Retrieval of Complex Objects Using a Four-Valued Logic

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

Abstract

The aggregated structure of documents plays a key role in full-text, multimedia, and network Information Retrieval (IR). Considering aggregation provides new querying facilities and improves retrieval effectiveness. We present a knowledge representation for IR purposes which pays special attention to this aggregated structure of objects. In addition, further features of objects can be described. Thus, the structure of full-text documents, the heterogeneity and the spatial and temporal relationships of objects typical for multimedia IR, and meta information for network IR are representable within one integrated framework.

The model we propose allows for querying on the content of documents (objects) as well as on other features. The query result may contain objects having different types. Instead of retrieving only whole documents, the retrieval process determines the least aggregated entities that imply the query.

1 Motivation and Background

New IR applications like full-text, multimedia, and network IR require a more sophisticated knowledge representation than the pure set of terms used in classical IR. What should the properties of a more sophisticated data model be?

We need an integrated framework which allows a unified view of text documents, multimedia documents, and databases by considering all of them as retrievable objects which have some aggregated structure and some local knowledge about other objects.

Classical IR only considers documents as atomic units, i. e. some kind of a black box. This black box thinking is very useful regarding the representation of the content, because this content representation should be abstract, i. e. independent of the actual media.

The model we present provides a unified content view plus the possibility to express other (factual) knowledge, i. e. features of objects. In addition, we also represent and exploit the aggregated structure of the documents. Thus, we get new query facilities: Querying for documents and facts can be combined. And, instead of retrieving only a reference to the whole document, the least aggregated entity that implies the query is identified. Due to the finer granularity of the retrieval result connected with the powerful querying language, we reduce the costs of obtaining the requested information and information retrieval becomes more effective.

Modelling the structure: In our approach each object can have an aggregated structure. Full-text documents may be structured in chapters, sections, and paragraphs. Multimedia documents comprise different objects with varying media types. In network IR several databases form a knowledge base and each database knows the local documents and possibly again other knowlege bases.

[Brown 89] pointed out that "maintaining the underlying logical structure of a document can have many advantages for document processing", since the ideas of the author are reflected by the structure and the system can provide intelligent help for navigating. Standards like ODA and SGML are widely used and so IR can take advantage of the structured information available. [Fuhr 95a] motivates the differentiation between the content structure and the logical structure for making IR more effective. [Macleod 90], [Colby et al. 94], and [Christophides et al. 94] propose retrieval models which take into account the structure of documents. The latter presents the interesting new facility of querying for paths leading to the relevant part of a document.

Modelling features: Besides modelling the structure, we need a means for representing knowledge about objects. Since we want to deal with objects of different types (e. g. documents, databases), we have to model propositions about classification for grouping the objects. In addition, we need a means to model relationships between objects (e. g. authorship of documents).

[Pfeifer et al. 95] demonstrates how classical content retrieval can be combined with searching for document features. [Fuhr & Rölleke 96] and [Fuhr 95b] present approaches to integrate such factual knowledge into a data model appropriate for indexing a collection. [Meghini 95] underlines the need of modelling relationships between objects for multimedia IR by distinguishing between the syntactical and the semantical level. (For example, the representation of "a picture showing Giulia playing with Francesco" requires modelling the semantical relationship between persons.) Classification and generalization are introduced as appropriate methods for modelling the semantical relations of objects. [Fuhr 96] also stresses the usage of object-oriented concepts for improving IR systems. From generalization (inheritance, respectively), we gain a unified view onto retrievable objects which is especially important in network IR.

Concluding, a sophisticated knowledge representation should incorporate the principles of object-oriented design, namely aggregation, classification, and generalization.

Following [Rijsbergen 86], we want the retrieval process to be expressed as proving the inference $d \rightarrow q$ between a document d and a query q. Therefore, we combine logical means and object-oriented modelling in our data model similar to [Kifer et al. 95].

In the following, we first introduce the concepts (sections 2, 3, 4) and the semantics (section 5) of the data model. For defining the semantics, we use Kripke structures as described in [Halpern & Moses 92] and we add a four-valued truth assignment. Section 6 presents the consideration of the intrinsic uncertainty of knowledge for achieving a ranking of the retrieved objects. Section 7 sketches the application of the model for multimedia and network retrieval.

2 Basic Components of the Data Model

Classical IR systems use a set of terms for representing the knowledge necessary for retrieving the documents which imply a query. This pure set of terms is a one-dimensional knowledge representation: it only aims at modelling the knowledge about the content of the documents. The advantages of the model are its simplicity, the possibility to add term weighting for getting a relevance measure, and its efficient processing when retrieving documents.

The model we propose aims at having the same advantages. In addition it allows for modelling knowledge more complex in structure. As argued in section 1, we have to represent the aggregated structure and features of objects. Figure 1 depicts the basic components of the data model.

Aggregation: The first clause expresses the aggregation relation. The document `d1` consists of two words (`hello`, `world`) and it has two subsections (`s11`, `s12`) which also consist of words. This aggregation expression reflects the logical structure of objects. In the case of full-text documents, the aggregation clause corresponds to a nested set of terms.

```
% Aggregation
d1[hello, world,
   s11[sailing], s12[boats]]
% Classification
book(d1)
% Attribute value
d1.author(peter)
% Generalization
document(X) :- book(X)
X.cousin(Y) :- X.parent(PX),
        Y.parent(PY), PX.sibling(PY)
```

Figure 1: Basic components of the data model.

Classification: Classification provides the necessary possibility for representing the features of objects. Objects are grouped into a classification structure and they have relationships with other objects.

The second clause states that document (object) `d1` is an instance of the class `book`.

The third clause indicates a relation between object `d1` and object `peter`. This attribute value assignment can be interpreted as a classification, i. e. `peter` is a member of class `d1.author`.

Generalization: Rules enable to formulate generalizations of classes. Reading the first rule: If X is known to be a book, then it is a document. By rules, we add the expressive power of predicate logic for maintaining the knowledge.

A set of aggregation, classification, and generalization clauses is called a program.

The most important aspect of our model is the possibility to arrange programs local to a certain context. These contexts correspond to the aggregation structure of the objects. Similar to [Rijsbergen 89], where each document represents a possible world, in our approach each context assigns truth values to propositions, independent of the truth value assignments of other contexts. Figure 2 shows a program local to document (context) `d1`. The propositions `hello`, `writer(peter)`, etc. hold within context `d1`. In IR, we search for documents (contexts) where a query formula is true. Usually, this query formula only contains simple predicates of arity zero, e. g. `hello`. In our approach, we can also refer to the features of objects which are known within the context and thus we can pose semantically richer queries.

```
d1[% Program within context d1
   hello, world, writer(peter),
   peter.friend(paul),
   peter.friend(mary)]
```

Figure 2: A program local to a context.

The locality provides the basis for regarding documents (contexts, respectively) as consistent sets of propositions. Section 3 shows how the knowledge of documents can be retrieved within the global context (i. e. on the level of the database), since the global context "learns" the knowledge of its components.

At the end of this section, we want to point out the semantical difference between content knowledge like `d1[writer(peter)]` and factual knowledgde like `d1.author(peter)`: The first clause represents that document `d1` has the knowledge that `peter` is a `writer`. This knowledge is local to context `d1`. Syntactically, this locality is indicated by brackets. The second proposition holds in the global context (also referred to as `this`-context) and shows the authorship relation between `d1` and `peter` which is independent of the knowledge of `d1`. We use the common notation of object-oriented languages separating object identifier (`d1`) and attribute name (`author`) by a dot. In the notation of predicate logic, these attributes could be considered as binary predicates, e. g. `author(d1, peter)`.

3 Querying Facilities — Retrieving Aggregated Entities

What do we gain from the introduced data model regarding IR purposes? In this section, we concentrate on exploiting the aggregation structure for making IR more effective and we present the integrated querying for documents and facts. In section 7, we demonstrate the application of the model for full-text, multimedia, and network IR.

In section 2, we have shown how to express the indexing knowledge. Having the knowledge, what can we query for and what should the result look like? A query can refer to the aggregated structure to draw conclusions about the semantical content of objects. This type of queries corresponds to the "aboutness" queries in IR. In addition, a query can refer to the features of objects, e. g. asking for all authors of a document. The set of retrieved objects can be restricted by classification. Further, we want to allow logical conjunction and disjunction for combining various subqueries.

The query result should make the underlying aggregated structure transparent in order to provide intelligent help while browsing. And, by taking into account the aggregated structure, the query process also can determine the least aggregated entity that implies the query. These aggregated entities are constructed by combining entities (contexts), as demonstrated in Figure 3. A query is indicated by `?-` followed by a formula possibly containing variables which differ syntactically from constants in that they are capitalized. The query result yields the variable instantiations which make the formula true. We use the `%`-character for indicating a comment.

```
d1.author(peter)
d1[sailing, boats, sailor(peter)]
d2[s21[sailing], s22[boats], s23[sea]]
d3[s31[peter, paul, mary]]

?- X[sailing] % Content query
d1
d2 ∩ s21

?- X[sailing, boats] % Combination of
d1                   % contexts
d2 ∩ (s21 ∪ s22)

?- X[sailing, mary]
d1 ∪ (d3 ∩ s31)
(d2 ∩ s21) ∪ (d3 ∩ s31)

% Document and fact retrieval
?- X[sailing], X.author(peter)
d1

?- sailing % Augmentation
T % TRUE

?- sailor(X)
peter
```

Figure 3: Query facilities.

Within our model, we have the following query facilities:

Content query: The first query asks for all objects where the proposition `sailing` is true. This type of "aboutness" query is identical to querying for all objects which contain an entity `sailing`. The query result is a set of paths leading to the least entities that fulfill the query. These paths reduce the costs for obtaining the requested information, since the result list has a finer granularity than just a set of document references.

Combination of contexts: In IR, often a single document on its own may not fulfill the query. Considering the aggregated structure of objects, single components will hardly ever satisfy the query completely. For this reason, we introduce the possibility of combining contexts in order to form an object that implies the whole query.

The results of the second and third query demonstrate this concept. `d1` is an answer to the first query. In `d2` the union `s21 ∪ s22` of sections fulfills the query. There is no entity that fulfills the query for `[sailing, mary]`, but the union of several entities make the query formula true. These combined contexts of the answer set form an aggregated entity which can be represented as a directed graph. The whole answer can be delivered as a hierarchical hypertext object and eases the handling for the user.

Combining document and fact retrieval: In document retrieval, one searches for contexts which make the query true. These contexts may contradict each other. On the other hand, fact retrieval in databases refers to the (consistent) knowledge of only one context, i. e. the global context.

In the fourth query, we combine a document query with a factual query. The document query (`X[sailing]`) refers to the local knowledge of a context and the factual query (`X.author(peter)`) addresses the knowledge of the global context. The query result contains the instantiations of the variable X, i. e. the objects (contexts) which make the whole query formula true.

Augmentation: So far, all knowledge is local to a certain context. However, in some applications, we also want to be able to pose queries referring to the local knowledge of the subcontexts directly. For example, in network IR, we would like to know the topics covered by a database, i. e. the topics occuring in the documents of the database. For this purpose, we introduce augmentation. This concept exports the knowledge of subcomponents into the enclosing context. Thus, the enclosing context "learns" the knowledge of its subcontexts and we can pose queries on this knowledge.

The last two queries show the possibility of querying the knowledge augmented from the documents. The predicate `sailing` is true in the global context, since the subcontext `d1` knows it to be true. By means of the classification predicates, we also can query on more complex knowledge. Since the `this`-context knows document `d1` which knows that `peter` is a `sailor`, this knowledge is available and retrievable in the enclosing context.

Summing up, from taking into account the aggregation structure as demonstrated and providing a classification means, we gain:

1. Retrieval on documents of different types is feasible, e. g. a collection of articles and a collection of books can be handled within the same framework. (For example, let `d1` be an article, and let `d2` be a collection of articles `s21`, `s22`, etc. The query process would find `d2` as well as `s21`.) This already indicates the multimedia and networking facilities of the model, since multimedia objects typically differ in their aggregation structure and for network IR a database could be considered as an object containing other objects, and so on (see section 7).

2. The retrieval result contains the least aggregated entities which imply the query. It could be presented as a hypertext object and thus ease browsing. IR becomes more effective.

   Since the retrieval process combines entities (contexts), the finer granularity does not lead to empty query results in the case of conjunctive subqueries. The result can only be empty if none of the predicates are satisfiable for any entity.

3. Querying for documents and facts can be combined.

4. Classification and generalization (rules) are powerful concepts for describing content and querying in a more sophisticated way.

5. Querying the knowledge of sets of documents becomes possible by augmenting the knowledge. This concept provides the nice possibility of using the knowledge of the documents itself instead of only indexing the knowledge about documents. The enclosing object (the database) "learns" from its components (the documents).

4 Negation

In this section, we outline the handling of negation within our model. Most classical IR models use a a closed-world assumption (CWA) for the indexing terms, i. e. if a term is not assigned, then the document is assumed to be *not* about the corresponding subject. We are convinced that the usefulness of negation in IR is strongly connected with making different assumptions for different predicates. Modelling an authorship relation, we would assume that we know all authors of a paper and thus a CWA for this class (predicate) is reasonable. On the other hand, we would not want to assume that we know all friends of a person and thus we want to make an open-world assumption (OWA) for this relationship between objects.

Reconsider the example from above and a query for documents which are not about `sailing`:

```
?- X[¬sailing]
d2 ∩ s22
d2 ∩ s23
d3 ∩ s31 % = d3
```

Like in Boolean retrieval, we use a CWA for the predicate `sailing`. This leads to the depicted result. `d3` on its own is not an element of the answer set, since there is the smaller entity `d3 ∩ s31` that fulfills the query. The same holds for `d2`, but in addition `d2[sailing]` is true, because of the knowledge augmentation, i. e. there exists an entity where `sailing` is true.

Making an OWA, we would have no evidence whether ¬`sailing` is true in any context (entity). Note that in this case only those entities would be in the answer set where ¬`sailing` is explicitly mentioned.

In the following, we use an OWA for `peter.friend` and a CWA for `d1.author`. Making an OWA requires a third truth value: *unknown* (**U**). In contrast to a CWA, where the truth value of a formula is taken to be false, if it is not explicitly defined to be true in a given program, an OWA uses the truth value unknown for a formula when no explicit truth value assignment is given. Consider the following example:

```
peter.friend(paul)
d1.author(peter)

?- peter.friend(paul)
T % TRUE
?- ¬peter.friend(paul)
F % FALSE
?- peter.friend(mary)
U % UNKNOWN
?- ¬peter.friend(mary)
U
?- d1.author(mary)
F
```

The first query yields true, since the proposition is in the database. Thus, the negation is known to be false. But the queries involving `mary` yield unknown, since nothing about this proposition is to be found in the database. Using a CWA, from the database we would have drawn the conclusion that `d1.author(mary)` is false and thus ¬`d1.author(mary)` is true.

Knowledge augmentation and combining contexts may lead to inconsistencies. A proposition may be known to be true in one context whereas the other context knows the same proposition to be false. For coping with this situation, we introduce the fourth truth value *inconsistent* (**I**), reflecting that we have evidence for *both* true *and* false. In the following example, we have two contexts `d1` and `d2` and each context is consistent on its own.

```
d1[¬peter.friend(mary)]
d2[peter.friend(mary)]

?- peter.friend(mary)
I % INCONSISTENT
```

In the global context, we have evidence that `peter.friend(mary)` is both true and false. In this case, the retrieval process yields inconsistent as truth value.

In case of a CWA for a predicate all facts are taken to be false that are not mentioned explicitly to be true. Thus a fact of this predicate can only be true (false) in a union of contexts with a CWA if it is known to be true (false) in all elements of the union. Concerning augmentation, a fact in the enclosing context is true (false) if it is true (false) in all subcontexts having a CWA.

This way, we can direct augmentation and context combination. For example, we could use a CWA on the document level and an OWA on the section level. A union of sections would make a formula true if it is true in one section. A union of documents would make a formula true only if it is true in all documents.

5 Semantics

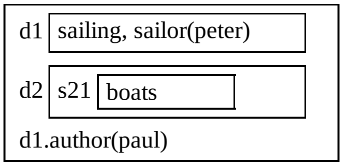
For defining the semantics of the described knowledge representation, we use Kripke structures as introduced in [Halpern & Moses 92], but choose a three-valued truth assignment. Let Φ be a set containing propositions like `sailing`, `sailor(peter)`, `d1.author(peter)`. A Kripke structure is a tuple $M = (S, \pi, \mathcal{K}_1, ..., \mathcal{K}_n)$, where S is a set of possible worlds (states), π is a family of truth value assignments on the propositions $p \in \Phi$ for each world $s \in S$ ($\pi(s) : \Phi \longrightarrow \{\mathbf{T}, \mathbf{F}, \mathbf{U}\}$), and each $\mathcal{K}_i$ is a binary relation over the worlds for an agent K_i. These relations are called accessibility relations. The function $K_i(s) := \mathcal{K}_{i,s} := \{t|(s,t) \in \mathcal{K}_i\}$ assigns the set of worlds which agent K_i can reach from the world s.

Beside being a standard for describing the meaning of a logical program, this model-oriented approach has the following advantages: The accessibility relations connected with the world-dependent truth assignments consider the context-dependency of knowledge. By defining axioms on the Kripke structure, specific properties of the knowledge can be defined. Extending the Kripke structure with a probability assignment provides the means for coping with uncertain knowledge as described in [Fagin & Halpern 94].

In our model the objects (documents) correspond to the agents. Depending on the current world an object knows a set of worlds. Consider the following example:

```
d1[sailing, sailor(peter)]
d2[s21[boats]]
d1.author(paul)
```

Figure 4 depicts the structure of the knowledge represented by the given progam with the corresponding world dependent truth assignment. We have four worlds $\{s_0, s_1, s_2, s_3\}$ and three agents `d1`, `d2`, and `s21` with the corresponding accessibility relations $\{(s_0, s_1)\}, \{(s_0, s_2)\}, \{(s_2, s_3)\}$. Since in this sample case each agent knows exactly one truth value for a proposition, the sets $\mathcal{K}_{i,s}$ of accessable worlds contain exactly one world and we may also use an agent's name for referring to a world. The so-called `this`-agent corres-



$\pi(s)(p)$	s (K_i)			
	s_0 (this)	s_1 (d1)	s_2 (d2)	s_3 (s21)
p				
d1	**T**	**U**	**U**	**U**
d2	**T**	**U**	**U**	**U**
sailing	**U**	**T**	**U**	**U**
sailor(peter)	**U**	**T**	**U**	**U**
s21	**U**	**U**	**T**	**U**
boats	**U**	**U**	**U**	**T**
d1.author(paul)	**T**	**U**	**U**	**U**

Figure 4: Worlds, accessibility, and truth values.

ponds to world s_0. The set Φ of propositions is given by {`d1`, `d2`, `sailing`, `sailor(peter)`, `s21`, `boats`, `d1.author(paul)`}.

For a Kripke structure M and a world s, an interpretation is a function $I_{(M,s)} : L^* \longrightarrow \{\mathbf{T}, \mathbf{F}, \mathbf{U}, \mathbf{I}\}$ which assigns a truth value to every closed formula of the language L. (L^* denotes the set of closed formulas of L.) Figure 5 outlines the grammar of the language. A program

program	::=	{proposition | rule}*
proposition	::=	entity | classification
entity	::=	obj-id | obj-id'['proposition']'
classification	::=	glob-class | attr-val
glob-class	::=	pred-name'('obj-id')'
attr-val	::=	obj-id.pred-name'('obj-id')'
rule	::=	rule-head ':-' rule-body
rule-head	::=	classification$_v$
rule-body	::=	subgoal-conjunction
subgoal	::=	proposition$_v$

Figure 5: Outline of the grammar of the language.

is a set of propositions and rules. A proposition is an entity or classification for describing the aggregation and knowledge of the current object. An entity itself again may have an aggregated structure and classification knowledge. A classification could be a global classification (e. g. `sailor(peter)`) or an attribute value assignment (e. g. `d1.author(paul)`). Within rules variable names may be used in place of object identifiers (indicated by the subscript v). We restrict the formal definition of the semantics to the truth values of closed formulas, assuming a variable valuation for instantiating the rules.

For defining the interpretation function, we consider the truth values {**U**, **T**, **F**, **I**} to be the powersets of {**true**, **false**}, thus obtaining {{}, {**true**}, {**false**}, {**true**, **false**}} accordingly. Figure 6 shows some of the constraints for defining the interpretation function. Let p be a basic entity

$$\begin{array}{lcl}
\textbf{true} \in I_{(M,s)}(p) & \Longleftarrow & \textbf{true} \in \pi(s)(p) \\
\textbf{false} \in I_{(M,s)}(p) & \Longleftarrow & \textbf{false} \in \pi(s)(p) \\
\textbf{true} \in I_{(M,s)}(K_i[\varphi]) & \Longleftrightarrow & \forall t((s,t) \in \mathcal{K}_i \Rightarrow \\
& & \textbf{true} \in I_{(M,t)}(\varphi)) \\
\textbf{false} \in I_{(M,s)}(K_i[\varphi]) & \Longleftrightarrow & \forall t((s,t) \in \mathcal{K}_i \Rightarrow \\
& & \textbf{false} \in I_{(M,t)}(\varphi)) \\
\textbf{true} \in I_{(M,s)}(\neg\varphi) & \Longleftrightarrow & \textbf{false} \in I_{(M,s)}(\varphi) \\
\textbf{false} \in I_{(M,s)}(\neg\varphi) & \Longleftrightarrow & \textbf{true} \in I_{(M,s)}(\varphi)
\end{array}$$

Figure 6: Constraints for the interpretation function.

(without further structuring, e. g. `sailing`) or a classification. The truth value of the interpretation contains **true** (**false**), if the initial truth assignment of the Kripke structure contains **true** (**false**). Remember that the interpretation may yield inconsistent for an atomic proposition according to context combining and augmentation. The third and fourth constraint define the truth value of a formula like `d1[sailor(peter)]`. It contains **true** (**false**), if for all worlds t which agent K_i (`d1`) can reach from the current world s the interpretation of φ (`sailor(peter)`) contains **true** (**false**). The interpretation of a negated formula contains **true** (**false**), if the truth value of the non-negated formula contains **false** (**true**).

Figure 7 depicts the truth values for conjunctions, rules, and negation. In [Rölleke & Fuhr 96], we give the complete set of constraints according to this truth value assignment and we show that this definition of truth values has some desirable properties of two-valued logic.

Now we add the constraints for the augmentation concept. Let $\mathcal{K}$ be the set union over all accessibility relations and let φ be a closed formula.

$$\begin{array}{lcl}
\textbf{true} \in I_{(M,s)}(\varphi) & \Longleftarrow & \exists t((s,t) \in \mathcal{K} \wedge \\
& & \textbf{true} \in I_{(M,t)}(\varphi)) \\
\textbf{false} \in I_{(M,s)}(\varphi) & \Longleftarrow & \exists t((s,t) \in \mathcal{K} \wedge \\
& & \textbf{false} \in I_{(M,t)}(\varphi))
\end{array}$$

If there is one world t where the interpretation of a formula φ contains **true** (**false**), then the interpretation of this formula in the current world s contains **true** (**false**). The truth value of a predicate depends on its truth value in the entities that form the current object. According to this augmentation rule, the formula `d1[sailing]` is true, if `sailing` is true in some section of `d1`.

According to the given definition for an interpretation, we get the following truth values for the formulas of the example:

Conjunction

$I_{(M,s)}(\varphi,\psi)$	$I_{(M,s)}(\psi)$			
	T	**U**	**F**	**I**
$I_{(M,s)}(\varphi)$				
T	**T**	**U**	**F**	**I**
U	**U**	**U**	**F**	**I**
F	**F**	**F**	**F**	**F**
I	**I**	**I**	**F**	**I**

Rule

$I_{(M,s)}(p \coloneq \varphi)$	$I_{(M,s)}(p)$			
	T	**U**	**F**	**I**
$I_{(M,s)}(\varphi)$				
T	**T**	**U**	**F**	**I**
U	**T**	**U**	**U**	**I**
F	**T**	**T**	**T**	**T**
I	**T**	**I**	**I**	**I**

Negation

$I_{(M,s)}(\varphi)$	$I_{(M,s)}(\neg\varphi)$
T	**F**
F	**T**
U	**U**
I	**I**

Figure 7: Truth values.

$I_{(M,s)}(p)$	s			
	this	d1	d2	s21
p				
d1	**T**	**U**	**U**	**U**
d2	**T**	**U**	**U**	**U**
sailing	**T**	**T**	**U**	**U**
sailor(peter)	**T**	**T**	**U**	**U**
s21	**T**	**U**	**T**	**U**
boats	**T**	**U**	**T**	**T**
d1.author(paul)	**T**	**U**	**U**	**U**

The proposition `sailing` is true in the `this`-context, since it is true in `d1`. And so forth.

For demonstrating the use of the truth value inconsistent, imagine we would have further evidence for `d2[¬sailing]`. In this case we would have the following truth value assignment in the Kripke structure with the corresponding interpretation function:

s	$\pi(s)$(¬sailing)	$I_{(M,s)}$(¬sailing)
this	**U**	**I**
d1	**F**	**F**
d2	**T**	**T**
s21	**U**	**U**

The initial truth assignment in the `this`-world is unknown, and the interpretation yields inconsistent according to augmenting inconsistent knowledge from `d1` and `d2`.

In the following, we investigate the combination of the knowledge of objects. We allow an agent to be constructed out of other agents according to the following rules:

$$\begin{aligned}
&\textbf{true} \in I_{(M,s)}(\{K_i \cup K_j\}[\varphi]) \\
&\quad\iff \textbf{true} \in I_{(M,s)}(K_i[\varphi]) \vee \textbf{true} \in I_{(M,s)}(K_j[\varphi]) \\
&\textbf{false} \in I_{(M,s)}(\{K_i \cup K_j\}[\varphi]) \\
&\quad\iff \textbf{false} \in I_{(M,s)}(K_i[\varphi]) \vee \textbf{false} \in I_{(M,s)}(K_j[\varphi]) \\
&\textbf{true} \in I_{(M,s)}(\{K_i \cap K_j\}[\varphi]) \\
&\quad\iff \textbf{true} \in I_{(M,s)}(K_i[K_j[\varphi]]) \\
&\textbf{false} \in I_{(M,s)}(\{K_i \cap K_j\}[\varphi]) \\
&\quad\iff \textbf{false} \in I_{(M,s)}(K_i[K_j[\varphi]])
\end{aligned}$$

The interpretation of a formula within the union of two contexts contains **true** (**false**), if one of its contexts contains **true** (**false**). The intersection of contexts represents a path to a context where the formula is true. The union of contexts is commutative, whereas intersection of contexts is not commutative.

The retrieval process yields for a closed formula the truth value and it determines for an open formula all closed formulas with the truth value **T**.

It determines the set of (combined) contexts where the query condition holds. Only the least contexts are included in the result set, since they imply the enclosing context which is indicated by the path to follow. In case a context on its own does not fulfill a query, a union with another context might fulfill the query (see example in figure 3).

The next table depicts the truth values of some formulas in combined contexts K_i:

$I_{(M,s)}(K_i[p])$	s
	this
$K_i[p]$	
{d1 ∪ d2}[sailing, boats]	**T**
{d2 ∩ s21}[boats]	**T**

The propositions `sailing`, `boats` are true in the combined context {d1 ∪ d2}, since `sailing` is true in context `d1` and `boats` is true in context `d2`. In the intersection {d2 ∩ s21} the formulas have the truth values of context `s21`.

Given this formalization of the semantics, we are able to add a world and proposition dependent CWA.

$$I_{(M,s)}(\varphi) = \mathbf{U} \implies I_{(M,s)}(\neg\varphi) = \mathbf{T}$$

We can use this rule for closing on a certain predicate (class) in a certain context, e. g. `this[CWA(document)]` expresses that the CWA is assumed for the predicate (class) `document` within the `this`-context.

6 Adding Uncertainty

Of course, a state-of-the-art IR system should provide a ranking of the retrieved objects to increase retrieval effectiveness. One reason for the success of the set of terms approach is certainly to be found in its straight forward extension with term weights which leads to a ranking function.

Assuming a probabilistic interpretation of the ranking function, the retrieval process tries to determine the probability of relevance given a query and a document ($P(R|q,d)$). This probability is supposed to be a function of the probability that the implication $d \rightarrow q$ holds ($P(R|q,d) \sim f(P(d \rightarrow q))$).

We have based the semantics of our data model on a four-valued interpretation. Keeping in mind the underlying OWA, we could assign probabilities for the truth values true or false which do not necessarily sum up to one. The overlap of the probabilities reflects the inconsistency, the gap between them mirrors the degree of unknown. The aim of future work will be to investigate the application of probability theory and Dempster-Shafer theory for coping with these uncertainty values.

As an example, here we demonstrate the modelling of the probabilistic interpretation of the vector space model, where $P(d \rightarrow q) = \sum_t (P(t|d) \cdot P(q|t))$ holds (see [Wong & Yao 95]). We consider two (disjoint) documents `d1` and `d2` and process some queries:

```
0.5 d1[0.9 sailing, 0.1 peter,
       0.8 sailor(peter)]
0.5 d2[0.5 sailing, 0.5 boats]

?- X[sailing]
0.9 d1
0.5 d2

1.0 r(X) :- X[sailing]
0.6 r(X) :- X[boats]
?- r(X)
0.9 d1
0.8 d2

?- sailor(peter)
0.4 T
```

The factor 0.5 in front of the document context reflects the portion of the current context that a document covers, i. e. in this case every document has the same importance for the collection. Analogously, the weights in front of the terms indicate the portions of the documents. We interpret these weights as conditional probabilities. For example, the weight 0.9 could be interpreted as the probability $P(t|d) = P(\texttt{sailing}|\texttt{d1}) = P(\texttt{d1[sailing]})$.

In the first and second query, we compute the resulting weights according to the vector space model. The ranking of the result of the first query is determined by the conditional probability $P(t|d)$. Here, $P(q|t) = 1$ is assumed, i. e. the retrieval function does not use query term weighting. In the second query, we use weighted rules (predicate `r`) for representing query term weighting. Document `d2` fulfills both rules. We get $P(\texttt{r(d2)}) = P(\texttt{r(d2)}|\texttt{d2[sailing]}) \cdot P(\texttt{d2[sailing]}) + P(\texttt{r(d2)}|\texttt{d2[boats]}) \cdot P(\texttt{d2[boats]}) = 1.0 \cdot 0.5 + 0.6 \cdot 0.5 = 0.8$ as resulting retrieval weight.

The third query demonstrates the knowledge augmentation. Since in document `d1` we know that `peter` is a `sailor`, we can query on this knowledge in the global context. Considering the disjointness assumption for contexts, knowledge augmentation defines the term space of the whole document collection. For the example above, the term space is defined by {0.7 `sailing`, 0.05 `peter`, 0.25 `boats`}.

The extended Kripke structure $M = (S, \pi, \mathcal{K}_1, ..., \mathcal{K}_n, P)$ defined in [Fagin & Halpern 94] provides a sound framework for integrating uncertainty into the semantics of our data model. The function $P(i,s)$ assigns a probability space $(S_{i,s}, \mu_{i,s})$ to each agent and world, where $S_{i,s}$ is a subset of the worlds $\mathcal{K}_{i,s}$ and $\mu_{i,s}$ is a probability distribution over the σ-algebra of $S_{i,s}$.

Given this structure, the interpretation of a weight is defined by the sum of the probabilities of the worlds, where the proposition is true. For example: $0.9 = \sum_{s \in S_{d1,s_0}} \mu_{d1,s_0}(\{s|\pi(s)(\texttt{sailing}) = \textbf{true}\})$. The agent corresponds to the enclosing context (`d1`) and the `d1`-context considers those worlds which it can reach from the `this`-world s_0.

7 Application Domains

The examples presented so far already indicate the use of the model for full-text document retrieval. The different parts of a document can be considered appropriately. Even documents which differ in their aggregation structure can be handled within one knowledge representation, since the result of a query is a set of paths which may differ in their depth.

Multimedia IR requires handling different types of objects within one knowledge representation. Consider two databases: one document database and one picture database:

```
document-db[d1[...]]
picture-db[p1[...]]
{document-db}[?- ...]
{document-db ∪ picture-db}[?- ...]
```

The above example shows the structure of an integrated knowledge representation for multimedia IR purposes. Queries can be posed context-dependent. The first query considers only the document database. By means of context combination, various contexts for querying can be chosen. For answering the second query the knowledge of both databases (contexts) is united.

Context-dependent querying provides intuitive possibilities for considering context-dependent knowledge (for example user preferences) and for combining several contexts to a new database for querying. This combination of contexts also supports network IR. Since the knowledge representation is object-oriented, media-independent and powerful enough to describe the relationships between semantical objects, the model is especially suitable for processing content retrieval on a distributed multimedia database.

8 Conclusion and Outlook

We have presented a data model built on the object-oriented principles aggregation, classification, and generalization for IR purposes. It provides an integrated knowledge representation for full-text, multimedia, and network IR. Structurally complex knowledge can be represented.

By considering the aggregated structure of objects, their classification and relationships, we gain new query facilities. Querying for documents and facts can be combined and instead of retrieving only whole documents, the least aggregated entity that implies the query is determined. The query result can be delivered as a hypertext document. This feature offers the possibility of improving the effectiveness of IR.

The major concerns of this paper are the knowledge representation and new querying facilities. We indicated how to take into account the intrinsic uncertainty of knowledge and vagueness of queries in order to achieve a ranking of the retrieved objects. The semantics we use offers a well-defined possibility to add different flows of probabilities for obtaining the desired ranking with a sound meaning.

Further work will be done in investigating the suitability of probability theory versus Dempster-Shafer theory. Concerning aggregation we want to exploit more knowledge about the given ordering. A database is just a set of documents, whereas the sections of a document have a specific order and should be considered as a list. This ordering could be useful for interpreting the knowledge of a document and for browsing the retrieval result.

A Truth Values

In this appendix, we present our considerations for choosing the truth assignment of the interpretation function as given in section 5. Review figure 6 and the following constraints:

$$\begin{array}{lcl}
\textbf{true} \in I_{(M,s)}(\varphi,\psi) & \iff & \textbf{true} \in I_{(M,s)}(\varphi) \wedge \textbf{true} \in I_{(M,s)}(\psi) \\
\textbf{false} \in I_{(M,s)}(\varphi,\psi) & \iff & \textbf{false} \in I_{(M,s)}(\varphi) \vee \textbf{false} \in I_{(M,s)}(\psi) \\
\textbf{true} \in I_{(M,s)}(p \text{ :- } \varphi) & \iff & \textbf{true} \in I_{(M,s)}(\varphi) \wedge \textbf{true} \in I_{(M,s)}(p) \vee \\
& & \textbf{false} \in I_{(M,s)}(\varphi) \wedge \textbf{true} \in I_{(M,s)}(p) \vee \\
& & \textbf{false} \in I_{(M,s)}(\varphi) \wedge \textbf{false} \in I_{(M,s)}(p) \\
\textbf{false} \in I_{(M,s)}(p \text{ :- } \varphi) & \iff & \textbf{true} \in I_{(M,s)}(\varphi) \wedge \textbf{false} \in I_{(M,s)}(p)
\end{array}$$

This was our first approach to define the truth values following the notation given in [**?**] for defining the constraints for an interpretation function. The constraints follow intuitively from the consideration of the truth values as the powerset of $\{$**true**, **false**$\}$. Given these constraints, we get the following truth value assignment:

Conjunction

$I_{(M,s)}(\varphi,\psi)$	$I_{(M,s)}(\psi)$			
$I_{(M,s)}(\varphi)$	**T**	**U**	**F**	**I**
T	**T**	**U**	**F**	**I**
U	**U**	**U**	**F**	**(F)**
F	**F**	**F**	**F**	**F**
I	**I**	**(F)**	**F**	**I**

Rule

$I_{(M,s)}(p \text{ :- } \varphi)$	$I_{(M,s)}(p)$			
$I_{(M,s)}(\varphi)$	**T**	**U**	**F**	**I**
T	**T**	**U**	**F**	**I**
U	**(U)**	**U**	**U**	**(U)**
F	**T**	**(U)**	**T**	**T**
I	**T**	**(U)**	**I**	**I**

Negation

$I_{(M,s)}(\varphi)$	$I_{(M,s)}(\neg\varphi)$
T	**F**
F	**T**
U	**U**
I	**I**

The parentheses indicate the truth values which we address in the following.

Adding rules for disjunction,

$$\begin{array}{lcl}
\textbf{true} \in I_{(M,s)}(\varphi\tilde{\vee}\psi) & \iff & \textbf{true} \in I_{(M,s)}(\varphi) \vee \textbf{true} \in I_{(M,s)}(\varphi) \\
\textbf{false} \in I_{(M,s)}(\varphi\tilde{\vee}\psi) & \iff & \textbf{false} \in I_{(M,s)}(\varphi) \wedge \textbf{false} \in I_{(M,s)}(\varphi)
\end{array}$$

we get the following truth value assignment:

$I_{(M,s)}(\varphi\tilde{\vee}\psi)$	**T**	**U**	**F**	**I**
T	**T**	**T**	**T**	**T**
U	**T**	**U**	**U**	**T**
F	**T**	**U**	**F**	**I**
I	**T**	**T**	**I**	**I**

In this case, DeMorgan $((\neg(\neg\varphi \tilde{\wedge} \neg\psi)) \iff (\varphi \tilde{\vee} \psi))$ would hold, but the implication equivalence $((\varphi \Rightarrow p) \iff (\neg\varphi \tilde{\vee} p))$ would not hold. See the following truth value assignments:

$I_{(M,s)}(\neg(\neg\varphi\tilde{\wedge}\neg\psi))$	**F**	**U**	**T**	**I**
F	**T**	**T**	**T**	**T**
U	**T**	**U**	**U**	**T**
T	**T**	**U**	**F**	**I**
I	**T**	**T**	**I**	**I**

$I_{(M,s)}(\neg\varphi\tilde{\vee}p)$	**T**	**U**	**F**	**I**
F	**T**	**U**	**F**	**I**
U	**(T)**	**U**	**U**	**(T)**
T	**T**	**(T)**	**T**	**T**
I	**T**	**(T)**	**I**	**I**

The border column and row show the truth value of $\neg\varphi$, etc. as used in conjunction and disjunction, respectively.

The intuitive definition of an implication yields unknown at the marked places, whereas the (equivalent) expression yields true.

Further Constraints: The problem lies (no surprise) in the truth values inconsistent and unknown and the combination of both:

In our first approach, we got $\mathbf{I}\tilde{\wedge}\mathbf{U} = \mathbf{F}!?$ and $\mathbf{I}\tilde{\vee}\mathbf{U} = \mathbf{T}!?$. If we change this definition to $\mathbf{I}\tilde{\wedge}\mathbf{U} = \mathbf{I} = \mathbf{I}\tilde{\vee}\mathbf{U}$, then we get a truth value assignment closer to the laws of [**?**] and DeMorgan *and* the implication equivalence hold.

Adding the following constraints leads to the desired truth values.

$$\begin{array}{lcl}
\textbf{true} \in I_{(M,s)}(\varphi\tilde{\wedge}\psi) & \iff & I_{(M,s)}(\varphi) = \mathbf{U} \wedge I_{(M,s)}(\psi) = \mathbf{I} \vee \\
 & & I_{(M,s)}(\psi) = \mathbf{U} \wedge I_{(M,s)}(\varphi) = \mathbf{I} \vee \\
\textbf{false} \in I_{(M,s)}(\varphi\tilde{\vee}\psi) & \iff & I_{(M,s)}(\varphi) = \mathbf{U} \wedge I_{(M,s)}(\psi) = \mathbf{I} \vee \\
 & & I_{(M,s)}(\psi) = \mathbf{U} \wedge I_{(M,s)}(\varphi) = \mathbf{I} \vee \\
\textbf{true} \in I_{(M,s)}(p \text{ :- } \varphi) & \iff & I_{(M,s)}(\varphi) = \mathbf{U} \wedge \textbf{true} \in I_{(M,s)}(p) \vee \\
 & & I_{(M,s)}(p) = \mathbf{U} \wedge \textbf{false} \in I_{(M,s)}(\varphi) \vee \ldots \\
\textbf{false} \in I_{(M,s)}(p \text{ :- } \varphi) & \iff & I_{(M,s)}(\varphi) = \mathbf{I} \wedge I_{(M,s)}(p) = \mathbf{I} \vee \\
 & & I_{(M,s)}(p) = \mathbf{I} \wedge I_{(M,s)}(\varphi) = \mathbf{U} \vee \ldots
\end{array}$$

The dots (...) indicate that these constraints are a completion of the previous definition.

See the following result of truth value assignments, where we indicated the truth value assignments in question by parentheses.

Conjunction

$I_{(M,s)}(\varphi,\psi)$	$I_{(M,s)}(\psi)$			
	T	**U**	**F**	**I**
$I_{(M,s)}(\varphi)$				
T	**T**	**U**	**F**	**I**
U	**U**	**U**	**F**	**(I)**
F	**F**	**F**	**F**	**F**
I	**I**	**(I)**	**F**	**I**

Rule

$I_{(M,s)}(p \text{ :- } \varphi)$	$I_{(M,s)}(p)$			
	T	**U**	**F**	**I**
$I_{(M,s)}(\varphi)$				
T	**T**	**U**	**F**	**I**
U	**(T)**	**U**	**U**	**(I)**
F	**T**	**(T)**	**T**	**T**
I	**T**	**(I)**	**I**	**I**

Now DeMorgan and the implication equivalence hold.

$I_{(M,s)}(\varphi\tilde{\vee}\psi)$	**T**	**U**	**F**	**I**
T	**T**	**T**	**T**	**T**
U	**T**	**U**	**U**	**(I)**
F	**T**	**U**	**F**	**I**
I	**T**	**(I)**	**I**	**I**

$I_{(M,s)}(\neg(\neg\varphi\tilde{\wedge}\neg\psi))$	**F**	**U**	**T**	**I**
F	**T**	**T**	**T**	**T**
U	**T**	**U**	**U**	**I**
T	**T**	**U**	**F**	**I**
I	**T**	**I**	**I**	**I**

$I_{(M,s)}(\neg\varphi\tilde{\vee}\psi)$	**T**	**U**	**F**	**I**
F	**T**	**U**	**F**	**I**
U	**T**	**U**	**U**	**I**
T	**T**	**T**	**T**	**T**
I	**T**	**I**	**I**	**I**

References

Brown, H. (1989). Standards for Structured Documents. *The Computer Journal 32(6)*, pages 505–514.

Christophides, V.; Abiteboul, S.; Cluet, S. (1994). From Structured Documents to Novel Query Facilities. In: Snodgrass, R. T.; M., W. (eds.): *Proceedings of the 1994 ACM-SIGMOD. International Conference on Management of Data.*, pages 313–324. ACM, New York.

Colby, L. S.; Saxton, L. V.; van Gucht, D. (1994). Concepts for Modeling and Querying List-Structured Data. *Information Processing & Management 30(5)*, pages 687–709.

Fagin, R.; Halpern, J. (1994). Reasoning About Knowledge and Probability. *Journal of the ACM 41(2)*, pages 340–367.

Fuhr, N.; Rölleke, T. (1996). A Probabilistic Relational Algebra for the Integration of Information Retrieval and Database Systems. (To appear in: ACM Transactions on Information Systems).

Fuhr, N. (1995a). Logical and Conceptual Models for the Integration of Information Retrieval and Database Systems. In: *East/West Database Workshop, Klagenfurt 1994*, pages 206–218. Springer, Berlin et al.

Fuhr, N. (1995b). Probabilistic Datalog - a Logic for Powerful Retrieval Method. In: Fox, E.; Ingwersen, P.; Fidel, R. (eds.): *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 282–290. ACM, New York.

Fuhr, N. (1996). *Object-Oriented and Database Concepts for the Design of Networked Information Retrieval Systems*. (submitted for publication). URL: http://ls6-www.informatik.uni-dortmund.de/reports/96/Fuhr-96.html.

Halpern, J.; Moses, Y. (1992). A Guide to Completeness and Complexity for Modal Logics of Knowledge and Belief. *Artificial Intelligence 54*, pages 319–379.

Kifer, M.; Lausen, G.; Wu, J. (1995). Logical Foundations of Object-Oriented and Frame-Based Languages. *Journal of the Association for Computing Machinery 42(4)*, pages 741–843.

Macleod, I. (1990). Storage and Retrieval of Structured Documents. *Information Processing and Management 26(2)*, pages 197–208.

Meghini, C. (1995). An Image Retrieval Model Based on Classical Logic. In: Fox, E.; Ingwersen, P.; Fidel, R. (eds.): *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 300–309. ACM, New York.

Pfeifer, U.; Fuhr, N.; Huynh, T. (1995). Searching Structured Documents with the Enhanced Retrieval Functionality of freeWAIS-sf and SFgate. In: D. Kroemker (ed.): *Computer Networks and ISDN Systems; Proceedings of the third International World-Wide Web Conference*, pages 1027–1036. Elsevier, Amsterdam - Lausanne - New York - Oxford - Shannon - Tokyo.

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

van Rijsbergen, C. J. (1989). Towards an Information Logic. In: Belkin, N.; van Rijsbergen, C. J. (eds.): *Proceedings of the Twelfth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 77–86. ACM, New York.

Rölleke, T.; Fuhr, N. (1996). *Retrieval of Complex Objects Using a Four-Valued Logic*. Technical report, University of Dortmund.

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