A pragmatic approach to the conceptualization of language

F.A. Grootjen

October 6, 2002

Contents

Chapter 1

Introduction

"Good morning!" said Bilbo, and he meant it. The sun was shining, and the grass was very green. But Gandalf looked at him from under long bushy eyebrows that stuck out further that the brim of his shady hat. "What do you mean?" he said. "Do you wish me a good morning, or mean that it is a good morning whether I want it or not; or that you feel good this morning; or that it is morning to be good on?"

J.R. Tolkien - The Hobbit

1.1 Information

Most introductions in recent Information Retrieval literature start with the observation that the digital and network revolution of the last decade lead to an enormous amount of information available at the users' fingertips [1]. The only problem seems to be *finding* the desired information.

Although this statement seems plausible, it is vague and misleading. It suggests that *information* is available. A computer (or the network) is capable of storing *data*, not information. After all, data and information are not interchangeable concepts. So, the abundance concerns the data, not the information. Data needs to be *interpreted* to become information. This interpretation (normally done by humans) is even in human communication troublesome.

Obviously, the objective of an Information Retrieval system is to retrieve information. However, since computers are only capable of storing and analysing *data*, a fundamental Information Retrieval problem is:

> How can we be sure that the retrieved data contains the information the user is looking for?

Although often not stated explicitly, all Information Retrieval research tries to solve this problem in one way or another. History shows that although a full

[1] At the moment of writing the search engine Google [BP98] claims to search through 2,469,940,685 webpages

solution is (theoretically) impossible, partial solutions work reasonably well.

1.1.1 Relevance

The objective of an Information Retrieval system, that is, to retrieve information, has traditionally been interpreted as finding *relevant* documents. The question of relevance has long been scrutinized and philosophized in information retrieval research (for a extensive survey see [Miz97]). According to Mizzaro [Miz98] there are different kinds of relevance, coupled with an inconsistent use of terminology. Througout this thesis we will use the most purest form of relevance, sometimes called *user relevance* [Vic59a, Vic59b]. User relevance involves a complex *human* judgement. Obviously such a definition defies formalization, since it deals with internal mental states. The relevance judgement we would like to ascribe to automated retrieval systems, is called *system relevance*. Somehow we would like to approximate the user relevance with system relevance. Effectively this involves solving the problem described in section 1.1.

The following section will shed a light on how classical approaches tried to solve this issue.

1.2 Information Retrieval Paradigm

To get a good impression of which themes are involved in the process of retrieving information we sketch the following hypothetical situation:

1.2.1 Classical library search

> *Imagine a painter called Michael who wants to make an artistic impression of the Martian night sky. He visits the local library to learn more about Mars.*

The search process starts with a *searcher* Michael with a certain *knowledge gap* [PB99]. Such a knowledge gap is rather *imprecise*, after all, it is the *absence* of knowledge that leads him to the search. Michael doesn't exactly know what he's looking for: he hopes to find information that resolves his knowledge gap. He therefore tries to picture the information he needs. This mental state is called *Information Need*.

> *In the library he talks with the librarian. He explains that he is looking for information about Mars.*

Here Michael formulates his Information Need *'I am looking for information about Mars'* in natural language and communicates it with the librarian. Michael

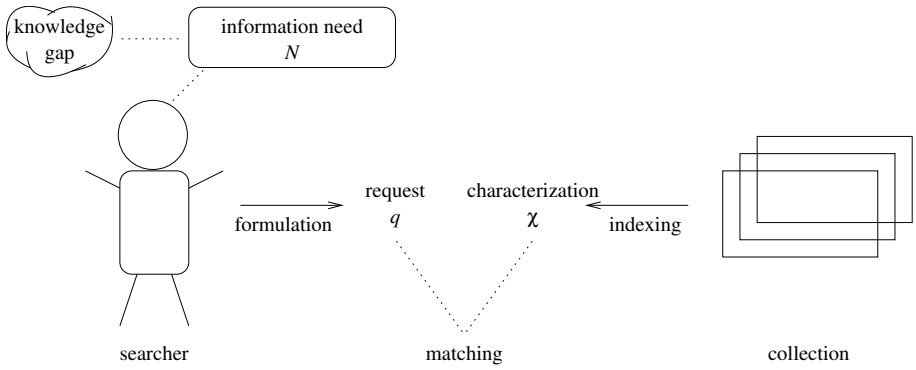


Figure 1.1: The Information Retrieval Paradigm

hopes his formulation (or *query*) is understood by the librarian. The librarian might ask Michael if he is looking for information about the *planet* Mars or about the candy bar with the same name.

> *The librarian walks to the astrophysics section of the library and finds a book called 'The Red Planet'.*

The books in the library are not stored randomly; the are organised in a particular manner such that finding the correct book is relatively simple. Note that the librarian selects this particular book because she hopes it is relevant for Michael. This selection process is called *matching*, since it matches the book's contents with the query. To determine if a book may be relevant, the librarian can read the entire book. Of course this is unfeasible for large libraries. In most cases, reading the summary, or just the title of the book is enough to determine the possible relevance. Such a summary (or title) is called a *characterisation* of the book. The process of assigning characterisations and organising the books is called *indexing*.

> *Michael browses through the book which contains information about Mars. He learns about the Marian atmosphere, the red sandstorms and, more importantly, about the two satellites Phobos and Deimos. Inspired by this new found knowledge he formulates another (more detailed) query to gather information on these two moons.*

After inspecting the book, Michael determines that the book is relevant. A part of his original knowledge gap has been resolved. Simultaneously he is able to reformulate his query because he has more insight on the issue.

Michael's search process is depicted in figure 1.1. We use N to denote the Information Need and q to denote the query. The characterisation of a document d will be denoted by $\chi(d)$.

1.2.2 General problems

The library search described above was successful. In general, finding information is not easy. We will shortly describe which obstacles may complicate the search process. Note that the following issues are general search related problems.

- *What is it that I am looking for?*
  The searcher's knowledge gap can be fairly specific such as learning about *the length of the Chinese Wall*, to rather vague, such as in our example *learning about the Martian night sky*. An important tool for dealing with unclear knowledge gaps is *interaction*. During the process of searching, the searcher learns new information which enables him to update his knowledge gap and reconsider his information need. To stress the interactiveness of this activity, researchers talk about *Information Discovery* instead of Information Retrieval.

- *How do I formulate what I am looking for?*
  Since information need is a mental state, it is not evidently easy to formulate it into a query. It is possible to have a precise concept in your mind, while unable to formulate it in words. This is most easily demonstrated by the difficulty of describing things in a foreign language. An extra difficulty is the fact that formulation needs to be *communicated*. In our example, the librarian (who eventually selects a book) must understand the query. This means that the searcher and the librarian must share a certain amount of knowledge and that the searcher has to express himself in a language understood by the librarian.

- *Inadequate indexing*
  In order to select a correct book, the librarian matches the query with the characterisations of the books. Firstly, characterisations may be incomplete (they are generally smaller than the corresponding book) and secondly, they must be understandable to the librarian. If, in our example, the characterisations are limited to the book's titles, the librarian must know that Mars and 'the red planet' share the same concept, otherwise the book is deemed to be non-relevant.

- *Coverage*
  Until now, we assumed that retrieved books contained information to solve the initial knowledge gap. We did not consider the process of gathering the information out of the books. Obviously, some of the books may be harder to read than others. It may even occur that the required information is scattered througout the retrieved material. In all these cases an optimal coverage is needed. Recent research [**?**] deals with these

issues. Elaborated educational systems like [**?**] are capable of handling hierarchical dependencies (pre-requisites) between (educational) books.

1.2.3 Automated Retrieval

At present, only a few libraries have librarians who perform search tasks as sketched in the above example. These tasks have been delegated to automated retrieval systems. These retrieval systems should be able to find relevant books, just like the librarian did. We will walk through our retrieval example and depict what capabilities such a system should have.

Of course, the first part of the search process remains the same. Michael forms a query and communicates it to the system. As in the case of the librarian, Michael hopes the system *understands* his query. Strictly speaking, the system does not understand a thing. But on the other hand, how was Michael so certain the librarian understood his query? Without falling into Turing test debates [**?**] it is reasonable to expect that Michael believes the system understands his query as long as he receives the right books[2].

In order to select the right books, the system should be able to:

1. extract essential information that specifically characterises the book in such a way that the system is able
2. to efficiently match this information with the query and
3. (exactly) select the books which the user is looking for.

In the next sections we will discuss how different IR systems try to perform these tasks.

1.2.4 Term based disclosure

At present, almost all automated retrieval systems use a variant of the *term based* disclosure system. The central principle in this approach is that terms hold a descriptive (conceptual) value.

These systems try to characterize documents using word sets. The words are normally taken from the document in an automated fashion. Essentially the systems assumes that:

> If a certain word occurs in a document, that document is *about* that word.

Note that *word order* plays no role in these models: the famous example about a document called 'The Hillary Clinton Bill Proposal' which contains the words

[2]One may call the set of all relevant books the *existential meaning* of his query.

'Bill' and 'Clinton' illustrates the importancy of word order, which is lost in this approach. Matching in this model involves comparing word occurrences in query and characterisations, which can be done in numerous ways. All term based models rely on the following hypothesis:

> *If a document and the query have a word in common, the document is likely to be relevant.*

1.2.5 Improvements to term based models

The rudimentary model described above can be improved easily. First of all the observation that not every word has the same descriptive power leads to stopword removal **??** and weighted word systems **??**. Different word forms which have the same underlying conceptual meaning can be mapped upon each other using stemming mechanisms [Por80]. Another way of boosting recall may be obtained by using thesaurus-like systems that add synonyms.

It is astonishing that term based disclosure works so well. The main reason for this is the fact that there is a strong relation between words and their conceptual meaning. This relation makes it possible to get a grip on document semantics. At the same time it displays it weaknesses:

- there are a lot of words with different meanings,
- the interword relations are not part of the disclosure,
- words that are 'meaningless' on their own, could have an important meaning in a bigger context. Compare 'president' with 'the president'.

Annual TREC [Har00] evaluations show that term based information retrieval systems have reached the limit of their performance.

1.3 Natural Language Processing

With the rise of the information age, possibilities opened up to use computers for Natural Language Processing that were not thinkable before. Executable language models, grammars and parsers create the possibility to automatically detect bigger linguistic units than words [Hie01]. However, using Natural Language Processing for Information Retrieval not necessarily mean better performance [SJ98]. It remains to be seen if NLP driven systems can outperform for example n-grams or systems exploiting co-occurrences. Nevertheless people start experimenting extending the descriptor language to for example *index expressions* [Bru93]. Index expressions are simplified noun phrases which obviously have a higher descriptive power. Obviously, the use of expressions will raise precision, but due to the rarity of common phrases will reduce recall.

A solution to this is introducing subexpressions which makes up for the lost recall.

Apart from the descriptive power, the nature of index expressions gives the possibility to create a structure by which searchers can navigate through, refine and enlarge their retrieval results.

1.4 Research questions

This thesis is an attempt to contribute to the Information Retrieval field by describing methods to find, represent and use conceptual structures from natural language input. It is a *pragmatic* approach, not only since its main reason is to practically improve retrieval systems, but also because it is inspired by pragmatism, the philosophical movement that claims that the meaning of conceptions is to be sought in their practical bearings [**?**].

This thesis answers the following research questions:

1. Is it possible to use recent Natural Language Processing techniques to increase the effectiveness of Information Retrieval?

2. Is it possible to index a collection in such a way that it captures the meaning of its information items more accurate than word based models?

1.5 Thesis organisation

This thesis is organised in two parts. Part I is dedicated to natural language and ways to extract information from natural language input. It starts with defining a relational language model (Chapter 2), followed by a demonstration the model's expressiveness by tackling a non-trivial linguistic problem (Chapter 3). To make things more concrete Chapter 4 presents an executable specification of an algorithm capable of parsing without a grammar. Finally Chapter 5 introduces Peirce's semiotic and suggests an approach for modelling the information content of natural language input.

Part II of this thesis puts the newly found knowledge to work. It starts with introducing 'formal concepts' (Chapter 6) and presents the relation with index expressions. Chapter 7 describes how concept lattices can be used to perform conceptual relevance feedback. Finally Chapter 8 concludes this thesis by summarising the research achievements and by suggesting directions for future research.

Part I

Language

Part II

Concepts

Chapter 6

Concepts and their properties

Parts of this chapter were taken from [Gro00b] and [Gro00a].

Given their simple nature, the success of keyword based retrieval systems is astonishing. Although these methods apparently only process words (and their word counts), they rely on and endorse most of their success to the keywords' *implicit semantics*. In fact each keyword is a representation of a thought, or *concept*.

Due to recent developments in NLP it is possible to use larger syntactical units (like noun phrases) for IR purposes. Simply using these units as 'large keywords' drastically boosts precision, but due to their rare occurrences they really hurt recall. A way to deal with this problem is to add subphrases. The attained phrases can be structured into a lithoid, which forms an ideal starting point for feedback mechanisms like *Query By Navigation* [BvdW92a].

This paper will go even further and tries to involve the phrases' support into the structure using a mathematical theory called *formal concept analysis* [Wil82, Har82]. The resulting concept lattice shows great similarity to the original lithoid and, since it consists of *formal concepts*, guides the way to handling phrase semantics.

6.1 Introduction

Mainly due to the progress in natural language processing, over the last decade there has been a widespread belief that NLP can successfully be used to solve IR problems. In fact, NLP has contributed to IR for years: every keyword based IR system uses an NLP based word stemming algorithm. However, modern IR research seems to focus on a more ambitious goal: the use of larger syntactical units than those of words, namely phrases. The question at issue remains whether phrasal terms contribute more to retrieval performance as opposed to statistical techniques exploiting for example co-occurence [SJ98, MBSC97].

One of the major issues is the fact that information retrieval must deal with *relevance*. After all, the purpose of a retrieval system is to retrieve all relevant documents, and as few of the non-relevant documents as possible. Since relevance is a judgement which relies heavily on semantics, it is obvious that an IR system should deal with semantics in some way or another. This may seem strange at first, but it is exactly why keyword based IR is so successful. Although the system only manipulates words and their frequencies, it implicitly uses the fact that keywords represent *concepts*. Syntactic noise (word forms, conjugations) can be eliminated by stemming while homonymy (word ambiguities) can only hurt recall. The tight coupling of words and their semantics is illustrated by attempts like [MRF+90] and [CH95].

For phrases the picture becomes vaguer. Although originating from a thought or concept, a phrase is syntax-ridden. Attempts like [AvdWKvB00] which use linguistically motivated indexing schemes show positive results, but it still seems to be very difficult to get a grip on the semantic value of a phrase, especially since the syntactical structure of a phrase in itself does not really contribute to the semantical value, it merely mediates between the semantical values of its units. Language models like Wordgrammar [Hud84] and NLCA [KS98], which concentrate on word-word relations instead of on a predefined phrase structure might be advantageous here.

Apart from semantic properties, phrases have an additional value in feedback mechanisms. Using *Query By Navigation* [BvdW92a], searchers may be confronted with several (phrasal) refinements or generalisations of their current focus.

6.1.1 Setting

In this chapter we will focus on document collections. In order to retrieve the information that they contain with minimal effort, the documents in the collections are coupled with a disclosure system to assist a searcher upon retrieval. We denote the collection of all documents with the letter $\mathcal{D}$. For individual members of this collection (thus the documents themselves) we will write d_1, d_2 etc, while subsets are written as D_1, D_2.

To avoid trivial definitions we use *function* and *relation lifting*. Suppose f is a function from set $\mathcal{A}$ to set $\mathcal{B}$. We define the lifted function $\acute{f}$ from $\mathcal{P}(\mathcal{A})$ to $\mathcal{P}(\mathcal{B})$ as:

$$\acute{f}(A) = \{f(a) \mid a \in A\}$$

For a (infix) relation $\sim \subseteq \mathcal{A} \times \mathcal{B}$ we define $\acute{\sim} \subseteq \mathcal{A} \times \mathcal{P}(\mathcal{B})$ and $\grave{\sim} \subseteq \mathcal{P}(\mathcal{A}) \times \mathcal{B}$ as:

$$a \acute{\sim} B \iff \underset{b \in B}{\forall} [a \sim b]$$

$$A \grave{\sim} b \iff \underset{a \in A}{\forall} [a \sim b]$$

Since it is clear (by their arguments) which function/relation is meant, we generally omit the accents.

During the disclosure process (sometimes called indexing) descriptors are attached to documents. Since we are interested in IR aspects we try to abstract from linguistic problems by using a simple characterisation language (which is in some degree comparable to natural language) in which we can express our descriptors. This language should be powerful enough to be illustrative and simple enough to allow reasoning without getting lost in linguistic detail.

To formalise the disclosure, a characterisation function χ is defined which assigns a set of descriptors to each document. The possibility of having more than one descriptor for a single document is intentional and can lead to a better disclosure (as will be shown in section 6.3.3).

6.2 The language of index expressions

Index expressions extend term phrases which model the relationships between terms. In this light, index expressions can be seen as an approximation of the rich concept of noun phrases. Their philosophical basis stems from Farradane's *relational indexing* [Far80a, Far80b]. Farradane projected the idea that a considerable amount of the meaning in information objects is denoted in the relationships between the terms.

Craven ([Cra78], [**?**]) uses a similar approach to Farradane in his *linked phrase indexes*. He describes a linked phrase index as a network of terms, in which the arcs correspond to relationships denoted by prepositions. Index expressions have roots which lie in linked phrase indexes.

Definition 1 (Language of index expressions)
Let T be a set of terms and C a set of connectors. The language of index expressions $\mathcal{L}(T, C)$ is defined over the alphabet $\Sigma = T \cup C \cup \{\,(\,,\,)\,\}$ using structural induction:

(i) t is an index expression (for $t \in T$).

(ii) $e_1 \circ c(\,e_2\,)$ is an index expression (for index expressions e_1, e_2 and $c \in C$).

Usually, if there is no means for confusion, we omit the parentheses when writing index expressions. Likewise, we will write $\mathcal{L}$ rather than $\mathcal{L}(T, C)$.

It is easy to see that an index expression $e \in \mathcal{L}$ can be written as $t \circ_{i=1}^{k} c_i(e_i)$, for some $k \in \mathbb{N}$, $t \in T$, $e_i \in \mathcal{L}$, and $c_i \in C$. The term t is called the *lead term* of this index expression.

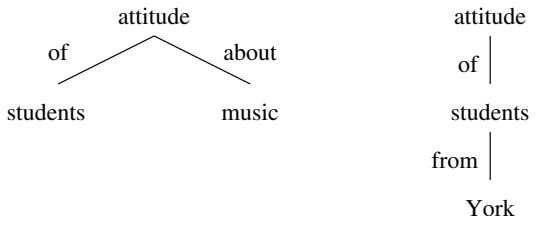
6.2.1 Tree diagrams

The structure of index expressions can be visualised using tree diagrams. A tree diagram can be constructed from an index expression $e = t \circ_{i=1}^{k} c_i(e_i)$ as follows:

- create a node and label it t,
- for each $i \in \{1, \ldots, k\}$, create a branch labelled by connector c_i to the tree diagram resulting from the index expression e_i.

Consequently, the empty index expression is represented by the empty tree.

Examples:



6.2.2 Subexpressions

Using index expressions instead of term descriptors yields a clear advantage in precision. However, due to the rare occurrences of exact phrases they cause bad recall. To experience the best of both worlds we will introduce subexpressions.

The notion of subexpressions is most easily introduced as the subtree relation on tree diagrams. We will present a formal definition of this later on.

Definition 2 (Subexpression)
We call index expression e_1 a subexpression of index expressions e_2, denoted as $e_1 \sqsubseteq e_2$, if and only if the tree associated with e_1 is a subtree of the tree associated with e_2.

Building upon the idea of an index expression, the so called *power index expression* is introduced. This notion bears a strong resemblance with the power set concept: it can be viewed upon as the set of all its subexpressions.

Definition 3 (Power index expressions)
Let $e = t \circ_{i=1}^{k} c_i e_i$ be an index expression. The set $\Lambda(e)$ of *lead expressions* belonging to e is defined as follows:

$$\Lambda(e) \stackrel{\text{def}}{\equiv} \bigcup_{(b_1,\ldots,b_k)\in\{0,1\}^k} t \circ_{i=1}^{k} (c_i \Lambda(e_i))^{b_i}$$

The power index expression belonging to e, denoted by $\mathcal{P}(e)$, is the set

$$\mathcal{P}(e) \stackrel{\text{def}}{\equiv} \Lambda(e) \cup \bigcup_{i=1}^{k} \mathcal{P}(e_i)$$

Using this definition we can now formally define what a subexpression is:

Definition 4 (subexpression)
Let e_1 and e_2 be two index expressions, then:

$$e_1 \sqsubseteq e_2 \stackrel{\text{def}}{\equiv} e_1 \in \mathcal{P}(e_2)$$

Note that the relation $\sqsubseteq$ is reflexive, antisymmetric and transitive; in fact, ($\mathcal{L}, \sqsubseteq$) is a poset.

Examples:

Λ(`attitude OF students FROM York`)= {`attitude OF students FROM York`,
`attitude OF students`, `attitude`}

$\mathcal{P}$(`attitude OF students FROM York`)= {`attitude OF students FROM York`,
`attitude OF students`, `attitude`,
`students FROM York`,
`students`, `York`}

The fact that two expressions are subrelated and that there is no (other) expression 'in between' is reflected by:

Definition 5 (direct subexpression)
e_1 is a direct subexpression of e_2, denoted as $e_1 \sqsubseteq_1 e_2$, if and only if:

$$e_1 \sqsubseteq e_2 \wedge \underset{e \in \mathcal{L}}{\forall} [e_1 \sqsubseteq e \sqsubseteq e_2 \Rightarrow e_1 = e \vee e_2 = e]$$

6.3 Getting the picture

We assume a finite collection $\mathcal{D}$ of documents, and a characterisation function χ that assigns a (yet again finite) set of descriptors to each document. These descriptors are taken from the index expression language $\mathcal{L}$. Each document will have at least one descriptor assigned to it.

Definition 6
We will write $e \sim d$ (e characterises d) if and only if $e \in \chi(d)$.

Define the set of all (initial) descriptors $\chi(\mathcal{D})$ as follows:

$$\chi(\mathcal{D}) = \bigcup_{d \in \mathcal{D}} \chi(d)$$

Let $\mathcal{E}$ be the set of all the used index expressions (including their subexpressions), a subset of $\mathcal{L}$.

$$\mathcal{E} = \bigcup_{i \in \chi(\mathcal{D})} \mathcal{P}(i)$$

We extend $\sim$ to subexpressions:

$$e \sim d \Longleftrightarrow \underset{i \in \chi(D)}{\exists} [e \sqsubseteq i \wedge i \sim d]$$

This extension states that if a document is characterised by an index expression, all subexpressions of that particular index expression are also characterisations of that document.[1]

6.3.1 The creation of the lithoid

In this section, we describe the so-called lithoid, a crystalline structure which organises document descriptions. This structure provides an overview of the contents of all the documents from the collection, and can be used to support searchers in formulating their information need via a process called *Query by Navigation* ([BvdW90]).

A lithoid is a graph containing nodes and edges. Each (sub)expression in $\mathcal{E}$ is represented in the graph by a node. Two nodes with their corresponding expressions e_1 and e_2 are connected with an edge if and only if $e_1 \sqsubseteq_1 e_2$.

For example, consider the following collection:

d_1	*PDP11 architecture handbook*	`handbook IS (pdp11) IS (architecture)`
	Instruction set of the PDP11 cpu	`set IS (instruction) OF (cpu IS (pdp11))`
d_2	*Microsoft 386 handbook*	`handbook IS (microsoft) IS (386)`
	Instruction set of the 386 cpu	`set IS (instruction) OF (cpu IS (386))`
d_3	*Comparing the PDP11 with the 386*	`comparing THE (pdp11) WITH (386)`
d_4	*386 architecture*	`architecture IS (386)`

This collection consists of 4 documents. Some of the documents have more than one description. Take note of the original description and the more structured index expressions. The special connector '`IS`' is used to reflect noun-adjective and noun-noun relations.

The corresponding lithoid (see figure 6.1) has 28 nodes.

[1] The validity of this assumption might be doubted, but is an explicit choice for the model we are using. Its argumentation goes beyond the scope of this chapter.

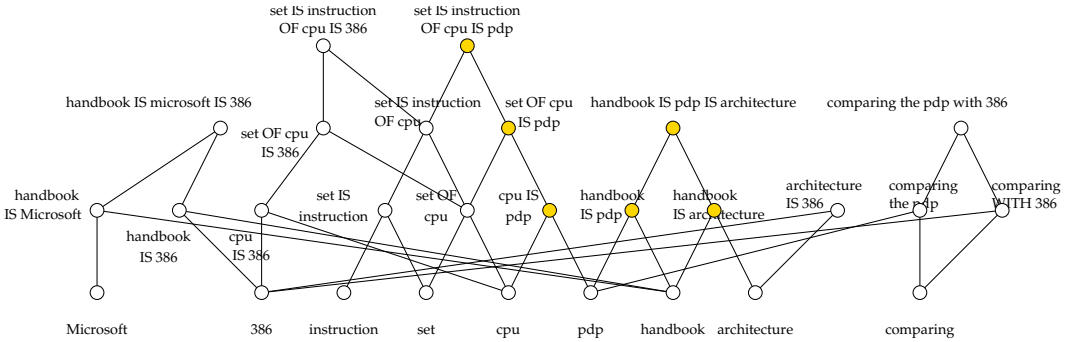


Figure 6.1: Example lithoid

6.3.2 Navigation

Before we show how the lithoid can be used as a hyperindex we must populate the nodes:

Definition 7 (Yield)
Each node of the lithoid has a *yield* containing all the documents that it supports:

$$\Upsilon(e) = \{d \in \mathcal{D} \mid e \sim d\}$$

Navigation starts at one of the lower (term index expression) nodes of the lithoid.[2] Let us assume the searcher starts with `386` as its focus. The following refining alternatives are presented:

- *386 handbook* (`handbook IS 386`)
- *386 cpu* (`cpu IS 386`)
- *386 architecture* (`architecture IS 386`)
- *comparing with the 386* (`comparing WITH 386`)

Suppose the searcher selects *386 handbook*. This new focus offers both a refinement as well as two enlargements:

- refine to *Microsoft 386 handbook* (`handbook IS microsoft IS 386`)
- enlarge the focus to *handbook*
- enlarge the focus (back) to *386*

Lithoid hyper-indices like this have successfully been used in the past (see for example [BBB91]).

[2]Sometimes an artificial 'bottom' element is added to the lithoid with connections to all of the term index expressions. This bottom element can be used as alternative startingpoint.

6.3.3 Incorporating support

While navigating through the example lithoid a few things catch our attention:

- There is little use of navigation between two nodes with exactly the same yield. For example, the shaded nodes in figure 6.1 have the same yield $\{d_2\}$.

- Two expressions with the same yield which are not both subexpressions of another expression will end up as two different (unconnected) lithoid nodes. This is highly undesirable, because searches might be able to benefit from this observation: it can give them a different view on their request. It is even possible that unexpected co-occurences may come to light. Obviously this situation can only occur when documents are characterised by more than one descriptor.

All of this has a simple reason: the lithoid is a structure based on *expressions*. It does not consider their support. You might say that the lithoid structure is static; it does not change if the number of documents at a particular node changes. The question remains: how do we incorporate support without losing the lithoid's navigational properties. In other words: how can the lithoid be structured both on expressions *and* on documents. As we shall see, the answer lies in a mathematical structure called *concept lattice*.

6.4 Concept lattices

6.4.1 The generation of a context

In order to make a semantical classification of the documents in our document collection we must look for relevant attributes. Within the theory of *Formal Concept Analysis* [Wil82], the relation between objects and attributes is called a context. In our case, it seems reasonable to use the characterisation of documents as a base to construct attributes. Using the linguistic structure of our characterisation language, we will use $\mathcal{E}$ as the set of *attributes*, $\mathcal{D}$ for the set of *objects*, and $\sim$ as the context relation. The context of our running example is depicted in figure 7.2.

6.4.2 The generation of a concept lattice

Using the context, we generate a classification of documents such that each class can be seen a *concept* in terms of properties of the document.

First it is important to recognise that documents share certain attributes.

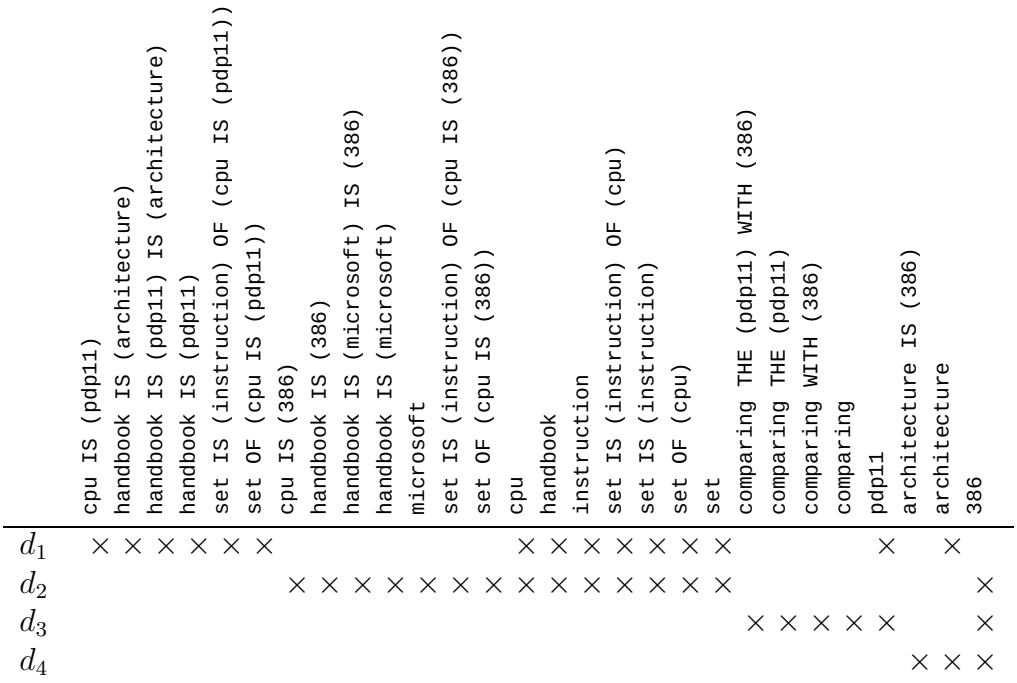


Figure 6.2: Example context

Definition 8
For the relation $\sim$ we define the right polar function **ComAttr** : $\mathcal{P}(\mathcal{D}) \rightarrow \mathcal{P}(\mathcal{E})$ as follows:

$$\textbf{ComAttr}\ (D) = \{e \in \mathcal{E} | e \sim D\}$$

On the other hand, documents shared by properties are captured by the left polar function **ComDocs** : $\mathcal{P}(\mathcal{E}) \rightarrow \mathcal{P}(\mathcal{D})$.

$$\textbf{ComDocs}\ (E) = \{d \in \mathcal{D} | E \sim d\}$$

Definition 9
A *concept* is a pair $(D, E) \in \mathcal{P}(\mathcal{D}) \times \mathcal{P}(\mathcal{E})$ with:

$$\begin{aligned} \textbf{ComAttr}\ (D) &= E \\ \textbf{ComDocs}\ (E) &= D \end{aligned}$$

Let $\mathcal{C}$ be the set of all concepts which can be derived from the set of objects $\mathcal{D}$, the set of attributes $\mathcal{E}$, and their context $\sim$.

Definition 10
Let $c_1 = (D_1, E_1) \in \mathcal{C}$ and $c_2 = (D_2, E_2) \in \mathcal{C}$. We define:

$$c_1 \subseteq c_2 \equiv D_1 \subseteq D_2$$

The fact that $(\mathcal{C}, \subseteq)$ is an order follows directly from the fact that $(\mathcal{P}(\mathcal{D}), \subseteq)$ is an order.

Definition 11
Let C be a subset of $\mathcal{C}$.

$$\begin{aligned} \textbf{lower bound}\ (C) &= \{c \in \mathcal{C} | c \subseteq C\} \\ \textbf{upper bound}\ (C) &= \{c \in \mathcal{C} | C \subseteq c\} \end{aligned}$$

If a greatest element exists in the set of lowerbounds of C, we call it the *greatest lower bound*. Similarly the smallest element in the set of upper bounds is called the *smallest upper bound*.

Lemma 1
Let T be an index set, and for all $t \in T$ let $D_t \subseteq \mathcal{D}$. Then:

$$\textbf{ComAttr}\ (\bigcup_{t \in T} D_t) = \bigcap_{t \in T} \textbf{ComAttr}\ (D_t)$$

Proof:

$$\begin{aligned} e \in \textbf{ComAttr}\ (\bigcup_{t \in T} D_t) &\iff e \sim d \ \text{ for all } \ d \in \bigcup_{t \in T} D_t \\ &\iff e \sim d \ \text{ for all } \ d \in D_t \ \text{ for all } \ t \in T \\ &\iff e \in \textbf{ComAttr}\ (D_t) \ \text{ for all } \ t \in T \\ &\iff e \in \bigcap_{t \in T} \textbf{ComAttr}\ (D_t) \end{aligned}$$

$\diamond$

Lemma 2
$(\mathcal{C}, \subseteq)$ is a total order.

Proof:
Since $(\mathcal{C}, \subseteq)$ is an order, it is sufficient to show that every subset C of $\mathcal{C}$ has a greatest lower bound and a smallest upper bound. Let T an index set and $c_t \in \mathcal{C}$ for all $t \in T$ such that:

$$\bigcup_{t \in T} c_t = C$$

Write c_t as (D_t, E_t). Consider the following pair:

$$\big(\textbf{ComDocs}\ (\textbf{ComAttr}\ (\bigcup_{t \in T} D_t)), \bigcap_{t \in T} E_t\big)$$

Since E_t is **ComAttr** (D_t) for all $t \in T$, rewriting this pair using lemma 1 results in:

$$\left(\textbf{ComDocs}\ (\textbf{ComAttr}\ (\bigcup_{t\in T} D_t)), \textbf{ComAttr}\ (\bigcup_{t\in T} D_t)\right)$$

And this is surely a concept, obviously being greater than all concepts c_t (since its extension is the intersection of the concepts c_t). That no smaller concept exists is trivial for the same reason. ◇

Consequently, a concept uniquely relates a set of documents (extension) to a set of attributes (intention): for a concept, the set documents implies the corresponding set of attributes and vice versa. For this reason a concept may be represented by its extension or its intension. Some precaution must be taken: not *every* set of documents represents a concept; if all their common attributes are shared by other (not included) documents they fail the premise of definition 15. The same holds for attributes.

Lemma 3
If (D_1, E_1) and (D_2, E_2) are concepts then:

$$D_1 \subseteq D_2 \Longleftrightarrow E_1 \supseteq E_2$$

Proof:
Using symmetry we only proof $(\Rightarrow)$.
Suppose $D_1 \subseteq D_2$, then obviously $D_2 \sim e \Rightarrow D_1 \sim e$ (for each attribute e), and thus $E_1 \supseteq E_2$. ◇

Consequently, the concepts form a lattice under $\subseteq$ for their extensions (or, alternatively, under $\supseteq$ for their intentions):

$$(D_1, E_1) \subseteq (D_2, E_2) \equiv D_1 \subseteq D_2$$

6.4.3 Example concept lattice

Using the context presented in figure 7.2 we find the following concepts:

$(\emptyset, \mathcal{E})$
$(\{d_1\}, \mathcal{P}(d_1))$
$(\{d_2\}, \mathcal{P}(d_2))$,
$(\{d_1, d_2\}, \{$`cpu, handbook, instruction, set IS instruction OF cpu, set IS instruction, set OF cpu, set`$\})$
$(\{d_3\}, \mathcal{P}(d_3))$,
$(\{d_4\}, \mathcal{P}(d_4))$,
$(\{d_2, d_3, d_4\}, \{$`386`$\})$
$(\{d_1, d_4\}, \{$`architecture`$\})$
$(\{d_1, d_3\}, \{$`pdp11`$\})$
$(\mathcal{D}, \emptyset)$

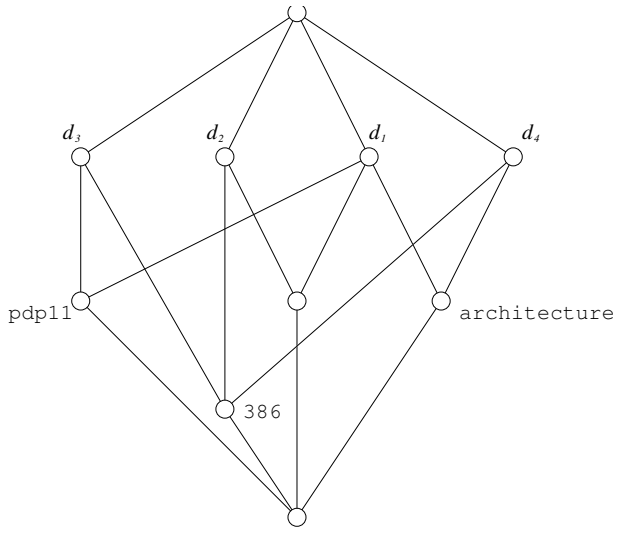


Figure 6.3: Example lattice

Figure 6.3 shows these concepts in their lattice.

6.5 The lithoid and the concept lattice

In this section we investigate the relationship between the lithoid and its corresponding concept lattice.

6.5.1 Concepts in the lithoid

Define an equivalence relation $\doteq$ between expressions as follows:

$$e_1 \doteq e_2 \iff \Upsilon(e_1) = \Upsilon(e_2)$$

Let $E_1, E_2, \ldots, E_n$ be a partition of $\mathcal{E}$ formed by $\doteq$, creating equivalence classes of expressions having the same yield. Because of this, and the fact that such a class in not empty, define:

$$\Upsilon(E_i) = \Upsilon(e) \text{ for some } e \in E_i$$

Extend the classes E_i to E_i^* as follows:

$$E_i^* = E_i \cup \{e \in \mathcal{E} \mid \Upsilon(E_i) \subset \Upsilon(e)\}$$

Effectively we extend a class E_i with expressions that have a yield which is a superset of the yield of E_i. Because all expressions in E_i have yield $\Upsilon(E_i)$ and all expressions that have $\Upsilon(E_i)$ are in E_i this definition may be rewritten to:

$$E_i^* = \{e \in \mathcal{E} \mid \Upsilon(E_i) \subseteq \Upsilon(e)\}$$

Lemma 4
ComDocs $(E_i^*) = \Upsilon(E_i)$

Proof:
Let T be an index set and e_t ($t \in T$) enumerating the set
$\{e \in E \mid \Upsilon(E_i) \subset \Upsilon(e)\}$.

$$\begin{aligned}
\textbf{ComDocs}\,(E_i^*) &= \textbf{ComDocs}\,(E_i \cup \{e \in E \mid \Upsilon(E_i) \subset \Upsilon(e)\}) \\
&= \textbf{ComDocs}\,(E_i \cup (\bigcup_{t\in T} \{e_t\})) \\
&= \textbf{ComDocs}\,(E_i) \cap (\bigcap_{t\in T} \textbf{ComDocs}\,(e_t)) \\
&= \Upsilon(E_i) \cap (\bigcap_{t\in T} \Upsilon(e_t)) \\
&= \Upsilon(E_i) \cap (\bigcap_{t\in T} (\Upsilon(E_i) \cup \Upsilon(e_t))) \\
&= \Upsilon(E_i) \cap (\Upsilon(E_i) \cup (\bigcap_{t\in T} \Upsilon(e_t))) = \Upsilon(E_i)
\end{aligned}$$

◇

Lemma 5
ComAttr $(\Upsilon(E_i)) = E_i^*$

Proof:

$$\begin{aligned}
e \in \textbf{ComAttr}\,(\Upsilon(E_i)) &\iff e \sim d \text{ for all } d \in \Upsilon(E_i) \\
&\iff \Upsilon(E_i) \subseteq \Upsilon(e) \\
&\iff e \in E_i^*
\end{aligned}$$

◇

Lemma 6
$(\Upsilon(E_i), E_i^*)$ is a concept over $(E, \mathcal{D}, \sim)$

Proof:
This can be derived directly from lemma 4 and 5. ◇

6.5.2 Concept lattice navigation

We will now show that navigation in the concept lattice is very similar to navigation in the lithoid. In fact, if two nodes are connected in the lithoid, their corresponding concepts are connected as well.

Definition 12
Let e be a (sub) expression, E its class formed by $\doteq$, and E^* its class extension. For each e we define the following:

$$\textbf{Concept}\ (e) = (\Upsilon(E), E^*)$$

Let e_1 and e_2 be two lithoid nodes.

Lemma 7

$$e_1 \sqsubseteq e_2 \Rightarrow \textbf{Concept}\ (e_2) \subseteq \textbf{Concept}\ (e_1)$$

Proof:
Let $E_1 = \Upsilon(e_1)$ and $E_2 = \Upsilon(e_2)$.

$$\begin{aligned}
e_1 \sqsubseteq e_2 &\Rightarrow \underset{d \in D}{\forall} [\text{if } e_2 \sim d \text{ then } e_1 \sim d] \\
&\Rightarrow \{d \in D \mid e_2 \sim d\} \subseteq \{d \in D \mid e_1 \sim d\} \\
&\Rightarrow \Upsilon(e_2) \subseteq \Upsilon(e_1) \\
&\Rightarrow \Upsilon(E_2) \subseteq \Upsilon(E_1) \\
&\Rightarrow (\Upsilon(E_2), E_2^*) \subseteq (\Upsilon(E_1), E_1^*) \\
&\Rightarrow \textbf{Concept}\ (e_2) \subseteq \textbf{Concept}\ (e_1)
\end{aligned}$$

$\diamond$

Note that two *directly* connected nodes need not necessarily have two directly connected concepts. The possibility exists that there is an intermediate concept.

6.6 Conclusions

This chapter presented an approach to extract semantical information from document collections using a combination of index expressions and concepts. In contrast to traditional methods, the disclosure process uses both the structure of the descriptors, as well as the structure of their support. This results in a lattice structure of formal concepts which can aid a searcher in formulating its queries. We have shown that the concept lattice provides a way to describe phrase semantics with the navigational properties of the original lithoid remaining intact.

Bibliography

[AA82] F. Aarts and J. Aarts. *English Syntactic Structures*. Pergamon Press, Oxford, 1982.

[Ada88] Douglas Adams. *The Restaurant at the End of the Universe*. Pan books Ltd., 1988.

[AvdWKvB00] Avi Arampatzis, Theo P. van der Weide, C. H. A. Koster, and P. van Bommel. An evaluation of linguistically-motivated indexing schemes. In *Proceedings of BCS-IRSG 2000 Colloquium on IR Research*, Sidney Sussex College, Cambridge, England, 5–7 April 2000. *0030*.

[BBB91] R. Bosman, R. Bouwman, and P. D. Bruza. The effectiveness of navigable information disclosure systems. In G. A. M. Kempen, editor, *Proceedings of the Informatiewetenschap 1991 conference*, pages 55–69, 1991.

[BDO95] M. W. Berry, S.T. Dumais, and G.W. O'Brien. Using linear algebra for intelligent information retrieval. *SIAM Review*, 37(4):573–595, 1995.

[BP98] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1–7):107–117, 1998, *0038*.

[Bru93] Peter D. Bruza. *Stratified Information Disclosure: A Synthesis between Information Retrieval and Hypermedia*. PhD thesis, University of Nijmegen, Nijmegen, The Netherlands, 1993.

[BvdW90] P. D. Bruza and Theo P. van der Weide. Two Level Hypermedia - An Improved Architecture for Hypertext. In A. M. Tjoa and R. Wagner, editors, *Proceedings of the Data Base and Expert Systems Applications*, pages 76–83, Vienna, Austria, 1990. Springer-Verlag.

[BvdW92a] P.D. Bruza and Theo P. van der Weide. Stratified hypermedia structures for information disclosure. *The Computer Journal*, 35(3):208–220, 1992, *0011*.

[BvdW92b] P.D. Bruza and Theo P van der Weide. Stratified hypermedia structures for information disclosure. *The Computer Journal*, 35(3):208–220, 1992.

[BYRN99] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley, 1999.

[CH95] A. S. Chakravarthy and K. B. Haase. NetSerf: Using Semantic Knowledge to Find Internet Information Archives. In E. A. Fox, P. Ingwersen, and R. Fidel, editors, *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 4–11, Seattle, Washington, July 1995. ACM Press.

[Cra78] Timothy C. Craven. Linked phrase indexing. *Information Processing & Management*, 14(6):469–476, 1978.

[DFS99] G. Debrock, J. Farkas, and J. Sarbo. Syntax from a Peircean perspective. In *5th International Congress on Terminology and Knowledge Engineering*, Innsbruck (Austria), 1999. *0023*.

[Dik68] Simon Dik. *Coordination. Its implications for the theory of general linguistics*. North Holland, Amsterdam, 1968.

[DP90] B.A. Dayvey and H.A. Priestley. *Introduction to lattices and order*. Cambridge University Press, 1990.

[DS98] G. J. Y. Debrock and J. J. Sarbo. Towards a Peircean model of language. Technical Report CSI-R9802, University of Nijmegen, 1998, *0031*.

[Far80a] J. Farradane. Relational indexing part i. *Journal of Information Science*, 1(5):267–276, 1980.

[Far80b] J. Farradane. Relational indexing part ii. *Journal of Information Science*, 1(6):313–324, 1980.

[FKS97] J. Farkaz, V. Kamphuis, and J. Sarbo. Natural Language Concept Analysis. Technical Report CSI-R9717, University of Nijmegen, 1997, *0024*.

[FS99] J. Farkaz and J. Sarbo. A Peircean framework of syntactic structure. In *Seventh International Conference on Conceptual Structures (ICCS'99)*, pages 112–126. Springer Verlag, 1999, *0025*.

[GKS98] F.A. Grootjen, V. Kamphuis, and J. Sarbo. A genealogy of phrase structure. Technical Report CSI-R9814, University of Nijmegen, 1998, *0036*.

[GKS99] F. Grootjen, V. Kamphuis, and J. Sarbo. Coordination and multi-relational modelling: 'X and X' revisited. In *6e conférence annuelle sur le Traitement Automatique des Langues Naturelles*, pages 345–351, Cargèse, Corse, 12–17 Julliet 1999. *0032*.

[Gro92] F.A. Grootjen. Efficient recursive backup parsing. Master's thesis, University of Nijmegen, Nijmegen, The Netherlands, 1992, *0037*.

[Gro98] F.A. Grootjen. NLCA: Towards an algorithmic implementation, presented at clin98. Technical Report CSI-R9811, University of Nijmegen, 1998.

[Gro00a] F. Grootjen. A semantical twist to syntactical navigation. In P Bruza, F Crestani, and M Lalmas, editors, *Proceedings of the eleventh international workshop on Data Base and Expert Systems Applications*, pages 523–528, Greenwich, London, UK, 2000. IEEE Computer Society, *0033*.

[Gro00b] Franc Grootjen. Employing semantical issues in syntactical navigation. In *Proceedings of BCS-IRSG 2000 Colloquium on IR Research*, Sidney Sussex College, Cambridge, England, 5–7 April 2000.

[Gro01] F.A. Grootjen. Indexing using a grammerless parser. In *Proceedings of the First International Workshop on Natural Language Processing and Knowledge Engineering (NLPKE 2001)*, Tucson, Arizona, USA, October 2001.

[GvdW02] F.A. Grootjen and Theo 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.

[Hae91] Liliane Haegeman. *Introduction to Government and Binding Theory*. Basil-Blackwell Ltd., 1991.

[Har82] G. M. Hardegree. An approach to the logic of natural kinds. In *Pacific Philosophical Quarterly*, volume 63, pages 122–132, 1982.

[Har93] D. Harman. Overview of the first TREC conference. In *Proceedings of 16th ACM SIGIR Conference*, 1993.

[Har96] Stephen J. Harlow. Generative grammar: Transformational grammar. In Keith Brown and Jim Miller, editors, *Concise Encyclopedia of Syntactic Theories*, pages 147–164. Pergamon/Elsevier, Oxford, 1996.

[Har00] Donna Harman. What we have learned, and not learned, from trec. In *Proceedings of BCS-IRSG 2000 Colloquium on IR Research*, Sidney Sussex College, Cambridge, England, 5–7 April 2000. *0042*.

[Hie01] Djoerd Hiemstra. *Using Language Models for Information Retrieval*. PhD thesis, University of Twente, 2001, *0041*.

[Hud84] R. A. Hudson. *Word Grammar*. Basil Blackwell Inc., 1984.

[Hui96] T. Huibers. *An Axiomatic Theory for Information Retrieval*. PhD thesis, University of Utrecht, 1996.

[Joh86] S. Johansson. The tagged LOB corpus user's manual. Technical report, Norwegian Computing Centre for the Humanities, 1986.

[Jol93] J. Jolly. Preposition assignment in English. In V. Valin, editor, *Advances in Role and Reference Grammar*. John Benjamins Publishing Company, Amsterdam Philadelphia, 1993.

[Kam98] V. Kamphuis. Coordination and surface structure. In *Actes du 16è Congrès International des Linguistes (Paris 20-25 juillet 1997)*. Elsevier Sciences, Oxford, 1998.

[KS98] V. Kamphuis and J. J. Sarbo. Natural Language Concept Analysis. In D. M. W. Powers, editor, *Proceedings of NeMLaP3/CoNLL98: International Conference on New Methods in Language Processing and Computational Natural Language Learning*, pages 205–214. ACL, 1998.

[Lam98] Joachim Lambek. Categorial and categorical grammars. In R.T. Oehrle, E. Bach, and D. Wheeler, editors, *Categorial Grammars and Natural Language Structures*, Dordrecht-Boston, 1998. D. Reidel Publishing Company.

[MBSC97] M. Mitra, C Buckley, A Singhal, and C. Cardie. An analysis of statical and syntactical phrases. In *Proceeding of RIAO-97, Computer-Assisted Information Searching on Internet*, Paris, 1997.

[Miz97] Stefano Mizzaro. Relevance: The whole history. *Journal of the American Society of Information Science*, 48(9):810–832, 1997, *0043*.

[Miz98] Stephano Mizzaro. How many relevances in information retrieval? *Interacting with Computers*, 10(3):303–320, 1998, *0044*.

[MRF+90] G. A. Miller, Beckwith R., C. Fellbaum, D. Gross, and K. J. Miller. Introduction to wordnet: An on-line lexical database. *Journal of Lexicography*, 3(4):234–244, 1990.

[MyGLLG02] M. Montes-y Gómez, A. López-López, and A Gelbukh. Information Retrieval with Conceptual Graph Matching. In *Proceedings of the 11th International Conference and Workshop on Database and Expert Systems Applications, (DEXA-2000)*, Greenwich, England, September 2002. Lecture Notes in Computer Science.

[PB99] H. A. Proper and P. D. Bruza. What is information discovery about. *Journal of the American Society for Information Science*, 50(9):737–750, July 1999, *0040*.

[Pei31] C. Peirce. *Collected Papers Of Charles Sanders Peirce*. Harvard University Press, Cambridge, 1931.

[Pei97] C.S. Peirce. Pragmatism as a principle and method of right thinking. In Patricia Ann Turrisi, editor, *The 1903 Harvard 'Lectures on Pragmatism'*. SUNY Press, Albany, NY, 1997.

[Por80] M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980, *0012*.

[PS94] Carl Pollard and Ivan Sag. *Head-driven Phrase Structure Grammar*. CSLI Publications, Chicago, 1994.

[QGLS85] R. Quirk, S. Greenbaum, G. Leech, and J. Svartvik. *A Comprehensive Grammar of the English Language*. Longman, London and New York, 1985.

[Sal90] R.A. Salvatore. *Exile*. TSR Hobbies, 1990.

[Sar96] Janos Sarbo. Lattice embedding. In *International Conference on Conceptual Structures*, pages 293–307, 1996.

[Sar97] J. Sarbo. Building sub-knowledge bases using concept lattices. *The Computer Journal*, 39(10):868–875, 1997, *0027*.

[Sar99] J. Sarbo. Formal conceptual structure in language. In D. M. Dubois, editor, *Proceedings of Computing Anticipatory Systems (CASYS'98),*, pages 289–300. Woodbury, New York, 1999, *0028*.

[SF95] J. Sarbo and J. Farkas. Knowledge representation and acquisition by concept lattices. In Shaul Markovitch, editor, *Proceedings of the 11th Izraeli Symposium on Artificial Intelligence (ISAI'95)*, Hebrew University of Jerusalem, Izrael., 1995. *0029*.

[SFG+99] J. Sarbo, J. Farkaz, F.A. Grootjen, P. van Bommel, and Theo P. van der Weide. Meaning extraction from a peircean perspective. In D. M. Dubois, editor, *Proceedings of Computing Anticipatory Systems (CASYS'99),*. Woodbury, New York, 1999.

[SGWW85] Ivan A. Sag, Gerald Gazdar, Thomas Wasow, and Steven Weisler. Coordination and how to distinguish categories. *Natural Language and Linguistic Theory*, (3):117–171, 1985.

[SJ93] Karen Sparck Jones. What might be in a summary? In Knorz, Krause, and Womser-Hacker, editors, *Information Retrieval 93: Von der Modellierung zur Anwendung*, pages 9–26, Konstanz, Germany, 1993. Universitätsverlag Konstanz, *0026*.

[SJ98] Karen Sparck Jones. Information retrieval: how far will *really* simple methods take you? In Djoerd Hiemstra, Franciska de Jong, and Klaus Netter, editors, *Proceedings Twente Workshop on Language Technology 14*, pages 71–78, 1998.

[ST93] Daniël D. Sleator and Davy Temperley. Parsing english with a link grammar. In *Proceedings of the Third International Workshop on Parsing Technologies*. 1993.

[Ste90] Mark J. Steedman. Gapping as constituent coordination. *Linguistics and Philosophy*, (13):207–263, 1990.

[Tol66] J.R.R Tolkien. *The Hobbit*. Houghton Mifflin Co., Boston, 1966.

[Vic59a] B. C. Vickery. The structure of information retrieval systems. In *Proceedings of the International Conference on Scientific Information*, volume 2, pages 1275–1290, Washington, DC, 1959. National Academy of Sciences.

[Vic59b] B. C. Vickery. Subject analysis for information information retrieval. In *Proceedings of the International Conference on Scientific Information*, volume 2, pages 855–865, Washington, DC, 1959. National Academy of Sciences.

[Wat94] Bill Watterson. *Homicidal Psycho Jungle Cat*. Andrews McMeel Publishing, 1994.

[Wil82] R. Wille. Restructuring lattice theory: An approach based on hierarchies of concepts. In I. Rival, editor, *Ordered sets*, pages 445–470. D. Reidel Publishing Company, Dordrecht–Boston, 1982.

[You67] D. H. Younger. Recognition and parsing of context-free languages in time n^3. In *Information and Control*, volume 10, pages 189–208, 1967.