Information Discovery

keywords:
Information Retrieval
Hyper{text|media}
{Inter|Intra}net

Th. P. van der Weide

April 14, 2001

Foreword

These lecture notes are the first part of a series on Information Retrieval and Hypertext. This part provides a formal introduction to Information Retrieval and Hypertext. The second part covers aspects of natural language, both as query and characterization language.

A summary (in dutch) of these notes can be found on Internet:

1. part 1: IR1 (www.cs.kun.nl/is/edu/ir1/)
2. part 2: IR2 (www.cs.kun.nl/is/edu/ir2/)

The lecture notes are base on PhD theses resulting from the Nijmegen Information Retrieval Group:

1. Peter Bruza
2. Theo Huibers
3. Frits Berger
4. Benrd Wondergem
5. Avi Arampatzis

Chapter 1

Introduction

[1] In the world there is increasing competition to manage information faster and more inexpensively. This competition is driven by the explosive growth of the amount of information available. For example, every year in the European Community about two million academic, scientific, medical and social-economic articles and books appear ([13]).

At the other end of the spectrum is the expansion of the amount of information available via the Internet (also known as the Web). The Internet can be viewed as a connected collection of accessible information stores, the so-called hosts. The tremendous growth in the number of hosts is shown in Table 1 (Source (www.nw.com/zone/WWW/top.html)). The digithrope Negroponte

Date	Hosts
08/81	213
05/82	235
08/83	562
10/84	1,024
10/85	1,961
02/86	2,308
11/86	5,089
12/87	28,174
07/88	33,000
10/89	159,000
10/90	313,000
10/91	617,000
10/92	1,136,000
10/93	2,056,000
10/94	3,864,000
01/95	4,852,000
07/95	6,642,000
01/96	9,472,000
07/96	12,881,000

Figure 1.1: The development of the number of Internet hosts

calculated that if the rate of growth of the number of Internet users were to continue at the rate at the time of writing (1996), which is of course impossible, the total number of Internet users would exceed the population of the world by 2003 ([73]).

To conclude, the amount of available information is currently growing at an incredible rate. Information about almost any subject is accessible leading to the presence of various information providers at the Internet and the development of powerful tools such as NCSA Mosaic and Netscape to browse, search and access this information. Since the digital highway virtually connects all parts of the world with each other, accessing information no longer presents a problem.

Be that as it may, a growing amount of information remains unused (or unread) simply because there are no means for retrieving this information effectively. In order to tackle this problem attempts have been made, since the early nineteen sixties, to create appropriate computer systems the so-called *Information Retrieval Systems*. (or: *IR systems* for short). In the next section we give a brief history of IR systems.

Information Retrieval Systems

IR systems

1.1 The history of information retrieval

In the early days of information storage, document collections were not large enough to be in need of an IR system, not even a manual one. For example, at the end of the fourteenth century,

[1]Based on [?]

the English poet Geoffrey Chaucer, in addition to being famous for his poems, also became very well-known for the fact that he owned about sixty books, which had cost him an immense amount of money ([75]). Of course, the number of books in the libraries was much larger, but still very modest. The library of the Sorbonne, which was known to be one of the largest in the fourteenth century, contained 1,722 books in those days ([75]).

catalogue

With the constant growth of the number of books an overview of a collection became necessary. Such an overview was presented in a *catalogue*. In 1604 the Catalogue of the Bodleain, the university library of Oxford, was printed for the first time. It was the largest general catalogue of the contents of any European library published up to that time. About 10,200 titles were described in the Catalogue, including subject lists of writers on Scripture, on Aristotle, on Law and on Medicine ([84]).

card-trays

The retrieval of documents was done by using the document references, which were first stored in catalogues and *card-trays*. With the arrival of the computer in the nineteen fifties, retrieval made a big step forward since as of that moment on, the references could be recorded in database systems. In the rest of this thesis we restrict ourselves to computerised systems only. Therefore, when referring to information retrieval (or IR systems), it is implicitly understood to be the computerised variant.

data retrieval

In the beginning of the computer era, information retrieval was focused on automatic data processing: not the information *in* a document but information *about* the document was stored. The representatives of a document were facts such as author name(s), title, number of pages, etc. These facts were recorded in database systems. The information *in* the document was only accessible if the document was in your hands. Literally, in those days information retrieval was *data retrieval*; if a requested fact *Writer: W. Shakespeare* was stored as a representation of a document entitled *Julius Caesar*, then a reference (place-code) of the document *Julius Caesar* was retrieved. With this reference in hand, one had to search for the actual document placed somewhere on a shelf.

Back then it was practically impossible to store the information that was contained in a document. However, the importance of being able to store this information was already recognised, as can be seen in the following quotation taking from the book 'Automatic Data Processing' of Brooks and Iverson ([14], page 52) that was written in 1963:

> *'The future importance of any aspect of an event is, however, not easily estimated, and much data are therefore recorded and retained for potential, though unspecifiable, future use. Such retention pertains particularly to* documents*, records which because of some validation are peculiarly acceptable as evidence.'*

information retrieval

A few years later, the *information* retrieval systems were establishing their prominent position among the computer systems. The information content of a document was represented by keywords or other representatives. So, the goal of an IR system was changed from *data retrieval* to *information retrieval*. For instance, in 1967 Martin ([71], page 164) described the task of an IR system as follows:

> *'A real-time [IR] system may be used to provide information about a service or a situation when it is required and where it is required.'*

For instance, given an IR system, the management of a company could obtain (or may wish to obtain) information about another company; the police could do a search in their document-bases to find out whether their is a suspect fitting their descriptions; a researcher may wish to know what literature exists on her topic.

Besides the fact that in data retrieval the user retrieves descriptors and in information retrieval she retrieves the object in question, there are some other essential differences between data and information retrieval. Books about information retrieval most often start with a list of the distinguishing properties of data and information retrieval. A summary of the principle differences as given by Blair ([10]), Turtle & Croft ([101]), and Van Rijsbergen ([78]) is shown in Table 1.1.

Given this table, it may be inferred that one of the primary tasks of an IR system is to provide information, definitely not just any piece of information but information that is relevant with respect to an information need.

From the nineteen seventies onward, IR systems steadily stored representatives of the information content of documents. Still, the IR systems did not provide information other than document ref-

	Data Retrieval	*Information Retrieval*
The representation of stored information	Well-defined types of objects and facts	Unstructured information
The method of answering a request for information	Direct, through facts	Information which will likely contain what the user wants
The relation between the formulated query to the system and the satisfaction of the user	Satisfaction or no satisfaction (deterministic)	A high likelihood that the user is satisfied
The definition of a successful system	Does the system deliver the requested facts?	Does the system satisfy the users' information need?

Figure 1.2: A summary of the differences between data and information retrieval

erences. For instance, Lancaster ([65]) formulated the terms *information retrieval* and *information retrieval system* in the context of items of literature as follows:

> *'The term information retrieval, as it is commonly used, refers to the activities involved in searching a body of literature in order to find items (i.e., documents of one kind or another) that deal with a particular subject area. An information retrieval system, then, is any tool or device that organizes a body of literature in such a way that it can be searched conveniently.'*

Natural Language Processing

In the nineteen eighties, a new generation of IR systems based on *Natural Language Processing* (NLP) techniques was proposed. These systems dealt with the text in the document as meaningful sequences of words rather than just as character strings (see for instance [9], [90]).

In the nineteen nineties, when multi-media is no longer a futuristic but a cognitively acceptable way to represent information, the information retrieval problem starts to explode. From this moment onwards, a user is not only searching for information contained in text, but also for information contained in sounds, images or video. Fortunately, a huge part of the media is stored in digital form. Documents are no longer exclusively on a shelf but also reside on a computer's accessible storage.

Now, being able to access the information content of a document directly, the task of an IR system has increased tremendously. The provision of information rather than a reference to a document has become the primary task of an IR system. As Van Rijsbergen and Lalmas ([82]) recently wrote:

> *'the purpose of an information retrieval system is to provide information about a request and that a request is a representation of an information need that an IR system attempts to satisfy. [...] if a user states a query then it behoves the IR system to find the objects that contain information about that query.'*

So, from the point of view of information storage, the brief history of information retrieval started with data retrieval and ends with multi-media information retrieval.

Internet

digital libraries

We can also detect another line of development in the history of information retrieval. At first most IR systems were originally developed to perform the 'classical information retrieval' task: a person was looking for relevant information in *one* collection of documents (often a library). Nowadays, the information retrieval problem is much broader as a user is searching for information contained in very large information stores, possibly spread around the globe. Two typical examples are the *Internet* as we mentioned previously, and *digital libraries*. Digital libraries consider related digital documents from all over the world as belonging to their collection. The user does not notice[2] the difference between consulting a document that is physically stored[3] in Australia or in the Netherlands. The task of a digital library is to retrieve documents which contain quality[4], non-redundant[5] information about a formulated request regardless of physical storage location.

[2]Or at least, the system should be able to keep this hidden for the user.
[3]In this sense, bits on a hard-disk, CD-Rom, tape or magnetic drum, and so on.
[4]Not all information is 'library' quality, such as the information in advertisement leaflets.
[5]An identical document taken from the Netherlands and Australia should not be retrieved twice.

Due to this rapid shift of paradigms, new requirements for IR systems have been recognised. The vast size of present-day information domains forces users to apply distributed retrieval systems to help them search distinct areas of cyberspace. These systems will in general not be static, but can adapt to the wishes of the user. They will also have to be autonomous to a considerable extent since it will be impossible for the user to oversee and guide their behaviour in detail.

rational agents

Over the past ten years, research in Artificial Intelligence has focused on defining highly autonomous systems displaying a rational behaviour and capable of solving complicated and elaborate tasks ([66]). These systems, which are commonly referred to as *rational agents*, seem tailor-made for helping to solve the information retrieval problem. Rational agents, with the ability to reason, communicate, gather and maintain information could probably be used as autonomous IR systems (cf. [26], [57], [58], [67]). On the Internet, information retrieval is performed by these agents: they are not called rational agents but they have fancy names such as spiders, webcrawlers, knowbots, and so on. It is commonly agreed that the success of the Web will depend strongly on the effectiveness of these agents. The following quotation from December ([41]) is a case in point:

> *'In an increasingly thin soup of redundant, poor quality, or incorrect information, even the smartest Web spiders won't be very effective. A flood of information unfiltered by the critical and noise-reducing influences of collaboration and peer review can overwhelm users and obscure the value of the Web itself. The Web certainly needs solutions in information discovery and retrieval –indeed, developing intelligent spiders, worms, robots, and ants is crucial to making sense of the Web.'*

One way of ensuring these changes is to combine and improve existing IR systems. This can only be done if we have a deep insight in the retrieval process, based on what is needed and what we already have. However, in the past thirty years of information retrieval research it has become clear that it is not evident at all how to analyse, compare, and improve the retrieval processes of different IR systems. Our main aim is to study and analyse IR systems in order to gain a better insight into the retrieval process.

1.2 Information retrieval paradigms

information objects **archive** **document**

In this series we are concerned with (very) large sets of *information objects* that are weakly structured. Examples of such collections are libraries, large archives (art, scientific, company information, etc). Examples of another order of magnitude are telephone books and technical manuals. The largest collection is the distributed collection offered by the World Wide Web. We use *archive* as a generic term for such collections. The elements of archives (the information objects) are also referred to as *documents*.

Information Retrieval **filter** **routing**

The term archive is a reminder to the static use of a document collection as a receptacle for information. The documents are passive objects, waiting to be retrieved by (active) searchers. The name for this activity is *Information Retrieval*. The roles can be interchanged, making the documents active, trying to find persons for which they are attractive. This is especially relevant on the Internet, where information is offered at such a pace that individual searcher will fail to keep track of all new developments. For this purpose, a *filter* is introduced as an active information agent. In document oriented organizations documents also play an active role. All activities of the organization are related to the flow of documents within the organization. However, the activities are organized around functions within the organization. In such organizations a considerable amount of time is usually spent to trace down documents. A modern approach is to organize the activities around product lines. In this approach we can see a document as an active element looking for the next processing person. This type of problem is referred to as *routing*.

Information Retrieval System **searcher**

The manager of the archive is called an *Information Retrieval System*. In classical archives, this system is completele driven by human processors. A most important person is the librarian, who is the intermediary between the client of the system (called the *searcher*) and the archive. In modern systems, human processors are replaced by machines. The intention of this replacement is a more effective and efficient Information Retrieval System. However, replacing human beings by machines introduces the problem of building an interface that imitates human behaviour and intelligence as close as possible.

Figure 1.3: The Information Disclosure Paradigm

searcher

When a *searcher* consults an archive, we can distinguish the following phases (see figure 1.2):

information need
formulation

1. The searcher has some unspecified *information need*, and tries to communicate this need to the Information Retrieval System. This phase is called *formulation*.

matching

2. The Information Retrieval System has (global) knowledge of the contents of documents in the archive, and tries to investigate how closely this contents meets the request of the searcher. This phase is called *matching*.

   characterization
   indexing

   The contents of a document is described in a *characterization*. The formation of a characterization is called *indexing*. From this characterization the Information Retrieval System will estimate the relevancy of the document. Note that this (objective) relevance estimation can differ from the (subjective) idea of relevance of the searcher.

presentation

3. The documents that are deemed to be (sufficiently) relevant have to be presented to the searcher. This phase is called *presentation*. In a conventional library the librarian will offer the selected documents. More sophisticated approaches are possible, for example, an automated Information Retrieval System will offer abstracts only in order to let the searcher select relevant documents.

The above paradigm dates from the late fifties when the field of information retrieval was emerging from the realms of Library Sciences. These days, objects need no longer be modelled as amorphous things. Due to the emerging standards such as $\TeX$, SGML and HTML, textual objects can be assigned a structure which can be taken into account. For other types of objects, for example images, a structural breakdown is a research issue (see for example [2]). An underlying structure is not only useful for disclosing the information contained in the information object, but must also be maintained in accordance to a structural specification. Therefore, the information disclosure paradigm should explicitly make provision for structural aspects.

Querying vs Browsing

In recent years, the formulation process has developed another angle of incidence due to the appearance of hypertext and hypermedia systems. From an information disclosure perspective, the primary facet of such systems is that the information need of the searcher is satisfied by a process of navigation (sometimes also referred to as browsing). Browsing implies that the information need is not formulated into a request. Formulation of the need is acknowledged as being difficult and error prone. (See [31]). Navigation is supported by advanced user interfaces, an aspect which is increasingly being recognized as something which influences the effectiveness of disclosing the underlying information.

The advantage of satisfying the information need by navigation is its disadavantage at the same time. The time required to reach at the required information is related directly to the distance from the starting point. This can be improved by special techniques such as a searcher letting set bookmarks in the archive.

query by navigation

In these lecture notes, a combination of formulation and navigation is demonstrated. The result is called *query by navigation*.

associative link

An important difference between hypermedia systems and (conventional) information systems is the concept of *associative link*, that enables the user to navigate through the information base. Furthermore, state-of-the-art hypermedia systems, in contrast to conventional information systems, feature almost no conceptual description of the stored data. The weaknesses of such an approach have been discussed by several authors ([47], [96]). There seems to be a growing need to be able to support a conceptual description with regard to both hypermedia and traditional document information systems. The combination of structured documents with hypermedia applications looks promising.

The state-of-the-art for finding (mostly textual) documents is based on the use of keywords, both for characterizing the information objects and as a base for the retrieval language. The performance of keyword-based systems is far from being satisfactory. The possibilities for the further improvement of recall and precision based on keywords seem to be rather marginal. Both for experienced and inexperienced users the use of keywords is complicated, not supportive and hardly robust. Besides, using keywords is inadequate for inflected languages (most other languages than English). Citing C.J. van Rijsbergen [81]:

> *"A big question, that has not yet received much attention, concerns the extent to which retrieval effectiveness is limited by the type of document description used. The use of keywords to describe documents has affected the way in which the design of an automatic classification system has been approached. It is possible that in the future, documents will be represented inside a computer entirely differently."*

The emergence of networks as information provider, especially the world wide web, has put a new challenge on Information Retrieval. The overwhelming amount of information that is being offered dayly requires an effective mechanism to subscribe to specific kinds of information, filtering the rest.

The main theme of the Filtering Problem is to provide an effective agent as an intermediate between information sources and information users. A filtering mechanism can thus be seen as an information broker.

Information sources contain hugh amounts of information, usually in the form of a collection of information objects. These sources may have a large modification rate. Information users may also have varying information needs. As a result, the information broker should track information users on a regular basis to detect such changes. Besides information-object-triggered and information-user-triggered information interchange, there may be cases of recurrent interest. In this case, the information broker takes the initiative for information interchange. The information broker should thus be as autonomous as possible.

The *filter* problem can be seen as a dual problem of Information Retrieval. In Information Retrieval, an information user has an *information need*, which is formulated as well as possible in terms of some query language. The resulting query is matched against the descriptions of information objects resulting in a relevancy estimate. In the filter problem, the information object has a *distribution need*. The description of the object can be seen as a description of its distribution need. The information broker will match this against user profile's, i.e., descriptions of the information users.

In these lecture notes, we discuss the fundamentals of Information Retrieval, and describe the stratified hypermedia architecture as a general architecture for hypertext and hypermedia applications. We discuss the traditional retrieval models, and pay attention to the various approaches to matching documents and queries. In these lecture notes, we mainly focus on formulation and matching. The problem of indexing is discussed in the second part of this series. Physical storage of documents falls outside the scope of this series. It should be noted that data compression plays a special role as collections, augmented with indexing information, can be very large.

IRIS Research Line

These lecture notes can also be seen as a reflection on the research which has been performed by the *IRIS Research Line* in the period from 1975. This research focussed on conceptual modelling on the one hand, and on information retrieval on the other hand. A major goal of the research group was the integration of both world, taking the best of both.

Chapter 2

Approaches to IR

In the remainder of these lecture notes we present two different possibilities for studying information retrieval. According to several authors ([10], [36], [88]) there are two possible avenues to follow for an information retrieval study, namely an experimental one and a theoretical one. The two approaches appear difficult to reconcile. The controversy between the people who were mainly inspired by mathematics and those who were influenced more by the empirical sciences originated in the field of analytical philosophy and dates back to the time of Pythagoras ([86]). The article 'The Formalism of Probability Theory in IR: A Foundation or an Encumbrance' by Cooper ([36]) is devoted to the internal struggle between these two strategies for information retrieval. Cooper started his article as follows:

> *'Some approaches to retrieval system design are strongly guided by theory. Others have little real theoretical underpinning, but are instead more experimental and ad hoc in character. Which is preferable? Obviously, theory-guidedness is a good thing if the theory leads to promising retrieval rules to try out. Good theories have inferential power, and inferential power can help minimize empirical floundering. However, having to stay within the constraints of a strict theoretical formalism can also impose costs and penalties. The true extent of these costs is not always fully recognized.'*

We highlight two famous information retrieval experiments, and discuss some of their observed limitations. We then present several current theoretical approaches for information retrieval.

2.1 Experimental approach

Traditionally the study of IR systems is purely experimental in the sense that it is 'based on tests one planned in order to provide evidence for or against a hypothesis' [1]. The experimental study is based on the paradigm as depicted in Figure 6. In an experimental approach an arbitrary information retrieval problem would be studied as follows. An information retrieval tool is proposed which could offer a solution for a typical information retrieval question. For instance: 'is it true that adding more documents to the document-base automatically leads to a higher recall in the context of this system'. Varying the setting of the parameters of the system and the number of documents, a collection of results is obtained. The results are studied and could possibly form an answer to the question under scrutiny. If such an answer is validated for several systems or in various settings, a hypothesis is formulated. When the hypothesis fits in a series of other related hypotheses, a theory is proposed.

In an experimental approach the comparison of two systems is formulated as: 'is system A better than system B with respect to C' and proceeds according to the method described above. A widely accepted manner of comparing experimental results of the retrieval performance of two IR systems is the study of the so-called *recall* and *precision* measures. These measures are used to conclude whether system A is to be preferred over system B or vice versa.

[1]Another interpretation, as for instance mentioned by Van Rijsbergen, is that experimental information retrieval is mainly carried out in a 'laboratory' situation ([78])

Figure 2.1: The experimental study

Recall and precision are two measures with as input the following two document sets: (1) the collection of documents which the user would judge to be relevant with respect to her information need, if she would be aware of all available documents (denoted as R_u), and (2) the collection of documents which an IR system retrieves according to the formulated query.

The main problem of recall and precision measures is the determination of the set R_u. In the next section we present two information retrieval tests. These tests show the construction of such sets. Important for the success of the evaluation is the ability to construct a suitable test collection. According to Hull ([59]), the fundamental components for a successful evaluation of a retrieval experiment are the availability of the following:

1. At least one document collection suitable for testing. The collection must include a number of queries and their relevance assessments. The relevance assessments determine sets of documents which are relevant, given the query (R_u);
2. A measure, based on the similarity ranking of relevant and non-relevant documents with respect to the query, that reflects the quality of the search; and
3. A valid (statistical) methodology for judging whether measured differences between retrieval methods can be considered statistically significant.

Next we present two of the more common information retrieval tests to give the reader an idea of how these tests are brought about.

2.1.1 Cranfield

One of the very first well-known information retrieval tests was the Cranfield test[2]. The test took place in the nineteen sixties. Strictly speaking, there were two Cranfield tests, both of which we present briefly.

2.1.1.1 Cranfield 1

The goal of the first Cranfield test was a comparative evaluation of four IR systems. It was a two-year project and the evaluation was focused on the indexing process rather than on the matching function. Four different indexing processes were examined:

[2]We recommend Cleverdon's article ([30]) for a detailed overview of the Cranfield tests.

1. Conventional Classification,
2. Alphabetical subject index,
3. Devised schedule of a facet classification,
4. Uniterm System of Coordinate Indexing.

The document collection consisted of 18,000 papers on aeronautical engineering. In addition, 1,200 queries based on a single document in the test collection were created (in our terminology, for each query R_u was a singleton set with the document based on the query in it). A search was successful if R_s contained the document used to create the query. The results showed that all four systems were 74 - 82% effective in retrieving the required document. The analysis was based on aspects such as time for indexing, learning process, and number of returns. The major point of criticism from the information retrieval community was that the construction of the search questions was based on documents in the test collection. So, in this case one cannot speak of a query which is formulated by someone with an information need.

2.1.1.2 Cranfield 2

The second Cranfield test kept the focus on the indexing process. The objective was to examine the effect of index languages, in isolation or in any possible combination, using recall and precision measures. The document collection was created in a way totally different from Cranfield 1.

Two hundred authors of recently published papers were asked to state in the form of a question the problem which their paper addressed. Furthermore, they had to add supplementary questions that arose in the course of their research. They were then requested to indicate, on a scale of 1 to 5, the level of relevance to each question of the references they had cited in their paper. Out of these references 1400 documents were selected and 279 queries were inspected by students and by the originator of the question.

The evaluation concentrated on the indexing of the documents. Here, the tests used a multi-stage process of indexing. An indexer manually identified the concepts in the document with one, two or three keywords. A weight in the range of 1 to 3 was assigned to each concept according to the importance for a particular document. Each single word was then listed with respect to the values of the concepts it occurred in. Finally the concepts were combined into themes. Given this indexing process, different representation languages were studied. Figure 2.1.1.2 is taken from Cleverdon ([30]) and presents the way the languages were obtained.

For each question it was inspected which documents would be retrieved and at which level (the latter aspect is important for the recall and precision ratio). The results were presented in the form of recall- and precision-curves.

Each question was indexed in all different representation languages. The results of the Cranfield 2 tests were not as evident as those from the Cranfield 1 tests. Salton ([88]) indicated for instance, that 'it is also the first evaluation project that produced unexpected and potentially disturbing results.' Among others, Salton was surprised by the result that an advanced indexing process (concept indexing) showed a worse performance than the simple indexing process (keywords indexing). After such a result there are two possible avenues: show that the test is not applicable and therefore the results are meaningless, or change the system in such a way that it will perform better with respect to the test. With respect to the former, it was remarked that the relevance assessments with respect to the corresponding queries might benefit the simple matching techniques at the expense of the more complex matching techniques.

2.1.2 TREC

In November 1992 the first TREC was held. TREC is the acronym of the annual Text REtrieval Conference. Its proceedings ([52], [54], [55]) contain papers about tests and their results. The tests are elements of the TREC programme, an officially organised activity, which has as its main goal to study different approaches to the retrieval of text for large document collections. At the moment,

Figure 2.2: Cleverdon's representation languages

TREC is the major experimental effort in the information retrieval field[3]. The test collection contains approximately one million documents (about 3 gigabytes of data). To compare the results obtained there is a detailed schedule that all the participants of TREC should obey. For TREC-4 the schedule is presented in Figure 8. As one can see, there is only one month of time to process

Jan	All potential participants should apply for a position in the tests. The program committee looks for as wide a range of text retrieval approaches as possible, and selects only those participants who are able to work with the large data collection.
Feb	For the accepted participants there are 3 gigabytes of information with queries and relevance judgements available (taken from previous TREC tests). With this collection they are able to train and improve the performance of their system.
May	A list of routing topics is distributed.
June	The test data is sent to the participants.
July	Fifty new test topics for ad hoc test are distributed.
Aug	The results should be submitted.
Oct	The evaluation process takes place.
Nov	The obtained results are presented during the TREC conference.

the queries. The very large amount of information makes it almost impossible to have manual interference in the indexing or matching process. Another point of interest is the distinction made between *routing* and *ad hoc* test topics. In the routing test mode, the situation is simulated in which the same questions are always being asked but new or more data is being searched. This task is similar to the one done by news clipping services or by library profiling systems ([53]). Then, the relevance decision depends on previous results, and thus a kind of *learning process* is involved.

[3]We recommend 'Overview of the Third Text REtrieval Conference (TREC-3)' ([53]) by Harman and 'Reflections on TREC' by Sparck Jones ([92]) for a detailed overview of TREC.

In an ad hoc test mode the document collection is fixed and the question is variable. The question 'is this document relevant for the query' is considered independently of previous results. This task is similar to how a researcher might use a library, where the collection is known but the questions which are likely to be asked are not ([53]). Therefore the time for processing ad hoc tests is much shorter than for the routing ones.

The document set in TREC is taken from several sources, individually varying and collectively varying in topics and genres, though with much news story material. While the relevance judgements in the Cranfield tests were done manually for the complete collection, for the TREC-collections this was practically impossible (3 gigabytes!). All TRECs have used the pooling method ([93]), which proceeds as follows: for each query and for each system the top 100 retrieved documents are merged in a pool[4], which is then shown to human assessors.

According to Harman ([53]), an important underlying assumption of this retrieval test is 'that the vast majority of relevant documents have been found and that documents that have not been judged can be assumed not to be relevant'.

The main results of TREC indicated not so much that one technique was shown to be significantly better than another one, but rather that individual retrieval systems were improving over time. Also, the achieved evaluation performance is an important result of TREC.

2.1.3 Reflections on the experimental approach

Since the early nineteen sixties experimental retrieval evaluations have been constructed. Very often the results of these evaluations were criticised by the experts (for an overview of various arguments see [88]). The next quotation from Cleverdon ([30]), one of the originators of the Cranfield tests, makes this particularly clear:

> *'The publication of the final report ([29]) attracted wide interest, caused considerable annoyance to the advocates of the different systems, and received some praise but much criticism.'*

Obviously some criticism was justified. A whole new discipline in the area of information retrieval developed, i.e., the evaluation of information retrieval evaluations. One typical example of such a meta-evaluation question is the following given by Hull ([59]):

> *'Why should experimental results based on collections with a very limited number of short documents on restricted topics be applicable to much larger and more variable documents collections that are found in real retrieval settings?'*

This intuitively acceptable concern was the underlying reason for the TREC-community to ensure that the amount of information in the test collection was enlarged.

The following list presents an overview of the main 'concerns' made by evaluation analysts ([10], [32], [59], [88]):

(i) The current measures, such as recall and precision, are not properly representing the *acuity* of an IR system because

- ▷ there is a retrieved and a unretrieved set of documents, without taking into account the possibility of an order of retrieval involving more than two classes;
- ▷ the utility factor of a document is not measured;
- ▷ returning a larger number of relevant documents is not always better: it may be that if the system highlights one relevant document this could be much more informative than returning a whole set;
- ▷ most often the measures are not dealing with interactive IR systems (such as relevance feedback systems).

[4]For TREC-1 and TREC-2, for TREC-3 it was 200.

(ii) The relevance assessments are not realistic, as the assessments are based on a formulated query rather than on an information need;

(iii) The evaluation must be based on knowledge of the complete set of relevant documents with respect to each query. This is hardly possible given the very large test collection;

(iv) Small document collections are not representative for large document collections (and vice versa).

To conclude, up to the time of writing IR systems are compared using statistical values such as recall and precision. We certainly perceive the utility of statistic values. However, to be able to make more strict statements concerning the qualities of one IR model when compared to another IR model, we feel that we should have more formal means of comparison at our disposal. Furthermore, to prove specific statements concerning the behaviour of IR systems, statistical tests are not adequate. There seems to be a definite need for a more formal characterisation of IR systems. In the following section we inspect theoretical approaches.

2.2 Theoretical approaches

One of the explanations of the word 'theory' in the Collins dictionary is *'a plan formulated in the mind only'*. This is certainly not what we have in mind. We prefer another description given in the dictionary, namely *'a set of hypotheses related by logical or mathematical arguments to explain a wide variety of connected phenomena in general terms'*. In this thesis the connected phenomena refer to the various stages of the retrieval process. There are various ways to study information retrieval in a theoretical way. We distinguish three approaches, namely

1. *Embedding*, which formalises an IR model that covers several other models.
2. *Categorisation*, which classifies different IR models based on a list of properties.
3. *Meta-theory*, in which a formalisation of a model and its properties in terms of a theory are presented.

These approaches do not necessarily exclude each other. One can propose a new model in terms of a theory and show how other models can be embedded. Or, one can describe properties in a theory and categorise existing IR models as to how they fulfil the described properties. In order to give the reader the essence of each approach, we discuss each of them briefly.

2.2.1 Embedding

In the case of embedding, different models are studied by mapping them to one model. We give one typical example, namely the *Inference Networks* as proposed by Turtle & Croft ([101]).

In the approach of Turtle & Croft different models are studied by mapping them to so-called *inference networks*. In an inference network retrieval model, retrieval is viewed as an 'evidential reasoning process in which multiple sources of evidence about document and query content are combined to estimate the probability that a given document matches a query' ([101]).

In an inference network retrieval model there are two directed, acyclic dependency graphs (networks), that are connected with each other. There is one graph for the document-base representation and one for the query. In the document network there are document nodes corresponding to abstract documents, text representation nodes representing information items of the document, and concept presentation nodes representing type information of the objects; the arrows in the document network represent the dependency relations. The query network is an 'inverted' directed acyclic dependency graph with a single leaf that corresponds to the event that an information need is met, and multiple roots that correspond to the concepts that express the information need. The query concept nodes define the mapping between the concepts used to represent the document collection and the concepts that make up the queries (the dotted line in Figure 2.2.1, shows where the mapping takes place). In the simplest case, the query concepts are constrained to be the same

Figure 2.3: Basic inference network

as the representation concepts, and each query concept has exactly one parent representation node (for instance, in Figure 2.2.1 the query node q_3 has as parent node t_7). In a more advanced network, the query concept may have more representation nodes (as depicted in Figure 2.2.1 where q_1 has as representation nodes t_5 and t_6).

By representing known models such as the boolean model, the vector-space model and the probabilistic model in terms of this network model, the authors showed that 'differences between current-generation retrieval models can be explained as different ways of estimating probabilities in the inference network' ([101]). By tuning or adjusting the network they are aiming to achieve the best retrieval performance. These results can then be used for proposing a *new* IR system. For instance, the INQUERY retrieval system ([22], [23] is a system based on the network model using the results from the investigation of several different models.

2.2.2 Categorisation

One typical example of an evaluating process based on categorisation is the work of Blair ([10]). Blair proposes that, in order to improve information retrieval, a good theory of document representation is needed which is primarily based on language and meaning. In order to study different approaches for modelling information in information retrieval, twelve principal formal models are defined. For example Blair's 'Model 12' is presented as follows:

Model 12 (Weighted Thesaurus)

I. Requests are single terms.

II. Index assignments: a set of one or more descriptors.

III. Documents are either retrieved or not.

IV. Retrieval rule: the request descriptor is looked up in a thesaurus (on-line) and semantically related descriptors above a given cut-off value (weight) are added (disjunctively) to the request descriptor. The cut-off value could be given by the inquirer.

After presenting a model, advantages and disadvantages are summed up. For instance, one of the advantages of model 12 is that it provides the user with a list of terms which are semantically related to those in the thesaurus, which is especially useful in systems with uncontrolled vocabularies ([10]).

As mentioned in Section 2.1.3, Blair is one of the critics of an evaluation based on statistical estimations only. The great expense of such evaluations (in time and cost) may prevent them from being performed very often (such as the scheduling of TREC, which covers almost a whole year!). Blair compares information retrieval to astronomy and quantum physics where experiments are expensive but a theoretical formalism exists that can be used to advance theoretical understanding of these disciplines independently from empirical verification. Blair states:

'Information retrieval would benefit greatly from the development of a similar theoretical formalism that would permit at least some of its advances to be done independently from empirical validation.'

His book ([10]) is focused on the use of language and its representation. Our approach is more directed towards the relevance decision. However we believe that both approaches could be used in tandem in order to study IR models.

2.2.3 Meta-theory

In a meta-theoretical approach, information retrieval is viewed in terms of a theory T. The model and its properties are formalised and explained in terms of the chosen theory. Typically, there are two kinds of arguments for choosing a specific theory T. Firstly, with theory T we should be able to formalise existing IR models. Secondly, some property P which is shown to be very important for information retrieval purposes should be well-covered by the theory T.

Next, we present the two main types of theories as used in information retrieval, namely those based on probability theory and those based on logic.

2.2.3.1 Probability Theory

One main direction in theoretical information retrieval research is based on probability theory (for an overview of probabilistic IR models see [45]). Typically, in a probabilistic retrieval model one estimates the probability that a user decides that a document is relevant given a particular document and query, denoted as $Prob\left(\text{Relevant}\,\middle|\,\text{Document}, \text{Query}\right)$. Here, an information retrieval theory is centred around the statistical uncertainty assumptions involved in information retrieval. At a meta-level the information retrieval theory could be studied in terms of probability theory.

For instance Cooper ([35], [37] inspects some probabilistic assumptions which have consisted of various combinations of the three statistical independence assertions $I1$, $I2$ and $I3$ defined as follows: In these formulae A and B are properties of documents or users, depending on the focus of

$$\begin{array}{lrcl}
\text{I1.} & Prob(A,B) & = & Prob(A) \times Prob(B) \\
\text{I2.} & Prob\left(A,B\,\middle|\,R\right) & = & Prob\left(A\,\middle|\,R\right) \times Prob\left(B\,\middle|\,R\right) \\
\text{I3.} & Prob\left(A,B\,\middle|\,\overline{R}\right) & = & Prob\left(A\,\middle|\,\overline{R}\right) \times Prob\left(B\,\middle|\,\overline{R}\right).
\end{array}$$

the study. The character R denotes the event of relevance. Assumption I1 reflects the assumption that A and B are independent, which is often assumed to be true in information retrieval for document and information need properties. Assumptions I2 and I3 are adopted by probabilistic model developers ([83]). In combination with well-known properties of conditional probabilities, such as $Prob\left(A\,\middle|\,B\right) = Prob(A,B)/Prob(B)$, or, phrased differently, $Prob(A,B) = Prob\left(A\,\middle|\,B\right) \times Prob(B)$, assertion $I1$ implies that properties A and B are absolutely independent ($Prob(A) = Prob\left(A\,\middle|\,B\right)$ and $Prob(B) = Prob\left(B\,\middle|\,A\right)$. Assertion $I2$ and $I3$ express that A and B are independent given relevance or its absence. Stated otherwise, the fact that A is relevant does not influence the fact that B is relevant and vice versa.

On a meta-level Cooper studies the contradiction of elementary laws of probability theories in this information retrieval setting. For instance, using the binary independence retrieval model ([45]), A is the occurrence of a specific document d, and B the occurrence of a specific query q. Then we estimate the probability that document d is judged relevant with respect to query q. Let:

$$\begin{aligned}
Prob(A) &= Prob(B) = Prob(R) = 0.1 && (2.1)\\
Prob\left(R\,\middle|\,A\right) &= 0.5 && (2.2)\\
Prob\left(R\,\middle|\,B\right) &= 0.5 && (2.3)
\end{aligned}$$

We can calculate that $Prob(A,B,R) > Prob(A,B)$ as follows. According to Cooper,

$$Prob(A,B) = Prob(A) \times Prob(B) = 0.01$$

and

$$\begin{aligned}
Prob(A,B,R) &= Prob\left(A,B\,\middle|\,R\right) \times Prob(R) && (2.4)\\
&= Prob\left(A\,\middle|\,R\right) \times Prob\left(B\,\middle|\,R\right) \times Prob(R) && (2.5)\\
&= (Prob\left(R\,\middle|\,A\right) \times Prob\left(R\,\middle|\,B\right) \times Prob(A) \times Prob(B))/Prob(R) && (2.6)\\
&= 0.025 && (2.7)
\end{aligned}$$

which is in conflict with the assumption that a removal of an event always leads to an increase of the probability value. To circumvent this kind of problem, Cooper ([37]) suggested a reformulation of the underlying assumptions in terms of probability theory. Cooper concludes

> *'When this is done, some models are found to be not only different in character but more realistic than had been supposed, for the true modelling assumptions are weaker and more plausible than the ones thought to be in force.'*

A meta-theory based on probability theory inspects IR models in terms of their uncertainty calculation. The probability calculus is the first-class citizen of this approach. For instance, one can analyse different relevance-functions in terms of a probabilistic inference model as shown in the inference network of Turtle & Croft. Without proposing a new model, Wong & Yao ([107]) showed that known models such as the boolean, fuzzy set, vector-space, and probabilistic models are special cases of the probabilistic inference models.

2.2.3.2 Logic

The first-class citizens of a logical theory are the inference process and the modelling of information (for an overview of logical IR models see [64]). If a formula φ can be inferred from a formula ψ in a logic L, this could imply that the information represented by ψ is relevant with respect to the information represented by φ. Cooper ([34]) originated the logical approach by viewing a part of the relevance decision as a logical inference process.

Van Rijsbergen suggested that if we are able to infer *relevance* in a logical sense, maybe a particular logic could be used for modelling information retrieval ([79], [80]). In a logical theory the study of IR models proceeds by inspecting the logical properties of the retrieval process.

One example of using logic in order to analyse information retrieval is given by Chiaramella & Chevallet ([27]). They study the semantics of the implication as used in IR models such as the boolean model. In terms of the underlying model they are able to propose some extensions based on their logical analysis. The authors conclude that a logical approach 'provides a better way of encompassing the fundamental aspects of information retrieval' ([27]).

Their conclusion was based on the following three observations. Firstly, the expressive power of the logical model. Secondly, the new insights gained from studying existing models in a logical setting. Lastly, the necessity of coping with the fast change of information retrieval paradigms as presented in Chapter 1.

A theory could also be a combination of two theories, one covering property P very well, the other property Q. For example, in the Logical Uncertainty Principle ([80]), the combination of a logical and a probabilistic approach is presented. The Logical Uncertainty Principle is founded on the idea that, if an IR system cannot deduce that a document d is about a query q given a logic $\mathcal{L}$, we have to add information to the data set[5] until we can determine the aboutness relation between the document and the query. The strength of aboutness can be associated with the measure of uncertainty $Prob(d \text{ about } q))$ which is based on how much information is added. For example, assume that d is indexed with a logical formula t_1, and that aboutness is defined in terms of classical logic, i.e., if $\vdash d \rightarrow q$ then d about q. Then we cannot derive that d is about $t_1 \vee t_2$. In this particular case we have to add t_2 to d in order to derive aboutness. Applying the Logical Uncertainty Principle one could for instance calculate the uncertainty of t_2 in order to measure $Prob(d \text{ about } t_1 \vee t_2)$.

A typical example of a model that combines a logical and probabilistic approach is shown in the article 'Towards a Probabilistic Modal Logic for semantic-based Information Retrieval' by Nie ([74]). Here he presents an integration of semantic inference (based on a Possible World semantics) and probabilistic measurement based on the Logical Uncertainty Principle.

[5]Van Rijsbergen is not explicit about what we could understand by the concept of a data set. It could be a document d, a query q, or a consulted knowledge-base.

Chapter 3

The IR Problem

The Information Retrieval paradigm is about a person (physical or not) having a need for information, and a set of information objects from which this need is to be satisfied. In this section, we provide general models to formalize the concept of information need.

A searcher has a need for certain information. This need may have both qualitative and quantitative aspects. In the context of an information collection, the searcher tries to formulate this need in terms of this collection, translating the need for information into a need for information objects (documents). The starting point of our discussion will be a searcher having a need for documents in the collection.

The information collection may be a stable (fixed) collection, in which case an Information Retrieval System may preprocess the collection for optimal disclosure. The intention of this preprocessing is that the Information Retrieval System gets an impression of the contents of each document. This calls for a contents description mechanism.

The collection may be a (continuous) stream of documents (like a newsgroup on the Internet). For each next document, the Information Retrieval System has to decide if it will be interesting for the searcher. In this case there can be no preprocessing of all documents. Descibing document contents then is either based on some universal system for contents description, or the system should try to build such a system incrementally when new documents arrive. This is also the situtation on the world wide web, an extremely large and volatile collection of documents.

3.1 What the searcher wants

It is the main objective of an Information Retrieval System to provide an effective disclosure mechanism for a collection $\mathcal{O}$ of n information objects. Suppose a searcher is interested in some of the information objects. A most simple approach is to model this interest as a partial order on the collection. This partial order could be as weak as partitioning the collection only in a set of relevant and a set of irrelevant documents. The task of the Information Retrieval System then can be described as producing a (total) ordering which ressembles as best as possible the interest of the searcher. This model is referred to as the *comparative model* for the *information need* of the searcher. **information need**

Example 3.1.1

Consider a collection of chess playing people. The organizer of a championchip would like to select the k best players for participation. A binary comparison operator, being the basis for a partial order of players, is playing a game of chess. Note that this operator is not *associative.*

Selecting the k best depending on a comparison operator is known as the selection problem. **selection problem**

Usually not all information objects have an equal importance for the searcher. The *information need* of a searcher therefore is modelled as a function N: **information need**

$$N : \mathcal{O} \mapsto [0, 1]$$

weighted model

We call $N(x)$ the need for information object x. This function can be seen as a subjective assignment of relevance to all documents. This is referred to as the *weighted model* for information need. Note that the comparative model is obtained from the weighted model by comparing documents according to their relevance.

A common simplification is a discrete information need, where each information object either is relevant or not:

$$N : \mathcal{O} \mapsto \{0, 1\}$$

discrete model

This *discrete model* is the traditional basis for information retrieval. The discrete model can be seen as a special case of the weighted model. In the discrete model, the information need can be interpreted as the so-called characteristic function (see appendix B.1) associated with the subset of all documents in which the searcher is interested.

incremental model

More sophisticated models to describe the information need may be employed. As an example, we mention the *incremental model*. In this model, it is assumed that the need for more documents is influenced by what the searcher already has retrieved from the archive. This can be modelled as a function

$$I : \wp(\mathcal{O}) \mapsto (\mathcal{O} \mapsto [0, 1])$$

(or, equivalently, as a function $I : \wp(\mathcal{O}) \times \mathcal{O} \mapsto [0, 1]$) where $I(S, x)$ is interpreted as the increment in searcher satisfaction when document x is presented after set S has already been retrieved to the searcher. The function I is also referred to as the *increment function*. Note that the weighted model can be obtained from the incremental model by defining $N = I(\varnothing)$. Thus, the need for a document is taken as the satisfaction obtained without any prior knowledge.

The incremental model is especially useful for (very) dynamic and distributed archives, such as the World Wide Web. Firstly, the increment function allows for real-time calculation. This is in contrast with approaches that try to cluster the retrieval result before presenting the clusters to the searcher. Clustering is only possible after all documents have been obtained. Secondly, for distributed archives recall is not useful as a measure for retrieval quality. We will propose a quality measure which is based on total searcher satisfaction, bypassing the need to have global knowledge of the collections involved.

3.2 How this can be formulated

It is the responsibility of the searcher to properly communicate the information need to the Information Retrieval System. For this purpose, the Information Retrieval System provides a retrieval language $\mathcal{Q}$. Suppose the searcher has formulated information need N as query q. The Information Retrieval System will then match the information objects against query q, and try to approximate as best as possible information need N, as formulated in query q. The matching function thus is a function:

$$\mathit{Match} : \mathcal{Q} \mapsto (\mathcal{O} \mapsto [0, 1])$$

For each q, $\mathit{Match}[q]$ is a function that assigns to each document of the archive its relevance, as estimated by the Information Retrieval System on the basis of query q. In order to be able to handle this approximation problem, the uncertain translation of the information need N into query q has to be eliminated. For this purpose, we assume that query language $\mathcal{Q}$ has assigned a semantical interpretation via the function *Norm*.

$$\mathit{Norm} : \mathcal{Q} \mapsto (\mathcal{O} \mapsto [0, 1])$$

normative information need

The function *Norm* is called the *official interpretation* of the query language $\mathcal{Q}$ in information base $\mathcal{O}$. $\mathit{Norm}[q]$ is called the *normative information need* associated with query q. The normative information need is an objective relevance assignent. Usually this function has no formal definition, but is investigated manually ad hoc by a panel of domain experts for a well-choosen set of test queries. In conventional database systems, the query language will be such that there exists a query for each information need which can be derived from the database. In Information Retrieval Systems a query language may not have this property.

3.3 How the result is obtained

The objective of an Information Retrieval System is to design a function *Match* such that $Match[q]$ is a good approximation of the normative information need $Norm[q]$ for all queries q. The quality of the approximation can be expressed as the distance $\Delta_{\mathcal{O}}(Match[q], Norm[q])$ between the functions $Match[q]$ and $Norm[q]$. The distance function can be defined is several ways. In this book, we presume a generic distance function Δ_X for functions in $X \mapsto [0,1]$ which for example can be defined by:

$$\Delta_X(f,g) = \left(\sum_{x \in X} |f(x) - g(x)|^p \right)^{1/p}$$

for some $p \geq 1$. We will write Δ when the underlying domain X is obvious from the context. The essentials for a metric space are layed down in the following axioms ([95]):

$$\begin{array}{ll} \text{MS1:} & \Delta(x,x) = 0 \\ \text{MS2:} & \Delta(x,y) = \Delta(y,x) \\ \text{MS3:} & \Delta(x,y) \leq \Delta(x,z) + \Delta(z,y) \end{array}$$

Axiom MS3 is referred to as the triangle inequality. If we interpret functions in $X \mapsto [0,1]$ as vectors then this distance measure is called the p-norm, denoted as $|f|_p$. For $p = 2$, this distance corresponds to the well-known Euclidian distance.

The intention of the searcher is to find a query q which minimizes $\Delta(N, Match[q])$. Rather than random probing, the searcher will try to find an appropriate query, which is evaluated as good as possible by the IR system. This process of separation of concerns can be understood from the triangle inequality:

$$\underbrace{\Delta(N, Match[q])}_{\textit{hopefully small}} \leq \underbrace{\Delta(N, Norm[q])}_{\textit{learning and support}} + \underbrace{\Delta(Norm[q], Match[q]))}_{\textit{quality IR system}}$$

The matching quality of the Information Retrieval System, the *system quality*, is defined as the functional distance **system quality**

$$\Delta_{\mathcal{Q}\times\mathcal{O}}(Match, Norm)$$

Let Q be a set of test queries, then this quality is estimated as

$$\frac{1}{m} \sum_{q \in Q} \Delta_{\mathcal{O}}(Match[q], Norm[q])$$

where m is the number of test queries.

Approximating the information need N is a fairly complex task. If the Information Retrieval System is only to retrieve documents in the order in which they are mostly wanted, a more modest system requirement is to rank the documents as good as possible according to the ranking induced from the information need N. A typical aproach is to assign each document an (objective) relevance estimate, and to sort the documents subsequently according to these estimates.

3.4 What a searcher should know

As stated before, it is the responsibility of the searcher to find an optimal formulation of the information need in question. The searcher thus has to find a query q which minimizes the expression:

$$\Delta_{\mathcal{O}}(N, Norm[q])$$

On the one hand, this requires a *learning process* of the searcher. The goal of this learning process is knowledge of the function *Norm*, which depends both on the archive $\mathcal{O}$ and its query language $\mathcal{Q}$. Usually, the language $\mathcal{Q}$ strongly depends on the collection $\mathcal{O}$. For example, $\mathcal{Q}$ might be constructed from keywords that are obtained from $\mathcal{O}$, using general construction rules for building queries. On the other hand, the Information Retrieval System may try to *support the process of query* **learning process** *formulation*. This can be compared to the situation when a casual visitor enters an exclusive bakery business. The shop assistant will try to sell as much as possible products (by investigating) in which the visitor is interested. Two main approaches can be attempted:

support the process of query formulation

1. *offer the searcher alternatives*
   This corresponds, in terms of the bakery, with a shop assistant that starts with suggesting general product terms (matching all products, or, in Infomation Retrieval terms, having a high recall), and then subsequently offering the client more detailed product descriptions (lowering recall, adding precision). The shop assistant thus provides the client with a guided navigation through the query space, in search for a proper formulation of the information need. The underlying cognitive assumption of a searcher may be formulated as ([16]):

   I don't know what I'm looking for, but I'll know when I find it

2. *provide a feedback mechanism*
   In this case, the shop assistant lets the client have a taste of the products that seam to be interesting, or products that give a goos impression of the product range of the bakery. The feedback information of the client gives the shop assistant a better impression of what the client wants.

a-priori support

query by navigation

The first approach (*a-priori support*) will be suitable for a searcher with (some) knowledge of the retrieval language, but unfamiliar with the actual archive. An example of this mechanism is *query by navigation* which will be introduced in a later chapter. A main advantage is that the system will offer the searcher productive queries only. This avoids the problem of finding the words and phrases by which the relevant information objects can be discriminated from the others. Blair ([11]) has stated this as follows:

It is not a simple matter for users to foresee the exact words and phrases that will be used in the documents they will find useful, and in only those documents.

a-posteriori support

query by example

relevance feedback

The second approach (*a-posteriori support*) fits best searchers that are unfamiliar with both retrieval language and archive. The will have no good impression of the effect of query terms. By confronting them with the result of initial queries, the query formulation process will be like *query by example*. An example of this approach is called *relevance feedback*. Searchers outside these categories probably will not need special support of the Information Retrieval System during query formulation.

3.5 Abstracting from the archive

unshaped Information Retrieval

full text retrieval

An *unshaped Information Retrieval* system is a structure $\langle \mathcal{Q}, Norm, Match, \mathcal{O} \rangle$. Typical for unshaped retrieval systems is that the information object is involved as a whole in the matching process. An example is a *full text retrieval* system, where queries are simply a list of keywords, and matching is done by trying to locate keywords textually within the document.

A strong point of unshaped retrieval is that matching can be done very precisely, as the whole document is involved in the matching process. Unshaped retrieval thus provides the opportunity for optimal matching. A drawback is that matching will be rather time consuming. Full text retrieval systems anticipate on this problem, at the cost of a significant lower precision. Spelling variations, synonyms, etc. may cause the recall of matching to be rather disappointing.

extensional aspects

intentional aspects

A better approach is to introduce document descriptions that provide a (short) approximation (impression) of the document. Some characteristics of a document are related with external aspects. Examples are: author(s) of a paper, title of a paper, style of art object. These aspects are referred to as *extensional aspects*. Extensional aspects are easily modeled in conventional relational database systems. Information disclosure on the basis of extensional aspects is handled by typical languages as LISA-D and SQL. Other characteristics are related with the subject or intention of the information object. These aspects are referred to as *intentional aspects*. If intentional aspects are derived automatically from the document, then the document contains more information than its characterization. If these aspects are added by experts, then the characterization may contain information that can not be derived (automatically) from the document. Conventional database systems are not capable of handling internal aspects adequately.

As document descriptions are short, the matching process will be much faster than in the case of unshaped retrieval. There is a trade-off between efficiency and precision, as document descriptions will not cover all aspects of the contents of documents.

Let $\mathcal{C}$ be the language that is used for document characterizations. The elements of $\mathcal{C}$ are called *descriptor*s. The characterization of a document provides the relevance of each descriptor from $\mathcal{C}$ for the document in question. The characterization of documents thus is layed down by:

descriptor

$$\chi : \mathcal{O} \mapsto (\mathcal{C} \mapsto [0, 1])$$

As a consequence, $\chi(x)$ is a function that assigns relevancy to each descriptor in $\mathcal{C}$; $\chi(x)(d)$ quantifies in what degree document x is about topic d. The relation between queries and characterizations is independent of the actual archive:

$$\mu : \mathcal{Q} \times (\mathcal{C} \mapsto [0, 1]) \mapsto [0, 1]$$

The expression $\mu(q, \chi(x))$ is the degree of correspondence between query q and characterization $\chi(x)$. Note that the outcome of this function is the same for documents with equal characterization. The matching function then can be defined by:

$$Match[q](x) = \mu(q, \chi(x))$$

An *Information Retrieval System* is introduced as a structure $\langle \mathcal{Q}, Norm, \mu, \mathcal{C}, \chi, \mathcal{O} \rangle$. Such systems can be considered outside the context of a specific archive. The restriction $\langle \mathcal{Q}, Norm, \mu, \mathcal{C} \rangle$ is referred to as a *disclosure mechanism*.

Information Retrieval System

disclosure mechanism

3.5.1 Hit list

The characterization function χ is set up as a characterization of documents in terms of a set of descriptors. Another way to view this function is as a characterization of descriptor in terms of documents. It could be argued that this descriptor characterization is the meaning of a desciptors in the context of some document collection. The descriptor characterization function $\mathsf{HitList}$ is also referred to as the hit list of a term, or as the inverted list asociated with he term.

$$\mathsf{HitList} : \mathcal{C} \mapsto (\mathcal{O} \mapsto [0, 1])$$

defined as:

$$\mathsf{HitList}(c)(x) = \chi(x)(c)$$

As a consequence, the semantical interpretation of descriptors can be defined as:

$$Norm(q) = \mathsf{HitList}(q)$$

in case descriptors can also be used as queries.

3.6 Structured characterization languages

A next refinement is adding structure to the characterization language $\mathcal{C}$. This structure might for example employ a thesaurus structure. Dependencies between elements from $\mathcal{C}$ will result in relevance dependencies. This leads to a calculation schema for characterizations.

WordNet (www.cogsci.princeton.edu/wn) is an on-line lexical reference system. WordNet is basically an online thesaurus. WordNet is different than other online or print thesauri because it allows the user to search conceptually rather than alphabetically. Through natural language searching, a user can enter almost any term that is part of the English language and find related terms. A searcher can find broader terms, narrower terms, and other related terms. WordNet was developed in 1985 in the Cognitive Science Laboratory at Princeton University. There are currently some 118,000 words and over 90,000 word senses in the database. The Cognitive Science Laboratory involves the work of psychologists, computer scientists, linguists, philosophers, and electrical engineers all working together to create computers that more closely model the human thought process.

WordNet has been described in a collection of 5 papers (ftp.cogsci.princeton.edu/pub/wordnet/5papers.pdf). We discuss the main concepts of these papers.

3.6.1 The lexical matrix

word forms
word meaning

A main task of a lexicon is to describe lexical semantics, or, the association between *word forms*, i.e. the actual representation of a word, and *word meaning*. Let $\mathcal{M}$ be the set of all meanings, and $\mathcal{F}$ the set of all word forms. Then the lexical matrix LexMat is a relation over $\mathcal{M} \times \mathcal{F}$. We will write $w\mathsf{Means}m$ as an alternative notation for $\langle m, w \rangle \in \mathsf{LexMat}$. Two special cases are considered. The first case is that a word form has associated several meanings by the lexical matrix:

Definition 3.6.1

polysem

The word form w is called a called a polysem *if it has more than one meaning:*

$$\mathsf{PolySem}(w) \equiv \exists_{m_1 \neq m_2} \left[w\mathsf{Means}m_1 \wedge w\mathsf{Means}m_2 \right]$$

On the other hand, different word form may be associated to the same meaning:

Definition 3.6.2

Word forms w_1 and w_2 are synonyms of each other if they can have the same meaning:

$$\mathsf{Synonym}(w_1, w_2) \equiv \exists_m \left[w_1\mathsf{Means}m \wedge w_2\mathsf{Means}m \right]$$

In this approach, a meaning is an abstract object which basically is identified by the ways it can be communicated. As a consequence, meanings can be identified with sets of word forms. For convenience, we introduce SynSet as a shorthand notation for the powerset of word forms:

$$\mathsf{SynSet} \equiv \wp(\mathcal{F})$$

glos

In natural language it may occur that there are no appropriate sysnonyms available to differentiate menaniings from each other. In such cases, short additional descriptions (referred to as *glos*ses) are added to the set of synonyms associated with that meaning. Let $\mathcal{G}$ be the set of natural language expressions that can be used as glosses. Then meanings are identified with??

$$\mathcal{M} \equiv \mathsf{SynSet} \times \mathcal{G}$$

The lexical matrix is not allways sufficient to denote the meaning

3.6.2 Semantical relations

reciprocated

Semantical relations are relations between meanings, and thus relations over SynSet. It is typical for a cluster of semantic relations that they can be *reciprocated*. This indicates that if R is a semantical relation, and $\langle x, y \rangle \in R$, then there is also a semantical relation S such that $\langle y, x \rangle \in S$.

3.6.2.1 Synonymy

A mathematical oriented definition for similarity of meaning is attributed to Leibniz:

> Two expression are synonymous if the substitution of one for the other never changes the truth value of a sentence in which the substitution. is made.

For natural language this definition is too strong, being synonymous is relative to a context:

Definition 3.6.3
Two expression are synonymous in a linguistic context C if the substitution of one for the other in C does not alter the truth value.

In natural language the situation is even more complex, as.......

3.6.2.2 Antonymy

3.6.2.3 Hyponymy

3.6.2.4 Meronymy

3.6.2.5 Morphological Relations

An example is a thesaurus structure for $\mathcal{C}$ in which descriptors are related via a *synonym* (denoted as SYN) relation. This relation is an equivalence relation: **synonym**

1. *reflexive*: $x \,\textsc{syn}\, x$
2. *symmetric*: $x \,\textsc{syn}\, y \Rightarrow y \,\textsc{syn}\, x$
3. *transitive*: $x \,\textsc{syn}\, y \wedge y \,\textsc{syn}\, z \Rightarrow xsynz$

Another example is the *is-a* (denoted as ISA) relation. For example, student ISA person. Thus there will be a correspondence between what is relevant for a student, and what is relevant for a person. The ISA -relation partially oders the descriptors: **is-a**

1. *anti-reflexive*: $\neg x \,\textsc{syn}\, x$
2. *transitive*: $x \,\textsc{syn}\, y \wedge y \,\textsc{syn}\, z \Rightarrow xsynz$

These relations distribute as follows:

1. $x \,\textsc{isa}\, y \wedge x \,\textsc{syn}\, z \Rightarrow z \,\textsc{isa}\, y$
2. $x \,\textsc{isa}\, y \wedge y \,\textsc{syn}\, z \Rightarrow x \,\textsc{isa}\, z$

3.7 A probabilistic view

Another way to look at the inforation retrieval problem is that a system is supposed to be able to prove the correctness of the retrieval result. For that purpose, the query q is considered to be the evidence for relevancy. That raises the question how likely a document is to be relevant given this peace of evidence for the information need of the searcher.

Chapter 4

Quality measures

In this chapter the quality of the approximation $\mathit{Match}[q]$ for the normative information need $\mathit{Norm}[q]$ is quantified (for each query q). This quality measure is an objective quality measure as it does not depend on the information need of a specific searcher. For a specific searcher, it will make more sense to relate the retrieval result $\mathit{Match}[q]$ to the original information need N, after having translated this information need into query q. In this chapter, first a set of primitive measures is introduced to evaluate a single query. Relating these measures leads to precision-recall graphs and fallout-recall graphs. Such graphs provide an elegant impression of the performance of the Information Retrieval system. Finally, we discuss the expected number of documents that the searcher has to inspect before having found some number of relevant documents.

4.1 Standard measures

4.1.1 The basic functions

We start with the double discrete case, where both query and characterization have a $\{0, 1\}$ range. Such functions are called *characteristic functions*. This allows us to consider both $\mathit{Norm}[q]$ and $\mathit{Match}[q]$ as subsets of $\mathcal{O}$. We consider the situation when the question q has resulted in retrieval of the set $\mathit{Match}[q]$ of documents, while $\mathit{Norm}[q]$ was intended. Relevant documents that have actually been retrieved are called *hits*:

characteristic functions

hits

$$\mathit{Hits}[q] = \mathit{Match}[q] \cap \mathit{Norm}[q]$$

The remainder of the retrieved documents thus are *noise* for the searcher:

noise

$$\mathit{Noise}[q] = \mathit{Match}[q] - \mathit{Norm}[q] = \mathit{Match}[q] \cap \overline{\mathit{Norm}[q]}$$

where $\overline{X}$ denotes set complementation with respect to the archive: $\overline{X} = \mathcal{O} - X$. Relevant documents which are omitted in the retrieval result (its *omission*) are:

omission

$$\mathit{Omission}[q] = \mathit{Norm}[q] - \mathit{Match}[q] = \mathit{Norm}[q] \cap \overline{\mathit{Match}[q]}$$

This is summarised in the following table:

		Relevant: $\mathit{Norm}[q]$	
		yes	no
Retrieved: $\mathit{Match}[q]$	yes	$\mathit{Hits}[q]$	$\mathit{Noise}[q]$
	no	$\mathit{Omission}[q]$	

The quality of the retrieval result can be measured via a number of traditional measures.

Firts we consider an example. Suppose $\{1..100\}$ is a collection of documents, from which we want to retrieve all multiples of 10 ($N = \{10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$). So, $\frac{10}{100}$ of the collection are relevant documents (this is referred to as the *generality*).

In order to meet our information need, the queries q_1 and q_2 are tried. The exact nature of these queries is not relevant in this example, only that these questions lead to the following retrieval results:

1. q_1: $\{10, \ldots, 34\}$
2. q_2: $\{10, 20, 30, 40, 50\}$

We assume these queries to be a proper reflection of the information need, i.e. $Norm[q_1] = Norm[q_2] = N$. Then the hit and noise results of these queries are:

1. $Hits[q_1] = \{10, 20, 30\}$, $Noise[q_1] = \{11..19, 21..29, 31..34\}$
2. $Hits[q_2] = \{10, 20, 30, 40, 50\}$, $Noise[q_2] = \varnothing$

For query q_1 we thus have:

		Relevant: $Norm[q]$	
		yes	no
Retrieved: $Match[q]$	yes	$\{10, 20, 30\}$	$\{11..19, 21..29, 31..34\}$
	no	$\{40, 50, 60, 70, 80, 90, 100\}$	– remaining documents –

As a consequence, only a fraction $\frac{3}{10}$ of the wanted documents have been retrieved (this will be referred to as the *recall* of q_1). In this respect, the query q_2 performs better with a recall of $\frac{5}{10}$.

Another difference between the queries is the amount of noise they produce. In the retrieval result of query q_1 only the fraction $\frac{3}{25}$ of the 25 documents is relevant (the *precision* of q_1). The precision of query q_2, on the other hand, amounts to 1, as this query produces no noise.

Another way to get an impression of the quality of the retrieval result is to consider the fraction of the irrelevant documents which have been retrieved. Note that 90 documents of the collection are irrelevant. Thus, $\frac{22}{90}$ of the irrelevant documents have been retrieved by q_1. This is denoted as the *fallout* of q_1. The fallout of query q_2 thus amounts to 0, as no irrelevant documents are retrieved.

In the general case, both query and characterization are considered as functions that assign to each descriptor a relevance value. Therefore we generalize as follows (see the transformation rules of table B.1):

$$\begin{aligned} Hits[q] &= Match[q] \bullet Norm[q] && (4.1)\\ Noise[q] &= Match[q] \bullet \overline{Norm[q]} && (4.2)\\ Omission[q] &= Norm[q] \bullet \overline{Match[q]} && (4.3) \end{aligned}$$

Note that a generalization of the formulation in terms of set difference is not felt to be appropriate.

Exercise 4.1.1

Show that the expressions $f - g$ and $f \bullet \overline{g}$ are not *equivalent.*

4.1.2 The measures

These quantities form the basis for the introduction of the conventional quality measures for in-
recall formation retrieval. The first measure, *recall*, quantifies the proportion of relevant documents that have been retrieved.

$$Recall(q) = \frac{|Hits[q]|}{|Norm[q]|}$$

if $|Norm[q]| \neq 0$, while $Recall(q) = 0$ otherwise. The proportion of retrieved documents that are
precision relevant is referred to as the *precision* of the retrieval action. This can be defined as:

$$Precision(q) = \frac{|Hits[q]|}{|Match[q]|}$$

if $|Match[q]| \neq 0$, and $Precision(q) = 0$ otherwise. The proportion of irrelevant documents that are retrieved is called the *fallout*, and is defined as:

fallout

$$Fallout(q) = \frac{|Noise[q]|}{\left|\overline{Norm[q]}\right|}$$

if $\left|\overline{Norm[q]}\right| \neq 0$, while $Fallout(q) = 0$ otherwise. Finally, the *generality* of a query is the proportion of documents that relevant:

generality

$$Gen(q) = \frac{|Norm[q]|}{|\mathcal{O}|}$$

The following property relates these standard measures for Information Retrieval systems.

Lemma 4.1.1

$$Precision(q) = \frac{Gen(q) \times Recall(q)}{Gen(q) \times Recall(q) + (1 - Gen(q)) \times Fallout(q)}$$

We postpone the proof of this property.

4.1.3 A probabilistic interpretation

First we present a probabilistic interpretation of the Information Retrieval problem. Suppose a searcher tries to find a relevant document (i.e. a document from $Norm[q]$) by randomly taking a document from the collection. The probability of being succesfull amounts to the generality of the interest of the searcher:

$$Gen(q) = Prob(Norm[q])$$

where $Prob(A)$ is the probability of a random element from $\mathcal{O}$ also to be an element of A, thus, in terms of functions: $Prob(A) = |A| \, / \, |\mathcal{O}|$. The Information Retrieval system tries to improve on this strategy by selecting a well chosen subset $Match[q]$ from the collection, to which the searcher may restrict the search process. The precision of this result equals the probability of randomly selecting a relevant document from this sub-collection:

$$Precision(q) = Prob\big(Norm[q] \,\big|\, Match[q]\big)$$

The conditional probability $Prob\big(X \,\big|\, Y\big)$ provides the probability of an element from Y to be also to be an element of X. This is formally introduced as the fraction $Prob(X \cap Y)/Prob(Y)$. The recall of the result amounts to the probability of an intended object to be retrieved:

$$Recall(q) = Prob\big(Match[q] \,\big|\, Norm[q]\big)$$

The fallout of the result amounts to the probability of an intended object to be missed:

$$Fallout(q) = Prob\big(Match[q] \,\big|\, \overline{Norm[q]}\big)$$

The proof of lemma 4.1.1 can now be formulated as:

Proof:

In this proof some general properties of probability theory are applied. First we note:

$$\begin{aligned} Gen(q) \times Recall(q) &= Prob(Norm[q]) \times Prob\big(Match[q] \,\big|\, Norm[q]\big) \\ &= Prob(Match[q] \cap Norm[q]) \end{aligned}$$

Furthermore:

$$\begin{aligned} (1 - Gen(q)) \times Fallout(q) &= (1 - Prob(Norm[q])) \times Prob\big(Match[q] \,\big|\, \overline{Norm[q]}\big) \\ &= Prob\left(\overline{Norm[q]}\right) \times Prob\big(Match[q] \,\big|\, \overline{Norm[q]}\big) \\ &= Prob\left(Match[q] \cap \overline{Norm[q]}\right) \end{aligned}$$

The probability of $Match[q]$ can be determined by:

$$\begin{aligned} Prob(Match[q]) &= Prob(Match[q] \cap Norm[q]) + Prob\Big(Match[q] \cap \overline{Norm[q]}\Big) \\ &= Gen(q) \times Recall(q) + (1 - Gen(q)) \times Fallout(q) \end{aligned}$$

By substitution we get:

$$\begin{aligned} &\frac{Gen(q) \times Recall(q)}{Gen(q) \times Recall(q) + (1 - Gen(q)) \times Fallout(q)} \\ &= \frac{Prob(Match[q] \cap Norm[q])}{Prob(Match[q])} \\ &= Prob\big(Norm[q] \,\big|\, Match[q]\big) \\ &= Precision[q] \end{aligned}$$

Exercise 4.1.2
Prove the following properties:

1. $Match[q] = Hits[q] + Noise[q]$
2. $0 \leq Recall(q) \leq 1$
3. $0 \leq Precision(q) \leq 1$
4. $0 \leq Fallout(q) \leq 1$

4.2 Relational Measures

ranking

The main idea is to focus at the progress of the standard measures. Let $x_1, \ldots, x_n$ be an ordering of documents according to non-increasing relevancy. Note that such an ordering need not be unique. We will come back on this later in this section. We assume the searcher will inspect the documents in this order of non-increasing relevancy. The satisfaction of the searcher after inspecting document x_i is defined as the accumulated relevancy sofar:

$$\sum_{k=1}^{i} Match[q](x_k) \quad = \quad |Match[q|i]|$$

where $Match[q|i]$ assigns the same relevancy as $Match[q]$ for the first i documents, while the other documents get relevancy 0. The corresponding recall, precision and fallout is denoted as $Recall(q|i)$, $Precision(q|i)$ and $Fallout(q|i)$ respectively. We wil refer to these values as intermediate values.

For a test case, a simplification of the general case to the double discrete case seems reasonable. A panel is requested to decide about being relevant or not, which seems to be a hard enough problem!

4.2.1 The double discrete case

For the double discrete case, the situation is analyzed further. Let $b_1, \ldots, b_m$ be the successive positions of the relevant documents in the sequence $x_1, \ldots, x_n$. For convenience, we introduce $b_0 = 0$ and $b_{m+1} = n + 1$. Now let $i > 0$ be a document position such that:

$$b_j \leq i < b_{j+1}$$

Note that $j > 1$. Then after inspecting document i, we have the following values for the intermediate retrieval measures:

$$Recall(q|i) \quad = \quad \frac{j}{m}$$

$$\begin{aligned} Precision(q|i) &= \frac{j}{i} \\ Fallout(q|i) &= \frac{i-j}{n-m} \end{aligned}$$

From this we conclude that the intermediate recall remains constant at the interval $[b_j, b_{j+1})$, and increases at interval boundaries. An example of this behaviour is shown in figure 4.1. This example is based on the following experiment. We assume a collection of 100 documents. Document i is relevant for the query in question iff the fraction $\frac{1}{i}$ has a finite decimal expansion. So, for example, document 25 is relevant as $\frac{1}{25} = 0.04$, contrary to document 9 as $\frac{1}{9} = 0.1111...$ Thus the query has 15 relevant documents associated as its normative information need. In figure 4.1 the larger dots correspond to relevant documents.

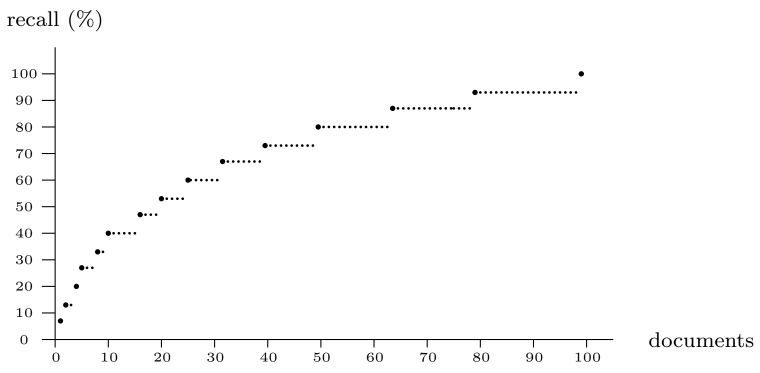

Figure 4.1: Intermediate recall levels

The intermediate precision decreases in an interval stepwise:

$$\frac{j}{b_j}, \frac{j}{b_j+1}, \frac{j}{b_j+2}, \ldots, \frac{j}{b_{j+1}-1}$$

The intermediate precision, however, will not decrease at the boundary of intervals:

$$\frac{j}{b_{j+1}-1} \leq \frac{j+1}{b_{j+1}}$$

as $j+1 \leq b_{j+1}$ and $j \geq 1$.

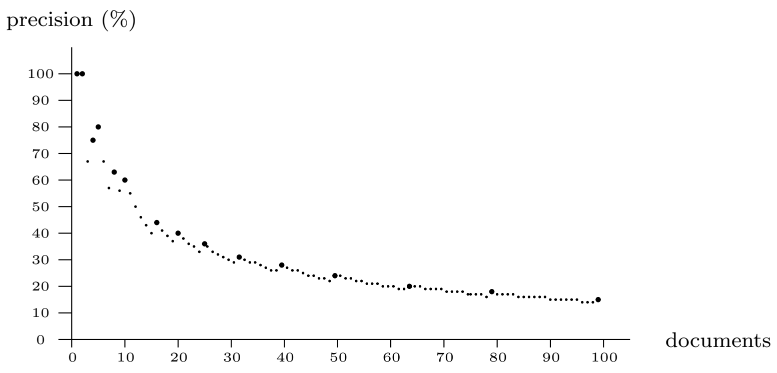

Figure 4.2: Intermediate precision levels

Proof:

Suppose $a+1 \leq b$ and $b > 1$, then

$$\frac{a}{b-1} - \frac{a+1}{b} = \frac{a-b+1}{b(b-1)} \leq 0$$

Furthermore, $b_{j+1} = 1$ would imply $j = 0$ and thus $i = 0$.

The behaviour of the intermediate precision is shown in figure 4.2. The intermediate fallout increases on the interval $[b_j, b_{j+1})$ as follows:

$$\frac{b_j - j}{n-m}, \frac{b_j + 1 - j}{n-m}, \ldots, \frac{b_{j+1} - 1 - j}{n-m}$$

Note that the intermediate fallout is constant at interval boundaries:

$$\frac{(b_j - 1) - j}{n-m} = \frac{b_{j+1} - (j+1)}{n-m}$$

Figure 4.3 shows the intermediate fallout for our example query.

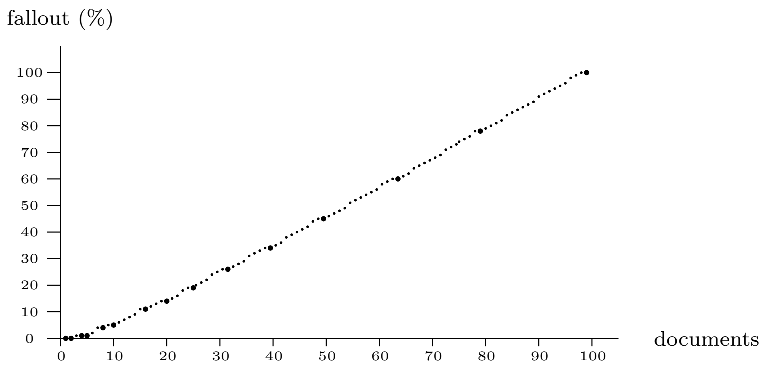

Figure 4.3: Intermediate fallout levels

Exercise 4.2.1
Prove the following properties:

$$(1)\ \mathit{Recall}(q|i+1) = \begin{cases} \mathit{Recall}(q|i) & \text{if } x_i \text{ not relevant} \\ \mathit{Recall}(q|i) + \frac{1}{n} & \text{otherwise} \end{cases}$$

$$(2)\ \mathit{Precision}(q|i+1) = \begin{cases} \dfrac{i \times \mathit{Precision}(q|i)}{i+1} & \text{if } x_i \text{ not relevant} \\ \dfrac{i \times \mathit{Precision}(q|i) + 1}{i+1} & \text{otherwise} \end{cases}$$

4.2.2 The Precision-Recall graph

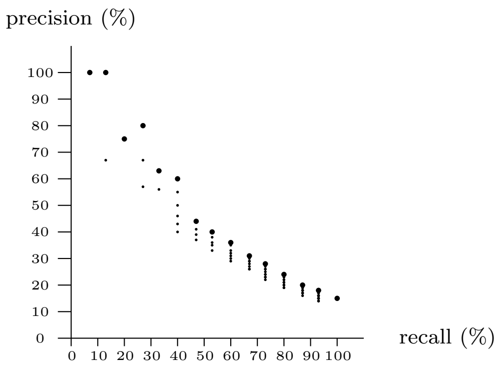

Figure 4.4: Precision-Recall graph

Precision-Recall graph Figure 4.4 shows the intermediate precision in relation to the intermediate recall. This graph is called the *Precision-Recall graph*. This graph is typical for the related query q. The only recall levels that occurr have the form $\frac{i}{m}$ with $1 \leq i \leq m$. Abstracting from q requires interpolation, leading to a precision-recall function. The points with highest precision value for a given recall level are called observed points:

$$\left\{\langle \mathit{Recall}(q|b_j), \mathit{Precision}(q|b_j)\rangle \;\middle|\; 1 \leq j \leq m\right\}$$

They are dotted in figure 4.4, and correspond to the relevant documents associated with query q. The interpolating function $PR[q]$ is defined by left-extension from the *observed points*:

$$PR[q](r) = \mathit{Precision}(q|b_{j+1}) \qquad \text{if } b_j < r \leq b_{j+1}$$

The precision-recall function that results from figure 4.4 is shown in figure 4.5. We see that this function is non-increasing with a few exceptions.

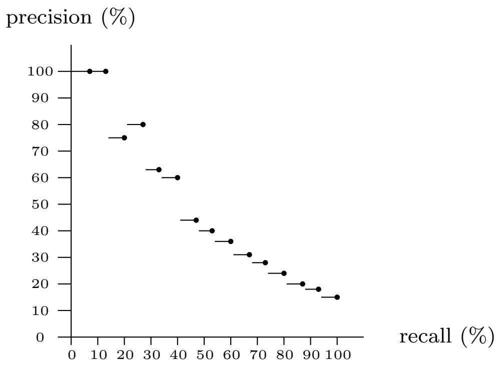

Figure 4.5: Precision-Recall function

Figure 4.6 shows how intermediate fallout is related to recall levels.

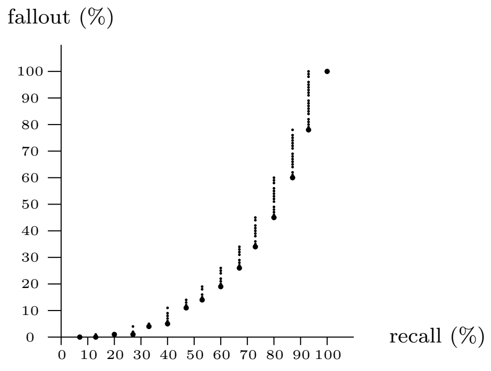

Figure 4.6: Fallout-Recall graph

4.3 Expected search length

A special quality measure is related to the sequential presentation by the IR system presents of its results. It is assumed that the searcher will inspect the documents in the list in this order, until a sufficient number of relevant documents are encountered. The presence of irrelevant documents will be irritating to the searcher. The measure introduced in this section tries to quantify the irritation level, by counting the number of inspected irrelevant documents before the requested number of relevant documents is obtained. Note that this measure does *not* require the knowledge of the total number of relevant documents in the collection. This is especially useful for applications as searching on the World Wide Web.

no unique order

Usually, there is *no unique order* for documents based on their relevancy values (see section 4.2). For example, if relevancy is determined by Jaccard's coefficient (see subsection 6.2.1) as the ratio between the number of terms in the characterization of a document which are also part of the query and the total number of terms from either characterization and query, then there will be a limited number of relevance levels. Suppose in some (large) archive the maximum number of terms in any document characterization equals 15, and the query in question contains 5 terms, then the only the following relevance levels are possible:

$$0, \frac{1}{20}, \frac{2}{20}, \ldots, \frac{19}{20}, 1$$

As a result, many documents will have the same relevance. This leaves the Information Retrieval System with the choice of an ordering of equally relevant documents. This choice, however, has an impact on the quality measures, making them behave more or less randomly. In order to abstract from these whims, we will consider the expected values of the quality measures.

weak order Relevance levels then impose a *weak order* on documents in the archive.

Definition 4.3.1
A binary relation α over a set A is called a weak order if:

1. (reflexive) $\alpha(x, x)$
2. (transitive) $\alpha(x, y) \wedge \alpha(y, z) \Rightarrow \alpha(x, z)$
3. (connected) $\alpha(x, y) \vee \alpha(y, x)$

total ordering Note that a *total ordering* is a weak ordering which is also *antisymmetric*: $\alpha(x, y) \wedge \alpha(y, x) \Rightarrow x = y$. The Information Retrieval system will order the documents according to their relevance. The outcome of the intermediate measures is strongly influenced by the actual order of the documents. We will analyze this situation, and assume that the Information retrieval system will take some random order of the documents. Given the number of relevant documents for each relevance level, we will compute the average position of relevant documents. This quantity may be used for another derived measure: the *expected search length*, which is defined as the number of nonrelevant documents a searcher has to inspect on the average before finding some number of relevant documents. The *search length* for the r-th document thus equals:

expected search length

search length

$$SL(r) = b_r - r$$

As a consequence:

$$Precision(q|b_r) = \frac{r}{b_r}$$

Next we compute the average positions of relevant documents. Let $L_1, \ldots, L_t$ be a partition of the archive according to relevance level, in order of decreasing relevance. We are interested in the expected position b_r of the r-th relevant document. Let class L_i contain l_i documents, of which r_i are relevant. Then the r-th relevant document will occurr in L_k,where:

$$\sum_{i=1}^{k-1} l_i < r \leq \sum_{i=1}^{k} l_i$$

The r-th relevant document thus appears in L_k as s-th relevant document, where:

$$s = r - \sum_{i=1}^{k-1} r_i$$

The expected distribution of relevant documents in L_k will divide L_k in $r_k + 1$ segments of equal size (see figure 4.7). The total number of nonrelevant documents in class L_k equals $l_k - r_k$. The expected length of a segment of nonrelevant documents thus equals:

$$\frac{l_k - r_k}{r_k + 1}$$

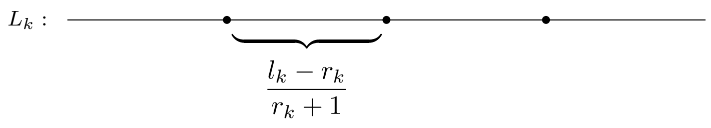

Figure 4.7: Optimal distribution of L_k

The position of the r-th relevant document thus equals:

$$\begin{aligned} b_r &= \sum_{i=1}^{k-1} l_i + s\left(\frac{l_k - r_k}{r_k + 1} + 1\right) \\ &= \sum_{i=1}^{k-1} l_i + \left(r - \sum_{i=1}^{k-1} r_i\right)\frac{l_k + 1}{r_k + 1} \\ &= \sum_{i=1}^{k-1} l_i - \frac{l_k + 1}{r_k + 1}\sum_{i=1}^{k-1} r_i + r\frac{l_k + 1}{r_k + 1} \end{aligned}$$

The expected search length for the r-th relevant document thus equals

$$ESL(r) = \sum_{i=1}^{k-1} l_i - \frac{l_k + 1}{r_k + 1}\sum_{i=1}^{k-1} r_i + r\frac{l_k - r_k}{r_k + 1}$$

The quality of the Information Retrieval system thus is derived from how it assigns relevant documents to relevance levels. This can be normalized by relating it to the performance of randomly **Normalization** searching the archive. In this case, all documents have the same relevance, and thus we have a single class containing all documents from the archive. The expected search length for m documents in a collection of n in this case amounts to:

$$ESL_{\text{random}}(r) = r\frac{n - m}{m + 1}$$

As an example, we consider boolean retrieval. An interpretation of boolean retrieval is that it assign relevance levels 0 or 1 to documents. This partitions the archive in classes L_1 and L_2. The expected search length to retrieve alle relevant documents amounts to:

$$ESL(m) = l_1 + \frac{l_2 + 1}{r_2 + 1}r_1 + m\frac{l_2 - r_2}{r_2 + 1}$$

Chapter 5

Stratified Architecture

As the paradigm depicted in figure 1.2 does not cover important aspects relevant to the present day, we advocate considering information disclosure in a broader, more modern Information System Architecture. (See [21]). Within this framework an information system is considered to have the following components (see figure 5.1):

1. The *information model*. It comprises:
   (a) a *conceptual description* (specification), describing the structure of the stored information, and the rules that govern modifications of the stored information (such as constraints).
   (b) an *information base*, containing the stored information according to the conceptual description. This is usually referred to as an instantiation (population) of the conceptual description.
2. An *information processor*, that processes user requests. The information processor accepts commands from the user via a user interface, interprets them in terms of the conceptual description, and responds in accordance with the information model (both the information base and the conceptual description).

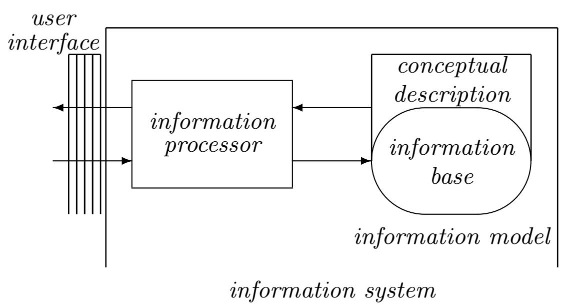

Figure 5.1: The Information System Paradigm

In order to be useful within the application domain, the information system should speak the language of this domain. This language is referred to as the *expert language*. It is the language which (probably in a standardized format) is spoken by the information processor. The grammar which governs this language is referred to as the *information grammar*. The motivation for this *communication-oriented* view is that natural langage can be seen as a frame for mental processes. It is the goal of the process of information analysis to derive the information grammar. From this grammar, the structure of the information to be stored is easily derived.

expert language

information grammar

communication-oriented

A more modern view on this architecture assumes the information to be grouped in so-called *objects*. The information base then evolves into an object base, consisting of a set of co-operating objects. The conceptual description then also forms an organizational description. The motivation for a *object-oriented* view is that this is line with human behaviour.

object

object-oriented

An important aspect of the NIAM approach to information systems is the clear distinction which is made between concrete and abstract objects. Abstract object are abstractions of entities in the application domain, while concrete objects (also referred to as labels) are used to qualify and identify these abstract objects. As special property of concrete objects is that they have a representation in some medium which is used for man-machine communication. Usually, for each label a textual representation is assumed. Abstract objects are denoted and represented in terms of concrete objects. In the stratified architecture the same distinction between abstract and concrete objects is made. However, a more rich mechanism is provided to concretize abstract objects. For example, an object may be presented as a sequence of video stills while playing some music.

strong identification

An important feature of conventional information systems is that all object have associated an object type, and that each object can be uniquely identified in terms of properties of the associated object type. This is referred to as *strong identification*. In the object-oriented approach, the requirement of strong identification is relaxed. Each object has some hidden property, its object identifier. Object identifiers are concrete objects, which have no representation. In the stratified architecture, the same identification mechanism is used.

An important difference between hypermedia systems and (conventional) information systems is the concept of *associative link*, that enables the user to navigate through the information base. Furthermore, state-of-the-art hypermedia systems, in contrast to conventional information systems, feature almost no conceptual description of the stored data. The weaknesses of such an approach have been discussed by several authors ([47], [96]). There seems to be a growing need to be able to support a conceptual description with regard to both hypermedia and traditional document information systems. The combination of structured documents with hypermedia applications looks promising.

hyperindex
hyperbase

In the literature there have been a number of papers which focus on formally defining hypermedia at a *conceptual level*. Several approaches can be recognized; in [47] for example, a model of hypermedia is presented using first-order logic. In [99] hypermedia is modelled in terms of hypergraphs. Recently, two level hypermedia architectures have been emerging. (See [19], [68], [1], [3], [49]). Such architectures feature an upper level, the *hyperindex* comprising a hypertext of indexing information which indexes the lower level, the *hyperbase*. The hyperbase contains the actual information. In our approach, both layers will be organized as information models (see figure 5.1). As a result, the layers constitute a stratified architecture.

An advantage of such architectures is that the searcher can navigate within the upper level to a description of their information need, and then transfer themselves to the lower level via interlayer navigation. Retrieving information is thus reduced to a process coined *Query By Navigation* ([19]). Furthermore, some architectures feature the possibility that a disoriented searcher in the lower level can navigate to the upper level in order to re-orient themselves. We use the term *interlayer navigation* as a generic term for traversal between layers. Relations between layers form the basis for interlayer navigation.

5.1 The Architecture

The *stratified hypermedia architecture* consists of a number of layers and their interrelations (see also [94]). Each layer provides a special way (abstraction level) to look at the information within the archive.

Definition 5.1.1

A stratified hypertext is a structure $H = (\mathbb{A}, \mathbb{L})$, where $\mathbb{A}$ is a set of layers, and $\mathbb{L}$ a set of interlayer links.

In conventional information systems this splitting up in layers is not available. Usually, such a system provides the information on one or two levels of abstraction. One layer, the information base, contains the actual data while the optional second layer provides the meta-data (for example the system tables in SQL).

A layer focusses at some level of abstraction, and offers the possibility to have differents (points of) view(s) at this level of abstraction. A layer thus presents different views on the same underlying base of fragments. Therefore, views not only allow modularization of the information, but also allow flexibility in the form of multiple views on the same information. Both aspects are generally recognized as desirable in hypermedia systems. Formally, a layer is introduced as follows:

Definition 5.1.2
A *Layer* is a structure $L = (\mathbb{F}, \mathbb{N}, \mathbb{R}, \mathbb{V})$ where

- $\mathbb{F}$ is a set of information fragments. This set is called the *Fragment Base*.
- $\mathbb{N}$ is a set of presentation units (or nodes). $\mathbb{N}$ is called the *Node Base*.
- $\mathbb{R}$ is a structure (E, P), where E is a set symbols denoting structural elements, and P is a set of context free production rules. $\mathbb{R}$ is referred to as the *Schema* of the layer.
- $\mathbb{V}$ is a set of views, called the *Mask*.

The similarity with conventional information systems is as follows. The set of fragments corresponds to the set of labels. The (abstract) information grammar corresponds to the grammar $\mathbb{R}$. The production rules are the composition mechanisms such as fact types. The verbalization rules lead to concrete representations of all instances in the information base. These concrete representations corresponds to the presentation units from $\mathbb{N}$. Finally, the population of a conceptual schema corresponds to a particular view.

5.1.1 Fragments

Fragments are the elementary parts of a document, which are not decomposed structurally into smaller components. Each fragment has associated a particular medium (such as text, video and audio). The criterium for judging whether a fragment is atomic or not is not necessarily a property of the fragment itself, but rather is dependent on the lowest level of granularity at which the information is to be considered. For example, animation can be considered as a single fragment, or as a sequence of frames.

5.1.2 Nodes

Nodes are units of presentation, and are used to present the structural components to the user. As a consequence, nodes are constructed from fragments. Formally, a node is a partially ordered set of fragments. We denote a node by the letter N.

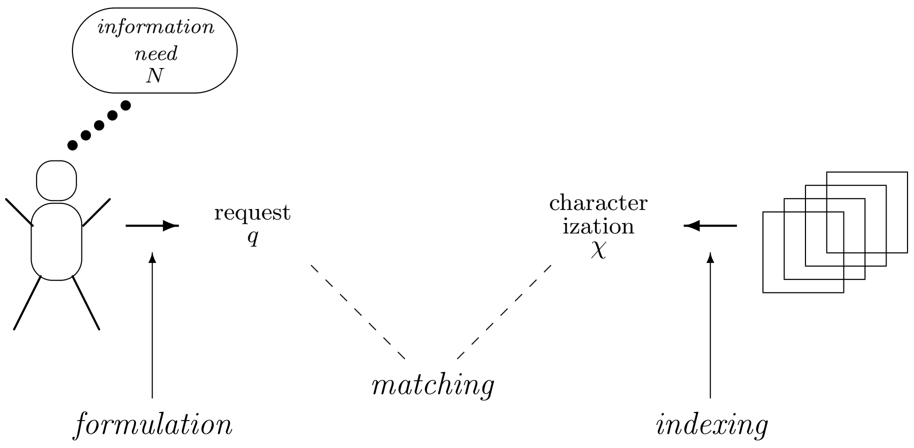

Figure 5.2: A multimedia presentation

As an example, in the node in figure 5.2 the fragments f_1, f_2 and f_3 are displayed on the screen, while at the same time the video v is played, accompanied with the audio fragment m (see figure 5.3

for hypermedia drawing conventions). This node can be represented as the following expression:

$$(f_1; f_2; f_3)\|v\|m$$

A calculus for expressions of this sort has been described in [18].

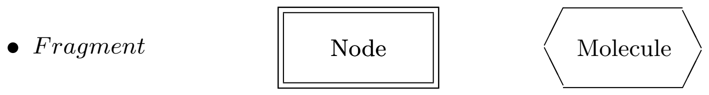

Figure 5.3: Hypermedia drawing conventions

fragment	*position*
f_1	$\langle s, 1\rangle$
f_2	$\langle s, 2\rangle$
f_3	$\langle s, 3\rangle$
v	$\langle v, 1\rangle$
m	$\langle a, 1\rangle$

Figure 5.4: Ordering of fragments in node

In terms of the conceptual schema, the partial order of above example would be represented as in table 5.4.

5.1.3 Rules

Usually information is structured according to some rules. For example, if the information has the form of a book, a book consists of chapters, a chapter consists of sections, etc. Context free rules are a powerful mechanism for such structural specification. A number of models have been defined using context free grammars as their basis ([50], [97]). Context free rules also have practical significance as they form the core of the Document Type Definitions of SGML ([60]), a language which is widely used in the publisher's world and shows signs of becoming a defacto standard for document specification. We adopt the convention of SGML and (basically) allow context free rules only. Rules are expressed in the extended BNF format. This convention is similar to the format adopted by SGML. A rule has a left hand side, which consists of a single symbol and a right hand side, which is a series of one or more symbols, where each symbol may have one of the following occurrence indicators:

- *, the so-called Kleene star, denoting an optional repetition.
- $^+$, the so-called Kleene plus, denoting a repetition.
- ?, denoting an optional occurrence.

Example 5.1.1
The structure of a book, as described in the beginning of this section, is described by:

$\mathsf{book} \longrightarrow \mathsf{chapter}^*$
$\mathsf{chapter} \longrightarrow \mathsf{section}^*$

5.1.4 Views

In the stratified hypermedia architecture a view is defined as follows:

Definition 5.1.3
A *View* is a structure $V = (S, \omega, M, \pi, \mathbb{L})$ where

- $S \in E$ is the *start symbol*
- ω is a set of parse trees generated from S using $\mathbb{R}$. ω is referred to as the *actual structure.*
- M is the set of vertices within ω. A vertex is also called a *molecule.*
- $\pi : M \rightarrow \mathbb{N}$ maps each molecule from M to a presentation unit.
- $\mathbb{L}$ is a set of *associative link* schemata.

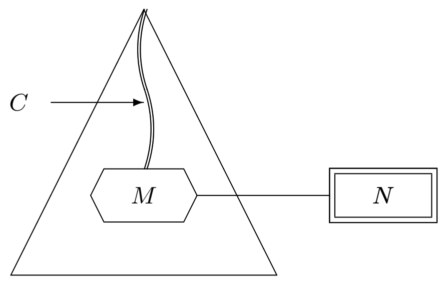

Figure 5.5: The complete context C of molecule M and its presentation by node N

Each vertex in a parse tree corresponds to an instance of a particular structural element, such as a *chapter* or *section.* Such structural elements are termed *molecules.* The *complete context* of a molecule is defined as the path from the root molecule in a parse tree to the molecule in question. On the other hand, an open context corresponds to a downward path in a parse tree, not starting from the root. The term *context* is a generic term for both complete and open contexts.

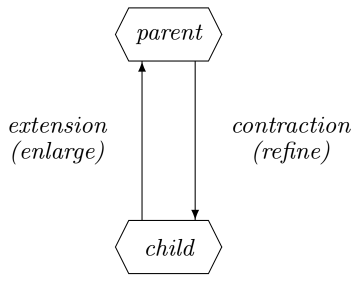

Figure 5.6: Contraction and Extension

The parse tree is useful for disclosure, as it allows the searcher to move through the information on the basis of an underlying structure. For example, moving from a chapter to a section, or from a section to a chapter. This kind of movement is termed *structural navigation.* Structural navigation features the underlying dichotomy that it either extends (enlarges) or contracts (refines) the current context (see figure 5.6).

The rules used to specify the actual structure have a context free format. We allow, however, a more liberal application of these rules than is usual in the theory of context free grammars. In particular, it is possible that a molecule occurs in more than one parse tree. This allows the possibility, for example, that a chapter be shared between different books.

Molecules as such are abstract objects, and need a mechanism to be presented. For this purpose, the function π maps molecules in the actual structure to *nodes.* In this way the actual structure ω is adorned with content in the form of information fragments resident in nodes. It is the task of the so called *author* to provide an actual structure with a proper adornment.

A view also contains a set of *associative link* schemata, where a particular link scheme consists of a set of links of the same category. A link originates from a fragment in a node and leads to a fragment in another node:

$$l \in \mathbb{L} \Rightarrow l \subseteq (\mathbb{N} \times \mathbb{F}) \times (\mathbb{N} \times \mathbb{F})$$

Note that the destination node is in the same layer as the source. This restriction contributes to the layerwise modularization of applications. It is the responsibility of the author to make the link sources, not only visible in the node, but also selectable by the searcher. By selecting a link, the searcher initiates the traversal of the associative link. This is denoted as *associative navigation*.

5.1.4.1 Ambiguity within Views

When traversing an associative link, several possibilities for *system disorientation* exist. First, the destination node of the link may be the presentation of more than one molecule. This is termed *presentational ambiguity*:

$$l \in \mathbb{L} \wedge \langle\langle n_1, f_1\rangle, \langle n_2, f_2\rangle\rangle \in l \wedge \pi(M_1) = \pi(M_2) = n_2 \wedge M_1 \neq M_2$$

Presentational ambiguity has to be resolved into a unique context from which the searcher can continue. One possibility is that the choice be made by the information processor. However, with this solution system disorientation then can lead to *searcher disorientation*. A better solution therefore is that the system provides information about the possible contexts and let the searcher make a choice. Presentational ambiguity can be avoided by employing the constraint that every molecule has a unique presentation. Formally, we require

$$\bigcup_{V \in \mathbb{V}} (\pi_V) \quad \text{is a } \textit{one to one} \text{ function}$$

Another possibility for system disorientation is *contextual ambiguity*. This occurs when the start molecule of a context has more than one parent in the actual structure. In this case, there is more than one conceptual framework with which the searcher may continue. (See molecule M_2 in figure 5.8). In contrast with presentational ambiguity, contextual ambiguity need not be resolved immediately. We will see in a later section how open contexts can be useful for a searcher.

5.1.4.2 The Expressive Power of Views

Currently there are a number of description languages for documents, such as SGML ([60]), TEX([61]) and ODA ([24]). The underlying conceptual model of these description languages is implicit in their definition, although it is recognized that the underlying conceptual model is important (see for example [89] for a conceptual description of SGML).

The concept of layer presented in the preceding sections is powerful enough to express the important aspects of these languages. For example, SGML-based documents are easily mapped into a layer in the following way: Each SGML document can be considered a separate view whose actual structure conforms to the grammar specified in the Document Type Definition (DTD) of the document. Cross references between SGML documents are modelled as associative links between views.

A feature of ODA documents is that they can be viewed both from a logical or a layout perspective. In our architecture, this is modelled by two views based on the same underlying set of fragments (*content portions* in ODA terminology). The actual structures of each view correspond to the *specific logical structure* and *specific layout structure* respectively. The start symbol of each view identifies a set of rules which define a *document class*. Figure 5.7 illustrates another example of multiples views on the same underlying set of fragments in the context of document maintenance. The document readers view has no structure. It presents the document as a whole, so that the document can basically be read sequentially. The reader can deviate from this line by following an associative link. The document maintenance view, on the other hand, takes the full structure of the document into account. This is useful when the component parts of the document are to be manipulated.

The notion of layer is also sufficiently powerful to model state-of-the-art hypermedia. We refer to such hypermedia as *flat hypermedia* as they are constructed by chopping the documents into chunks (called nodes) and linking these pieces together to form a network structure suitable for navigation. Such hypermedia are modelled as a layer whose schema $\mathbb{R} = (\{S\}, \varnothing)$. That is, the layer has no structure (the productions, $P = \varnothing$). As a consequence, each parse tree consists of a single molecule which corresponds to the start symbol S. (See the reader's view depicted in figure 5.7).

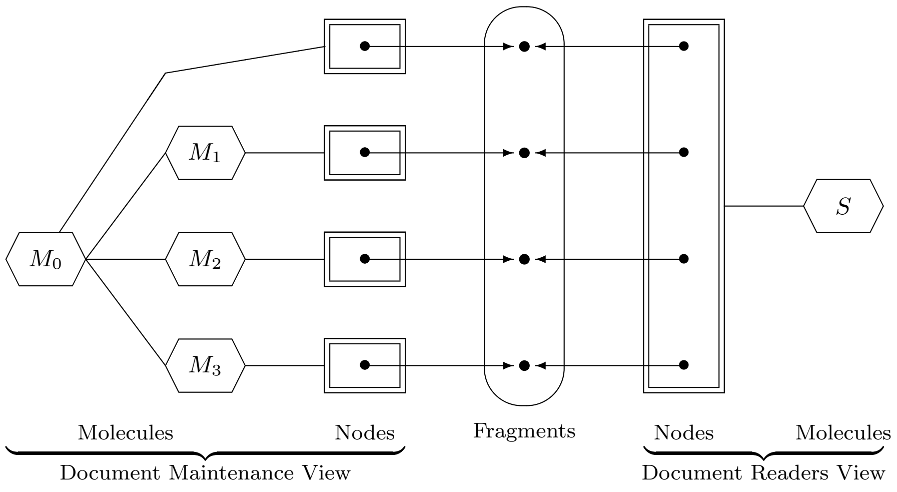

Figure 5.7: Document Maintenance Views

The presentation of each molecule comprises a node containing a document fragment (chunk). The network structure is realized by imposing an associative link scheme over these presentations.

Associative link schemes offer flexibility as they impose no restrictions on how nodes can be linked together. Structural aspects are only simulated by a special link scheme, for example, the hierarchical link. This implies that there is no possibility to constrain the actual structure which is the principal reason why flat hypermedia can readily degenerate into an interweaved mess.

5.2 An Example Layer

In this section we present a concrete example of a layer in the form of a hyperbase, the lower level of a two level hypermedia. We start with the following set $\mathbb{F}$ of information fragments:

f_1 : The effects of pollution on fish can be related to various aspects.

f_2 : The increased industrial capacity of most countries has led to higher concentrations of heavy metals in rivers.

f_3 : These metals have caused the destruction of the ecosystems on which the fish depend as well as killing the fish directly.

f_4 : The effects of the heavy metals remain predominant for years because they sink to the river bottom and are only very slowly flushed out by river currents.

f_5 : Many lakes in Scandinavia have been rendered lifeless by acid rain.

f_6 : This is caused principally by the coal burners in the Ruhr and industrial centres in East Germany and Poland.

f_7 : Because of the economic importance of salmon we consider the effects of river pollution on their migration.

f_8 : There is a higher concentration of heavy metals in rivers due to the increased industrial capacity of most countries.

From these fragments, the following nodes are composed:

N_1 (f_1) The effects of pollution on fish can be related to various aspects.

N_2 ($f_2 + f_3$) The increased industrial capacity of most countries has led to higher concentrations of heavy metals in rivers. These metals have caused the destruction of the ecosystems on which the fish depend as well as killing the fish directly.

N_3 (f_4) The effects of the heavy metals remain predominant for years because they sink to the river bottom and are only very slowly flushed out by river currents.

N_4 ($f_5 + f_6$) Many lakes in Scandinavia have been rendered lifeless by acid rain. This is caused principally by the coal burners in the Ruhr and industrial centres in East Germany and Poland.

N_5 (f_7+f_8) Because of the economic importance of salmon we consider the effects of river pollution on their migration. There is a higher concentration of heavy metals in rivers due to the increased industrial capacity of most countries.

We structure this information by the following grammar:

1. Non-terminals: $E = \{S, M\}$.
2. Rules: $P = \{\mathsf{S} \rightarrow \mathsf{M}^+\}$

We identify the views $\mathcal{V}_1$, $\mathcal{V}_2$ and $\mathcal{V}_3$. Each of these views has S as start symbol, and consists of a single parse tree in the associated actual structure. (This leads to (see figure 5.8):

$\mathcal{V}_1$ the *Pollution and Fish* view. This view has S_1 as start molecule. The start molecule leads to the sequence $M_1, \ldots, M_5$, presented respectively as $N_1, \ldots, N_5$.

$\mathcal{V}_2$ the *River Pollution and Salmon Migration* view, has start molecule S_2, and an underlying sequence M_2, M_3, M_5.

$\mathcal{V}_3$ the *Lake Pollution* view, consisting of start molecule S_3 refined as the sequence M_4, M_3.

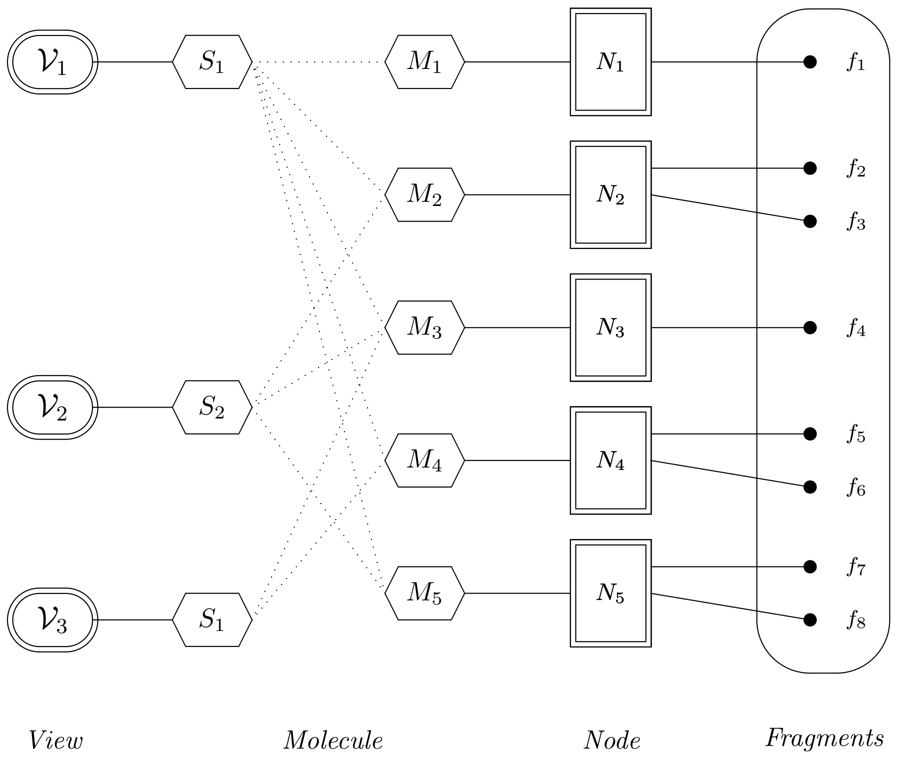

Figure 5.8: The hyperbase of the Pollution example

Note that this example suffers from contextual ambiguity. However, there is no presentational ambiguity. Finally, we consider the links in the views. We assume that in all three views the associate link schemes to be empty. The resulting structure is represented is Figure 5.8. Note that the presentations of the molecules S_1, S_2 and S_3 have been omitted in this figure for reasons of clarity.

In these views we can discern some interesting phenomena: Firstly, there is redundancy within the *River Pollution and Salmon Migration* view. This view contains *twice* a sentence about how increased industrialization has led to higher heavy metal concentrations in rivers. (See nodes 2 and 5). Secondly, the *Lake Pollution* view contains irrelevant aspects, namely node 3 is about heavy metals in rivers and has thus nothing to do with the pollution of lakes.

5.3 The Hyperindex

A hyperindex is a layer of indexing information within the stratified architecture. In such a layer, the fragment base consists of a set of descriptors. The hyperindex typically consists of a single view. Usually, this view is organized as a flat hypermedia. We present two examples of hyperindices. (A more sophisticated hyperindex in the form of a lattice of index expressions is described in [15]).

5.3.1 A Vocabulary as Hyperindex

In this section we describe an example of a hyperindex based on a set of index terms. The underlying set of fragments (the vocabulary) is:

$\mathbb{F}=$ { concentration, destruction, ecodeme, economy, ecosystem, fish, heavy-metal, industry, lake, lifelessness, migration, pollution, river, riverbottom, salmon, Scandinavia, trek, waterbody }

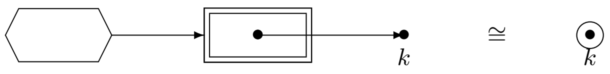

Figure 5.9: Embedding of keyword in hyperindex

We will not exploit any structure in the hyperindex. As a result, we choose for a flat hypermedia having a grammar $G = (\{E\}, \emptyset)$. One view is introduced. For each index term a molecule is introduced, and adorned as shown in figure 5.9. This figure also gives a shorthand pictorial representation for such a structure. This enables us to depict the hyperindex as in figure 5.10.

We introduce two schemata for associative links:

isa The isa-relation expresses the categorial class of index terms. In our example, this relation only consists of:

salmon isa fish
river isa waterbody
lake isa waterbody

corr The corr-relation is a symmetric relation, expressing that an index term corresponds to another index term. In our example we have:

ecodeme corr ecosystem
trek corr migration

These associative links are also represented in figure 5.10.

5.3.2 A Thesaurus as Hyperindex

Iconclass is a tool for the characterization and disclosure of subjects, themes, and motifs pertaining to art of the Western world. The term *Iconclass* is derived from [*Icon*]ographic [*class*]ification. It has been developed over the last forty years by de Waal and Couprie. See [105].

Iconclass divides the world of art into nine main divisions which are depicted in figure 5.11. Each of these main classifications is further subdivided in so called primary subclassifications which are in

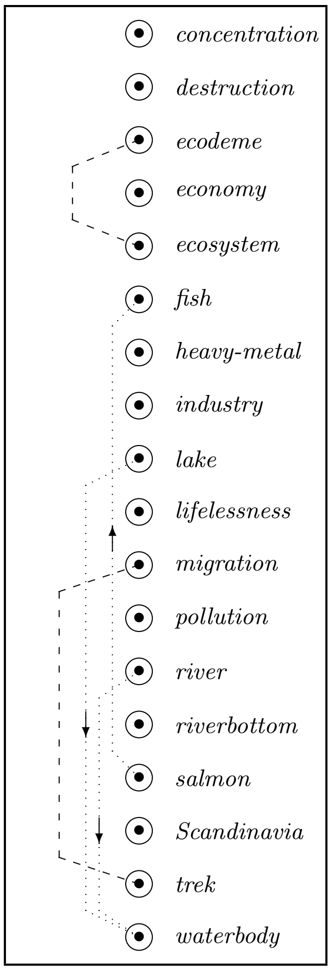

Figure 5.10: The hyperindex

1	The Supernatural, God and Religion
2	Nature
3	Human being, man in general
4	Society, civilization, culture
5	Abstract ideas or conceptions
6	History
7	The Bible
8	Myths, legends and tales (not from classical antiquity)
9	Myths, legends and tales from classical antiquity

Figure 5.11: Main Classifications of Iconclass

1.1	Christianity
1.2	Non-Christian religions
1.3	Magic and Occultism
1.4	Astrology

Figure 5.12: Primary Subclassifications of `The Supernatural, God and Religion`

turn divided into secondary subclassifications and so forth. For example, the primary subdivisions of `The Supernatural, God and Religion` are depicted in figure 5.12.

Iconclass can be modelled as a layer in the following way. The fragment base $\mathbb{F}$ comprises terms like those depicted in the previous figures. Two possibilities exist to model the hierachical structure of the thesaurus. The first is a structural approach: The schema $\mathbb{R}$ specifies a context free grammar which generates hierarchical structures. For example, $\mathbb{R} = \langle\{S\}, \{\mathsf{S} \rightarrow \mathsf{S}^{+}\}\rangle$. A second possibility is to model the thesauras as a flat hypermedia, in which the hierarchy is simulated with a special link scheme. For the purposes of this example, we adopt the structural approach. Only one view will be introduced. This view necessarily has S as start symbol. As the thesaurus consists of a singly hierarchy, the actual structure will contain only one parse tree. The presentation of the molecules depicts the hierarchical relationship between the classifications. Refinement and enlargement of a classification are realized by structural navigation. (In the presentation depicted in figure 5.13 context refinement is denoted by (▽) and context enlargement by (△)). Iconclass does not feature cross references between classifications, meaning that there is no possibility for associative navigation ($\mathbb{L} = \varnothing$).

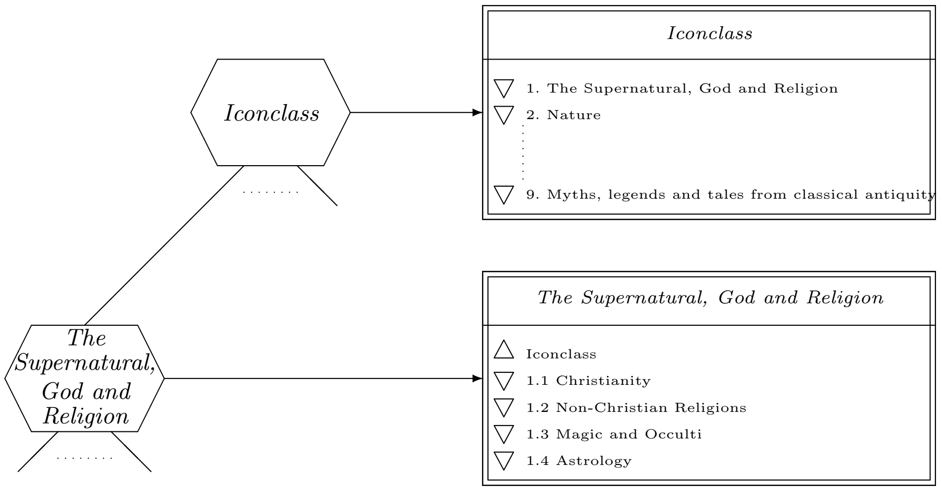
Iconclass
Iconclass
▽ 1. The Supernatural, God and Religion
▽ 2. Nature
▽ 9. Myths, legends and tales from classical antiquity
The Supernatural, God and Religion
The Supernatural, God and Religion
△ Iconclass
▽ 1.1 Christianity
▽ 1.2 Non-Christian Religions
▽ 1.3 Magic and Occulti
▽ 1.4 Astrology

Figure 5.13: Actual Structure and Presentation of Iconclass

5.4 Query by Navigation

5.5 Matching

The Disclosure Machine, when fed with a query q, evaluates the function $Prob(rel\,|A, q)$ for all $A \in \mathcal{O}$, and then selects what molecules are to be retrieved. The following strategies can be adopted (see figure **??**):

1. retrieve the *most general* molecules only. This is usually done in existing disclosure systems. For example, in libraries, it is sufficient to deliver the booknumbers, as they can only be taken as a whole from the shelf.

2. retrieve the *most specific* molecules only, i.e., molecules that are sufficiently relevant, but whose descendants are not relevant enough. In this approach, the information need is satisfied in its finest granularity. The advantage is the minimization of time that has to be spend reading documents in order to find the answer to a specific question ([**?**]).

3. retrieve by *structural element*. For example, the system can be asked to only retrieve relevant sections.

4. retrieve the *relevant subtree*, i.e., all relevant molecules as a separate parse tree.

Chapter 6

Matching Strategies

6.1 Discrete Retrieval

The most simple approach to the disclosure problem is characterizing information objects by keywords. The characterization of a document then will consist of the set of keywords that have been best correspond with the contents of the information object. A user query may be seen as a set of keywords that best corresponds with the contents of the information need of the searcher. As (partial) characterization of these lecture notes is:

$$\{\textit{computing science, information retrieval, archiving, hypertext, hypermedia}\} \tag{6.1}$$

If someone is looking for the use of hypertext in archives, then a proper description of this information need is:

$$\{\textit{archiving}, \textit{hypertext}\} \tag{6.2}$$

Obviously, our lecture notes will be relevant for the searcher. Suppose the archive contains a large number of other documents that match the qualification of the searcher, but many of which adress the issue of implementation. The searcher then may decide to reformulate the query to exclude implementation aspects. This new query is formulated as a logical proposition:

$$\textit{archiving} \land \textit{hypertext} \land \neg \textit{implementation} \tag{6.3}$$

Generalizing this style of querying leads to the idea of a query language consisting of all logical propositions that can be constructed from keywords by application of the boolean operators $\land$, $\lor$ and $\neg$. Note that such queries strongly ressemble conditions as formulated in query languages like SQL. The question is whether our lecture notes match this new query. The matching procedure for logical propositions is based on the *closed world assumption* (or CWA for short). This assumption is well known in other areas. For example, a conventional database system contains facts that hold in the associated universe of discourse. Facts that are not presents in the database are assumed to be false in this universe of discourse. So the closed world assumption states that the description of the current state of the universe of discourse is *complete.* In the case of characterizations of information objects, the closed world assumption states that characterizations are complete. So the object is assumed to handle about all terms from the characterization, and assumed *not* to handel about all terms not being member of this characterization. In our example we can conclude that our lecture notes still will match he reformulated query.

closed world assumption

The model introduced is known as the *boolean retrieval model*. Formally it is described by:

boolean retrieval model

1. The characterization language equals the set $\mathcal{K}$ of keywords that are used to describe documents from the archive.
2. The query language is defined as the propositional logic based on the set $\mathcal{K}$ of keywords interpreted as logical variables. Propositions are defined by the following construction rules:
   (a) each variable is a proposition.

   (b) if p and q are propositions, then also $p \wedge q$, $p \vee q$ and $\neg p$.
   (c) the only propositions are those constructed by the above rules.

3. The matching function yields all documents with a characterization that turns the query into a valid proposition:
$$\mathit{Match}[q] = \left\{x \in \mathcal{O} \;\middle|\; \chi(x) \textit{ makes true } q\right\}$$
When considering document x, the variables from $\mathcal{K}$ are valuated according to the following rule:
$$\mathit{val}(k) = (k \in \chi(x))$$
Document x thus is retrieved if this valuation of variabels validates query q.

It will be clear that boolean retrieval will lead to a high precision. Recall may usually be disappointing. For example, the query
$$\left\{\textit{information retrieval}, \textit{indexing}\right\}$$
will not retrieve our lecture notes, allthough the description indicates that some relevancy may be expected. In order to solve the recall problem, a more subtle view on relevance has to be introduced, which takes (semantical) relations between descriptors into account.

Exercise 6.1.1

Give an informal description of the information need that have probably led to the following queries, and compute the relevance of our lecture notes for them:

1. archiving $\vee$ $\neg$hypertext
2. $\neg$($\neg$archiving $\wedge$ hypermedia $\vee$ hypertext)

Another drawback of boolean retrieval model is of a cognitive nature. The problem is that the logical operators (and, or) do not correspond to the terms 'and' and 'or' in natural language. This may be illustrated by the question:

provide me with all documents on mathematics and computing science

The intended query will probably be:
$$\textit{mathematics} \vee \textit{computing science}$$
Another way to introduce the boolean retieval model is as follows inductively in terms of the construction of boolean propositions:
$$\begin{array}{rcl} \mathit{Match}[t] & = & \mathsf{HitList}(t) \\ \mathit{Match}[p \wedge q] & = & \mathit{Match}[p] \cap \mathit{Match}[q] \\ \mathit{Match}[p \vee q] & = & \mathit{Match}[p] \cup \mathit{Match}[q] \\ \mathit{Match}[\neg p] & = & \mathcal{O} - \mathit{Match}[p] \end{array}$$
It is easy to see that this defintion is equivalent to the previous one, or:

Lemma 6.1.1 $\chi(x)$ makes true $q \iff x \in \mathit{Match}[q]$.

Proof:

We will use induction on q:

1. If q is a single term query, then $\mathit{val}(q)$ yields true is equivalent with $x \in \mathsf{HitList}(t)$.
2. Suppose the property has been proven for propositions p_1 and p_2, then:
   - Let $q = p_1 \wedge p_2$. From the induction hypothesis we conclude: $\chi(x)$ makes true both p_1 and p_2 iff both $x \in \mathit{Match}[p_1]$ and $x \in \mathit{Match}[p_2]$.
   - the case $q = p_1 \vee p_2$ is analogous to the previous case.
   - if $q = \neg p_1$, then $\mathcal{O} - \mathit{Match}[p_1]$ is equivalent with $\chi(x)$ makes true $\neg p_1 = q$.

6.2 Semi-discrete Retrieval

A drawback of discrete retrieval is its rather unrefined relevance estimate. Documents that leave the slightest doubt of being relevant are rejected from the retrieval result. Relevancy has to be proven in a mathematical sense within the boolean retrieval model. In a later chapter, we will see how the proces of relevancy deduction can be generalized to a proces of plausible deduction. In this section we take a more simple approach, and assume the query and characterization language to be the same. Thus both query and characterization are sets of descriptors. We confine ourselves with some methods to estimate the similarity between sets of descriptors.

The relation with the boolean retrieval model goes via the disjunctive normalform for logical propositions:

Theorem 6.2.1 Let F be a logical proposition, then F can be rewritten as follows:

$$F = \bigvee_{i=1}^{k} \bigwedge_{j=1}^{l_i} x_{ij}$$

where each x_{ij} is either a variable, or the negation of a formula.

Exercise 6.2.1
Rewrite the following formulae in disjunctive normalform:

1. archiving $\wedge \neg$(hypertext $\wedge \neg$implementation)
2. archiving $\vee \neg$hypertext
3. $\neg(\neg$archiving $\wedge$ hypermedia $\vee$ hypertext)

Let query

$$q = \bigvee_{i=1}^{k} q_i$$

be a query in disjunctive normalform. The relevancy $\mu(q, c)$ of a document with characterization c then is computed by:

$$\mu(q, c) = Sim(q, c) = 1 - \prod_{i=1}^{k} (1 - Sim(q_i, c))$$

Each term in the disjuctive expansion of q is a conjunction of variables or negated variables. Let p_i be the set of unnegated variables in disjunctive term q_i, and n_i the set of negated variables. Then the similarity of this term with characterization c is computed as:

$$Sim(q_i, c) = Sim(p_i, c) * (1 - Sim(n_i, c))$$

For the computation of the similarity between two sets of descriptors a number of heuristics have been introduced in literature. Those heuristics each have their own way of estimating the degree of overlap between two sets. We will express the alternatives in terms of operations on characteristic functions rather than in terms of set operations

6.2.1 Jaccard's coefficient

Jaccard's coefficient expresses the degree of overlap between two sets A and B as the proportion of the overlap from the whole: $|A \cap B| \,/\, |A \cup B|$ By converting sets to their characteristic function, using the translation rules from table B.1 in appendix B.2, this can be generalized into:

Definition 6.2.1

$$Sim_{\text{Jac}}(f, g) = \frac{|f \bullet g|}{|f| + |g| - |f \bullet g|}$$

Jaccard's coefficient assigns a similarity value between 0 and 1. Similarity of unsimilar documents is mapped onto 0, similarity of equal documents is maximal (in the discrete case).

Lemma 6.2.1

1. $0 \leq Sim_{Jac}(f, g) \leq 1$
2. if f is a characteristic function, then $Sim_{Jac}(f, f) = 1$

Proof:

As both $f_i \leq 1$ and $g_i \leq 1$, we have $f_i g_i \leq f_i$ and $f_i g_i \leq g_i$, and thus

$$2\sum_{i=1}^{k} f_i g_i \leq \sum_{i=1}^{k} f_i + \sum_{i=1}^{k} g_i$$

from which the first property easily is derived.

The second property is obvious for characteristic functions, as in this case $f_i^2 = f_i$ for all $1 \leq i \leq k$, but does not hold generally.

We give some examples of this coefficient. Let $D = \{$*computing science, information retrieval, archiving, hypertext, hypermedia*$\}$, $q_1 = \{$*archiving, hypertext*$\}$, $q_2 = \{$*information retrieval, indexing*$\}$. and let q_3 be the boolean query *archiving* $\vee$ *hypertext* $\wedge \neg$*implementation*. Then:

1. $Sim_{Jac}(q_1, D) = \frac{2}{5} = 0.4$
2. $Sim_{Jac}(q_2, D) = \frac{1}{6} = 0.167$
3. $Sim(q_3, D) = 1 - (1 - Sim(archiving, D))(1 - Sim(hypertext \wedge \neg implementation))$.

   By applying Jaccard's coefficient, the similarity of resulting from the term *archiving* amounts to $Sim_{Jac}(\{archiving\}, D) = 1/5$.

   For the second factor we have: $Sim(hypertext \wedge \neg implementation, D) = Sim(hypertext, D) * (1 - Sim(implementation, D)) = 1/5 * (1 - 0) = 1/5$.

   As a resul we get: $Sim(q_3, D) = 9/25 = 0.36$.

6.2.2 Dice's coefficient

Dice's coefficient relates the overlap of sets A and B to an estimate of their average size. The strategy of Dice is to use the expression $\frac{1}{2}|A|\,|B|$ for this purpose. Using the arithmetical average seems a more natural approachs. The arithmetical average of $f_1, \ldots, f_n$ is defined as

$$\frac{1}{n}\sum_{i=1}^{n} f_i$$

The alternative definition for Dice's coefficient becomes:

Definition 6.2.2

$$Sim_{\text{Dice}}(f, g) \;=\; \frac{|f \bullet g|}{\frac{1}{2}(|f| + |g|)}$$

This bounds the coefficient range to $[0, 1]$. Furthermore, equal documents get the maximal similarity value if their characterization is a characteristic function:

Lemma 6.2.2

1. $0 \leq Sim_{Dice}(f, g) \leq 1$
2. if f is a characteristic function, then $Sim_{Dice}(f, f) = 1$

Proof:
Obvious!

For the example of the previous section we get:

1. $Sim_{Dice}(q_1, D) = \frac{2}{3.5} = 0.5714$
2. $Sim_{Dice}(q_2, D) = \frac{1}{3.5} = 0.286$
3. The similarity of resulting from the term *archiving* amounts to $Sim_{Dice}(\{archiving\}, D) = 1/2.5$.

   For the second factor we have: $Sim(hypertext \wedge \neg implementation, D) = Sim(hypertext, D) * (1 - Sim(implementation, D)) = 1/2.5 * (1 - 0) = 1/2.5$.

   As a resul we get: $Sim(q_3, D) = 0.64$.

6.2.3 The cosine measure

The cosine measure relates the overlap of the sets A and B to their geometric average. The geometric average of $f_1, \dots, f_n$ is defined as

$$\sqrt[n]{\prod_{i=1}^{n} f_i}$$

For the binary case the expression $(|A|\,|B|)^{\frac{1}{2}}$ is obtained.
This is generalized into $\|\kappa_A\|_2 \, \|\kappa_B\|_2$. This leads to the following definition of the the cosine measure:

Definition 6.2.3

$$Sim_{\cos}(f, g) \;=\; \frac{|f \bullet g|}{\|f\|_2 \, \|g\|_2}$$

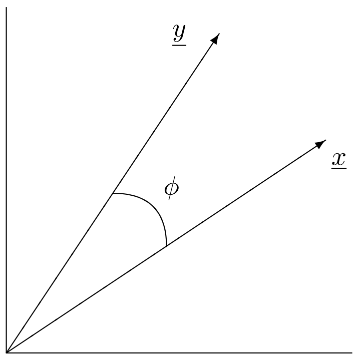

Figure 6.1: The interpretation as vectors

The use of the 2-norm has a nice geometrical interpretation. For this purpose, we interpret functions as vectors. The 2-norm then corresponds to the Euclidian distance between vectors. The inner product for vectors is defined as:

$$(\underline{x}, \underline{y}) = \sum_{i=1}^{n} x_i y_i = |\underline{x} \bullet \underline{y}|$$

The inner product has the following property:

Lemma 6.2.3 Let $\underline{x}$ and $\underline{y}$ be two vectors, and ϕ the angle between these vectors, then

$$(\underline{x}, \underline{y}) = \|\underline{x}\|_2 \, \|\underline{y}\|_2 \cos(\phi)$$

As in our case all vectors are located in the first quadrant, the angle between two vectors can only range from 0 to $\frac{1}{2}\pi$, in which range the cosine function has as nonnegative result. As a consequence we have:

Lemma 6.2.4

1. $0 \leq Sim_{cos}(f, g) \leq 1$
2. $Sim_{cos}(f, f) = 1$

Using the sets of previous section, we get:

1. $Sim_{cos}(q_1, D) = \frac{2}{\sqrt{10}} = 0.632$
2. $Sim_{cos}(q_2, D) = \frac{1}{\sqrt{10}} = 0.316$
3. $Sim_{cos}(q_3, D) = 1 - (1 - 1/\sqrt{5})^2 = 0.694$

6.2.4 The inclusion measure

The inclusion measure quantifies the degree in which set A is covered by set B. In other words, the measure indicates how good A is a subset of B. Therefore it compares the number of common terms in A and B to the number of terms in A. The general definition is:

Definition 6.2.4

$$Sim_{\text{incl}}(f, g) = \frac{|f \bullet g|}{|f|}$$

Note that this measure is not symmetric. A typical application is when we are looking for documents d that satisfy as complete as possible our query q. Thisis expressed by $Sim_{incl}(q, d)$. On the other hand, when using $Sim_{incl}(d, q)$ we are interested in documents d only about topics from q. In terms of probabilities this measure is expressed as $Prob\big(B \,\big|A\big)$. Using the sets of previous section, we get:

1. $Sim_{incl}(q_1, D) = \frac{2}{2} = 1$
2. $Sim_{incl}(D, q_1) = \frac{2}{5} = 0.4$
3. $Sim_{incl}(q_2, D) = \frac{1}{2} = 0.5$
4. $Sim_{incl}(D, q_2) = \frac{1}{5} = 0.2$
5. $Sim_{incl}(q_3, D) = 1 - 0 = 1$
6. $Sim_{incl}(D, q_3) = 1 - (1 - 1/5)(1 - 1/5(1 - 0)) = 9/25 = 0.36$

6.2.5 The overlap coefficient

The idea of the overlap coefficient is to determine the degree in which the sets A and B overlap each other.

Definition 6.2.5

$$Sim_{\text{ovl}}(f, g) = \frac{|f \bullet g|}{\min(|f|, |g|)}$$

As a result: $Sim_{ovl}(f, g) = \max(Sim_{incl}(f, g), Sim_{incl}(g, f))$. We have the following properties:

Lemma 6.2.5

1. $0 \leq Sim_{ovl}(f, g) \leq 1$
2. $Sim_{ovl}(f, f) = 1$

Using the sets of previous section, we get:

1. $Sim_{ovl}(q_1, D) = \frac{2}{\min(5,2)} = 1$
2. $Sim_{ovl}(q_2, D) = \frac{1}{\min(5,2)} = 0.5$
3. $Sim_{ovl}(q_3, D) = 1$

6.3 The General case

In the general case, both query and characterization are non-discrete functions. This leads to the following generalization of 6.4 (by converting sets to their characteristic function, using the translation rules from table B.1 in appendix B.2):

$$\begin{array}{rcl}
Match[t] & = & \mathsf{HitList}(t) \\
Match[p \wedge q] & = & Match[p] \bullet Match[q] \\
Match[p \vee q] & = & Match[p] + Match[q] - Match[p] \bullet Match[q] \\
Match[\neg p] & = & 1 - Match[p]
\end{array}$$

Chapter 7

Simple Characterization

7.1 Introduction

In chapter 3 the dependency of information disclosure on the quality of the characterization of information objects has been discussed. A reason why information disclosure is so hard is the incompleteness and unpreciseness of characterizations. This is also referred to as *loss of information*. **loss of information**

An *incomplete* characterization means that there are aspects of the object that are not represented in its associated characterization. As a result, if there is a request directed at any of these hidden aspects, the object cannot be disclosed, thereby reducing recall. An *unprecise* characterization means that the object may also be unwanted disclosed, lowering precision, thereby increasing recall. **incomplete** **unprecise**

How to characterize an object to facilitate its disclosure has long been one of the driving questions in information disclosure. In terms of the terminology introduced in chapter 3, this question implies the choice of a proper descriptor (characterization) language $\mathcal{C}$. This choice implicitly involves other questions. How can it be determined that a language $\mathcal{C}_1$ offers better disclosure than a language $\mathcal{C}_2$? Another issue is whether $\mathcal{C}$ should also serve as the language of requests. It is thinkable that a given language may be well suited for the purposes of characterization, but not suitable for request formulation.

For the disclosure of textual objects the characterization and query language will usually be similar. However, for non-textual objects (images, sound, video) there will be differences. In this lecture we will restrict ourselves to textual objects.

The choice of a characterization language strongly depends on the context of the retrieval process. If retrieval is restricted to a delimited collection, then an optimal set of content descriptors is to be pursued. However, if the collection is very dynamic, for example a stream of incoming documents that are to be filtered according to user profiles, then a universal characterization language is demanded.

Maron [70] points out that once charaterization language $\mathcal{C}$ has been chosen, the following two issues must be resolved.

1. The first is the so called *indexing problem*, namely how to assign those descriptors in $\mathcal{C}$ to an object x to facilitate x's disclosure. **indexing problem**

2. The second issue is how the disclosure mechanism is to use the characterization χ of information objects in order to realize effective information disclosure.

These two questions constitute underlying themes for this and succeeding chapters. This chapter provides a background in state-of-the-art characterization methods.

7.2 Characterization of Information Objects

In the world around us an object usually has a unique identification so that it can be distinguished from other objects. Such a unique identification is necessary to singularly disclose that object. In the framework of the information disclosure paradigm introduced earlier, an object x has a characterization $\chi(x)$ drawn from the language $\mathcal{C}$. Therefore, in order to effectively disclose x, its characterization $\chi(x)$ must distinguish object x from other objects. For example, if a book about movie stars is characterized by the index term stars, it will not be distinguishable from books dealing with astrophysics.

Furthermore, $\chi(x)$ must also *usably* distinguish x. The disk address of an information object distinguishes it perfectly from other objects, but for the purposes of disclosure by a human searcher this characterization is almost certainly useless.

Information Disclosure Principle

Usability and discrimination are fundamental to the *Information Disclosure Principle* which states that in order to effectively disclose an information object it must be characterized so that it is usably distinguishable from other information objects.

It follows from this principle that a descriptor language must have sufficient expressive power to realize discrimination and at the same time be useful in a pragmatic sense. (Chapter 4 of Blair's book [11] gives a linguio-philosophical motivation regarding the pragmatics of characterization languages).

indexing

information object

In the past, *indexing* was performed by a person who scanned the *information object* and assigned descriptors from $\mathcal{C}$ which (s)he believed were both a good reflection of the content of the object, and were likely to be used in a request for that object. With the advent of computers, *automatic indexing* has come to the fore, as it is requisite to disclosing the ever growing mountains of information.

The indexing task is usually seen as two processes:

1. first, assign to each information object, descriptors from $\mathcal{C}$, which are capable of representing its content, and
2. and then, assign to each descriptor a *weight*, or value, reflecting its presumed importance for purposes of content identification.

In the case of textual objects, the first and most obvious place where a characterization language $\mathcal{C}$ might be constructed from is the text of the documents themselves. The descriptors of such a language could merely be (a subset of the) the set of all words (*keywords*) which occur in the object base. We will discuss several methods to describe the contents of documents using only those keywords.

More advanced methods try to use more of the linguistic structure. This can be based on the grammar of the language used in the information objects, or it may come from a statistical analysis of the sentences occurring in the object base. In chapter 8 we will discuss index expressions as a simple format for *phrases*.

In this chapter we will also discuss the rationale behind *N-grams*, aiming at recognizing similarity between words that can be derived from their spelling.

7.3 Keywords

Keyword descriptors have the advantage that there are a number of straightforward indexing algorithms to derive them automatically from the objects ([87]). Most automatic indexing methods use the observation that words occur *unevenly* in natural language texts, and the frequency of occurrence of an individual word has something to do with its importance for purposes of content representation. It is instructive to look at this observation more closely.

Let $\texttt{intfreq}(t, x)$ denote the occurrence frequency of term t in information object x. As a consequence, an information object may be seen as a multiset of terms.

Remark 7.3.1

The occurrence frequency may also be extended by involving term similarity. Let $n(x)$ *be the* *original frequencies of words in document x, then:*

$$\texttt{intfreq}(w,x) = \sum_s \mathit{Sim}(s,w)n(s,x)$$

If a meaning can be seen as a list of words (see esction 3.6), then the occurrence weight of meaning m can be quantified as:

$$\texttt{intfreq}(m,x) = \overline{\{\mathit{Sim}(s,w)n(x) \mid s \in \mathcal{C}\}}$$

For indexing purposes, the following quantities are relevant, and have to be derived from the occurrence frequency of terms:

1. How good does a term describe the contents of a document. This will be referred to as the weight of the term for that document. If the weigth descibes to what extent a document is about a term, then this weight may also be seen as the relevance of that document when this term is used as a qeury.
2. How good is a term as discriminator between documents. This is referred to as the goodness of that term.

For convenience, we introduce the following quantities. The size of a document is introduced as the number of terms it contains: $\mathsf{DocSize}(x) = \sum_t \texttt{intfreq}(t,x)$. The size of a term (total occurrence frequency of a term) is obtained by summing its term occurrences over all documents from the collection under consideration: $\mathsf{TermSize}(t) = \sum_{x \in \mathcal{O}} \texttt{intfreq}(t,x)$.

In fact, if the words of an object base are ranked in a descending order of their occurrence frequency, the occurrence characteristics of the vocabulary allways seem to be characterized according to *Zipf Law*: **Zipf Law**

$$\mathsf{TermSize}(t) \times \mathsf{rank}(t) \approx \mathit{constant} \tag{7.1}$$

This law has been explained by citing a general "principle of least effort" which makes it easier for a writer to repeat certain words instead of coining new words. The same principle accounts for the fact that the most frequent words tend to be short function words, such as and, of,the, etc. Zipf Law has been verified many times using text materials in different areas.

7.3.1 tf-weighting

Based on equation 7.1, it is possible to construct a simple characterization language. First, $\mathsf{TermSize}(t)$ is calculated for every term, and terms are arranged in a decreasing order according to their $\mathsf{TermSize}(t)$. Then, high frequency function words are eliminated by choosing some suitable frequency threshold value. Last, low frequency words are eliminated, since these occur so infrequenctly that their presence does not affect the effectiveness of the disclosure. The remaining medium-frequency words can now be used for assignment to the documents as index terms, and a simple weighting scheme, expressing *to what extent* a document is about a term, can be defined as:

$$\mathsf{Weight}_{\mathrm{tf},0}(t,x) = \texttt{intfreq}(t,x) \tag{7.2}$$

This weighting scheme can be normalized by

$$\mathsf{Weight}_{\mathrm{tf},1}(t,x) = \frac{\texttt{intfreq}(t,x)}{\mathsf{TermSize}(t)} \tag{7.3}$$

In terms of the probabilistic approach (see 3.7), this fraction may be interpreted as the probability of document x being relevant given evidence t. This may be denoted as $\mathit{Prob}(x \mid t)$.

A characterization language based on the previously mentioned observations will be too crude for practical disclosure environments, as it does not fulfill the requirements of the Information Disclosure Principle (see section 7.2) of being both usable and distinguishing. For instance, the term computer is not distinguishing within an object base about computing, no matter what its frequency of occurrence is, because it is likely to occur in every object. Thus, with the *Information Disclosure Principle* in mind, a *good* term is one The following rationale is often used to identify terms that usably discriminate objects from each other:

> A term t usably discriminates an object x, if it occurs relatively frequently in x and relatively infrequently in the other objects.

weighting schemes

Several *weighting schemes* have been derived from this rationale, including an inverse document frequency function, the signal-noise ratio, and the term discrimination value. These weighting functions are introduced in the following sections.

7.3.2 idf-weighting

The *external* frequency $\texttt{extfreq}(t, \mathcal{O})$ of a term t with respect to an object base $\mathcal{O}$ is the number of information objects that contain that term (counting the objects x having $\texttt{intfreq}(t, x) > 0$). When the object base is understood, $\texttt{extfreq}(t)$ will be used for short. The factor $\frac{n}{\texttt{extfreq}(t)}$ is usually referred to as *inverse document frequency*, denoted as $\text{IDF}_((t))$, where $n = |\mathcal{O}|$ is the number of documents in the collection $\mathcal{O}$. The inverse document frequency can be used as a scheme to evaluate the quality of terms for discrimination between objects. This is also referred to as the *goodness* of terms.

inverse document frequency

goodness

$$\mathsf{Goodness}_{\text{idf},0}(t) = \text{IDF}_((t)) \tag{7.4}$$

Note that $1/\text{IDF}_((t))$ can be seen as the probability $\mathit{Prob}(t)$ of term t to occur in a document of the collection $\mathcal{O}$. Terms that have a high discriminative power can be characterized as terms with a small probability of occurrence. As a consequence, good terms will have a high Goodness-value.

7.3.3 tf×idf-weighting

A composite expression, satisfying the Information Disclosure Principle, measuring the importance of a term t in a given document x would increase as $\texttt{intfreq}(t, x)$ increases, but decrease as the $\texttt{extfreq}(t)$ increases. A possible weighting function is

$$\mathsf{Weight}_{\text{tf}\times\text{idf}}(t, x) = \texttt{intfreq}(t, x) \times \text{IDF}_((t)) \tag{7.5}$$

In terms of probabilities, this weighting scheme can be interpreted as follows:

$$\frac{\mathsf{Weight}_{\text{tf}\times\text{idf}}(t, x)}{\mathsf{TermSize}(t)} = \frac{\mathit{Prob}(x \mid t)}{\mathit{Prob}(t)} = \mathit{Prob}(x, t) \tag{7.6}$$

7.3.4 Signal-Noise Ratio

signal-noise ratio

The *signal-noise ratio* weighting scheme is based on information theory, where the information value $I(e)$ of an event e occurring with probability p_e amounts to:

$$I(e) = -\log_2 p_e \tag{7.7}$$

The information value of an event signifies the degree of surprise of an observer of that event. For example, randomly picking the queen of spade from a deck of bridge cards has a probability of $\frac{1}{52}$, and thus an information value of $-\log_2 \frac{1}{52} \approx 5.7$. Picking a spades card has associated the probability of $\frac{1}{4}$, and is thus much more likely to occur. The information value $-\log_2 \frac{1}{4} = 2$ of this event is thus less than that of picking the queen of spades. An observer thus will be more surprised when the queen of spaces is randomly taken rather than just a spades card.

entropy

The average information value of all events (E) is referred to as the *entropy* $H(p)$ of the associated probability function p.

$$H(p) = \sum_{e \in E} p_e I(e) = -\sum_{e \in E} p_e \log_2 p_e$$

The entropy of a distribution can be seen as the average surprise of its outcome, or, the degree of uncertainty of its outcome. The maximal uncertainty occurs when all probabilities are equal. For a uniform distribution, the entropy takes its maximal value. In that case all n (say) events have the same probability $\frac{1}{n}$:

$$\begin{aligned} H(p) &= -\sum_{e \in E} p_e \log_2 p_e \\ &= -\sum_{e \in E} \frac{1}{n} \log_2 \frac{1}{n} \\ &= \log_2 n \end{aligned}$$

The entropy is minimal if a single event (t_0) has probability 1. In that case there is no uncertainty over the outcome of the statistical process:

$$\begin{aligned} H(p) &= -\sum_{e \in E} p_e \log_2 p_e \\ &= -Prob(t_0) \log_2 Prob(t_0) \\ &= 0 \end{aligned}$$

In order to apply this, we focus on the probability $Prob(x \,|t)$ of document x being relevant given evidence t. We will use $p_{x,t}$ as a short notation for this conditional probability. The uncertainty of what documents are relevant when term t is used as a query term (i.e., as a piece of evidence for the information need of the searcher), may thus be expressed as:

$$H_t = -\sum_x p_{x,t} \log_2 p_{x,t}$$

This entropy can be used as an impression of the discriminating quality of the associated keyword. It is also referred to as the *noise* of term t, denoted as $Noise(t)$. It is usual to normalize this outcome, leading to the following definition for the goodness of term t: **noise**

$$\mathsf{Goodness}_{\mathrm{entr},1}(t) = 1 - \frac{H_t}{H_{max}} \tag{7.8}$$

where H_{max} is the maximal entropy a term can have. Note that terms which lead to maximal uncertainty about document relevance have goodness 0. Terms that occur in only one document only have a maximal goodness score. If the characterization language $\mathcal{C}$ contains m terms, then the maximal entropy amounts to $\log_2 m$.

$$\mathsf{Goodness}_{\mathrm{entr}}(t) = 1 - \frac{H_t}{\log_2 m} \tag{7.9}$$

A weighting scheme for terms is introduced as follows.

Terms with a low entropy are terms that are very specific for a small number of documents. An inverse of the noise of a term might be used to indicate the value of a term, for example:

$$\mathsf{Signal}(t) = \log_2 \mathsf{TermSize}(t) - Noise(t)$$

and a possible weighting scheme, analogous to the tf $\times$ idf is

$$\mathsf{Weight}(t,x) = \mathtt{intfreq}(t,x) \times \mathsf{Signal}(t) \tag{7.10}$$

Nevertheless, the signal-noise weighting does not give optimal performance in a disclosure environment.

7.3.5 Term Discrimination Value

For the collection O an "average" document x^{avg} can be constructed, for which its terms exhibit average frequency characteristics:

$$\mathtt{intfreq}(t, x^{avg}) = \frac{1}{n} \sum_x \mathtt{intfreq}(t,x) = \frac{1}{n} \mathsf{TermSize}(t)$$

The virtual information object x^{avg} is usually referred to as the *centroid* of the collection. **centroid**

Let $Sim(x, y)$ be a similarity measure for information objects x and y, based on their assigned indexing terms. A similarity value of zero represents no agreement among the indexing terms, and a value of one represents perfect agreement. The average similarity can be computed as:

$$Sim^{avg} = \frac{1}{n} \sum_{x} Sim(x^{avg}, x)$$

The average similarity represents the *density* of the object base, i.e. how similar the objects are to each other. A low density will result from a collection with documents on varying subjects. If all documents from the collection more or less cover the same topic, then the collection density will be close to 1.

In order to get an impression on the effect of term t, we consider the collection $\mathcal{O}$ with term t removed from each document. Let Sim_t^{avg} represent the average similarity in this case. The *term discrimination value* of term t is defined as the incremental effect of term t on the collection density: **term discrimination value**

$$\mathsf{DiscValue}(t) = Sim_t^{avg} - Sim^{avg}$$

As a consequence, DiscValue(t) measures the degree to which the use of term t will help to distinguish the objects from each other. A positive discrimination value for a term means that its removal will compress the object base (increase its density), making objects less distinguishable. Thus, a term with a positive discrimination value is a good term and should not be removed. Conversely, the removal of a term with a negative discrimination value expands the object base (decreases its density), consequently it is a bad term and should be removed. Terms with close to zero discrimination value are *indifferent*, i.e. their removal leaves the object space unchanged.

The term discrimination value can be used to compute a weight for each term t as:

$$\mathsf{Weight}(t, x) = \mathtt{intfreq}(t, x) \times \mathsf{DiscValue}(t) \tag{7.11}$$

7.4 Other Weighting Techniques

Some different weighting techniques arise, if judgments about the relevance of the objects with respect to some query exist; these are the term relevance and utility value weighting.

7.4.1 Term Relevance

The probabilistic indexing theory states that the best index terms are those that tend to occur in the relevant documents with respect to some query. A measure of term value is obtained from the *term relevance* TermRel(t), which is the ratio of the proportion of relevant objects in which term t occurs to the proportion of non-relevant objects in which the term occurs. **term relevance**

With respect to a query, if R is the number of relevant objects, then $n - R$ is the number of non-relevant objects. Let r_t be the number of relevant objects containing the term t. Then $\mathtt{extfreq}(t) - r_t$ is the number of non-relevant objects containing t. Term relevance is then computed as:

$$\begin{aligned} \mathsf{TermRel}(t) &= \frac{\frac{r_t}{R - r_t}}{\frac{\mathtt{extfreq}(t) - r_t}{n - R - (\mathtt{extfreq}(t) - r_t)}} \\ &= \frac{r_t}{R - r_t} \times \frac{n - R - (\mathtt{extfreq}(t) - r_t)}{\mathtt{extfreq}(t) - r_t} \end{aligned}$$

and a weighting scheme based on term relevance can be:

$$\mathsf{Weight}(t, x) = \mathtt{intfreq}(t, x) \times \mathsf{DiscValue}(t)$$

Under the assumption that terms occur in objects *independently*, the weighting scheme of equation 7.4.1 is *optimal*. However, it cannot be computed unless relevance judgments of the objects are available with respect to some queries. Some techniques exist which can estimate r_t and R based on $\texttt{extfreq}(t)$. Experiments have shown that even when r_t and R are estimated, the term relevance weighting performs better than tf × idf weighting, and this suggests that more information is contained in $\texttt{extfreq}(t)$ data, than is normally used by weighting schemes.

Nevertheless, in text classification, routing, and filtering, where training sets of objects with relevance judgments are usually available, the term relevance weighting can be applied accurately. The same holds for the utility-based weighting described next.

7.4.2 Utility

Utility

Utility is one of the metrics for evaluating information disclosure systems. The utility of a query is defined as the sum of values achieved for every retrieved relevant and rejected non-relevant object, minus the sum of the costs for every non-retrieved relevant and retrieved non-relevant object.

Let v_1 be the value for every relevant retrieved object, v_2 the value for every non-relevant rejected, c_1 the cost for every relevant rejected, and c_2 the cost for every non-relevant retrieved. Then the utility $\mathsf{Util}(t)$ for term t with respect to a query can be defined as:

$$\mathsf{Util}(t) = (v_1 + c_1)r_t - (v_2 + c_2)(\texttt{extfreq}(t) - r_t)$$

where r_t is the number of relevant objects containing the term t. A corresponding term weighting function for term t in object x is then given by:

$$\mathsf{Weight}(t, x) = \texttt{intfreq}(t, x) \times \mathsf{Util}(t)$$

7.5 Term Phrases

If computer is a term descriptor which characterizes objects dealing computers and analogously for programming, the problem of how to characterize an object that is specifically about computer programming arises. A term phrase is an extension of the term descriptors allowing more specific descriptors; the phrase computer programming is more detailed, and therefore distinguishes objects better, than the term computer or programming.

Even though straight forward indexing methods exist for the generation of term phrases ([87]), these methods often suffer from the problem that either too many non-meaningful phrases are generated, or conversely a large percentage of the phrases are meaningful but the resulting characterizations are incomplete.

A good survey of term phrases together with advanced indexing algorithms can be found in the work of Gay and Croft [GC9O]. With regard to the information disclosure principle, term phrase characterization languages offer better possibilities to distinguish objects, but the open question seems to be how to effectively use the term phrases in the associated information disclosure mechanism. There is no empirical evidence that suggests that term phrases offer better information disclosure than keywords.

7.6 N-grams

Up to this point all descriptors have been based on whole words. The n-grams of a word w are overlapping substrings of w of length n. For example, if w = theory, then the 3-grams are: the, heo, eor, ory.

The n-gram characterization of an object can be indexed by the union of the n-grams produced by each word in the object. For example, the information objects:

1. queuing theory is the basis of server systems
2. server systems are based upon queuing theory

have the following 3-gram based characterizations which are presented in alphabetical order:

1. {*are*, *ase*, bas, ein, ems, eor, erv, eue, heo, ing, ory, emphpon, que, rve, *sed*, ser, ste, sys, the, uei, ueu, *upo*, ver, yst}
2. {*asi*, bas, ein, ems, eor, erv, eue, heo, ing, ory, que, rve, ser, *sis*, ste, sys, *tem*, the, uei, ueu, ver, yst}

These characterizations contain 24 and 22 elements respectively, 19 of which are in the intersection.

The number of different n-grams grows exponentially with n. For $n = 3$ (assuming an alphabet of 26 letters) there are $26^3 = 17,576$ different trigrams, whereas there are $26^4 = 56,976$ distinct tetra-grams. Teufel and Schmidt [T588], however, state that only about 25estimate on tests done using German and English texts. Research with Dutch texts has found that only between six and seven thousand trigrams actually occur [Sta9O]. This represents roughly 35notably higher than the Teufel and Schmidt estimate. (The difference can be explained by the occurrence of vowel combinations in Dutch not found in German or English). Teufel and Schmidt claim that $n = 3$ is optimal with respect to the computational cost of producing the n-grams and information disclosure effectiveness.

N-grams have proven to be versatile because they can be readily produced by a straightforward syntactic process. For example, trigrams are often used in spelling checkers because spelling variations of a word normally have a similar set of associated trigrams [5al89]. In another application, trigrams and tetragrams were used to as characterization mechanism for information in a wide area network [WF89].

Even though the language of trigrams allows objects to be usably distinguished, there is little empirical evidence regarding the effectiveness of disclosure systems based on this language. It can be predicted that the recall of trigram based disclosure system will be higher than that of a term based system because of the former's ability to deal with spelling variations. How this increase in recall will be paired by a decrease in precision. The Document Information Technology section of the Dutch research organization TNO claims that the loss in precision can be compensated by using trigram based characterizations only in conjunction with small information objects.

Sometimes the spelling variations of a term are so variable, that there is little or no overlap between their respective trigrams. For example, the word Krushchev versus Chroesjtsjov. (This word apparently has 2880(!) spelling variations [NvJ91]). The effect is that if Kruschev is given as a request and a document contains the word Chroesjtsjov it is not possible to establish the link between these two terms by matching the respective trigram characterizations because the overlap of the respective trigram characterizations is empty. An offshoot of the trigrams, the so called triphones, have been developed to counter this sort of problem. A triphone is a three letter trace which denotes a sound. In the above case the two terms would map to a similar set of triphones thereby allowing the information disclosure mechanism or spelling checker to detect the strong relationship between these terms.

Chapter 8

The Language of Index Expressions

Index expressions are an extension to the term phrases whereby the relationships between terms are modelled. Their philosophical basis stems from Farradane's *relational indexing* ([43], [44]). Farradane projected the idea that much of the meaning in information objects is denoted in the relationships between terms. A parallel can be drawn here with the conceptual model from the database world where relationship types between entities play an important role; the characterization of an object consisting solely of keywords would be like an entity relationship model without relationship types.

Index expressions

As there are many possible relationships between terms, Farradane proposed a framework of nine relationship *types* with which any given term relationship could be classified. For example, author wrote book exhibits a *functional dependence* relationship type between book and author. Farradane motivated his relationship types on the basis of psychological thought mechanisms.

In relational indexing, trained indexers would peruse an object and classify the term relationships. The resulting characterization comprises a network of terms, where each arc in the network represents one of the nine relationship types. Even though the resulting characterizations clearly capture more of the content of an object than the descriptor languages presented so far, the disadvantage is that indexing has to be performed manually. This is probably the reason why Farradane's relational indexing never blossomed.

Craven uses a similar approach to Farradane in his *linked phrase indexes* ([38], [39]). Like relational indexing, the basis of a linked phrase index is a network of terms, in which the arcs correspond to relationships denoted by prepositions. Such networks are also produced by a manual indexing process, although Craven does propose that automatic network derivation is possible from the titles of objects.

In this chapter we will consider index expresiosn as a special form of noun phrases. Noun phrases can be seen as reprentations of human concepts.

8.1 Linguistic Backgrounds

From natural language theory, it is well known that the so-called *noun phrase* is an essential part of sentences. The entities shown in Table 8.1 play a role in noun phrases. Noun phrases a such

noun phrase

NP	noun phrase	PP	prepositional phrase	Art	article
NG	noun group	AP	adjectival phrase	Noun	noun

Figure 8.1: Abbreviations

are too complex to handle for retrieval systems, especially automatic indexing is no feasible. For the purposes of information retrieval a sufficiently rich subset is more practical. Adopting the work

basic noun phrases

presented in [7] and [48], a syntax for *basic noun phrases* is given by:

$$\begin{array}{rcl}
\mathsf{NP} & \rightarrow & \{\mathsf{Art}\}\mathsf{NG} \\
\mathsf{NG} & \rightarrow & \mathsf{NGPP} \mid \mathsf{APNG} \mid \mathsf{N} \\
\mathsf{PP} & \rightarrow & \mathsf{PNP} \\
\mathsf{P} & \rightarrow & p \in \mathsf{Prepositions} \\
\mathsf{N} & \rightarrow & n \in \mathsf{Nouns} \\
\mathsf{AP} & \rightarrow & a \in \mathsf{Adjectives} \\
\mathsf{Art} & \rightarrow & b \in \mathsf{Articles} \cup \mathsf{Determiners}
\end{array}$$

This grammar is reprented as a syntax diagram in figure 8.2. As the full syntax of noun-phrases

Figure 8.2: Syntax diagram for basic noun phrases

is quite rich, the following simplification is mentioned as a sufficiently rich subset of noun phrases, which is yet covering most of the noun phrases in documents (see [7]):

$$\begin{array}{rcl}
\mathsf{simpleNP} & \rightarrow & \mathsf{pre\text{-}modifier}^{*}\mathsf{headpost\text{-}modifier}^{*} \\
\mathsf{head} & \rightarrow & N
\end{array}$$

Such expressions form the basis of the approaches taken in the **DORO** project [62] and the Profile project [56]. Index expressions result from this simplification of noun phrases by the following simplification for pre-modifier and post-modifier:

$$\begin{array}{rcl}
\mathsf{IndexExpr} & \rightarrow & \epsilon|\mathsf{simpleNP} \\
\mathsf{pre\text{-}modifier} & \rightarrow & \mathsf{Adjective} \\
\mathsf{post\text{-}modifier} & \rightarrow & \mathsf{ConnectorsimpleNP} \\
\mathsf{Term} & \rightarrow & t \in \mathcal{T} \\
\mathsf{Adjective} & \rightarrow & a \in \mathcal{A} \\
\mathsf{Connector} & \rightarrow & c \in \mathcal{C}
\end{array}$$

term
prepositions
gerunds
connector
null connector

It should be noted that our approach is more powerful than in [20], as in our definition adjectives are also allowed. A *term* t basically corresponds to a noun, noun-qualifying adjective or noun phrase; Basically, connectors correspond to *prepositions* and *gerunds*. For instance 'by' and 'with' are prepositions, while 'traveling' and 'including' are both a gerund. A *connector* c denotes a relationship type between two terms and is basically restricted to the prepositions and the so-called *null connector*[connector ? null] which is denoted by $\cdot$. Table 8.3 shows some of the allowable connectors and the relationship types they denote.

Example 8.1.1

Suppose we have a document with the title the use of hierarchic clustering in information retrieval. *In the table of Figure 8.4 we identify the category to which each word of this title belongs. The parse tree for this sentence is shown in Figure 8.1. Figure* **??** *b shows the index expression tree for the title.*

Generally speaking, a leaf of an index expression tree consists of one or more nouns, preceded by one or more adjectives and followed by one or more *post modifier.*

Example 8.1.2

Consider the sentence hierarchic clustering in information retrieval, *with 'clustering' the head, 'hierarchic' the pre-modifier and 'in information retrieval' the post-modifier.*

Connector	*Relationship Type*	*Examples*
of	possession action-object	castle of queen pollination of crops
by	action-agent	voting by students
in, on, *etc.*	position	trees in garden
to, on, for, in	directed assoc- iation	attitudes to courses research on voting
with, ·, and	association	assistance with problems fruit · trees
as	equivalence	humans as searchers

Figure 8.3: Connector Table

Word	Category
the	determiner
use	noun
of	preposition
hierarchic	adjective
clustering	definite noun
in	preposition
information	noun as adjective
retrieval	noun

Figure 8.4: Identifying the category of words in a sentence

Figure 8.5: Parse tree for example sentence

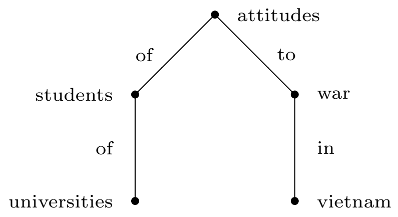

Figure 8.6: Example index expression

8.2 Notation

The general form of an index expression I is

$$I = p_1 \dots p_n h \bigotimes_{i=1}^{k} c_i(I_i)$$

In this expression $p_1 \dots p_n$ are the adjectives and h is the head, whereas $c_i, 1 \leq i \leq k$ are connectors which join subexpressions I_i to the head. This denotation also shows the relative position of a subexpression I_i with respect to a subexpression I_j. If $i < j$ then I_i appears earlier in sentence I than I_j. In order to treat adjectives as prepositions, the special connector $\cdot$ is used tranfoming the index expressions I into the following format:

$$I = h \bigotimes_{j=1}^{n} \cdot(p_j) \bigotimes_{i=1}^{k} c_i(I_i)$$

Figure 8.7 shows a graphic representation of this general form.

lead term

index expression

nested subexpression

The first term in an index expression I is referred to as the *lead term* denoted by $\lambda(I)$. (Later in this chapter the lead term will feature in many proofs of index expression properties). For notational convenience in algebraic operations, the *index expression* depicted in figure 8.7 is signified by $t_0 \bigotimes_{i=1}^{k} c_i I_i$. Note that if $k = 0$, then $I = t_0$, meaning that I is a term. Each $I_i, 1 \leq i \leq k$ is referred to as a *nested subexpression*.

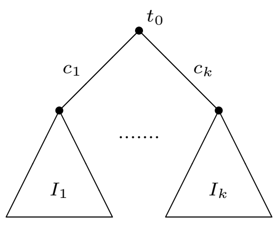

Figure 8.7: Representation of expression I

Finally, two interesting classes of index expressions are introduced. Both are depicted in figure 8.8

Definition 8.2.1

path expression

A path expression P_n *of* $n(\geq 1)$ *terms is defined as:* $P_n = t_0c_1(t_1c_2(...c_{n-1}(t_{n-1})...)$

Definition 8.2.2

umbrella expression

An umbrella expression U_n *of* $n(\geq 1)$ *terms is defined as* $U_n = t_0 \bigotimes_{i=1}^{n-1} c_it_i$

8.3 Normalization

With respect to the meaning of index expressions, it should be stressed that no meaning is assigned to the order in which the subexpressions appear, i.e. This enables us to interpret a descriptor

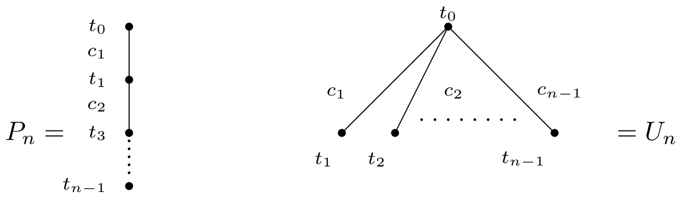

Figure 8.8: Path and umbrella expressions

Figure 8.9: Commuting subexpressions

$I_1, \ldots, I_k$ as the following index expression: Furthermore, we assume that repeating connector-subexpression pairs can be eliminated.

Note that the similarity relation *Sim* is an equivalence relation for index expressions. If we assume a total order for connectors, then this can be used to derive a normal form for expressions. We assume that the connectors appear in alphabetical order:

Figure 8.10: Order of connectors

For convenience, we assume that index expressions will always be presented in this normal form, and always be simplified as far as possible.

8.4 Expressiveness

Earlier this chapter the hypothesis was put forward that better characterization of objects will lead to more effective disclosure. One way of deciding if a language L_1 is better than a language L_2 is to compare their expressive power in terms of formal language theory. The more expressive nature of the *index expressions* over terms and term phrases will be evident from the following observation. Let $\mathcal{L}(T, C)$ denote the language of index expressions based on a set of terms T and a set of connectors C such that the null connector, $\cdot \in C$. Then, the term phrases are described by the language $\mathcal{L}(T, \{\cdot\})$ and the terms by $\mathcal{L}(T, ?)$. Now note that

index expressions

$$\mathcal{L}(T, C) \quad \supseteq \quad \mathcal{L}(T, \{\cdot\}) \quad \supseteq \quad \mathcal{L}(T, ?)$$

Also note that $\mathcal{L}(\{t\}, \{c\})$ is an infinite language whereas $\mathcal{L}(\{t\}, ?) = \{t\}$ is finite.

The index expressions can be related to the Boolean language of terms in the following way:

$$\mathcal{L}(T, C) \supset \mathcal{L}(T, \{\textsf{and}, \textsf{or}\})$$

The above relationship formally expresses Farradane's opinion that *Boolean expressions* only capture a "small part of the relations between terms which we try to indicate in language". Farradane regarded the Boolean and as "as almost completely unspecific", and the or as purely a mechanism for term replacement.

Boolean expressions

8.5 Lithoids

Building on the notion of an index expression, the so called *power index expression* is introduced. This notion bears a strong resemblance to the power set concept: the power index expression of an index expression is the set of all its index subexpressions. First, the notion of a *subexpression* of an index expression is informally introduced in terms of the graphical representation: An index subexpression of a given index expression I is an index expression represented by a subtree of the

power index expression

subexpression

tree representation of I. We will use $\Subset$ to denote this relation. Note that the relation $\Subset$ is reflexive, antisymmetric and transitive; in fact, $(\mathcal{L}(T, C), \Subset)$ is a poset.

The power index expression of a given index expression forms a lattice where the underlying ordering relation is $\Subset$. The top of the lattice is the index expression itself and the bottom is the empty index expression ϵ. The Hasse diagram of the power index expression of the index expression represented in figure 8.6 is depicted in figure 8.11.

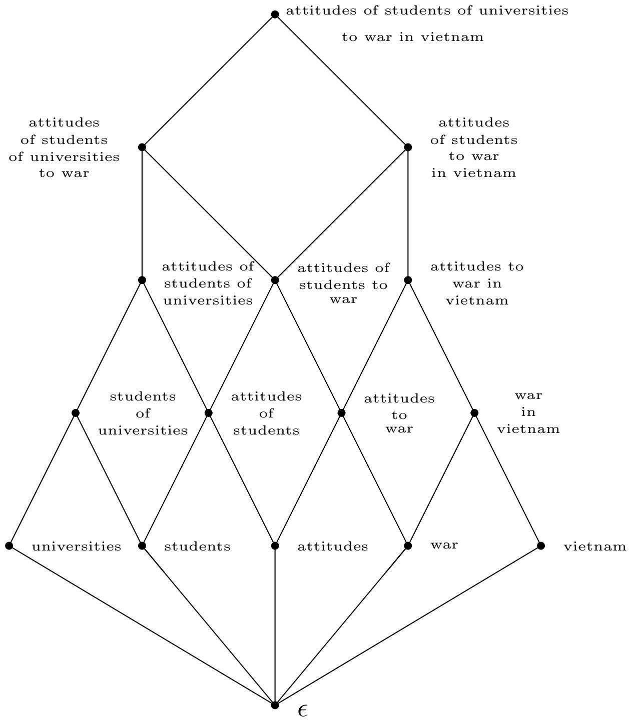
attitudes of students of universities to war in vietnam
attitudes of students of universities to war
attitudes of students to war in vietnam
attitudes of students of universities
attitudes of students to war
attitudes to war in vietnam
students of universities
attitudes of students
attitudes to war
war in vietnam
universities
students
attitudes
war
vietnam
ϵ

Figure 8.11: Example power index expression

Thus far, the power index expression of a single index expression has been considered. For a set of objects, however, a core set of index expressions is generated each of which gives rise to a power index expression. These power index expressions may have a non-trivial overlap. For example, consider the power index expressions of the index expressions effective · information · retrieval and people in need of information. (See figure 8.12). By forming the union of all power index expressions for a set of objects, that is, by taking

$$\wp(\mathcal{I}) = \bigcup_{I \in \mathcal{I}} \wp(I)$$

where $\mathcal{I}$ is the core set of index expressions, a lattice-like structure is rendered. For the index expressions mentioned above this results in the structure shown in figure 8.12. Such a structure
lithoid $(\wp(\mathcal{I}), \Subset)$ will be termed a *lithoid* because the associated diagram resembles a crystalline structure.

The lithoid forms the contents of the hyperindex generated from the characterizations from the documents of a hyperbase. (see section 5.1). One way of exploiting the lithoid for the purpose of information disclosure is as follows. If we take every vertex in the lattice as a potential focus of the searcher, then the surrounding vertices are enlargements or refinements of the context represented
lithoid by the focus. The searcher can browse across the *lithoid* by refining or enlarging the current focus until a focus is found that fits the information need. Searching in this way has been coined *Query By Navigation* (see section 5.4) ([19], [15]).

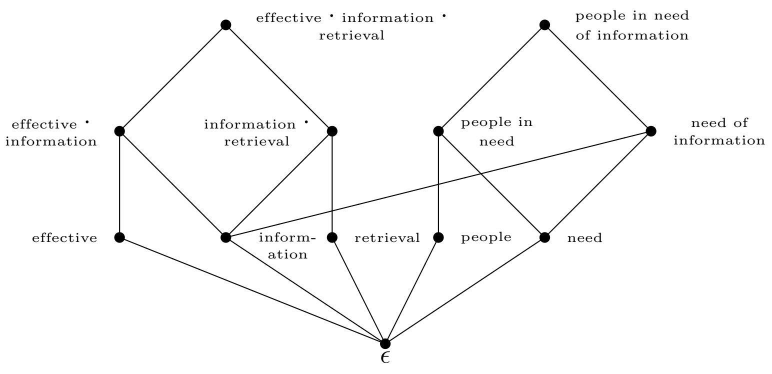

Figure 8.12: Example lithoid

8.6 Matching

8.6.1 Double discrete case

In the double discrete case the matching function for index expressions selects a document x after request I if and only if some of index expressions characterizing x contains I as a subsexpression:

$$Match[I] = \left\{ x \in \mathcal{O} \;\middle|\; \exists_{J \in \chi(x)} \left[I \subseteq J \right] \right\}$$

Note that the *Match*-function forms the basis for the beam down function (see section 5.1). Let $\mathcal{I}$ be the guide which has been selected bij the searcher during the process of Query by Navigation. Then the result of the beam down operator is defined by:

$$\Downarrow(\mathcal{I}) = \bigcup_{I \in \mathcal{I}} Match[I]$$

8.6.2 Semi-discrete case

For the semi-discrete case, a similarity measure is used:

$$Match[I](x) = Sim(I, \chi(x))$$

A comparison between index expressions can be made by considering their associated sets of *twigs*. **twig** A twig is a subexpressions of an index expression consisting of one or two nodes. So, we restrict the lithoid to all expressions at level 1 and 2 (see figure 8.12). The set of twigs associated with index expression I is denoted as $\mathsf{Twigs}(I)$. Note that $|\mathsf{Twigs}(I)| = 2n(I) - 1$. This notion is extended in the obvious way to handle sets of index expressions.

Then the similarity between two sets of index expressions may be computed by applying one of the similarity funtions from section 6.2 on their associated sets of twigs, for example:

$$Sim(\mathcal{I}, \chi(x)) = Sim_{Jac}(\mathsf{Twigs}(\mathcal{I}), \mathsf{Twigs}(\chi(x)))$$

The beam down operator will present the documents according to decreasing similarity with the current guide.

Example 8.6.1

Suppose I has lead term t, then

$$Sim(t, I) = \frac{1}{2n(I) - 1}$$

If index expressions I and J consist of the same terms, but have no connectors in common, then

$$Sim(t, I) = \frac{n(I)}{2n(I) - 1} \approx \frac{1}{2}$$

8.6.3 General case

For the general case, the function Twigs is defined as a function that, given index expression I, assigns for each (sub)expression J its internal relevancy. The internal relevancy of a subexpression is supposed to be inversely proportional to its depth of occurrence (in terms of the tree representation of index expressions): The following formula also takes handles the case that subexpressions occur more than once.

$$\mathsf{Twigs}(I)(J) = \sum_{k:\mathsf{SubExpr}(J,I,k)} \frac{1}{k+1}$$

where the expression $_{(}J, I, k)$ indicates that I has J as a subsexpression on depth k. The extension to a set of index expressions is performed by averaging the individual Twigs-functions:

$$\mathsf{Twigs}(\mathcal{I}) = \frac{1}{|\mathcal{I}|} \sum_{I \in \mathcal{I}} \mathsf{Twigs}(I)$$

The co-occurrence product for index expressions I and J indicates the degree of overlap for each subexpression and is obtained by the functional product (see appendix B.2) of the associated twigs-functions:

$$C(I, J) = |\mathsf{Twigs}(I) \bullet \mathsf{Twigs}(J)|$$

The similarity then is determined by applying the similarity functions for the general case (see section 6.3). For example:

$$\mathit{Sim}(\mathcal{I}, \chi(x)) = \mathit{Sim}_{incl}(\mathsf{Twigs}(\mathcal{I}), \mathsf{Twigs}(\chi(x)))$$

Example 8.6.2

For a single term t, the associated twigs function is:

$$\mathsf{Twigs}(t)(I) = \begin{cases} 1 & if\, J = t \\ 0 & otherwise \end{cases}$$

Let I be an indez expression, then the co-occurrence product is:

$$C(t, I) = \sum_{k:\mathsf{SubExpr}(t,I,k)} \frac{1}{k+1}$$

Suppose I contains term t only once at level k, then

$$C(t, I) = \frac{1}{k+1}$$

8.7 Automatic Indexing

index expressions

An important problem to address is how to arrive at the core set of *index expressions* from which a lithoid can be constructed. In his book Craven states that when stopwords such as the and a are omitted from the titles of documents, sections, subsections, figures etc., the resultant strings often have a form amenable to the automatic derivation of linked phrase indexes (an index expression variant) ([39]). Therefore, the characterization of an object using index expressions can be formed by the automatic derivation of such expressions from the titles of structural elements in an object. Note that titles are not always content revealing. Sometimes they are used purely for structural organization, for example, the heading *Introduction*. In that case they should be ignored.

We begin by summarizing the main algorithm from [12], which was used to index the titles of slides of an art-history library. After the removal of stopwords the remaining tokens of the title are successively processed in order to to attribute an interpretation to this expression by deriving a structure from it. The expectation is that the resultant structure corresponds to the interpretation that most searchers would expect. The underlying basis of structure detection is to consider the connectors as operators with an associated priority. The priority is used to decide whether the current structure is to be deepened, or broadened. To this end, a two level priority scheme over the connectors is employed based on the observation that some connectors bind terms more strongly than others; those that bind stronger lead to deepening in the structure. An example of how the structure detection algorithm works is depicted in figure 8.13 using The Elimination of Special Functions from Differential Equations as input and the two level priority scheme specified in figure 8.14.

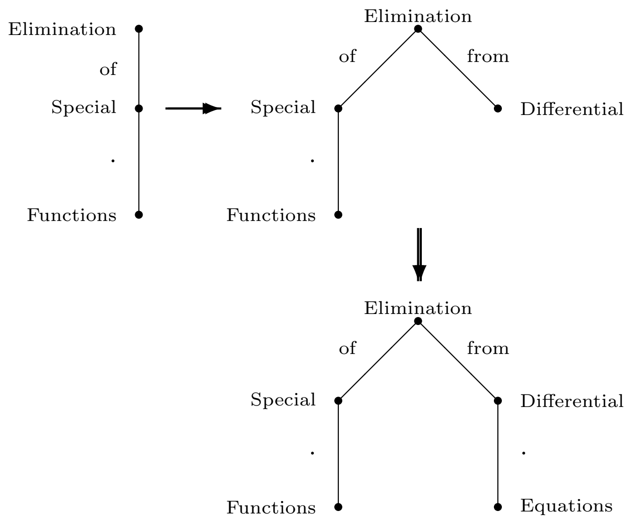

Figure 8.13: Example of structure detection

This illustration exemplifies the underlying idea behind the automatic derivation of index expressions. Attention will now be directed to specific details of the structure detection algorithm. The input is assumed to be a string S from the language T{CT}*, where C corresponds one of the allowable connectors (see figure 8.14) and T is a token from the title in question which is not a connector and not a stop word. This language corresponds to the concrete syntax of the index expressions, brackets excluded. Preprocessing of the title being indexed renders S. For example, certain stop words are removed and connector irregularities are resolved. The input string S is consumed by processing initially the first term separately and thereafter successive connector-term pairs. Parallel to this a structure conforming to the abstract syntax of index expressions is built up. To begin with, the structure is empty, that is it corresponds to the empty index expression. This empty structure is initialized using the first term t_0 taken from S. The resultant structure is an index expression comprising t_0. More formally,

$$\texttt{initexpr}(\epsilon, t_0) \quad = \quad t_0$$

The handling of a *connector-term pair* denoted c, t can be likened to the placing of a branch in the current tree structure which will be denoted by I. A branch is placed by a *broaden* or *deepen* function according to the priority of the connector. *Broaden* and *deepen* are functions that take an index expression and a connector-term pair as input and have an index expression as output. The following algorithm specifies the main body of the structure detection algorithm:

connector-term pair

Broaden deepen

```
$I \leftarrow$ initexpr($getterm(S)$)
while $S$ not processed do
    $c, t \leftarrow getconntermpair(S)$
    if $isprio0connector(c)$ then
        $I \leftarrow$ deepen($I, c, t$)
    else
        $I \leftarrow$ broaden($I, c, t$)
    fi
od
```

The function *isprio0connector* returns true if connector c is a higher priority connector. (See figure 8.14). The *broaden* and *deepen* operations are defined recursively as follows where the current structure is denoted by $t_0c_1(I_1)\dots c_k(I_k)$.

$$\begin{aligned}
\texttt{deepen}(t_0, c, t) &= t_0c(t) \\
\texttt{deepen}(t_0c_1(I_1)\dots c_k(I_k), c, t) &= t_0c_1(I_1)\dots c_k(\texttt{deepen}(I_k, c, t))
\end{aligned}$$

$$\begin{aligned}
\texttt{broaden}(t_0, c, t) &= \texttt{deepen}(t_0, c, t) \\
\texttt{broaden}(t_0c_1(I_1)\dots c_k(I_k), c, t) &= t_0c_1(I_1)\dots c_k(I_k)c(t)
\end{aligned}$$

Note that when the current structure I comprises only a term, *broaden* and *deepen* are identical operations. Furthermore, deepening occurs in the rightmost nested subexpression and broadening takes place at the root of the structure.

After an initial test of this algorithm on the titles of the CACM document collection, the above structure detection algorithm was found to produce well formed structures approximately ninety percent of the time. The priority scheme used is depicted in figure 8.14. Most of the ill formed structures involved improper handling of so called term sequences. A *term sequence* $\mathcal{T}_l$ is a path expression of length l which involves only the null connector. Term sequences are constructed by a process of continual deepening, the motivation being that null connectors bind terms strongly. Taking $\mathcal{T}_1 = t$, then $\mathcal{T}_l = \texttt{deepen}(\mathcal{T}_{l-1}, \cdot, t), l > 1$.

Priority	Connector
0	·
	about
	and
	as
	for
	including
	of
	or
	see
	with
1	are
	around
	at
	behind
	between
	by
	from
	in
	into
	is
	on
	over
	through
	to
	under
	using
	within
	without

Figure 8.14: Two level connector priority scheme

It was observed that after an initial term sequence the structure should be deepened, regardless of the priority of the connector. This is because the first connector encountered after the end of the term sequence very often refers to the last term in the sequence. For example, the structure proposed · (interpretation) in (ALGOL) is ill formed as the *interpretation* is in the language *ALGOL*, implying that interpretation and ALGOL should be related via the connector in. In other words, the path expression proposed · (interpretation in (ALGOL)) is the correct structure. The structure detection algorithm was improved to take the above problem into account.

Other ill formed structures came about because broadening at the root of the structure is not always appropriate. Consider the structure depicted in figure 8.15. If the next two tokens in the stream are the connector in and the term language, then broadening at the root establishes a relationship between the terms handling and language. Note that establishing a relationship between the terms identifiers and language is clearly more appropriate. (See figure 8.16).

On the basis of cases such as the previous example, a new broading heuristic was developed which

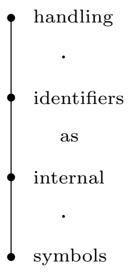

Figure 8.15: Structure before broadening

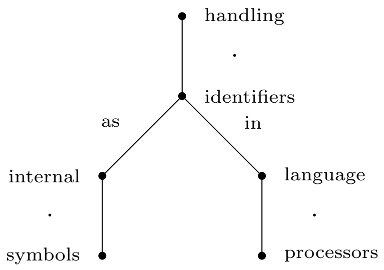

Figure 8.16: Complete structure

broadens the structure at the *father* of first term in a term sequence. More formally,

$$\texttt{broaden}(t(\Gamma)c_x(\bar{t}c_y(\mathcal{T}_l)), c_z, \hat{t}) \quad = \quad t(\Gamma)c_x(\texttt{broaden}(\bar{t}c_y(\mathcal{T}_l), c_z, \hat{t})$$

This broadening heuristic is depicted in figure 8.17. The father of the term phrase $\mathcal{T}_l$ is denoted by $\bar{t}$. If no father exists, *broadening* occurs at the root as specified earlier. **broadening**

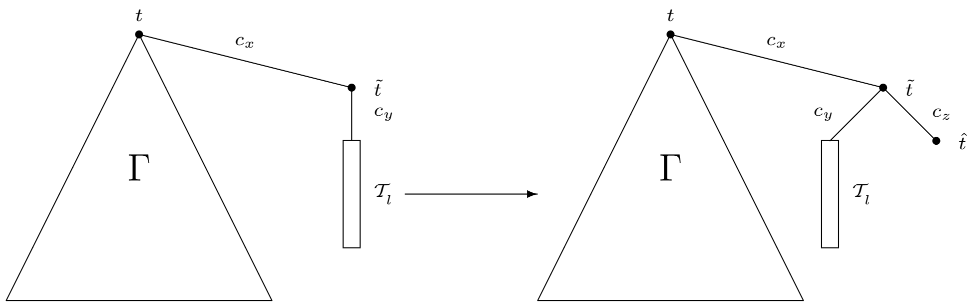

Figure 8.17: Enhanced broadening heuristic

The above improved structure detection algorithm can be implemented efficiently because only a single pass of the input string is necessary and the broadening and deepening heuristics do not involve any complex analysis. On the negative side, however, the restriction to titles results in incomplete characterizations. This restriction was taken because titles and headings are often in a form which permits a ready transformation to index expressions. The automatic derivation of index expressions from general text requires more advanced parsing techniques; a problem which was beyond the scope of this thesis. A potential solution to this problem may be found in the work of the SIMPR ESPRIT project [91]. The syntactic trees devised in that project are not unlike index expressions.

8.7.1 A note on the and connector

Consider the title *The attitudes of administrators, students and teachers to courses in universities.* The indexing algorithm of the previous section would produce the *index expression* depicted in **index expression** figure 8.18.

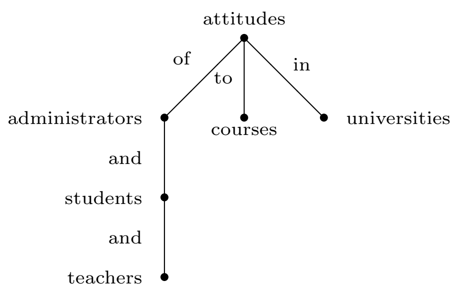

Figure 8.18: Example index expression involving the and connector

The and connector should not be confused with logical conjunction; it should be interpreted as a *linguistic* conjunctor. Furthermore, and and commas in the input string really suggest the existence of three separate index expressions, namely

attitudes of administrators to courses in universities
attitudes of students to courses in universities
attitudes of teachers to courses in universities

One way of allowing the generation of such expressions is to adopt a network representation. By way of illustration, see figure 8.19 which is taken from [40]. The disadvantage of the network representation is that it is more difficult than tree structures to derive automatically.

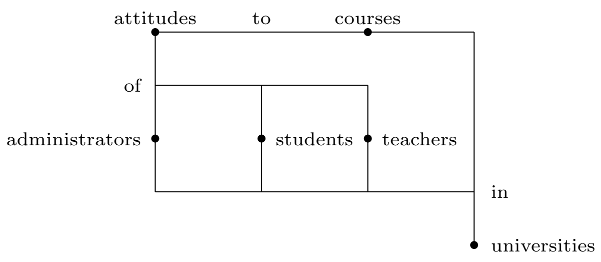

Figure 8.19: Network representation of an index expression

8.8 Properties

Given an arbitrary index expression, how large is $\wp(I)$? This is a pertinent question as the power index expression forms an integral part of an information disclosure mechanism to be featured in ensuing chapters. We begin by defining $\wp^+(I) = \wp(I) - \{\epsilon\}$, i.e., the information bearing subexpressions of I, in a way which will facilitate the counting of its elements. The theorem states that each subexpression that does not contain the root term must be part of a nested subexpression. So a subexpression is either a *lead term* (i.e. a subexpresion containing the root term) or part of a nested subexpression: The set of lead expressions is denoted as $\Lambda(I)$. is the set of lead terms of I.

lead term

Theorem 8.8.1 Power Index Expression
Let $I = t_0 \bigotimes_{i=1}^{k} c_i I_i$ be an index expression, then

$$\wp^+(I) = \Lambda(I) \cup \bigcup_{1 \leq i \leq k} \wp^+(I_i)$$

Each lead term is constructed from the root term, augmented with any combination of (possibly empty) subexpressions of the nested subexpressions:

Theorem 8.8.2 The Lead Expressions
Let $I = t_0 \bigotimes_{i=1}^{k} c_i I_i$ be an index expression, then

$$\Lambda(I) = \{t_0\} \bigotimes_{i=1}^{k} [\{\epsilon\} \cup \{c_i\} \cdot \Lambda(I_i)]$$

where $\bigotimes$ and $\cdot$ denote expression concatenation analogous to formal language theory. In order to determine the magnitude of $\wp^+(I)$ using theorem 8.8.1, the number $l(I)$ of *lead expressions* of I must be determined. From theorem 8.8.2 it directly follows that:

lead expressions

Theorem 8.8.3 How many Lead Expressions
Let $I = t_0 \bigotimes_{i=1}^{k} c_i I_i$ be an index expression, then

$$l(I) \quad = \quad \prod_{i=1}^{k} (1 + l(I_i))$$

The groundwork has now been laid to express the *size of the power index expression*.

size of the power index expression

Theorem 8.8.4 Power index expression size
Let $I = t_0 \bigotimes_{i=1}^{k} c_i I_i$ be an index expression, then

$$p(I) = l(I) + \sum_{i=1}^{k} p(I_i)$$

The result of the above theorem can be simplified further using the observation that if expression J is a term, then the set of information bearing subexpressions of J is simply the singleton set containing J, which is also by definition equal to $\Lambda(J)$. Thus by recursive application of theorem 8.8.1 an *expression* must result consisting of a union of Λ's.

expression

Lemma 8.8.1 Let $I = t_0 \bigotimes_{i=1}^{k} c_i I_i$ be an index expression, then

$$\wp^+(I) = \bigcup_{t \in \tau(I)} \Lambda(I^t)$$

where $\tau(I)$ is the set of terms (nodes) in indexexpression I. The *size of the power index expression* can thus also be calculated as follows.

size of the power index expression

Lemma 8.8.2 Let I be an index expression, then

$$p(I) = \sum_{t \in \tau(I)} l(I^t)$$

The significance of lemma 8.8.1 and lemma 8.8.2 is that they underly an efficient mechanism for generating the power index expression. The function Λ need only be implemented and unleashed over I. As there is no overlap between component lead expression sets, the union operation can be efficiently implemented by list concatenation. Such a strategy underlies the practical experiments involving power index expressions documented in this thesis.

Another useful result of lemma 8.8.2 is that it offers a simple way of calculating $p(I)$ by *hand*. For example, if I is the expression depicted in figure 8.16, then one need only determine the number of lead expressions contributed by each nested subexpression and then sum them up. This is done by working upward from the leaves and applying theorem 8.8.3. (See figure 8.20). In this case $p(I) = 10 + 9 + 2 + 2 + 1 + 1 = 25$

hand

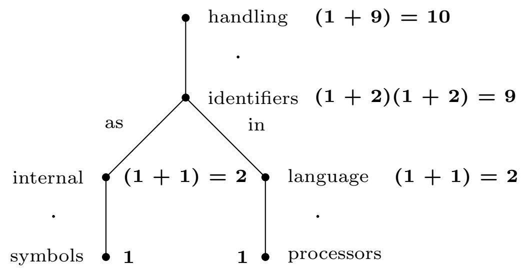

Figure 8.20: Manual determination of $p(I)$

8.8.1 The lower bound of power index expression size

From our discussions earlier we know that the size of the power index expression is in fact dependent on the number of lead expressions. Theorem 8.8.3 gives an expression for the number of lead expressions. This expression shows that the number of leads is determined by a product which, in turn, is dependent on the number of connectors connected to the lead term of I. The number of lead expressions decreases as the number of such connectors decreases. The path expression (definition 8.2.1) is the extreme of this and therefore it can be supposed that these expressions generate the smallest *power index expressions*. To prove that this is in fact the case is the goal of this section. We first begin by counting the *lead expressions* in a path.

power index expressions

lead expressions

Lemma 8.8.3 Leads of a Path
Let P_n be a path expression of n terms, then $l(P_n) = n$.

Proof:
Apply $I(P_n) = 1 + l(P_{n-1})$.

power path expression

The magnitude of a *power path expression* is established as follows.

Theorem 8.8.5 Power path expression size
Let P_n be a path expression of n terms, then $p(P_n) = \frac{n(n+1)}{2}$.

Proof:
Apply $p(P_n) = p(P_{n-1}) + l(P_n)$.

The following theorem establishes the lower bound of power index expression size. Shortly thereafter it will be shown that *power path expressions* have this lower bound.

power path expressions

Theorem 8.8.6 Lower bound of power index expression size
Let I be an index expression of n terms, then $p(I) \geq \frac{n(n+1)}{2}$

Proof:

(by induction on n, the number of terms in I)
For $n = 0$, implying $I = \epsilon$, we conclude $p(I) = 0$. For the induction step we assume

For all index expressions I of i terms, $0 \leq i \leq n$: $p(I) \geq \frac{i(i+1)}{2}$

It will now be shown that by extending expression I, which has n terms, to expression $\overline{I}$ by adding a term in an arbitrary way, the minimal size of $p(\overline{I})$ is $\frac{(n+1)(n+2)}{2}$ (See figure 8.21).

Note that a subexpression of $\overline{I}$ is either a subexpression of I or it is a subexpression containing the added term t. It is obvious that there are at least $n+1$ expressions in the latter category. As a result,

$$p(I_{n+1}) \geq p(I_n) + (n+1)\frac{n(n+1)}{2} + (n+1) = \frac{(n+1)(n+2)}{2}$$

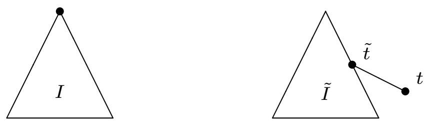

Figure 8.21: Adding a branch

8.8.2 The upper bound of power index expression size

The reason why power path expressions are minimal lies in the fact that the "fan out" of terms in the expression is minimal. In umbrella expressions (definition 8.2.2), the fan out of the lead term is maximal. This section explores *power umbrella expression* size and shows that this in fact constitutes the upper bound of *power index expression size*. As was the case with the path expressions, the *lead expressions* of an umbrella will first be counted.

power umbrella expression

power index expression size

lead expressions

Lemma 8.8.4 Leads of an Umbrella

Let U_n be an umbrella expression of n terms, then $l(U_n) = 2^{n-1}$.

Proof:

Follows directly from theorem 8.8.3 with $k = n - 1$

The magnitude of the *power umbrella expression* can now be established as follows.

power umbrella expression

Theorem 8.8.7 Power umbrella expression size

Let U_n be an umbrella expression of n terms, then $p(U_n) = 2^{n-1} + (n - 1)$.

Proof:

Apply theorem 8.8.4 and lemma 8.8.4.

The above theorem states that the fanout of the lead term produces a power index expression with a size which is exponential in size with respect to the number of terms n. This turns out to be the upper bound as the following lemma and theorem prove. The lemma proves that the promotion of a *nested subexpression* never generates less expressions in the power index expression.

nested subexpression

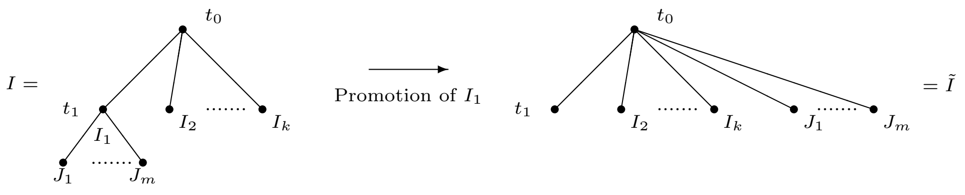

Figure 8.22: Promotion of a Nested Subexpression

Lemma 8.8.5 Promotion

Let $I = t_0 \bigotimes_{i=1}^{n} c_i I_i$ and $I_1 = t_1 d_1(J_1) \dots d_m(J_m)$,
and $\overline{I} = t_0 c_1(t_1) \;\; c_2(I_2) \dots c_n(I_n) \;\; d_1(J_1) \dots d_m(J_m)$, then $p(\overline{I}) \geq p(I)$

Proof:

It will be shown that $p(\overline{I}) - p(I) \geq 0$. (See figure 8.22). For I we have:

$$\begin{aligned} p(I) &= l(I) + \sum_{i=1}^{k} p(I_i) && \langle \texttt{Theorem 8.8.4} \rangle \\ &= l(I) + p(I_1) + \sum_{i=2}^{k} p(I_i) \\ &= l(I) + l(I_1) + \sum_{i=1}^{m} p(J_i) + \sum_{i=2}^{k} p(I_i) && \langle \texttt{Theorem 8.8.4} \rangle \end{aligned}$$

For $\overline{I}$ we can derive:

$$p(\overline{I}) = l(\overline{I}) + p(t_1) + \sum_{i=2}^{k} p(I_i) + \sum_{i=1}^{m} p(J_i) \qquad \langle \texttt{Theorem 8.8.4} \rangle$$

Therefore

$$p(\overline{I}) - p(I) = l(\overline{I}) + 1 - l(I) - l(I_1)$$

For simplification we introduce $X = l(I_1)$. Focussing on counting the respective lead expressions

$$\begin{aligned} l(\overline{I}) &= (1+1)\prod_{i=2}^{k}(1+l(I_i))\prod_{i=1}^{m}(1+l(J_i)) \qquad \langle \texttt{Theorem 8.8.3} \rangle \\ &= 2YX \end{aligned}$$

where Y is a shorthand for the first product. Futhermore:

$$\begin{aligned} l(I) &= \prod_{i=1}^{k}(1+l(I_i)) \qquad \langle \texttt{Theorem 8.8.3} \rangle \\ &= (1+l(I_1))\prod_{i=2}^{k}(1+l(I_i)) \\ &= (1+X)\prod_{i=2}^{k}(1+l(I_i)) \\ &= (1+X)Y \end{aligned}$$

Combining these two reults, we get:

$$\begin{aligned} p(\overline{I}) - p(I) &= 2XY - (1+X)Y - X + 1 \\ &= 2XY - Y - XY - X + 1 \\ &= XY - X - Y + 1 \\ &= X(Y-1) - (Y-1) \\ &= (X-1)(Y-1) \end{aligned}$$

The last result implies $p(\overline{I}) - p(I) \geq 0$ as X, which denotes the number of lead expressions of I_1 is certainly greater than zero and for similar reasons $Y > 0$. (See figure 8.22).

Corollary 8.8.1 Power umbrella expressions are maximal
Let I be an index expression of n terms, then

$$p(I) \leq 2^{n-1} + n - 1$$

Proof:
This follows from lemma 8.8.5 and theorem 8.8.7. Continual promotion in an arbitrary index expression will result in an umbrella expression.

8.8.3 The bounds of growth of power index expressions

Theorem 8.8.6 and corollary 8.8.1 give the bounds of the power index expression size. These bounds are depicted graphically in figure 8.23.

8.9 Growth of Lithoids

Recall that a lithoid is a union of power index expressions. From a practical standpoint it is important to have some indication of the growth characteristics of *lithoids*. In this section growth

lithoids

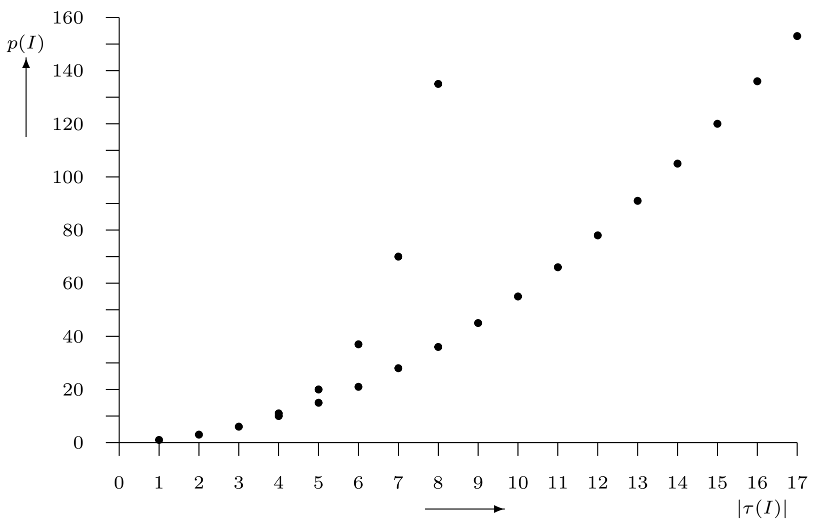

Figure 8.23: Power index expression growth

analyses of two lithoids are featured. The first lithoid was derived from the CACM collection of 3204 documents and the second from the first 750 documents of the Cranfield collection. These document collections are often used for research in information disclosure. In both cases, the titles of the documents in the collection were extracted and indexed using the improved algorithm of section 8.7. In each case a core set $\mathcal{I}$ of index expressions resulted from which a lithoid was constructed.

The theoretical arguments of the previous section show that umbrella expressions spawn the largest power index expressions. The upper bound of lithoid size, therefore, is if $\mathcal{I}$ consists solely of umbrella expressions and moreover that there is no overlap between respective power umbrella expressions. Assuming that on average a core index expression consists of x terms and there are m core expressions ($|\mathcal{I}| = m$), the upper bound of lithoid size can be approximated as follows:

$$\begin{aligned}
\left|\bigcup_{U\in\mathcal{I}} \wp(I)\right| &= \quad \langle \texttt{no overlap} \rangle \\
& \sum_{U\in\mathcal{I}} |\wp(U)| \\
&= \sum_{U\in\mathcal{I}} p(U) \\
&\approx \quad \langle \texttt{thm:maxpowerexp} \rangle \\
& \sum_{1\leq i\leq m} 2^{x-1} + x - 1 \\
&\approx m2^{x-1} + mx - m
\end{aligned}$$

We will refer to this upper bound as the worst case.

An important question is how the actual size of a typical lithoid compares with the worst case. During the construction of the Cranfield lithoid, its size was recorded after the processing of 200, 400, 600 and 750 titles. In addition, a value of $x = 7.5$ was calculated for the Cranfield titles so the worst case could be analyzed. On the basis of these measurements, growth curves can be plotted. These are depicted in figure 8.24. This figure shows that the actual growth is between the worst case and the growth of the unary expressions (the terms). It is interesting to compare the actual growth with the term growth because most current disclosure systems use only terms. In this way an indication can be gained of the space overhead of a lithoid based disclosure system and a term based equivalent. Note that from roughly 1330 terms more than 26000 index expressions have been generated. From the above figure we conclude that the speed at which the lithoid grows is worrying from a storage overhead perspective[1]. Similar growth was experienced in [12] in the context of an art history lithoid.

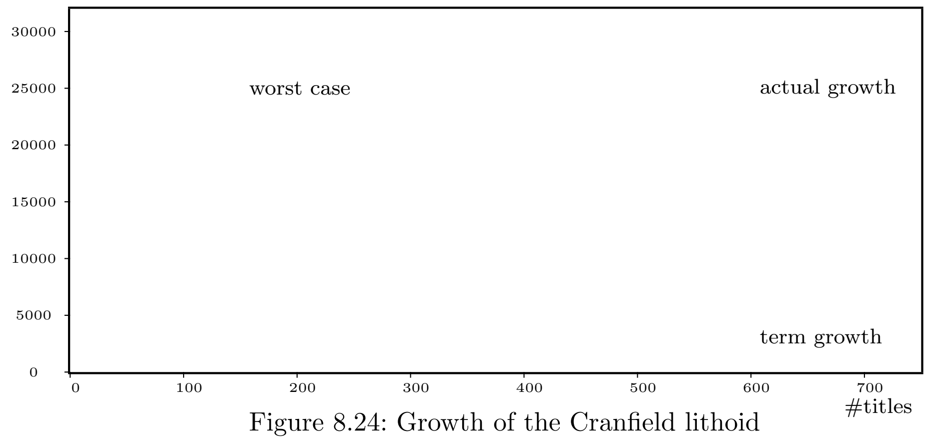

Figure 8.24: Growth of the Cranfield lithoid

The growth of the CACM lithoid proved markedly less than that of the Cranfield but is nonetheless substantial. (See figure 8.25) The reason for the difference is that the titles from the CACM collection contain on average 3.5 terms as opposed to 7.5 in the Cranfield collection. On average a power index expression in the CACM lithoid contained only ten expressions compared to 41 expressions in the Cranfield lithoid.

The distribution of the index expressions over the terms is interesting because it reflects the general topology of the lithoid. Figure 8.26 shows that binary and ternary index expressions are easily the most frequently occurring in the CACM lithoid, whereas the Cranfield lithoid has a flatter peak. This is due to the Cranfield titles being longer. Keep in mind that the height of the peaks is not relevant due to the different sizes of the respective core index expression sets.

One would expect that there would be some overlap between the power index expressions that comprise a given lithoid because it is extremely unlikely that titles in a document collection don't share some terms. Such overlap can be exploited for information disclosure as it furnishes the possibility to establish a relationship between expressions, in particular between a characterization and a request. In chapter 3, we will show how expression overlap within a lithoid can be used to drive context free plausible inference over index expressions. Furthermore, a shared expression resides in different contexts denoted by the expressions which it is part of. Showing these contexts to the searcher is a useful guide to help him or her clarify their information need. This aspect will be dealt with in length in chapter 4. In order to scrutinize the overlap factor in a lithoid, the concept of the *uniqueness* of an index expression is introduced. An *index expression* is termed *unique* if it is a subexpression of a single core expression.

index expression

Figure 8.27 depicts the uniqueness of index expressions against the number of terms they contain. It is not surprising that uniqueness increases in relation to the number of the underlying terms because the specificity of an index expression increases with its length, thereby reducing the likelihood of it being shared. Note that uniqueness increases rapidly. This opens possibilities for optimizing the storage overhead of the lithoid. Basically, unique index expressions which are not core expressions are redundant and could be ignored by the plausible inference mechanism. This point will be further addressed in chapter 5. In both lithoids, more than ninety percent of ternary index expressions are unique. As a final note, [85] contains additional analyses of the CACM and Cranfield lithoids.

[1]Out of this concern, a strategy was developed to dynamically generate the necessary part of the lithoid according to the current context built up by the searcher. This strategy was incorporated in the workbench produced by Esprit project APPED.

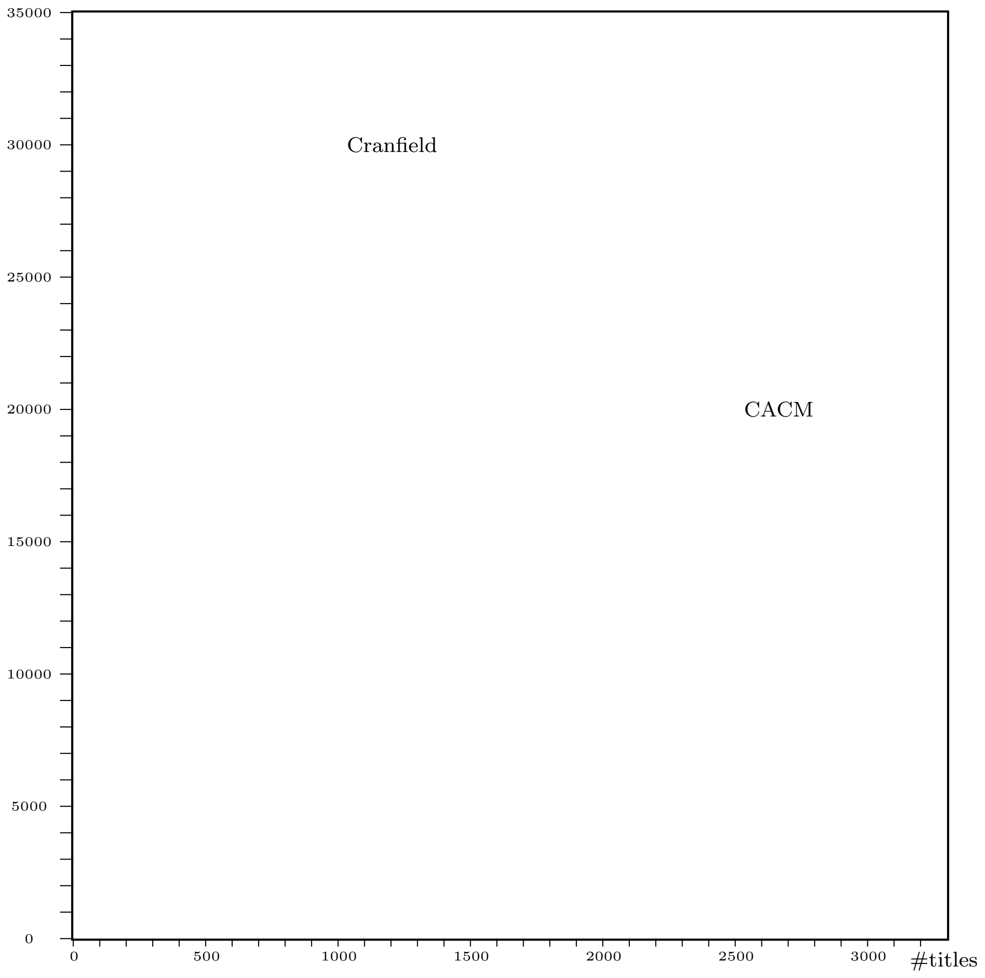

Figure 8.25: CACM vs. Cranfield growth

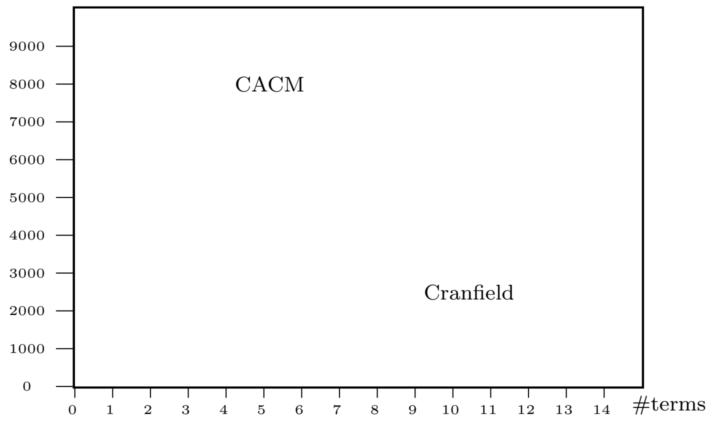

Figure 8.26: Distribution of expressions over terms

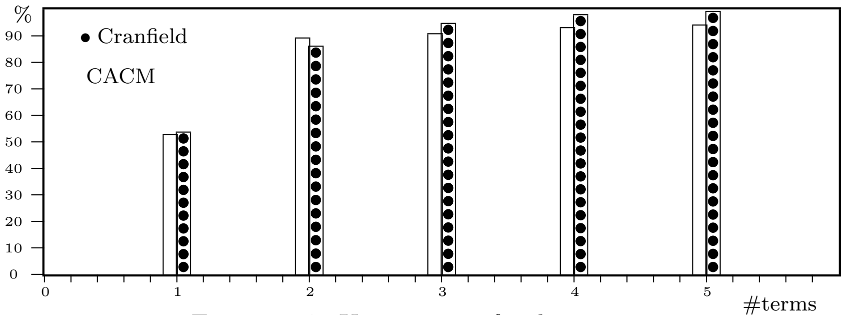

Figure 8.27: Uniqueness of index expressions

Chapter 9

Probabilistic Retrieval

In this chapter we discuss Probabilistic Retrieval according to [46].

9.1 The Binary Independence Retrieval Model

A typical problem in Information Retrieval is that a query may not exactly match the information need of the searcher. As a result, searchers may judge differently about the relevance of a document for some query. Therefore we focus at the probability of a document d being judged relevant by a random searcher, for query q. Note that this approach is based on comparing the information need N with the matching function *Match*, rather than comparing the normative information need *Norm* with *Match*. In this probability the following stochastic variables are involved:

1. a stochastic variable r ranging over the set of possible relevance judgements: $\{rel, nonrel\}$.
2. a stochastic variable d which takes document as its value
3. a stochastic variable q for queries.

This leads to the probability $Prob(r, d, q)$. The Information Retrieval system has to decide about relevance for a given document and query, and thus has to estimate:

$$Prob(rel \mid d, q)$$

In order to estimate these probablities, the Information Retrieval system bases itself on the characterization of the documents. This leads to the introduction of the stochastic variable x which takes document characterizations as its value.

$$Prob(rel \mid d, q) \quad \text{is estimated by} \quad Prob(rel \mid x, q) \quad \text{where } x = \chi(d)$$

This probability will be estimated from the distribution of characterization terms over documents in the collection (this distribution usually is determined during the indexing process). This (global) distribution reflects the general interpretation of how terms relate to documents. The personal interpretation of the searcher can be involved by asking the searcher for relevance feedback. For the distribution of terms over documents the *cluster hypothesis* ([76]) is assumed:

cluster hypothesis

terms are distributed differently over relevant documents than over nonrelevant documents.

Note that $Prob(t \mid rel, q)$ is the distribution of term t over documents that are relevant for query q, and $Prob(t \mid nonrel, q)$ the distribution of t over nonrelevant documents.

double discrete case

In the double discrete case, the Information Retrieval system has only to make the decision whether a document should be retrieved or not. A simple strategy is: retrieve the document if its probability of being relevant is greater than its probability of being irrelevant. This decision criterion, referred to as $D1$, can be formulated as:

$D1$

$$D1(x,q) \equiv Odds(rel\,|x,q) > 1$$

The quality of a decision rule can be quantified as the expected probability of making a bad decision. Two kinds of bad decisions can be made:

noise

1. *noise*: an irrelevant document is estimated to be relevant.

omission

2. *omission*: a relevant document is estimated to be irrelevant.

expected error probability

The expected probability for any decision rule X of making such an error for query q amounts to:

$$\begin{aligned} Prob(Error_X\,|q) &= \sum_x Prob(Error_X\,|x,q)\,Prob(x) \\ &= \sum_{x:X(x,q)} Prob(nonrel\,|x,q)\,Prob(x) + \sum_{x:\neg X(x,q)} Prob(rel\,|x,q)\,Prob(x) \end{aligned}$$

optimality

where the first summation covers noise, while the second summation quantifies omission. We will show that decision rule $D1$ minimizes the expected error. In order to prove this, let rule X any other decision rule that takes the same decisions as rule $D1$, except for documents with characterization c. Then the difference in expected error for these two rules is determined by the difference for characterization c:

$$\begin{aligned} & Prob(Error_X\,|q) - Prob(Error_{D1}\,|q) \\ &= \sum_x \left(Prob(Error_X\,|x,q) - Prob(Error_{D1}\,|x,q)\right) Prob(x) \\ &= \left(Prob(Error_X\,|c,q) - Prob(Error_{D1}\,|c,q)\right) Prob(c) \end{aligned}$$

With respect to rule $D1$ we have two cases:

1. *document c is retrieved by* $D1$
   In this case $Odds(rel\,|c,q) > 1$. The probability of making this error is $Prob(nonrel\,|c,q)$. Document c is *not* retrieved by rule X, so rule X being in error amounts to $Prob(rel\,|c,q)$. As a result in this case:

   $$Prob(Error_X\,|c,q) - Prob(Error_{D1}\,|c,q) = Prob(rel\,|c,q) - Prob(nonrel\,|c,q) > 0$$

2. *document c is not retrieved by* $D1$
   In this case $Odds(rel\,|c,q) \leq 1$. Analogously to the previous case, we find

   $$Prob(Error_X\,|c,q) - Prob(Error_{D1}\,|c,q) = Prob(nonrel\,|c,q) - Prob(rel\,|c,q) \geq 0$$

Combining both cases, we conclude:

$$Prob(Error_X\,|q) \geq Prob(Error_{D1}\,|q)$$

We conclude that decision rule $D1$ minimizes the average probability of making a bad decision.

Next we analyse rule $D1$ within the context of the Binary Independence Retrieval Model (BIR). By applying Bayes' rule on odds, we get:

$$Odds(rel\,|x,q) = Odds(rel\,|q)\,\frac{Prob(x\,|rel,q)}{Prob(x\,|nonrel,q)}$$

By this transformation we isolate the uncertain behavior of the searcher. The factor $Odds(rel\,|q)$ is constant when we are confronted with a searcher having issued request q. The Information Retrieval System will inspect documents and will try to estimate probability characteristics for this searcher and this request. Note that $Prob(x\,|rel,q)$ is the probability distribution for documents that the searcher has judged to be relevant for query q, and $Prob(x\,|nonrel,q)$ the analogous distribution for nonrelevant documents. The Binary Independence Retrieval Model is based on the assumption that keywords are more or less independent of each other. This assumption obviously is too weak, but provides a first approach to a probabilistic retrieval strategy. In later sections we will see how this assumption may be relaxed. The weakest version of the independence assumption is called the *linked dependence assumption* ([33]):

linked dependence assumption

$$\frac{Prob(x \mid rel, q)}{Prob(x \mid nonrel, q)} = \prod_{i=1}^{t} \frac{Prob(x_i \mid rel, q)}{Prob(x_i \mid nonrel, q)}$$

where it is assumed that the set of terms is $\{T_1, \ldots, T_t\}$ and $x_i = x(T_i)$.

Exercise 9.1.1

Show that the linked dependence assumption holds if all keywords are mutually independent.

Using the linked dependence assumption, the odds for a document can be further elaborated:

$$Odds(rel \mid x, q) = Odds(rel \mid q) \prod_{i=1}^{t} \frac{Prob(x_i \mid rel, q)}{Prob(x_i \mid nonrel, q)}$$

As we assumed a discrete characterization, each x_i can be either 0 or 1. Let p_i be the probability that a document which is found to be relevant by the searcher, actually has assigned term T_i in its characterization, or, $p_i = Prob(x_i = 1 \mid rel, q)$. Let q_i be the corresponding probability for nonrelevant documents, or, $q_i = Prob(x_i = 1 \mid nonrel, q)$. For terms that do not occurr in the query $p_i = q_i$ is assumed. Then we have:

$$\frac{Prob(x_i \mid rel, q)}{Prob(x_i \mid nonrel, q)} = \frac{p_i^{x_i}(1-p_i)^{1-x_i}}{q_i^{x_i}(1-q_i)^{1-x_i}}$$

This result is easily verified by successively substituting 0 and 1 for x_i. As a result we get:

$$Odds(rel \mid x, q) = Odds(rel \mid q) \prod_{i=1}^{t} \frac{p_i^{x_i}(1-p_i)^{1-x_i}}{q_i^{x_i}(1-q_i)^{1-x_i}}$$

The factor $Odds(rel \mid q)$ does not depend on characterization x, and is therefore irrelevant for the judgement of the relevancy of characterization x for query q. So we focus on the second factor. This expression can be simplified by taking the logarithm. This results in:

$$\begin{aligned} &\sum_{i=1}^{t} \left[x_i \log\left(\frac{p_i}{q_i}\right) + (1-x_i)\log\left(\frac{1-p_i}{1-q_i}\right)\right] \\ = &\sum_{i=1}^{t} x_i \log\left(\frac{p_i}{q_i}\frac{1-q_i}{1-p_i}\right) + \sum_{i=1}^{t} \log\left(\frac{1-p_i}{1-q_i}\right) \end{aligned}$$

The latter term of this expression does not contain the term x_i, and thus is independent of the (characterization of the) document in question. The first term thus suffices to find the differences between documents. This term is called the *retrieval status value* of the document for query q:

retrieval status value

$$\begin{aligned} RSV(x|q) &= \sum_{i=1}^{t} x_i \log\left(\frac{p_i}{q_i}\frac{1-q_i}{1-p_i}\right) \\ &= \sum_{T_i \in x} \log\left(\frac{Odds(p_i)}{Odds(q_i)}\right) \end{aligned}$$

The retrieval status value forms also the basis for the semi discrete retrieval approach of the Binary Independence Retrieval Model.

semi discrete case

The estimation of the parameters p_i and q_i can be done in several ways. First, a thorough analysis (see 7.3) can be done, leading to an estimation of these probabilities.

A more elegant way is to interactively estimate these probabilities by relevance feedback. The Information Retrieval system in this case repeatedly retrieves a test set of f documents and asks the searcher to provide relevance judgements for these documents. Let r documents be marked as being relevant by the searcher. The system then tries to estimate the term probabilities:

$$\begin{aligned} f_i &= \text{number of documents in test set containing term } T_i \\ r_i &= \text{number of relevant documents containing term } T_i \end{aligned}$$

Then the parameters can be estimated as follows:

$$\begin{aligned} p_i &= \frac{r_i}{r} \\ q_i &= \frac{f_i - r_i}{f - r} \end{aligned}$$

These estimates may be improved after the next feedback iteration.

9.2 An impression of BIR's performance

In order to have an idea of the potentials of the Binary Independence Retrieval Model, we show its behaviour in a theoretical example. In this example, we consider an archive of 100 documents, numbered $1, \ldots, 100$. We choose the numbers $0, \ldots, b-1$ (with $b = 94$) as indexing terms. The index terms associated with document i are obtained as the first 10 terms from the b-adic expansion of $\frac{1}{i}$. Some examples are:

i	d_0	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9
16	0	5	23	47	82					
17	5	11	22	44	49	71	82	88		
18	5	20	52	83						
34	2	5	11	22	44	49	71	82	88	
53	1	5	14	17	23	30	37	67	69	72
63	1	5	23	46	82	91				
64	0	1	5	23	44	47	82			
66	1	5	14	22	39	56	65	74	82	91
68	1	5	11	22	35	44	49	71	82	88
69	1	5	17	21	34	42	66	70	74	87
75	1	5	20	23	38	42	57	61	76	80
89	1	2	4	5	10	21	26	38	52	76
98	0	5	15	32	51	57	69	74	76	90

In this example, we see all 13 documents that are characterized by term 5. This set of documents will be the information need of the experiment. As a result, $n = 100$ and $m = 13$. The initial distribution of relevant documents over the collection is presented as the Precision-Recall function in figure ??. This figure indicates a bad positioning of relevant documents. The normative recall is 0.4350, slightly worse than the average normative recall (0.5). The distribution of relevant documents is as shown on the following axis (where a star represents a relevant document, and a dot a nonrelevant one).

```
....5...10...15...20...25...30...35...40...45...50...55...60...65...70...75...80...85...90...95..100
....+....|....+....|....+....|....+....|....+....|....+....|....+....|....+....|....+....|....+....|
...............***...............*..................*.........**.*.**.....*.............*........*..
```

The Information Retrieval system starts with taking a random sample of 15 documents, and ask the searcher to mark which documents of this sample are relevant. The sample appears to contain 3 relevant documents. After the computation of the retrieval status value, the improved ordering of documents is displayed in figure 9.1. The precision-recall function shows a significant improvement of the positions of relevant documents. The normative recall now amounts to 0.7905. The new distribution of elements:

```
....5...10...15...20...25...30...35...40...45...50...55...60...65...70...75...80...85...90...95..100
....+....|....+....|....+....|....+....|....+....|....+....|....+....|....+....|....+....|....+....|
*****.*.....*...........*..*..............*....*............*..........................*............
s.ss.s.....s..........s.......s....s....s....s............s...............ss.......s.s..............
```

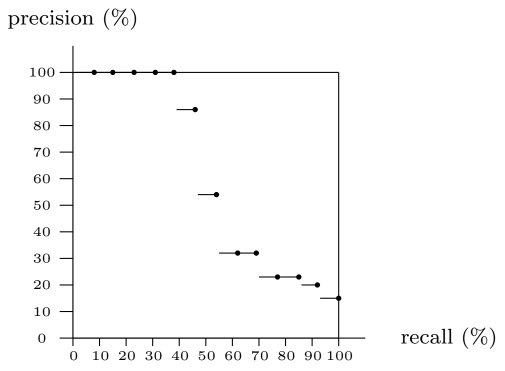

Figure 9.1: Precision-Recall function after random sample

The Information Retrieval systems presents the 15 documents with highest retrieval status value to the searcher, and once more asks the searcher to provide relevance feedback. After processing the feedback information, the ordering of documents is improved to the situation in figure 9.2. From this figure we see that the distribution of relevant documents has improved slightly. The normative recall has increased to 0.8090. The distribution of documents:

```
....5...10...15...20...25...30...35...40...45...50...55...60...65...70...75...80...85...90...95..100
....+....|....+....|....+....|....+....|....+....|....+....|....+....|....+....|....+....|....+....|
******.*...........***...................................*.....*............................*.......
ssssss..s.s.s....sss...s....s....s....s......s.s....s.......s........ss.....s.........s........s....
```

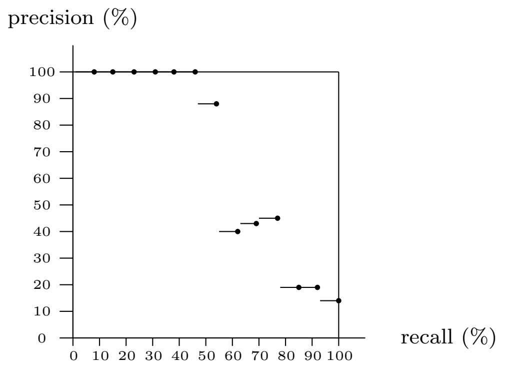

Figure 9.2: Precision-Recall function after next step

By continuing this proces of incorporating user feedback, the successive values of the normative recall value are found in the following table:

step	$Recall_{\text{norm}}$
0	0.4350
1	0.7905
2	0.8090
3	0.8462
4	0.8515
5	0.8515
6	0.8515
7	0.8515
8	0.8515
9	0.8515

We see that the normative recall has not improved in the latter iterations. An obvious reason is that the set of documents, which is presented to the searcher, contains a decreasing number of documents with unknown relevancy. Thus the growth of the knowledge of the Information Retrieval system of the intentions of the searcher is also decreasing, making each further step less effective. The situation after the final step is shown in figure 9.3. The function is very similar to figure 9.2.

```
....5...10...15...20...25...30...35...40...45...50...55...60...65...70...75...80...85...90...95..100
....+....|....+....|....+....|....+....|....+....|....+....|....+....|....+....|....+....|....+....|
********..........**......*......................................*..........................*........
sssssssssssssssss......s.s...ss.....s.s.......s.s......s.........s.....s......s.....ss..........s...
```

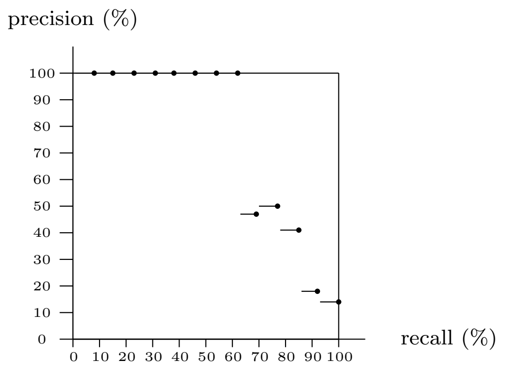

Figure 9.3: Precision-Recall function after final step

Chapter 10

Plausible Inference

In the section 8 index expressions were introduced as a characterization mechanism resulting in more complete object characterizations than keyword, or term phrase based characterizations. The matching of index expressions (see section 8.6) was based on a comparisons of the elementary components of such trees. In this section a matching mechanism for index expressions based on the concept of *plausible inference* is introduced. In this approach, the expressions that characterize a document are considered as a logical theory, i.e.,a set of axioms. The matching mechanism tries to find the most plausible inference of the query from this set of axioms. During plausible inference, the matching mechanism now and then has to make a (plausible) assumption in order to proceed in deive the query from the axiom.

plausible inference

There are three rules of strict inference called *modus continens*, *modus generans* and *modus substituens*. In addition, there are two rules of plausible inference; *refinement* and *plausible substitution*.

After the relevance of each document has been derived, the associative links (see section 5) will also be taken into account in section 10.5. The idea is that if a higly relevant document refers to another document, then this document will also be relevant to some extent. This is referred to as *spatial coherence*.

spatial coherence

10.1 Strict Rules

Strict rules of logical inference transform index expressions into equivalent expressions.

10.1.1 Modus Continens

The intuition behind *modus continens* is (logical) deduction by way of containment. Consider the following example. Suppose an information object has the index expression

modus continens

pollution of rivers

as an axiom. We may then also say that the object is about pollution. This is because the information conveyed by pollution is also inherent in pollution of rivers. But, the document is also about rivers with an analogous argument!

Modus continens is formally defined as follows:

Definition 10.1.1
If J is a subexpression of I, then J can be deduced from I, or:

$$\frac{J \subseteq^{*} I, I}{J}$$

Note that there is an analogy here with the rule *modus ponens*, which states that if we have proven both p and $p \Rightarrow q$, then we can infer the validity of q.

10.1.2 Modus Equivalens

modus equivalens The intuition behind *modus equivalens* is deduction by way of synonym. If an information object is about index expressions I, then it is also about index expressions that result by replacing some term of that expression by a synonym of that term. Modus equivalenc is formally defined as follows:

Definition 10.1.2

Given terms I and J satisfy the thesaural relation I SYN J then modus equivalence *affirms J can be deduced from I, or:*

$$\frac{I \text{ SYN } J, I}{J}$$

This rule could be augmented to handle equivalence of connectors also.

10.1.3 Modus Generans

modus generans The intuition behind *modus generans* is deduction by way of generalization. This rule of inference is based on the assumption that a more general term can be deduced from any of its specializations. For example, given the following generalization salmon ISA fish, then any information object that deals with salmon also implicitly deals with fish.

Modus Generans is formally defined as follows:

Definition 10.1.3

Given terms I and J satisfy the thesaural relation I ISA J then modus generans *affirms J can be deduced from I, or:*

$$\frac{I \text{ ISA } J, I}{J}$$

10.1.4 Modus Substituens

The intuition behind *modus substituens* is deduction by way of substitution. From a previous example we know that pollution is deducible from pollution of rivers. From that we may conclude that any object that is about the effects of POLLUTIONOFRIVERS in Australia is also about the effects of POLLUTION in Australia. Formally:

Definition 10.1.4

Suppose index expression K contains index expression I as a subexpression and $K[I := J]$ is the expression K with I substituted by J, then modus substituens *affirms:*

$$\frac{I \vdash J, K}{K[I := J]}$$

10.1.5 Proofs involving Index Expressions

The four rules of inference *modus continens*, *modus equivalens*, *modus generans* and *modus substituens* provide the driving mechanism with which an index expression can be proven from other index expressions. Such a proof has the character of transforming an index expression into another in a step-wise fashion. This transformation process is similar to the game in which one has to transform one word into another word by successively changing single letters, for example, we can put the cat into bed as follows: cat, bat, bad, bed.

Basically, the strict rules of inference derive from an index expression all its subexpressions, where terms may be replaced by equivalent of more general terms (see section 3.6 for properties of te thesaurus relations SYN and ISA):

Theorem 10.1.1

$$I \vdash J \iff \exists_{s_1,\dots,s_n,t_1,\dots,t_n} \left[J[s_1 := t_1; \dots s_n := t_n] \mathrel{\subset\!\!\!\ast} I \wedge \forall_i \left[t_i \;\textsc{syn}\; s_i \vee t_i \;\textsc{isa}\; s_i \right] \right]$$

10.2 The Hook Theorem

In fishing the goal is to catch the biggest possible fish with your hook. This analogy is relevant in information disclosure in the sense that one tries to catch relevant information objects. In the context of information disclosure each of the axioms can be considered a hook which can be used to catch the associated information object.

The hook theorem states that if the axioms are in the form of index expressions then the relevance of an object can be established by proving the request from a *single* characterization of the object.

Theorem 10.2.1 If $\chi(D) \vdash q$ then $I \vdash q$ for some $I \in \chi(D)$

The proof follows directly from theorem 10.1.1. Note that an object may be caught by different hooks and the proofs associated with these hooks will vary in length. The best hook for an object in relation to a particular request is the one in which the associated proof is the shortest. This issue will be dealt with more fully later in this document.

10.3 Plausible Rules

The rules for plauible inference mak it possible to construct new index expressions from those that have already ben proven relevant.

10.3.1 Plausible Refinement

The key question in regard to plausible deduction is how the assumptions should be added to the axiom set in order to strengthen it. If the axioms have the form of index expressions, existing axioms are *strengthened* by an operation called *refinement*. Refining of index expressions has to do with making them more specific. This is typically achieved by adding a *connector-term* pair as is demonstrated by the following example: Given the index expression courses. This can be refined into courses of students which can in turn be refined into attitudes to courses of students.

The above refinements are a direct result of the $\subset\!\!\!\ast$ relation over the index expressions. Refinement can also occur via the inverse of the ISA relation between terms. For example, the expression fish can be refined into salmon.

In more precise terms an index expression I can be refined into an expression J, denoted $I \twoheadrightarrow J$, if and only if

$$I \mathrel{\subset\!\!\!\ast} J \quad \wedge \quad \forall_K \left[I \mathrel{\subseteq\!\!\!\ast} K \mathrel{\subseteq\!\!\!\ast} J \Rightarrow K = J \vee K = I \right] \tag{10.1}$$

$$\vee \tag{10.2}$$

$$I \;\textsc{isa}\; J \quad \wedge \quad \neg\exists_K \left[I \;\textsc{isa}\; K \;\textsc{isa}\; J \right] \tag{10.3}$$

The refinement operation can be considered as the basis of a plausible inference mechanism. In the simplest case there is *plausible inference through refinement*. Assume for the moment that within the set of information objects characterized by pollution there are objects that deal with the pollution of rivers. On the basis of this we can deduce pollution of rivers from pollution with a certainty that is proportional to the ratio of objects about pollution of rivers to the objects about pollution. For the moment we will not consider the certainty of the deduction, only the fact that it is plausible. That is, pollution of rivers can be *plausibly* deduced from pollution, or, formally:

Definition 10.3.1

Index expression J can be plausibly deduced from expression I via refinement*, denoted $I \mathrel{|\!\sim} J$, if $I \twoheadrightarrow J$*

10.3.1.1 Plausible Substitution

Plausible deduction by refinement can be extended into a rule of deduction by *plausible substitution.* This rule is similar to *modus substituens*

Definition 10.3.2

$$I \mathrel{|\!\sim} J \Rightarrow K \mathrel{|\!\sim} K[I := J]$$

10.3.2 Proofs involving Plausible Inference

A proof involving plausible inference starts with the characterization of a document x as a set of axioms, and aims at proving an expression q, i.e., making it paluible in the context of $\chi(x)$. From the axioms other relevant expressions can be derived, using rules of inference. This can be seen as extending the initial set of relevant expresions step by step. The proof is completed as soon as the expression q is contained within the set of relevant expressions. This is denoted as $\chi(x) \mathrel{|\!\sim} q$. If the shortest proof of q from $\chi(x)$ involves n inference steps, then this is denoted as $\chi(x) \mathrel{|\!\sim}^{n} q$. We will denote $\{I\} \mathrel{|\!\sim} J$ also as $I \mathrel{|\!\sim} J$.

For example, pollution of rivers $\mathrel{|\!\sim}$metals, as:

pollution of rivers
$\vdash$ ⟨`modus continens`⟩
pollution
$\mathrel{|\!\sim}$ ⟨`refinement`⟩
pollution from metals
$\vdash$ ⟨`modus continens`⟩
metals

The following example demonstrates plausible substitution:

effects of POLLUTION OF RIVERS
$\vdash$ ⟨`modus continens`⟩
effects of POLLUTION
$\mathrel{|\!\sim}$ ⟨`plausible substitution`⟩
effects of POLLUTION FROM METALS
$\vdash$ ⟨`modus continens`⟩
effects of METALS

In the above only a single plausible substitution was involved in the (plausible) deduction of effects of metals from effects of pollution of rivers. This means that these expressions have a fairly high degree of similarity. That is, if effects of metals is a request and an information object O is characterized by effects of pollution of rivers, then it is fairly likely that O would be relevant.

The following plausible rule can be derived form the other rules:

Theorem 10.3.1 Let A be a set of axioms, I and J index expressions, and c a connector, then

$$A \overset{n}{\mid\!\sim} I \wedge A \overset{m}{\mid\!\sim} J \Rightarrow A \overset{n+m+1}{\mid\!\sim} I\ c\ (J)$$

Proof:

Suppose $A \overset{n}{\mid\!\sim} I \wedge A \overset{m}{\mid\!\sim} J$. Let s and t be the lead terms of I and J respectively. Then using *modus continens* yields the relevancy of s. The proof of the expresion $I\ c\ (J)$ starts with

$$\begin{array}{ll} & s \\ \mid\!\sim & \qquad \langle\texttt{refinement}\rangle \\ & s\ c\ (t) \end{array}$$

Then the proof proceeds with the proof of expresion I, followed by the proof of expression J. Note that the number of plausible steps in the proof of $I\ c\ (J)$ thus amounts to $n+m+1$.

10.4 Similarity

The principle of minimal axiomatic extension states:

The probability of an object being relevant to a request is inversely proportional to the minimal extension of the object description allowing to prove the request.

It is important to note that either the description must be extended with new axioms, or some axiom(s) of the description have to be *strengthened*. The opposite will not do simply because if the axiom set of a document is not strong enough to prove the request, then any subset, or *weakening*, of the axioms is also insufficient. An inverse approach to description strengthening is query weakening ([?]). We will restrict ourselves to the process of strengthening the axioms of the object description. In another approach a knowledge base is being used ([?]). It has even been suggested that the user supplies the extra semantics interactively ([?]).

By strengthening the axioms the deduction process becomes less certain because it involves suppositions that were not originally a part of the semantics of the object. This process of plausible deduction is driven by rules of plausible inference. Each such rule has associated a degree of plausibility. From this we derive the plausibility of plausible derivations.

Plausible deduction can be understood as a process in which we try to transform or *evolve* a set of axioms into the request. The more evolution necessary, the more dissimilar the request from the originating axioms. The amount of evolution can be formalized under the notion of the *evolutionary distance* between descriptions denoted $\delta(I, J)$. This distance may also be seen as the number of assumptions (plausible inference steps) that have to be taken in order to derive the second from the first.

As J may evolve from I in a number of ways, meaning that there may be several plausible deductions of J from I, the evolutionary distance between I and J is defined as:

$$\delta(I, J) = \min \left\{ \text{plausibility } I \mid\!\sim J \right\}$$

A simple approach is to define the plausibility of a plausible derivation as the number of plausible inference steps in this derivation, i.e., the number of assumptions that were made. The similarity $Sim(I, J)$ between descriptions I and J is then specified as a function that is inversely proportional to $\delta(I, J)$. For example, the similarity between two index expressions may be defined as:

Definition 10.4.1

$$Sim(I, J) = 2^{-delta(I,J)}$$

Applying this measure, we btain the following similarity: $Sim(\mathsf{effects of POLLUTION OF RIVERS}, \mathsf{effects of METALS}) = \frac{1}{2}$.

10.5 Spatial Coherence

The previous sections provide a logic-based mechanism to etimate the relevancy of each document. In a hypertext environment, however, there are many content-based relations between documents. In the stratified architecture (see section 5) there are structural and associative links betwen documents. Structural links describe how documents are composed out of smaller information objects, while associative links provide *see also* relations between presentations of information objects. Associative links are a special form to introduce an *accessibility relation* between documents. ¡p¿ The idea behind *spatial coherence* is that if a higly relevant document refers (via the accessibility relation) to another document, then this document will also be relevant to some extent. ¡p¿ There are a number of possibilities in this regard:

accessibility relation

spatial coherence

1. The first is that an accessibility relation R is given. This is the case in Hypertext object bases where the information objects are related with cross references (associative links).
2. Another possibility is that the relation R is derived.
   - In one approach object clustering is used [102].
   - Another approach is that the coherence be derived from the plausible inference mechanism.

     In this regard, there are two forms of coherence; strong and weak. In weak coherence, relatedness between objects p and q is established via a single descriptor of q:

     $$p\ R\ q \quad \Longleftrightarrow \quad \exists_{x \in \chi(q)}\,[\chi(p) \mid\!\sim x]$$

     Strong coherence, on the other hand, demands that all descriptors of q must be plausibly deducible from p:

     $$p\ R\ q \quad \Longleftrightarrow \quad \forall_{x \in \chi(q)}\,[\chi(p) \mid\!\sim x]$$

     Strong coherence can sometimes be too strong, therefore adversely effecting the first order relevance balancing process mentioned earlier.

     The weight of the coherence relation between p and q is defined as the best similarity measure between the two objects:

     $$w(pRq) \quad = \quad \max_{x \in \chi(q)} \left(\mathit{Sim}(\chi(p), x) \right)$$

After the relevance calculations from the previous section, we next take context considerations into account to smoothly balance relevance over the object network, using the spatial coherence of objects.

Spatial coherence can be understood from a modal logic approach: Each information object is considered a world; the worlds being connected to each other by an accessibility relation. If two worlds are connected the intented meaning is that the two objects have a level of coherence indicated by the strength function for this coherence.

In terms of logical inference, if the proof of a request q is not possible from $\chi(A)$, then an impression of relevance is attempted by considering similar objects. To this end, we introduce the coherence matrix D of objects. For convenience, we number the objects as $\{x_1, \ldots, x_n\}$. The coherence matrix is then defined by:

$$D_{ij} \quad = \quad \begin{cases} w(\langle x_i, x_j \rangle) & \text{if } \langle x_i, x_j \rangle \in R \\ 0 & \text{otherwise} \end{cases}$$

using the proximity weighting function w. Note that D is the null-matrix if there are no coherence relations between objects. Note that D^i describes the weights of paths of length i between desciptors.

Let r be the columnvector with initial values for relevance:

$$r_i = \mathit{Sim}(\chi(x_i), q)$$

The coherence matrix is used as a linear differential schema for relevance interpolation:

$$y \quad = \quad r + Dy$$

which is a compact notation for the equation: $y = \sum_{i \geq 0} D^i r$. The solution of this equation gives the spatial relevance of the objects with respect to request q. The equation may be rewritten as $(I - D)y = r$. This equation has a solution provided the matrix $I - D$ is regular. In that case we get:

$$y = (I - D)^{\leftarrow} r$$

We will not further elaborate on this approach in this paper. Usually the number of documents is very large, which makes the computation of the inverse of $I - D$ computationally intrctable. For efficiency reasons the restriction to first-order relevance balancing can be used:

$$y \quad = \quad r + Dr$$

In this approximation only the nearest neighbouring objects are used. By using higher order approximations, we move further and further from the original world therefore diminishing the probability of relevance.

The probability of relevance now is expressed as:

$$Prob\big(rel \,\big|x_i, q\big) = y_i$$

Example 10.5.1

Suppose the collection $\mathcal{O}$ contains 3 documents, with the following coherence matrix:

$$D = \begin{bmatrix} 0 & \frac{1}{2} & \frac{1}{8} \\ 0 & 0 & \frac{1}{4} \\ \frac{1}{4} & 0 & 0 \end{bmatrix}$$

The inverse of $I - D$ is:

$$(I - D)^{\leftarrow} = \frac{1}{14} \begin{bmatrix} 16 & 8 & 4 \\ 2 & 15 & 4 \\ 8 & 4 & 16 \end{bmatrix}$$

Suppose after the result of the relevancy computation has resulted in:

$$r^T = \begin{bmatrix} 0.75 & 0 & 0 \end{bmatrix}$$

Then after spatial coherence balancing we get the adjusted relevancy figures:

$$y^T = [(I - D)^{\leftarrow} r]^T = \begin{bmatrix} 0.857 & 0.107 & 0.429 \end{bmatrix}$$

By iterating we get an impression of the convergence speed in this case:

$$\begin{aligned} r^T &= \begin{bmatrix} 0.75 & 0 & 0 \end{bmatrix} \\ (r + Dr)^T &= \begin{bmatrix} 0.75 & 0 & 0.375 \end{bmatrix} \\ (r + Dr + D^2 r)^T &= \begin{bmatrix} 0.797 & 0.094 & 0.375 \end{bmatrix} \\ (r + Dr + D^2 r + D^3 r)^T &= \begin{bmatrix} 0.844 & 0.094 & 0.398 \end{bmatrix} \end{aligned}$$

Chapter 11

A Feedback Mechanism for Query by Navigation

11.1 Introduction

Relevance feedback gives a searcher the opportunity to compare a retrieval system's perception of the Information Need with the need at hand. With the information gleaned from this feedback, it is hoped that the system can improve its performance. This chapter presents a mechanism for handling relevance feedback in the context of Query by Navigation. With this type of query construction, the searcher is allowed to travel through the hyperindex in order to specify the Information Need. The descriptors associated with a rejected document are penalized, while those associated with an accepted document are rewarded.

In our approach, we intend to let a searcher describe the Information Need by travelling through the Hyperindex. This is done by moving from one descriptor to another along the arcs of the relations which structure the set of descriptors. The resulting sequence of desciptors called a *search path*.

search path

In this chapter, a theory is presented that llows the Retrieval System to anlyze the behaviour of the searcher during the query formulation process. The system will try to get an impresion of:

1. the goal the searcher is heading for
2. how certain (familiar) the searcher is in that area of the hyperindex.

In order to model a searcher's behaviour, we intend to utilize the concept of Markov theory. This theory can be used to study the effects of random walks through a state space. Recently, Mittendorf and Schaüble [72] applied so-called Hidden Markov Models to Information Retrieval.

11.2 Search path modelling

Suppose a person travelling Europe by train departs from Amsterdam. One could ask what the probability is that the searcher's final destination is, say, Rome. We shall call this probability the *target probability*. Consider the situation where the searcher has travelled to Munich. Moving toward Vienna next can be seen as having the effect of increasing Rome's target probability. Moving in another direction, e.g. Brussels, is translated to a decrease in Rome's target probability.

target probability

Basically, there are two types of usage of an information retrieval system: searching and browsing[69]. To illustrate the process of searching with this example, consider a traveller who wants to travel to, say, Athens. At each station, the traveller has to determine whether or not he or she is in Athens. If so, the final destination has been reached. If not, another train has to be boarded which should take the traveller closer to the final destination. The effect of browsing can be translated as a searcher who at each station visited wishes to know what, for instance, the local tourist attractions are.

Each new leg of the tour is chosen with no particular strategy in mind. With Query by Navigation the searcher constructs a query by travelling through the Hyperindex.

In this Hyperindex is seen as a graph $G = \langle \mathcal{D}, \textsf{related-to} \rangle$ where $\mathcal{D}$ is the set of descriptors and $\textsf{related-to} \subseteq \mathcal{D} \times \mathcal{D}$ the set of all connections between desciptors in the hyperindex. The subset REFINES of this relation between descriptors reflects the decomposition of concepts into narrower concepts. If $\langle a, b \rangle$ is an element of REFINES, then we say that a is a narrower concept than b, and b is a broader concept than a. Moving from descriptor b to a is called *refining*; moving the other way is called *broadening*.

A search path of length k is then a sequence of k descriptors $d_1 \ldots d_k$, such that $d_i \textsf{related-to} d_{i+1}$ or $d_{i+1} \textsf{related-to} d_i$ (for $1 \leq i < k$).

As an example, consider the Hyperindex graph of Figure 11.2. The refinements which hold in this example are:

1. theft of car REFINES car
2. theft of car REFINES theft
3. theft of bicycle REFINES bicycle
4. theft of bicycle REFINES theft

Figure 11.1: Refinement graph

Two example search paths are:

- theft ∘ theft of car ∘ car
- theft ∘ theft of bicycle ∘ theft ∘ theft of car

where the symbol ∘ denotes the concatenation of descriptors.

11.2.1 Interpretation of a search path

With every decision taken by the searcher more information is gained about the searcher's intentions and background. Each transition from d to e involves a conscious choice by the searcher by preferring e over the other options in d. The example suggests that the probability of a descriptor d being the target of the search process is related to the distance fluctuations while travelling the search path. Recent fluctuations have more impact than past fluctuations. In terms of the previous example, at some point in time a possible interest in Paris may be concluded from the move to Brussels. Since the traveller opted to go to Rome from Munich, the probability of the traveller ever stopping the tour in Brussels decreases as the path progresses.

Typically, the farther removed a descriptor is from the path, the less likely that it is the target of the search. This is captured by the probability $Prob\left(d \left| p\right.\right)$ of d being the search target, after having traversed search path p.

For this purpose we will transform the search space into a transition network, allowing the use of Markov chains theory. First, the set of states is defined as the set of descriptors augmented with two special states.

The first of these special states is called the start state start. The purpose of this state is that it represents the initial state of the search process, where no information about the searcher's intentions is known. The second special state is called stop. This state represents the termination of the search process.

The transitions between states are defined as follows: if descriptors a and b are connected via the relation related-to, then they are related in the transition network. Regarding the special state, we introduce the following conventions:

- from each descriptor there is a transition to stop.
- from start there is a transition to each descriptor

Figure 11.2.1 shows the construction of the transition network from the 'theft' subgraph of the original REFINES network of Figure 11.2.

We assume for each transition e a probability $q_e > 0$ of occurring. In a transition network, for each state a:

$$\sum_{b:a\rightarrow b} q_{a\rightarrow b} = 1$$

The transition matrix T is then defined for descriptors as (see [51]):

Definition 11.2.1

$$T(x,y) = \begin{cases} q_e & if x and y are connected via edge e \\ 0 & otherwise \end{cases}$$

The transition from the start state to a descriptor d occurs under a probability s_d. Going from a descriptor d to the stop state is done with probability h_d.

For all possible transitions (including the ones from start and to stop) we have the matrix $\overline{T}$.

The start state is a state which can only be left; there is no return to this state possible, so for each descriptor d, $\overline{T}(x, \mathsf{start}) = 0$.

Since the state stop is the final state, $\overline{T}(\mathsf{stop}, d) = 0$ for each descriptor d. $\overline{T}(\mathsf{stop}, \mathsf{stop}) = 1$ because this state cannot be left. In each descriptor d, stopping is possible: $h_d > 0$.

$T^i(d,e)$ is the probability of reaching e starting from d in i transitions. $T^0 = I$, the identity matrix. Thus, the sum

$$\sum_{i=0}^{\infty} T^i(e,d)$$

is the probability of reaching d from e in any number of steps. The continuation from e to d is introduced as:

Figure 11.2: Derivation of the transition net

Definition 11.2.2

$$C_e(d) \quad = \quad \sum_{i=0}^{\infty} T^i(e,d)h_d$$

The infinite sum

$$\sum_{i=0}^{\infty} T^i(e,d)$$

converges to

$$(I-T)^{-1}(e,d) = M(e,d)$$

(see [100]). This requires the calculation of the inverse matrix of an $|\mathcal{D}| \times |\mathcal{D}|$ matrix.

Lemma 11.2.1 $C_e(d) = M(e,d)h_d$

An important property of the matrix M is that the diagonal element of a column is the *maximum* of that column. In other words, the probability on a path of any length between two descriptors d and e is maximal when such a path both starts and ends in the same descriptor, or: $e = d$.

Lemma 11.2.2 $\forall_{e \in \mathcal{D}} \, [M(e,d) \leq M(d,d)]$

Proof:

[1] Let M be the inverse of $(I-T)$. M can be written as

$$M = I + T + T^2 + \ldots$$

[1]With kind regards to Dominique Bernardi of the faculty of Number Theory, Pierre & Marie Curie University, France, for pointing us in the right direction

Therefore, we have

$$M = I + TM$$

Let m_{ij} be the maximum of column j, and suppose that $j \neq i$. The equation (**??**) gives:

$$m_{ij} = \sum_k t_{ik} m_{kj}$$

m_{kj} is less than m_{ij}, so the right hand side of this equation is less than $m_{ij} \sum_k t_{ik}$. This latter sum is < 1. Hence $m_{ij} = 0$ or there is a contradiction. If $m_{ij} = 0$ then m_{ii} is also a maximum.

Lemma 11.2.3

$$\sum_{d \in \mathcal{D}} M(x,d)T(d,y) = \begin{cases} M(x,y) & \text{if } x \neq y \\ M(x,x) - 1 & \text{if } x = y \end{cases}$$

Proof:
Trivial

Lemma 11.2.2 has shown that the probability of a path of any length between descriptors d and e is maximal when $e = d$. This lemma can be used to prove that the continuation probability $C_e(d)$ is maximal when $e = d$. So, when the searcher is in a descriptor e he or she is most likely to stop searching in that descriptor, if (as explained earlier) previous behaviour is ignored.

Corollary 11.2.1 $\forall_{e \in \mathcal{D}} \left[C_e(d) \leq C_d(d)\right]$
The equality occurs only when $e = d$.

From any descriptor in the Hyperindex it is to be expected that we always continue to another descriptor. This is formalized in the following lemma:

Lemma 11.2.4 $\forall_{d \in \mathcal{D}} \left[\sum_{x \in \mathcal{D}} C_d(x) = 1\right]$

Proof:
Let $\vec{x} = (x_1, \ldots, x_n)^T = M\vec{h}$, where $\vec{h} = (h_{d_1}, \ldots, h_{d_n})^T$.
Left-multiplying this by M leads to

$$\vec{x} = \vec{h} + T\vec{x}$$

This equation is obviously satisfied by $\vec{x} = (1, \ldots, 1)^T$, due to property (**??**). This is the only solution, since if both $\vec{x}_1 = \vec{h} + T\vec{x}_1$ and $\vec{x}_2 = \vec{h} + T\vec{x}_2$, then $\vec{x}_1 - \vec{x}_2 = T(\vec{x}_1 - \vec{x}_2)$, which implies $\vec{x}_1 = \vec{x}_2$, as $|T| < 1$.

An important characteristic of the transition network is that for each row the sum of its elements equals 1:

$$\sum_{d \in \mathcal{D}} T(e,d) + h_d = 1$$

Next we focus at the probability $Prob\left(d \,|p\right)$ of a descriptor d being the search target of a path p. First we will investigate this probability for the case of the empty path, i.e. $Prob\left(d \,|\epsilon\right)$ for a descriptors d.

The a priori probability of d being the target of a search path can be expressed as:

Definition 11.2.3
$Prob\left(d \,|\epsilon\right) = \sum_{x \in \mathcal{D}} s_x C_x(d)$

This is referred to as the initial or *a priori* destination probability. Note that in general the initial probability is not a flat distribution. In this definition, the vector $s = (s_1, \ldots, s_n)^T$ contains the probability of starting in a certain descriptor. Since we always start in a descriptor, we have

Corollary 11.2.2 $\sum_{x \in \mathcal{D}} s_x = 1$

As some descriptor must eventually be the target of the search process, we have:

Lemma 11.2.5 $\sum_{d \in \mathcal{D}} Prob(d \mid \epsilon) = 1$

Proof:

$$\begin{aligned}
\sum_{d \in \mathcal{D}} Prob(d \mid \epsilon) &= \sum_{d \in \mathcal{D}} \sum_{x \in \mathcal{D}} s_x C_x(d) \\
&= \sum_{x \in \mathcal{D}} s_x \sum_{d \in \mathcal{D}} C_x(d) \\
&= \quad \langle \texttt{lemma 11.2.4} \rangle \\
& \quad \sum_{x \in \mathcal{D}} s_x \\
&= \quad \langle \texttt{cor. 11.2.2} \rangle \\
& \quad 1
\end{aligned}$$

The *a priori* target probability for a descriptor falls in a certain interval. An upper bound for this interval is:

Lemma 11.2.6 $Prob(d \mid \epsilon) < C_d(d)$

Proof:

$$\begin{aligned}
Prob(d \mid \epsilon) &= \sum_{x \in \mathcal{D}} s_x C_x(d) \\
&< \quad \langle \texttt{lem.11.2.1} \rangle \\
& \quad \sum_{x \in \mathcal{D}} s_x C_d(d) \\
&= C_d(d) \sum_{x \in \mathcal{D}} s_x \\
&= \quad \langle \texttt{lem. 11.2.2} \rangle \\
& \quad C_d(d)
\end{aligned}$$

The a priori destination probability defines the target probability when the searcher is still in the start state. The searcher will not remain in this state. If the searcher moves away from this state by choosing a descriptor, the target probability for the descriptors will fluctuate.

In general, after travelling a path p through the descriptor space, the searcher will extend the search path with a descriptor e from the options available at the end of p. After travelling path p the target probability will be $Prob(d \mid p)$ for a descriptor d. Extending the search path will change the distribution. It can be argued that the previous history (i.e. the path p) can be ignored and that only the effect of moving to e has to be looked at. However, while travelling p certain decisions have been made by preferring one descriptor over the other available options. Hence we propose to let $Prob(d \mid p)$ play a role in determining $Prob(d \mid pe)$. The second component in determining the new target probability is formed by the move towards descriptor e. Since the past already is taken into account by looking at $Prob(d \mid p)$ we aim to ignore the past in the case of this second component. This past includes the decision of preferring e over the other options. These other options are thus *implicitly rejected*. The second component will look at the probability of continuing from e to d and stopping in d.

The past therefore carries a certain weight when a new target probability has to be calculated. This influence is captured by assigning a weight 1 to the continuation probability and a weight λ to $Prob\big(d\,|p\big)$. $\lambda \geq 0$ is the forgetfulness coefficient, indicating the influence past decisions have on the target probability after a new step. If $\lambda = 0$, then the past is completely forgotten. Prior decisions can be made more important than later decisions by choosing $\lambda > 1$. The usual case is that the past carries less weight than the present, which is reflected by setting $\lambda < 1$. A reasonable choice is $\lambda = \frac{1}{2}$.

Next, we introduce the probability function $Prob\big(d\,|p\big)$ for a composed search path p. This is done recursively as follows:

Definition 11.2.4

$$Prob\big(d\,|pe\big) = \frac{\lambda Prob\big(d\,|p\big) + C_e(d)}{\lambda + 1} \tag{11.1}$$

The components which play a part in determining the target probability are shown in Figure 11.2.1. This figure shows within the oval the search space of descriptors, $\mathcal{D}$. The searcher has travelled a path $p = p_1 \ldots p_k$. Time progresses as descriptors are added to the search path. Each descriptor d has at any point on the path a probability that the search will end in d (i.e. a transition to d followed by a transition to `stop`).

Figure 11.3: The components

The property that each path has an ultimate target is captured by:

Lemma 11.2.7 $\sum_{d \in \mathcal{D}} Prob\big(d\,|p\big) = 1$

Proof:

By induction on the construction of the search path.

$$\begin{aligned}
&\sum_{d\in\mathcal{D}} Prob(d\,|pe) \\
&= \frac{1}{\lambda+1}\sum_{d\in\mathcal{D}}((\lambda Prob(d\,|p) + C_e(d)) \\
&= \frac{1}{\lambda+1}\left(\lambda\sum_{d\in\mathcal{D}} Prob(d\,|p) + \sum_{d\in\mathcal{D}} C_e(d)\right) \\
&= 1
\end{aligned}$$

After the searcher has traversed a certain path p each descriptor will have a target probability $Prob(d\,|p)$. It is possible to define an interval for this target probability. First we look at the upper bound of this interval:

Lemma 11.2.8 $Prob(d\,|p) < C_d(d)$

Proof:

Suppose $p = p_1 \dots p_n$, then:

$$\begin{aligned}
Prob(d\,|p) &= \left(\frac{\lambda}{\lambda+1}\right)^n Prob(d\,|\epsilon) + \sum_{i=0}^{n-1}\left(\frac{\lambda}{\lambda+1}\right)^i \frac{C_{p_{n-i}}(d)}{\lambda+1} \\
&< \quad \langle\texttt{lem. 11.2.6}\rangle \\
&\quad \left(\frac{\lambda}{\lambda+1}\right)^n C_d(d) + \sum_{i=0}^{n-1}\left(\frac{\lambda}{\lambda+1}\right)^i \frac{C_{p_{n-i}}(d)}{\lambda+1} \\
&< \quad \langle\texttt{cor. 11.2.1}\rangle \\
&\quad \left(\frac{\lambda}{\lambda+1}\right)^n C_d(d) + \sum_{i=0}^{n-1}\left(\frac{\lambda}{\lambda+1}\right)^i \frac{C_d(d)}{\lambda+1} \\
&= C_d(d)\left(\left(\frac{\lambda}{\lambda+1}\right)^n + \frac{1}{\lambda+1}\frac{1-\left(\frac{\lambda}{\lambda+1}\right)^n}{\frac{1}{\lambda+1}}\right) \\
&= C_d(d)
\end{aligned}$$

Returning to the train traveller analogy, consider someone who at time t decides to travel in the general direction of Paris. It is to be expected that the target probability of Paris increases after this decision. The concept of *in the general direction of* is somewhat vague and is difficult to capture. This is because we have not yet introduced a measure of *distance*. Therefore we start by looking at the situation where the searcher is in the start state start. No decision has been taken yet, and each descriptor has a target probability $Prob(d\,|\epsilon)$. If the searcher now decides to perform a transition to a particular descriptor d, the expectation is that the target probability for d has increased:

Theorem 11.2.1 $\forall_{d\in\mathcal{D}}\left[Prob(d\,|\epsilon d) > Prob(d\,|\epsilon)\right]$

Proof:

$$\begin{aligned}
Prob(d\,|\epsilon d) &= \frac{\lambda Prob(d\,|\epsilon) + C_d(d)}{\lambda+1} \\
&> \quad \langle\texttt{lem. 11.2.8}\rangle
\end{aligned}$$

$$\begin{aligned}
& \frac{\lambda Prob(d|\epsilon) + Prob(d|\epsilon)}{\lambda+1} \\
= \quad & \langle \texttt{def. 11.2.4} \rangle \\
& Prob(d|p)
\end{aligned}$$

Note that the forgetfulness coefficient does not appear anymore. The theorem is hence valid for any value of the forgetfulness coefficient.

This theorem can be generalized to the situation where an arbitrary path p is extended with a desciptor d. Again, the expectation is that the target probability for d has increased after this move has been made.

Theorem 11.2.2 $Prob(d|pd) > Prob(d|p)$

Proof:

$$\begin{aligned}
Prob(d|pd) \quad = \quad & \frac{\lambda Prob(d|p) + C_d(d)}{\lambda+1} \\
> \quad & \langle \texttt{def. 11.2.4} \rangle \\
& \frac{\lambda Prob(d|p) + Prob(d|p)}{\lambda+1} \\
= \quad & \langle \texttt{def. 11.2.4} \rangle \\
& Prob(d|p)
\end{aligned}$$

Now that the effect of extending a path with a descriptor d on the target probability for d has been studied, we next look at the effect of moving away from a descriptor.

Once again we refer to the train traveller analogy. If the traveller moves away from Paris, it is to be expected that the target probability for Paris decreases. As with the previous discussion, we first examine the effect of moving one step away from a descriptor.

This penalty for moving away from a descriptor is captured in the following theorem:

Theorem 11.2.3 $Prob(d|pde) < Prob(d|pd)$

Proof:

Suppose $p = p_1p_2 \dots p_n$, then:

$$\begin{aligned}
Prob(d|pde) \quad = \quad & \left(\frac{\lambda}{\lambda+1}\right)^{n+2} Prob(d|\epsilon) + \\
& + \sum_{i=0}^{n-1} \left(\frac{\lambda}{\lambda+1}\right)^{i+2} \frac{C_{p_{n-i}}(d)}{\lambda+1} + \\
& \frac{\lambda C_d(d)}{\lambda+1} + \frac{C_e(d)}{\lambda+1} \\
< \quad & \langle \tfrac{\lambda}{\lambda+1} < 1) \rangle \\
& \left(\frac{\lambda}{\lambda+1}\right)^{n+1} Prob(d|\epsilon) + \\
& + \sum_{i=0}^{n-1} \left(\frac{\lambda}{\lambda+1}\right)^{i+1} \frac{C_{p_{n-i}}(d)}{\lambda+1} \\
& + \frac{\lambda C_d(d)}{\lambda+1} + \frac{C_e(d)}{\lambda+1}
\end{aligned}$$

$$
\begin{aligned}
< \quad & \langle \texttt{cor. } 11.2.1 \rangle \\
& \left(\frac{\lambda}{\lambda+1}\right)^{n+1} Prob\big(d \,\big|\, \epsilon\big) \\
& + \sum_{i=0}^{n-1} \left(\frac{\lambda}{\lambda+1}\right)^{i+1} \frac{C_{p_{n-i}}(d)}{\lambda+1} + \\
& + \frac{\lambda C_d(d)}{\lambda+1} + \frac{C_d(d)}{\lambda+1} \\
= \quad & \left(\frac{\lambda}{\lambda+1}\right)^{n+1} Prob\big(d \,\big|\, \epsilon\big) \\
& + \sum_{i=0}^{n-1} \left(\frac{\lambda}{\lambda+1}\right)^{i+1} \frac{C_{p_{n-i}}(d)}{\lambda+1} + \frac{C_d(d)}{\lambda+1} \\
= \quad & \langle \texttt{def. } 11.2.4 \rangle \\
& Prob\big(d \,\big|\, pd\big)
\end{aligned}
$$

Chapter 12

The incremental model

In this chapter, the *incremental searcher satisfaction model* for Information Retrieval is introduced. In this model, documents are not presented according to decreasing relevancy only, but also on the level of novelty in the context of the documents previously presented. Documents which are judged to be insufficiently surprising (according to a searcher determined threshold) are not presented to the searcher.

This is especially useful for Information Retrieval in certain contexts (e.g. Internet applications such as search engines), when a searcher does not want all relevant documents to be shown, but only have a global idea of the variety of what the corpus may contain on a topic. Important properties of this model are discussed, such as the relation between the reductional effect and the order of presentation.

12.1 Introduction

A typical problem which may occur when using state of the art search engines is that a retrieval result does not nicely reflect an overview of *all* aspects which have deemed to be relevant to the query submitted by the searcher. For example, a document may occur several times in the retrieval result (if it is available at different sites). In addition, documents which are very similar appear at the same relevancy level.

Different categories of searchers will react differently to this. Searchers in need of a complete overview of all matching documents might be satisfied with the above presentation of search results. However, this presentation is not convenient for a searcher requiring a broad impression of matching documents. In that case, a document should only be presented to the searcher if it offers sufficient new information.

We propose the *incremental searcher satisfaction model* as a mechanism to handle such problems. This model has been introduced in [106]. We will use the terms incremental satisfaction, novelty and surprise as synonyms in this text. The main idea is to try to estimate how surprising the next document is for the searcher. This mechanism can be used in combination with other techniques, such as ranking (see e.g. [98]) and link generation (e.g. [8] and [108]).

A different approach to the problem mentioned above is to classify the documents of the retrieval result, and then to present a typical document of each class. The difference however is that this will be more time consuming, while our technique can be applied in real time (in the sense of algorithmic complexity theory).

maximal marginal relevance

In [25] the *maximal marginal relevance* (MMR) function is introduced. This function is constructed as a linear combination of two similarity functions, one to be used to quickly select a set of documents, which is more closely investigated by the second, more accurate similarity function. After presenting a set of documents, the MMR function is used to perform diversity-based reranking by determining the differential relevance with respect to those documents. The MMR function is applied to automatically construct a query-relevant summarization of documents, although it might also be applicable for incremental searcher satisfaction models. In this chapter, a more fundamental approach to differential relevance is taken. The elegance of this approach lies in the fact that elementary properties can be formulated and proven nicely. The use of formal models to handle this ordering concept leads us to a better understanding of the nature of ordering documents in Information Retrieval.

Note that a document summary is obtained by omitting certain parts of the documents, which are judged to be insufficiently relevant for the required level of summarization. Typical for this application is that the original order of the information objects in the document is not disturbed. In this chapter we also consider the situation when there is no a-priori order for information objects.

Some extra references: [4], [6], [5], [42].

The structure of this chapter is as follows. In section 2, the incremental searcher satisfaction model is introduced as the (increment) function with a number of axioms which this function has to satisfy. Next, the relative need function is introduced as a special instance of the increment function. In section 3 we focus on the effects of presenting a series of documents. Each presentation sequence has an accumulated level of additional surprises. This could be used as a quality measure of the retrieval result. In section 4 some properties of the incremental model are presented, with a special emphasis on content relations between documents.

12.2 The incremental searcher model

The Information Retrieval paradigm is about a person (physical or not) having a need for information, and an information collection from which this need is to be satisfied. Elements of the information collection are referred to as documents or information objects.

The need for information is satisfied by documents. The need for information thus induces a need for information objects. However, after having been presented some (relevant) documents, the yet unsatisfied part of the information need will induce a different need for information objects.

For example, Suppose a searcher is interested in information about A, B and C, which has to be satisfied from the following collection:

- x_1 *All about A and something about C*
- x_2 *About B and C*
- x_3 *About B and A*
- x_4 *All about A*

The first document x_1 contains the requested information about A, and some information about C. After inspecting this document, the need for information thus is restricted to information B and C in a limited extend. This is summarized in table12.1.

document	*title*	*initial need*	*need after x_1*
x_1	All about A and something about C	++	presented
x_2	About B and C	++	++
x_3	About B and A	++	+
x_4	All about A	++	-

Figure 12.1: The residual need for documents

In this section, we provide a general model to formalize the information need in terms of a need for information objects, and introduce the incremental searcher model as a framework for so-called increment functions. As a specific instance the relative need function is discussed. Finally, the retrieval approach within this framework is shown.

As stated in section 3.1, it is assumed in the incremental (searcher satisfaction) model that the need for more documents is influenced by what the searcher already has retrieved from the archive. This can be modelled as a function

$$I : \wp(\mathcal{O}) \times \mathcal{O} \mapsto [0, 1]$$

$I(S, x)$ is interpreted as the increment in searcher satisfaction when document x is presented after set S has already been presented to the searcher. The function I is also referred to as the *increment function*. Note that the weighted model can be obtained from the incremental model by defining $N(x) = I(?, x)$. Thus, the need for a document is taken as the satisfaction obtained without any prior knowledge. As a consequence, S can also be interpreted as previous personal knowledge of the searcher (sometimes also called a user profile).

The incremental model is especially useful for (very) dynamic and distributed archives, such as the World Wide Web. Firstly, as the increment function allows for real-time calculation, This is in contrast with approaches that try to cluster the retrieval result before presenting the clusters to the searcher. Clustering is only possible after all documents have been obtained. Secondly, for distributed archives recall is not useful as a measure for retrieval quality. We will propose a quality measure in section 12.7.3 which is based on total searcher satisfaction, bypassing the need to have global knowledge of the collections involved.

The increment function has to satisfy a number of conditions. The first condition states that presenting a document twice does not add anything. The second condition expresses that the incremental value of a document can not grow after supplying more documents:

$$\begin{array}{llrcl} \text{IM1} & \textit{law of repetition} & x \in S & \Rightarrow & I(S, x) = 0 \\ \text{IM2} & \textit{law of growing knowledge} & S \subseteq T & \Rightarrow & I(S, x) \geq I(T, x) \end{array}$$

A third axiom will be formulated in a later section. Note that an alternative approach would be to have IM1':

$$\text{IM1'} \quad \textit{law of repetition} \quad I(\{x\}, x) = 0.$$

IM1' immediately follows from axiom IM1. On the other hand, IM1 can be derived from IM1' combined with IM2:

Proof:

Suppose $x \in S$, then $\{x\} \subseteq S$, and thus from IM2 it follows: $I(S, x) \leq I(\{x\}, x)$. From $I(\{x\}, x) = 0$ we conclude $I(S, x) = 0$.

The following property is an immediate consequence of axiom IM2:

Lemma 12.2.1 $I(S, x) \leq N(x)$

Proof:

As $? \subseteq S$, it follows from IM2 that $I(?, x) \geq I(S, x)$, and thus $N(x) \geq I(S, x)$.

In other words, the maximal satisfaction which can be obtained from a document x is its information need $N(x)$. If x would be presented after the document set S has already been presented, then the incremental searcher satisfaction $I(S, x)$ is at most this maximal satisfaction. As a consequence of this interpretation, the increment function is also referred to as the *residual information need*, i.e., the restant of the information need after being confronted with S.

Next we isolate the effect of presenting a single document.

Lemma 12.2.2 $I(S \cup \{y\}, x) \leq I(S, x) + I(\{y\}, x)$

Proof:

From IM2 we conclude $I(S \cup \{y\}, x) \leq I(\{y\}, x)$, and thus also $I(S \cup \{y\}, x) \leq I(S, y) + I(\{y\}, x)$.

If presenting the documents from set S does not affect the amount of new information provided by document x after (also) presenting document y, then all information contained in document y is available in the documents from S.

Lemma 12.2.3 $I(S \cup \{y\}, x) = I(S, x) + I(\{y\}, x) \Rightarrow I(S, x) = 0$

Proof:

Suppose $I(S \cup \{y\}, x) = I(S, x) + I(\{y\}, x)$. From IM2 it follows that $I(S \cup \{y\}, x) \leq I(\{y\}, x)$, and thus $I(S, x) + I(\{y\}, x) = I(S \cup \{y\}, x) \leq I(\{y\}, x)$, leading to $I(S, x) \leq 0$.

Corollary 12.2.1

$I(\{x, y\}, z) \leq I(\{x\}, z) + I(\{y\}, z)$
$I(\{x, y\}, z) = I(\{x\}, z) + I(\{y\}, z) \Rightarrow I(\{x\}, z) = 0$

After having introduced the requirements for the increment function I, we will look at some concrete definitions in the next section.

12.3 Fundamentals of increment functions

In this section we introduce some concrete definitions for increment functions. For this purpose, we also consider similarity functions. We show how the increment function may easily be added to an existing IR situation. In such a case, some information need function N and some measure *Sim* for similarity already have been defined. Furthermore, we assume documents are characterized in terms of a set $\mathcal{D}$ of descriptors by the function $\chi : \mathcal{O} \mapsto \wp(\mathcal{D})$.

Using these functions, we first introduce a special class of increment functions, based on the similarity of a document to a set of documents.

12.3.1 The individual approach

In this approach, the similarity between a document and a set of documents in measured as the maximal similarity between the document and any instance of this set:

$$\mathit{Sim}_i(S, x) = \max\left\{\mathit{Sim}(\chi(x), \chi(y)) \;\middle|\; y \in S\right\}$$

The expression $\mathit{Sim}_i(S, x)$ provides the maximal similarity between document x and any of the elements from S of previously presented documents. Thus, $1 - \mathit{Sim}_i(S, x)$ can be seen as the amount of new information provided by document x compared to set S. Finally, the outcome is scaled into the interval $[0, N(x)]$ (see lemma 12.2.1). Based on this similarity relation, the individual increnent function is defined as:

$$I_i(S, x) = N(x)\,(1 - \mathit{Sim}_i(S, x))$$

Thus $I(S, x)$ gives the fraction of the need $N(x)$ for document x not yet being covered by any previously presented document from S. Consequently, for two documents bringing an equal quantity of new information, the more relevant one is displayed before the less relevant one, as one would expect. Otherwise, the most exotic (and therefore probably highly surprising) documents would be presented before relevant ones.

Assuming the similarity function *Sim* has the property $\mathit{Sim}(A, A) = 1$ for each A, the axioms IM1 and IM2 are satisfied:

Proof:

IM1: Let $x \in S$, then $\mathit{Sim}_i(S, x) = 1$, and thus $I_i(S, x) = 0$.
IM2: Let $S \subseteq T$, then $\mathit{Sim}_i(S, x) \leq \mathit{Sim}_i(T, x)$, and thus $I_i(S, x) \geq I_i(T, x)$.

As a example, similarity between documents can be computed using Jaccard coefficient, comparing sets A and B of document chacterizations as: $\mathit{Sim}(A, B) = |A \cap B| \,/\, |A \cup B|$. Then obviously $\mathit{Sim}(A, A) = 1$ for each A.

In [25] the maximal marginal relevance function is introduced as a mechanism to estimate differential relevance of documents. In terms of our formalism, the essence of the maximal marginal relevance function is the following function:

$$\mathsf{MMR}_q(S, x) = \alpha\overline{\mathit{Sim}}(\chi(x), q) + (1 - \alpha)\,\mathit{Sim}_i(S, x)$$

where $\overline{Sim}$ and Sim are similarity functions. In this definition, the expression $\overline{Sim}(\chi(x), q)$ quantifies the *objective* meaning of query q. The increment function $I(S, x)$ is concerned with a *subjective* estimation of relevance. For an objective searcher, this can be identifed with the need function N. This leads to

$$\mathsf{MMR}(S, x) = \alpha N(x) + (1 - \alpha)\, Sim_i(S, x)$$

Note that the function MMR does not satisfy the axiom IM1 for increment functions. The motivation behind the maximal marginal relevance function has some ressemblance to the increment function, but they are rather different in their properties.

12.3.2 The collective approach

In the collective approach, a new document is compared to a set S of previously presented documents by comparing the characterization of x with a summary of all presented material from S. The summary $\sigma(S)$ of the set is defined as follows:

$$\sigma(S) = \cup_{x \in S}\, \chi(x)$$

Thus, $\sigma(?) = ?$ and $\sigma(S \cup \{x\}) = \sigma(S) \cup \chi(x)$. The similarity between a document x and a set S of documents then is defined as

$$Sim_c(S, x) = Sim(\chi(x), \sigma(S))$$

The expression $Sim_i(S, x)$ provides the degree document x is covered by the total of information provided by the elements from S of previously presented documents. Analogously to the individual approach, the collective increment function is defined as:

$$I_c(S, x) = N(x)\,(1 - Sim_c(S, x))$$

Note that $I(?, x) = N(x)$ as $Sim(?, \chi(x)) = 1$. We assume the similarity function Sim has the following properties.

1. $A \subseteq B \Rightarrow Sim(A, B) = 1$
2. $A \subseteq B \Rightarrow Sim(C, A) \leq Sim(C, B)$

In the next sections we see an example of such a similarity function. Under this assumption, the axioms for increment functions are satisfied by the collective increment function:

Proof:

IM1: Let $x \in S$, then $\chi(x) \subseteq \sigma(S)$, and thus $Sim_c(\chi(x), \sigma(S)) = 1$. As a consequence $I_c(S, x) = 0$.
IM2: Let $S \subseteq T$, then $\sigma(S) \subseteq \sigma(T)$, and thus then $Sim_c(\chi(x), \sigma(S)) \leq Sim_i(\chi(x), \sigma(T))$. As a consequence, $I_c(S, x) \geq I_c(T, x)$.

An alternative would be to introduce the increment function as $I(S, x) = N(x)\, Sim(\sigma(S), \chi(x))$. The essence of this choice is that $Sim(\sigma(S), \chi(x))$ corresponds to the degree in which the information already presented ($\sigma(S)$) is contained within the new document x. However, this function does not satisfy the axioms IM1 and IM2, and therefore is not a valid increment function.

12.4 The general case

The observations from the previous sections lead to the following definition for personal document similarity.

Definition 12.4.1

Let I be an increment function, then the personal similarity between a relevant document x (i.e. $N(x) > 0$) and a set S of documents is defined as

$$Sim(S, x) = 1 - \frac{I(S, x)}{N(x)}$$

Note that personal document similarity is related to the information need as expressed by the increment function I. Another information need leads to another view on similarity! In terms of this definiton, the similarity of a relevant document x with a document y is expressed as $Sim(\{y\}, x))$. From the axioms IM1 and IM2 the following properties are immediate:

Lemma 12.4.1
$x \in S \Rightarrow Sim(S, x) = 1$
$S \subseteq T \Rightarrow Sim(S, x) \leq Sim(T, x)$

12.5 The relative need function

In this section, the *relative need* function is proposed as a special instantiation of the increment function in the collective approach. The term *relative* indicates that the need is considered in relation to previously presented documents.

12.5.1 Similarity

The relative need function results from applying in the collective approach the inclusion measure for similarity ([81]), measuring the degree in which one set is covered by another. The motivation for choosing the inclusion measure is that we will be interested in the degree in which the (characterization of) a new document is covered by the (characterization of) the previously presented documents.

The inclusion measure is defined by:

$$Sim(A, B) = |A \cap B| \,/\, |A|$$

in case $A \neq ?$, while $Sim(?, B) = 1$. This measure thus qualifies up to what extent set A is a subset of B. If $A \subseteq B$, then $Sim(A, B) = 1$, while $Sim(A, B) = 0$ if and only if A and B are disjoint. Furthermore, note that $B_1 \subseteq B_2$ implies $Sim(A, B_1) \leq Sim(A, B_2)$.

The inclusion measure satisfies the properties (mentioned in section 12.3.2) for similarity functions in the collective approach:

1. Let $A \subseteq B$, then $A \cap B = A$, and thus $Sim(A, B) = 1$.
2. Let $A \subseteq B$, then $|C \cap A| \leq |C \cap B|$, and thus $Sim(C, A) \leq Sim(C, B)$.

12.5.2 Some examples

Next we consider some examples of the behavior of the relative need function.

Example 12.5.1
Let $A = \{a, b\}$, $B = \{c, d\}$ and $C = \{d, e, f\}$. Note that $Sim(A, ?) = 0$. Furthermore, $Sim(A, B) = Sim(B, A) = 0$ while $Sim(B, C) = \frac{|B \cap C|}{|B|} = \frac{1}{2}$. Finally, $Sim(C, A \cup B) = Sim(\{d, e, f\}, \{a, b, c, d\}) = \frac{1}{3}$.

The next example elaborates on this example.

Example 12.5.2
Let documents x_A, x_B and x_C be characterized by A, B and C respectively. This leads for the relative information need to:

$$\begin{aligned}
I(?, x_A) &= N(x_A) \cdot (1 - Sim(A, ?)) = N(x_A) \\
I(\{x_A\}, x_B) &= N(x_B) \cdot (1 - Sim(B, A)) = N(x_B) \\
I(\{x_A, x_B\}, x_C) &= N(x_C) \cdot (1 - Sim(C, A \cup B)) = N(x_C) \cdot (1 - \frac{1}{3}) = \frac{2}{3} N(x_C)
\end{aligned}$$

In the next example, we quantify the outcome of the relative infomation need in terms of the cardinalities of characterizations used.

Example 12.5.3

Let the sets A and B be such that $|A - B| = n$, $|B - A| = m$, while $|A \cap B| = k$. Then $Sim(A, B) = \frac{k}{n+k}$ and $Sim(B, A) = \frac{k}{m+k}$. Let d_1 and d_2 be documents characterized by A and B respectively. Then:

$$I(\{d_1\}, d_2) = N(d_2)\,(1 - Sim(B, A)) = \frac{m}{m+k} N(d_2)$$

So, the surprise of presenting d_2 after d_1 increases by more new terms in d_2 with respect to d_1 (i.e., increasing m), but decreases by more common terms in d_1 and d_2 (i.e., increasing k).

12.5.3 A probabilistic interpretation

The relative need can also be interpreted in terms of probabilities. The inclusion measure corresponds to a conditional probability: $Sim(A, B) = Prob(B \,|\, A)$. Then $I(S, x)$ corresponds to the probability of x being relevant after presenting the documents of S. If the information need of a document is seen as the probability of its characterization being relevant:

$$N(x) = Prob(\chi(x))$$

Then the relative need can be formulated as:

$$\begin{aligned} I(S, x) &= Prob(\chi(x))\,\big(1 - Prob(\sigma(S) \,|\, \chi(x))\big) \\ &= Prob(\chi(x)) Prob(\overline{\sigma(S)} \,|\, \chi(x)) \\ &= Prob\left(\overline{\sigma(S)} \cap \chi(x)\right) \\ &= Prob(\chi(x) - \sigma(S)) \end{aligned}$$

So the probability of x adding new information after presenting the documents from S, equals the probability of going its characterization beyond the knowledge $\sigma(S)$. This is a step towards the application of increment functions in the context of probabilistic retrieval ([46]). Note that the searcher query is not mentioned in these probabilities as is usual in the context of probabilistic retrieval (see [81]).

Having introduced the relative need as a special increment function, we will use this function in later sections for examples.

12.6 The retrieval appoach

An application of the incremental (searcher satisfaction) model is the decision about the documents to be presented to the searcher in response to request q. Typically, the IR system will offer the documents in decreasing order of estimated relevance. Suppose x is the next relevant document, while the documents in S have already been presented to the searcher. In the proposed approach, document x will be presented if (and only if) it offers something *new* to the searcher. The level of required novelty is a searcher adaptable parameter. If a searcher wants completeness, then this parameter will be set to zero. However, trying to get a broad impression quickly will correspond to a high level of required novelty.

So the system has to decide whether the searcher (1) will find document x relevant, and (2) will find a sufficient amount of new information in this document. The first question in this decision process is if document x is interesting enough in its own right. This is derived from the value $I(?, x) = N(x)$. The next question is whether document x contains information not already presented in previous documents. The value $I(S, x)$ can be used to decide on this question by comparing with the surprise threshold associated with the searcher.

		Relevant ($N(x)$) yes	Relevant ($N(x)$) no
Sufficiently surprised ($I(S,x)$)	yes	*accept*	*reject*
	no	*reject*	*reject*

Suppose the system decides that document x is not to be presented to the searcher. Let y be the next candidate document for presentation. The negative decision about x, however, still allows document y to contain sufficiently new information. This is typically the case when document x is highly relevant, but adds nothing new, i.e., $I(S,x) \ll I(?,x)$.

Example 12.6.1

Suppose an Information Retrieval System is based on the archive consisting of documents $d_1,\ldots,d_6$ characterized as follows:

d_1	*surfing*
d_2	*surfing*
d_3	*surfing in Australia*
d_4	*surfing in Australia in the evening*
d_5	*surfing over the world wide web*
d_6	*problems with searching over the world wide web*

This system employs the relative need defined in section 12.5, while the information need N after being communicated as query 'surfing' is estimated by Jaccard's coefficient (see for example [81]). Table 12.2 summarizes the behavior of the system for the following searcher profiles: (1) an anonymous searcher (no initial knowledge assumed), (2) a searcher who is known to the system as an expert in the context of Australia (i.e. having initial knowledge {Australia}), (3) a surfing expert, and (4) a person who is known as a surfing expert from Australia. The surprise threshold has been set to 0.15 *in this experiment, which is a rather low surprise threshold.*

Note that rejection of document d_2 in all cases is based on the assumption of rather complete characterizations of documents. If this assumption does not hold, then the document identifier might be taken as an extra characterization. It that case, the novelty of document d_2 after the presentation of document d_1 is its being another document. The surprise treshold determines if the searcher will be sufficiently surprised by that fact.

		anonymous		**Australia expert**		**surfing expert**		**surfing expert from Australia**	
doc	*need N*	*I*	*shown*	*I*	*shown*	*I*	*shown*	*I*	*shown*
1	1.0	1.00	yes	1.00	yes	0.00	no	0.00	no
2	1.0	0.00	no	0.00	no	0.00	no	0.00	no
3	0.5	0.25	yes	0.00	no	0.25	yes	0.00	no
4	0.33	0.11	no	0.11	no	0.11	no	0.11	no
5	0.25	0.19	yes	0.19	yes	0.19	yes	0.19	yes
6	0	0.00	no	0.00	no	0.00	no	0.00	no

Figure 12.2: Response to several searcher profiles

In this section the retrieval context for increment functions has been considered. In the next section we will focus at the optimal sequence to rank documents in this retrieval context.

12.7 The quality of document sequences

In the previous section, the increment function is used to filter out documents which do not present sufficient new information. However, the order in which the ranking mechanism of the retrieval system initially puts the documents is not disturbed. The intention of this section is to choose the best ordering of (filtered) documents to improve searcher satisfaction when incrementally viewing the documents.

12.7.1 The sequential increment function

In this section we focus on the effects of presenting a series of documents. For this purpose, we accumulate the increment value (for example, the relative need) of each document presented, supposing that the surprise actually accumulates in an additive way for searchers. The increment function is extended to handle document sequences as follows:

$$\begin{aligned}\widehat{I}(S, \langle\rangle) &= 0 \\ \widehat{I}(S, \langle x_1, \ldots, x_n\rangle) &= I(S, x_1) + \widehat{I}(S \cup \{x_1\}, \langle x_2, \ldots, x_n\rangle)\end{aligned}$$

The iterative version of this definition is formulated as:

Lemma 12.7.1

$$\widehat{I}(S, \langle x_1, \ldots, x_n\rangle) = \sum_{i=1}^{n} I(S \cup \cup_{j=1}^{i-1} \{x_j\}, x_i)$$

The function $\widehat{I}$ is referred to as the *sequential increment function*. Thus, the total effect of presenting the documents $\langle x_1, \ldots, x_n\rangle$ (in that order), starting from initial knowledge S, consists of the increment value $I(S, x_1)$ augmented with the cumulative effect of presenting $\langle x_2, \ldots, x_n\rangle$ starting from initial knowledge $S \cup \{x_1\}$. **sequential increment function**

As a consequence, $\widehat{I}(S, \langle x_1, \ldots, x_n\rangle)$ quantifies the total effect of presenting the documents $x_1, \ldots, x_n$ in that order to a searcher having initial knowledge S. Analogously we will introduce the function $\widehat{N}$, referred to as the *sequential relevance* function, to cover the need for document sequences: **sequential relevance**

$$\widehat{N}(\langle x_1, \ldots, x_n\rangle) = \widehat{I}(?, \langle x_1, \ldots, x_n\rangle)$$

The order in which the retrieval result is presented will be important for a searcher. In other words, $\widehat{I}(S, \langle x, y\rangle)$ and $\widehat{I}(S, \langle y, x\rangle)$ will not be equal in general. The following property majorizes the outcome of $\widehat{I}$ in terms of its components:

Lemma 12.7.2

$$\widehat{I}(S, \langle x_1, \ldots, , x_k\rangle) \leq \sum_{i=1}^{n} I(S, x_i)$$

Proof:

We use induction on the sequence length n. For $n = 0$, the property is obvious since $\widehat{I}(S, \langle\rangle) = 0$. Next suppose the property has been proven for all sequence of length $n-1$. Then applying the induction hypothesis to the definition of $\widehat{I}$ leads to:

$$\begin{aligned}\widehat{I}(S, \langle x_1, \ldots, x_n\rangle) &= I(S, x_1) + \widehat{I}(S \cup \{x_1\}, \langle x_2, \ldots, x_n\rangle) \\ &\leq I(S, x_1) + \sum_{i=2}^{n} I(S \cup \{x_1\}, x_i)\end{aligned}$$

Using IM2, we conclude $I(S \cup \{x_1\}, x_i) \leq I(S, x_i)$ $(2 \leq i \leq n)$, which leads to the required result.

The significance of the above lemma is that an upper bound is presented for the sequential increment function. In section 12.8.3 we will show that this upper bound can be reached if the documents from the sequence are suffiently different from each other.

12.7.2 Examples

In the next example, the behaviour of the sequential increment function is demonstrated.

Example 12.7.1

The presentation quality of the retrieval response, depending on the profiles from the earlier *example, are summarized in table 12.3. This table presents the outcome of* $\widehat{I}(P, \{d_1, \ldots, d_{i-1}\})$ *in the colums with header* tot *for the various profiles P of this example. As a consequence, the retrieval results appear to be most surprising to the anonymous searcher (total score 1.55), but rather boring for surfing experts from Australia (scoring only 0.30).*

		anonymous		**Australia expert**		**surfing expert**		**surfing expert from Australia**	
doc	*need*	*I*	*tot*	*I*	*tot*	*I*	*tot*	*I*	*tot*
d_1	1.0	1.00	1.00	1.00	1.00	0.00	0.00	0.00	0.00
d_2	1.0	0.00	1.00	0.00	1.00	0.00	0.00	0.00	0.00
d_3	0.5	0.25	1.25	0.00	1.00	0.25	0.25	0.00	0.00
d_4	0.33	0.11	1.36	0.11	1.11	0.11	0.36	0.11	0.11
d_5	0.25	0.19	1.55	0.19	1.30	0.19	0.55	0.19	0.30
d_6	0	0.00	1.55	0.00	1.30	0.00	0.55	0.00	0.30

Figure 12.3: Incremental satisfaction for different kinds of searchers

Next we consider the effects of different document sequences on the outcome of the incremental satisfaction function.

Example 12.7.2

In table 12.4 we see the effect of the learning curve on different order of presentation. First we give the total effect if the results are presented in order of their need, next the reverse order is shown, and after that four random orders are used.

Form this table we see that for surfing experts from Australia the outcome seems to be determined by the mutual order of the documents d_5 *and* d_6 *(scoring 0.30 if* d_5 *precedes* d_6*, and 0.11 otherwise).*

order	anonymous	Australia expert	surfing expert	surfing expert from Australia
$d_1, d_2, d_3, d_4, d_5, d_6$	1.55	1.30	0.55	0.30
$d_6, d_5, d_3, d_3, d_2, d_1$	0.28	0.17	0.22	0.11
$d_6, d_2, d_4, d_3, d_1, d_5$	1.22	1.11	0.22	0.11
$d_1, d_5, d_4, d_3, d_6, d_2$	1.41	1.30	0.41	0.30
$d_6, d_5, d_3, d_2, d_4, d_1$	0.42	0.17	0.36	0.11
$d_1, d_2, d_3, d_4, d_6, d_5$	1.36	1.11	0.36	0.11

Figure 12.4: Various presentation sequences

The order of relevance is not necessarily lead to a maximal value of sequential relevance. For example, consider documents d_4 and d_5 in the context of the query 'surfing in the evening'. Then:

1. $N(d_4) = \frac{1}{2}$
2. $N(d_5) = \frac{2}{3}$

There are two orderings to present these two documents with the following sequential relevance associated (using the relation $\widehat{N}(\langle x, y\rangle) = N(x) + N(y) - N(y)\, Sim(\{y\}, x)$):

1. $\widehat{N}(\langle d_4, d_5\rangle) = N(d_4) + N(d_5) - N(d_5)\, Sim_c(\{d_5\}, d_4)] = \frac{1}{2} + \frac{2}{3} - \frac{2}{3} \cdot \frac{1}{2} = 0.9444..$
2. $\widehat{N}(\langle d_5, d_4\rangle) = N(d_5) + N(d_4) - N(d_4)\, Sim_c(\{d_4\}, d_5)] = \frac{2}{3} + \frac{1}{2} - \frac{1}{2} \cdot \frac{1}{3} = 0.9166..$

As a result, while $N(d_4) < N(d_5)$, for the sequential relevance we find $\widehat{N}(\langle d_4, d_5\rangle) > \widehat{N}(\langle d_5, d_4\rangle)$. However, if the increment function is based on a symmetric similarity function, then the order of relevance corresponds to maximal symmetric similarity:

Lemma 12.7.3 Let *Sim* be symmetric in the following sense: $\forall_{x,y}\left[Sim(\{x\},y) = Sim(\{y\},x)\right]$. Then:

$$\widehat{N}(\langle x,y\rangle) > \widehat{N}(\langle y,x\rangle) \iff N(x) > N(y)$$

Proof:

Via a simple computation we find:

$$\widehat{N}(\langle x,y\rangle) - \widehat{N}(\langle y,x\rangle) = N(x)\,Sim(\{x\},y) - N(y)\,Sim(\{y\},x)$$

From the symmetry of the similarity function, the result directly follows.

12.7.3 The quality function

The order in which documents are presented to the searcher has, in general, an impact on the total searcher satisfaction. Therefore, the degree in which the information contained in a collection S is covered by the information from a collection T can be defined as the minimal sequential increment value of any order of presentation for the documents from T, assuming S as initial knowledge:

$$\widehat{I}(S,T) = \min\left\{\widehat{I}(S,t) \;\middle|\; t \in \pi(T)\right\}$$

where $\pi(T)$ is the set of all (non-repeating) sequences which can be obtained from the elements of T. This definition corresponds to the most smooth presentation of documents fom T, i.e. the presentation which minimizes suprises to the searcher. In this chapter, we propose to use the fraction

$$\frac{\widehat{I}(S,t)}{\widehat{I}(S,T)}$$

as a quality measure for the retrieval result. This quality measure only depends on the documents presented, and quantifies the order of presentation.

Another application of the function $\widehat{I}$ is when a searcher is aiming at reaching a certain amount (I_0) of surprising information with a minimal number of documents (i.e. a minimal effort). This can be accomplished, using the function $\widehat{I}$, as follows. Let $\pi_n(T)$ be the subset of $\pi(T)$, consisting of all sequences of length n. Furthermore, let

$$T_n = \left\{t \in \pi_n(T) \;\middle|\; \widehat{I}(S,t) > I_0\right\}$$

Then, in order to satisfy the searcher requirements, the minimal value of n, for which $T_n \neq$? should be determined.

Having considered the quality of document sequences, we discuss basic properties of the incremental model in the next section. We will focus on the following content relations between documents: (1) the reductional effect, (2) information containment and (3) information preclusion.

12.8 Properties of the incremental model

In this section we focus on relations between documents. We particularly consider relations defined in terms of the increment function. We first quantify the amount in which a document influences the need for another document. This is used as a basis to define document containment and preclusion. Document containment and preclusion play a role in logical models for Information Retrieval. The containment relation gives a clue for aboutness between a query and a document, while the preclusion relation gives evidence for not being about. The law of minimal knowledge requires the transitivity of the document containment relation, and is the third axiom of the incremental model.

12.8.1 Reductional effect between documents

First we introduce the function ε as the reductional effect between documents. The reductional effect of document x on document y expresses the influence of presenting document y on the need for x as follows:

$$\varepsilon(x,y) = N(x) - I(\{y\},x)$$

This function may be generalized to involve prior knowledge in the obvious way: $\varepsilon_S(x,y) = I(S,x) - I(S \cup \{y\}, x)$. Note that $\varepsilon(x,y) \geq 0$ follows from axiom IM2.

Example 12.8.1

For the situation of example 12.5.3 the reductional effect of document d_2 on document d_1 amounts to:

$$\begin{aligned} \varepsilon(d_2,d_1) &= N(d_2) - I(\{d_1\},d_2) \\ &= N(d_2) - N(d_2)\,(1 - \mathit{Sim}(B,A)) \\ &= \frac{k}{m+k} N(d_2) \end{aligned}$$

Thus, the reductional effect decreases with more new terms in d_2.

Next we consider the relation between order of presentation of documents and their reductional effect. The effect of changing the order of presentation of two successive documents amounts to the difference between their mutual reductional effects:

Lemma 12.8.1 $\widehat{I}(S, \langle x,y,\ldots\rangle) - \widehat{I}(S, \langle y,x,\ldots\rangle) = \varepsilon_S(x,y) - \varepsilon_S(y,x)$

Proof:

Applying the definition the increment function for document sequences we get:

$$\begin{aligned} &\widehat{I}(S, \langle x,y,z_1,\ldots,z_n\rangle) - \widehat{I}(S, \langle y,x,z_1,\ldots,z_n\rangle) \\ &\quad= \Big(I(S,x) + I(S \cup \{x\}, y) + \widehat{I}(S \cup \{x,y\}, z_1,\ldots,z_n)\Big) \\ &\qquad - \Big(I(S,y) + I(S \cup \{y\}, x) + \widehat{I}(S \cup \{y,x\}, z_1,\ldots,z_n)\Big) \\ &\quad= \varepsilon_S(x,y) - \varepsilon_S(y,x) \end{aligned}$$

Thus, the function ε can be used to decide on the order between documents (at run time). In the next sections we consider special relations between documents (containment and preclusion), and their effect on the sequential increment function.

12.8.2 Information containment

An important notion for information retrieval models is information containment for documents. This notion for example is used as a basis for aboutness in the context of matching information objects with queries. In terms of the incremental model, the information containment relation is defined as:

$$x \subseteq_I y \;\equiv\; I(\{y\}, x) = 0$$

where $x \subseteq_I y$ is verbalized as: *the information in x is contained within y*, in the context of the information need represented by I. In the sequel, we will omit the index I, and denoting information containment as $\subseteq$. If the information in document x is contained within y, then presenting document y eliminates the need for document x:

Lemma 12.8.2 $x \subseteq y \wedge y \in S \Rightarrow I(S,x) = 0$

Proof:

Suppose $x \subseteq y$, then $I(\{y\}, x) = 0$. Let $y \in S$, then form axiom IM2 we conclude $I(\{y\}, x) \geq I(S,x)$, and thus $I(S,x) = 0$.

Irrelevant documents (i.e. $N(x) = 0$) do not contain any information that is relevant for the searcher. Such documents thus can be seen as empty-information objects. As a consequence, irrelevant documents have special properties:

Lemma 12.8.3 $N(x) = 0 \Rightarrow x \subseteq y$

From axiom IM1 it directly follows that the relation $\subseteq$ is reflexive. A next requirement to the incremental function is the containment relation to be transitive as this makes the containment relation a partial order on documents. This partial order plays a vital role in the reasoning process within logical models of Information Retrieval (see [63] or [28]). Transitivity is enforced by the following axiom:

$$\text{IM3} \quad \textit{law of effective knowledge} \quad x \subseteq y \Rightarrow I(S,x) \leq I(S,y)$$

So, if the information from document x is contained within y, then document x can not be more surprising than document y.

Lemma 12.8.4 $x \subseteq y \wedge y \subseteq z \Rightarrow x \subseteq z$

Proof:

Suppose $x \subseteq y \wedge y \subseteq z$. From $x \subseteq y$, we conclude from IM3: $I(\{z\},x) \leq I(\{z\},y)$. From the definition of $y \subseteq z$ we conclude $I(\{z\},y) = 0$. As a consequence, $I(\{z\},x) = 0$, or, $x \subseteq z$.

The implication from IM3 may be reversed:

Lemma 12.8.5 $\forall_S \, [I(S,x) \leq I(S,y)] \Rightarrow x \subseteq y$

Proof:

Suppose $\forall_S \, [I(S,x) \leq I(S,y)]$. By substituting $\{y\}$ for S, we get: $I(\{y\},x) \leq I(\{y\},y) = 0$. In the latter step, axiom IM1 is applied.

Example 12.8.2

The relative need of section 12.3 satisfies axiom IM3. In order to prove this, note that $I(\{y\},x) = 0$ is equivalent with $Sim(\chi(y),\chi(x)) = 1$, or, $\chi(x) \subseteq \chi(y)$. So, from $x \subseteq y$ and $y \subseteq z$ we derive $\chi(x) \subseteq \chi(y)$ and $\chi(y) \subseteq \chi(z)$, and thus $\chi(x) \subseteq \chi(z)$, from which $x \subseteq z$ directly follows.

Lemma 12.8.6 $x \subseteq y \wedge N(x) > N(y) \Rightarrow \widehat{N}(\langle x,y \rangle) > \widehat{N}(\langle y,x \rangle)$

Proof:

Suppose $x \subseteq y$, then $N(x) = 0$ or $Sim(\{x\},y) = 1$. The case $N(x) = 0$ is obvious. So suppose $Sim(\{x\},y) = 1$. Then

$$\begin{aligned}
\widehat{N}(\langle x,y \rangle) - \widehat{N}(\langle y,x \rangle) &= N(x)\, Sim(\{x\},y) - N(y)\, Sim(\{y\},x) \\
&= N(x) - N(y)\, Sim(\{y\},x) \\
&\geq N(y) - N(y)\, Sim(\{y\},x) \\
&= I(\{x\},y) \\
&\geq 0
\end{aligned}$$

Information containment is, generally, not a symmetric relation between documents: different documents may mutually contain each others information. As a consequence, documents x and y are considered *equally informative*, denoted as $x \approx y$, if:

$$x \approx y \;\equiv\; x \subseteq y \wedge y \subseteq x$$

If two documents are equally informative, then from lemma **??** it follows that, under no circumstance, one of those documents can add something new to the other.

Example 12.8.3

For the documents from example 12.6.1 we have $d_2 \subseteq d_3$ and $d_1 \approx d_2$.

The relation $\approx$ is an equivalence relation for documents. If one document of an equivalence class is found to be relevant for some query, then the other documents from that class are equally relevant. The notion of information preclusion can be used to further distinguish between the documents within an equivalence class.

12.8.3 Information preclusion

Informational preclusion for IR purposes has been introduced (see [108] or [17]) as an interesting property which can be used to determine non-aboutness. Preclusion of documents is introduced in this chapter in two steps. First we consider the notion *absolutely not about*. A document y is absolutely not about document x, denoted as $x\rfloor_I y$, if:

$$x\rfloor_I y \;\equiv\; I(\{y\}, x) = N(x)$$

The relation $x\rfloor_I y$ expresses that presenting document y does not influence the need for document x. The index I will be omitted in the rest of this chapter. Irrelevant documents have a special place: Specifically, they have nothing to do with any other document:

Lemma 12.8.7 $N(x) = 0 \Rightarrow x\rfloor y$

The preclusion relaion is a reflexive relation. Note that the relation $\rfloor$, in general, is not a symmetric relation. However, for special increment functions (such as the relative need) the relation $\rfloor$ is symmetric. For the relative need function, the relation $x\rfloor y$ corresponds to $\chi(x) \cap \chi(y) = ?$. Later we will introduce *preclusion* as the symmetric variant of the relation *absolutely not about*. The nature of the preclusion relation is layed down in the following axiom:

IM4 *law of independent knowledge* $x\rfloor y \Rightarrow I(S \cup \{y\}, x) = I(S, x)$

Example 12.8.4

The relative need function satisfies axiom IM4. This directly follows from the observation that $x\rfloor y$ is equivalent in this case with $\chi(x) \cap \chi(y) = ?$. As a consequence, the expression $\chi(x) \cap (\sigma(S) \cup \chi(y))$ can be rewritten as $\chi(x) \cap \sigma(S)$.

Lemma 12.8.8 $I(S, x) = N(x) \wedge y \in S \Rightarrow x\rfloor y$

Next we consider the effect of the relation *absolutely not about* on the order of presentation. If document x is absolutely not about document y, or $x\rfloor y$, then presenting document x does not influence the need for document y. As a consequence, presenting x before y is not worse than the other way around.

Lemma 12.8.9 $x\lfloor y \Rightarrow \widehat{N}(\langle x, y\rangle) \geq \widehat{N}(\langle y, x\rangle)$

Proof:

Suppose $x\lfloor y$. From the definition of $\widehat{N}$ and lemma 12.8.1 it follows: $\widehat{N}(\langle x, y\rangle) - \widehat{N}(\langle y, x\rangle) = \widehat{I}(?, \langle x, y\rangle) - \widehat{I}(?, \langle y, x\rangle) = \varepsilon(x, y) - \varepsilon(y, x)$. From $x\lfloor y$ it directly follows that $\varepsilon(y, x) = 0$. As a consequence $\widehat{N}(\langle x, y\rangle) - \widehat{N}(\langle y, x\rangle) = \varepsilon(x, y)) \geq 0$.

For relevant documents y, the relations $y \subseteq x$ and $x\lfloor y$ exclude each other. In other words, if x is absolutely not about y, then the infomation of y cannot be contained within x:

Lemma 12.8.10 If $N(y) > 0$, then $x\lfloor y \Rightarrow y \not\subseteq x$.

Proof:

Suppose y is a relevant document. If $x\lfloor y$ then $I(\{x\}, y) = N(y)$. As $N(y) > 0$, we conclude that $y \subseteq x$ does not hold.

Next we consider the symmetric variant of the relation *absolutely not about*. This is called information *preclusion*. Two documents x and y are said to preclude each other (denoted by $x \perp y$) if the information of these documents is mutually complementary:

$$x \perp y \;\equiv\; x \lfloor y \wedge y \lfloor x$$

As a result, two documents precluding each other, can be presented in any order:

Lemma 12.8.11 $x \perp y \Rightarrow \widehat{I}(S, \langle x, y, \ldots \rangle) = \widehat{I}(S, \langle y, x, \ldots \rangle)$

This is a direct consequence of lemma 12.8.1. Apparently, the upper bound from lemma 12.7.2 is sharp.

Appendix A

The example

```
collectie grootte: 100
aantal termen: 94
max aantal termen per document: 10
karakterisatie per document
  1:   0
  2:   0 47
  3:  31
  4:   0 23 47
  5:  18 75
  6:  15 62
  7:  13 26 40 53 67 80
  8:   0 11 47 70
  9:  10 41 73
 10:   9 37 56
 11:   8 17 25 34 42 51 59 68 76 85
 12:   7 31 78
 13:   7 21 65
 14:   6 13 26 40 53 67 80
 15:   6 25
 16:   0  5 23 47 82
 17:   5 11 22 44 49 71 82 88
 18:   5 20 52 83
 19:   4 89
 20:   4 18 65 75
 21:   4 17 44 58 71 85
 22:   4  8 17 25 34 42 59 68 76 85
 23:   4  8 12 16 24 32 36 53 65 73
 24:   3 15 62 86
 25:   3 15 22 33 41 52 60 71 78 90
 26:   3 50 57 79
 27:   3 13 24 34 45 55 66 76 87
 28:   3 13 26 33 40 53 67 80
 29:   3 22 51 64 74 77 81
 30:   3 12 50
 31:   3
 32:   0  2 11 47 70 88
 33:   2 11 19 28 37 45 54 71 79 88
 34:   2  5 11 22 44 49 71 82 88
 35:   2 29 42 51 64 91
 36:   2 10 41 57 73
 37:   2 20 30 45 50 55 68 76 78 83
 38:   2 44 49
```

```
39:   2 38 53
40:   2 32 37 56 84
41:   2  6 13 25 27 41 48 50 71 82
42:   2  8 22 35 49 76 89
43:   2  4 17 24 34 45 48 76 85 91
44:   2  8 12 17 34 42 51 68 76 85
45:   2  8 33 39 64 71
46:   2  4  8 12 16 32 36 53 65 73
47:   0  2
48:   1  7 31 78 90
49:   1  9 11 21 30 44 55 59 65 86
50:   1  7 11 26 30 45 63 67 82 86
51:   1  7 23 29 35 46 79 90
52:   1 28 72 75 86
53:   1  5 14 17 23 30 37 67 69 72
54:   1  6 17 27 38 48 59 69 80 90
55:   1 27 32 41 44 49 52 61 66 92
56:   1 13 26 40 53 63 67 73 80
57:   1 61
58:   1 25 32 38 48 58 84 87
59:   1 11 14 19 25 30 46 55 65 71
60:   1  6 25 53
61:   1  4 12 30 36 50 52 58 77 80
62:   1 48
63:   1  5 23 46 82 91
64:   0  1  5 23 44 47 82
65:   1 13 20 23 41 88
66:   1  5 14 22 39 56 65 74 82 91
67:   1 12 33 37 58 63 72 74 82 89
68:   1  5 11 22 35 44 49 71 82 88
69:   1  5 17 21 34 42 66 70 74 87
70:   1 21 32 45 48 61 72
71:   1 30 34 39 42 48 51 63 67 92
72:   1 20 28 52 67 83
73:   1  3 10 11 27 28 34 55 72 81
74:   1 10 15 22 25 27 38 81 86 88
75:   1  5 20 23 38 42 57 61 76 80
76:   1 22 24 69
77:   1 15 20 45 63 64 65 70 75 86
78:   1 19 26 48
79:   1 11 17 20 32 45 67 77 79 84
80:   1 16 18 28 42 75
81:   1  4  8 11 15 22 32 56 60 81
82:   1  6 13 20 25 41 50 59 71 82
83:   1  3 10 11 12 18 30 35 37 43
84:   1  4 11 17 44 58 71 85
85:   1  9 17 21 23 54 65 89
86:   1  2  8 17 24 45 59 69 85 89
87:   1  7 17 27 52 56 88
88:   1  6  8 17 25 34 38 42 51 68
89:   1  2  4  5 10 21 26 38 52 76
90:   1  4 16 35 48 66 79
91:   1  3  9 27 63 83
92:   1  2  4  8 16 32 36 53 65 73
93:   1
94:   0  1
95:   0 93
96:   0  3 15 62 86 92
```

```
 97:   0  7  8 42 46 48 60 67 78 91
 98:   0  5 15 32 51 57 69 74 76 90
 99:   0 23 29 40 43 66 69 77 80 89
100:   0  3 15 22 33 52 60 78 88 90
karakterisatie per term
  0: ***************
  1: ***********************************************
  2: *******************
  3: *************
  4: *************
  5: *************
  6: *******
  7: *******
  8: ************
  9: ****
 10: ******
 11: **************
 12: *******
 13: ********
 14: ***
 15: *********
 16: *****
 17: **************
 18: ****
 19: ***
 20: ********
 21: ******
 22: ***********
 23: **********
 24: *****
 25: **********
 26: *******
 27: *******
 28: *****
 29: ***
 30: ********
 31: ***
 32: **********
 33: *****
 34: *********
 35: *****
 36: ****
 37: ******
 38: *******
 39: ***
 40: *****
 41: *******
 42: **********
 43: **
 44: *********
 45: *********
 46: ****
 47: ******
 48: **********
 49: ******
 50: ******
 51: *******
 52: ********
 53: *********
 54: **
 55: *****
 56: *****
 57: ****
 58: *****
 59: ******
 60: ****
 61: ****
 62: ***
 63: ******
 64: ****
 65: **********
 66: *****
 67: **********
 68: *****
 69: ******
 70: ****
 71: ***********
 72: *****
 73: ******
 74: *****
 75: *****
 76: **********
 77: ****
 78: ******
 79: *****
 80: ********
 81: ****
 82: ***********
 83: ****
 84: ***
 85: *******
 86: *******
 87: ***
 88: *********
 89: ******
 90: ******
 91: *****
 92: ***
 93: *
gezocht: documenten met term: 5
 16:   0  5 23 47 82
 17:   5 11 22 44 49 71 82 88
 18:   5 20 52 83
 34:   2  5 11 22 44 49 71 82 88
 53:   1  5 14 17 23 30 37 67 69 72
 63:   1  5 23 46 82 91
 64:   0  1  5 23 44 47 82
 66:   1  5 14 22 39 56 65 74 82 91
 68:   1  5 11 22 35 44 49 71 82 88
 69:   1  5 17 21 34 42 66 70 74 87
 75:   1  5 20 23 38 42 57 61 76 80
 89:   1  2  4  5 10 21 26 38 52 76
 98:   0  5 15 32 51 57 69 74 76 90
```

Appendix B

Mathematical Background

B.1 Characteristic Functions

A simple way to encode sets as computabel objects is the representation via a characteristic function. Therefore we restrict ourselves to some universe U. The characterstic function of some subset A of the universe U is a function

$$\kappa_A : U \mapsto \{0,1\}$$

defined by:

$$\kappa_A(x) = \begin{cases} 1 & \text{if } x \in A \\ 0 & \text{otherwise} \end{cases}$$

The elementary set operations are implemented as follows in terms of this representation:

$$\begin{aligned} \kappa_{A \cap B}(x) &= \kappa_A(x)\,\kappa_B(x) \\ \kappa_{\overline{A}} &= 1 - \kappa_A(x) \\ \kappa_{A \cup B}(x) &= \kappa_A(x) + \kappa_B(x) - \kappa_A(x)\,\kappa_B(x \end{aligned}$$

B.2 Functional Calculus

In this section we consider the class of functions from a set X into the real interval $[0,1]$. First we introduce some operators on such functions:

Definition B.2.1 (functional multiplication)
Let f and g be elements from $X \mapsto [0,1]$, then the function $f \bullet g$ is defined by: $(f \bullet g)(x) = f(x)g(x)$.

Other arithmetic operators (+, -, /) are defined analogously.

Definition B.2.2
The function $1_X : X \mapsto [0,1]$ takes value 1 for each argument: $1_X(x) = 1$.

Definition B.2.3 (functional inversion)
Let f be an element from $X \mapsto [0,1]$, then the function $\overline{f}$ is defined by: $\overline{f} = 1_X - f$.

The number of elements of a set is generalized to taking the 1-norm of its characteristic function:

$$\begin{aligned} |X| &= \#_{x \in X}(\kappa_X(x) = 1) \\ &= \sum_{x \in X} \kappa_X(x) \\ &= |\kappa_X| \end{aligned}$$

For characteristic functions, the following relation may be employed: $\kappa_X^2 = \kappa_X$, leading to another way to generalize number of elements of a set:

$$\begin{aligned} |X| &= \sum_{x \in X} \kappa_X(x) \\ &= \sum_{x \in X} \kappa_X^2(x) \\ &= (\|\kappa_A\|_2)^2 \end{aligned}$$

Furthermore we use the following relations:

$$\begin{aligned} \kappa_{X \cap Y} &= \kappa_X \bullet \kappa_Y \\ \kappa_{X \cup Y} &= \kappa_X + \kappa_Y - \kappa_X \bullet \kappa_Y \end{aligned}$$

expression	*translation*
$\|A\|$	$\|\kappa_A\|$
$\|A\|$	$(\\|\kappa_A\\|_2)^2$
$A \cap B$	$\kappa_A \bullet \kappa_B$
$A \cap B$	$\min(\kappa_A, \kappa_B)$
$A \cup B$	$\kappa_A + \kappa_B - \kappa_A \bullet \kappa_B$

Figure B.1: Translation table

B.3 Probability Theory

odds Frequently we will encounter in expressions the ratio between a probability and its counterprobability. This ratio is referred to as the *odds* of the corresponding event. In daily life this is encountered in expressions like: chances are 1 against 5 that we will throw 6 with a dice. Note that the numbers in this expression are simplified by removing common factors. So we would not say: chances are 2 against 10 to throw 6 with a dice. If chances are p against n, then the associated probability is $p/(p+n)$. Formally, the odds of an event is an element from $[0, \infty) \cup \{\infty\}$, defined as follows:

Definition B.3.1

$$Odds(A) = \begin{cases} \dfrac{Prob(A)}{1 - Prob(A)} & \textit{if } Prob(A) \neq 1 \\ \infty & \textit{otherwise} \end{cases}$$

As a result, the odds of throwing 6 with a dice are $\frac{1}{6}/\frac{5}{6} = 1/5$. In other words, 1 against 5. We will also write $Odds(p)$ as the odds of an event with probability p. Small probabilities have small odds associated, the more certain, the higher the corresponding odds.

Exercise B.3.1

Proof the following properties:

1. $Odds\left(\frac{1}{i}\right) = \frac{1}{i-1}$
2. $Odds\left(1 - \frac{1}{i}\right) = i - 1$
3. $Odds(A)\,Odds\left(\overline{A}\right) = 1$

From the odds of an event, the probability can be computed. Let $Odds(()A)$ be the odds of event A, then the chances of event A to happen are $Odds(()A)$ against 1. The probability is derived as above:

Lemma B.3.1

$$Prob(A) = \begin{cases} \dfrac{Odds(A)}{Odds(A)+1} & \text{if } Odds(A) \neq \infty \\ 1 & \text{otherwise} \end{cases}$$

The probabilistic view on information retrieval is based on conditional probabilities. Conditional probabilities are defined by:

Definition B.3.2

$$Prob\big(A\,\big|B\big) = \frac{Prob(A \cap B)}{Prob(B)}$$

The following theorem is most useful when dealing with conditional probabilities. It states that the order in which multiple events are serialized does not reflect their joint probability. **Bayes' Rule**

Theorem B.3.1 (Bayes Rule) $Prob\big(A\,\big|B\big)Prob(B) = Prob\big(B\,\big|A\big)Prob(A)$

The corresponding rule for odds is easily derived:

Lemma B.3.2 $Odds\big(A\,\big|B\big) = Odds(A)\ Infl\big(B\,\big|A\big)$

where $Infl\big(B\,\big|A\big)$ is the influence of A on B:

$$Infl\big(B\,\big|A\big) = \frac{Prob\big(B\,\big|A\big)}{Prob\big(B\,\big|\overline{A}\big)}$$

Appendix C

Indexing Example

We consider the followoing documents (abstract from real papers):

d_1 [104]

A main issue of Object Orientation is reuse of software components. Software reuse will be effective when mechanisms are provided that support reuse at a conceptual level. In traditional system analysis several methods (such as ISAC and DFD) have been proposed from which a global construction of the system in terms of smaller components can be derived. In this paper we introduce the multi-pipe model, a universal vision on such components, and an algebra that describes the construction of new components. Finally we propose an information retrieval system for a repository of reusable components.

d_2 **Title**

In this article, the incremental searcher satisfaction model for Information Retrieval is introduced. In this model, documents are not presented according to decreasing relevancy only, but also on the level of novelty in the context of the documents previously presented. Documents which are judged to be insufficiently surprising (according to a searcher determined threshold) are not presented to the searcher. This is especially useful for Information Retrieval in certain contexts (e.g. Internet applications such as search engines), when a searcher does not want all relevant documents to be shown, but only have a global idea of the variety of what the corpus may contain on a topic. Important properties of this model are discussed, such as the relation between the reductional effect and the order of presentation.

d_3 **Title**

In this paper we discuss cognitive requirements for a natural language based modeling process for those who are involved in this process, i.e. domain experts and system analysts. For both domain experts and system analysts these requirements are presented as their base skills. The interaction between domain experts and system analysts is guided by these skills. Furthermore, a way of working in relation with these skills is outlined.

d_4 **Title**

Random search trees have the property that their depth depends on the order in which they are built. They have to be balanced in order to obtain a more efficient storage-and-retrieval data structure. Balancing a search tree is time consuming. This explains the popularity of data structures which approximate a balanced tree but have lower amortised balancing costs, such as AVL trees, Fibonacci trees and 2-3 trees. The algorithms for maintaining these data structures efficiently are complex and hard to derive. This observation has led to insertion algorithms that perform local balancing around the newly inserted node, without backtracking on the search path. This is also called a fringe heuristic. The resulting class of trees is referred to as 1-locally balanced trees, in this note referred to as hairy trees. In this note a simple analysis of their behaviour is povided.

d_5 **Title**

In this paper we discuss the construction of an automated information system for a medium-sized collection of visual reproductions of art objects. Special attention is payed to the economical aspects of such a system, which appears to be mainly a problem of data entry. An approach is discussed to make this feasible, which also strongly provokes consistency between descriptions.

Another main target of such a system is the capability for effective disclosure. This requires a disclosure mechanism on descriptions which is easy to handle by non technical users. We show the usefulness of query by navigation for this purpose. It allows the searcher to stepwise build a query in terms of (semi-)natural language. At each step, the searcher is presented with context sensitive information.

The resulting system is described and we discuss an experiment of its use.

The internal frequencies of the terms in these documents are:

Term	Doc 1	Doc 2	Doc 3	Doc 4
1-locally	0	0	0	1
2-3	0	0	0	1
a	5	4	2	5
according	0	2	0	0
algebra	1	0	0	0
algorithms	0	0	0	2
all	0	1	0	0
allows	0	0	0	0
also	0	1	0	1
amortised	0	0	0	1
an	2	0	0	0
analysis	1	0	0	1
analysts	0	0	3	0
and	2	1	3	2
another	0	0	0	0
appears	0	0	0	0
applications	0	1	0	0
approach	0	0	0	0
approximate	0	0	0	1
are	1	4	2	2
around	0	0	0	1
art	0	0	0	0
article	0	1	0	0
as	1	2	1	3
aspects	0	0	0	0
at	1	0	0	0
attention	0	0	0	0
automated	0	0	0	0
avl	0	0	0	1
....	...	...	...	...
document size:	94	127	64	139
document entropy:	5.770	5.850	5.292	6.016

Term	Doc 1	Doc 2	Doc 3	Doc 4	Doc 5
1-locally	-	-	-	0.0116	-
2-3	-	-	-	0.0116	-
a	-	-	-	-	-
according	-	0.0253	-	-	-
algebra	0.0171	-	-	-	-
algorithms	-	-	-	0.0232	-
all	-	0.0127	-	-	-
allows	-	-	-	-	0.0117
also	-	0.0040	-	0.0037	0.0037
amortised	-	-	-	0.0116	-
an	0.0195	-	-	-	0.0201
analysis	0.0097	-	-	0.0066	-
analysts	-	-	0.0754	-	-
and	-	-	-	-	-
another	-	-	-	-	0.0117
appears	-	-	-	-	0.0117
applications	-	0.0127	-	-	-
approach	-	-	-	-	0.0117
approximate	-	-	-	0.0116	-
are	0.0024	0.0070	0.0070	0.0032	-
around	-	-	-	0.0116	-
art	-	-	-	-	0.0117
article	-	0.0127	-	-	-
as	0.0024	0.0035	0.0035	0.0048	-
aspects	-	-	-	-	0.0117
at	0.0097	-	-	-	0.0067
attention	-	-	-	-	0.0117
automated	-	-	-	-	0.0117
avl	-	-	-	0.0116	-

Bibliography

[1] M. Agosti, A. Archi, R. Colotti, R.M. Di Giorgi, G. Gradenigo, B. Inghirami, P. Matiello, R. Nannuci, and M. Ragona. New prospectives in information retrieval techniques: a hypertext prototype in environmental law. In *Online Management 89, Proceedings 13th International Online Information*, pages 483–494, London, United Kingdom, 1989.

[2] M Agosti, A Bleeker, and Th.P. van der Weide. Content-based Image Retrieval through Hypermedia Browsing. to appear, Department of Information Systems, University of Nijmegen, The Netherlands, 1996.

[3] M. Agosti, R. Colotti, and G. Gradenigo. A two-level hypertext retrieval model for legal data. In A. Bookstein, Y. Chiaramella, G. Salton, and V.V. Raghavan, editors, *Proceedings of the 14th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 316–325, Chicago, Illinois, October 1991. ACM Press.

[4] M. Bates. Where should the person stop and the information search start? *Information, Processing and Management*, 26(5):575–591, 1990.

[5] N. J. Belkin, C. Cool, A. Stein, and U. Thiel. Cases, scripts and information-seeking strategies: On the design of interactive information retrieval systems. *Expert Systems with Applications*, 9(3):379–395, 1995.

[6] N.J. Belkin, P.G. Marchetti, and C. Cool. BRAQUE: Design of an interface to support user interaction in information retrieval. *Information Processing & Management*, 29(3):325–344, 1993.

[7] P. Bennett. *A Course in Generalized Phrase Structure Grammar.* UCL Press Limited, University College London, United Kingdom, 1995.

[8] F.C. Berger and P. van Bommel. Personalized Search Support for Networked Document Retrieval Using Link Inference. In R.R. Wagner and H. Thoma, editors, *Proceedings of the 7th International Conference on Database and Expert System Applications (DEXA)*, pages 802–811, Zurich, Switzerland, September 1996. Springer-Verlag.

[9] C. Berrut. *Une méthode d'indexation fondée sur l'analyse sémantique de documents spécialisés. Le prototype RIME et son application à un corpus médical.* PhD thesis, Laboratoire de Génie Informatique, Université Joseph Fourier, Grenoble, France, December 1988.

[10] D.C. Blair. *Language and Representation in Information Retrieval.* Elsevier Science Publishers, Amsterdam, 1990.

[11] D.C. Blair. *Language and Representation in Information Retrieval.* Elsevier, 1990.

[12] R. Bosman and R. Bouwman. The Automation and Disclosure of a Slides Library. Master's thesis, University of Nijmegen, Nijmegen, The Netherlands, April 1991.

[13] L. de Brabandere. *Les Infoducs.* Editions Duculot, Gembloux, 1985.

[14] F.P. Brooks and K.E. Iverson. *Automatic Data Processing.* John Wiley & Sons, Inc., New York, 1963.

[15] P.D. Bruza. Hyperindices: A Novel Aid for Searching in Hypermedia. In A. Rizk, N. Streitz, and J. Andre, editors, *Proceedings of the European Conference on Hypertext - ECHT 90*, pages 109–122, Cambridge, United Kingdom, 1990. Cambridge University Press.

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

[17] P.D. Bruza and T.W.C. Huibers. Investigating aboutness axioms using information fields. In W. Bruce Croft and C.J. van Rijsbergen, editors, *Proceedings of the Seventeenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 112–121, Dublin, July 1994. Springer-Verlag, Berlin.

[18] P.D. Bruza and Th.P. van der Weide. The Semantics of Data Flow Diagrams. In N. Prakash, editor, *Proceedings of the International Conference on Management of Data*, Hyderabad, India, 1989.

[19] P.D. Bruza and Th.P. van der Weide. Two Level Hypermedia - An Improved Architecture for Hypertext. In A.M. Tjoa and R. Wagner, editors, *Proceedings of the Database and Expert System Applications Conference (DEXA)*, pages 76–83, Vienna, Austria, 1990. Springer-Verlag.

[20] P.D. Bruza and Th.P. van der Weide. The Modelling and Retrieval of Documents using Index Expressions. *ACM SIGIR FORUM (Refereed Section)*, 25(2), 1991.

[21] J.A. Bubenko. Information System Methodologies - A Research View. In T.W. Olle, H.G. Sol, and A.A. Verrijn-Stuart, editors, *Information Systems Design Methodologies: Improving the Practice*, pages 289–318. North-Holland, Amsterdam, The Netherlands, 1986.

[22] J.P. Callan, W. Bruce Croft, and S.M. Harding. The INQUERY retrieval system. In *Proceedings of the Third International Conference on Database and Expert Systems Applications (DEXA 92)*, pages 78–83, Valencia, Spain, 1992. Springer-Verlag, Berlin.

[23] J.P. Callan, Z. Lu, and W. Bruce Croft. Searching distributed collections with inference networks. 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 21–28, Seattle, July 1995. ACM, ACM Press, New York.

[24] I.R. Campbell-Grant and P.J. Robinson. An Introduction to ISO DIS 8613 - Office Document Architecture - and its Application to Computer Graphics. *Computer and Graphics*, 11(4):325–341, 1987.

[25] J.G. Carbonell, Y. Geng, and J. Goldstein. Automated Query-Relevant Summarization and Diversity-Based Reranking. In I. Ferguson, editor, *Proceedings of the IJCAI-97 Workshop on AI and Digital Libraries*, pages 9–14, Nagoya, Japan, August 1997.

[26] A. Cawsey, G. Galliers, S. Reece, and K. Sparck Jones. Automating the librarian: Belief revision as a base for system action and communication with the user. *The Computer Journal*, 35(3):221–232, 1992.

[27] Y. Chiaramella and J.-P. Chevallet. About retrieval models and logic. *The Computer Journal*, 35(3):233–241, 1992.

[28] Y. Chiarmarella and J.P. Chevallet. About Retrieval Models and Logic. *The Computer Journal*, 35(3):233–242, 1992.

[29] C.W. Cleverdon. Comparative efficiency of indexing systems. 1960–1962. 2 Volumes.

[30] C.W. Cleverdon. The significance of the Cranfield tests on index languages. In A. Bookstein, Y. Chiaramella, G. Salton, and V.V. Raghavan, editors, *Proceedings of the Fourteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3–12, Chicago, October 1991. ACM, ACM Press, New York.

[31] C.W. Cleverdon. The Significance of the Cranfield Tests on Index Languages. In A. Bookstein, Y. Chiaramella, G. Salton, and V.V. Raghavan, editors, *Proceedings of the 14th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3–12, Chicago, Illinois, October 1991. ACM Press.

[32] W.S. Cooper. Expected search length: A single measure of retrieval effectiveness based on the weak ordering action of retrieval systems. *American Documentation*, pages 30–41, 1968.

[33] W.S. Cooper. A Definition of Relevance for Information Retrieval. *Information Storage and Retrieval*, 7:19–37, 1971.

[34] W.S. Cooper. A definition of relevance for information retrieval. *Information Storage and Retrieval*, 7:19–37, 1971.

[35] W.S. Cooper. Some inconsistencies and misnomers in probabilistic information retrieval. In A. Bookstein, Y. Chiaramella, G. Salton, and V.V. Raghavan, editors, *Proceedings of the Fourteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 57–61, Chicago, October 1991.

[36] W.S. Cooper. The formalism of probability theory in IR: A foundation for an encumbrance? In W. Bruce Croft and C.J. van Rijsbergen, editors, *Proceedings of the Seventeenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 242–247, Dublin, July 1994. Springer-Verlag, Berlin.

[37] W.S. Cooper. Some inconsistencies and misidentified modeling assumptions in probabilistic information retrieval. *ACM Transactions on Information Systems*, 13(1):100–111, January 1995.

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

[39] T.C. Craven. *String Indexing.* Academic Press, London, United Kingdom, 1986.

[40] T.C. Craven. Adapting of string indexing systems for retrieval using proximity operators. *Information Processing & Management*, 24(2):133–140, 1988.

[41] J. December. Challenges for web information providers. *Computer-Mediated Communication Magazine*, 1(6):8 – 14, October 1994.

[42] N. Denos, C. Berrut, and M. Mechkour. An image system based on the visualization of system relevance via documents. In *(DEXA 97)*, pages 379–395, 1997.

[43] J. Farradane. Relational Indexing Part I. *Journal of Information Science*, 1(5):267–276, 1980.

[44] J. Farradane. Relational Indexing Part II. *Journal of Information Science*, 1(6):313–324, 1980.

[45] N. Fuhr. Probabilistic models in information retrieval. *The Computer Journal*, 35(3):243–255, 1992.

[46] N Fuhr. Probabilistic Models in Information Retrieval. *The Computer Journal*, 35(3):243–255, 1992.

[47] P. Garg. Abstraction mechanisms in hypertext. *Communications of the ACM*, 31(7):863–870, July 1988.

[48] G. Gazdar, E. Klein, G.K. Pullum, and I.A. Sag. *Generalized Phrase Structure Grammar.* Basil Blackwell Publisher Ltd, Oxford, United Kingdom, 1985.

[49] R. Godin, J. Gecsei, and C. Pichet. Design of a Browsing Interface for Information Retrieval. In N.J. Belkin and C.J. van Rijsbergen, editors, *Proceedings of the 12th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 32–37, Cambridge, Massachusetts, June 1989. ACM Press.

[50] G. Gonnet and F. Tompa. Mind Your Grammar: a New Approach to Modelling Text. In *Proceedings of the Thirteenth Conference on Very Large Data Bases*, pages 339–346, Brighton, United Kingdom, 1987.

[51] Grimmett, G.R., and D.R. Stirzaker. *Probability and Random Processes.* Oxford University Press, Oxford, UK, 1992.

[52] D.K. Harman, editor. *The first Text REtrieval Conference (TREC-1)*, NIST Special Publication 500–207, Gaithersburg MD, 1993.

[53] D.K. Harman. Overview of the third Text REtrieval Conference (TREC-3). In D.K. Harman, editor, *The third Text REtrieval Conference (TREC-3)*, pages 1–20, Gaithersburg, Maryland, 1994. ACM Press.

[54] D.K. Harman, editor. *The second Text REtrieval Conference (TREC-2)*, NIST Special Publication 500–215, Gaithersburg MD, 1994.

[55] D.K. Harman, editor. *The third Text REtrieval Conference (TREC-3)*, NIST Special Publication 500–255, Gaithersburg MD, 1994.

[56] E. Hoenkamp, L. Schomaker, P. van Bommel, C.H.A. Koster, and Th.P. van der Weide. Profile - A Proactive Information Filter. Technical Note CSI-N9602, Computing Science Institute, University of Nijmegen, Nijmegen, The Netherlands, 1996.

[57] T.W.C. Huibers and B. van Linder. Formalising intelligent information retrieval agents. In F. Johnson, editor, *Proceedings of the 18th BCS IRSG Annual Colloquium on Information Retrieval Research*, pages 125–143, Manchester, March 1996.

[58] T.W.C. Huibers, B. van Linder, and J.-J. Ch. Meyer. An agent-oriented approach to information retrieval. To appear in proceedings of NAIC'96, Utrecht, The Netherlands, 1996.

[59] D. Hull. Using statistical testing in the evaluation of retrieval experiments. In R. Korfhage, E. Rasmussen, and P. Willett, editors, *Proceedings of the Sixteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 329–338, Pittsburgh, 1993. ACM Press, New York.

[60] ISO. *Information Processing - Text and Office Systems - Standard General MarkUp Language (SGML).* ISO8879, 1986.

[61] D.E. Knuth. *The TEXbook.* Addison-Wesley, Reading, Massachusetts, 1984.

[62] C.H.A. Koster. **DO**cument *ro*outing project, 1997. ESPRIT Project 22716, Embedded HPC.

[63] S. Kraus, D. Lehmann, and M. Magidor. Nonmonotonic Reasoning, Preferential Models and Cumulative Logics. *Artificial Intelligence*, 44:167–207, 1990.

[64] M. Lalmas. The use of logic in information retrieval modelling. Departmental Research Report IR-96-1, Department of Computing Science, University of Glasgow, Scotland, January 1996.

[65] F.W. Lancaster. *Toward Paperless Information Systems.* Academic Press, New York, 1978.

[66] B. van Linder. *Modal Logics for Rational Agents.* PhD thesis, Department of Computer Science, Utrecht University, The Netherlands, June 1996.

[67] B. Logan, S. Reece, and K. Sparck Jones. Modelling information retrieval agents with belief revision. In W. Bruce Croft and C.J. van Rijsbergen, editors, *Proceedings of the Seventeenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 142–151, Dublin, July 1994. Springer-Verlag, Berlin.

[68] D. Lucarella. A Model for Hypertext-Based Information Retrieval. In *Proceedings of the European Conference on Hypertext - ECHT 90*, pages 81–94, Cambridge, United Kingdom, 1990. Cambridge University Press.

[69] D. Lucarella and Z. Zanzi. Information Retrieval from Hypertext: An Approach using Plausible Inference. *Information Processing & Management*, 29(3):299–312, 1993.

[70] M.E. Maron. On Indexing, Retrieval and the Meaning of About. *Journal of the American Society for Information Science*, 28(1):38–43, 1977.

[71] J. Martin. *Design of real-time computer systems.* Series in Automatic Computation. Prentice Hall Inc., Englewood Cliffs, New Jersey, 1967.

[72] E. Mittendorf and P. Schäuble. Document and Passage Retrieval Based on Hidden Markov Models. In *Proceedings of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, Dublin, Ireland, July 1994.

[73] N. Negroponte. *Being Digital.* Hodder and Stoughton, London, 1995.

[74] J.-Y. Nie. Towards a probabilistic modal logic for semantic-based information retrieval. In N. Belkin, P. Ingwersen, and A.M. Pejtersen, editors, *Proceedings of the Fifteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 140–151, Copenhagen, June 1992. ACM Press, New York.

[75] F. van Oostrom. *De Waarde van het Boek.* Amsterdam University Press, Amsterdam, 1994. (In Dutch).

[76] C. van Rijsbergen and K. Sparck Jones. A test for the separation of relevant and non-relevant documents in experimental retrieval collections. *Journal of Documentation*, 29:251–257, 1973.

[77] C.J. van Rijsbergen. *Information Retrieval.* Butterworths, London, United Kingdom, 1975.

[78] C.J. van Rijsbergen. *Information Retrieval.* Butterworth & Co (Publishers) Ltd, London, second edition, 1979.

[79] C.J. van Rijsbergen. A new theoretical framework for information retrieval. In F. Rabiti, editor, *Proceedings of the 9th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 194–200, Pisa, Italia, September 1986. ACM, ACM Press, New York.

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

[81] C.J. van Rijsbergen. *Information Retrieval.* Butterworths, London, United Kingdom, 1990.

[82] C.J. van Rijsbergen and M. Lalmas. An information calculus for information retrieval. *Journal of the American Society of Information Science*, 47(5):385–398, 1996.

[83] S.E. Robertson. The probability ranking principle in IR. *Journal of Documentation*, 33(4):294–304, 1977.

[84] D. Rogers. *The Bodleian Library and its Treasures 1320–1700.* Aidian Ellis, Oxon, 1991.

[85] A. Rosing. An Evaluation of the Hyperindex Machine. Master's thesis, University of Nijmegen, Nijmegen, The Netherlands, 1991.

[86] B. Russell. *History of Western Philosophy.* George Allen & Unwin Ltd, second edition, 1961. Reprinted version available from Routledge, London.

[87] G. Salton. *Automatic Text Processing–The Transformation, Analysis, and Retrieval of Information by Computer.* Addison-Wesley, Reading, Massachusetts, 1989.

[88] G. Salton. The state of retrieval system evaluation. *Information Processing & Management*, 28(4):441–449, 1992.

[89] H. Schouten. SGML∗CASE89–The storage of documents in Data Bases. Technical Report 03-11, TFDL/ECIT, PO 356, 6700 Wageningen, The Netherlands, 1989.

[90] A.F. Smeaton. *Using parsing of natural language as part of document retrieval.* PhD thesis, University College Dublin, Dublin, Ireland, 1988.

[91] A.F. Smeaton. Indexing and Text Representation. In *Proceedings of the European Summer School in Information Retrieval*, pages 171–235, 1990.

[92] K. Sparck Jones. Reflections on TREC. *Information Processing & Management*, 31(3):291–314, 1995.

[93] K. Sparck Jones and C.J. van Rijsbergen. Report on the need for and provision of an "ideal" information retrieval test collection. British Library Research and Development Report 5266, Computer Laboratory, University of Cambridge, 1975.

[94] P.L. van der Spiegel, J.T.W. Driessen, P.D. Bruza, and Th.P. van der Weide. A Transaction Model for Hypertext. In D. Karagiannis, editor, *Proceedings of the Data Base and Expert System Applications Conference (DEXA 91)*, pages 281–286, Berlin, Germany, 1991. Springer-Verlag.

[95] D.F. Stanat and D.F. McAllister. *Discrete mathematics in computer science.* Prentice-Hall, Englewood Cliffs, New Jersey, 1977.

[96] P. Stotts and R. Furuta. Petri-Net-Based Hypertext: Document Structure with Browsing Semantics. *ACM Transactions on Information Systems*, 7(1):3–29, 1989.

[97] J. Tague, A. Salminen, and C. McClellan. A Complete Model for Information Retrieval Systems. In A. Bookstein, Y. Chiaramella, G. Salton, and V.V. Raghavan, editors, *Proceedings of the 14th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 14–20, Chicago, Illinois, October 1991. ACM Press.

[98] I. Tomek and H. Maurer. Helping the user to select a link. *Hypermedia*, 4(2):111–122, June 1992.

[99] F. Tompa. A Data Model for Flexible Hypertext Database Systems. *ACM Transactions on Information Systems*, 7(1):85–100, January 1989.

[100] K.S. Trivedi. *Probability and Statistics with Reliability, Queuing, and Computer Science Applications.* Prentice-Hall, Englewood Cliffs, New Jersey, 1982.

[101] H.R. Turtle and W. Bruce Croft. A comparison of text models. *The Computer Journal*, 35(3):279–290, 1992.

[102] H.R. Turtle and W.B. Croft. Inference Networks for Document Retrieval. In J.L. Vidick, editor, *Proceedings of the 13th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1–24. ACM Press, 1990.

[103] H.R. Turtle and W.B. Croft. A Comparison of Text Retrieval Models. *The Computer Journal*, 35(3):279–298, 1992.

[104] T.F. Verhoef and Th.P. van der Weide. Gerichte Ondersteuning voor Systeemontwikkeling met Bouwstenen. Technical Report CSI-N9501, Computing Science Institute, University of Nijmegen, Nijmegen, The Netherlands, 1995. in dutch, to be published in Informatie.

[105] H. van der Waal. *ICONCLASS an iconographic Classification System.* North-Holland, Amsterdam, The Netherlands, 1985. Completed and edited by L.D.Couprie, E.Tolen and G.Vellenkoop.

[106] Th. P. van der Weide, T.W.C. Huibers, and P. van Bommel. The Incremental Searcher Satisfaction Model for Information Retrieval. Technical Report CSI-R9719, Computing Science Institute, University of Nijmegen, Nijmegen, The Netherlands, November 1997.

[107] S.K.M. Wong and Y.Y. Yao. On modeling information retrieval with probabilistic inference. *ACM Transactions on Information Systems*, 13(1):36–68, January 1995.

[108] Y.Y. Yao. Measuring Retrieval Effectiveness Based on User Preference of Documents. *Journal of the American Society for Information Science*, 46(2):133–145, 1995.

Contents

Dit is de achterkant van het P3 diktaat.