to incorporate some form of association between terminology if we were to closely replicate the thought processes of a human expert. Our solution was found in the data representation technique that closely resembled Associationist's Theories (section 5.5), namely, semantic networks (section 5.6). Quillian

(1968) showed how semantic networks closely resembled the manner in which humans stored and retrieved information.

Although work on semantic networks began in the 1960s, there were very few built expert systems that incorporated this data representation technique that were being implemented in the real world. From those few that existed, such as GRANT (Cohen & Kjeldsen 1987), PROSPECTOR (Duda, Gaschnig & Hart 1979; Gaschnig 1982), parts of MYCIN (Shortliffe 1976;Buchanan & Shortliffe 1984 a & b) and DENDRAL (Lindsay, Buchanan, Feigenbaum & Lederberg 1980), we realised that FUND had to incorporate some *constraints* to prune the search in order to return relevant links only.

A review of the Constrained Spreading Activation technique (Cohen & Kjeldsen 1987; Crestani 1997; Preece 1981; Crestani & Lee 2000) demonstrated that this technique was well suited to solve our problem. We therefore reviewed existing systems such as GRANT (Cohen & Kjeldsen 1987; Kjeldsen & Cohen 1987) that utilised the Constrained Spreading Activation technique, as well as theorists on the subject such as Cohen and Kjeldsen (1987), Crestani (1997), Preece (1981), etc. Particular attention was devoted to the drawbacks and shortcomings of these systems, that were enunciated in Chapter Five (sections 5.1 and 5.3) and how FUND could overcome them.

As pointed out in the introduction, with regard to the scope and delimitation of the project (section 1.5), the four "directed-themes", identified by the NRF as their designated research areas of funding for technikons, were chosen as the application domain space to build our prototype system, FUND. These themes included Competitive Industry, Improved Quality of Life, Sustainable Environment, and Effective SET Education and Awareness that formed the highest level of our knowledge base. These were in turn linked to the various programmes contained within each of the themes, which were in turn linked to lower level sub-programmes or sub-themes and related terminology. All of these themes, programmes, sub-themes, sub-programmes and terms formed the nodes of the semantic network that

were linked to one another via one of the following association links: **programme_of**, **isa** and **ako** respectively.

The lowest levels of the links were highly interconnected because of the nature of the associations within the semantic network as well as inheritance. Thus, FUND's knowledge base accommodated *all* technikon themes and programmes, and was not restricted to only approved programmes and themes within a particular technikon, that is, it was *institution-independent*. This meant that a prospective researcher was not restricted in any way by his or her choices.

With regard to the programming language to be used for our prototype, Prolog with its exceptional backtracking facility and default backward chaining, depth-first search technique was most suited to our problem space (section 6.2). The features of Prolog that were pivotal to FUND as well as its user interface capabilities were highlighted in detail throughout Chapter Six.

FUND's search routine had to be adapted to incorporate both the inherent nature of the depth-first search of Prolog as well as the constraints imposed on the search. The primary constraint placed on the search was that of *depth-limitation* (section 6.5). The depth of the search was initially limited to a depth level of four because it was found, from past research (Cohen & Kjeldsen 1987), that further links usually returned relevant as well as non-relevant nodes. However, sometimes four links were not sufficient to return a hit for any of the programmes or themes, and because of the nature of the problem domain, it was decided to extend the search to a maximum depth of five in the event that the fourth level search was unsuccessful.

As pointed out in Chapter Six (section 6.5), the very generic highest levels, namely, the four directed themes, did not adequately give a great deal of diversification, and therefore, it was more crucial to arrive at a second level hit, namely, a programme, than a theme. Therefore, a fifth level search was necessitated in some cases. Although all previous

unifications of variables were lost prior to the fifth level search, it had no drastic effect on the efficiency of the system. However, it is difficult to predict whether this would be the case when the size of the knowledge base is scaled up to incorporate *all* the themes funded by the NRF.

In addition, as pointed out in Chapter Six (section 6.2 and section 6.3), recursion was used to aid in the transitivity of links, in particular in relation to the inheritable **ako** links, and backtracking. The algorithms supplied by Bratko (1994) were adapted to incorporate all of these features and constraints to finish our search technique.

As demonstrated in Chapter Seven with regards to the testing and evaluation of FUND (section 7.3), the results are promising in that our original aim of returning nodes that ordinary keyword searches would usually miss was achieved. The semantically linked nodes returned by FUND have a stronger relevancy than those revealed by standard Boolean search techniques, in particular an exhaustive depth-first search. At constrained searches, namely, at depths 4 and 5, FUND revealed a precision of 72% and 64%, respectively. In comparison with an exhaustive search precision of only 54%, this was highly optimistic with regard to the *relevancy* of the hits.

Although the recall ability of FUND was low, i.e., 50% for depth 4 and 82% for depth 5, it proved to be in keeping with most IIR searches and compares very favourably with GRANT (Kjeldsen & Cohen 1987). This only confirms that the deeper we search, the more hits we are bound to get, but the crucial issue is that the deeper we search, the greater the number of false positives (links that usually do not lead us to relevant hits) we pursue. The elimination of false positives would require a great deal of problem-specific heuristics to be incorporated within the system and is left as a future research initiative, particularly if it were to be scaled up to a real-world application.

From the optimistic results obtained from testing of the prototype, that were documented in Chapter Seven (section 7.4), we demonstrated successfully that an AI approach, in

particular an expert IIR system, *could* be implemented successfully to provide better funding advice to prospective researchers seeking funding from the NRF, thereby achieving our original objective.

Nevertheless, it is still unknown how FUND will react to scaling-up of the knowledge base. Current performance however, suggests that although the complexity of the increased interconnectivity will add to the time delay during scaling, the performance will not diminish and the time-loss will most likely be inconsequential.

The semantic network data representation technique proved to be most beneficial in that no known standard technique would have been able to link some of the remotely related proposal terms to the existing themes of the NRF. Furthermore, as demonstrated in Chapter Six (section 6.4), this technique was highly adaptable and made scaling-up of the knowledge base relatively easy. Any new programmes or themes could easily be linked to the existing semantic network structure without having to redo many of the higher levels. Terms could be linked to existing ones or new bottom-level links could be added onto the network by simply using one of the existing links, namely **ako** or **isa**.

Examples demonstrating the ease of scaling-up are provided in Chapter Six (section 6.4). The pre-condition to scaling-up however, is that this has to be accomplished by someone who has been using the system, preferably an expert who is already familiar with the existing network. This is necessary because of the subjectivity of the semantic links already established and to achieve consistency of results. It must be pointed out that FUND need not be used only by experts, but could be used by anyone that wishes to search for relevant funding sources.

In terms of accuracy of solutions produced, we concluded in Chapter Seven (section 7.3) that the themes returned via the Constrained Spreading Activation technique were highly *relevant* with regard to the initial query. However, certain performance issues such as the order of importance of returned themes and biased links were possibly degraded due to data

inconsistencies and subjectivity of the system designer. Optimistic results however, were demonstrated with regard to dealing with synonyms and ambiguity.

In terms of validation of FUND, the results of experiments in Chapter Seven demonstrated that its implementation performance was not significantly different to that of the research assistant's results with regard to common generic terms, but there was a significant performance improvement with specific terms or areas of specialisation (section 7.4). For instance, from the research assistant's perspective, there was no immediate significance between "conservation" and "marine biology" because, conservation is frequently thought of as a land issue first. FUND however, was able to link these issues successfully and return nodes that were relevant due to these links.

With regard to retrieval time results, FUND outperformed the human expert and, performed exhaustive searches as opposed to a, usually incomplete, human search. It must be noted however, that the objective of this research did not focus on the retrieval times, but rather concentrated on the accuracy and efficacy of the search technique employed. FUND's principal aim was not to replace the human expert, but rather to assist him/her, or any other user in pursuit of relevant funding sources, with better advice.

In conclusion, the purpose of this research was to investigate whether an IIR system, to solve the dichotomy that existed in the matching of prospective researchers with relevant funding sources, could be more effective than the current process. It has been successfully demonstrated that a search technique using a semantic network as the data representation structure and the distance constraint of the Constrained Spreading Activation technique was adequate to solve our problem. From an application point of view, FUND's design was based on established IIR principles and implemented in a suitable and significant form in terms of matching prospective research proposals with funded themes of the NRF. Results indicate that the performance of the techniques utilised matches the theoretical basis for its design and is satisfactory in comparison to other IIR research undertaken to bridge the gap between reasoning in human beings and machines. The overall conclusion of this research

dissertation has sown the seeds of a promising area and forms the foundation upon which future research work can be accomplished.

8.3 Future Research

Although FUND was successful in attaining our original objective, we must concede that it did not solve the NRF problem in its entirety. However, this was not our intention and the principal aim of this research effort was to explore the possibility of an AI approach to partly resolve this problem. FUND demonstrated that such a solution could be found by implementing the Constrained Spreading Activation search technique over a semantic network. Future research efforts along the same lines as FUND could result in a workable real-life application, but many other issues need to be investigated before a practical solution is imminent.

Following the optimistic results of FUND, there are at least four research areas that require further investigation. The most obvious of which is the question of how the system will cope to scaling-up of the knowledge base to incorporate *all* the themes funded by the NRF across *all* institutions. The added challenge would be to eliminate as many false positives as possible in pursuit of relevant hits as the system is scaled-up. Much effort with this regard has already been devoted in other systems such as GRANT (Cohen & Kjeldsen 1987 & Kjeldsen & Cohen 1987), etc. and, researchers are optimistically pursuing new techniques to accomplish this.

Also, the responses that FUND can provide need to be more accessible to people seeking funding. Future research could also investigate the possibility of making this system web-enabled and encourage on-line funding applications without any institutional prejudice. In keeping with this, future researchers could also tackle the problem of making the user-interface more interactive and allow the user, with the aid of question-and-answer type probing, further assist in the reasoning process of the search. Many researchers have

already demonstrated that the greater the interaction with the user, the more refined and streamlined the search process becomes and the results are invariably more relevant. In order to accomplish a highly "intelligent" Human-Computer Interface (HCI), one would have to pursue investigations in areas such as Natural Language Processing (NLP) (section 3.6) and ultimately query reformulation.

The task of ridding the search process of any form of bias, especially with respect to subjectivity of the system designer(s), as well as the complexity of handling synonyms and ambiguity are also areas of concern that future research can address. This might be difficult because it may require an investigation of Natural Language Processing (NLP) as well, and the assistance of other systems, such as CYC (Lenat, Prakash & Shepherd 1986), to arrive at a standardized platform that all searches could utilize as a front-end. In addition, the inclusion of an indexing mechanism will also assist in favouring or disfavouring paths within the semantic network, thereby increasing the "degree of relevancy" factor. However, no indexing technique is totally free of a subjectivity factor, and therefore, it is our suggestion that a group of system designers will be better suited to resolve this dilemma than a single designer, if a reasonable degree of consistency is to be achieved. As a final comment regarding the subjectivity issue, we suggest that a more rigorous range of evaluation techniques be employed to strengthen the legitimacy of the performance results.

Finally, we cannot profess that the use of a semantic network as a knowledge representation structure is the *best* technique for problem spaces of this nature. More recently, considerable research activity has been expended on meshing a semantic network with other structures, such as frames, to form hybrid systems and these have demonstrated optimistic results. The Constrained Spreading Activation search technique has also demonstrated that it is an extremely viable reasoning scheme for most IIR systems and therefore is under continual focus for improvement. Although the implementation of a combination of constraints adds to the complexity of the problems, they still add a great deal of heuristic knowledge to the system that will eventually lead to a greater retrieval relevancy.

References

Aldous, KJ 1996. *A system for the automatic retrieval of information from a specialist database.* Information Processing and Management. Vol. 32, no. 2: 139 – 154.

Bates, MJ 1996. *The Getty end-user online searching project in the humanities: Report no. 6: Overview and conclusions.* College and Research Libraries. Vol. 57: 514 – 523.

Belkin, NJ 1980. *Anomalous states of knowledge as a basis for information retrieval.* Canadian Journal of Information Science. Vol. 5: 133 – 143.

Belkin, NJ, Oddy, RN & Brooks, HM 1982. *ASK for Information Retrieval 1 : Background and theory.* Journal of Documentation. Vol. 38 (2): 61 – 71.

Belkin, NJ, Seeger, T & Wersig, G 1983. *Distributed expert problem treatment as a model for information system analysis and design.* Journal of Information Science. Vol. 5: 153 – 167.

Bishop, A & Starr, SL 1996. *Social Informatics of digital library use and infrastructure.* In: Williams, ME. Annual Review of information science and technology. Vol. 3. Medford, NJ: Information Today: 301 – 401.

Black, WJ 1986. *Intelligent Knowledge Based Systems : An Introduction.* Van Nostrand Reinhold Co Ltd.(UK). Berkshire, England.

Bookstein, A 1985. *Probability and fuzzy-set applications to information retrieval.* In : *Annual Review of Information Science and Technology.* M.E. Williams, 1985 ed. Knowledge Industry Publications, Inc. White Plains, N.Y. Vol. 20: 117 – 151.

Borgman, CL 1996. *Why are online catalogues still hard to use?* Journal of the American Society for Information Science. Vol 47, no. 7: 493 – 503.

Borgman, CL, Hirsh, SG & Hiller, J 1996. *Rethinking online monitoring methods for information retrieval systems: from search product to search process.* Journal of the American Society for Information Science. Vol. 47, no. 7: 568 – 583.

Bratko, I 1994. PROLOG : *Programming for Artificial Intelligence* . 2nd ed. Addison-Wesley Publishing.

Brooks, HM 1987. *Expert systems and intelligent information retrieval.* Information Processing and Management. Vol. 23, no. 4: 367 – 382.

Brooks, HM, Daniels, PJ & Belkin, NJ 1985. *Problem description and user models : developing an intelligent interface for document retrieval systems.* In: *Informatics 8 : advances in intelligent retrieval.* 16 – 17 April 1985. Oxford, England, London: Aslib: 191 – 214.

Buchanan, B 1982. *New research on expert systems.* In: Hayes, J, Michie, D & Pao, Y eds. Machine intelligence. Vol.10. New York. Wiley: 269-300.

Buchanan, B & Feigenbaum, EA 1978. *DENDRAL and METADENDRAL: their applications dimensions.* Artificial Intelligence. Vol.11: 5 – 24.

Buchanan, BG & Shortliffe, EH 1984 (a). *Explanation as a topic of AI reasoning.* In: Buchanan, BG & Shortliffe, EH eds. *Rule-based expert systems.* Reading, MA : Addison-Wesley: 331 – 337.

Buchanan, BG & Shortliffe, E H 1984 (b). *Rule-based expert systems : the MYCIN experiments of the Stanford Heuristic Programming project.* Reading, MA . Addison-Wesley.

Carballo, JP & Strzalkowski, T 2000. *Natural Language Information Retrieval: Progress Report.* Information Processing and Management. Vol. 36: 155 – 178.

Chen, H 1994. *Machine Learning for Information Retrieval: Neural Networks, Symbolic Learning, and Genetic Algorithms.* http://ai.bpa.arizona.edu/papers/mhr93/mhr93.html.

Chris, M & Gerald, G 1989. *Natural Language Processing : An Introduction to computational Linguistics.* Addison Wesley Publishing Company. Kent, Great Britain.

Chrisment, C. et al 1997. *Knowledge Extraction and Synthesis form Heterogenous Collection.* Vol. 5., no. 3: 367 – 400.

Chu, H & Rosenthal, M 1996. *Search engines of the World Wide Web: A comparative study and evaluation methodology.* In: Proceedings of the 59th ASIS annual meeting, Medford, NJ: Information Today: 127 – 135.

Claudia, M 1986. *Prolog Programming : Applications for database systems and natural language systems.* Addison Wesley Publishing Company. Reading, Massachusetts.

Cleverdon, CW, Mills, J & Kleen, M 1966. *Factors determining the performance of Indexing Systems. ASLIB Cranfield Project.* Vol. 1, Design. Vol. 2. Test Results. Cranfield.

Cohen, PR & Kjeldsen, R 1987. *Information retrieval by constrained spreading activation in semantic networks.* In: Information Processing and Management. Vol. 23, no. 4: 255 – 268.

Crestani, F 1997. *Applications of spreading activation techniques in Information Retrieval.* Artificial Intelligence Review. Vol.11 (6): 453 – 582.

Crestani, F & Lee, PL 2000. *Searching the web by constrained spreading activation.* Information Processing and Management. Vol. 36: 585 – 605.

Croft, WB 1981. *Document representation in probabilistic models of information retrieval.* Journal of the American Society of Information Science. Vol. 32: 451 – 457.

Croft, WB 1987. *Approaches to Intelligent Information Retrieval.* Information Processing and Management. Vol. 23, no. 4: 249 – 254.

Croft, WB 1986. *User-specified domain knowledge for document retrieval.* Proceedings of the ACM SIGIR international conference on research and development in information retrieval. Rabbitti, F ed. Pisa, Italy : 201 – 206.

Croft, WB 2001. *What do people want from Information Retrieval? The top ten research issues for companies that use and sell IR systems.*

Croft, WB 2002. *Overviews of Research Areas.* http://ciir.cs.umass.edu/research/index.html.

Croft, WB & Belkin, NJ 2001. *Supporting Effective Access through User-and Topic-based Language models.* http://ciir.cs.umass.edu/research/mongrel.html.

Croft, WB & Lewis, D 1987. *An approach to natural language processing for document retrieval.* Proceedings of the ACM SIGIR international conference on research and development in information retrieval.

Croft, W, Lucia, T & Cohen, P 1988. *Retrieving Documents by plausible inference: a preliminary study.* In: Proceedings of ACM SIGIR, Grenoble, France.

Croft, W, Lucia, T, Crigean, J & Willet, P 1989. *Retrieving documents by plausible inference: an experimental study.* Information Processing and Management. Vol. 25, no. 6: 599 – 614.

Croft, W & Thompson, R 1987. I^3R: *A new approach to the design of Document Retrieval Systems.* Journal of the American Society for Information Science. Vol. 38, no. 6: 389 – 404.

Dennis, M 1989. *Building expert systems in Prolog.* Springer Compass International.

Doyle, LB 1965. Is automatic classification a reasonable application of statistical analysis of text?. Journal of the ACM. Vol. 12: 473 – 489.

Ding, W & Marchionini, G 1996. *A comparative study of Web search service performance.* In: Proceedings of the 59th ASIS annual meeting, Medford, NJ: Information Today: 136 – 142.

Duda, R, Gaschnig, J & Hart, P 1979. *Model design in PROSPECTOR consultant system for mineral exploration.* In : Michie, D ed.. *Expert systems in the microelectronic age.* Edinburgh. Edinburgh University Press: 153 – 167.

Duda, RO, Hart, R, Nilsson, N J & Sutherland, G 1978. *Semantic network representation in rule-based inference systems.* In: Waterman, D & Hayes-Roth, F ed. *Pattern-directed inference systems.* New York. Academic Press: 203-221.

Efraim, T 1992. *Expert systems and Applied Artificial intelligence.* Macmillan Publishing Company. New York.

Elaine, R & Kevin, K 1991. *Artificial Intelligence.* 2nd ed. Mc-Graw Hill Inc.: 251 – 300.

Fahlman, S, Touretzky, D & Van Roggen, W 1981. Cancellation in Parallel Semantic Network. In: Proceedings of the International Joint Conference on Artificial Intelligence: 257 – 263.

Fairthorne, RA 1961. *The mathematics of classification. Towards information retrieval.* Butterworths. London: 1 –10.

Feigenbaum, EA 1979. *Themes and case studies of knowledge engineering.* In Michie, D ed. *Expert systems in the micro-electronic age.* Edinburgh . Edinburgh University Press: 3 – 25.

Feigenbaum, E, Buchanan, B & Lederberg, J 1971. *On generality and problem-solving : a case study using the DENDRAL Program.* In : Meltzer, B & Mitchie, D eds., Machine Intelligence. Vol. 6, New York. Elsevier: 165 – 170.

Fikes, RE & Hendrix, G 1977. *A network-based knowledge representation and its natural deduction system.* Proceedings of the fifth International Joint Conference on Artificial Intelligence: 235 – 246.

Fogel, DB 1994. *An introduction to simulated evolutionary optimisation.* IEEE Transactions on Neural Networks, Vol. 5, no. 1: 3 – 14.

Forgy, CL 1982. *RETE: A fast algorithm for the many pattern / many object pattern match problem.* Artificial Intelligence. Vol. 19: 17 –37.

Fox, EA 1987. *Development of the CODER system: A testbed for AI methods in Information Retrieval.* In: Information Processing and Management, Vol. 23, no. 4: 341 – 366.

Fox, EA 1989. *I. Introduction: Research and development of information retrieval models and their applications.* Information Processing and Management. Vol. 25, no.1: 1 – 5.

Fuhr, N 1999. *Towards data abstraction in networked information retrieval systems.* Information Processing and Management. Vol. 35: 101 – 119.

Fuhr, N & Pfeifer, U 1994. *Probabilistic information retrieval as a combination of abstraction, inductive learning, and probabilistic assumptions.* ACM Transactions on Information Systems, Vol. 12, no. 1: 92 – 115.

Fung, R & Favero, BD 1995. *Applying Bayesian networks to information retrieval.* Communications of the ACM. Vol. 38, no. 3: 42 – 48.

Gallant, SI 1998. *Connectionist Expert System.* Communications of the ACM. Vol. 31, no. 2: 152 – 169.

Gaschnig, J 1982. *Application of the PROSPECTOR system to geological exploration problems.* In : Hayes, JH, Michie, D & Pao, Y eds. Machine Intelligence . Vol. 10. New York. Wiley: 301 –323.

George, FL & William, AS 1997. *Artificial Intelligence: Structures and Strategies for complex problem solving.* Addison Wesley Inc. Reading, Massachusetts.

Giarratano, JC & Gary, R 1989. *Expert Systems : Principles and Programming.* PWS-Kent Publishing Company: 68 – 84.

Glass, RL 1995. *A structured-based critique of contemporary computing research.* Journal of Systems Software. Vol. 28: 3 – 7.

Good, IJ 1958. *Speculations concerning information retrieval.* Research report PC-78. IBM Research Centre. Yorktown Heights, New York.

Gordon, MD 1988. *Probabilistic and genetic algorithms for document retrieval.* Communications of the ACM. Vol. 31, no. 10: 1208 – 1218.

Gordon MD 1991. *User-based document clustering by re-describing subject descriptions with a genetic algorithm.* Journal of the American Society for Information Science. Vol. 42, no. 5: 311 – 322.

Gordon, MD & Pathak, P 1999. *Finding information on the World Wide Web: the retrieval effectiveness of search engines.* Information Processing and Management. Vol. 35: 141 – 180.

Gorry, GA, Silverman, H. & Pauker, SG 1978. *Capturing clinical expertise : a computer program that considers clinical responses to digitalis.* American Journal of Medicine. Vol. 64: 452 – 460.

Greene, DP & Smith, SF 1998. *COGIN: Symbolic induction with genetic algorithms.* In: Proceedings of the Tenth National Conference on Artificial Intelligence (AAAI): 111 –116.

Harp, S, Samad, T & Guha A 1998. *Towards genetic synthesis of neural networks.* In: Proceedings of the Third International Conference on Genetic Algorithms. Morgan Kaufmann Publishers, Inc., San Mateo, CA.

Hayes, PJ 1973. *The frame problem and related problems in artificial intelligence.* In: *Artificial and human thinking*, ed. Elithorn, A & Jones D. San Francisco. Jossey-Bass.

Hayes-Roth, F, Waterman, DA & Lenat, DB 1983. *An overview of expert systems: Chapter 1 : Building expert systems.* Reading, MA : Addison-Wesley: 3 – 29.

Hull, D 1993. *Using statistical testing in the evaluation of retrieval experiments.* In: Proceedings of ACM SIGIR. Pittsburgh. P.A. USA: 328 – 338.

Kidd, AL 1985. *What do users ask? Some thoughts on diagnostic advice. .* In : Merry, M., ed. Research and development in expert systems 85 : proceedings of the conference on the BCS specialist group on expert systems. 17-19 December 1985. University of Warwick, Warwick, England. Cambridge, England. Cambridge University Press: 9 - 19.

Kitano, H 1996. *Emperical studies on the speed of convergence of neural network training using genetic algorithms.* In: Proceedings of the Eight National Conference on Artificial Intelligence (AAAI): 789 – 795.

Kjeldsen, R & Cohen, R 1987. *The Evolution and Performance of the GRANT System.* IEEE Expert Systems. Special Feature. CAIA 87. Summer 1987: 73 – 79.

Kunz, JC, Fallat, RJ, McClung, DH, Osborn, JJ, Votteri, RA, Nii, HP, Aikins, JS & Fagan, LMA 1978. *A physiological rule-based system for interpreting pulmonary function test results.* Report No. HPP (Heuristic Programming Project). Stanford, CA: Computer Science Department. Stanford University.

Lenat, D, Prakash, M & Shepherd, M 1986. *CYC: Using Common Sense Knowledge to Overcome Brittleness and Knowledge Acquisition Bottlenecks.* AI Magazine. Vol. 6: 65 – 85.

Levesque, HJ & Brachman, RJ 1985. *Readings in knowledge representation.* Los Altos, CA . Morgan-Kaufmann. Eds.

Lindsay, RK, Buchanan, BG, Feigenbaum, EA & Lederberg, J 1980. *Applications of Artificial Intelligence for organic chemistry: The DENDRAL project.* New York. McGraw-Hill.

Lokendra, S 1988. *Semantic Networks: An evidential formalisation and its connectionist realisation.* Morgan Kaufmann Publishers Inc., San Mateo, California, USA.

Lucas, R 1996. *Mastering Prolog.* University College London Press Limited. London: 157 – 228.

Luhn, HP 1957. *A statistical approach to mechanised encoding and searching of library information.* IBM Journal of Research and Development. Vol. 1: 309 – 317.

Maron, ME, Kuhns, JL 1960. *On relevance, probabilistic indexing and information retrieval.* Journal of the ACM. Vol. 7: 216 – 244.

McCune, BP, Tong, RM, Dean, JS & Shapiro, DG 1985. *RUBRIC: a system for rule-based information retrieval.* IEEE Transactions on Software Engineering. SE-11: 939 – 944.

McDermott, J 1981. R1 : *Formative years.* Artificial Intelligence Magazine. Vol. 2 (2): 21 – 29.

McDermott, J 1982. *R1: A rule-based configurer of computer systems.* Artificial Intelligence, Vol.19: 39-88.

Michael, G & Praveen, P 1999. *Finding information on the WWW: the retrieval effectiveness of search engines.* Information Processing and Management. Vol. 35: 141-180.

Minsky, M 1975. *A framework for Representing Knowledge.* In: The Psychology of Computer Vision, ed. Patrick WH. McGraw-Hill: 211 – 217.

Mladenic, D & Stefan, J 1999. *Text-learning and related Intelligent Agents: A survey.* IEEE Intelligent Systems. July/August 1999: 44 – 61.

Mothe, J, Dkaki, T & Dousset, B 1997. *Mining Information to Extract Hidden and Strategic Information.* Proceedings: Computer-Assisted Information Searching on Internet (RIAO97), C.I.D., Paris: 32 – 51.

Muller, RA 1985. *A "standard" for expert system terms.* BCS Specialist Group on Expert Systems Newsletter, Vol. 13: 7.

Neil, AS 1987. *Cognitive Science : An Introduction.* The MIT Press.
Nilsson, NJ 1980. *Principles of Artificial intelligence.* Palo Alto. CA: Morgan Kaufmann: 137 – 142.

Oard, DW & Resnik, P 1999. *Support for interactive document selection in cross-language information retrieval.* Information Processing and Management. Vol. 35: 363 – 379.

Pauker, S, Gorry, GA, Kassirer, JP & Schwartz, WB 1976. *Towards the simulation of clinical cognition.* American Journal of Medicine. Vol. 60: 981 – 996.

Pollock, A & Hockley, A 1997. *What's wrong with Internet searching.* D-Lib Magazine, Vol. 3: 1 – 5. http://www.dlib.org/march97/bt/03pollock.html.

Pople, HE 1982. *Heuristic methods for imposing structure on ill-structured problems: the structuring of medical diagnosis* . In: Szolovits, P ed. Artificial Intelligence in medicine. Boulder, CO. Westview Press.

Petry, F, Buckles B., Prabhu, D & Kraft, D 1993. *Fuzzy information retrieval using genetic algorithms and relevance feedback.* In: Proceedings of the ASIS Annual Meeting: 122 – 125.

Preece, S 1981. *A Spreading Activation model for Information Retrieval.* Phd Thesis, Department of Computing Science. University of Glasgow, Glasgow, Scotland.

Quillian, MR 1968. *Semantic Memory.* In : Semantic Information Processing, ed., Marvin Minsky. MIT Press: 227 – 270.

Raghavan, VV & Wong, SKM 1986. *A critical analysis of vector space model in information retrieval.* Journal of the American Society of Information Science. Vol. 37: 279 – 287.

Robert, JS 1990. *Artificial Intelligence : An Engineering Approach.* McGraw-Hill Inc.: 156 – 297.

Rumelhart, D & Norman, D 1983. *Representation in memory.* Technical Report. Department of Psychology and Institute of Cognitive Science, UCSD La Jolla, USA.

Rumelhart, D, Widrow, B & Lehr, MA 1994. *the basic ideas in neural networks.* Communications of the ACM, Vol. 37, no. 3: 87 – 92.

Salton, G 1972. *Paper given at the 1972 NATO Advanced Study Institute for on-line mechanised information retrieval systems.*

Salton, G & Buckley, C 1988. *On the use of spreading activation methods in automatic information retrieval.* SIGIR: 147 –160.

Salton, G & McGill, MH 1983. *Introduction to modern information retrieval.* New York . McGraw-Hill: 142 – 177.

Saracevic, T 1975. *Relevance: a review and a framework for thinking on the notion in information science.* Journal of the American Society for Information Science. Vol. 26. 1975: 321 – 343.

Shastri, L 2001. *Why Semantic Networks?* In: *Principles of semantic Networks: Explorations in the Representation of Knowledge.* Morgan Kaufmann Publishers, Inc., San Mateo, CA.

Shneiderman, B, Byrd, D & Croft, WB 1997. *Clarifying search: A user-interface framework for text searches.* D-Lib Magazine, Vol. 1: 1 – 18. http://www.dlib.org/dlib/january97/retrieval/01shneiderman.html.

Shortliffe, EA 1976. *Computer-based medical consultation : MYCIN.* New York : Elsevier.

Simon, H 1991. *Artificial Intelligence: Where has it been, and where is it going?* IEEE Transactions on Knowledge and Data Engineering, Vol. 3, no. 2: 128 – 136.

Smith, P 1990. *Expert system development in Prolog and Turbo Prolog.* SIGMA Press. England: 133 – 244.

Spark, JK 1971. *Automatic Keyword Classification for Information Retrieval.* Butterworths. London.

Sparck, JK 1983. *Report on a visit to the U.S. 27.1.83 – 4.2.83.* British Library Research and Development Report 5762. London, British Library.

Sparck, JK 1991. *The role of Artificial Intelligence in Information Retrieval.* Journal of the American Society of Information Science. Vol. 42, no. 8: 558 – 565.

Sparck, JK, Walker, S & Robertson, SE 2000. *A probabilistic model of information retrieval: development and comparative experiments. Part 1.* Information Processing and Management, Vol 36, no 36: 779 – 808.

Spink, A & Qin, J 2000. *Editorial: Introduction to the special issue on web-based information retrieval research.* Information Processing and Management. Vol. 36: 205 – 206.

Steven, HK 1991. *Knowledge Systems through PROLOG : An Introduction.* Oxford University Press, Inc., New York. Oxford.

Stevens, ME, Guiliano, VE & Heilprin, LB 1964. *Statistical Association Methods for Mechanised Documentation.* National Bureau of Standards. Washington.

Stiles, HF 1961. *The association factor in information retrieval.* Journal of the ACM. Vol. 8: 271 – 279.

Swartout, W 1977. *A digitalis therapy advisor with explanations.* In: Proceedings of the 5th international conference on artificial intelligence. 22-25 August 1977. Cambridge, MA, Los Altos, CA : William Kaufmann.: 819-823.

Syu, I & Lang, SD 2000: *Adapting a diagnostic problem-solving model to information retrieval.* Information Processing and Management. Vol. 36: 313 – 330.

Tague-Sutcliffe, J 1992. *The pragmatics of information retrieval experimentation revisited.* Information Processing and Management. Vol. 28, no. 4: 467 – 490.

Touretzky, D & Hinton, GE 1998. *A distributed connectionist production systems.* Cognitive Science. Vol. 12, no. 3: 423 – 466.

Townsend, C 1987. *Mastering expert systems with Turbo Prolog.* Howard W. Sams & Co.

Turtle, H & Croft, WB 1991. *Evaluation on an inference network-based retrieval model.* ACM Transactions on Information Systems. Vol. 9 (3): 187 – 222.

Van Rijsbergen, CJ 1983. *Information Retrieval*. 2nd ed.. London . Butterworths .

Van Rijsbergen, CJ 1986. *A new theoretical framework for information retrieval.* In: Proceedings of the 1986 ACM conference on research and development in information retrieval. Pisa, Italy: 194 – 200.

Wang, P & Pouchard, L 1997. *End-user searching of Web resources: Problems and Implications*. In: Advances in Classification Research: Proceedings of the 8th ASIS SIG/CR Classification Research Workshop, Washington, DC: 73 – 85.

Wang, P, Hawk, WB, Tenopir, C 2000. *Users' interaction with World Wide Web resources: an exploratory study using holistic approach.* Information Processing and Management. Vol. 36: 229 – 251.

Weiss, S, Kulikowski, C & Safir, A 1978 (a). *A Glaucoma consultation by computer.* Computers in Biology and Medicine. Vol. 1: 25 – 40.

Weiss, S, Kulikowski, C & Safir, A 1978 (b). *Model-based method for computer-aided medical decision making.* Artificial Intelligence. Vol. 11(1): 145 – 172.

Weiss, S & Kulikowski, C 1991. *Computer Systems that learn: Classification and prediction methods from statistics, neural networks, machine learning, and expert systems.* Morgan Kaufmann Publishers, Inc., San mateo, CA.

Westhuysen, A 1999. *NRF Guide to Research Support: Technikon Programmes*: 26 – 85.

Winston, PH 1984. *Artificial Intelligence.* 2nd ed.. Addison-Wesley Publishers.

Wong, SKM & Yao, YY 1989. *A probability distribution model for information retrieval.* Information Processing and Management. Vol. 25, no. 1: 39 – 53.

Yang, J, Korfhage, RR & Rasmussen, E 1993. *Query improvement in information retrieval using genetic algorithms: a report on the experiments of the TREC project.* In: Text Retrieval Conference (TREC-1): 31 – 58.

Appendix A

Listing of FUND

```
/*
                              F  U  N  D

   Search - using Semantic networks with depth constraints - S. Hansraj,
   01/2000-06/2001
   ---------------------------------------------------------------------

   This program is designed to help you find an NRF programme that will
   most likely fund your research proposal by entering in keywords of
   your proposal. The normal interface to the programs is via the front-
   end relation 'menu', which is called as follows:

      ?- menu.

   The program implements a knowledge-base in the form of a semantic
   network. Searches, that are constrained by a selection of depth
   limitations, are made on the Semantic net.
*/

/******* START ******/

/*************************************************************************/
/*                                                                       */
/*                        THE KNOWLEDGE-BASE                             */
/*                                                                       */
/*************************************************************************/


/***** THEMES *****/


theme(competitive_industry).
theme(improved_quality_of_life).
theme(sustainable_environment).
theme(effective_set_education_and_awareness).


/***** PROGRAMME_OF *****/


programme_of(primary_resource_benefication, competitive_industry).
programme_of(manufacturing_advancement, competive_industry).
```

```
programme_of(information_and_infrastructure_systems,
competitive_industry).
programme_of(rural_and_urban_development, improved_quality_of_life).
programme_of(food_production_and_food_security,
improved_quality_of_life).
programme_of(inland_resources, sustainable_environment).
programme_of(marine_and_coastal_resources, sustainable_environment).
programme_of(innovation_and_change_in_education,
     effective_set_education_and_awareness).
programme_of(preparation_and_development_of_educators,
     effective_set_education_and_awareness).
programme_of(public_understanding_of_set,
     effective_set_education_and_awareness).


/***** ISA *****/


isa(minerals_processing, primary_resource_benefication).
isa(forestry, primary_resource_benefication).
isa(raw-materials, primary_resource_benefication).


isa(discrete_manufacturing, manufacturing_advancement).
isa(process_manufacturing, manufacturing_advancement).
isa(pharmaceuticals, manufacturing_advancement).
isa(biomedicine, manufacturing_advancement).


isa(information_technology, information_and_infrastructure_systems).
isa(communications_systems, information_and_infrastructure_systems).
isa(industrial_electronics, information_and_infrastructure_systems).
isa(control_systems, information_and_infrastructure_systems).
isa(logistic_systems, information_and_infrastructure_systems).


isa(farming_systems, food_production_and_food_security).
isa(sustainable_use_of_natural_resources_for_food_production,
                              food_production_and_food_security).
isa(animal_production_and_management, food_production_and_food_security).
isa(crop_and_horticultural_production_and_management,
      food_production_and_food_security).
isa(crop_and_horticultural_production,
food_production_and_food_security).
isa(crop_and_horticultural_management,
food_production_and_food_security).
isa(innovation_in_the_food_and_beverage_industry,
      food_production_and_food_security).


isa(water_and_sanitation, rural_and_urban_development).
isa(water, rural_and_urban_development).
isa(sanitation, rural_and_urban_development).
isa(waste_and_pollution_management, rural_and_urban_development).
```

```
isa(waste_management, rural_and_urban_development).
isa(pollution_management, rural_and_urban_development).
isa(housing, rural_and_urban_development).
isa(transportation_issues, rural_and_urban_development).
isa(energy, rural_and_urban_development).
isa(land-use_and_rehabilitation, rural_and_urban_development).
isa(land-use, rural_and_urban_development).
isa(land_rehabilitation, rural_and_urban_development).
isa(community-based_industrialisation, rural_and_urban_development).


isa(biodiversity_and_conservation, inland_resources).
isa(biodiversity, inland_resources).
isa(indigenous_species_utilisation, inland_resources).
isa(systems_management, inland_resources).
isa(restoration_and_rehabilitation, inland_resources).
isa(land_rehabilitation, inland_resources).
isa(land_restoration, inland_resources).
isa(weather_and_climate, inland_resources).
isa(weather, inland_resources).
isa(climate, inland_resources).
isa(ecotourism, inland_resources).


isa(coastal_communities_and_living_resources,
marine_and_coastal_resources).
isa(coastal_communities, marine_and_coastal_resources).
isa(coastal_living_resources, marine_and_coastal_resources).
isa(the_coast_as_a_resource, marine_and_coastal_resources).
isa(offshore_living_resources_and_society, marine_and_coastal_resources).
isa(offshore_living_resources, marine_and_coastal_resources).
isa(offshore_society, marine_and_coastal_resources).
isa(mariculture, marine_and_coastal_resources).
isa(biodiversity_and_conservation, marine_and_coastal_resources).
isa(biodiversity, marine_and_coastal_resources).
isa(ocean_dynamics_and_coastal_geomorphology,
marine_and_coastal_resources).
isa(ocean_dynamics, marine_and_coastal_resources).
isa(coastal_geomorphology, marine_and_coastal_resources).
isa(weather_and_climate, marine_and_coastal_resources).
isa(weather, marine_and_coastal_resources).
isa(climate, marine_and_coastal_resources).
isa(ecotourism, marine_and_coastal_resources).
isa(afrotourism, marine_and_coastal_resources).


isa(curriculum_development, innovation_and_change_in_education).
isa(assessment_criteria, innovation_and_change_in_education).
isa(cost-effective_and_quality_delivery_of_teaching,
      innovation_and_change_in_education).
isa(cost-effective_teaching, innovation_and_change_in_education).
isa(quality_delivery_of_teaching, innovation_and_change_in_education).
isa(language_and_learning, innovation_and_change_in_education).
isa(language, innovation_and_change_in_education).
```

```
isa(learning, innovation_and_change_in_education).


isa(initial_teacher_education, preparation_and_development_of_educators).
isa(teacher_education, preparation_and_development_of_educators).
isa(professional_development, preparation_and_development_of_educators).
isa(staff_development, preparation_and_development_of_educators).
isa(teachers, preparation_and_development_of_educators).
isa(teaching, preparation_and_development_of_educators).


isa(scientific_literacy, public_understanding_of_set).
isa(evaluations, public_understanding_of_set).
isa(set_communication, public_understanding_of_set).



/******** AKO ********/


ako(mining_optimisation, minerals_processing).
ako(mining_technology, minerals_processing).
ako(optimisation_of_resources, minerals_processing).
ako(strategies_for_extending_the_life_of_resources, minerals_processing).
ako(new_processes, minerals_processing).
ako(new_products, minerals_processing).
ako(process_optimisation, minerals_processing).
ako(process_control, minerals_processing).
ako(environmental_considerations, minerals_processing).
ako(conservation, environmental_considerations).


ako(long_term_sustainability, forestry).
ako(silviculture, long_term_sustainability).
ako(genetics, long_term_sustainability).
ako(physiology, long_term_sustainability).
ako(site, long_term_sustainability).
ako(soil_and_nutrition, long_term_sustainability).
ako(soil, long_term_sustainability).
ako(nutrition, long_term_sustainability).
ako(forest_management, long_term_sustainability).
ako(wood_properties, long_term_sustainability).
ako(forest_protection, long_term_sustainability).
ako(fire, conservation).
ako(fire, forest_protection).
ako(disease, forest_protection).
ako(pests, forest_protection).
ako(environmental_management, long_term_sustainability).


ako(forest_engineering, forestry).
ako(harvesting, forest_engineering).
ako(transport, forest_engineering).
ako(roads, forest_engineering).
```

```
ako(oak, trees).
ako(trees, conservation).


ako(pulp_and_paper, forestry).
ako(energy_conservation, pulp_and_paper).
ako(water_conservation, pulp_and_paper).
ako(effluent_reduction, pulp_and_paper).
ako(pulping, pulp_and_paper).
ako(bleaching, pulp_and_paper).
ako(bleach, pulp_and_paper).
ako(recovery_of_chemicals, pulp_and_paper).
ako(control, pulp_and_paper).


ako(timber_processing, forestry).
ako(saw_milling, timber_processing).
ako(timber_drying, timber_processing).
ako(board_making, timber_processing).
ako(waste_use, timber_processing).
ako(recycling, waste_use).
ako(recycling, conservation).
ako(process_optimisation, timber_processing).


ako(metals, materials).
ako(alloys, materials).
ako(abrasives, materials).
ako(protective_coatings, materials).
ako(plastic, protective_coatings).
ako(plastic, re-cycling).
ako(polymers, materials).
ako(synthetic_oils, materials).
ako(dyes, materials).
ako(pigments, materials).
ako(concrete, materials).
ako(cements, materials).
ako(clays, materials).
ako(ceramics, materials).
ako(composites, materials).
ako(smart_materials, materials).
ako(conductive_materials, materials).
ako(semiconductors, materials).
ako(glasses, materials).
ako(fluid_compounds, materials).
ako(bleach, fluid_compounds).
ako(natural_materials, materials).
ako(wood, natural_materials).
ako(wood, forestry).
ako(plant, wood).
ako(leather, natural_materials).
ako(essential_oils, natural_materials).
```

```
/* sixth level exclusion example */
ako(ear, ring).
ako(ring, gold).
ako(gold, metals).
ako(materials, raw-materials).


ako(innovation, discrete_manufacturing).
ako(creativity_development, discrete_manufacturing).
ako(new_and_improved_processes, discrete_manufacturing).
ako(product_design, discrete_manufacturing).
ako(industrial_design, discrete_manufacturing).
ako(plant_design, discrete_manufacturing).


ako(innovation, process_manufacturing).
ako(creativity_development, process_manufacturing).
ako(new_and_improved_processes, process_manufacturing).
ako(product_design, process_manufacturing).
ako(industrial_design, process_manufacturing).
ako(plant_design, process_manufacturing).


ako(process_control, discrete_manufacturing).
ako(process_control, process_manufacturing).
ako(plant, plant_design).
ako(plant, forestry).


ako(product_design, process_manufacturing).
ako(product_design, discrete_manufacturing).


ako(production_technology, discrete_manufacturing).
ako(quality, discrete_manufacturing).
ako(reliability, discrete_manufacturing).
ako(economic_feasibility, discrete_manufacturing).
ako(materials_selection, discrete_manufacturing).
ako(recyclability, discrete_manufacturing).
ako(discrete_manufacturing_processes, discrete_manufacturing).
ako(continuous_and_batch_processes, discrete_manufacturing).
ako(time_compression_technologies, discrete_manufacturing).
ako(environmental_considerations, discrete_manufacturing).
ako(environmental_management, environmental_considerations).
ako(environment, environmental_management).


ako(production_technology, process_manufacturing).
ako(quality, process_manufacturing).
ako(reliability, process_manufacturing).
ako(process_control, reliability).
ako(economic_feasibility, process_manufacturing).
ako(materials_selection, process_manufacturing).
ako(materials, materials_selection).
```

```
ako(recyclability, process_manufacturing).


ako(recycling, recyclability).
ako(discrete_manufacturing_processes, process_manufacturing).
ako(continuous_and_batch_processes, process_manufacturing).
ako(time_compression_technologies, process_manufacturing).
ako(environmental_considerations, process_manufacturing).


ako(ergonomics, discrete_manufacturing).
ako(optimisation, discrete_manufacturing).
ako(automation, discrete_manufacturing).
ako(enterprise_integration, discrete_manufacturing).
ako(materials_forming, discrete_manufacturing).
ako(materials, materials_forming).
ako(materials_removing, discrete_manufacturing).
ako(materials, materials_removing).
ako(assembly, discrete_manufacturing).
ako(separation, discrete_manufacturing).
ako(rapid_prototyping, discrete_manufacturing).

ako(organic_synthesis, continuous_and_batch_processes).
ako(separation, continuous_and_batch_processes).
ako(catalysis, continuous_and_batch_processes).
ako(bioprocesses, continuous_and_batch_processes).
ako(high_valve_chemicals, continuous_and_batch_processes).
ako(fine_chemicals, continuous_and_batch_processes).
ako(chemicals, high_valve_chemicals).
ako(chemicals, fine_chemicals).


ako(fine_chemicals, pharmaceuticals).
ako(drug_delivery_systems, pharmaceuticals).
ako(pharmacognoscy, pharmaceuticals).
ako(pharmacokinetics, pharmaceuticals).
ako(medicinal_chemistry, pharmaceuticals).
ako(chemicals, medicinal_chemistry).
ako(fine_chemicals, biomedicine).
ako(drug_delivery_systems, biomedicine).
ako(pharmacognoscy, biomedicine).
ako(pharmacokinetics, biomedicine).
ako(medicinal_chemistry, biomedicine).


ako(software_engineering, information_technology).
ako(software_quality_assurance, information_technology).
ako(quality, software_quality_assurance).
ako(software_process_assessment, information_technology).
ako(assessment, software_process_assessment).
ako(computer_system_security, information_technology).
ako(distributed_network_and_real_time_computing, information_technology).
ako(networks, distributed_network_and_real_time_computing).
ako(visualisation, information_technology).
```

```
ako(multi-media, information_technology).
ako(virtual_reality, information_technology).
ako(applications_of_high_speed_microprocessors, information_technology).
ako(safety-critical_systems, information_technology).
ako(database_systems, information_technology).
ako(knowledge-based_industry_models_and_systems, information_technology).


ako(universal_service, communications_systems).
ako(digital_encoding, universal_service).
ako(compression, universal_service).
ako(high_capacity_cellular_telephony, universal_service).
ako(evaluation_of_the_rural_telecommunictaions_infrastructure,
      universal_service).
ako(distribution_of_medical_and_educational_tele-services,
      universal_service).
ako(satellite_technology, universal_service).
ako(integration_of_computing_and_telecommunications_networks,
      universal_service).


ako(network_technology, communications_systems).
ako(networks, network_technology).
ako(network_management, network_technology).
ako(protocols, network_technology).
ako(dimensioning, network_technology).
ako(optimisation, network_technology).
ako(security, network_technology).
ako(signalling, network_technology).
ako(traffic_analysis, network_technology).
ako(forecasting, network_technology).
ako(broadband_technologies, network_technology).
ako(intelligent_services, network_technology).
ako(the_internet_and_its_applications, network_technology).


ako(measurement, industrial_electronics).
ako(sensing, industrial_electronics).
ako(information_and_energy_procesing, industrial_electronics).
ako(industrial_infrastructure_processes_and_systems,
industrial_electronics).
ako(transport_infrastructure_processes_and_systems,
industrial_electronics).
ako(transport, transport_infrastructure_processes_and_systems).
ako(transport, transportation_issues).


ako(artificial_intelligence, control_systems).
ako(heuristic_systems, control_systems).
ako(fuzzy_logic, control_systems).
ako(adaptive_and_robust_control, control_systems).
ako(new_mathemetical_developments, control_systems).
```

```
ako(process_optimisation, new_mathemetical_developments).
ako(optimisation, process_optimisation).
ako(pid, new_mathemetical_developments).
ako(plc, new_mathemetical_developments).
ako(scada, new_mathemetical_developments).
ako(logistic_systems, control_systems).


/*  Improved quality of life dependancies */


ako(crop_integration, farming_systems).
ako(forestry, farming_systems).
ako(optimisation, crop_integration).
ako(livestock, farming_systems).
ako(non_conventional_animal_production, farming_systems).
ako(intervention_aimed_at_improving_farming_systems, farming_systems).
ako(future_farmers, farming_systems).
ako(extension_officers, farming_systems).
ako(agricultural_economists, farming_systems).
ako(research, farming_systems).
ako(existing_system_information, research).
ako(farming_information, existing_system_information).
ako(optimising_farming_systems, research).
ako(constraints, optimising_farming_systems).
ako(mathematical_modelling, optimising_farming_systems).
ako(farming_policies, research).
ako(food_production, farming_policies).
ako(household_food_security, farming_policies).


ako(climate, sustainable_use_of_natural_resources_for_food_production).
ako(water, sustainable_use_of_natural_resources_for_food_production).
ako(soil, sustainable_use_of_natural_resources_for_food_production).
ako(range-lands,
sustainable_use_of_natural_resources_for_food_production).


ako(adverse_climatic_conditions, climate).
ako(hail, adverse_climatic_conditions).
ako(drought, adverse_climatic_conditions).
ako(flood, adverse_climatic_conditions).
ako(tornado, adverse_climatic_conditions).
ako(snow, adverse_climatic_conditions).
ako(food_production, adverse_climatic_conditions).
/* include other adverse weather conditions here.*/


ako(water_supply, water).
ako(conservation, water).
ako(reticulation, water).
ako(use_and_mangement_for_agricultural_purposes, water).
ako(optimisation, water).
```

```
ako(soil_degradation, soil).
ako(soil_erosion, soil).
ako(erosion_fertility, soil).
ako(conservation, soil).
ako(optimal_utilisation_and_management_for_agricultural_purposes, soil).
ako(optimisation,
optimal_utilisation_and_management_for_agricultural_purposes).


ako(productivity, range-lands).
ako(efficient_utilistaion_and_management, range-lands).
ako(land_tenure_systems_affecting_animal_production_systems, range-
lands).
ako(establishment_of_pastures, range-lands).
ako(utilisation_of_pastures, range-lands).
ako(maintenance_of_pastures, range-lands).
ako(optimisation, efficient_utilistaion_and_management).


ako(optimising_indigenous_and_adapted_biodiversity,
      animal_production_and_management).
ako(optimisation, optimising_indigenous_and_adapted_biodiversity).
ako(animal_traction, animal_production_and_management).
ako(utilisation_of_wastes_and_by-products,
animal_production_and_management).
ako(prevention_and_containment_of_animal_diseases,
      animal_production_and_management).


ako(characterisation_of_indigenous_and_adapted_animal_genetic_resources,
          optimising_indigenous_and_adapted_biodiversity).
ako(conservation_of_indigenous_and_adapted_animal_genetic_resources,
          optimising_indigenous_and_adapted_biodiversity).
ako(conservation,
      conservation_of_indigenous_and_adapted_animal_genetic_resources).
ako(utilistaion_of_indigenous_and_adapted_animal_genetic_resources,
          optimising_indigenous_and_adapted_biodiversity).
ako(sustainable_livestock_production,
      optimising_indigenous_and_adapted_biodiversity).
ako(diversity_of_livestock_for_food_production,
        optimising_indigenous_and_adapted_biodiversity).
ako(research_and_animal_improvement_programmes,
        optimising_indigenous_and_adapted_biodiversity).
ako(food_production, diversity_of_livestock_for_food_production).
ako(research, research_and_animal_improvement_programmes).


ako(adverse_climatic_conditions, animal_traction).
ako(intensified_and_diversified_use_of_draught_animals, animal_traction).
ako(feeding_strategies_of_draught_animals, animal_traction).
ako(alternative_sources_of_draught_power, animal_traction).
```

```
ako(new_uses_for_wastes_of_animal_production,
      utilisation_of_wastes_and_by-products).
ako(recycling, new_uses_for_wastes_of_animal_production).
ako(new_uses_for_by-products_of_animal_production,
      utilisation_of_wastes_and_by-products).
ako(recycling, new_uses_for_by-products_of_animal_production).
ako(additional_income, recycling).
ako(job_creation, recycling).


ako(devising_cost-effective_and_sustainable_interventions,
            prevention_and_containment_of_animal_diseases).
ako(competetive_production_of_diagnostic_kits,
      prevention_and_containment_of_animal_diseases).
ako(vaccines_for_control_of_ticks,
      prevention_and_containment_of_animal_diseases).
ako(strategies_aimed_at_tick-borne_diseases,
      prevention_and_containment_of_animal_diseases).
ako(internal_parasites, prevention_and_containment_of_animal_diseases).
ako(diseases_that_constrain_reproductive_performance,
          prevention_and_containment_of_animal_diseases).
ako(production_of_milk_and_meat_and_other_edible_consumable_animal_
      products,  prevention_and_containment_of_animal_diseases).
ako(research, devising_cost-effective_and_sustainable_interventions).
ako(research, competetive_production_of_diagnostic_kits).
ako(research, vaccines_for_control_of_ticks).
ako(research, strategies_aimed_at_tick-borne_diseases).
ako(research, internal_parasites).
ako(research, diseases_that_constrain_reproductive_performance).
ako(research, production_of_milk_and_meat_and_
                  other_edible_consumable_animal_products).


ako(optimal_utilistaion_of_indigenous_plants,
      crop_and_horticultural_production_and_management).
ako(optimisation, optimal_utilistaion_of_indigenous_plants).
ako(new_crops_and_new_uses_of_plants,
      crop_and_horticultural_production_and_management).
ako(utilistaion_of_wastes_and_by-products,
      crop_and_horticultural_production_and_management).
ako(recycling, utilistaion_of_wastes_and_by-products).
ako(plant_protection, crop_and_horticultural_production_and_management).
ako(post-harvest_treatment,
crop_and_horticultural_production_and_management).
ako(environmental_protection, recycling).


ako(identifying_and_utilisation_of_indegenous_plants,
      optimal_utilistaion_of_indigenous_plants).
ako(indigenous_plants_with_market_potential_for_food,
      optimal_utilistaion_of_indigenous_plants).
ako(assessing_indigenous_plants_market_potential,
      optimal_utilistaion_of_indigenous_plants).
ako(food_production, assessing_indigenous_plants_market_potential).
```

```
ako(fast_propagation_of_indigenous_plants,
      optimal_utilistaion_of_indigenous_plants).
ako(protection_of_inigenous_plants,
optimal_utilistaion_of_indigenous_plants).
ako(conservation, protection_of_inigenous_plants).
ako(understanding_and_improving_growth,
      optimal_utilistaion_of_indigenous_plants).
ako(maturation_of_indigenous_plants,
      optimal_utilistaion_of_indigenous_plants).
ako(stress_physiology_of_indigenuos_plants,
      optimal_utilistaion_of_indigenous_plants).
ako(production_and_adaptation_of_indigenous_plants,
      optimal_utilistaion_of_indigenous_plants).


ako(new_species, new_crops_and_new_uses_of_plants).
ako(additional_income, new_species).
ako(crops_for_arid_areas, new_crops_and_new_uses_of_plants).
ako(additional_income, crops_for_arid_areas).
ako(innovative_use_of_existing_crops, new_crops_and_new_uses_of_plants).
ako(additional_income, innovative_use_of_existing_crops).
ako(horticultural_plants, new_crops_and_new_uses_of_plants).
ako(innovative_use_of_horticultural_plants, horticultural_plants).
ako(additional_income, innovative_use_of_horticultural_plants).


ako(waste_or_by-products_of_horticultural_plants,
      utilistaion_of_wastes_and_by-products).
ako(recycling, waste_or_by-products_of_horticultural_plants).
ako(waste_or_by-products_of_crop_production,
      utilistaion_of_wastes_and_by-products).
ako(recycling, waste_or_by-products_of_crop_production).


ako(plant-insect_relationships, plant_protection).
ako(plant-pathogen_relationships, plant_protection).
ako(crop-weed_relationships, plant_protection).
ako(management_and_control_of_pests, plant_protection).
ako(management_and_control_of_weeds, plant_protection).
ako(management_and_control_of_diseases, plant_protection).
ako(environmental_protection, management_and_control_of_pests).
ako(environmental_protection, management_and_control_of_weeds).
ako(environmental_protection, management_and_control_of_diseases).


ako(post-harvest_physiology, post-harvest_treatment).
ako(post-harvest_storage, post-harvest_treatment).
ako(post-harvest_transport, post-harvest_treatment).
ako(transport, post-harvest_transport).


ako(new_products_and_improved_processing,
      innovation_in_the_food_and_beverage_industry).
ako(enhanced_shelf_life, innovation_in_the_food_and_beverage_industry).
```

```
ako(quality_and_safety, innovation_in_the_food_and_beverage_industry).
ako(nutritional_value, innovation_in_the_food_and_beverage_industry).
ako(food_production, innovation_in_the_food_and_beverage_industry).


ako(new_and_better_food_and_beverage_products,
      new_products_and_improved_processing).
ako(development_and_improvement_of_processing,
      new_products_and_improved_processing).
ako(preservation_and_increase_of_nutritional_value,
      new_products_and_improved_processing).
ako(artificial_preservatives, new_products_and_improved_processing).
ako(colourants_and_flavourants, new_products_and_improved_processing).
ako(environmental_protection, development_and_improvement_of_processing).


ako(prevention_of_spoilage, enhanced_shelf_life).
ako(basic_and_processed_foods_and_bevearges, enhanced_shelf_life).
ako(methods_for_poorer_and_rural_areas, enhanced_shelf_life).
ako(accessability_of_food_and_beverages, enhanced_shelf_life).
ako(preservatives, prevention_of_spoilage).
ako(preservatives, preservation_and_increase_of_nutritional_value).


ako(food_and_beverages, quality_and_safety).
ako(production_of_safe_food_and_beverages, quality_and_safety).
ako(quality, production_of_safe_food_and_beverages).
ako(food-borne_diseases, quality_and_safety).
ako(improvement_of_sensory_quality, quality_and_safety).
ako(poorer_communities, quality_and_safety).


ako(products_to_eradicate_malnutrition, nutritional_value).
ako(processes_to_eradicate_malnutrition, nutritional_value).
ako(procedures_to_eradicate_malnutrition, nutritional_value).
ako(enhancement_of_nutritional_value_of_food_and_beverages,
      nutritional_value).
ako(safer_nutritional_alternatives, nutritional_value).


ako(fresh_water_and_sanitation_provision, water_and_sanitation).
ako(maintenance_and_management, water_and_sanitation).
ako(optimising_water_reources, water_and_sanitation).
ako(optimisation, optimising_water_reources).
ako(energy-efficient, water_and_sanitation).
ako(cost-effective, water_and_sanitation).
ako(rural_communities, water_and_sanitation).
ako(minimising_ground_water_pollution, water_and_sanitation).
ako(minimising_surface_water_pollution, water_and_sanitation).
ako(recycling, water_and_sanitation).
ako(conservation, water_and_sanitation).
ako(socially_acceptable_sanitation_systems, water_and_sanitation).
ako(by-product_disposal, water_and_sanitation).
ako(environmental_protection, by-product_disposal).
```

```
ako(prevention, waste_and_pollution_management).
ako(reduction, waste_and_pollution_management).
ako(disposal, waste_and_pollution_management).
ako(utilisation, waste_and_pollution_management).
ako(by-products_from_industry, waste_and_pollution_management).
ako(by-products_from_urbanisation, waste_and_pollution_management).
ako(by-products_from_agriculture, waste_and_pollution_management).
ako(by-products_from_municipals, waste_and_pollution_management).
ako(environmental_protection, waste_and_pollution_management).
ako(recycling, by-products_from_industry).
ako(recycling, by-products_from_urbanisation).
ako(recycling, by-products_from_agriculture).
ako(recycling, by-products_from_municipals).
ako(policies, waste_and_pollution_management).
ako(research, policies).
ako(optimisation, waste_and_pollution_management).


ako(affordable, housing).
ako(policies, housing).
ako(new_building_design, housing).
ako(new_construction_methods, housing).
ako(development_of_new_materials_and_methods, housing).
ako(demographic_projections, housing).
ako(township_and_village_layout, housing).
ako(cultural_dimensions, housing).
ako(capacity_building, housing).
ako(job_creation, housing).
ako(low-cost_housing, new_building_design).
ako(energy-efficient, low-cost_housing).
ako(economically_viable, low-cost_housing).
ako(innovative_materials, development_of_new_materials_and_methods).
ako(materials, innovative_materials).
ako(additional_income, housing).
ako(environmentally_sound_high-density_development, housing).
ako(environmental_protection,
      environmentally_sound_high-density_development).


ako(development_of_rural_transport_systems, transport).
ako(maintenance_of_rural_transport_systems, transport).
ako(management_of_rural_transport_systems, transport).
ako(efficient_modes_of_transport, transport).
ako(minimising_traffic_congestion, transport).
ako(accident_avoidance_systems, transport).
ako(energy-efficient, transport).
ako(minimising_pollution, transport).
ako(fuel_technology, transport).


ako(energy_resources, energy).
ako(informal_production, energy).
ako(community_services, energy).
```

```
ako(conservation, energy).
ako(rural_energy_needs, energy).
ako(alternative_energy_sources, energy).
ako(energy-saving_appliances, energy).
ako(cost_effective_and_socially_acceptable_energy_systems, energy).
ako(energy-efficient, energy).
ako(new_materials, energy).
ako(materials, new_materials).
ako(policies, energy_resources).


ako(optimum_use_of_available_land, land-use_and_rehabilitation).
ako(optimisation, optimum_use_of_available_land).
ako(soil_erosion, land-use_and_rehabilitation).
ako(nutrient_exhaustion, land-use_and_rehabilitation).
ako(mining_activities, land-use_and_rehabilitation).
ako(pollution, land-use_and_rehabilitation).
ako(deforestation, land-use_and_rehabilitation).
ako(forests, deforestation).
ako(ecological_use_of_available_land, land-use_and_rehabilitation).
ako(ecology, ecological_use_of_available_land).
ako(protection_of_land, land-use_and_rehabilitation).
ako(environmental_protection, protection_of_land).


ako(poorer_communities_income_generation, community-
based_industrialisation).
ako(empowering_people, community-based_industrialisation).
ako(entrepreneurial_skills, community-based_industrialisation).
ako(decreasing_dependency, community-based_industrialisation).
ako(rural_and_urban_development, community-based_industrialisation).


/* sustainable environment dependencies */


ako(identification_of_potential_threats, biodiversity_and_conservation).
ako(pollution, identification_of_potential_threats).
ako(classification_of_systems, biodiversity_and_conservation).
ako(ecology_of_endemic_species, biodiversity_and_conservation).
ako(distribution_of_endemic_species, biodiversity_and_conservation).
ako(conservation_of_endemic_species, biodiversity_and_conservation).
ako(conservation, conservation_of_endemic_species).
ako(protected_areas, biodiversity_and_conservation).
ako(friendly_land_use, biodiversity_and_conservation).
ako(protection_of_natural_features, biodiversity_and_conservation).
ako(environmental_protection, protection_of_natural_features).
ako(alien_invasion_and_harmful_species, biodiversity_and_conservation).
ako(natural_habitats, biodiversity_and_conservation).
ako(ecosystems, biodiversity_and_conservation).
ako(landscapes, biodiversity_and_conservation).
ako(land_use, biodiversity_and_conservation).
ako(waterscapes, biodiversity_and_conservation).
ako(protected_areas, biodiversity_and_conservation).
```

```
ako(indigenous_flora_and_fauna, indigenous_species_utilisation).
ako(use_by_agricultural_indutries, indigenous_species_utilisation).
ako(use_by_pharmaceutical_industries, indigenous_species_utilisation).
ako(use_by_ecotourism_industries, indigenous_species_utilisation).
ako(use_by_traditional_communities_and_healers,
      indigenous_species_utilisation).
ako(hunting_and_tourism, indigenous_species_utilisation).
ako(conservation, indigenous_species_utilisation).


ako(waste_products, systems_management).
ako(ecosystems, systems_management).
ako(cachments, systems_management).
ako(wetlands, systems_management).
ako(rivers, systems_management).
ako(estuaries, systems_management).
ako(forestry, systems_management).
ako(grasslands, systems_management).
ako(conservation_of_natural_forest_systems, systems_management).
ako(conservation, conservation_of_natural_forest_systems).
ako(species_migration_patterns, systems_management).


ako(infrastructural_development, restoration_and_rehabilitation).
ako(waste_disposal, restoration_and_rehabilitation).
ako(by-product_disposal, waste_disposal).
ako(injudicious_use_of_natural_resources,
restoration_and_rehabilitation).
ako(replacement_of_dominant_and_harmful_alien_species,
      restoration_and_rehabilitation).
ako(riparian_zone_management, restoration_and_rehabilitation).
ako(reclamation_of_dunes_and_mine_dumps, restoration_and_rehabilitation).
ako(reclamation_of_detoxic_areas, restoration_and_rehabilitation).


ako(coastal_and_marine_ecosystems,
coastal_communities_and_living_resources).
ako(use_of_coastal_living_resources,
      coastal_communities_and_living_resources).
ako(dependent_communities, coastal_communities_and_living_resources).
ako(resource_management, coastal_communities_and_living_resources).
ako(resource_utilisation, coastal_communities_and_living_resources).
ako(new_resources_and_improved_methods,
      coastal_communities_and_living_resources).
ako(perception_about_marine_reserves,
      coastal_communities_and_living_resources).
ako(multiple_resource_fisheries,
coastal_communities_and_living_resources).
ako(impacts_of_developments, coastal_communities_and_living_resources).
ako(fisheries_management, coastal_communities_and_living_resources).
ako(conservation, coastal_communities_and_living_resources).
```

```
ako(conservation, the_coast_as_a_resource).
ako(human_needs, the_coast_as_a_resource).
ako(opportunities_for_better_use_of_the_coast, the_coast_as_a_resource).
ako(sensitive_areas_in_need_of_protection, the_coast_as_a_resource).
ako(estuaries, the_coast_as_a_resource).
ako(harbours_and_bays, the_coast_as_a_resource).
ako(dunes, the_coast_as_a_resource).
ako(habitat_restoration_or_rehabilitation, the_coast_as_a_resource).
ako(habitat_degradation, the_coast_as_a_resource).
ako(management_of_waste, the_coast_as_a_resource).
ako(marine_reserves, conservation).


ako(fishing, offshore_living_resources_and_society).
ako(ecology_and_biology_of_demersal_fish,
      offshore_living_resources_and_society).
ako(ecology_and_biology_of_cephalopods,
      offshore_living_resources_and_society).
ako(ecology_and_biology_of_crustaceans,
      offshore_living_resources_and_society).
ako(fish_population_dynamics, offshore_living_resources_and_society).
ako(fisheries_management, offshore_living_resources_and_society).
ako(environmental_influences, offshore_living_resources_and_society).


ako(farming_of_organisms_in_seawater, mariculture).
ako(replenishment_of_marine_organisms, mariculture).
ako(conservation, mariculture).
ako(employment_opportunities, mariculture).
ako(anthropogenic_effects_of_marine_culture, mariculture).
ako(evaluation_of_geographic_locations_and_sites, mariculture).
ako(effects_of_toxic_blooms, mariculture).
ako(effects_of_disease_on_stocks, mariculture).
ako(selection_of_candidate_species, mariculture).


ako(maintenance_of_ecosystems, biodiversity_and_conservation).
ako(tourism_and_recreation, biodiversity_and_conservation).
ako(protection_of_resources, biodiversity_and_conservation).
ako(stock_enhancement, biodiversity_and_conservation).
ako(conservation, biodiversity_and_conservation).
ako(population_dynamics, biodiversity_and_conservation).
ako(marine_and_coastal_ecosystems, biodiversity_and_conservation).
ako(threats, biodiversity_and_conservation).
ako(biological_diversity, biodiversity_and_conservation).


ako(oceanographic_variability, ocean_dynamics_and_coastal_geomorphology).
ako(currents_and_temperatures, ocean_dynamics_and_coastal_geomorphology).
ako(seasonal_and_inter-annual_time_scales,
      ocean_dynamics_and_coastal_geomorphology).
ako(dispersal_of_eggs_and_larvae,
      ocean_dynamics_and_coastal_geomorphology).
ako(water_quality, ocean_dynamics_and_coastal_geomorphology).
```

```
ako(prediction_of_biogeochemical_conditions,
      ocean_dynamics_and_coastal_geomorphology).


ako(forecasting, weather).
ako(extreme_variations_in_weather_and_climate, weather_and_climate).
ako(environmental_and_socio-economical_impacts, weather_and_climate).
ako(oceanic_circulation_patterns, weather_and_climate).
ako(marine_influences_on_continental_climate, weather_and_climate).
ako(climate_modelling, weather_and_climate).
ako(atmospheric_and_terrestrial_models, weather_and_climate).


ako(climate_and_wildlife_biodiversity, ecotourism).
ako(natural_resources, ecotourism).
ako(low-impact_nature_based_tourism, ecotourism).
ako(imacts_of_over-utilisation, ecotourism).
ako(conservation, ecotourism).
ako(rehabilitation_of_ecotourism_sites, ecotourism).
ako(waste_management, ecotourism).
ako(forecasting_of_tourist_expectations, ecotourism).


ako(commonly_accepted_curriculum_frameworks_in_science,
      appropriate_curriculum_development).
ako(commonly_accepted_curriculum_frameworks_in_mathematics,
      appropriate_curriculum_development).
ako(commonly_accepted_curriculum_frameworks_in_technology,
      appropriate_curriculum_development).
ako(vocational_subjects_in_the_technical_field,
      appropriate_curriculum_development).
ako(curriculum_materials, appropriate_curriculum_development).


ako(measurement, assessment).
ako(assessment, examinations).
ako(assessment, assesment_protocols).
ako(examinations, assessment_criteria).
ako(assessment_protocols, assessment_criteria).
ako(address_existing_biases, assessment_criteria).
ako(developing_assessment_models, assessment_criteria).
ako(appraisal, evaluation).
ako(evaluation, assessment).
ako(assessment, staff_development).
ako(in-service_training, fde).


ako(cost-effective_modes_of_instructional_delivery,
      cost-effective_and_quality_delivery_of_teaching).
ako(large_class_instructional_techniques,
      cost-effective_and_quality_delivery_of_teaching).
ako(use_of_accessible_resources,
      cost-effective_and_quality_delivery_of_teaching).
ako(distance_education, cost-effective_and_quality_delivery_of_teaching).
```

```
ako(use_of_educational_technology,
      cost-effective_and_quality_delivery_of_teaching).
ako(disadvantage_of_learning_mathematics_and_technology_in_a_
            different_language, language_and_learning).
ako(disadvantage_of_learning_science, language_and_learning).
ako(instructional_designs_aimed_at_minimising_language_disadvantages,
      language_and_learning).
ako(focus_on_pre-tertiary_level, language_and_learning).


ako(colleges_of_education, initial_teacher_education).
ako(reform_of_programmes_and_curricula, initial_teacher_education).
ako(programme_design, initial_teacher_education).
ako(curricular_alternatives, initial_teacher_education).
ako(innovative_practica, initial_teacher_education).
ako(pedagogical_content_knowledge, initial_teacher_education).
ako(reflective_practice, initial_teacher_education).
ako(cotep, initial_teacher_education).


ako(teacher_development, professional_development).
ako(subject_advisory_services, professional_development).
ako(in-service_training, professional_development).
ako(fde, professional_development).
ako(demonstratable_sustained_modification, professional_development).
ako(technology, professional_development).
ako(professional_preparation, staff_development).
ako(effective_models, staff_development).


ako(public_understanding_and_awareness_of_set, scientific_literacy).
ako(develop_appropriate_forms_of_measurements, scientific_literacy).
ako(develop_optimal_levels_of_content, scientific_literacy).


ako(science_centres_and_museums, evaluations).
ako(impact_assessment, evaluations).
ako(qualitative_examination_of_visitor_experience, evaluations).
ako(effectiveness_of_exhibit_design_and_layout, evaluations).
ako(new_models_of_set_awareness_activities, evaluations).


ako(media, set_communication).
ako(develop_expertise_in_set_communication, set_communication).
ako(diversity_of_contexts, set_communication).
```

```
/*************************************************************************/
/*                                                                       */
/*                        THE INFERENCE ENGINE                           */
/*                                                                       */
/*************************************************************************/


/***** RECURSIVE INHERITANCE LINKS - WITH DEPTH LIMITATION OF 4/5 *****/


akko(A, B, DEEP) :- DEEP > 0,
                    ako(A, C),
                    DEEP1 is DEEP - 1,
                    akko(C, B, DEEP1),
                      write(C),nl.

akko(A, B, DEEP) :- DEEP > 0,
                    ako(A, B), write(B), nl.


akko2(A, B) :-    ako(A, C),
            akko2(C, B),
            write(C),nl.

akko2(A, B) :-    ako(A, B), write(B), nl.


/***** SEARCH CONSTRAINED BY DEPTH *****/


depth_4_search :-
      repeat,(
            nl, write('Enter a keyword <"quit." to end> :'),
                  read(ANS),nl,
                  (p(ANS, 2), (ANS = quit ))).

depth_5_search :-
      repeat,(
            nl, write('Enter a keyword <"quit." to end> :'),
                  read(ANS),nl,
                  (p(ANS, 3), (ANS = quit ))).

depth_4_or_5_search :-
      repeat,(
            nl, write('Enter a keyword <"quit." to end> :'),
                  read(ANS),nl,
                  (p2(ANS, 2), (ANS = quit ))).
```

```
exhaustive_search :-
      repeat,(
            nl, write('Enter a keyword <"quit." to end> :'),
                  read(ANS),nl,
                  (p3(ANS), (ANS = quit ))).

p3(quit) :- quit.

p2(quit, _) :- quit.

p(quit, _) :- quit.

quit :- write('finish.'),nl, menu.

end :- write('finish.'),nl.

p(SOL, Dep) :-
            (SOL = quit, !);

            (theme(SOL), write(SOL),
                  write(' is an established theme!!!'),nl),! ;
            (programme_of(SOL, Y), write(SOL),
                  write(' is a theme of '), write(Y),nl),!;
            (isa(SOL, X), programme_of(X, Y), write(SOL),nl,
                  write('     PROGRAMME   :'),
                  nl, write(X),nl,nl,
                  write('YOU MAY APPLY VIA THE FOLLOWING THEME = '),
                  write(Y),nl,nl),!;
            (Depp is Dep + 2,write('MAXIMUM DEPTH = '),write(Depp),nl,
            nl,
                  f(SOL, BB, CC, DD, Dep));
            Depp is Dep + 2, nl,write('END OF SEARCH FOR KEYWORD --> '),
            write(SOL), write(' <-- AT DEPTH = '), write(Depp),nl.


p2(SOL, Dep) :-
            (SOL = quit, !);
            (write('ONE HIT ONLY :'), nl),
            (theme(SOL), write(SOL),
                  write(' is an established theme!!!'),nl),! ;
            (programme_of(SOL, Y), write(SOL),
                  write(' is a theme of '), write(Y),nl),!;
            (isa(SOL, X), programme_of(X, Y), write(SOL),nl,
                  write('     PROGRAMME   :'),nl, write(X),nl,nl,
                  write('YOU MAY APPLY VIA THE FOLLOWING THEME = '),
                  write(Y),nl,nl),!;
            (Depp is Dep + 2, write('MAXIMUM DEPTH = '), write(Depp), nl,
                  f(SOL, BB, CC, DD, Dep)),!;
                  (Depp is Dep + 3, nl,nl, write('MAXIMUM DEPTH = '),
                  write(Depp), nl,
                  Dept is Dep + 1, f(SOL, BB, CC, DD, Dept)),!;
            (nl, write('******* NO HIT AT DEPTH 4 OR 5 ********'),nl).
```

```
p3(SOL) :-
            (SOL = quit, !);

            (theme(SOL), write(SOL),
                  write(' is an established theme!!!'),nl),! ;
            (programme_of(SOL, Y), write(SOL),
                  write(' is a theme of '), write(Y),nl),!;
            (isa(SOL, X), programme_of(X, Y), write(SOL),nl,
                  write('     PROGRAMME   :'),
                  nl, write(X),nl,nl,
                  write('YOU MAY APPLY VIA THE FOLLOWING THEME = '),
                  write(Y),nl,nl),!;
            (write('UNLIMITED DEPTH : '), f2(SOL, BB, CC, DD));
            nl,write('END OF SEARCH FOR KEYWORD --> '),
            write(SOL), write(' <-- AT LOWEST DEPTH. '),nl.


f(AA,BB, CC,DD, DEEP) :-   write(AA),nl,write('AKOs [in reverse order]
            :'),nl,nl,
                  akko(AA,BB, DEEP), isa(BB, CC), programme_of(CC,
                  DD),
                  nl, write('    PROGRAMME  : '),nl,write(CC),nl,
                  write('YOU MAY APPLY VIA THE FOLLOWING THEME ==> '),
                  write(DD),nl,nl,
                  write('    AKOs [if any - in reverse order] :'),nl.

f2(AA,BB, CC,DD) :-        write(AA),nl,write('    AKOs [in reverse
order]
            :'),nl,nl,
                  akko2(AA,BB), isa(BB, CC), programme_of(CC, DD),
                  nl, write('    PROGRAMME  : '),nl,write(CC),nl,
                  write('YOU MAY APPLY VIA THE FOLLOWING THEME ==> '),
                  write(DD),nl,nl,
                  write('    AKOs [if any - in reverse order]
:'),nl.
```

```
/*************************************************************************/
/*                                                                       */
/*                        THE USER INTERFACE                             */
/*                                                                       */
/*************************************************************************/

% choose a search type from a listbox and then display suitable menus

menu :-
  menu_choice( Search ),
  Search.

% if there are no more alternatives of the selected type then close the
% display

menu(Search) :-
  wclose( menu_display ).

% create the windows for the menu display dialog

create_menu_display( DialogName ) :-
  wdcreate(  menu_display,
                    DialogName,
                              100,  80, 430, 330, [ws_popup,
                                                   ws_caption,
                                                   ws_sysmenu,
                                                   dlg_modalframe] ),
  wccreate( (menu_display,10),
            static, ``,
                               30,  20, 370, 240, [ws_child,
                                                   ws_visible] ),
  wccreate( (menu_display,1),
            button, `More`,
                               80, 270,  80,  22, [ws_child,
                                                   ws_visible,
                                                   ws_tabstop,
                                                   bs_defpushbutton] ),
  wccreate( (menu_display,2),
            button, `Finish`,
                              260, 270,  80,  22, [ws_child,
                                                   ws_visible,
                                                   ws_tabstop] ),
  wfont( (menu_display,10), 0 ).

% return the result from a multiple selection listbox

menu_choice( Selection ) :-
  create_menu_choice,
  fill_menu_choice,
  call_dialog( menu_choice, Result ),
  get_single_selection( (menu_choice,400), Selection ),
  wclose( menu_choice ),
  Result = ok.
```

```
% create the windows for the listbox dialog

create_menu_choice :-
  wdcreate(  menu_choice,
                      `FUND Theme/Programme Selector - NRF`,
                                200, 200, 300, 180, [ws_sysmenu,
                                                     ws_caption,
                                                     dlg_modalframe] ),
  wccreate( (menu_choice,1000),
            static,  `Choose a search type :`,
                                 20,  10, 112,  16, [ws_child,
                                                     ws_visible,
                                                     ss_left] ) ,
  wccreate( (menu_choice,400),
            listbox, `ListBox`,
                                 20,  30, 210,  90, [ws_child,
                                                     ws_visible,
                                                     ws_border,
                                                     ws_tabstop,
                                                     ws_vscroll] ),
  wccreate( (menu_choice,1),
            button,  `Ok`,
                                 20, 120,  80,  22, [ws_child,
                                                     ws_visible,
                                                     ws_tabstop,
                                                     bs_defpushbutton] ),
  wccreate( (menu_choice,2),
            button,  `Cancel`,
                                150, 120,  80,  22, [ws_child,
                                                     ws_visible,
                                                     ws_tabstop,
                                                     bs_pushbutton] ).

% fill the listbox with the available menus

fill_menu_choice :-
  wlbxadd( (menu_choice,400), -1, `depth_4_search` ),
  wlbxadd( (menu_choice,400), -1, `depth_5_search` ),
  wlbxadd( (menu_choice,400), -1, `depth_4_or_5_search` ),
  wlbxadd( (menu_choice,400), -1, `exhaustive_search` ),
  wlbxadd( (menu_choice,400), -1, `end` ),
  wlbxsel( (menu_choice,400), 0, 1 ),
  wfocus(  (menu_choice,400) ).

% given a listbox return its selected item

get_single_selection( Lbx, Selection ) :-
  wlbxsel( Lbx, 0, Sel ),
  (  Sel = 1
  -> wlbxget( Lbx, 0, ItemStr ),
     atom_string( Selection, ItemStr )
  ;  wlbxfnd( Lbx, 0, ``, NextItem ),
     get_single_selection( Lbx, NextItem, Selection )  ).
```

```
% find the current selection in a listbox

get_single_selection( Lbx, 0, Selection ) :-
  !,
  fail.

get_single_selection( Lbx, Item, Selection ) :-
  wlbxsel( Lbx, Item, Sel ),
  (  Sel = 1
  -> wlbxget( Lbx, Item, ItemStr ),
     atom_string( Selection, ItemStr )
  ;  wlbxfnd( Lbx, Item, ``, NextItem ),
     get_single_selection( Lbx, NextItem, Selection )
  ).

% wait for a binding of the wait/1 argument

wait_dialog(V) :-
  wait((1,V)).
```

The End.