Intelligent Retrieval of Hypermedia Documents

Mounia Lalmas[1], Thomas Rölleke[1 2], and Norbert Fuhr[3]

[1] Department of Computer Science, Queen Mary, University of London, London E1 4NS, England
[2] HySpirit GmbH, Postfach 30 02 58, 44232 Dortmund, Germany
[3] Informatik VI, University of Dortmund, 44221 Dortmund, Germany

Abstract. Intelligent retrieval of hypermedia documents requires sophisticated document representations and querying facilities that allow for content-based and fact-based querying as well as considering the structure of documents. This paper describes POOL, a Probabilistic Object-Oriented four-valued Logic, which allows a uniform view on hypermedia documents for the purpose of their retrieval: documents, images, authors, dates, etc. are treated as objects and POOL models the content of objects, the facts about objects, and the structure of objects to provide for a relevance-based ranking of hypermedia documents.

Keywords. Hypermedia retrieval, knowledge retrieval, complex objects, content, fact and structure.

1 Introduction

For nearly a decade, an exponentially growing amount of electronic multimedia information has become widely available, e.g. on CD-ROM, on the Internet and in digital libraries. Information retrieval (IR) [29] aims to provide effective and efficient methods of representing, managing, retrieving and displaying such information. Increasingly typical for electronic documents is their *hypermedia* character: (1) they are composed of objects of various media (e.g. text, image); (2) they can be classified into groups (e.g. in an image object peter is classified a sailor); (3) they may be organised into a logical structure (e.g. a chapter and its sections) or a semantic structure (e.g. objects created by the same authors are linked together); and (4) relationships may exist between objects (e.g., spatial and temporal relationships in images and videos, respectively). Hypermedia retrieval requires sophisticated document representations and querying facilities [4] that go beyond the purely text-based (term-based) representation and querying of classical IR.

The need for a conceptual model for representing and retrieving hypermedia documents has been pointed out by many (e.g. [11,18,19]). For instance, [21,6] propose new IR models specifically designed for hypermedia retrieval. All aim at obtaining a flexible conceptual data model that is general enough to capture the needs for any digitally available data in hypermedia documents, as well as supporting end-users in their information seeking activities. One agreed consensus is that such a model must provide for *content-based* querying, *fact-based* querying and consider the *structure* of documents. Content-based querying refers to the content

of objects. For example, a query on videos about sailing boats is a content-based query. Fact-based querying refers to the facts about objects. For example, a query like who was the last captain of the sailing ship Cutty Sark is a fact-based query. Content of objects can be viewed as *knowledge of (contained in) objects*, whereas facts about objects can be viewed as *knowledge about objects* [25]. The structure refers to the way hypermedia documents are organised. For example, a hypermedia repository may consist of documents such as a journal, each structured into several articles. Hypermedia retrieval requires to determine the best entry in the structure. To perform such retrievals, the knowledge of an object must be *augmented* with that of its structurally related objects [25].

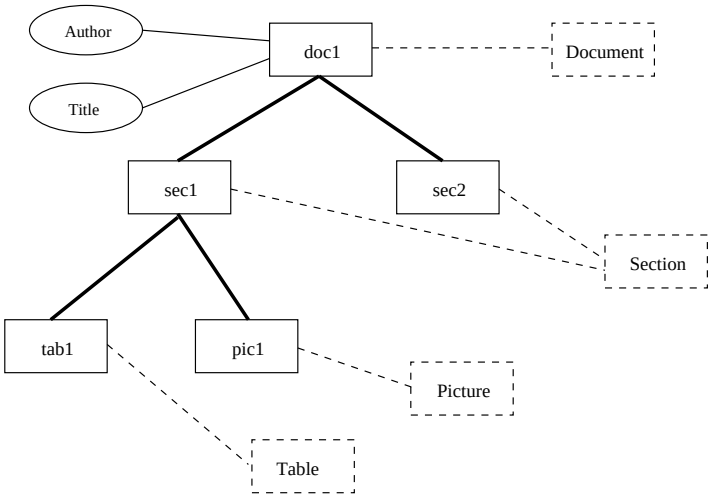
For representing and retrieving knowledge of objects, knowledge about objects such that the structure of objects is taken into account, we developed a model for hypermedia retrieval based on **POOL**, a **Probabilistic Object-Oriented four-valued Logic**. POOL combines elements of probability theory, object-oriented modelling, and four-valued logic into one framework that meets the requirements of intelligent hypermedia retrieval. *Probability* theory captures the intrinsic uncertainty of knowledge and provides an estimation of the probability of relevance for ranking hypermedia documents [2]; *object-oriented* modelling makes the the representation of hypermedia documents adaptable to many application areas and its integration into database system technology possible [1]; and *four-valued logic* captures incomplete and inconsistent knowledge necessary for representing knowledge augmented from contradictory and unknown knowledge sources [26].

The outline of this paper is as follows. Section 2 describes the concept of *complex objects*, which allows us to consider the different views associated with hypermedia documents. Section 3 presents in detail the *representation* of hypermedia documents in POOL. Particular attention is paid to the modelling of content, fact and structure. Section 4 describes how queries are expressed in POOL. Section 5 concentrates on the concept of knowledge augmentation. Section 6 demonstrates the applicability of POOL in a hypermedia environment. We conclude in Section 7.

2 Complex objects

We consider hypermedia documents as objects with a content, facts and a structure. We refer to these objects as *complex objects* since they contain knowledge and they are composed of objects. Figure 1 shows an example of a complex object. The tree reflects the aggregated structure of the complex object. The thick lines mean that object doc1 consists of object sec1 and object sec2. Object sec1 is further structured into object tab1 and object pic1. Object sec1 is referred to as the *supercontext* with respect to tab1 and pic1, whereas tab1 and pic1 are referred to as *subcontexts* with respect to sec1. The dashed lines connect objects to the classes they belong to. For example, object doc1 is a document, objects sec1 and sec2 are sections, object tab1 is a table and object pic1 is a picture. The ellipses indicate attributes that characterise objects. For example, a document has an author and a title.

Fig. 1. A complex object



The abstraction level of complex objects provides a uniform framework for representing hypermedia documents and lead us to base POOL on the object-oriented modelling framework, where objects are characterised by their *feature values*. A feature corresponds to a method, referred to as attribute, that yields a value with respect to the object. This value can be a reference to another object, thus, an attribute can define a relationship between two objects. For example, the author of a document is a feature of the document object, and the author object is the corresponding feature value.

Object-oriented modelling has specific features such as *aggregation* and *classification* that are modelled orthogonally to other *attributes*. In principle, two special attributes *part_of* and *instance_of* could be used to describe the aggregated structure of an object and group instances of object types into classes, respectively. These features are common to all objects, whereas the other attributes are object-dependent. The common approach in object-oriented modelling is to use orthogonal modelling concepts for *aggregation*, *classification*, and *attributes* (see [15,23]), which is also the approach followed in POOL.

POOL combines the object-oriented modelling with the classical information retrieval data model (e.g. set or bag of terms), with a probabilistic representation of uncertainty inherent to IR. The object-oriented nature of POOL was motivated by F-Logic [16], which combines object-oriented principles with logical rules. The semantics of POOL is based on the semantic structure of modal logics [14]. This allows for a context-dependent interpretation of knowledge augmentation, which is necessary for modelling the structure of hypermedia documents. The uncertainty

of knowledge is modelled with a probabilistic-extended semantics [9]. Retrieval functions are implemented as inference processes based on the logical approach to IR [30,31,8,27,7,22], which computes the probability that a document implies a query. Finally, the evaluation of POOL is based on a translation of POOL to the probabilistic relational algebra (PRA) described in [13].

3 Representing complex objects

This section describes the representation of complex objects with POOL. First, we describe the modelling of document views in hypermedia documents (Section 3.1), then inheritance with respect to aggregation, classes and attributes (Section 3.2). We continue with the modelling of semantic knowledge in thesauri and heterogeneous information sources (Section 3.3). We finish with the representation of inconsistent and incomplete knowledge (Section 3.4), and the representation of uncertainty (Section 3.5).

3.1 Document views

In hypermedia document retrieval, we distinguish different views on documents (see [20,12,6]): content, logical and layout views. In POOL, the content of an object is described by true propositions where a proposition is a term, a classification, or an attribute value assignment. Term propositions correspond to the classical IR approach for representing content. Object-oriented modelling concepts, such as classification, attributes and aggregation, are used for representing the logical and layout view. We describe through examples how these three views are modelled in POOL. The full syntax of POOL is given in Appendix A.

We use terms, classifications, and attribute values to represent the *content view* of an object. Consider the following example:

```
doc1[ sailing                     % term
      sailor(peter)               % classification
      giulia.brother(francesco) ] % attribute value
```

The term clause states that the word “sailing” is contained in document doc1. The classification clause expresses that the object peter belongs to the class sailor (peter is a sailor). The attribute clause expresses a relationship between two objects: francesco is the brother of giulia. These propositions are defined within the *context* of object doc1.

We use aggregation, classification, and attribute values for representing the *logical view* (the logical structure) of an object. Consider the following example:

```
doc1[ title1[] sec1[] ] % aggregation
section(sec1)           % classification
sec1.version(1.0)       % attribute value
```

The first clause *opens* the context of object doc1: the structure of doc1 is reflected. Object doc1 consists of two objects: title1 and sec1. When performing hypermedia retrieval, the knowledge in doc1 must be *augmented* with that of objects title1 and sec1. That is, we are not just concerned with determining whether doc1 is relevant to a query, but whether the *augmented context*, denoted doc1(title1,sec1) is relevant to the query. We want to capture the logical structure of complex objects for ranking them according to their estimated relevance. Knowledge augmentation, a key concept of POOL, is discussed in detail in Section 5.

We can use an attribute for representing the logical structure. For example, D.section(S) expresses that S is a section of D. Such a modelling makes explicit that the knowledge about "which sections belong to which objects" is general knowledge about the collection, which can then be queried.

In the above example, object sec1 is a section which has a version attribute whose value is 1.0. Here, we model the classification and attribute knowledge about the object sec1 outside the context of doc1, i. e. the knowledge is available at global level, not in the context of doc1. This approach allows us to distinguish knowledge about the classification and attributes of components of objects from the content knowledge of an object.

POOL makes it possible to distinguish between a component and an attribute value of an object. For example, the title of a document can be both an attribute and a component. It is an attribute in the sense that it characterises a document. It is a component in the sense that it is a part of the document. A similar consideration holds for the author of a document and the author specification in a document. Often, it is not reasonable to consider such a component as a retrievable document part on its own. As attribute, we would store the title as the text string "doc1.title("sailing")". In term-based IR, this difference between attributes and content is often not made explicit. For example, the author specification is treated as content and the information whether the document is written *by* or *about* the author is lost in the representation.

An example to model the *layout view* of a hypermedia document in POOL is the following:

```
page1[figure1[]]
page(page1)
page1.textfraction(0.1)
```

Similarly to the logical view, the object page1 contains the object figure1. The text fraction of object page1 is 0.1.

These first three views, content, logical, and layout view, were identified in [12]. With regard to hypermedia document retrieval, we consider three additional views: the hypermedia view, the spatial view and the temporal view. The *hypermedia view* relates to the fact that hypermedia documents can be connected via links, e.g. a bibliographic reference, a web anchor. The relationship "doc1.link(doc2)" expresses that document doc1 is linked to document doc2. In this representation "link" is an attribute, not an object. Another modelling alternative is to consider links as objects:

```
link1.source(doc1)
link1.destination(doc2)
bibref(link1)
link1[ anchor_from[]  anchor_to[]  text[] ]
```

Here, link1 is an object with a source and a destination, is classified as a bibliographic reference, and consists of two anchors (from and to) and a text entity. A third modelling alternative is to apply the aggregation construct doc1[doc2[]]. However, the aggregation construct means that doc2 is a part of doc1, implying a tree structure between doc1 and doc2, not a graph, the latter being usually the case in the hypermedia view.

The *spatial view* is important in image retrieval, where we want to consider spatial relationships between image parts and connect image parts to "real" world objects. Consider the following example:

```
image1[ p1[]  p2[] ]
p1.includes(p2)
p1.isa(cage)   p2.represents(tweety)
polygon(p1)  p1.point(100,200)  p1.point(200,200)
```

The image image1 includes the parts p1 and p2, where part p1 is a cage and p2 represents tweety. The "include" attribute defines a spatial relationship between the image parts, the "isa" attribute defines a relationship between an image part and a concept (class) of the real world, and the "represents" attribute defines a relationship between an image part and an object of the real world. We can also describe image parts. Here, part p1 is a polygon with several points.

The *temporal view* of documents composed of data streams such as speech and video is modelled with rules (the syntax of rules is given in Appendix A). Consider the following example:

```
video1[ p1[]  p2[] ]
p1.before(p2)
X.before(Z) :- X.before(Y) & Y.before(Z)
```

Here the "before" attribute describe that part p1 is shown before part p2. The rule defines the transitive closure over the attribute "before", specifying all parts that are before all others in a stream video for instance.

3.2 Inheritance

Inheritance is a key concept in object-oriented modelling expressing generalisation and specification. Hypermedia retrieval requires two types of inheritance, which are expressed with rules in POOL. The first type of inheritance is with respect to the object classes and their attributes: objects can be grouped into classes and attribute values. Consider the following example:

```
document(D) :- picture(D)
D.publisher(springer-verlag) :- fqas_paper(D)
```

The first rule expresses that every picture is a document. The second rule specifies the publisher of all “fqas” papers.

The second type of inheritance is with respect to the aggregation: a subcontext can inherit attribute values from its supercontext and vice versa. Consider the following example of a rule which implements a downward propagation of the author attribute:

```
S.author(A) :- D[S[]] & section(S) & D.author(A)
```

It expresses that object A is an author of object S if S is part of D, S is a section, and A is author of D. Upward propagation is expressed via the following rule:

```
D.author(A) :- D[S[]] & section(S) & S.author(A)
```

The rule models that object A is an author of object D if S is part of D, S is a section, and A is author of S. In the two examples, the expression D[S[]] reflects the logical structure of the hypermedia document.

3.3 Semantic knowledge

The intelligent retrieval of hypermedia documents must capture the fact that different concepts (e.g. terms) may be semantically related. This is why thesauri are often used for instance to perform query expansion (a query construction process that aims at bridging the gap between the query terms and the terms used to index documents [2]). Consider the following example:

```
D[transport] :- D[bus & train & plane] & document(D)
D[politician(X)] :- D[president(X)] & document(D)
```

The first rule states that the proposition “transport” is true in a document D if the propositions “bus”, “train”, and “plane” are true in D. The second rule expresses that the object X is classified as “politician” if it is classified as “president”. These rules apply within the boundaries of a context represented by “D[. . .]”. A rule such as “politician(X) :- president(X)” corresponds to the general knowledge that every president is a politician.

A hypermedia document may be composed of document components from heterogeneous information sources. These document components may be in different databases, each with their own schemata, written in different languages, annotated using different ontologies. The retrieval of such documents requires that objects, names, attributes from different sources are mapped onto each other. For instance the rule “D.autor(A) :- D.author(A)” maps English to German attribute names. Such mapping is a crucial issue for the Semantic Web integration, where domain knowledges are represented by different ontologies, which then need to be mapped onto each other to allow for knowledge exchange.

3.4 Inconsistency and incompleteness

POOL provides the possibility of specifying four truth values: *true*, *false*, *inconsistent*, and *unknown* [26]. The truth value *unknown* allows for an independent assignment of true and false values to propositions. Instead of using a closed-world assumption, which would assume a term to be false in a context if it is not assigned, we can use *unknown* as additional truth value to *true* and *false*. This open-world assumption is more reasonable in IR, since the assignment of terms is by nature incomplete and it would be not appropriate to assume false for all terms not assigned explicitly. The truth value *inconsistent* allows for combining contradictory knowledge of subcontexts. Although some propositions may become inconsistent as the result of the augmentation, we can do reasonable inference among the consistent propositions.

3.5 Uncertainty

A major concern in IR is the incorporation of the intrinsic uncertainty of knowledge. POOL addresses two dimensions of uncertainty: (1) uncertainty of the content representation and (2) uncertainty that a supercontext accesses its subcontexts. The uncertainty of the content representation is expressed through probability values assigned to the four truth values *true*, *false*, *inconsistent*, and *unknown* of a proposition. These are represented in a weight list in front of the proposition, where the different values are separated by "/".

```
sailing              % 1.0/0.0/0.0/0.0 sailing
0.9 boats            % 0.9/0.0/0.0/0.1 boats
0.4/0.3 peter        % 0.4/0.3/0.0/0.3 peter
0.4/0.3/0.2 paul     % 0.4/0.3/0.2/0.1 paul
```

where probabilities for *true*, *false*, *inconsistent* and *unknown* sum up to 1. A missing weight is interpreted as the probability of 1.0 for the truth value *true* (e. g. sailing). If probabilities for *true* (e. g. 0.9 boats), *true* and *false* (e. g. 0.4/0.3 peter), or *true* and *false* and *inconsistent* (e. g. 0.4/0.3/0.2 paul) are explicitly defined, then the remainder to 1.0 is assumed to be the probability of *unknown*.

The uncertain access reflects the influence of a subcontext on the knowledge of an augmented context. Consider the following example:

```
d1[ 0.9 s1[ 0.8 sailing ]
    0.7 s2[ 0.6 sailing ] ]
```

The weight 0.9 is interpreted as the probability that s1 determines the knowledge (content) of d1's augmented context d1(s1,s2).

4 Querying complex objects

In the previous section, we described the representation of complex objects. The present section concentrates on the novel query facilities we gain from this repre-

sentation. A detailed description of the querying facilities can be found in [17] and [28], and the full syntax of queries is given in Appendix A.

4.1 Content-oriented Querying

Content-oriented querying refers to the *content* of documents. For example, a query on videos about sailing boats is a content-oriented query. We distinguish three types of content-oriented queries: terms, classifications, and attributes, each of which are illustrated in the example below.

```
?- D[sailing]
?- D[sailor(peter)]
?- D[peter.friend(mary)]
```

The first query searches for all documents D which are about sailing, i. e. the term proposition "sailing" is true in document D. The second query searches for all documents D, in which peter is a sailor, i. e. the classification proposition "sailor(peter)" is true in document D. The third query searches for all documents D, in which peter is a friend of mary, i. e. the attribute proposition "peter.friend(mary)" is true in document D. In contrast to classical purely term-oriented approaches for representing content, we can ask for classification and attribute knowledge on objects.

Content-oriented retrieval also takes takes into account the logical structure of complex objects. Consider the following object d that consists of two sections s1 and s2:

```
d[ s1[sailing boats]
   s2[ocean boats] ]
```

The query "?-D[sailing]" retrieves both s1 and d. Context s1 is retrieved, since sailing describes the content of s1, whereas context d is retrieved, since the augmented content knowledge of d contains the "sailing" proposition. Consider now the query "?- D[ocean & sailing]". The conjunction "ocean & sailing" is true in the augmented context d(s1,s2). This is because "ocean" is true in d(s1,s2), since it is true in s2, and "sailing" is true in d(s1,s2), since it is true in s1. No leaf context on its own satisfies the query, only the augmented context d(s1,s2) satisfies the query.

When the representation is uncertain, the retrieved objects become ranked according to how well they satisfy a query. Consider the following example:

```
d[ 0.5 s1[ 0.8 sailing ]
   0.5 s2[ 0.6 sailing ] ]
```

The query "?-D[sailing]" retrieves three contexts:

```
0.8  s1
0.6  s2
0.58 d(s1,s2)
```

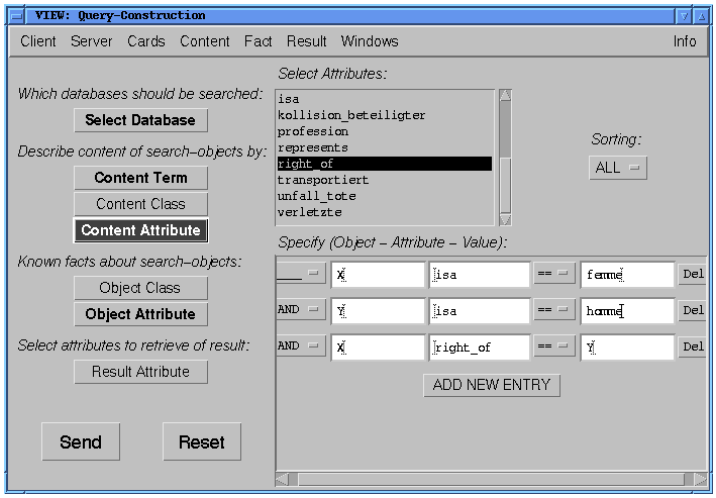
The subcontexts are retrieved with the probabilities of sailing. The augmented context d(s1,s2) is retrieved with a probability of 0.58 which is the summation of three probabilities: the probability that sailing is true if both subcontexts are accessed ($0.5 \cdot 0.5 \cdot (0.8 \cdot 0.6 + 0.8 \cdot 0.4 + 0.2 \cdot 0.4) = 0.23$) plus the probability that sailing is true if only s1 is accessed ($0.5 \cdot 0.8 \cdot 0.5 = 0.2$) plus the probability that sailing is true if only s2 is accessed ($0.5 \cdot 0.6 \cdot 0.5 = 0.15$) (see Section 5 and [25] for the semantics of the probability computation).

With hypermedia data, a special form of content-based querying requires attention: queries that refer to *relationships* between objects. For instance, in image retrieval, particular attention must be paid to spatial relationships. To formulate a query on spatial relationships, predicate logic is used. Consider the following formulation of the query:

```
?- D[X.isa(femme) & Y.isa(homme) & X.right_of(Y)]
```

The formal notation expresses that the user is seeking all documents D in which an object X exists that is a “femme” (French for woman), and in which an object Y exists that is an “homme” (French for man), and in which X is right of Y. Formulating the above query expression directly is a task for expert users. Figure 2 corresponds to a graphical user interface that supports “semi-expert” users in formulating such a query. We can display the possible attributes (all possible attributes are listed in the window above the logical formulation) and we automatically add a corresponding entry into the specification of the query.

Fig. 2. Content-based query with relationships [17]



4.2 Factual Querying

Fact-based querying refers to the *facts* about documents. For example, a query like who was the last captain of the sailing ship Cutty Sark is a fact-based query. A factual query searches for all documents that belong to a specific class or have specific attribute values. In the following example, the first query searches for all document images, whereas the second query searches for all documents of author peter.

```
?- image(D)
?- D.author(peter)
```

In content-oriented querying, we are looking for formulae true in documents; in factual querying, we are looking for formulae true in the database (i.e. over the collection). In factual querying, the document itself is used as a parameter of a predicate (i. e. a class or an attribute predicate). We query among the knowledge about the collection.

We can directly ask for facts known in documents instead of performing first a content-oriented search and extracting the facts afterwards. Consider the following program:

```
doc1[sailor(peter)]
?- sailor(X)
```

The database contains object doc1. The factual query is searching for all sailors. The answer "peter" will be returned if the knowledge about the collection is augmented by the knowledge contained in doc1. The knowledge augmentation is not only with respect to terms, but also with respect to classification and attributes.

4.3 Vague Querying

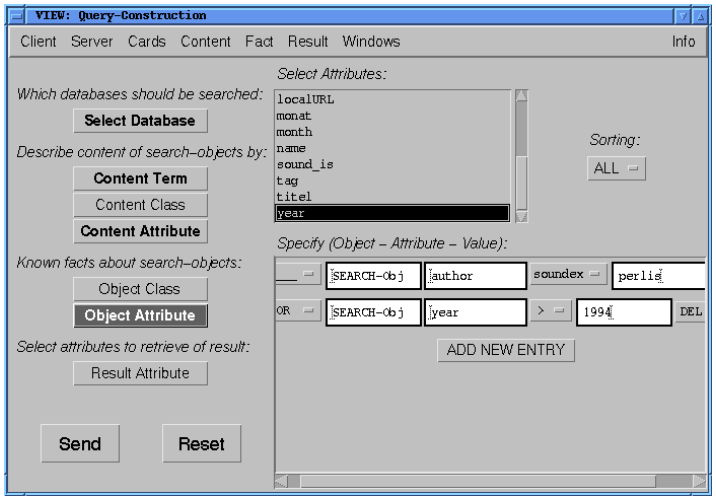
It is important to allow for vague attribute values for fact-based querying. A typical example is the specification of author names. Often, it is necessary to consider different spellings of a name. For example, "maier", "mayer", "meier" are three possible spelling of the same German name. Assume a predicate "soundex_code" that associates a code with a name (m600 is the soundex_code for the above three names). We can define a predicate soundex as follows:

```
X.soundex(Y) :- X.soundex_code(Code) & Y.soundex_code(Code)
```

The pairs (X,Y) are objects that are associated with the same soundex code. We call "soundex_code" a vague predicate since it selects a group of objects for one attribute value. We use the predicate soundex for formulating the vague query that searches for all documents of authors that have a name sounding like maier.

```
?- D.author(X) & X.soundex(maier)
```

Fig. 3. Fact query and vague predicate [17]



Vague predicates are predicates that select a set of objects. This selection is an uncertain process (see [25] for how this uncertainty is represented). Figure 3 shows an example of a fact-based query and a vague predicate. Relevant documents should be those written from 1994 by authors whose name sounds like “perlis”.

Factual queries and vague predicates play a crucial role in hypermedia retrieval. Non-text objects such as images and video have attribute values (also called features) such as colour, contour, size, and length, upon which most “content-based” image retrieval approaches are based (e.g., the QBIC system [10]). We can deduce or extend the content description of images from their attribute values. For example, a typical news clip lasts about 30 seconds, and a typical landscape picture has a certain distribution of colours, contours, and textures. With vague predicates, we can represent similar objects. For example, p1.colour(p2) means that two pictures are similar in colour distribution.

5 Knowledge Augmentation

Knowledge augmentation is a key concept of POOL. It allows taking into account the logical structure of hypermedia documents for estimating the relevance of document parts. Let a document d1 be composed of two objects, s1 and s2. The relevance of s1 and s2 is estimated on their respective knowledge (i.e. term, classification and attribute propositions true within their context). The relevance of the supercontext d1 is based on the knowledge in d1 itself and the knowledge contained in its subcontexts s1 and s2, i.e. the knowledge in the augmented context d1(s1,s2). The knowl-

edge of an augmented context is defined as the combination of the knowledge of the supercontext and its subcontexts.

For a appropriate combination regarding inconsistent and incomplete knowledge, POOL has four truth values *true*, *false*, *inconsistent* and *unknown* (see Section 3.4). Consider the following example:

```
d1[ s1[ peter.friend(mary) ]
    s2[ not peter.friend(mary)   sailor(peter) ] ]
```

Subcontext s1 knows that peter is a friend of mary; subcontext s2 states the opposite. The example makes evident that the combination of the subcontexts s1 and s2 leads to inconsistent knowledge regarding the proposition "peter.friend(mary)", since we have evidence for true from s1 and false from s2. In the augmented context d1(s1,s2), "peter.friend(mary)" is inconsistent. In s1, no truth value for the proposition "sailor(peter)" is specified; the knowledge is incomplete. In the augmented context d1(s1,s2), the proposition "sailor(peter)" is true since the proposition is unknown in s1 and true in s2.

For uncertain content as well as uncertain access, consider the following example:

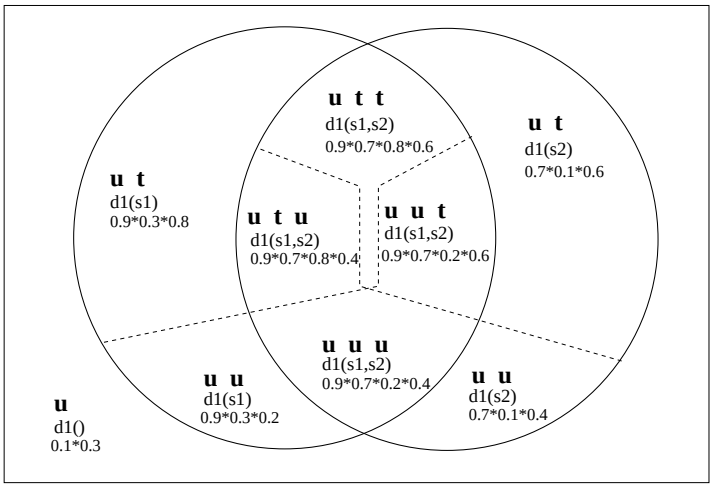
```
d1[ 0.9 s1[ 0.8 sailing ]
    0.7 s2[ 0.6 sailing ] ]
```

We want to derive the probability that the proposition "sailing" is true within the augmented context d1(s1,s2). We consider an augmented context as a set of possible worlds [3], where a world represents a possible aggregation of the augmented context.

Figure 4 shows a graphical illustration of the augmentation of d1. The two circles represent the subcontexts s1 and s2; they divide the context d1 into four parts: a part where both subcontexts are considered (the intersection), a part where only s1 is considered (left part of the left circle), a part where only s2 is considered (right part of the right circle), and a part where no subcontext is considered (outside the circles). Each part can be seen as a set of possible worlds where a world represents the supercontext augmented with either both subcontexts, one, or no subcontext. The two dashed lines divide each circle in the parts corresponding to the truth values of the propositions. The parts are sets of possible worlds, in which the corresponding truth value of a proposition holds. Above the dashed lines, we find the worlds where sailing is true, and below the dashed lines we find the worlds where sailing is unknown. The truth values are shown above the context names.

The probability of a truth value of a proposition is defined as the sum over the probabilities of the possible worlds, where the proposition has the truth value (see [24]). For example, consider nine possible worlds $w_1, \ldots, w_9$ in Table 1, one world for each part depicted in Figure 4. The truth value assignment follows the open-world assumption; i.e., in Figure 4 the proposition "sailing" is *unknown* in the worlds where it is not *true*. In the worlds $w_1, \ldots, w_4$ both s1 and s2 are considered; the worlds of both subcontexts are reached by the supercontext d. In worlds w_5 and

Fig. 4. Knowledge augmentation: context d1(s1,s2) [25]



w_6, only s1 is considered. The same holds for s2 with respect to worlds w_7 and w_8. In world w_9, no subcontext is considered (denoted d1()). From world w_1, d1 reaches the worlds in which s1 and s2 assign the truth value *true* to the proposition sailing. From world w_2, d1 reaches the worlds in which s1 assigns *true* and s2 assigns *unknown*. Similarly for w_3 and w_4. From worlds w_5 and w_7, d reaches worlds where the worlds where sailing is *true*. Similarly for w_6 and w_8 and *unknown*. From world w_9, no world of a subcontext is accessed. In each world, d1 contributes *unknown* to the augmented truth value. Table 1 shows the truth values of the supercontexts (we omit the truth value *unknown* associated with d1). By considering truth values as the sets *true*= $\{t\}$, *false*= $\{f\}$, *inconsistent*= $\{t, f\}$ and *unknown*= $\{\}$, the union of the truth values defines the truth value within the augmented context d1(s1,s2).

We sum over the probabilities of the worlds, where sailing has a specific truth value. Thus, we obtain:

```
d1(s1,s2)[ 0.8376/0/0/0.1624 sailing ]
```

Within the augmented context d1(s1,s2), sailing is true with a probability of 0.8376, and sailing is unknown with a probability of 0.1624. The probability distribution over the worlds used in this example is based on two independence assumptions: subcontexts affect independently the knowledge of an augmented context; and truth values of propositions in different contexts are independent events [25].

Consider now the following example:

```
d1[ 0.9 s1[ 0.8/0.2 sailing ]
    0.7 s2[ 0.6/0.4 sailing ] ]
```

Table 1. Knowledge augmentation: open-world assumption

World	Probability	Truth value of sailing
w_1	$0.9 \cdot 0.7 \cdot 0.8 \cdot 0.6$	*true* = *true* ∪ *true*
w_2	$0.9 \cdot 0.7 \cdot 0.8 \cdot 0.4$	*true* = *true* ∪ *unknown*
w_3	$0.9 \cdot 0.7 \cdot 0.2 \cdot 0.6$	*true* = *unknown* ∪ *true*
w_4	$0.9 \cdot 0.7 \cdot 0.2 \cdot 0.4$	*unknown*=*unknown* ∪*unknown*
w_5	$0.9 \cdot 0.3 \cdot 0.8$	*true*
w_6	$0.9 \cdot 0.3 \cdot 0.2$	*unknown*
w_7	$0.7 \cdot 0.1 \cdot 0.6$	*true*
w_8	$0.7 \cdot 0.1 \cdot 0.4$	*unknown*
w_9	$0.1 \cdot 0.3$	*unknown*

We have evidence for true and false, which leads to an inconsistency in d1(s1,s2). Consider the set of possible worlds of Table 2.

Table 2. Knowledge augmentation: negative knowledge

World	Probability	Truth value of sailing
w_1	$0.9 \cdot 0.7 \cdot 0.8 \cdot 0.6$	*true* = *true* ∪ *true*
w_2	$0.9 \cdot 0.7 \cdot 0.8 \cdot 0.4$	*inconsistent* = *true* ∪ *false*
w_3	$0.9 \cdot 0.7 \cdot 0.2 \cdot 0.6$	*inconsistent* = *false* ∪ *true*
w_4	$0.9 \cdot 0.7 \cdot 0.2 \cdot 0.4$	*false* = *false* ∪ *false*
w_5	$0.9 \cdot 0.3 \cdot 0.8$	*true*
w_6	$0.9 \cdot 0.3 \cdot 0.2$	*false*
w_7	$0.7 \cdot 0.1 \cdot 0.6$	*true*
w_8	$0.7 \cdot 0.1 \cdot 0.4$	*false*
w_9	$0.1 \cdot 0.3$	*unknown*

The worlds w_2 and w_3 assign now the truth value *inconsistent* to the proposition sailing. We therefore obtain:

```
d1(s1,s2)[ 0.5604/0.1324/0.2772/0.03 sailing ]
```

Knowledge augmentation is also used to determine the knowledge associated with *virtual contexts*. Virtual contexts are aggregated contexts that contain only selected subcontexts. They allow for searching the smallest contexts where a query formula holds. Consider the following example:

```
d1[ s1[ 1/0 sailing ]
    s2[ 0/1 sailing ]
    s3[ 1/0 boats ] ]
```

Supercontext d1 consists of the three subcontexts s1, s2, and s3. The virtual augmented context d1(s1,s3) contains only the subcontexts s1 and s3. It is "virtual" because in the original structure, d1 is composed of s1, s2 and s3. For the virtual context d1(s1,s3), we obtain "d1(s1,s3)[sailing & boats]". The conjunction "sailing & boats" is true in the virtual context d1(s1,s3). In the virtual context d1(s1,s3), "sailing" is true, because s1 gives evidence for true and s3 gives evidence for unknown. In the real context d1(s1,s2,s3), "sailing" is inconsistent, because s1 gives evidence for true and s2 gives evidence for false. Thus, a query for "sailing" would not retrieve d1(s1,s2,s3), but could retrieve d1(s1,s3).

A virtual context is an augmented context. The probabilities of facts in an augmented context such as d1(s1,s2) consider four possible aggregations of d1(s1,s2): d1(s1 and s2), d1(s1 and not s2), d1(s2 and not s1), d1(). If the augmented context is virtual, a selection of some subcontexts is first performed, then the concept of knowledge augmentation is applied.

6 A hypermedia Collection

To demonstrate the use POOL in a hypermedia environment, we built a hypermedia collection consisting of text, images, and videos. This demonstration shows the uniform manipulation of the representations of documents that vary in their media type, structure and available indexing information. The prototype demonstration system works with the following collections:

IRIS: A collection of 1200 colour pictures in TIFF format. This collection has been built for developing automatic semantical indexing methods (see [5]). The indexing algorithm is trained for landscape pictures and the collection contains typical landscape as well as other pictures.

PARIS: A collection of 650 black and white pictures in GIF format. The purpose of this collection is the evaluation of retrieval models that enable a richer query formulation than just keywords. The images of the collection show motives of Paris in the early years of the 20th century. In particular, the logical structure of the images and relationships between objects are represented in the document index, which has been created manually.

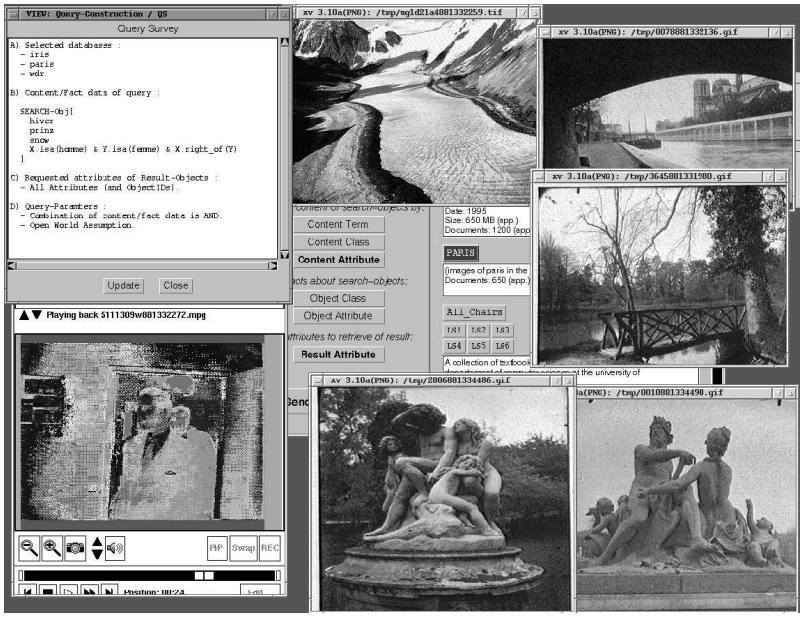
WDR: A collection of 200 Video news clips of the German TV broadcast station WDR. This collection has been set up for working with a rich semantical index where the events and persons occurring in the news are identified. Events and persons are classified and attribute values are specified.

We translated the indexing information into POOL. Figure 5 shows some retrieved documents for the following query:

```
retrieve(D) :- D[hiver]
retrieve(D) :- D[prinz]
retrieve(D) :- D[snow]
retrieve(D) :- D[X.right_of(Y) & femme(X) & homme(Y)]
```

We search for all documents that are about “hiver” (French for winter), or are about a prince (in German!), or about snow (in English), or show a woman right of a man (partly French!).

Fig. 5. The hypermedia result [17]



The window in the upper left corner summarises the query formulation of the content description, the factual description, and global query parameters such as open-world or closed-world assumption. In the query formulation, the context name “Search-Obj” reflects the objects searched for. Within the context of the object, each line contains a conjunction that has to be true in the retrieved objects. In the lower left corner, a video with Prince Charles is played. We see winter pictures, pictures of snow, and pictures with women right of men. The centre of the pictures shows the query construction window.

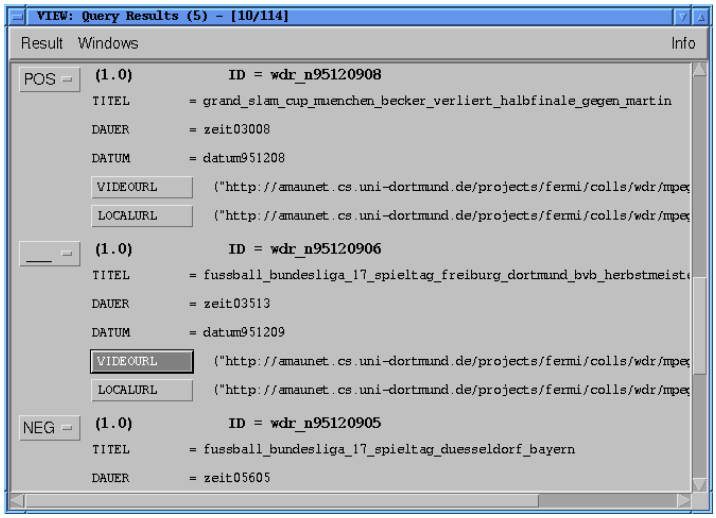
The query formulation contains English, French, and German predicates. The IRIS collection is indexed in English, Paris in French, and WDR in German. We incorporate rules such as the following for mapping the languages:

```
D[prince] :- D[prinz]
D[woman(X)] :- D[femme(X)]
```

The rules assure that we retrieve documents of each collection independently of the language used for the predicates in the actual query formulation.

The WDR collection contains structured objects. Up to 11 video clips form the augmented context of a day. The clips are indexed separately. Querying for a news event yields video clips and days as objects in the retrieval result. If a news is reported in several clips, then the day object that contain the clips can be ranked higher in the result list than the single clips. Starting from a day object, the user can browse the clips broadcasted during that day. Figure 6 shows the retrieved clips. The result attributes “TITEL”, “DAUER”, etc. are listed for each video.

Fig. 6. Video result [28]



The application of POOL with the described collection show the expressiveness of POOL for representing and retrieving hypermedia documents.

7 Conclusion

This paper describes a model for the representation and retrieval of hyppermedia documents. The model is based on POOL, a Probabilsistic Object-oriented Four-valued Logic. POOL supports the representation of hypermedia objects with respect to their content, aggregation, classification, attributes, thus allowing for content-based querying and fact-based querying, as well as taking into account the structure of hypermedia objects for providing a relevance-based ranking of hypermedia documents.

Acknowledgement

Thanks to Gabriella Kazai for her insightful comments on earlier versions of this paper.

References

1. S. Abiteboul, S. Cluet, V. Christophides, T. Milo, G. Moerkotte, and J. Simeon. Querying documents in object databases. *International Journal on Digital Libraries*, 1:1–9, 1997.
2. R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, 1999.
3. B.F. Chellas. *Modal Logic*. Cambridge University Press, 1980.
4. Y. Chiaramella. Browsing and querying: two complementary approaches for multimedia information retrieval. In *Proceedings Hypermedia - Information Retrieval - Multimedia*, Dortmund, Germany, 1997.
5. Y Chiaramella and M Mechkour. Indexing an image test collection. Technical Report FERMI Deliverable 10, ESPRIT BRA 8134, University of Dortmund, Informatik VI, 1997.
6. Y. Chiaramella, P. Mulhem, and F. Fourel. A model for multimedia information retrieval. Technical Report Fermi ESPRIT BRA 8134, University of Glasgow, 1996.
7. F. Crestani, M. Lalmas, and C.J. van Rijsbergen, editors. *Information Retrieval: Uncertainty and Logics - Advanced Models for the Representation and Retrieval of Information*. Kluwer Academic Press, 1998.
8. F. Crestani and C.J. van Rijsbergen. Probability kinematics in information retrieval. In *Proceedings of ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 291–299, 1995.
9. R. Falgin, J. Harpen, J. Moses, and M. Vardi. *Reasoning about Knowledge*. MIT Press, Cambridge, Massachusetts, 1995.
10. M. Flicker, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker. Query by image and video content. In *Intelligent Multimedia Information Retreival*, pages 23–31. AIII Press, 1997.
11. P. Fraternali and P. Paolini. A conceptual model and a tool environment for developing more scalable, dynamic, and customizable web applications. In *Proceedings of the 6th International Conference on Extending Database Technology (EDBT)*, pages 421–435, 1998.
12. N. Fuhr. Logical and conceptual models for the integration of information retrieval. In *East/West Database Workshop, Klaenfurt*, pages 206–218, 1994.
13. N. Fuhr and T. Roelleke. A probabilistic relational algebra for the integration of information retrieval and database systems. *ACM Transactions on Information Systems*, 14(1):32–66, 1997.
14. J. Halpern and Y. Moses. A guide to completeness and complexity for modal logics of knowledge and belief. *Artificial Intelligence*, 54:319–379, 1992.
15. A. Heuer. *Objektorientierte Databanken*. Addison-Wesley, New York, 92.
16. M. Kifer, G. Lausen, and J. Wu. Logical foundations of object-oriented and frame-based languages. *Journal of the Association for Computing Machinery*, 42(4):741–843, 1995.
17. M. Lalmas, T. Roelleke, F. Turra, and N. Fuhr. Concepts for a graphical user interface for hypermedia retrieval. In *proceedings of Flexible Question-Answering Systems (FQAS 2000)*, 2000.

18. D. Levy and C. Marshall. Going digital: A look at assumptions underlying digital libraries. *Communications of the ACM*, 38(4):77–84, 1995.
19. E. Megalou, T. Hadzilacos, and N. Mamoulis. Conceptual title abstractions: Modeling and querying very large interactive multimedia repositories. *Multimedia Modeling Towards the Information Superhighway*, pages 323–338, 1996.
20. C. Meghini, F. Rabitti, and C. Thanos. Conceptual modelling of multimedia documents. *IEEE Computer*, 24(10):23–30, 1991.
21. C. Meghini, F. Sebastiani, U. Straccia, and C. Thanos. A model of information retrieval based on a terminological logic. In *Proceedings of the Sixteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 298–308, 1993.
22. C. Meghini and U. Straccia. A relevance terminological logic for information retrieval. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 197–205, 1996.
23. B. Meyer. *Object-Oriented Software Construction*. Prentice Hall International (UK) Ltd, Hemel Hempstead, Hertfordshire, 1988.
24. N.J. Nilsson. Probabilistic logic. *Artificial Intelligence*, 28:71–78, 1986.
25. T. Roelleke. *POOL: Probabilistic Object-Oriented Logical Representation and Retrieval of Complex Objects - A Model for Hypermedia Retrieva*. PhD thesis, University of Dortmund, Germany, 1999.
26. T. Roelleke and N. Fuhr. Retrieval of complex objects using a four-valued logic. In *Proceedings of ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 206–214, Zurich, Switzerland, 1996.
27. F. Sebastiani. A probabilsitic terminological logic for modelling information retrieval. In *Proceedings of 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 122–131, 1994.
28. F. Turra. Interktive logische anfrageformulierung fur hypermedia-retrieval. Master's thesis, University of Dortmund, 1998.
29. C. J. van Rijsbergen. *Information Retrieval*. Butterworths, London, 2 edition, 1979.
30. C.J. van Rijsbergen. A non-classical logic for information retrieval. *The Computer Journal*, 29(6):481–485, 1986.
31. C.J. van Rijsbergen. Probabilistic retrieval revisited. *The Computer Journal*, 35(3):291–298, 1992.

A Syntax of POOL

```
program          ::= clause | clause ';' program
clause           ::= fact | context | query | rule
fact             ::= proposition | NOT proposition | prob-list proposition
proposition      ::= term | classification | relationship
term             ::= NAME | STRING
classification   ::= NAME '(' constant ')'
relationship     ::= NAME '.' NAME '(' constant ')'
constant         ::= NAME | STRING | NUMBER
prob-list        ::= prob | prob '/' prob
prob             ::= NUMBER
context          ::= NAME '[' program ']' | NAME '[' ']' |
                     prob NAME '[' program ']' | prob NAME '[' ']'
query            ::= '?-' subgoal-list
subgoal-list     ::= subgoal | subgoal '&' subgoal-list
subgoal          ::= fact-subgoal | context-subgoal
fact-subgoal     ::= atom | NOT atom
atom             ::= term |
                     NAME '(' parameter ')' |
                     NAME '.' NAME '(' parameter ')' |
                     VAR '.' NAME '(' parameter ')'
                     COND '(' parameter ',' parameter ')'
context-subgoal  ::= VAR '[' subgoal-list ']' | VAR '[' ']'
parameter        ::= VAR | constant
rule             ::= goal ':-' subgoal-list
goal             ::= fact-goal | context-goal
fact-goal        ::= atom | NOT atom
context-goal     ::= VAR '[' fact-goal ']'
NAME             ::= [a-z][a-zA-Z0-9_]*
STRING           ::= '"'.*'"'
NUMBER           ::= [0-9]+(\.[0-9]*)?
VAR              ::= [A-Z][a-zA-Z0-9_]*
NOT              ::= 'not' | 'NOT'
COND             ::= '=' | '==' | '!=' | '<>' | '>=' | '<=' | '>' | '<' | '=~' | '!~'
```