ELECTRONIC **WORKSHOPS IN COMPUTING**

Series edited by Professor C.J. van Rijsbergen

Jonathan Furner, School of Information and Media Studies, and David Harper, School of Computer and Mathematical Studies, The Robert Gordon University, Aberdeen, Scotland. (Eds)

Information Retrieval Research

Proceedings of the 19th Annual BCS-IRSG Colloquium on IR Research, Aberdeen, Scotland, 8-9 April 1997

Paper:

Retrieving Complex Objects with HySpirit

T. Rölleke and N. Fuhr

Published in collaboration with the British Computer Society

BCS

Springer

Retrieving Complex Objects with HySpirit

Thomas Rölleke, Norbert Fuhr
[roelleke,fuhr]@ls6.informatik.uni-dortmund.de

Abstract

Traditional Information Retrieval (IR) considers documents as atomic units. In this paper, we show the retrieval of the components of the documents which satisfy best the information need. This finer granularity eases the browsing of the retrieval result. The approach supports multimedia and networked IR since multimedia documents are composed of other objects and networks combine several collections comprising the documents. We gain a unified view on networks, databases, and multimedia documents by considering them as complex objects — retrieval among a heterogeneous document corpus can be modeled appropriately.

We present a probabilistic retrieval function where the initial estimation of probabilistic parameters is based on the logical structure of documents and the retrieval process is described as probabilistic logical inference. Probabilistic parameters and the retrieval process are represented in probabilistic Datalog programs which are executed by HySpirit — a system for processing probabilistic inference.

1 Introduction

As large and heterogeneous document collections are built because of multimedia and network facilities, there is a growing need of searching for information among these collections. A multimedia document can be characterized by its logical structure: it consists of several components. The atomic components (text blocks, graphics, tables, etc.) vary in their type. Also a collection (database) can be regarded as a complex and retrievable object: it comprises documents. A network can be seen as a complex object containing databases. Retrieving the components of complex objects unifies the retrieval among heterogeneous sets of documents. The document corpus may contain small documents like images and articles and at the same time it may contain larger objects like proceedings and other collections of objects.

Yielding a pointer to a database or an encyclopedia as retrieval result is obviously not very satisfying for the user — she wants to retrieve the components itself which contain the information searched for. The retrieved objects should be as large as necessary and as small as possible for minimizing the costs of gaining the information.

This paper concentrates on the following aspects of the retrieval of complex objects:

- Representing the logical structure of complex objects and using this knowledge for retrieving the best component instead of yielding only a pointer to the whole object.

- Basing the initial estimation of probabilisitic parameters alone on the logical structure and the content description in order to achieve a "fair" probabilistic ranking over a heterogeneous set of retrieved objects.
- Describing the content (knowledge) of complex objects by means of *knowledge propagation*. The atomic components propagate their knowledge to the enclosing objects.
- Localizing probabilistic parameters for the modular injection of uncertain knowlege. The content description and the logical structure form the basic knowledge sources. The uncertainty of the indexing process, relevance feedback data, domain knowledge, user background and query term weighting can be included as additional knowledge sources and probabilistic parameters.

[Chiaramella et al. 96] propose a model paying special attention to the logical structure of multimedia data. They introduce characteristics of propagating knowledge in order to infer on the knowledge of complex documents. [Rölleke & Fuhr 96] present a theory of using a four-valued logic for modeling the retrieval of complex objects. Both address the *propagation* of knowledge in order to conclude from the content of the components of a complex object onto the content of the object itself. Both models are deterministic and they have not yet been implemented.

[Lalmas 97] applies Dempster-Shafer's theory to the model of [Chiaramella et al. 96] in order to achieve a ranking of the retrieved objects. The work points out the generality of Dempster-Shafer's theory when propagating and combining knowledge. Instead, we define a retrieval function based on probability theory. We show that a four-valued truth assignment supports the consistent propagation and combination of uncertain knowledge in a probabilistic framework.

In the probabilistic logical approach the retrieval weight can be interpreted as the probability $P(d \rightarrow q)$ that a document d implies a query q (see [Rijsbergen 86]). For computing this probability, we use a semantics proposed in [Wong & Yao 95]. For describing the retrieval function and for representing uncertain knowledge, we use probabilistic Datalog programs (see [Fuhr 95]). These programs are executed with HySpirit (see [Rölleke & Fuhr 97]), a prototypical system for processing probabilistic inference.

This paper is organized as follows: Section 2 introduces the representation of the logical structure of complex objects. The nodes of the logical structure are regarded as *contexts*. We point out the locality of a logical program with respect to its context. Section 3 proposes an estimation of probabilistic parameters based on the logical structure of objects. The well-known idf (inverse-document-frequency) approach is used for estimating a probability distribution. Section 4 presents the description of the probabilistic parameters and the retrieval process using a probabilistic Datalog program. Section 5 refers to a particular aspect when propagating knowledge: different sources of evidence may lead to inconsistencies. A four-valued truth assignment makes the propagation of uncertain knowledge consistent. Section 6 summarizes the meaning and use of probabilistic parameters.

2 Representing Complex Objects

Complex object may be composed of other objects and this composition gives the logical structure. We refer to an object also as *context*, since an object defines the context where the truth value of propositions is defined. Consider the following example of a logical structure:

```
c1[
    d1[
        s1[
            water
            sailing]
        s2[
            water
            boats]]
    d2[
      water]]
c2[...]
```

On the top level we have two objects: object `c1` and object `c2`. These objects may be different collections. Collection `c1` contains several documents and also documents have a logical structure: document `d1` is structured into two sections.

In each context holds a logical program for representing knowlege. In this example, we use simple logical programs consisting of a set of terms. The set of terms is derived by a process that analyzes the raw data of the documents and assigns terms to the atomic objects. The knowledge of the atomic objects is *propagated* to the enclosing objects. For example, the proposition `water` also holds in `d1` and in `c1`.

The logical structure of objects makes evident that composed objects (supercontexts) are retrieved if a component (subcontext) is retrieved. The content of subcontexts determines the content of supercontexts which corresponds to classical retrieval: a document is retrieved if a section contains a query term. But now, we consider the logical structure and thus the section itself also is retrieved. The task is to define a ranking over the objects that reflects if the whole document or the section is more relevant (more satisfying) according to the given query.

For each supercontext, we can define a proposition space (term space) which contains the propositions occurring in the subcontexts. A query is posed in a certain context, for example, a query can be posed in the context of collection `c1`. The chosen context determines the probabilistic parameters of the retrieval function. Each context has its own term space with a context-dependent probability distribution. This locality of uncertainty values and knowledge supports the context-dependent execution of the retrieval process. The context can include domain knowledge, user background and relevance feedback data for defining the probabilistic parameters. In the next section, we present the exploitation of the logical structure for estimating an initial probability distribution over the term space.

3 IDF-based Probabilistic Term Space

Given the logical structure of collections and documents, we can estimate probabilistic parameters of the retrieval function. In contrast to many other heuristic term-weighting approaches (see e. g. [Salton & Buckley 88]), our approach is based on a probabilistic model which is then mapped onto a probabilistic Datalog program. The aim is to achieve a media independent estimation of probability values. Therefore, we base the initial estimation alone on the representation of the logical structure and the context-dependent propositions and we do not consider the subjective uncertainty values of the analyzing processes. Consider the following common

features of a collection:

$N(c)$	Number of subcontexts of context c
$n(t,c)$	Number of subcontexts of context c where t is true
$I(c)$	Set of propositions of context c

Reconsider the example of section 2. Document `d1` contains two subcontexts, we get $N(d1) = 2$. The term `sailing` occurs in one subcontext, we get $n(\text{sailing}, d1) = 1$. The set of propositions of an atomic object is given by the indexing process. The set of propositions of a composed object is defined as the union over all proposition sets of its subcontexts.

Given these features, we can define a disjoint probability distribution $P(t|c)$ over the term space of a collection. Let $P_{df}(t|c)$ be the probability that term t occurs in a document (subcontext) of collection c. The value of $H_{c,t}$ corresponds to the inverse document frequency (idf) value of term t in collection c. The disjoint probability distribution over the term space is defined as the fraction of the idf value of a term ($H_{c,t}$) and the sum of all idf values (H_c).

$$
\begin{aligned}
P_{df}(t|c) &:= \frac{n(c,t)}{N(c)} \\
H_{c,t} &:= -\log P_{df}(t|c) \\
H_c &:= -\sum_{t \in I(c)} \log P_{df}(t|c) \\
P(t|c) &:= \frac{H_{c,t}}{H_c}
\end{aligned}
$$

As an example, consider the following values, we may get for a collection like `c1` in section 2 (Let $N(c1) = 1000$). For each term, we assume a document frequency $P_{df}(t|c)$. The term `water` occurs often, the term `boats` is the most rare (discriminating) one.

Proposition t	$P_{df}(t\|c1)$	$H_{c1,t}$	$P(t\|c1)$
water	100/1000	2.3	0.17
sailing	10/1000	4.6	0.33
boats	1/1000	6.9	0.5

The higher the document frequency, the lower is the term space probability. The term space probability reflects the discriminating power of a term.

We have defined an estimation of an initial probability distribution over the proposition space. The estimation is based alone on the logical structure — thus, it is applicable for heterogeneous collections containing for example text and non-textual data. The concept of defining probabilistic parameters for each context follows the principle of defining knowledge local to a context. If more knowledge is availabe in a context (like for example relevance feedback data, domain knowledge or user background) this knowledge can be used for improving the estimation of the probabilistic parameters of a context. In the next section, we use probabilistic relations for representing the probabilistic parameters and we describe the retrieval process as a probabilistic logical program.

4 Describing the Retrieval Function $P(d \rightarrow q)$

For describing the retrieval function, we use a probabilistic Datalog program. The representation of complex objects and the term space of a context is described in a probabilistic Datalog program via probabilistic relations. In the following, we use six relations:

- The relation `term(Term,Context)` is used for modeling the term space of a context.
- The relation `cond_term(Term,Doc,Context)` supports the retrieval by conditional probability.
- The relation `d_term(Term,Document)` reflects the occurence of a term in a document — the term *is true* in the context of the document.
- The relation `q_term(Term,Query)` yields the query terms.
- The relation `part_of(SubContext,SuperContext)` represents the logical structure of objects.
- The relation `r(Query,Document,Context)` describes the retrieval function.

A probabilistic Datalog program consists of declarations, facts, and rules for defining the relations. Consider the program in figure 1.

```
#term(av,dk).          (* Disjoint term space *)        (* Declarations *)
#cond_term(av,dk,dk).

0.17 term(water,c1).     (* Term space *)                (* Facts *)
0.33 term(sailing,c1).
0.50 term(boats,c1).

d_term(water,s1).        (* Binary index *)
d_term(sailing,s1).
d_term(water,s2).
d_term(boats,s2).
d_term(water,d2).

part_of(s1,d1).          (* Logical structure *)
part_of(s2,d1).

  (* Knowledge Propagation *)                          (* Rules *)
d_term(T,D) :- d_term(T,D1) & part_of(D1,D).

  (* Retrieval function *)
r(Q,D,C) :- q_term(T,Q) & d_term(T,D) & cond_term(T,D,C).
```

Figure 1: The probabilistic Datalog program

The declarations define the schemas and the disjointness keys of the relations. Tuples having the same disjointness key value are disjoint events. For example, terms within the same context are assumed to be disjoint. Thus, the second attribute of relation `term` — the context — forms the disjointness key. Attributes forming the disjointness key are indicated in the declaration by "dk", the other attributes by "av". Tuples differing in the disjointness key value are assumed to be independent events.

The weights preceeding the facts reflect the probability distribution. A missing weight corresponds to a probability of 1.0. The program contains the term space, the binary index of the atomic objects, and the logical structure.

The first rule propagates the knowledge of subcontexts to supercontexts. If a term is true in a subcontext then it is also true in the supercontext of the subcontext.

The second rule describes the retrieval function: A document D is retrieved according to a query Q in a context C if a document term is a query term. The rule computes the probability function $P(d \rightarrow q)$ according to [Wong & Yao 95] as the probabilistic interpretation of the vector space model. We reconsider the derivation using a conditional probability, i. e. the probability depends on the context (the collection) where the inference is evaluated.

$$\begin{aligned}
P(d \rightarrow q|c) &:= P(q|d \wedge c) \\
&= \sum_t P(q|t \wedge d \wedge c) \cdot P(t|d \wedge c) \\
&\approx \sum_t P(q|t \wedge c) \cdot P(t|d \wedge c) \\
&= \sum_t P(q|t \wedge c) \cdot \frac{P(d \wedge c|t) \cdot P(t)}{P(d \wedge c)} \\
&= \sum_t P(q|t \wedge c) \cdot \frac{P(d \wedge c|t) \cdot P(t)}{P(d \wedge c)} \\
&= \sum_t P(q|t \wedge c) \cdot \frac{P(d|t \wedge c) \cdot P(t|c)}{P(d|c)}
\end{aligned}$$

The probability $P(d \rightarrow q|c)$ of the implication is defined as the conditional probability $P(q|d \wedge c)$. Given a disjoint term space and using the tree-independence-assumption (the query does not depend on the document given the terms for connecting document and query), we achieve the given formula. The rule for describing the retrieval function shall yield a probability according to this probability semantics. The following table shows the semantics of the probability distributions of the relations:

$P(t\|c)$	term(t,c)
$P(t\|c)/P(d\|c)$	cond_term(t,d,c)
$P(d\|t \wedge c)$	d_term(t,d)
$P(q\|t \wedge c)$	q_term(t,q)

The relation `term` represents the probability of the term space. The relation `cond_term` reflects the fraction $P(t|c)/P(d|c)$ which is used in combination with the relation `d_term` for computing the probability $P(t|d)$. For the above example, we obtain the following `cond_term` relation:

0.34 cond_term(water,s1,c1).
0.66 cond_term(sailing,s1,c1).
0.25 cond_term(water,s2,c1).
0.75 cond_term(boats,s2,c1).
0.17 cond_term(water,d1,c1).
0.33 cond_term(sailing,d1,c1).
0.50 cond_term(boats,d1,c1).
1.0 cond_term(water,d2,c1).

The probability values of the tuples of `cond_term` are computed as the fraction of the term space probability and the sum of the term space probabilities of the document terms.

$$P(\text{cond\_term(t,d,c)}) = \frac{P(\text{term(t,c)})}{\sum_{\{t|\text{d\_term(t,d)}\}} P(\text{term(t,c)})}$$

Now, consider query `q1`:

q_term(sailing,q1).
?- r(q1,D,c1).
$(0.66 = 1.0 \cdot 1.0 \cdot 0.66)$ (s1).
$(0.33 = 1.0 \cdot 1.0 \cdot 0.33)$ (d1).

We retrieve section `s1` with a probability of 0.66 and the whole document `d1` with a probability of 0.33. The section has a greater weight since the conditional probability is higher in smaller contexts. It is higher because fewer terms occur in a smaller context like `s1` than in a composed context like `d1`. Now, consider query `q2`:

q_term(sailing,q2).
q_term(boats,q2).
?- r(q2,D,c1).
$(0.83 = 1.0 \cdot 1.0 \cdot 0.83)$ (d1).
$(0.75 = 1.0 \cdot 1.0 \cdot 0.75)$ (s2).
$(0.66 = 1.0 \cdot 1.0 \cdot 0.66)$ (s1).

Document `d1` is ranked higher than the sections, since the query contains more terms. Using the conditional probability brings an advantage to smaller contexts, since there the probability $P(t|d)$ is higher than in larger contexts. Many query terms brings an advantage to larger contexts, since they tend to contain more query terms than the smaller contexts.

We have used the conditional probability of a term in a document for computing the retrieval weight of components of complex objects. The conditional probability is derived from the term space of the collection. The relations for modeling the logical structure and the assignments of terms were deterministic. In the next section, we point out that the indexing process assigns truth values to the propositions of an atomic context. The propagation of this knowledge to supercontexts may lead to inconsistencies. We have to use a four-valued truth assignment in order to define a consistent propagation of uncertain knowledge.

5 The Propagation of "Negative" Knowledge

So far, we have dealt with the propagation of deterministic knowledge from the subcontexts to the supercontexts. We propagated only "positive" knowledge — only the propositions that are true in a subcontext were propagated to the supercontext. Being more precise, it is the truth value of a proposition that is propagated to the supercontext.

The initial assignment of propositions to atomic contexts is done by an indexing process. If this process assigns a proposition, then we define that this proposition has the truth value **true** in the corresponding context. If a proposition is not assigned, then we do not want to conclude per default that the proposition is false in this context. This is not reasonable, because a content description is always incomplete due to its nature. Therefore, we use the truth value **unknown** for propositions which are not explicitly assigned by an indexing process. An "intelligent" indexing process assigns **true** or **false** explicitly for representing the content.

Since one subcontext may give evidence for **true** and another subcontext may give evidence for **false**, the supercontext has to deal with inconsistent knowledge. The fourth truth value **inconsistent** is introduced for combining the knowledge of different subcontexts.

The semantics of probabilistic Datalog uses a closed-world assumption (CWA). For example, the query "`?- !d_term(t,d)`" yields the contrary probability $1 - P(\text{d\_term(t,d)})$. If a term is not assigned, then we get a probability of 1.0. This corresponds to the knowledge that the document `d` is not about `t`. Using a CWA, we would derive this knowledge for all propositions which are not true in a context. As argued above, a CWA is not useful for describing content. Here, we need an open-world assumption (OWA). If a proposition is not assigned, then the truth value is **unknown**. A CWA is only useful when dealing with complete knowledge like the authors of a document.

For modeling the four truth values of term proposition in a context, we add an additional parameter to the relation `d_term`: the first parameter reflects the truth value. We use t for **true**, f for **false**, i for **inconsistent**, and u for **unknown**. For propagating the knowledge, we use two rules: a positive rule for propagating the truth value **true** and a negative rule for propagating the truth value **false**. Only these two truth values are propagated to the supercontext. We assume the propagations to be independent in order to get a well-defined probability of the truth values in the supercontext. The probability of **inconsistent** then can be computed as the product of the probability of being positive and negative. Consider the following example:

```
#term(av,dk).
#d_term(av,dk,dk).
#q_term(dk,dk).
1.0 term(x,c1).      (* Term space *)

0.9 part_of(s1,d1).
0.8 d_term(t,x,s1).
0.7 part_of(s2,d1).
0.6 d_term(t,x,s2).

pos_d_term(T,D) :- d_term(t,T,D).
neg_d_term(T,D) :- d_term(f,T,D).
pos_d_term(T,D) :- pos_d_term(T,D1) & !neg_d_term(T,D1) & part_of(D1,D).
neg_d_term(T,D) :- neg_d_term(T,D1) & !pos_d_term(T,D1) & part_of(D1,D).

r(Q,D,C) :- q_term(T,Q) & pos_d_term(T,D) & !neg_d_term(T,D) & term(T,C).
```

The relation `term` represents the disjoint term space of a collection. The term space does not influence the knowledge propagation and for keeping the example simple, we use only one term. The relation `part_of` reflects the logical structure. For example, the subcontext `s1` is part of `d1`. The probability of 0.9 reflects the effect of the knowledge of `s1` on the knowledge of `d1`. The relation `d_term` models the truth values of a proposition in a context. For example, the proposition `x` is true in context `s1` with a probability of 0.8.

The first and second rule define the positive and negative relations of a proposition. The positive relation reflects the evidence for **true**, the negative relation the evidence for **false**. The probability of a proposition to be positive and *not* negative corresponds to the probability of the truth value **true**. In an analogous way the probability of **false** is defined. The probability of being positive and negative corresponds to the probability of the truth value **inconsistent**.

The third and fourth rule define the propagation of the knowledge: If a term proposition is true in a context `D1` and `D1` is a part of a supercontext `D`, then the supercontext has positive evidence.

The second rule defines the propagation of of false propositions.

The fifth rule represents the retrieval function: We want to retrieve all documents where the query proposition `T` is true. Consider the query `q1`:

```
q_term(x,q1).
?- r(q1,D,c1).
0.8376 (d1).
0.8000 (s1).
0.6000 (s2).
```

When executing query `q1`, we have no evidence for **false**. The retrieval weight of the supercontext `d1` is higher than the retrieval weight of the subcontexts, since in the supercontext the evidence for **true** is combined. In this case, HySpirit computes the probability of the union "evidence from `s1` united with evidence from `s2`": $0.9 \cdot 0.8 + 0.7 \cdot 0.6 - 0.9 \cdot 0.7 \cdot 0.8 \cdot 0.6 = 0.8376$. The first term reflects the evidence from `s1`, the second term the evidence from `s2`, and the third term the evidence coming from both subcontexts.

Now we add evidence for **false** in the subcontexts. This leads to an inconsistency in the supercontext and the probability of being **true** is decreased. Consider the query `q2`:

```
0.2 d_term(f,x,s1).
0.4 d_term(f,x,s2).
q_term(x,q2).
?- r(q2,D,c1).
0.8000 (s1).
0.6000 (s2).
0.5604 (d1).
```

We have evidence that `x` is false in the subcontexts. For explaining the computed probability consider three disjoint sets of possible worlds: (1) the set of worlds where `s1` is a part of `d1` and `s2` is not a part of `d1`; (2) the set of worlds where `s2` is part of and `s1` is not part of; (3) the set of worlds where both subcontexts are part of `d1`. The third set contains worlds where we have evidence for **true** and **false**. These are the inconsistent worlds. The probability of the retrieval result gives the probability of **true** and not **false**. HySpirit does the following computation: $0.9 \cdot 0.3 \cdot 0.8 + 0.1 \cdot 0.7 \cdot 0.6 + 0.9 \cdot 0.7 \cdot 0.8 \cdot 0.6 = 0.5604$. The first term is the sum over the probabilities of the worlds of set (1) where `x` is true, the second term is the sum over set (2) and the third term is the sum over set (3). This computation assumes the tuples of the `part_of` relation and the positive and negative propagation to be independent.

We have demonstrated the usage of probabilistic Datalog programs for dealing with incomplete knowledge using an OWA for propositions describing a context. The indexing process is based on a four-valued truth value assignment and uncertain knowledge is propagated in a consistent probabilistic framework.

6 Summary of Probabilistic Parameters

In this section, we summarize the meaning and use of the probabilistic parameters. In section 3, an initial estimation of one probabilistic parameter is given: the probability distribution over the term space of a context is based on the logical structure. In section 4, the probabil-

ities of the term space are transferred to document terms using the definition of conditional probability. Already this strategy yields a non-monotonic retrieval function with respect to the size of contexts: *smaller or larger* contexts are retrieved according to the query. In section 5, the uncertainty of the indexing process and the quantification of the effect of subcontexts on supercontexts is introduced.

This set of probabilistic parameters influences the retrieval weight. The probabilistic parameters are used for incorporating different knowledge sources into the retrieval process. Summarizing up, they have the following meanings:

Term space: The term space (relation `term`) contains all terms occurring in a context. An initial probability distribution $P(t|c)$ could be improved by using other knowledge sources available in the context. For example, relevance feedback data could be considered for adapting the probability distribution.

Probability transfer: The term space probabilities are transferred to the document terms. One strategy is the conditional probability (relation `cond_term`), other strategies (see [Crestani & Rijsbergen 95]) may take into account more knowledge to direct the probability transfer.

Indexing process: The uncertainty of the indexing process is reflected by the probability $P(d|t)$ that a document covers a term (relation `d_term`). In case of a binary assignment, we have a retrieval function based alone on the term space and the probability transfer. As pointed out in section 5, the uncertainty of the indexing process can be reflected by using probabilities for the truth values of propositions when describing the content of a context.

Logical structure: Knowledge is propagated from subcontexts to supercontexts in order to conclude on the content of supercontexts. The probability distribution over the logical structure (relation `part_of`) quantifies the effect of the content of the subcontext on the content of the supercontext.

Query term weighting: The probability $P(q|t)$ that a query refers to a term (relation `q_term`) reflects the query term weighting. Knowledge sources like relevance feedback, user background and the query formulation itself can be the basis for an estimation of the probability distribution.

The probabilistic parameters are useful for combining different sources of uncertain knowledge with a well-defined probabilistic retrieval function having a semantics of $P(d \rightarrow q)$ as proposed in section 4. In particular, the localization of the parameters and their context-dependency allows for a modular and consistent variation of the retrieval strategy.

7 Conclusions and Outlook

We have presented the retrieval of complex objects. Complex objects can be characterized by their logical structure. This abstraction level leads to a unified view on retrievable objects and to an enrichment of the retrieval result: a retrievable object consists of components and the retrieval result contains also the components of objects that satisfy the information need.

The probabilistic ranking function is based on an interpretation of the probability $P(d \rightarrow q)$ that a document d implies a query q. The logical and probabilistic framework allows the well-defined injection of other knowledge sources like relevance feedback data and domain knowledge.

We have based an initial estimation of a context-dependent probabilistic term space on the logical structure of objects. Further work will be done for estimating the probability $P(d|t)$ that a document covers a term. This probability can be interpreted as the uncertainty of the indexing process in combination with the effect of a subcontext on its supercontext. The incorporation of these additional probability distributions has required the usage of a four-valued truth assignment in order to deal with inconsistent knowledge in a probabilistic consistent framework.

The approach we have presented is currently implemented using HySpirit. HySpirit executes probabilistic Datalog programs and we can vary the fact and rule database in order to investigate the proposed retrieval strategy for complex documents. In [Fuhr 96] a four-valued frontend to probabilistic Datalog is defined which supports the propagation of knowledge in a general manner.

References

Chiaramella, Y.; Mulhem, P.; Fourel, F. (1996). *A Model for Multimedia Information Retrieval*. Technical report, FERMI ESPRIT BRA 8134, University of Glasgow.

Crestani, F.; van Rijsbergen, C. J. (1995). Probability Kinematics in Information Retrieval. In [Fox et al. 95], pages 291–299.

Fox, E.; Ingwersen, P.; Fidel, R. (eds.)(1995). *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York. ACM.

Fuhr, N. (1995). Probabilistic Datalog - a Logic for Powerful Retrieval Methods. In [Fox et al. 95], pages 282–290.

Fuhr, N. (1996). Extending Probabilistic Datalog. In: *Proceedings 2nd Workshop on Information Retrieval, Uncertainty and Logic*. University of Glasgow, Department of Computing Science. http://www.dcs.gla.ac.uk/wirul96/.

Lalmas, M. (1997). *Modelling Structured Documents with Dempster-Shafer's Theory of Evidence*. Technical report, University of Glasgow, Department of Computing Science.

van Rijsbergen, C. J. (1986). A Non-Classical Logic for Information Retrieval. *The Computer Journal 29(6)*, pages 481–485.

Rölleke, T.; Fuhr, N. (1996). Retrieval of Complex Objects Using a Four-Valued Logic. In: *Proceedings of the 19th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 206–214. ACM, New York.

Rölleke, T.; Fuhr, N. (1997). *HySpirit - a Flexible System for Investigating Probabilistic Reasoning in Multimedia Information Retrieval*. Technical report, University of Dortmund, Computer Science. (Submitted).

Salton, G.; Buckley, C. (1988). Term Weighting Approaches in Automatic Text Retrieval. *Information Processing and Management 24(5)*, pages 513–523.

Wong, S.; Yao, Y. (1995). On Modeling Information Retrieval with Probabilistic Inference. *ACM Transactions on Information Systems 13(1)*, pages 38–68.