DIFFERENT LEVELS OF EXPERTISE FOR AN EXPERT SYSTEM IN INFORMATION RETRIEVAL

DEFUDE.B
Groupe "Systèmes Intelligents de Recherche d'Informations"
Laboratoire Génie Informatique
Institut IMAG Grenoble
BP 68 38402 ST MARTIN D'HERES CEDEX
FRANCE

INTRODUCTION

The aim of this article is to describe the specifications of an Information Retrieval Expert System (IRES). The resort to expert systems is a recent way which seems to be very convenient to realize performant and convivial systems. In fact the capabilities provided by classical Information Retrieval Systems (IRS) are not adequate.
This inadequacy comes from some important restrictions :

- limited indexing vocabulary, very often designed a priori.
- the search terms have to belong to this vocabulary and only some simple semantical relations (as synonymy for example) can be used during the search.

Consequently, if the terms used in the query are not those used for indexing we do not have a good answer.

The problem is that a user does not know the indexing vocabulary so that he needs an information retrieval expert for search assistance. This expert can then formulate the query with proper terms and possibly reformulate it.
This approach is, of course, very restrictive and different studies are actually undertaken (2, 6, 11) to free the user from these constraints.
Our work is also in this direction and our goals are the following :

- free language of interrogation, implies a syntactical analysis to recognize the important concepts of the query. Our hypothesis is that these concepts are noun groups.
- use of semantics during the query process : the confrontation between the query and the corpus is semantical (we have to use deductive mechanisms).
- use of adaptative mechanisms related to the user typology and the corpus (the query process uses strategies).

In a previous study (7) we have shown that artificial intelligence techniques (and especially expert systems) are a very interesting way for such studies.

We will expose the specifications of an IRES under two principal aspects :

- general software architecture and particularly the different strategies and adaptative capabilities considered for the process control,
- the detail of the different system modules which underly the expertise involved.

1. THE ARCHITECTURE OF AN IRES

The central components of an IRES are the knowledge base and the thesaurus. We consider the thesaurus as a set of concepts and a set of semantical relations (synonymy, ...) connecting them.
This thesaurus can be designed a priori or built from the texts with automatic tools (5).

The problem of the coexistence of the knowledge base of the expert system and the thesaurus may have two solutions :

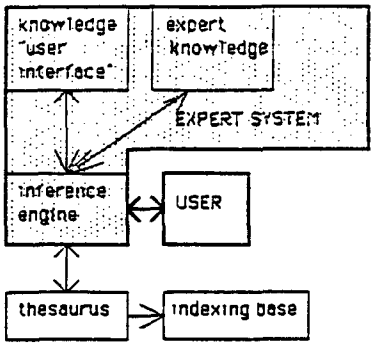
- the thesaurus is integrated in the expert system knowledge base,
- the thesaurus is considered as an external source of knowledge for the expert system.

We have shown (8) the consequences of such choices. We have chosen the hypothesis of an independent thesaurus because it has some interesting advantages :
larger independence from the corpus, the knowledge base is easier to manage, and there are some capabilities to treat higher level knowledge.

We can distinguish among four knowledge classes in this system :

- some knowledge describing the corpus area (or thesaurus),

- some knowledge about the "user interface" permitting a most convivial user communication,
- some expert knowledge describing the behaviour of an information retrieval expert stored as production rules (1) : it constitutes the knowledge base of the expert system,
- an indexing base permitting document accesses.



We will expose the functional analysis of this system and the software architecture involved. We can decompose this architecture in :
- a set of specialized modules,
- a supervisor which manages them.

1.1. Functional analysis

We can decompose the process functionally in four principal phases (see figure 1) :
- preprocessing of the query :

The query is treated and understood i.e. we recognize the important concepts of the query and the logical items connecting them. This process implies a morphological and syntactical analysis of the query.
- the reference access :

For each recognized group, we access the associated references if they exist.
- the answer composition :

We use the logical items connecting the groups to determine the answer.
- the query reformulation :

Depending on the last obtained answers, we can reformulate the query if needed.

1.2. The software architecture

The software architecture corresponds to the functional analysis i.e. a specialized module is associated to each function and materializes it.
This set of specialized modules has to cooperate to perform the complete process of a query. This cooperation is managed by a supervisor which decides how the modules have to be sequentially related and controlled.

Two control variables are used during all the process :
- the user typology :

It concerns only the user expertise level in the concerned restricted area ; we do not want to evaluate his intrinsic expertise.
- the query formulation quality :

It consists of the adequation of the query groups to those recorded in the thesaurus, i.e. the number of transformations we have to apply to the query to adapt it to the thesaurus content.

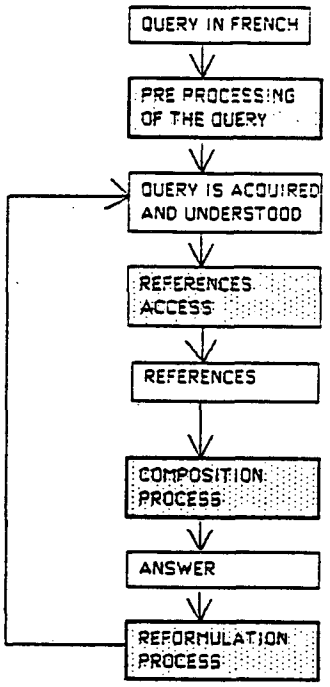


Figure 1 Functional decomposition

1.2.1. The control variables

We consider those two evaluations as some choice criteria in the resolution strategy of a query. We use them as parameters of the query processor expert system.
- user typology :

It allows the adaptation of the answer to the user and possibly the query reformulation in function of this level.
This evaluation could be made by statistical models applied to the query. These models are measures of query precision compared to the thesaurus best fitting items. Our basic hypothesis is that an accurate query denotes a specialist user.

Each query item is evaluated through a measure of specificity (stored in the thesaurus) and we can determine from this evaluation a global specificity measure for the query (we can use the mean of the query item measures for example).
For this evaluation we consider also the unknown terms of the query (if they exist) as very generic terms (we assign 0 to their associated measure).

- the quality of the query formulation :
It allows a control over the reformulation process. In fact, we do not want to reformulate a query with a weak formulation quality.
The quality evaluation is function of the number of syntactical breakdowns we have to do and of the corresponding loss of information resulting from this (see syntactical breakdown process below).

1.2.2. The supervisor

The query process can be considered as a sequence of specialized modules , each of them realizing part of the query process. We can point out that this sequence of operations is maximal and that it is interesting to determine the optimal sequence we have to execute in function of each particular query to process.
This goal of dynamic adaptation to the problem to solve is one of the most important characteristics of expert systems; this task of optimization can be achieved by a supervisor which is in fact an expert planner (10).
At each step the supervisor has to determine the following step. The knowledge required is

- the state of the system, i.e. the current query, the current step, the user typology and the quality of query formulation.
- some knowledge about this state which allows the planner to determine the following state.
Example :
IF undetermined answer THEN goto reformulation step

- some knowledge about particular types of query to which a predetermined path is associated.
Example :
the query is reduced to an isolated term
path : query , morphology , isolated term concept , answer.

Another important supervisor task is the answer evaluation, i.e. deciding whether the answer obtained is relevant or not. If it is, we can give the answer to the user otherwise we decide to reformulate (we can consider that this knowledge concerns a state knowledge).

The knowledge needed for evaluation is typically an expert one. It concerns three criteria :
the user typology, the number and the quality of answers.
There are two levels of evaluation :
- global evaluation : we determine if the answer is conform to our attempt.
- specific evaluation : we determine if each reference is very relevant.

*** global evaluation :**
We can describe the knowledge required as production rules.
Example :
IF <specialist user> and <little number of references> and <good quality of references>
THEN correct answer
IF <non specialist user> and <very little number of references>
THEN incorrect answer (the query is not adapted to the corpus)

The evaluation of the condition part is made by functions attached to the meta symbols.

*** specific evaluation :**
The goal is to select the references which have an important relevance measure (see composition process) to obtain very pertinent references.
The idea is to increase or decrease this measure by confronting it with another observation (the indexing terms list associated to this reference). If all the query terms are semantically equivalent, generic, specific or close to the indexing terms, we say that the adequation between the query and the reference is maximal.
These optimal references can be used to reformulate the query by extending it with the indexing terms (12, 13).

2. The detail of specialized modules specifications

We will detail the specifications of each specialized module. It is interesting to give first the exploitation mode we consider for the thesaurus (this constitutes the basic function of the modules).
We use three exploitation modes :
- generic terms of a given term :
They correspond to the given term without a qualification (i.e. without a qualificative adjective or without a noun complement, it depends on the syntax used) or to the terms obtained by the generic relations.
- specific terms of a given term :
They correspond to the given term with an additional qualification (if it exists in the thesaurus) or to the terms obtained by the specific relations. Adding a qualification consists in searching the set of thesaurus items including the given term.

These items must have an associated syntactical model which corresponds to an additional qualification compared to the initial model.

- common environment to a given set of terms:

It corresponds to the intersection of terms attached to the given set of terms in the thesaurus.

2.1. The preprocessing

The query preprocess comprises three steps (see figure 2) :

- morphological and syntactical analysis,
- pattern matching process,
- syntactical break process.

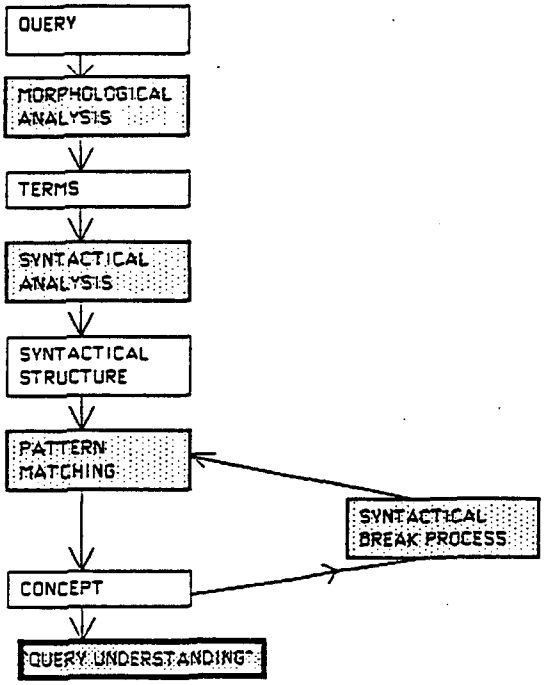


Figure 2 Query pre processing

2.1.1. Morphological and syntactical analysis

The morphological analysis consists, for each query item, in recognizing its lexical category and its lemma (or root, for example the singular form for a noun). We use morphological rules and dictionnaries in this process (morpho syntactical parser).

We want to consider the problem of the unknown terms we can detect during the analysis. We try first to infer their lexical category using the tools integrated in the parser and then to propose to the user a set of possible substitute terms belonging to the thesaurus content (see pattern matching process).

The goal of the syntactical analysis is to recognize the significant items of the query. Our basic hypothesis is that these items are noun groups (the verbs are not considered).
The other important query components are the logical utterances connecting groups which determine the answer composition. There are two problems to solve concerning these utterances :

- their recognition in the query (they can either be explicit as 'and' or implicit as a comma for example),
- the determination of their scope.

Up to now we have defined a recognition syntax based upon noun groups (we do not yet consider the logical utterances). This syntax can be expressed in the Backus-Naur Form formalism :

```
QUALIF_GROUP ::== A* N / N A*
NOUN_GROUP ::== QUALIF_GROUP P QUALIF_GROUP
CONCEPT ::== NOUN_GROUP*
A ::== (adjectives)
N ::== (nouns)
P ::== (prepositions)
```

In this application, an interesting problem to solve is that of the ambiguities resulting from the syntactical analysis. In fact, the parser can produce many correct analyzes for a same query and we have to determine the good one. The use of semantics is necessary. In our case the semantics of the corpus is given by the thesaurus. The idea is then a semantical validation of each analysis by confronting it to the thesaurus content. We choose either the analysis (if it exists) in which each item fits a thesaurus one or the nearest one (i.e. those which have lost least information during the confrontation).
This type of validation is equivalent to a query interpretation in terms of thesaurus concepts.

At the end of this step, we have a set of noun groups connected by logical utterances.

2.1.2. The pattern matching process

We confront each query group with the thesaurus items. These items are basically cliques (or maximal subgraphs) extracted from the term-term matrix constructed from the texts (4).
This cut of the graph is not unique; some other choices are practical (the set of paths of the graph for example).

The goal of the confrontation is to determine whether the query groups correspond to those extracted from the texts. If this confrontation fails we break the groups into noun subgroups (see syntactical breakdown process) and we iterate the pattern matching process until it succeeds.

At this level we use of synonymy relations to limit the loss of information resulting from the breakdown.

The confrontation criterion is simple, it is the inclusion test of the group in the cliques. The first clique corresponding to this criterion is selected and allows to access to the associated indexing terms (see references access process).
This confrontation can fail because there are some unknown items in the group. In this case, we use the common environment of other group members to propose some solutions to the user. The user can either accept the proposed solution in which case the process starts over again or reformulate his query.
When the concepts corresponding to all the query groups are selected, the system exposes its query understanding to the user who can either confirm or infirm it (we do not have some formulation strategies but we can give the user the execution sequence of the process).

2.1.3. Syntactical breakdown process

If the inclusion test of a query group fails, we break this group in one or many subgroups which have a higher probability in verifying the inclusion test (the clique syntax is a superset of the group syntax; so the process always stops, and possibly with isolated terms).
This breakdown process is performed according to syntactical rules which do not allow the generation of syntactically incorrect subgroups.

Example of decomposition rule :

Noun Preposition Noun

Noun Noun

This decomposition implies a loss of information (in the example we have lost the fact that the two nouns were related) which we have to evaluate. This is done to value the distance from the initial group and is used consequently to evaluate the quality of query formulation.
Since several syntactical decomposition are possible for a given group, the whole process is supervised by a general strategy aimed to minimize the loss of information associated to this operation.

Among several breaking solutions for a group, we select the one which corresponds to the isolation of most specific components (according to the thesaurus).
Evaluation of specificity is actually given by semantical information stored in the thesaurus.

2.2. References access process

We have to access the indexing terms associated to the selected concepts and consequently to the text references. This access is immediate from a given clique but two cases are considered :

- the group belongs to the indexing terms list, we have a direct access to the set of references as an answer.
- the group does not belong to the indexing term list (it corresponds to a concept of the area concerned but not to a concept extracted from the texts), there are no references corresponding to the group and it would be difficult to compose the answer (we have to reformulate probably).

2.3. Answer composition process

The composition selects the set of references corresponding to the logic items of the query. We have to consider the problem of the group implying no references. There are two cases to observe :

- the group occurs with an utterance of type 'or' :

In this case the set of references does not consider this union,
---> silence phenomenon

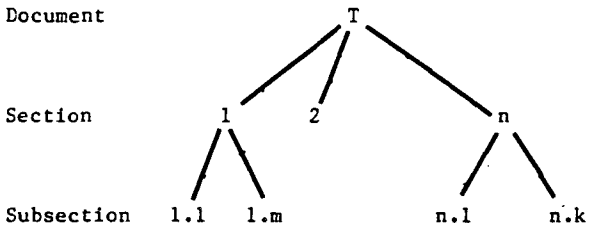
- the group occurs with an utterance of type 'and' or 'except' :

We can not restrict the resulting set of references,
---> noise phenomenon
These two problems have to be solved during the reformulation step.

We give to the user as answer a valuated references list. The valuation considers the indexing valuation and the way the answer is composed.
The indexing process we use (9) consists of an automatic process considering structural properties of texts (ie logical structure of a text). The indexing terms are not selected but are valuated (the measure belongs to (0,1)). Besides, an indexing term is related to a textual item (extracted from the logical structure) and not to an entire text.

To each tree node is associated an indexing terms list. An indexing term can be associated to many trees but it exists only once in the subtree it belongs to (it is associated to the node for which the indexing valuation is maximal).

Example :



The answer composition consists in retrieving a set of valuated textual items and not a set of texts as in classical systems. This composition is made through three basic logical operators ('and', 'or', 'except') :

- we search for a term conjunction :

Example : T1 AND T2

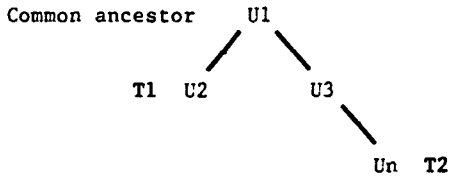
This corresponds to the intersection of the reference set associated to T1 and T2. This intersection is a tree intersection and some cases have to be considered :

* the trees are completely disconnected (ie they do not have a common ancestor),

ANSWER = NIL

* the trees are disconnected but they have at least a common ancestor,

Example :



The items U2 and Un are disconnected but it seems that the item U1 concerns effectively T1 AND T2 because its daughter items are concerned with. The problem is that U2 concerns more T1 than U1 because the indexing valuation is optimal for U2 (by definition) and it is the same for T2.
So we can give U1 as answer in weighing it with a coefficient MA which estimates the resulting noise :

$$MA = \frac{\text{pertinent informations}}{\text{total informations}} = \frac{\text{size U2+size Un}}{\text{size U1}}$$

ANSWER = the tree generated by the first common ancestor of the textual item containing T1 and the textual item containing T2 with the associated coefficient MA

* the trees are not disconnected,

ANSWER = higher level tree

Another problem is to determine the resulting measure associated to a reference belonging to the answer. We use for this a fuzzy measure (3) defined as :

$$M(Ri) = \frac{\min_j(P(Tj,i))}{1 - (\max_j(P(Tj,i)) - \min_j(P(Tj,i)))}$$

$M(Ri) \in (0,1)$
with $P(Ti,j)$ = indexing measure associated to the couple (Ti,Rj)

This measure limits the effects resulting from too dispersed terms (it have an average effect).

- we search for a term disjunction :

Example : T1 OR T2

This corresponds to the union (a tree union) of the reference set associated to T1 and T2. We use as resulting measure :

$$M(Ri) = \max_j(P(Tj,i)) \in (0,1)$$

- we search a term negation :

Example : NOT T1

We remove the set of references associated to T1.

With this three logical operators we can compose the final answer and order the reference set in function of the measure M (if M is near from 1 the reference is a very good answer).
We can filter some references in function of this ordering and of the user typology (this constitutes an expert knowledge).

Example :

IF <beginner user> and <large number of references>
THEN filter severely

IF <specialist user> and <little number of references>
THEN do not filter (a reference with a weak measure is yet interesting)

2.4. Reformulation step

This step allows to reformulate a query to obtain a set of answers more conform to the user needs.
We can distinguish three ways in reformulating

- the query is judged too accute compared to the corpus,

---> the query is broadened i.e. the too specific query terms are replaced by their generic terms.

- the query is judged too large compared to the corpus,

---> the query is narrowed i.e. the too generic query terms are replaced by their specific terms.

- the query is judged to be outside of the corpus (i.e. the formulation quality is weak or there are some unknown terms),

---> we try to reformulate with the thesaurus items which are semantically close to those recognized in the query (query shift).

The problem is to determine the query groups implying the incorrect answer and reformulate only them. Four cases are concerned :

- items with much information,
- items weak in information,
- items with a weak apparition frequency in the corpus,
- items with no associated references.

The reformulation level we apply is function of expert knowledge :
user typology and query formulation quality.

Example :
IF <specialist user> and <broadening> and <good formulation quality>
THEN broaden specific terms

CONCLUSION

This system tries to enhance the adaptation aspects :

- user adaptation with the evaluation of user typology,
- query adaptation with the use of a supervisor which determines the best fitting sequence of actions to be executed for a given query.

This is the central originality of this system with its expert system structure.

The future developments are based on two points :

- refining the specifications of some modules (syntax particularly), the thesaurus definition and the evaluation models of user typology and of query formulation quality.
- beginning to test some modules as reformulation and supervisor modules. For this we will use an existing inference engine CRIQUET (14) available on VAX 11/780. It will allow to test quickly the knowledge base we have obtained from previous studies.

A further step will be the realization of a specific inference engine able to integrate the control level and the specialized level.

REFERENCES

(1) Barret J.A, Bernstein M.I., (1977) Knowledge based systems : a tutorial. US Dept of Commerce

(2) Bassano JC., (1985) Un système convivial pour la recherche documentaire. Conférence RIAO85, Grenoble, 18-20 mars

(3) Bruandet MF., (1980) A conceptual framework for automatic and dynamic thesaurus updating in information retrieval systems. COLING80, Tokyo, sept.30-oct.4

(4) Bruandet MF., (1982) Concept notion for automatic and dynamic thesaurus updating. International Conference on systems documentation, ACM SIGDOC SIGOA, Los Angeles, jan 21-23

(5) Bruandet MF., (1985) Partial knowledge model for an information retrieval system. Conférence RIAO85, Grenoble, 18-20 mars

(6) Croft W.B., (1985) An expert assistant for a document retrieval system. Conférence RIAO85, Grenoble, 18-20 mars

(7) Defude B., (1983) Incorporation d'une part d'expertise dans un système documentaire automatisé. Rapport de DEA, INPG, Grenoble.

(8) Defude B., (1984) Knowledge based system versus thesaurus : an architecture problem about expert system design. 3rd ACM and BCS Symposium, Research and development in information retrieval, Cambridge, 2-6 july.

(9) Kerkouba D., (1985) Automatic indexing and structural properties of texts. Conférence RIAO85, Grenoble, 18-20 mars

(10) Nilsson N.J., (1980) Principles of artificial intelligence. Tioga Press, Palo Alto, CA.

(11) Pollitt A.S., (1982) An expert system as an online search intermediary. 5th International online meeting, London, 8-10 dec 1981

(12) Van Rijsbergen C.J., (1979) Information retrieval. 2nd edition, Butterworth, London.

(13) Salton G., McGill M.H., (1983) Introduction to modern information retrieval. New York, NY : McGraw Hill.

(14) Vignard P., (1984) CRIQUET un outil de base pour construire des sytèmes experts. RR 316, INRIA Sofia Antipolis.