# Dualistic Ontologies

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

**Abstract**

To effectively use and share knowledge among AI systems, a formal specification of the representation of their shared domain of discourse – called an ontology – is indispensable. In this paper we demonstrate the role of dualistic ontologies in human activities such as searching, in-depth exploration and browsing.

We start with the introduction of a formal definition of dualistic ontologies, based on a dual view on the universe of discourse.

Then we relate this formal definition to three different (well known) kinds of ontologies, based on the vector model, on formal concept analysis and on fuzzy logic respectively. The vector model leads to concepts derived by latent semantic indexing using the singular value decomposition. Both the set model as the fuzzy set model lead to Formal Concept Analysis, in which the fuzzy set model is equipped with a parameter that controls the fine-grainedness of the resulting concepts. We discuss the relation between the resulting systems of concepts.

Finally, we demonstrate the use of this theory by introducing the dual search engine. We show how this search engine can be employed to support the human activities addressed above.

**Keywords:** formal concepts, latent semantics, dual search engine, ontology, knowledge discovery

## 1 Introduction

Sharing knowledge is a real challenge when we can not simply rely on some common underlying body of conceptualization and representation. An ontology is a knowledge structure that describes concepts and their relations. Ontologies play an important role in application areas like Artificial Intelligence and Information Retrieval. In this paper, we will be focusing on communication aspects, assuming an information providing and an information requesting agent. In the context of Information Retrieval, the information requesting agent is a human being, referred to as a searcher. The information providing agent will be referred to as a search engine in this paper. This does not mean that the application of dualistic ontologies is restricted to human agents. The formal description of this theory makes it generally applicable to agents.

In order to express an information need, a searcher could show an example of a relevant information object (or, document). But a searcher may also be capable to formulate the information need by a combination of search terms. In practice, searchers find it difficult to provide a proper query formulation, but have no problems in recognizing a document as being relevant. Ontologies usually provide a knowledge structure on the level of search terms, while searchers would benefit from a conceptualization materialized by information objects.

From a general point of view, 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

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

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 [Salton and McGill, 1983]), 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. In general, we will use the term *dualistic system*. To demonstrate the look and feel of a realistic system, we present DUALITY (see figure 2).

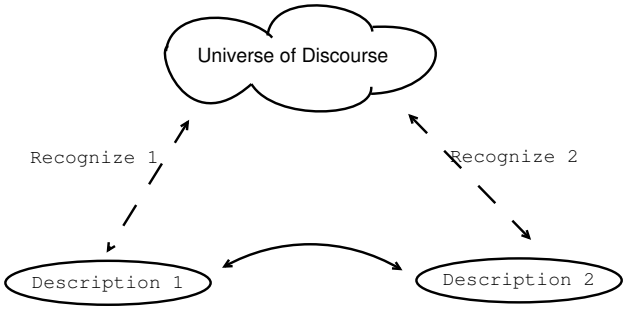


Figure 1: Different models of real world

An example of an intentional object is a query while an example of a extensional object is the outcome of search. We will use the terms *query* and *search result* as alternative terms for intentional and extensional objects respectively. 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 dualistic system may be seen as a mutual semantics assigning system.


Figure 2: Initial query

In this paper, we focus on dualistic systems. In section 2 we show how different views on the real world can be combined to recognize concepts as semantical fixed points. We provide a formal definition and discuss some properties. 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 ([Deerwester et al., 1990]). This approach is based on the singular value decomposition, usually applied when noise removal is an issue. In section 4 we study concepts in the set model, and find the relation with formal concept analysis ([Ganter and Wille, 1996]). This approach is very fine-grained, and can be used to find a needle in the haystack. In section 5 the fuzzy set model and fuzzy logic are the basis the formal concept approach is generalized, to cover some degree of uncertainty. This degree is a parameter steering the trade-off between granularity and (computational) complexity. In section 6 we apply this general theory by introducing the dual search engine DUALITY, and show the validity of the approach taken in this paper. Finally, in section 7 we present some conclusions.

# 2 Dualistic systems

Consider a dualistic 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). The motivation to introduce similarity relations is to be able to handle for example equivalences that originate from syntactic variety in queries.

## 2.1 The model

Intentional and extensional objects are assigned a meaning in terms of each other. The function **match** : $\mathcal{I} \rightarrow \mathcal{E}$ interprets intentional objects in terms of extensional objects, the function **index** : $\mathcal{E} \rightarrow \mathcal{I}$ does it the opposite way (see figure 3). We assume the assignment of meaning to be closed under similarity:

DS 1. *Similar queries yield a similar query result:*
$q_1 \equiv_i q_2 \implies \mathbf{match}(q_1) \equiv_e \mathbf{match}(q_2)$

DS 2. *Similar collections have a similar description:*
$d_1 \equiv_e d_2 \implies \mathbf{index}(d_1) \equiv_i \mathbf{index}(d_2)$

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

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

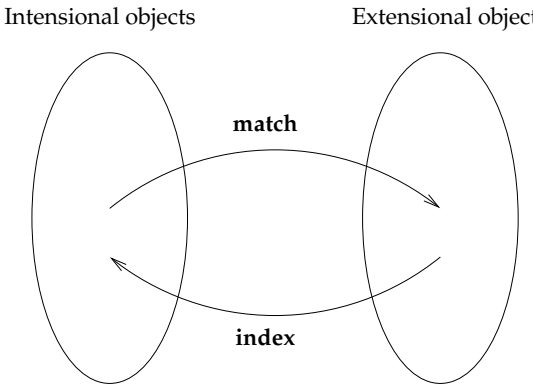


Figure 3: A dualistic system

We do not make any special assumptions on the relation between the functions **match** and **index** governing their interaction.

## 2.2 Proto-concepts

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 *mirroring*, i.e. the subsequent application of **index** and **match** in either order. We introduce proto-concepts as objects that have a similar mirror:

$$\mathbf{Qpc} = \{q \mid \mathbf{index}(\mathbf{match}(q)) \equiv_i q\} \quad \textit{query proto-concepts}$$
$$\mathbf{Dpc} = \{d \mid \mathbf{match}(\mathbf{index}(d)) \equiv_e d\} \quad \textit{search result proto-concepts}$$

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

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

## 2.3 Abstracting from variation

The similarity relations on intentional and extensional objects may be seen as a relation dealing with the variation that is offered by the underlying description mechanism. In this subsection we abstract from these variations.

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:

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

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, $m(i(dc)) = dc$. $\diamond$

## 2.4 Concepts

As we are looking in a dualistic system for sharing of meaning, we concentrate 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**
$\mathcal{C} = \{(qc, m(qc)) \mid qc \in \mathbf{Qc}\} = \{(i(dc), dc) \mid dc \in \mathbf{Dc}\}$

Concepts consist of an intentional and an extensional part. Concepts may be ordered by the knowledge they reflect, as represented both by their intention and extension. We will not further elaborate on this ordering of concepts, as such an ordering will become meaningful only is some further properties are assumed on the interaction between the functions **index** and **match**.

## 2.5 Descriptor approximation

An interesting operator is the approximation of intentional or extensional objects. Let $d$ be some extensional object. Then $d$ is described by intentional object **index**$(d)$. It is possible, however, that no intentional object can produce this meaning, or: $\forall_q$ [**match**$(q) \neq d$]. The question then is what descriptors are good approximations of the contents of this query result. We call an intentional object $q$ an approximation of query result $d$ if the materialization **match**$(q)$ of this descriptor has the same (intentional) meaning as query result $d$.

**Definition 2** *The set* **Approx**$(d)$ *of approximations of extensional object* $d$ *is defined by:*

$$\mathbf{Approx}_e(d) = \{q \mid \mathbf{index}(\mathbf{match}(q)) \equiv_i \mathbf{index}(d)\}$$

Analogously we can introduce the approximations of a descriptor $q$:

**Definition 3** *The set* **Approx**$(q)$ *of approximations of extensional object* $d$ *is defined by:*

$$\mathbf{Approx}_i(q) = \{d \mid \mathbf{match}(\mathbf{index}(d)) \equiv_e \mathbf{match}(q)\}$$

Approximations are an important feature in the dualistic system. If a searcher would offer a query result as a typical specimen of the information need. Approximations of this query result can be used as a starting point during the process of Query by Navigation, supporting the searcher in finding a proper formulation of the information need.

**Lemma 3**
If query result $d$ and query $q$ are approximations of each other, then $([\mathbf{index}(d)]_i, [\mathbf{match}(q)]_e)$ is a concept.

**Proof:** Suppose query result $d$ and query $q$ are approximations of each other, or:
**index**$(\mathbf{match}(q)) \equiv_i$ **index**$(d)$ and **match**$(\mathbf{index}(d)) \equiv_e$ **match**$(q)$. Then **match**$(q) \in$ **Dpc** and **index**$(d) \in$ **Qpc** are easily verified. The result then follows from theorem 2. ◇

## 2.6 Application

The following sections show how different IR models, the vector model, the set model and the fuzzy 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. The fuzzy model provides the opportunity to balance between granularity and cost of computation.

# 3 The vector model

Assume a set D of documents and a set T of terms, and an *aboutness* 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 $\lambda$.

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

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

These functions satisfy the similarity closure assumptions:

**Lemma 4**

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:

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

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

2. Analogously.

$\diamond$

A value $\lambda$ such that $Aq = \lambda d$ and $A^T d = \lambda q$ for non-zero vectors $q$ and $d$, is called a singular value of matrix $A$. The vectors $d$ and $q$ are called left-singular and right-singular values for $\lambda$, respectively. Invariance under subsequent application of **match** and **index** leads to the eigenvectors of $A^T A$ and $AA^T$ respectively:

**Lemma 5**

1. $\textbf{Qpc} = \left\{ q \mid \exists_{\lambda>0} \left[ A^T A q = \lambda q \right] \right\}$
2. $\textbf{Dpc} = \left\{ d \mid \exists_{\lambda>0} \left[ A A^T d = \lambda d \right] \right\}$

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 ([Deerwester et al., 1990], [Berry et al., 1995]), 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, $VV^T = V^T V = 1$

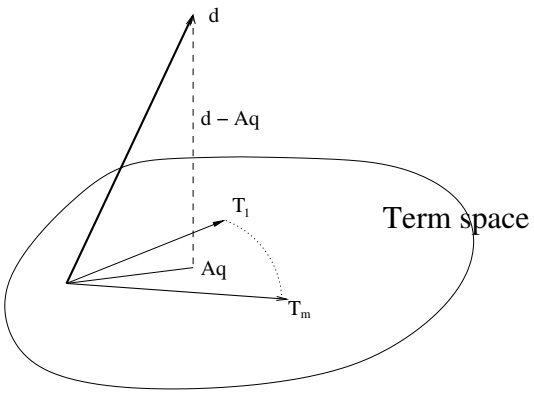


Figure 4: Projecting an extensional object onto term space

The set **Approx**$(d)$ of approximations of extensional object $d$ is described by:

$$\textbf{Approx}(d) = \left\{ q \mid A^T Aq = A^T d \right\}$$

In terms of linear algebra, the vector $q$ is the best solution of the equation $Aq \approx d$. Being the best solution means that (see figure 4) $d - Aq$ is orthogonal on the image space of $A$ (the term space), i.e., $A^T(d - Aq) = 0$. This optimal query $q$ thus is the solution of the equation $A^T Aq = A^T d$. As a consequence, the set **Approx**$(d)$ consists of the projection from $d$ onto term space.

# 4 The set model

## 4.1 Formal Concept Analysis

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 $d$. For example, $t \sim d \iff A_{d,t} > 0$. The tuple $(\mathcal{I}, \mathcal{E}, \sim)$ is called a *formal context* (see [Ganter and Wille, 1996]). It will be convenient to overload the similarity relation as follows:

$$t \sim D \equiv \forall_{d \in D} \left[ t \sim d \right]$$

$$Q \sim d \equiv \forall_{t \in Q} \left[ t \sim d \right]$$

$$Q \sim D \equiv \forall_{t \in Q, d \in D} \left[ t \sim d \right]$$

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:

$$x \equiv y \iff x = y$$

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

$$\textbf{index}(D) = \{ t \in \mathcal{T} \mid t \sim D \}$$

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

$$\textbf{match}(Q) = \{ d \in \mathcal{D} \mid Q \sim d \}$$

Due to the simplicity of the similarity relation for both intentional and extensional objects, the similarity closure assumptions DS1 and DS2 are trivially satisfied. Notice that **index** and **match** form a *Galois connection*, a pair of reverse order functions between two partially ordered sets.

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

Before further focusing on the nature of concepts in this case, we summarize some properties that will be needed (for proofs, see [Grootjen and van der Weide, 2002]. The polar functions introduce mutuality between documents and terms.

**Lemma 6**
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 7**
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 8**
$A \sim D \iff D \subseteq \mathbf{match}(A) \iff A \subseteq \mathbf{index}(D)$

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 9**
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 10**
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 11**
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 10.2 we conclude $A \subseteq \mathbf{index}(\mathbf{match}(A))$, and thus by lemma 7 we get: $\mathbf{match}(A) \supseteq \mathbf{match}(\mathbf{index}(\mathbf{match}(A)))$.
On the other hand, using lemma 10.1, substituting $D$ by $\mathbf{match}(A)$, we get: $\mathbf{match}(A) \subseteq \mathbf{match}(\mathbf{index}(\mathbf{match}(A)))$.
As a consequence: $\mathbf{match}(\mathbf{index}(\mathbf{match}(A))) = \mathbf{match}(A)$. $\diamond$

The set **Approx**$(d)$ of approximations of extensional object $d$ has been introduced as:

$$\textbf{Approx}(d) = \{q \mid \textbf{index}(\textbf{match}(q)) = \textbf{index}(d)\}$$

Let **index**(**match**$(q)$) = **index**$(d)$, then also **match**(**index**(**match**$(q)$)) = **match**$(q)$ = **match**(**index**$(d)$) So **Approx**$(d)$ is the set containing the intentional object that approximates to the concept determined by intentional object **index**$(d)$.

# 5 The fuzzy set model

In this section we consider a fuzzy model for information retrieval based on the construction of a fuzzy formal context. The basis for interpreting Information Retrieval in terms of many-valued logics is the introduction of a fuzzy implication. In [van Rijsbergen, 1986] a non classical logic is proposed for information retrieval (see also [Crestani and van Rijsbergen, 1995]). We will use $\rightarrow_f$ as a generic symbol for fuzzy implementation. Fuzzy implementation is seen as a function with signature $[0,1] \times [0,1] \rightarrow [0,1]$. $a \rightarrow_f b$ indicates how certain we are over the validity of the implication given how certain we are over its arguments ($a$ and $b$ respectively). Fuzzy logic provides a logics of vagueness ([Hjek et al., 1996]). Fuzzy logics may be based on a conjunction operator $t(x,y)$ and an implication operator $i(x,y)$. They form an adjoint couple if $z \leq i(x,y) \Longleftrightarrow t(x,y) \leq y$. There are three main variants:

1. Łukasiewicz' logic
   $x \& y = \max(0, x+y-1)$
   $x \rightarrow_{\text{Ł}} y = \min(1, 1-x+y)$
2. Gödel's logic
   $x \wedge y = \min(x,y)$
   $x \rightarrow_G y = (x \leq y \rightarrow 1;\ y)$
3. product logic
   $x \odot y = xy$
   $x \rightarrow_P y = (x \leq y \rightarrow 1;\ y/x)$

In these logic's, the constants *true* and *false* correspond to $1$ and $0$ respectively. As in the set model, we assume a set D of documents and a set T of terms. The aboutness relation is seen as a fuzzy relation, i.e., for each document $d$ and term $t$ the $A_{d,t}$ describes the degree in which document $d$ is supposed to be about term $t$. This fuzzy relation may be identified with the aboutness matrix from the vector model.

In our fuzzy model for Information Retrieval, an intentional object is a fuzzy set over terms T, while an extensional object is a fuzzy set of documents D. Intentional and extensional objects are similar when they are equal. The similarity closure assumptions thus obviously are satisfied.

Indexing a set of documents can be seen as finding for each term the degree in which this term is implied by the (fuzzy) collection being indexed. The result of indexing is an intentional object, or a fuzzy set of terms. This may be expressed as:

$$\textbf{index}(D) = \lambda_{t \in \mathcal{T}} [\wedge_d [D(d) \rightarrow_f A_{d,t}]]$$

During matching it is determined to what degree documents are implied by the query. The result is an extensional object, or a fuzzy document set.

$$\textbf{match}(T) = \lambda_{d \in \mathcal{D}} [\wedge_t [T(t) \rightarrow_f A_{d,t}]]$$

Small certainty may originate from noise. A threshold $\theta$ is introduced for the recognition of noise. Scoring above this threshold means acceptance, otherwise the statement is believed to be invalid. In [Elloumi et al., 2004] this is effectuated by:

$$\textbf{match}_\theta(T) = \lambda_{d \in \mathcal{D}} [\wedge_t [T(t) \rightarrow_f A_{d,t} \geq \theta]]$$

where the outcome of the comparison operator is to be interpreted using the identities: *true*=1 and *false*=0. As a consequence, the result $\mathbf{match}_\theta(T)$ is a set of documents.

Using Gödel's logic, the index operator is further elaborated as follows:

$$\begin{aligned}\mathbf{index}(D) &= \lambda_{t\in\mathcal{T}}\left[\wedge_d\left[D(d) \rightarrow_G A_{d,t}\right]\right] \\ &= \lambda_{t\in\mathcal{T}}\left[\min_d\left(D(d) \leq A_{d,t} \rightarrow 1;\ A_{d,t}\right)\right] \\ &= \text{(in case } D \text{ is a crisp set) } \lambda_t\left[\min_{d\in D} A_{d,t}\right]\end{aligned}$$

So it seems reasonable to restrict extensional objects to sets of documents. For the match operator we get:

$$\begin{aligned}\mathbf{match}_\theta(T) &= \lambda_{d\in\mathcal{D}}\left[\wedge_t\left[T(t) \rightarrow_G A_{d,t} \geq \theta\right]\right] \\ &= \lambda_{d\in\mathcal{D}}\left[\min_t\left(T(t) \rightarrow_G A_{d,t} \geq \theta \rightarrow 1;\ 0\right)\right] \\ &= \lambda_{d\in\mathcal{D}}\left[\min_t\left(A_{d,t} \geq \min(T(t),\theta) \rightarrow 1;\ 0\right)\right]\end{aligned}$$

Thus $\mathbf{match}_\theta$ corresponds to the (crisp) set $\{d \mid A_{d,t} < T(t) \implies A_{d,t} < \theta\}$. So documents should satisfy sufficient information on each term, except if the term is noise in that document. Note that a drawback of this approach is that documents with noisy terms only, will be retrieved.

In [Elloumi et al., 2004] a hybrid approach for matching is taken. The conjunction operator is defined as in Gödel's logic, while the implication is substantiated according to Łukasiewicz' logic. The matching operator then is elaborated as follows:

$$\begin{aligned}\mathbf{match}_\theta(T) &= \lambda_{d\in\mathcal{D}}\left[\wedge_t\left[T(t) \rightarrow_{\text{Ł}} A_{d,t} \geq \theta\right]\right] \\ &= \lambda_{d\in\mathcal{D}}\left[\min_t(\min(1, 1 - T(t) + A_{d,t}) \geq \theta)\right]\end{aligned}$$

Thus in this case, $\mathbf{match}_\theta$ corresponds to the (crisp) set

$$\{d \mid \forall_t\left[A_{d,t} < T(t) \implies (T(t) - A_{d,t}) \leq 1 - \theta\right]\}$$

So if a document would fail the requested supply of some term, then the term shortage for this document should be limited by $1 - \theta$.

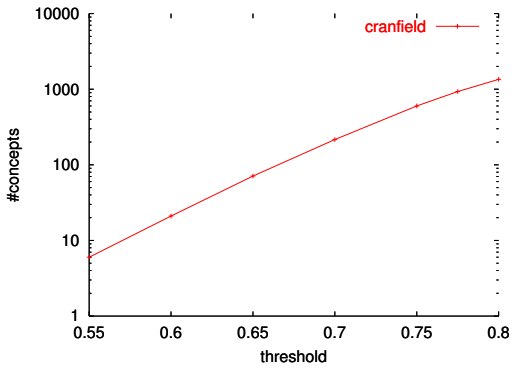


Figure 5: Granularity of concepts

First we note the special case $\theta = 1$. In that case the limit for term shortage is so strict that uniform term supply is requested:

$$\mathbf{match}_1 = \{d \mid \forall_t\left[A_{d,t} \geq T(t)\right]\}$$

**Lemma 12**
For $\theta = 1$ the fuzzy set model is equivalent to the set model

**Proof:** For the formal concept we have: $t \sim d \iff A_{d,t} > 0$. If we assume $A_{d,t} \in \{0, 1\}$, then $A_{d,t} \geq T(t)$ is equivalent with $A_{d,t} = 1$, and therefore with $t \sim d$. ◇

For the case $\theta = 0$ all documents will pass the membership test to match the query. The resulting concept lattice thus will contain only 1 concept. The noise threshold may be used to take a position in between. For example, if small variation is not likely to be a consequence of noise, then $\theta$ could be chosen near to 1. If a limited number of concepts is required, then a smaller

value should be taken for the noise threshold. In figure 5 we see how for an example document collection the number of concepts depends on the noise threshold. Note that the figure suggests an almost linear dependency on a logarithmic scale.

The set **Approx**$(D)$ of approximations of extensional object $D$ has been introduced as:

$$\textbf{Approx}(D) = \{Q \mid \textbf{index}(\textbf{match}(Q)) = \textbf{index}(D)\}$$

Let $Q$ be some query, then the associated query result reflects the degree in which the documents support the query. If this query result :

$$\textbf{index}(\textbf{match}(Q)) \quad = \quad \lambda_t\,[\wedge_d\,[\wedge_t\,[Q(t) \rightarrow_f A_{d,t}] \rightarrow_f A_{d,t}]]$$

The condition **index**$(\textbf{match}(Q)) = \textbf{index}(D)$ thus is formulated as:

$$\wedge_d\,[\wedge_t\,[Q(t) \rightarrow_f A_{d,t}] \rightarrow_f A_{d,t}] = \wedge_d\,[D(d) \rightarrow_f A_{d,t}]$$

This expression may be further simplified using the rules of the underlying logic.

# 6 The dual search engine

In this section we show how dualistic systems may be employed in practice to support information retrieval. We will describe a search engine based on dualistic system technology. The resulting search engine is called a *dual search engine*. This engine is capable of processing two kinds of request:

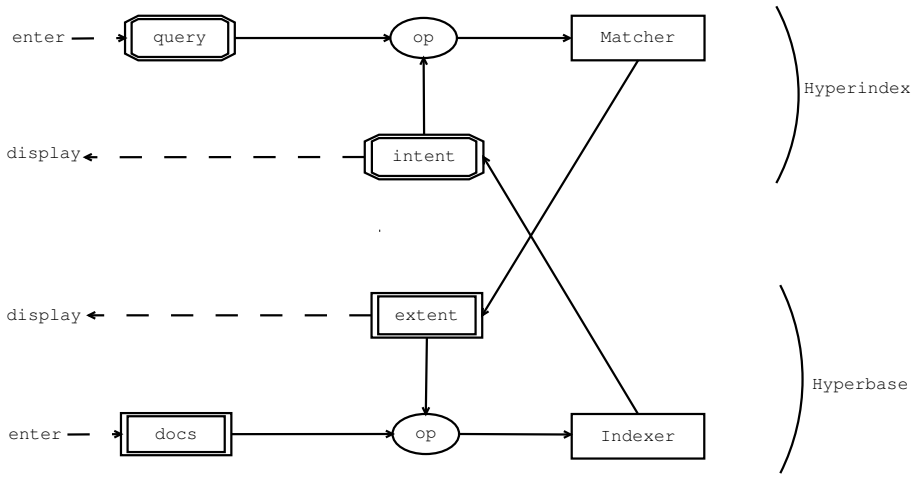


Figure 6: Dual search engine architecture

Q 1. after entering a query $q$, the engine evaluates the search result **match**$(q)$. This will produce the conventional list of documents, ordered by relevancy.

Q 2. after entering a weighted set $d$ of documents, the engine will produce a common desciption by evaluating **index**$(d)$. This will produce a list of terms, ordered by their weight.

Furthermore, the dual search engine makes it possible to further elaborate on the results obtained:

R 1. 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 $r)$.

R 2. 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 6. Note that this architecture has a clear resemblance with the stratified architecture ([Bruza and van der Weide, 1992]), 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.

## 6.1 A sample session

We now demonstrate how a searcher may perform a search using the dual search engine DUALITY. The results are calculated by the BRIGHT system [Grootjen and van der Weide, 2004], an generic tool for experimental Information Retrieval research. The underlying collection is the Cranfield Collection [Cleverdon, 1967].

### 6.1.1 Query by Example

Suppose a searcher wants to know about problems associated with high speed aircraft. As an initial attempt the query 'high speed aircraft' is entered into the search engine (figure 2) which produces a classical ranked list of documents (see figure 7). Notice that the output of the search engine has two different panels, one called the *Intentional View* containing the (weighted) entered query keywords, and one called the *Extensional View* which shows the set of ranked documents.

After inspecting the document titles and excerpts of the top 10 ranked documents, the searcher assesses the $4^{th}$ document (d12) to be relevant, and selects the document's checkbox. Since the selected document covers the desired topic area, the user decides to use 'Query by Example'. This is done in two steps: first the corresponding Intentional object is created by pressing the index button (the button marked with the symbol $>$). This will update the Intentional View panel and shows a new list of weighted terms. The second step is to update the Extensional View panel using these new terms. This is done by pressing the match button (the button marked with the symbol $<$). The result, depicted in figure 8, shows the new list of documents. Since this query is part of the Cranfield Collection, and therefore accompanied by relevance judgements, we can calculate the performance of the retrieval result. Not surprisingly, the performance improved drastically (see figure 9). Note that, in contrast to the example, more than one document can be selected when performing Query by Example.

### 6.1.2 Coverage

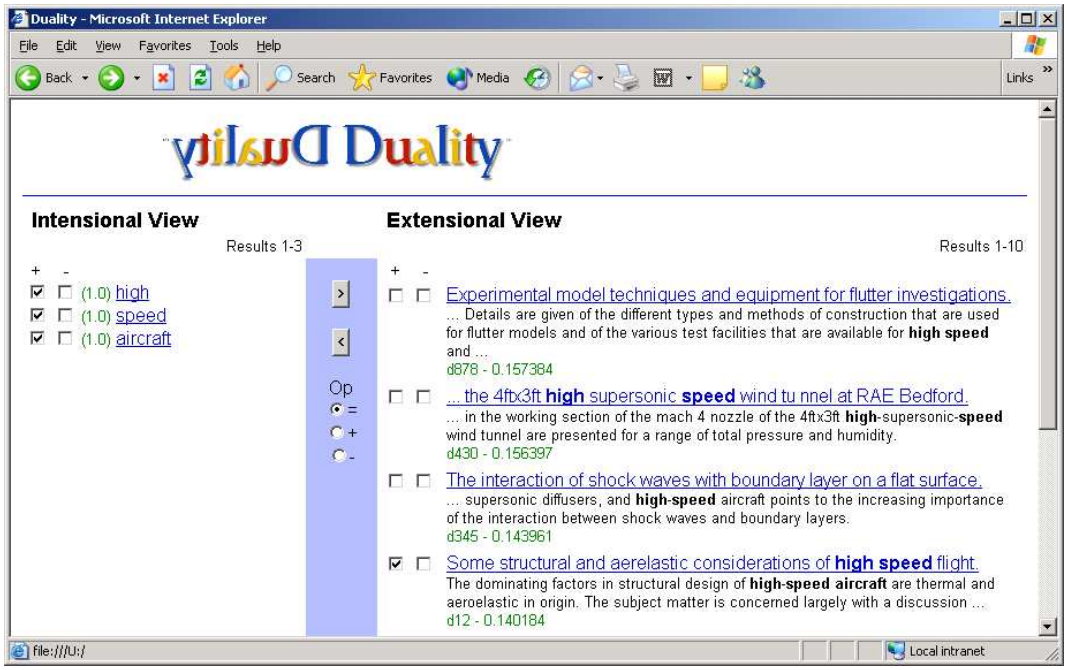


Figure 7: Ranked list

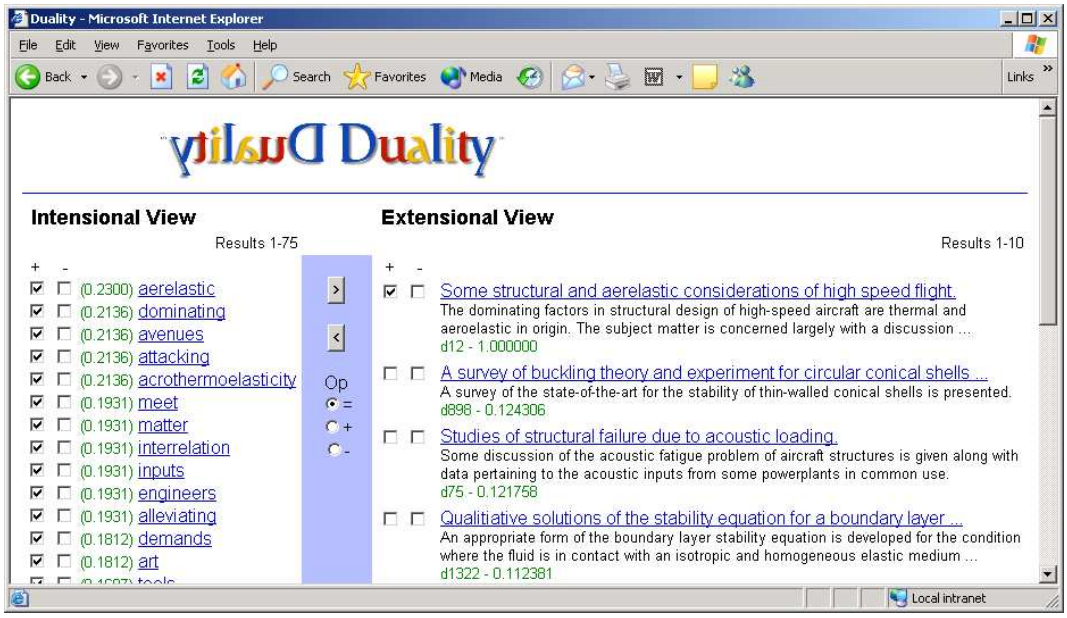


Figure 8: Query by example

In the previous example a document selection is used to create a new set of terminals, which is directly used as input to a subsequent match call; the original query terminals are *replaced* by the new ones. DUALITY offers the possibility to do more than that: in some cases we don't want the old terminal set to be replaced (Op =). So two simple operators are implemented allowing the user to add (Op +) or subtract (Op –) the index or match result to the current set. Another possibility is to modify the resulting terminal list before invoking the index function: the searcher might add or penalize terminals.

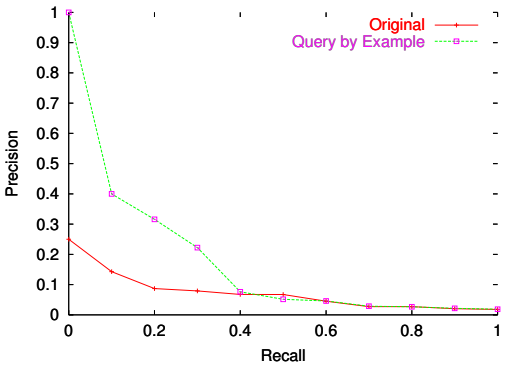


Figure 9: Retrieval performance

These operators can be used to disambiguate the original query: for example a query about operating systems returns pages about Linux which we want to ignore. Or when our information need is already partly covered, and we are looking for additional (new, residual) information.

Note that after selecting a relevant document, and invoking index with Op + followed by a match is equivalent to applying the standard Rocchio technique for relevance feedback (see for example [Baeza-Yates and Ribeiro-Neto, 1999]).

### 6.1.3 Crossing the boundary

We will conclude this example section by showing the benefits of combining two dualistic ontologies: we will show how the vectorspace model and the fuzzy model can be combined to yield one powerful search tool.
Suppose a searcher as an information need described as query 173 of the Cranfield collection:

> *References on Lyapunov's method on the stability of linear differential equations with periodic coefficients.*

From the relevance judgement we know that there are only 3 relevant documents in the collection. Assume that during (vectorspace based) browsing the searcher finds the relevant document d532. Instead of using Query by Example or Relevance feedback, the searcher decides to switch to the fuzzy concept model[1]. Using the approximation function DUALITY presents the fuzzy concept containing d532 with its fuzzy terminal set. One of the extra features of the conceptual model is that the concepts are *ordered*. This enables the user to navigate to related concepts (Query by Navigation). As shown in figure 10 the searcher can beam down to the bottom concept of the lattice or beam up to 3 different superconcepts. After inspecting the terminals presented by the concepts, the searcher decides to beam up to the concept containing both d532 and 367 (which happen to be both relevant). The process of beaming up increases the extension, and reduces the intention. This is clear when we look at the result (figure 11): the list of terminals is shorter since it covers two documents. In this new fuzzy concept the searcher can beam up to the top concept of the lattice, beam down back to the concept containing only d532, or beam down to the concept of document d367. If the searcher switches back to the vectorspace model, doing a Query by Example of the two found documents, he will get a ranked document list as extension, with the three relevant documents ranked 1, 2 and 3.

[1]The used fuzzy concept lattices is generated with threshold 0.775 and contains 993 concepts

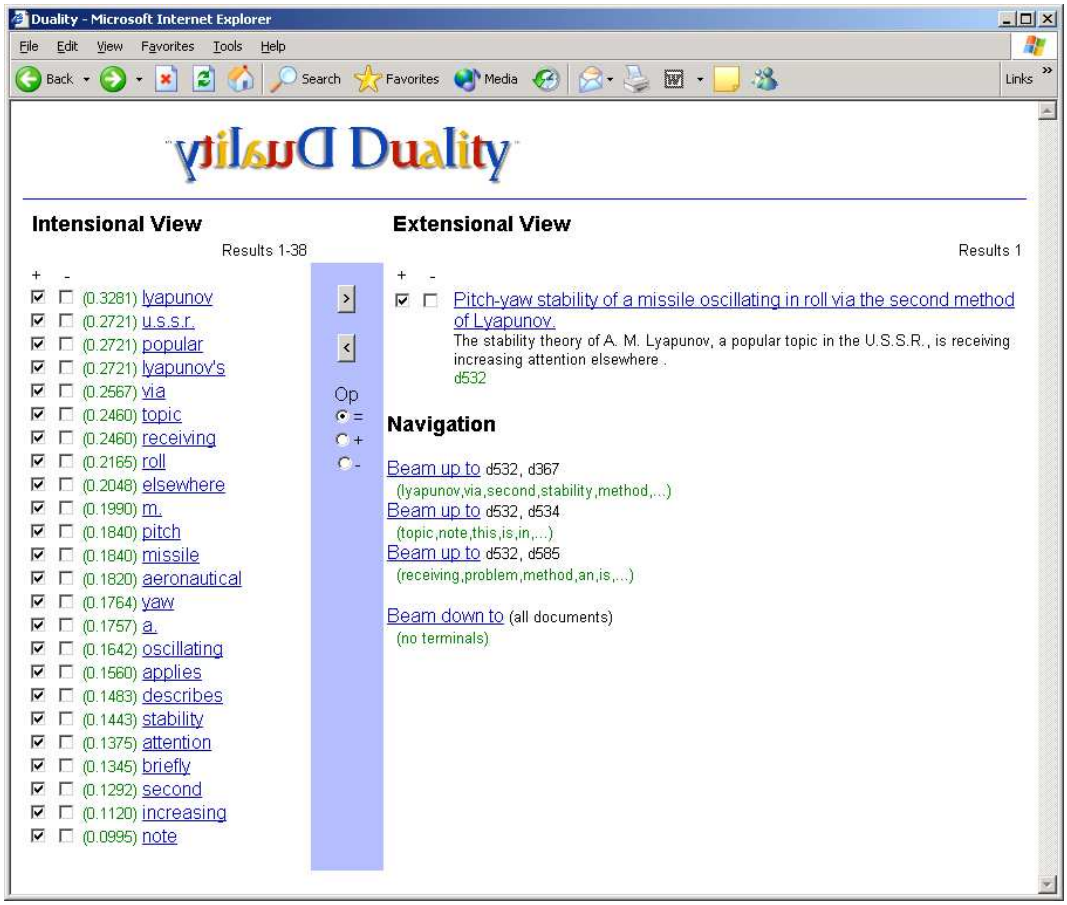


Figure 10: Switching to fuzzy concept model

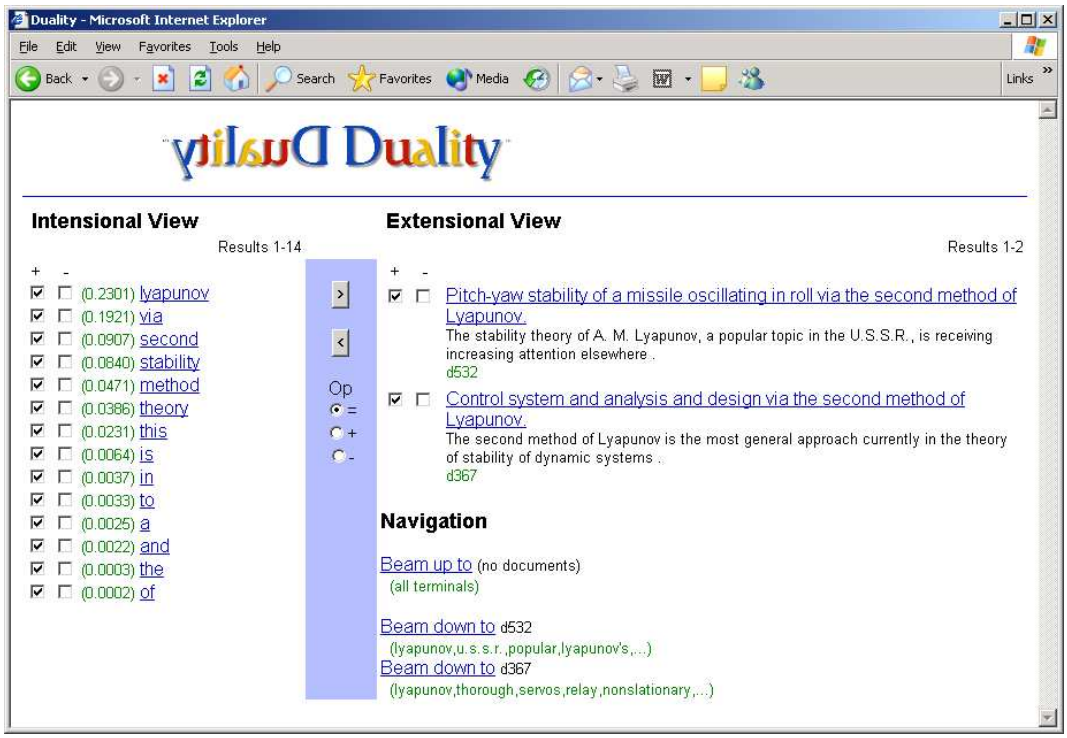


Figure 11: Beaming up

# 7 Conclusions

In this paper we have introduced the concept of dualistic ontologies, discussed properties of such ontologies, and related them to some well-known retrieval models. In order to demonstrate their usefulness, dual search engines were introduced and illstrated by a sample session of the dual search engine DUALITY.

Further investigations might consider the question of how the combination of several dualistic ontologies may offer further opportunities for searchers to improve their retrieval efefctiveness.

# References

[Baeza-Yates and Ribeiro-Neto, 1999] Baeza-Yates, R. and Ribeiro-Neto, B. (1999). *Modern Information Retrieval*. Addison-Wesley.

[Berry et al., 1995] Berry, M. W., Dumais, S., and O'Brien, G. (1995). Using linear algebra for intelligent information retrieval. *SIAM Review*, 37(4):573–595.

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

[Cleverdon, 1967] Cleverdon, C. (1967). The cranfield tests on index language devices. *Aslib Proceedings*, (19):173–194.

[Crestani and van Rijsbergen, 1995] Crestani, F. and van Rijsbergen, C. J. (1995). Information retrieval by logical imaging. *Journal of Documentation*, 51:3–17.

[Deerwester et al., 1990] Deerwester, S. C., Dumais, S. T., Landauer, T. K., Furnas, G. W., and Harshman, R. A. (1990). Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407.

[Elloumi et al., 2004] Elloumi, S., J., J., A., H., A., J., and I., N. (2004). A multi-level conceptual data reduction approach based in the lukasiewicz implication. *Information Sciences*, 163(4):253–262.

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

[Grootjen and van der Weide, 2002] Grootjen, F. and van der Weide, Th. P. (2002). Conceptual relevance feedback. In *Proceedings of the 2002 IEEE International Conference on Systems, Man and Cybernetics, (NLPKE 2002)*, Tunis.

[Grootjen and van der Weide, 2004] Grootjen, F. and van der Weide, Th. P. (2004). The Bright side of information retrieval. Technical Report NIII, Radboud University of Nijmegen.

[Hjek et al., 1996] Hjek, P., Godo, L., and Esteva, F. (1996). A complete many-valued logic with product-conjunction. *Archive for mathematical logic*, 35:191–208.

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

[van Rijsbergen, 1986] van Rijsbergen, C. J. (1986). A non-classical logic for information retrieval. *The Computer Journal*, 29(6):481–485.