Concept Based Retrieval in Classical IR Systems

H.P.Giger
Swiss Federal Institute of Technology (ETH)
Zurich, Switzerland
CSNET: giger@ifi.ethz.ch

This paper describes some aspects of a project with the aim of developing a user-friendly interface to a classical Information Retrieval (IR) System in order to improve the effectiveness of retrieval. The character by character approach to IR has been abandoned in favor of an approach based on the meaning of both the queries and the texts containing the information to be sought. The concept space, locally derived from a thesaurus, is used to represent a query as well as documents retrieved in atomic concept units. Dependencies between the search terms are taken into account. The meanings of the query and the retrieved documents (results of Elementary Logical Conjuncts (ELCs)) are compared. The ranking method on the semantical level is used in connection with existing data of a classical IR system. The user enters queries without using complex Boolean expressions.

1. Introduction

1.1. From Keywords to Information Structures

The basis for most Information Retrieval (IR) systems consists of determining the absence or presence of *keywords* in conjunction with their counting and distribution information. It can be said, therefore, that as a rule IR systems are based on a purely statistical examination of text. The *meaning* of these keywords (usually called terms or descriptors) is hardly ever used in today's IR systems; in fact, many authors explicitly state that these systems should not "attempt to 'understand' the content of a document" [VRI 79] when indexing or retrieving information. Since classical retrieval algorithms have a character by character approach to information handling, the meaning of the texts is ignored.

The performance of IR systems based on such statistical techniques has improved considerably over the years; nonetheless, it is expected that these techniques will soon reach their limit.

The keywords to be employed by a retrieval algorithm are usually issued by a user. In striking contrast to the statistical view taken by the IR system, the user invariably associates a *meaning* with such a keyword. In order to express what information is needed, the user reasons in terms of concepts. In addition, it is not until a query is

formulated, that these concepts are transformed into an appropriate set of keywords. As a result, the success of a query depends heavily on this transformation which is made by the user. It is still questionable, whether the transformation made subconsciously by the user corresponds exactly to the one made by the information supplier.

Given the reasons outlined above, we feel that the only way to improve IR processes is by studying *information structures* on a somewhat higher level than is currently done. The character by character approach to IR has to be abandoned in favor of an approach based on the *meaning* of both the queries and the texts containing the information to be sought. This also implies that the *syntactic* level has to be clearly separated from the *semantic* level: the former being based on *keywords*, the latter on *concepts* [BAE 84, SCH 87].

More precisely, a *keyword* is the character string to be used as *identifier* to the concept which is considered to be an adequate interpretation of that keyword. As a matter of fact, it is usually the case that there are several keywords identifying the same meaning (synonyms). Because several independent keywords can be assigned to the same text, these keywords sometimes identify concepts which have a similar meaning. Unfortunately, such similarities are not always detected by the classical IR techniques.

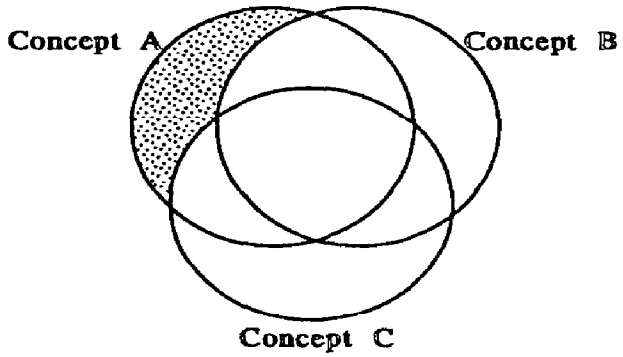

Figure 1-1

Consider a concept space consisting of three concepts A, B and C.

A possible base for the concept space would consist of the following eight *independent components*:

$\{(A\cap B\cap C),(A\cap B\cap\neg C),$
$(A\cap\neg B\cap C),(\neg A\cap B\cap C),$
$(A\cap\neg B\cap\neg C),(\neg A\cap B\cap\neg C),$
$(\neg A\cap\neg B\cap C),(\neg A\cap\neg B\cap\neg C)\}$

According to [WON 85] and [SCH 87], these components are called *atomic concepts*. With this base, every concept is the union of four atomic concepts.

1.2. Using Classical IR Systems

Today's commercially available IR systems provide access to many millions of facts and citations, covering a wide range of subject areas. As they are extremely useful, they are extensively used by all kinds of institutions both in the public and private sectors. Unfortunately, the logical design of most of these systems was done in the early seventies. Compared to todays techniques they are rather old-fashioned. However they are so well established and contain such a great deal of crucial information that it is very unlikely that they will change their appearance in the near future. So we are faced with the fact that a standard user will have to use the IR systems of today for some time in the future. That's one reason why we suggest to improve the usage of the available interface.

Another reason is that up till now, IR research has had very little impact on the design of operational IR systems. One of the reasons might be that most research has been conducted on small laboratory systems containing collections of only a few hundred documents. (e.g. the Extended Boolean Retrieval [SAF 83] has never been used in connection with classical IR systems). Most of these laboratory systems were not designed to run with large document collections (i.e. performance is not considered).

We are trying to overcome this lack of realism by developing a front-end to an existing classical IR system with the aim of improving retrieval effectiveness compared to the effectiveness of direct usage of the same system. In addition, the front end offers the user similar features to those of the system Caliban [FRJ 83]. As in Caliban, we are making extensive use of both modern graphics capabilities and the availability of inexpensive local storage. The belief is that more intelligent interfaces may become part of commercially available IR systems sometime in the future.

Every designer of a front-end to a classical system is restricted to the kind of information obtainable from the host databases as well as the amount of information that can be stored locally. One example of such a restriction is that there are usually no term frequencies within documents available in classical systems. Hence, no mechanisms involving these frequencies can be implemented [MOR 82].

Likewise, there are no *a priori* relevance data available relating the document collection to a query. In [FUH 87] it has been shown, that in our environment (classical IR systems in combination with search terms taken from a controlled vocabulary) no improvement over a simple coordination level matching can be achieved with probabilistic search term weighting. For this two and other reasons we did not use the probabilistic search term theory.

It's a fact, that it is immensely difficult for an average user to compose a query which will retrieve exactly those documents he or she envisaged [COO 83, HEI 82, MOR 82, SAL 82]. The user is often confronted with either too many or too few information items in the answer set.

The main reason for this situation is that most operational classical IR systems rely on Boolean queries which consist of unweighted terms and which operate on unweighted descriptors. The meaning of these terms and descriptors (their concepts) is usually not taken into account by the retrieval algorithms of commercial systems. Furthermore, Boolean IR systems are usually unable to *rank* the individual information items of the resulting set. This is the reason why we demonstrate in this section how lengthy Boolean queries can be decomposed into simpler parts. In subsequent sections, these parts are going to be used both by retrieval and by ranking algorithms.

It is to be noted that every Boolean expression in n terms t_i can be represented as a disjunction of conjunctions, the so called "Disjunct Normal Form" (DNF).

Example:

$(t_1 \text{ OR } t_2) \text{ AND } (t_3 \text{ OR } t_2) \text{ AND } t_4 \cong$

$(t_1 \text{ AND } t_2 \text{ AND } t_3 \text{ AND } t_4) \text{ OR}$
$(t_1 \text{ AND } t_2 \text{ AND NOT } t_3 \text{ AND } t_4) \text{ OR}$
$(t_1 \text{ AND NOT } t_2 \text{ AND } t_3 \text{ AND } t_4) \text{ OR}$
$(\text{NOT } t_1 \text{ AND } t_2 \text{ AND } t_3 \text{ AND } t_4) \text{ OR}$
$(\text{NOT } t_1 \text{ AND } t_2 \text{ AND NOT } t_3 \text{ AND } t_4)$

Such a DNF consists of a number of conjunctions which are called *Elementary Logical Conjuncts* (ELC), i.e. given n terms $t_1, t_2, \ldots, t_n$, an ELC is

$\bigcap_{i=1}^{n} (\mu_i t_i)$, where μ_i is either the identity or the NOT operator.

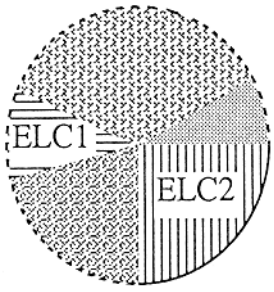

Figure 1-2

Notice that n terms yield 2^n possible ELCs. The set of documents resulting when an ELC is submitted as a query to a Boolean IR system is called the *result of the ELC.* It can be shown that the *sets of documents* induced by the ELCs of a Disjunct Normal Form are *disjoint.*

In order to rank the information items of the resulting set, a ranking algorithm usually assigns a *numerical value* to every item and arranges the items in decreasing order of these values. One of the most common ranking algorithms produces values which are a function of the number and weights of terms common to the query and the item description.

We pursue another approach: The information items are retrieved in disjoint groups, each of which is the result of an ELC. As the items of the result of an ELC contain identical search terms, the assumption is that these documents have a similar content. This is why we decided to assign the *same weight* to these information items. In order to do that, we define a virtual document $[D_m]$ consisting of all the not negated terms of ELC_m.

This virtual document $[D_m]$ determines the weight of the entire result of ELC_m. $w(ELC_m) = RSV([D_m])$, where $RSV([D_m])$ is the Retrieval Status Value of the virtual document $[D_m]$.

Note that all the individual documents represented by the same $[D_m]$ are weighted identically by our weighting algorithm. Likewise, this algorithm does not take into account document terms not belonging to $[D_m]$.

1.3. Dependencies between Search Terms

Most IR models assume *independence of document features.*

However, every human that handles information knows that the various concepts employed to describe information are far from being independent of each other. Human beings *implicitly* assume dependencies between the concepts. These dependencies have to be *defined exactly* as soon as an information *structure* is considered. Here an information structure is meant in the sense of well defined knowledge about a specific domain of discourse [SCH 87].

When handling information, such information structures play an important role, as they express the dependencies between concepts. Human indexers are used to dealing with information structures -- normally in the form of classifications and thesauri -- when analyzing and describing information, i.e. when classifying or indexing an information item. Nonetheless, at the other end of the IR process, at the actual *retrieval*, these information structures are *seldom used.*

Some theories have been suggested for use in IR systems which take advantage of *feature dependencies* :

- The coefficients used in the well known Bahadur Lazarsfeld Expansion are difficult to estimate as features in relevant documents.
- The Maximum Spanning Tree [VRI 79] model is restricted to dependencies between certain term pairs.
- The Generalized Term Dependence Model [YU 83] was never used in a practical environment.
- In the Generalized Vector Space Model (GVSM) [WON 85], atomic concepts are defined by co-occurrence data.
- In [WON 86], the GVSM is extended to handle situations where queries are specified as (extended) Boolean expressions.

Most of these theories are based on some statistical measure (e.g. co-occurrence data) instead of information structures. It is our belief that retrieval can be improved significantly if the retrieval algorithms are *based on* those *information structures* which were previously used to model the information base.

2. Information Structures

2.1. Thesaurus

We restrict ourselves to thesauri consisting of a finite set of terms T and of the three binary relations SY, NT, and RT. These relations are regarded as subsets of T×T such that

(a,b) ∈ SY iff b is an exact synonym of a; (a,b) ∈ NT iff b is a narrower term of a

(a,b) ∈ RT iff b is a related term of a

These or equivalent relations are supported by many commercial thesauri (e.g. [INS], [EUR]). Although these particular relations are usually explained in the introductory part of the thesaurus, it is often unclear what they really mean. The description of the RT-relation is notably often unsatisfactory. A formal semantics for the above relations has been developed in [SCH 87].

Figure 2-1 shows a minute thesaurus with nine terms that have been taken from [EUR]. The NT-, SY-, and RT-relations consist of six, two and six pairs respectively.

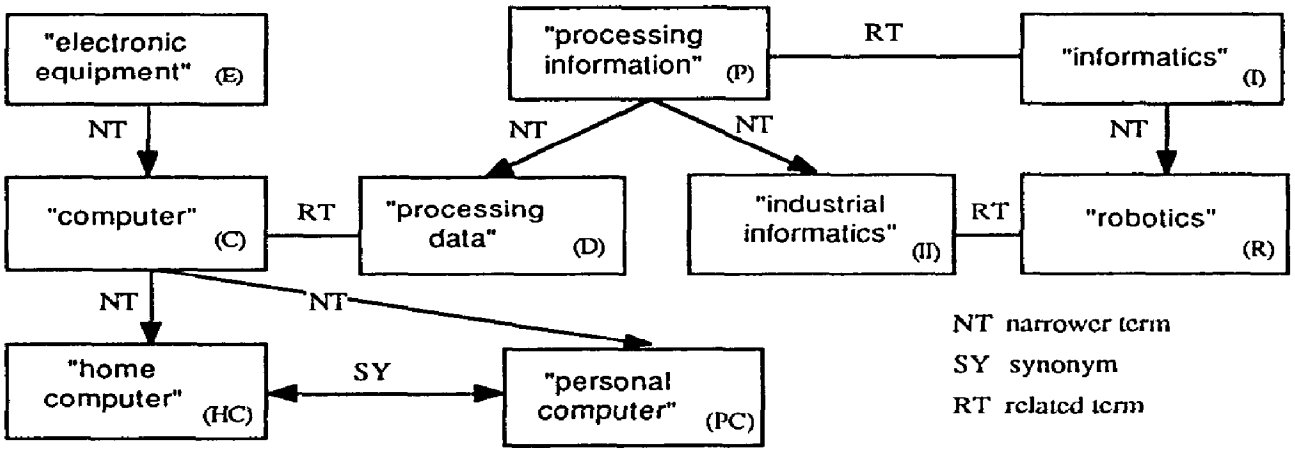

Figure 2-1

It is to be noted that in this thesaurus the term "computer" is not related to the term "informatics". But the term "computer" is indirectly related to the term "processing information" using the term "processing data" (interpretation of [SCH 87]).

2.2. Deriving Concept Spaces from Thesaurus

In this section we will present an algorithm to deduce a concept space representation from the thesaurus representation (Theoretical details to be found in [SCH 87]).

The concepts are represented as sets. Each member of these sets represents an atomic concept. Given a thesaurus T, the concept space is constructed according to the following algorithm :

- The algorithm uses two tables in order to construct the concept space. The first is called the *atomic concept->term table (ACT)* and contains records of the following type:
  - the *atomic concept type* (simple element set or double element set), the *atomic concept code* (integer number) and the *term codes array.*

The second is called the *term->atomic concept table (TAC)* in which the following records are stored:

- the *term string*, the *term code* (integer number), the *term type* (top-term or not top-term) and the *atomic concept code* array.

- In the first step of the algorithm an atomic concept is assigned to each term (if two or more terms are synonyms, then the assigned atomic concept must be the same) and a record is inserted into the TAC table.
- For each created atomic concept, a record is inserted into the ACT table with the following components: the atomic concept type (simple element set), the atomic concept code, the assigned term code (or the assigned term codes, in the case of synonyms);
- An atomic concept is assigned to each pair of related terms of the thesaurus and a record is inserted into the ACT table with the following components: the atomic concept type (double element set), the atomic concept code and the term codes of both terms in the pair.
- The atomic concept assigned to a pair of related terms will be assigned separately to each term of the pair (and obviously to synonyms of these terms).
- For each top-term of the thesaurus, its narrower terms are recursively traversed (with a depth-first procedure) and the stored information is copied. An internal mark is used to avoid double copying.

Using the thesaurus of figure 2-1 the following two tables result:

ACT: ctype	accode	tcodes
single	1	3
single	2	6
single	3	8
single	4	2,3
single	5	1,6
single	6	6,7
single	7	8,9
single	8	2,3,4,5
double	9	6,8
double	10	1,2,3,6
double	11	6,7,8,9

TAC: termstring	tcode	top	accodes
processing data	1	no	5,10
computer	2	no	4,8,10
electronic equipment	3	yes	1,4,8,10
home computer	4	no	8
personal computer	5	no	8
processing information	6	yes	2,5,6,9,10,11
industrial informatics	7	no	6,11
informatics	8	yes	3,7,9,11
robotics	9	no	7,11

Figure 2-2 shows the concept space of the example thesaurus given in Figure 2-1. This concept space consists of 8 concepts generating 12 atomic concepts: α_1, α_2, ..., α_{12}.

An example is the atomic concept α_6 which belongs to the concepts of PROCESSING INFORMATION and INDUSTRIAL INFORMATICS. In Figure 2-2, α_6 corresponds to the shadowed area. Atomic concepts can be expressed by a Boolean expression consisting of the original concepts. In the case of α_6, the corresponding Boolean expression is: $\alpha_6 = (\neg E \cap P \cap \neg I \cap \neg C \cap \neg D \cap I \cap \neg R \cap \neg H)$

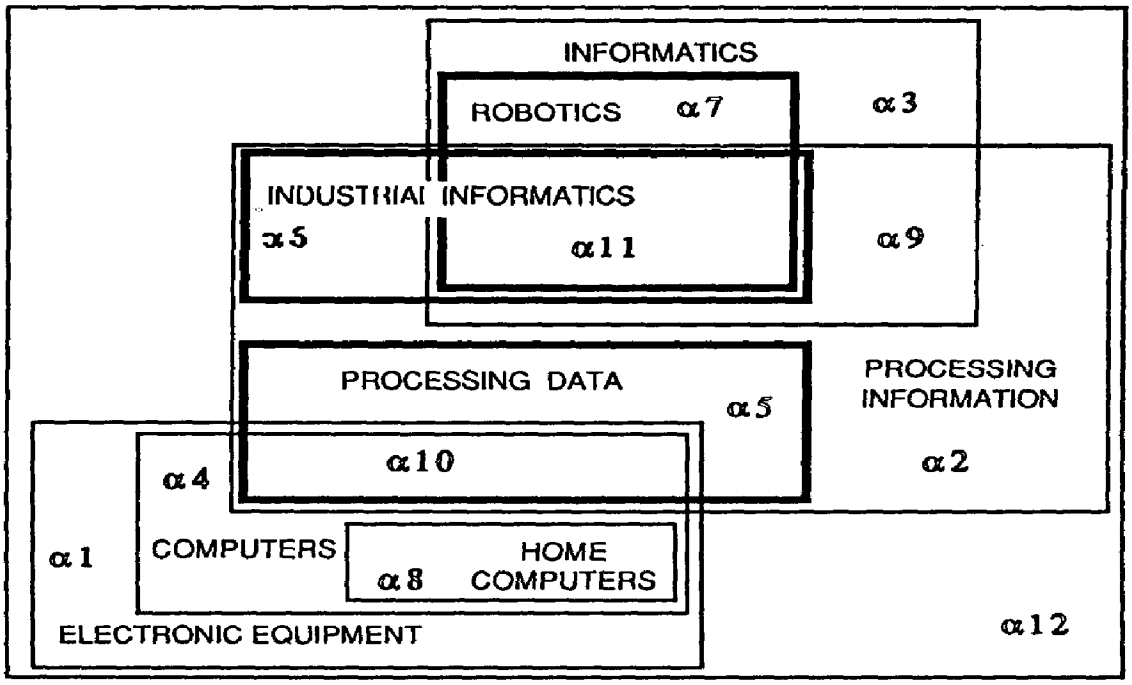

Figure 2-2

In the concept space, the dependences between terms are given by the topology of the corresponding concept sets. When the sets are disjoint, the terms are independent (e.g. computers and informatics are independent, which have disjoint atomic concept sets). A set is contained by the other if the corresponding concept has a restricted meaning (e.g. the concept of COMPUTER is a restriction of the concept of ELECTRONIC EQUIPMENT).

The above algorithm has been used to construct a concept space from the INSPEC thesaurus [INS] (with 7127 Terms, where some inconsistencies had to be removed first). This concept space consists of 14481 atomic concepts. The system presented in section 5 is working on documents contained in the INSPEC database.

3. Concept based Queries and Information Items

3.1. Concept Space Representation

Let us, examine a query based on the sample concept space of section 2.2.

Query:	ROBOTICS /	INDUSTRIAL INORMATICS /	PROCESSING DATA
Weights:	0.5	1.0	0.4

The query can be represented by the vector (0,0,0,0,1,1,1,0,0,1,1+λ,0) in the concept space.

The parameter λ is a *weighting factor* indicating the increasing importance of an atomic concept to which *several* original concepts have contributed. Such multiple contributions have already been acknowledged in [BAE 84] where λ is implicitly set to 1. We hypothesize that the actual value of λ depends on the particular concept space as well as on the user's interpretation. However, the effect of varying the value of λ is still under investigation.

Let us assume that there is a document described by the two keywords "*computers*" and "*processing information*". Such a document obviously contains information about the concepts "COMPUTER" and "PROCESSING INFORMATION" and is represented by the vector (0,1,0,1,1,1,0,1,1,1+λ,1,0) in the concept space. It is to be

noted that although the term "*robotics*" is not assigned to this particular document, the concept of "ROBOTICS" is partly included as can be seen from the vector representation (α_{11}) .

3.2. Concept Space based Similarity

In our model, query and documents are both represented in the vector space constructed from the atomic concepts. The concepts of the query and the retrieved documents are compared.

The similarity evaluation needs weighted atomic concepts. Bärtschi [BAE 84] proposed assigning a weight of 1 to every atomic concept.

Based on intuition we propose the following weighting scheme:

- The *weight of an atomic concept α_i with respect to the concept C_j* is defined as: $w_j(\alpha_i) := \frac{w(C_j)}{|\{\alpha_k | \alpha_k \in C_j\}|}$ if α_i occurs in C_j and 0 otherwise. The sum of weights of all atomic concepts with respect to a given Concept C_j is equal to the weight of Concept C_j.
- If an atomic concept occurs in different concepts, it might have different weights according to the above formula. There is no reason, for the weight of an atomic concept to decrease, if it occurs in another concept too. That's why we define the *weight of an atomic concept α_i* as $w(\alpha_i) := MAX(w_j(\alpha_i))$.

Example: $w_{II}(\alpha_6) := \frac{w(II)}{2}$; $w_P(\alpha_6) := \frac{w(P)}{6}$; $w_x(\alpha_6) := 0$ for all other concepts

$$w(\alpha_6) := MAX\left(\frac{1}{2}, \frac{0.4}{6}\right) = 0.5$$

Different similarity measures of the vector space model are compared in section 4.3. Within the framework of the vector space model Wong [WON 87] presented a statistical similarity measure between document and query. In his approach the assumption that term vectors are pairwise orthogonal is not required. Instead of using the standard cosine function his similarity is based on the entropy function.

4. Concept based Retrieval and Ranking

4.1. Basic Algorithm

Evaluating a query means comparing the query with information items and identifying those items which best satisfy the information need of the user. As pointed out in section 1, it is our objective to base the retrieval process on the semantic rather than the syntactic level. Using the concept space and ELCs we will develop an algorithm, how to retrieve documents from a classical IR System.

Figure 4-1 explains how the retrieval and ranking process is composed of a number of individual steps (step 1 to step 10).

1 The *query* is formulated as a set of weighted search terms taken from a thesaurus. We transform the terms of the user query into concepts. Based on intuition, we use as *weight of a concept C_j* the weight of the corresponding search term: $w(C_j) := w(Term_j)$.

2 The n-dimensional concept space is restricted to the k dimensions necessary to represent the atomic concepts induced by the query terms. The query is represented in this space as a weighted atomic concept vector.

3-4 Disjoint groups of similar documents are identified as the results of ELCs. In this section, only documents containing the issued query terms are identified,

whereas in section 4.2 documents containing similar terms (having atomic concepts in common with query concepts) are identified as well.

5-6 Using the virtual documents, ELCs can be represented by the atomic concepts.

7-8 This semantic representation is then used to compute a similarity (see definitions in section 3.2) between the query and various virtual documents [D_m] yielding a ranked list of retrieved documents. Note that the search term weights are taken into account when the ELCs are ranked.

Our ranking algorithm is based on the similarity between the ELCs and the query. Since these similarities depend exclusively on [D_m], they can be evaluated *even before the query is submitted. This is called a pre-ranking:*

9 ELCs with highest weight are submitted first.

10 The documents are retrieved set by set (the best documents are retrieved first), ranked according to the similarity computed previously.

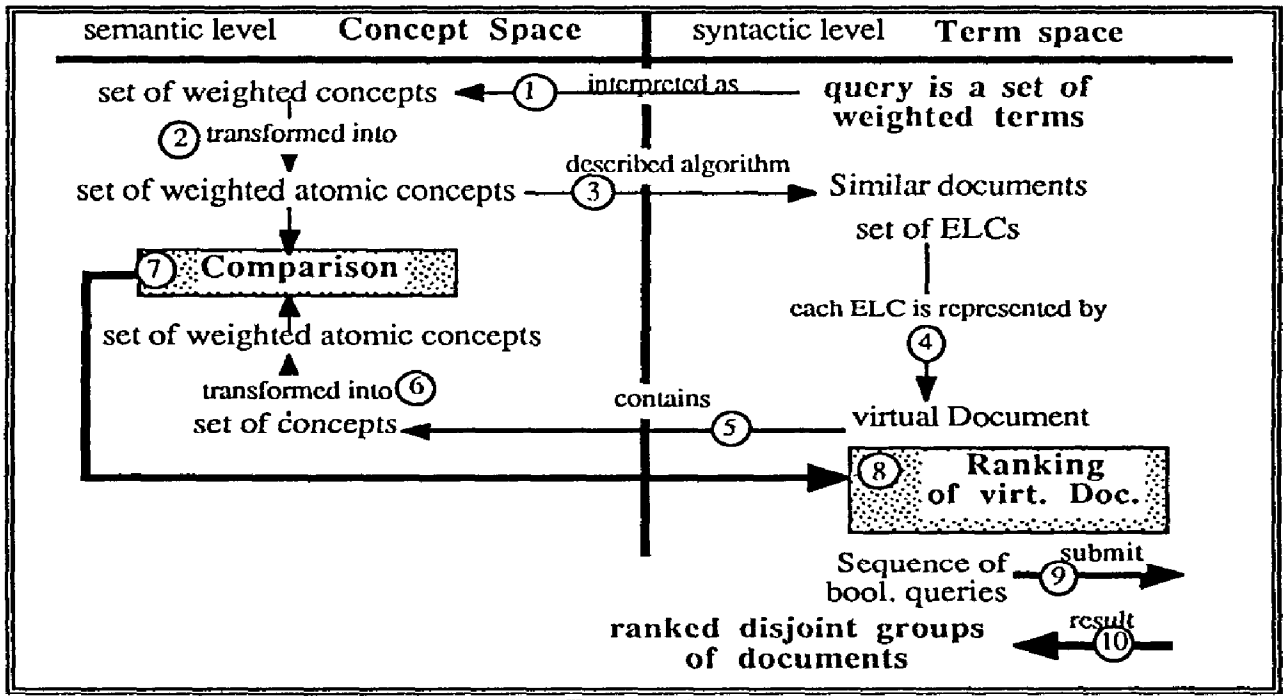

Figure 4-1

Let us examine a query based on the sample concept space of section 2.2.

Query: robotics / industrial informatics / processing data
Weights: 0.5 1.0 0.4

The representations of the query and the ELCs in atomic concepts are:

ELC	α_1	α_2	α_3	α_4	α_5	α_6	α_7	α_8	α_9	α_{10}	α_{11}	α_{12}
Query	(0,	0,	0,	0,	1,	1,	1,	0,	0,	1,	1+λ,	0)
1 {R,II,D}	(0,	0,	0,	0,	1,	1,	1,	0,	0,	1,	1+λ,	0)
2 {R,II}	(0,	0,	0,	0,	0,	1,	1,	0,	0,	0,	1+λ,	0)
3 {R,D}	(0,	0,	0,	0,	1,	0,	1,	0,	0,	1,	1,	0)
4 {II,D}	(0,	0,	0,	0,	1,	1,	0,	0,	0,	1,	1,	0)
5 {R}	(0,	0,	0,	0,	0,	0,	1,	0,	0,	0,	1,	0)
6 {II}	(0,	0,	0,	0,	0,	1,	0,	0,	0,	0,	1,	0)
7 {D}	(0,	0,	0,	0,	1,	0,	0,	0,	0,	1,	0,	0)

Let us examine the Boolean query "robotics AND industrial informatics AND NOT processing data" (on the syntactic level) which retrieves documents described by the terms "robotics" and "industrial informatics" but not by the term "processing data". These documents contain the concepts ROBOTICS and INDUSTRIAL INFORMATICS. In the chosen base of atomic concepts, their representation on the semantic level is (0,0,0,0,0,1,1,0,0,0,1+λ,0) .

The produced ranking is listed in section 4.3.

It would not be difficult to implement the extension mentioned by [CAK 87] (a query is a description of a finite set of "perfect" documents) to our algorithm.

4.2. Algorithm using Additional Concepts

In our example of 4.1 only the three concepts corresponding to the three query terms are used by the retrieval algorithm. However, parts of other concepts (such as COMPUTERS) are implicitly included in the query due to overlap in the concepts space. These additional concepts can be identified via the atomic concepts which create additional query terms and hence different and additional ELCs. Careful attention has to be paid to the fact that the number of ELCs may become very high. This is why we leave the selection of these additional terms to the user. The rest of the algorithms of section 4.1 remains unchanged.

Let us suppose that in the example above the user decided to extend the query by "processing information" (P), "informatics" (I), and "computers" (C). The following table shows the concept space representation of some ELCs being formed with the 3 original query terms and the 3 extension terms.

ELC		α_1	α_2	α_3	α_4	α_5	α_6	α_7	α_8	α_9	α_{10}	α_{11}	α_{12}
1	{R,II,D}	(0,	0,	0,	0,	1,	1,	1,	0,	0,	1,	1+λ,	0)
1b	{R,II,D,P,I,C}	(0,	1,	1,	1,	1+λ	1+λ,	1+λ,	1,	1+λ,	1+2λ,	1+3λ,	0)
1c	{R,II,D,P,I}	(0,	1,	1,	0,	1+λ,	1+λ,	1+λ,	0,	1+λ,	1+λ,	1+3λ,	0)
1d	{R,II,D,P,C}	(0,	1,	0,	1,	1+λ,	1+λ,	1,	1,	1,	1+2λ,	1+2λ,	0)
1e	{R,II,D,I,C}	(0,	0,	1,	1,	1,	1,	1+λ,	1,	0,	1+λ,	1+2λ,	0)
1f	{R,II,D,P}	(0,	1,	0,	0,	1+λ,	1+λ,	1,	0,	1,	1+λ,	1+2λ,	0)
1g	{R,II,D,I}	(0,	0,	1,	0,	1,	1,	1+λ,	0,	1,	1,	1+2λ,	0)
1h	{R,II,D,C}	(0,	0,	0,	1,	1,	1,	1,	1,	0,	1+λ,	1+λ,	0)
2	{R,II}	(0,	0,	0,	0,	0,	1,	1,	0,	0,	0,	1+λ,	0)
2b	{R,II,P,I,C}	(0,	1,	1,	1,	1,	1+λ,	1+λ,	1,	1+λ,	1+λ,	1+3λ,	0)
o													
o													
5	{R}	(0,	0,	0,	0,	0,	0,	1,	0,	0,	0,	1,	0)
5d	*{R,P,C}*	*(0,*	*1,*	*0,*	*1,*	*1,*	*1,*	*1,*	*1,*	*1,*	*1+λ,*	*1+λ,*	*0)*
5f	*{R,P}*	*(0,*	*1,*	*0,*	*0,*	*1,*	*1,*	*1,*	*0,*	*1,*	*1,*	*1+λ,*	*0)*
o													
o													
8b	*{P,I,C}*	*(0,*	*1,*	*1,*	*1,*	*1,*	*1,*	*1,*	*1,*	*1+λ,*	*1+λ,*	*1+λ,*	*0)*
8c	*{P,I}*	*(0,*	*1,*	*1,*	*0,*	*1,*	*1,*	*1,*	*0,*	*1+λ,*	*1,*	*1+λ,*	*0)*
8d	{P,C}	(0,	1,	0,	1,	1,	1,	0,	1,	1,	1+λ,	1,	0)
8e	{I,C}	(0,	0,	1,	1,	0,	0,	1,	1,	0,	1,	1,	0)
8f	{P}	(0,	1,	0,	0,	1,	1,	0,	0,	1,	1,	1,	0)
8g	{I}	(0,	0,	1,	0,	0,	0,	1,	0,	1,	0,	1,	0)
8h	{C}	(0,	0,	0,	1,	0,	0,	0,	1,	0,	1,	0,	0)

4.3. Comparison with other Algorithms

Since our algorithm represents both the query and the ELCs in terms of atomic concepts, our ranking algorithm *relies entirely on semantic concepts*. This is in contrast to the usual ranking algorithms which rely on terms.

In order to assess this kind of ranking, we compare it with three published algorithms which are using--at least to a certain extent--the ELC approach.

1) The Coordination Level Matching. A query consisting of a number of search terms for which the evaluation process favors those items in the output set which contain many matching index terms [COO 83].

2) The algorithm described in [SAL 82] can be explained with the function: $w(ELC) = \left(\sum_{i=1}^{n} x_i\right)^2 + \sum_{i=1}^{n} q_i$ where $x_i=1$ and q_i = Inverted Document Frequency (IDF) of the term i, if t_i occurs positively in the ELC; otherwise $x_i=0$ and $q_i=0$. The first part of the sum dominates the second, thus yielding an improved coordination level matching algorithm.

3) The Algorithm used in [HEI 82] and [JAM 79] can be explained with the function: $w(ELC) = \sum_{i=1}^{n} q_i$ as defined above.

Radecki [RAD 87] has used the virtual document approach too. He proposed to assign weights to the virtual documents based on the probability ranking principle(PRP). *In our opinion it is impossible to apply the PRP to classical IR systems.* In general the needed parameters can not be estimated.

However, it is not only the algorithm which influences the result of a query but also its similarity measure. This is why we measured the similarity in the concept space employing two different functions:

- the well known cosine measure.
- the entropy function introduced by [WON 87].

The table below illustrates the results of the various algorithms with the value of λ set to 0.5 and the weights of the atomic concepts according to the formulas of section 3.2 User assigned weights are assumed to be identical to the IDF.

	Independence Algorithms			Concept Space	
	1	2	3	Cosine	Entropy
Rank1	ELC1 (3)	ELC1 (10.9)	ELC1 (1.9)	ELC1 (1.0)	ELC1 (1.0)
Rank2	ELC2 (2)	**ELC2** (5.5)	**ELC2** (1.5)	**ELC4** (0.949)	**ELC4** (0.924)
	ELC3 (2)	**ELC4** (5.4)	**ELC4** (1.4)	**ELC2** (0.919)	**ELC2** (0.885)
	ELC4 (2)	ELC3 (4.9)	ELC6 (1.0)	ELC6 (0.868)	ELC3 (0.847)
	ELC5 (1)	ELC6 (2.0)	ELC3 (0.9)	ELC3 (0.854)	ELC6 (0.797)
	ELC6 (1)	ELC5 (1.5)	ELC5 (0.5)	ELC5 (0.769)	ELC5 (0.705)
	ELC7 (1)	ELC7 (1.4)	ELC7 (0.4)	ELC7 (0.393)	ELC7 (0.403)

All the five algorithms rank ELC1 highest. Algorithm 1 and 2 simply take into account the number of query terms assigned to a document, whereas Algorithm 3 accounts for the fact that a single positively occurring term may have a higher weight than the sum of the weights of two other positively occurring terms. A common weakness of the first three algorithms is the lack of information about the semantics of the terms.

A good way of comparing the different results is to observe the positions of ELC2 {R,II} and ELC4 {II,D}. The concepts "ROBOTICS" and "INDUSTRIAL INFORMATICS" depend on each other whereas the concepts "INDUSTRIAL INFORMATICS" and "PROCESSING DATA" are independent.

The matching algorithm 1 is unable to distinguish the virtual document D_2 from D_4, whereas algorithms 2 and 3 rank D_2 higher than D_4. Contrastingly, the algorithms based on the concept space prefer D_4 over D_2. Based on common sense, this order seems to be better as documents covering independent search concepts should be ranked higher than documents covering dependent search concepts, especially if these concepts have similar weights.

Based on various such experiments we found:

A model using atomic concepts performs better than a model where no dependencies between concepts or terms are taken into account. This is not surprising and has been shown by Wong [WON 87] as well. In addition, Wong pointed out that the entropy function yields better results than the standard cosine function.

In contrast to the example above our tests with more terms showed differences between the cosine and the entropy function. We cannot yet verify, which one is better in general. Experiments are currently under way investigating the effect of varying λ, using different similarity measures and assigning different weights to atomic concepts.

As pointed out in 4.2, a query can be extended by additional terms which are identified through the atomic concepts they have in common with concepts of the original query. If such an extended query is decomposed into ELCs, usually many more ELCs result than would be the case with the original query.

The same process which was previously described assigns weights to these ELCs. With this data we implemented another way of ranking documents, which itself produced some surprising results.

Let us take our old example and compare the rank order of the following four virtual documents: {R,P,C},{R,P},{P,I,C},{P,I} (= [D5d], [D5f], [D8b] and [D8c]). See the Appendix for weights produced with the various similarity measures.

Matching algorithms 1 to 3 are unable to distinguish all the extensions due to a particular original ELC (e.g. ELC5, ELC5b, ELC5c, ... ELC5h get the same ranking). For that reason {R,P,C} and {R,P} appear towards the end of the list having the same weight as the virtual documents {R}. The additional similar terms in {R,P,C} and {R,P} are not considered. Likewise, documents which do not contain at least one of the original search terms are not found by these algorithms (e.g. {P,I},{P,I,C}).

The concept based algorithm ranks {P,I} at a top position because this document contains the broader terms of all query terms. These broader terms are related in the same way as the query terms. For similar reasons {R,P} is ranked at a top position.

The positions of {R,P,C} and {P,I,C} have been improved by our concept based algorithm because the additional non query concepts contain atomic concepts covered by the query. The relations within the virtual documents are similar to the relations within the query concepts.

5. Overview of the System CB-AIR

An experimental interface to the Boolean IR system Data-Star of Radio Switzerland was developed in 1986 and 1987 using the concept space approach. This interface called CB-AIR (Concept Based - Advanced Information Retrieval) consists of three parts:

- A user interface based on the graphics screen, the three button mouse, and the keyboard of the personal computer Ceres [EBE 87] allows the user to select terms graphically from a local copy of a thesaurus.
- An intermediary part constructs weighted ELCs, performs pre-searching, predicts the results of ELCs from previous results of the same session, generates the necessary requests, translates the requests into the command language of Data-Star, and interprets the results.
- A terminal emulator performs some routine functions.

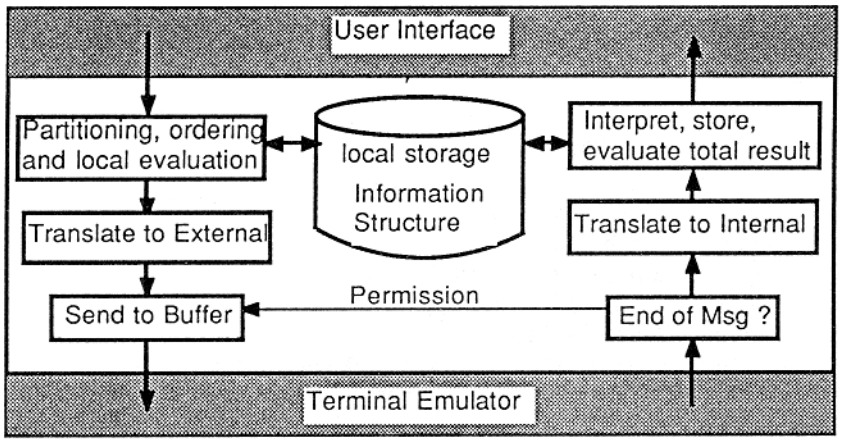

Figure 6-1

The system is written in MODULA-2. Due to the modularity of the software, weighting functions can easily be exchanged and compared. The algorithm of Jamieson as well as concept based similarity functions are implemented.

The interface built allows the user to express queries in a more natural way than is the case with most existing classical systems. In particular, it is no longer necessary to formulate queries in the form of complex Boolean expressions. In addition, retrieval results are not presented to the user as huge unstructured sets but as lists of ranked items.

Further details on the implementation and on the results of the experiments carried out with our method shall be presented at a later date.

6. Conclusion

We are convinced that the method described above improves retrieval results when using classical Boolean IR evaluation. This is mainly due to the fact that queries and information items are no longer compared on a character by character basis, but rather on the basis of their meaning. Additionally, significant documents not found by standard methods are returned at the beginning of the ranked result list.

These improvements are accomplished by translating terms into concepts, and by decomposing these concepts into elementary semantic units. We call these units *atomic concepts* and use them as the base of the concept space, a space which can be derived from such traditional information structures like thesauri. The query itself consists of weighted terms which have been issued by the user of the system. It is then decomposed into Elementary Logical Conjuncts (ELCs) to which weights are assigned by the algorithm described above.

The query decomposition and ranking produces the weighted ELCs which are sent to the commercial Boolean IR system. The end user interacts exclusively with the workstation through a user friendly interface, an interface which allows to formulate

queries without using Boolean operators. The output is delivered in a ranked order. In conclusion, we would like to emphasize that our system does not operate on an experimental test collection but on the bibliographic database of a commercial information supplier.

Acknowledgements

I would like to thank Prof. H.P. Frei for guiding and helping me, especially with the introduction; P.Schäuble and B.Teufel for discussions, comments and ideas; as well as G. Della Bruna, A. Göllü, G. Jäger, C. Lanz, F. Menozzi, M. van Scherpenzeel and D. Stieger, who contributed to the software.

References

[BAE 84] Bärtschi, M.: *Term Dependence in Information Retrieval Models*. D.Sc. Thesis ETH, No. 7525, ETH Zurich, 1984.

[COO 83] Cooper W. : *Exploiting the Maximum Entropy Principle to Increase Retrieval Effectiveness*, Journal of the American Society for Information Science. 34(1): 31-39; 1983.

[EBE 87] Eberle H.: *Hardware Description of the Workstation Ceres*, Swiss Federal Institute of Technology (ETH) Zurich, Switzerland.Institut für Informatik, Report Nr. 70.

[EUR] *EUROVOC Alphabetical thesaurus*. Annex to the Index of the Official Journal of the European Communities, Luxembourg, 1984.

[FRJ 83] Frei H.P. , Jauslin F. : *Graphical Presentation of Information and Services: A User-Oriented Interface*, Information Technology: Research and Development. 2: 23-42; 1983.

[FUH 87] Fuhr P. : *Probabilistic search term weighting - some negative results*, Proc. of the 10th ACMSIGIR Conf. on R&D in Information Retrieval, ACM, New Orleans: 13-18;1987

[HEI 82] Heine M.H. : *A Simple Intelligent Front End for Information Retrieval Systems using Boolean Logic*, Information Technology and Development. 1: 247-260; 1982.

[INS] *INSPEC Thesaurus*. The Institution of Electrical Engineers, 1987.

[JAM 79] Jamieson S. : *The Economic Implementation of Experimental Retrieval Techniques on a Very Large Scale using an Intelligent Terminal*, Proc. of the 2nd ACMSIGIR Conf. on R&D in Information Retrieval, ACM, Dallas: 45-51; 1979.

[MOR 82] Morrisey J. : *An Intelligent Terminal for Implementing Relevance Feedback on Large Operational Retrieval Systems*, Research and Development in Information Retrieval, Lecture Notes in Computer Science. 146: 38-50; 1982.

[RAD 87] Radecki T. : *Exploiting the Probability Ranking Principle to Increase the Effectiveness of Conventional Boolean Retrieval Systems*, Proc. of the 1st Int. Conf. on Bibliometrics and Theoretical Aspects of Information Retrieval, Limburg University, Belgium; 1987. (in print)

[SAL 82] Salton G. : *A Blueprint for Automatic Boolean Query Processing*, SIGIR Forum, ACM. 17(2): 6-24; 1982.

[SAF 83] Salton G. , Fox E. : *Extended Boolean Information Retrieval*, Communication of the ACM, 26(11):1022-1036; 1983

[SCH 87] Schäuble P. : *Thesaurus Based Concept Spaces*, Proc. of the 10th ACMSIGIR Conf. on R&D in Information Retrieval, ACM, New Orleans: 254-262; 1987

[VRI 79] van Rijsbergen C.J.: *Information Retrieval, Second Edition.* Butterworths, London, Boston, 1979.

[WON 85] Wong S.: *Generalized Vector Space Model for Boolean Query Processing.* Proc. of the 8th ACMSIGIR Conf. on R&D in Information Retrieval, ACM, 18-25, 1985

[WON 86] Wong S.: *On Extending the Vector Space Model for Boolean Query Processing.* Proc. of the 9th ACMSIGIR Conf. on R&D in Information Retrieval, ACM, Pisa: 175-185; 1986

[WON 87] Wong S.: *A Statistical Similarity Measure.* Proc. of the 10th ACMSIGIR Conf. on R&D in Information Retrieval, New Orleans: 3-12; 1987

[YU 83] Yu. T: *A Generalized Term Dependence Model in Information Retrieval,* Information Technology: Research and Development. 2: 129-154; 1983

Appendix Weights of ELCs

Independence Model
Used Algorithm: #3

ELC	Weight
1 to 1h	1.9
2 to 2h	1.5
3 to 3h	1.4
4 to 4h	1.0
5 to 5h	0.9
(*[R]*..*[R, P, C]*, *[R, P]*..)	
6 to 6h	0.5
7 to 7h	0.4
8 to 8h	0
(..*[P, I, C]*, *[P, I]*..)	

Concept space
Cosine - Function

ELC	Weight
[R, II, D]	1.0
[II, D, I]	1.0
[R,P]	*1.0*
[P,I]	*1.0*
[R, II, D, P, I]	0.99869
[R, II, D, P, I, C]	0.99760
[R, II, D, P]	0.99587
[II, D, P, I]	0.99587
[R, II, P, I, C]	0.99583
[R, II, D, C]	0.99522
[II, D, I, C]	0.99522
[R,P,C]	*0.99522*
[P,I,C]	*0.99522*
[R, II, P, C]	0.99451
[II, P, I, C]	0.99451
[R, II, P]	0.99331
[II, P, I]	0.99331
[R, II, P, I]	0.99303
[R, II, D, P, C]	0.99205
[II, D, P, I, C]	0.99205
[R, D, P]	0.99128
[D, P, I]	0.99128
[R, D, P, I]	0.99104
[R, II, D, I, C]	0.99061
[R, P, I, C]	0.99061
[R, II, D, I]	0.99043
[R, P, I]	0.99043
[R, D, P, I, C]	0.98565
[R, D, P, C]	0.97924
[D, P, I, C]	0.97924
[R, II, C]	0.97883
[II, I, C]	0.97883
[R, II, I, C]	0.96996
[R, II]	0.95720
[II, I]	0.95720
[R, II, I]	0.94910
[II, P]	0.94850
[II, P, C]	0.94769
[II, D, P]	0.94727
[II, D]	0.94727
[P]	0.94727
[II, D, P, C]	0.94030
[II, D, C]	0.93469
[P, C]	0.93469
[II, C]	0.92603
[D, P]	0.92449
[II]	0.90447
[D, P, C]	0.90392
[R, D, I]	0.85135
[R, D, I, C]	0.84783
[R, D]	0.84526
[D, I]	0.84526
[R, D, C]	0.82684
[D, I, C]	0.82684
[R, I, C]	0.82648
[R, C]	0.82299
[I, C]	0.82299
[R]	0.80085
[I]	0.80085
[R, I]	0.80085
[D]	0.28943
[D, C]	0.28381
[C]	0.20466

Concept space
Entropy Function

ELC	Weight
[R, II, D]	1.0
[II, D, I]	1.0
[R,P]	*1.0*
[P,I]	*1.0*
[R, II, D, P, I]	0.99952
[R, II, D, P, I, C]	0.99849
[R, II, D, P]	0.99720
[II, D, P, I]	0.99720
[R, II, D, C]	0.99674
[II, D, I, C]	0.99674
[R,P,C]	*0.99674*
[P,I,C]	*0.99674*
[R, II, P, I, C]	0.99665
[R, II, P, C]	0.99597
[II, P, I, C]	0.99597
[R, D, P, I]	0.99574
[R, II, P]	0.99526
[II, P, I]	0.99526
[R, II, D, I]	0.99525
[R, P, I]	0.99525
[R, II, D, I, C]	0.99518
[R, P, I, C]	0.99518
[R, D, P]	0.99452
[D, P, I]	0.99452
[R, II, D, P, C]	0.99443
[II, D, P, I, C]	0.99443
[R, II, P, I]	0.99412

All ohter ELCs with weights less than 0.994