Information Retrieval as Semantics Transformation

F.A. Grootjen and Th.P. van der Weide*

Abstract

In this paper we present Information Retrieval as a semantics transformation problem. We describe a general theory for deriving concepts, and discuss 2 special cases: the vector model and the set model. The vector model leads to concepts derived by latent semantic indexing using the singular value decomposition. The set model leads to Formal Concept Analysis. We discuss the relation between the resulting systems of concepts. Finally, the dual search engine is introduced.

Keywords: formal concepts, latent semantics, dual search engine, information retrieval, knowledge discovery

1 Introduction

The information retrieval problem may be seen as a semantics transformation problem. We assume a searcher has some mental model of the world. It is within this model that a searcher is aware of a knowledge gap. The searcher will try to find information objects that help the searcher to fill this gap. Unfortunately, information objects are not easy to find. In order to facilitate finding information objects, a typical solution is to construct a catalogue which offers the searcher the opportunity to have a more directed avenue for search.

Traditionally (see [7]), Information Retrieval systems try to relate a set of *descriptors* (or terms) with a set of *information objects* (or documents). Since single terms have only limited descriptive power, Information Retrieval systems allow individual terms to be combined into bigger semantical units. These units as referred to as *intentional objects*. How terms are combined to form intentional objects depends on the actual Information Retrieval system. Likewise, a combination of documents will be called an *extensional object*. We will call an Information Retrieval system *dual* if it can transform intensional objects into extensional objects, and vice versa.

*University of Nijmegen, Faculty of Science, Mathematics and Computing Science, P.O. Box 9010, 6500 GL Nijmegen, The Netherlands

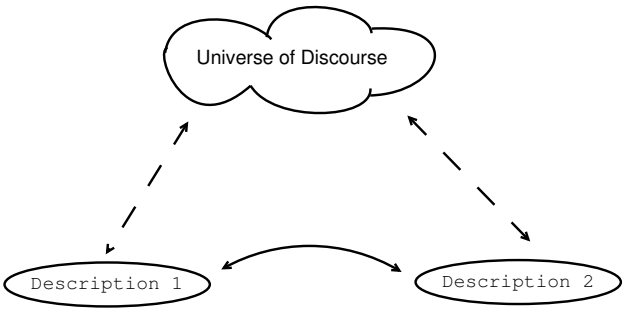

Figure 1: Different models of real world

An example of an intentional object is a query while an example of a extensional object is the outcome of search. Being a combination of terms, intentional objects can be used to capture the meaning of a document, while extensional objects (being a combination of documents) can be used to capture the meaning of a terms. As such, a dual Information Retrieval system may be seen as a mutual semantics assigning system.

In this paper, we focus on dual Information Retrieval systems. In section 2 we show how different views on the real world can be combined to recognize concepts as semantical fixed points. In section 3 we focus on the interpretation of concepts in the context of the vector model, and find a relation with the latent semantical indexing approach ([3]). In section 4 we study concepts in the Set Model, and find the relation with formal concept analysis ([4]). In section 5 we apply this general theory by introducing the dual search engine. Finally, in section 6 we present some conclusions.

2 Dual Information Retrieval systems

Consider a dual Information Retrieval system as described in the previous section. Let $\mathcal{I}$ be its set of intentional objects and $\mathcal{E}$ its set of extensional objects. We assume an equivalence relation $\equiv_i$ for comparing intentional objects expressing their similarity, and its counterpart $\equiv_e$ on extensional objects (we will leave out the indices when no confusion is likely to occur).

Intentional and extensional objects are assigned a meaning in terms of each other. The function **match** : $\mathcal{I} \to \mathcal{E}$ interprets intentional objects in terms of extensional objects, the function **index** : $\mathcal{E} \to \mathcal{I}$ does it the opposite way (see figure 2). We assume the assignment of meaning to be closed under similarity:

DIRS 1. $q_1 \equiv_i q_2 \implies \mathbf{match}(q_1) \equiv_e \mathbf{match}(q_2)$

DIRS 2. $d_1 \equiv_e d_2 \implies \mathbf{index}(d_1) \equiv_i \mathbf{index}(d_2)$

These requirements are referred to as the *similarity closure assumptions*. The resulting dual Information Retrieval system is denoted as

$$\langle \langle \mathcal{I}, \equiv_i \rangle, \langle \mathcal{E}, \equiv_e \rangle, \mathbf{match}, \mathbf{index} \rangle$$

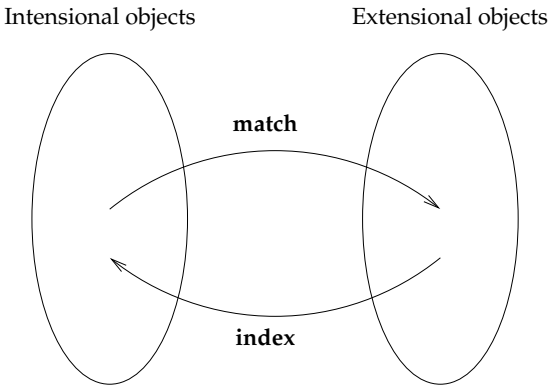

Figure 2: A dual Information Retrieval system

The meaning assigning functions **match** and **index** are not assumed to be inverse to each other. As a consequence, mutual sharing of meaning is a special property. One might wonder what objects are invariant under subsequent application of **index** and **match**. We introduce:

$$\begin{aligned} \mathbf{Qpc} &= \{q \mid \mathbf{index}(\mathbf{match}(q)) \equiv_i q\} \\ \mathbf{Dpc} &= \{d \mid \mathbf{match}(\mathbf{index}(d)) \equiv_e d\} \end{aligned}$$

The functions **match** and **index** can be restricted and seen as mappings between these sets **Qpc** and **Dpc**:

Lemma 1

1. $q \in \mathbf{Qpc} \implies \mathbf{match}(q) \in \mathbf{Dpc}$
2. $d \in \mathbf{Dpc} \implies \mathbf{index}(d) \in \mathbf{Qpc}$

Furthermore, as a direct consequence of the similarity closure assumptions, this restricted functions respect similarity:

Lemma 2

1. $q \in \mathbf{Qpc} \wedge q \equiv_i q' \implies q' \in \mathbf{Qpc}$

2. $d \in \mathbf{Dpc} \wedge d \equiv_e d' \implies d' \in \mathbf{Dpc}$

It is easily verified that the restriction of the relation $\equiv_i$ to **Qpc** still is an equivalence relation. Let $\mathbf{Qc} = \mathbf{Qpc} \backslash \equiv_i$ be the corresponding set of equivalence classes. The equivalence class containing q is denoted as $[q]_i$. The same holds for the restriction of $\equiv_e$ to **Dpc**. The set **Dc** is introduced analogously, $[d]_e$ will denote equivalence class of $d \in \mathbf{Dpc}$.

The functions $m : \mathbf{Qc} \rightarrow \mathbf{Dc}$ and $i : \mathbf{Dc} \rightarrow \mathbf{Qc}$ are the generalizations of the restricted versions of **match** and **index** over equivalence classes. Let $qc \in \mathbf{Qc}$ be some equivalence class from $\mathbf{Qpc} \backslash \equiv_i$ then $m(qc)$ is obtained by taking any q from class qc, and taking the equivalence class containing $\mathbf{match}(q)$. As a result of lemma 1 we have $\mathbf{match}(q) \in \mathbf{Dpc}$. As a consequence of lemma 2 the resulting class does not depend on the actual q taken from qc. The function i is introduced analogously:

$$\begin{aligned} m(qc) &= [\mathbf{match}(q)] \quad \text{for } q \in qc \\ i(dc) &= [\mathbf{index}(d)] \quad \text{for } d \in dc \end{aligned}$$

This brings us to a main result of this paper:

Theorem 1
The functions m and i are inverse functions.

Proof:

1. Assume $qc \in \mathbf{Qc}$, and let $q \in qc$. As $q \in \mathbf{Qpc}$, we conclude $\mathbf{index}(\mathbf{match}(q)) \equiv_i q$, and thus $qc = [\mathbf{index}(\mathbf{match}(q))]_i$. Consequently, $i(m(qc)) = qc$.

2. Assume $dc \in \mathbf{Dc}$, and let $d \in dc$. As $d \in \mathbf{Dpc}$, we conclude $\mathbf{match}(\mathbf{index}(d)) \equiv_e d$, and thus $dc = [\mathbf{match}(\mathbf{index}(d))]_e$. Consequently, $m(i(dc)) = dc$. $\diamond$

Next we focus on combinations of intentional and extensional objects. Symmetry in mutual meaning assignment for such combinations is a central issue in text and data mining environments. Such combinations are referred to as concepts.

Definition 1 *A pair (qc, dc) is called a concept if:* $m(qc) = dc \wedge i(dc) = qc$

Let C be the set of concepts, then the following is a direct consequence of theorem 1:

Theorem 2
$\mathcal{C} = \{(qc, m(qc)) \mid qc \in \mathbf{Qc}\} = \{(i(dc), dc) \mid dc \in \mathbf{Dc}\}$

The following sections show how two different IR models, the vector model and the set model use this observation and basically follow the same line of reasoning. However, the resulting conceptual view is of a rather different nature. In the vector model, the focus is on finding a minimal set of concepts spanning the conceptual space available in a document collection. With each concept a value is associated that describes the relevancy of that concept in the collection. This provides the opportunity to eliminate concepts that are a consequence of semantical noise.

The set model results in a much more refined look, trying to give a complete view on the concepts in the collection, providing an ontology that describes the nature of concepts in terms of generality. This conceptual view will usually be much larger than the conceptual view obtained by the vector model. However, in cases likes looking for a needle in the haystack, the searcher actually may be looking for rare information that would be interpreted as noise in the vector model approach.

3 The vector model

Assume a set D of documents and a set T of terms, and a function $A : \mathcal{D} \times \mathcal{T} \rightarrow [0, 1]$. This function A is usually represented as a matrix. The value $A_{d,t}$ describes the degree in which document d is about term t.

In the vector model intentional objects are linear combinations of terms, referred to as document vectors. On the other hand, extensional objects are seen as a linear combination of documents. As a consequence, both intentional and extensional objects are seen as vectors. The equivalence relations $\equiv_1$ and $\equiv_e$ are straightforward: two vectors are considered to be equivalent if they are a (positive) linear combination of each other:

$$x \equiv y \Longleftrightarrow \exists_{\lambda>0} [x = \lambda y]$$

One might say that x and y cover the same topic, but only differ in degree of intensity, which is expressed by the scalar λ.

The functions **match** and its dual function **match** are defined as follows:

$$\begin{aligned} \textbf{match}(q) &= Aq \\ \textbf{index}(d) &= A^T d \end{aligned}$$

These functions satisfy the similarity closure assumptions:

Lemma 3

1. $q_1 \equiv_i q_2 \implies \textbf{match}(q_1) \equiv_e \textbf{match}(q_2)$

2. $d_1 \equiv_e d_2 \implies \textbf{index}(d_1) \equiv_i \textbf{index}(d_2)$

Proof:

1. Suppose $q_1 \equiv_i q_2$, then $q_1 = \lambda q_2$ for some $\lambda > 0$. Consequently:

$$\textbf{match}(q_1) = Aq_1 = \lambda Aq_2 = \lambda \textbf{match}(q_2)$$

   and thus $\textbf{match}(q_1) \equiv_e \textbf{match}(q_2)$

2. Analogously.

$\diamond$

Invariance under subsequent application of **match** and **index** leads to the eigenvectors of $A^T A$ and AA^T respectively:

Lemma 4

$$\begin{aligned} \textbf{Qpc} &= \left\{q \mid \exists_{\lambda>0} \left[A^T Aq = \lambda q\right]\right\} \\ \textbf{Dpc} &= \left\{d \mid \exists_{\lambda>0} \left[AA^T d = \lambda d\right]\right\} \end{aligned}$$

Finding the eigenvalues and eigenvectors of $A^T A$ for a given matrix A is called *Singular Value Decomposition* (SVD). This approach, well known as Latent Semantic Indexing in IR research ([3]), is commonly used to sort out noise and relevant data. The idea behind this decomposition is that eigenvectors with relatively small eigenvalues can be eliminated (set to 0) without essentially disturbing the relevant data.

The singular value decomposition of a square matrix A_n (note that in our case $A^T A$ is square) results in the following decomposition:

$$A_n = U \begin{pmatrix} X_{r,r} & 0 \\ 0 & 0 \end{pmatrix} V^T$$

Where:

- U is the matrix of left singular vectors, $UU^T = U^T U = 1$
- X is a diagonal matrix containing of the roots of the eigenvalues, r is the rank of $A^T A$.
- V is the matrix of right singular vectors, $VV^T = V^T V = 1$

4 The set model

Assume a set D of documents and a set T of terms, and assume a relation $\sim \subseteq \mathcal{T} \times \mathcal{D}$: we write $t \sim d$ to denote that term t describes document d. The tuple $(\mathcal{I}, \mathcal{E}, \sim)$ is called a *formal context* (see [4]). It will be convenient to overload the similarity relation as follows:

$$t \sim D \equiv \forall_{d \in D} [t \sim d] \Longleftrightarrow \{t\} \times \mathcal{D} \subseteq \sim$$

$$Q \sim d \equiv \forall_{t \in Q} [t \sim d] \Longleftrightarrow \mathcal{T} \times \{d\} \subseteq \sim$$

$$Q \sim D \equiv \forall_{t \in Q, d \in D} [t \sim d] \Longleftrightarrow Q \times D \subseteq \sim$$

While the vector model uses vectors as a grouping mechanism, the set model uses *sets* for this purpose. In the set model, intentional objects thus are sets of terms, while extensional objects are sets of documents. Like before, intentional objects represent both queries and document meaning, while extensional objects represent the outcome of a search, or describe the meaning of a term. Similarity on intentional and extensional objects is introduced as set equivalence:

$$x \equiv y \Longleftrightarrow x = y$$

The function **match** is introduced as the left-polar function:

$$\mathbf{match}(D) = \{t \in \mathcal{T} \mid \{t\} \times D \subseteq \sim\}$$

The function **index** corresponds to the right-polar function:

$$\mathbf{index}(Q) = \{d \in \mathcal{D} \mid Q \times \{d\} \subseteq \sim\}$$

Due to the simplicity of the similarity relation for both intentional and extensional objects, the similarity closure assumptions DIRS1 and DIRS2 are trivially satisfied.

Note the special similarity relation implies that the set **Qpc** and **Qc** are isomorph, as is the case with **Dpc** and **Dc**.

Before further focussing on the nature of concepts in this case, we summarize some properties that will be needed (for proofs, see [6]. The polar functions introduce mutuality between documents and terms.

Lemma 5

1. $\mathbf{index}(D) \sim D$
2. $Q \sim \mathbf{index}(Q)$

Both polar functions are non-increasing functions as larger sets have more restrictions for sharing than smaller sets: the larger a set, the less the elements have in common.

Lemma 6

1. $D_1 \subseteq D_2 \implies \mathbf{index}(D_1) \supseteq \mathbf{index}(D_2)$
2. $Q_1 \subseteq Q_2 \implies \mathbf{match}(Q_1) \supseteq \mathbf{match}(Q_2)$

Mutual sharing of meaning between documents and attributes is a special case. First we provide a better characterization of this situation. In the next section, mutual sharing of meaning will be the basis for the introduction of concepts.

Lemma 7

$A \sim D \iff D \subseteq \mathbf{match}(A) \iff A \subseteq \mathbf{index}(D)$

The polar functions can be decomposed in terms of elementary set operations. The following property shows how these operations distribute over the polar functions.

Lemma 8

1. $\mathbf{index}(D_1 \cup D_2) = \mathbf{index}(D_1) \cap \mathbf{index}(D_2)$
2. $\mathbf{match}(A_1 \cup A_2) = \mathbf{match}(A_1) \cap \mathbf{match}(A_2)$

Both document class and term class are extensions of their argument set:

Lemma 9

1. $D \subseteq \mathbf{match}(\mathbf{index}(D))$
2. $A \subseteq \mathbf{index}(\mathbf{match}(A))$

After these properties we return to the sets **Qpc** and **Dpc**. From each starting point, these sets are encountered after one step:

Lemma 10

1. $\mathbf{match}(q) \in \mathbf{Dpc}$
2. $\mathbf{index}(d) \in \mathbf{Qpc}$

Proof: We will only prove the first statement, the second is proven analogously. From lemma 9.2 we conclude $A \subseteq \mathbf{index}(\mathbf{match}(A))$, and thus by lemma 6 we get: $\mathbf{match}(A) \supseteq \mathbf{match}(\mathbf{index}(\mathbf{match}(A)))$.

On the other hand, using lemma 9.1, substituting D by $\mathbf{match}(A)$, we get: $\mathbf{match}(A) \subseteq \mathbf{match}(\mathbf{index}(\mathbf{match}(A)))$.

As a consequence: $\mathbf{match}(\mathbf{index}(\mathbf{match}(A))) = \mathbf{match}(A)$. $\diamond$

5 The dual search engine

In this section we show how dual Information Retrieval systems may be employed in practice. The resulting search engine is called a *dual search engine*. This engine is capable of processing two kinds of request:

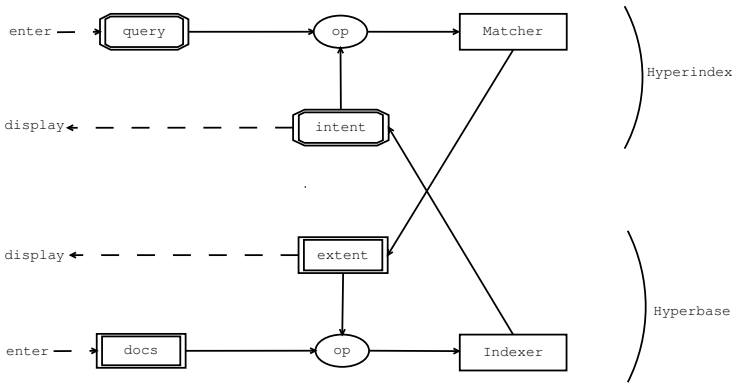

Figure 3: Dual search engine architecture

Q 1. after entering a query q, the engine computes evaluates **match**(q). Hitting the *display*-button will produce the conventional list of documents, ordered by relevancy.

Q 2. after entering a weighted set d of documents, the engine will evaluate **index**(d). Hitting the *display*-button will produce a list of terms, ordered by their weight.

Q 3. The result r of an **match**-operation may be combined with a new query q into q Op r. This combination then is evaluated by the dual search engine, producing **match**$(q$ Op $r)$.

Q 4. The result r of an **index**-operation may be combined with a new weighted set of documents d into d Op r. This combination then is evaluated by the dual search engine, producing **index**$(d$ Op $r)$.

The architecture of the dual search engine is displayed in figure 5. Note that this architecture has a clear resemblance with the stratified architecture ([1]), as this architecture also has a separation in a hyperindex and a hyperbase. The contrast to the stratified architecture is that the primary focus of the stratified architecture is the support of Query by Navigation. The focus of the dual search engine is on exploiting the ability to switch between hyperbase and hyperindex

The dual search engine may be employed in several ways.

Query by example A searcher may offer the dual search engine a document d that is very much alike the kind of documents wanted. The dual search engine determines these documents by evaluating **match**(**index**($\{d\}$)).

The searcher may also use this for relevance feedback, by selecting a relevant subset from the initial query result.

Document contents Offering a document (or a set of documents) to the dual search engine may also be done in order to get an impression of its contents.

Coverage After entering a query q, the dual search engine evaluates **match**($\{q\}$). After inspecting some document d, the searcher might conclude a partial satisfaction of the information need. This can be done by requesting the dual search engine to extract the characterization **index**($\{d\}$) from the original query q, leading to a new query for evaluation.

5.1 A sample session

In this subsection we show how a searcher may perform an incremental search using the dual search engine (see [8] for an overview of the Incremental Model of Information retrieval). The results are simulated using the Bright System ([5]), an generic tool for experimental Information Retrieval. The underlying collection is the Cranfield Collection ([2]).

The searcher starts with expressing the following query:

what are the structural and aeroelastic problems associated with flight of high speed aircraft?

The dual search engine produces the following extent, with the following top documents:

```
3.022597 d12
2.475174 d792
2.303622 d14
2.132593 d746
2.019930 d172
1.787645 d141
1.743521 d364
1.733658 d1089
1.696477 d798
1.695868 d51
1.569504 d36
1.547566 d1263
```

This list shows the document score and the the document number. At the end of subsection a list of some document titles can be found. The searcher is interested in the first document (d12: *Some structural and aerelastic considerations of high speed flight*) and wonders what supplementary information can be obtained from the collection. Document d12 is characterized as follows:

> *aerelastic(1.0000) dominating(0.9286) avenues(0.9286) attacking(0.9286) acrothermoelasticity(0.9286) meet(0.8397) matter(0.8397) interrelation(0.8397) inputs(0.8397) engineers(0.8397) alleviating(0.8397) demands(0.7878) art(0.7878) tools(0.6989) origin(0.6791) aeronautical(0.6212) aeroelastic(0.6173) summarized(0.6101) structural(0.5820) summary(0.5815) largely(0.5732) another(0.5212) fundamental(0.5109) failure(0.5014) factors(0.4979) considerations(0.4943) concerned(0.4766) subject(0.4728) modes(0.4728) finally(0.4590) structures(0.4526) suggested(0.4406) respect(0.4406) combined(0.4406) state(0.4197) research(0.4173) aircraft(0.4116) discussion(0.4016) structure(0.3955) analytical(0.3915) thermal(0.3851) upon(0.3769) flight(0.3702) available(0.3669) load(0.3622) speed(0.3457) their(0.3435) design(0.3128) high(0.2984) problems(0.2898) methods(0.2855) well(0.2725) into(0.2680) some(0.2591) under(0.2527) transfer(0.2488) discussed(0.2463) heat(0.2188) one(0.2163) experimental(0.1900) presented(0.1856) these(0.1658) layer(0.1612) boundary(0.1429) as(0.1107) from(0.1042) with(0.0484) are(0.0457) is(0.0290) in(0.0166) to(0.0161) and(0.0113) a(0.0099) the(0.0013) of(0.0010)*

This characterization is used to build a new query, which is offered to the dual search engine (see figure 5). We assume that the operator `op` combines the original query q and the characterization c of d12 by evaluating for each term t the expression $q(t)(1 - c(t))$. The resulting query is:

> *of(0.9990) the(0.9987) and(0.9887) with(0.9516) high(0.7016) speed(0.6543) are(0.9543) problems(0.7102) structural(0.4180) flight(0.6298) aircraft(0.5884) aeroelastic(0.3827) associated(1.0000) what(1.0000)*

This leads to:

```
1.862300 q1 d792
1.699372 q1 d12
1.485877 q1 d172
1.362866 q1 d14
1.316785 q1 d364
1.308882 q1 d798
1.291517 q1 d1089
1.273609 q1 d36
1.263685 q1 d746
1.082772 q1 d1268
1.034336 q1 d141
1.025378 q1 d810
```

```
0.990003 q1 d712
```

The searcher may continue to take the most relevant document, and pursue with the residual information need. This will give the searcher a summary impression of the topic addressed by the initial query.

d792 : some low speed problems of high speed aircraft

d12 : some structural and aerelastic considerations of high speed flight

d14 : piston theory - a new aerodynamic tool for the aeroelastician

d746 : aeroelastic problems in connection with high speed flight

d172 : some aerodynamic considerations of nozzle afterbody combination

d51 : theory of aircraft structural models subjected to aerodynamic heating and external loads

6 Conclusions

This paper has shown how two different Information Retrieval models have an abstract foundation in common. Both models look for special elements that capture 'meaning' by using the same methodology, resulting in two different, but well known Information Retrieval approaches. The vector model leads to Singular Value Decomposition while the set model leads to Formal Concept Analysis. Although both models derive a set of concepts, the applicability of these methods depends on the objected goal. Singular Value Decomposition yields a limited number of concepts which globally describe the document-term relationship. Using this approach it is possible to eliminate noise and focus on the most relevant information. Formal Concepts Analysis however, yields a large number of highly detailed *structured* concepts. Here, every single conceptual detail is visible. In situations were feedback and navigation is possible or situations where discovering irregularities is needed instead of eliminating them, this method is preferable.

References

[1] P.D. Bruza and Th. P. van der Weide. Stratified hypermedia structures for information disclosure. *The Computer Journal*, 35(3):208–220, 1992.

[2] C.W. Cleverdon, J. Mills, and M. Keen. Factors determining the performance of indexing systems, vol i design, vol ii test results. *ASLIB Cranfield Research Project*, 1966.

[3] Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

[4] B. Ganter and R. Wille. *Formale Begriffsanalyse, Mathematische Grundlagen*. Springer-Verlag Berline, 1996.

[5] F.A. Grootjen. *A Pragmatic Approach to the Conceptualisation of Language*. PhD thesis, University of Nijmegen, Nijmegen, The Netherlands, to appear, 2003.

[6] F.A. Grootjen and Th. P. van der Weide. Conceptual relevance feedback. In *Proceedings of the Second International Workshop on Natural Language Processing and Knowledge Engineering (NLPKE 2002)*, Tunis, October 2002.

[7] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill New York, NY, 1983.

[8] Th. P. van der Weide, T. W. C. Huibers, and P. van Bommel. The incremental searcher satisfaction model for information retrieval. *The Computer Journal*, 41(5):311–318, 1998.