SEMANTICS-BASED INFORMATION RETRIEVAL

Ralf Möller, Volker Haarslev, Bernd Neumann [1]

Abstract:
In this paper we investigate the use of conceptual descriptions based on description logics for content-based information retrieval and present several innovative contributions. We provide a query-by-examples retrieval framework which avoids the drawback of a sophisticated query language. We extend an existing DL to deal with spatial concepts. We provide a content-based similarity measure based on the least common subsumer which extracts conceptual similarities of examples.

1 Introduction

As more and more information of various kinds becomes available for an increasing number of users, one major challenge for Computer Science is to provide efficient access and retrieval mechanisms. This is not only true for Web-based information which by its nature tends to be highly unorganized and heterogeneous, but also for dedicated databases which are designed to provide a particular service. The guiding example of this paper is a "TV-Assistant" with a database containing TV-program information. Its task is to assist TV watchers in selecting "their" favorite program item from a potentially large set of candidates. For example, the TV-Assistant should be able to identify "a pirate movie with sailing-boats" among the 300 movies which a new German digital TV channel broadcasts every 30 minutes.

There is obviously a large variety of criteria by which TV watchers would like to express their preferences. They may want to refer to the contents of the program item in terms of its genre type, main characters, location, historical events, plot etc. They may also want to refer to production information, e.g. producer, cast, recording technique, date of origin etc., maybe also to their particular viewer situation, e.g. language and age requirements. While some of these criteria can already be used in existing TV-program services (e.g. genre, cast, age recommendations), content-based retrieval is in its infancy.

The prevailing approaches for content-based access and retrieval utilize textual information in terms of keywords and word statistics. Surface-based textual information retrieval, typically based on string-indexing, offers several advantages, in particular the use of queries involving natural language terms, and the availability of text documents. On the other hand, string-indexing is unreliable in several respects. First, documents may not be produced with the aim to support textual retrieval. Hence it is a matter of chance whether or not a desirable keyword really appears in the document. Second, as examples of TV-program selection show, naturally expressed queries may involve terms which are less specific than the textual descriptor of the data (or only conceptually close to it), e.g. "sailing ship" instead of "fregate". Similarly, one

[1]University of Hamburg, Computer Science Department, Vogt-Kölln-Str. 30, 22527 Hamburg, Germany, `http://kogs-www.informatik.uni-hamburg.de/~<name>/`

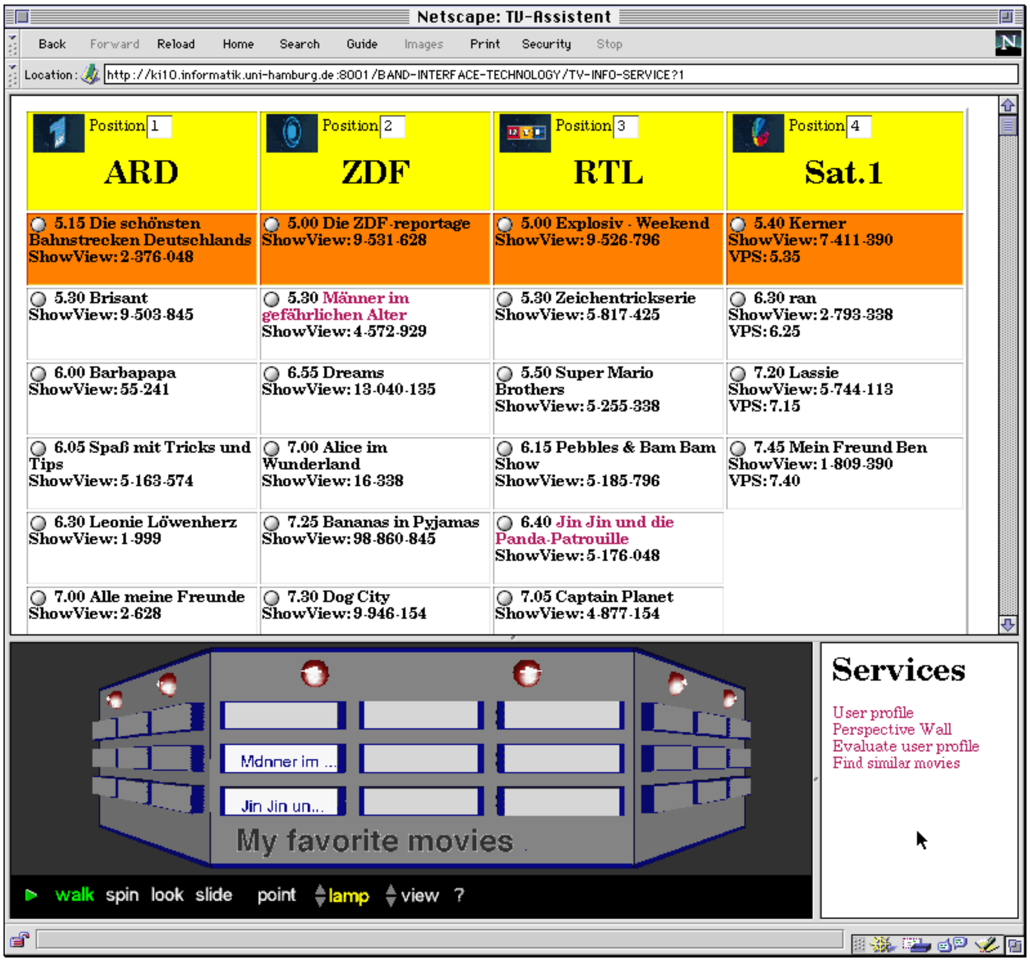


Figure 1: Screenshot from the TV-Assistant HTML page (German TV-program). Program information is printed in the large frame. The user can build up a collection of films. Parts of the collection can be placed on different pinboards which are arranged as a Perspective Wall [9]. The Perspective Wall is implemented using the VRML language. It can be rotated with the mouse, items on pinboards can be opened etc.

may be interested in a movie about one's home town, say Hamburg. Content-based retrieval should not only index into descriptors involving the string "Hamburg" but possibly also into locations spatially related to Hamburg, e.g. "Northern Germany" or "Reeperbahn" (Hamburgs famous red-light district). It is also apparent that additional conceptual information must be exploited to avoid unwanted retrieval hits involving certain popular food items.

In this paper we investigate the use of conceptual descriptions based on description logics (DL) for content-based information retrieval wrt. the *semantics* of the data model and present several innovative contributions. (1) We provide a query-by-examples retrieval framework which avoids the drawback of a sophisticated query language (see Figure 1). (2) We extend an existing DL to deal with spatial concepts. (3) We provide a content-based similarity measure based on the LCS (Least Common Subsumer) which computes conceptual similarities of examples.

2 Domain modeling with description logics

Description logic is a very general name for different theories and practical systems [15]. One implementation of a DL is the CLASSIC system [1] which will be used for the examples presented in this paper[2]. The logical semantics of DL modeling constructs provides a sound basis for runtime type inferences. The main advantages of the DL perspective are that (i) inferences about domain objects can be formally modeled and (ii) incomplete "conceptual" information about objects (see below) can be adequately handled. Thus, rather than relying on surface-based comparisons, information retrieval can be based on the semantics of the data model. It is necessary to ensure the decidability of the satisfiability and subsumption problems (see below for a deeper discussion of the notion of subsumption). This section introduces the main features of description logics using a small application scenario.

2.1 Language constructs: Syntax

Basically, DL formalisms distinguish between two kinds of building blocks: a set of *atomic concepts* $\mathcal{C}$ (in the following, members of $\mathcal{C}$ are denoted by A) and a set of *atomic roles* $\mathcal{R}$ (members are denoted by R). Concepts denote sets of domain objects. Roles denote tuples of domain objects. We assume that the set of all domain objects is referred to by the predefined concept `classic-thing`. For a given domain object o, the objects related to o via a role R are referred to as the *fillers* of R (with respect to o). A subset $\mathcal{F}$ of $\mathcal{R}$ denotes roles which have at most one filler on the righthand side. These roles are called *attributes* or *features* and are denoted by F and G (possibly with index). Using these *atomic* concepts and roles, *concept terms* (denoted by C, possibly with index) can be defined by the following grammar (n is a natural number):

$C \rightarrow$	`classic-thing` $\mid A$	*Atomic Concepts*
	$\mid$ `(and` $C_1 \ldots C_n$`)`	*Conjunction*
	$\mid$ `(all` $R\ C$`)`	*Role Value Restriction*
	$\mid$ `(at-least` $n\ R$`)`	*Minimum Number Restriction*
	$\mid$ `(at-most` $n\ R$`)`	*Maximum Number Restriction*
	$\mid$ `(equal (compose` $F_1 \ldots F_k$`) (compose` $G_1 \ldots G_h$`))`	*Equality Restriction*

Concept terms are also called concept descriptions or, more briefly, concepts. Intuitively, concept terms denote those objects that fulfill the description. For the operator `and`, both concepts must be fulfilled. An `all` restriction imposes constraints on the filler of a role while the `at-least` and `at-most` operators impose restrictions on the cardinality of the set of role fillers. The `equal` operator declares the fillers of attribute chains (constructed by `compose`) to be equal (see below for a formal definition of the semantics).

2.1.1 TBox

The notion of a *terminology* is used to assign the meaning of concept terms to atomic concepts using *terminological axioms* (sometimes called *definitions* or *declarations*). A so-called "primitive" atomic concept is declared with the terminological axiom `(define-primitive-concept`

[2]CLASSIC can be licensed from AT&T Bell Labs.

A C). Intuitively, A is only partially defined with necessary conditions. If also sufficient conditions are given, a so-called "defined" concept is declared (introduced with `(define-concept` A C `))`. In this case, intuitively, the atomic concept on the lefthand side is "equivalent" to the concept term on the righthand side (see below for a formal definition). For instance, when the description logic system proves that the (sufficient) concept term on the righthand side of a definition is fulfilled for a certain object, then the concept on the left side of the definition is also fulfilled. Roles and attributes are declared by `(define-primitive-role` R `)` and `(define-primitive-attribute` F `)`, respectively.[3]

The set of terminological axioms is called *TBox* or *terminology* if (i) no cyclic definitions are present, i.e. definitions where a concept name A on the lefthand side occurs either directly or indirectly in some construct on the righthand side are not allowed, and (ii) for each atomic concept on the lefthand side only one terminological axiom is given.

Our initial TBox for the example domain is defined as follows:

```
(define-primitive-concept person classic-thing)
(define-primitive-concept cargo-object classic-thing)
(define-primitive-concept port classic-thing)
(define-primitive-concept container cargo-object)
(define-concept passenger (and person cargo-object))
(define-primitive-concept captain person)
```

CLASSIC also supports a limited form of role terms: the inverse of a role can be declared as well. In this paper, we use additional arguments to `define-primitive-attribute` to introduce inverse roles and attributes.

```
(define-primitive-attribute has-captain :inverse has-ship :inverse-attribute t)
(define-primitive-attribute has-home-port)
(define-primitive-attribute current-port)
(define-primitive-role has-cargo-object)

(define-primitive-concept ship
  (and (all has-captain captain) (all has-home-port port) (all current-port port)))
```

The concept `ship` is explained in detail: The declaration specifies that the filler of the attribute `has-home-port` must be an instance of `port`, i.e. in DL terminology, the filler must be *subsumed* by `port`. The filler of the attribute `has-captain` must be a captain and so on.

For attributes (or attribute chains) it is possible to declare the equality of fillers. Attribute chains are denoted with `(compose` F_1 F_2 ... `)`. If only one attribute is mentioned in a chain, the `compose` operator can be omitted. For instance, the concept `ship-in-home-port` could be defined as follows:

```
(define-primitive-concept ship-in-home-port
    (and ship (equal has-home-port current-port)))
```

[3]In CLASSIC the members of $\mathcal{C}$, $\mathcal{R}$ and $\mathcal{F}$ are implicitly defined by the lefthand sides of the corresponding terminological axioms (hence, axioms are also called declarations).

2.1.2 ABox

In order to represent knowledge about individual worlds, the language is extended. Let $\mathcal{O}$ be a set of object names or *individuals* (denoted with I, possibly with index). CLASSIC provides two kinds of so-called *assertional axioms* of the form `(state (instance` I C`))` or `(state (related` I_1 I_2 R`))`. Intuitively, the first axiom states that the individual I fulfills the concept description C. With the second axiom, the individuals I_1 and I_2 are related via the role R. The complete set of assertional axioms is called *ABox*.

Before we discuss the application of the DL for information retrieval in general and specific extensions in detail we first give a brief overview of the semantics of the language constructs in order to formally define the inference services which a DL reasoner provides.

2.2 Language constructs: Semantics

The meaning of the DL constructs can be given in terms of a set-theoretic semantics by means of an *interpretation*. An interpretation is a tuple $\langle \mathcal{D}, \xi \rangle$ where the set $\mathcal{D}$ is the domain (universe of discourse) and ξ an assignment function such that $\xi : \mathcal{C} \longrightarrow 2^{\mathcal{D}}$, $\xi : \mathcal{R} \longrightarrow 2^{\mathcal{D}} \times 2^{\mathcal{D}}$ where $2^{\mathcal{D}}$ denotes the powerset of the domain $\mathcal{D}$. Attributes are mapped onto functions. ξ is also called the *interpretation function* and must satisfy the following conditions:

$$\begin{aligned}
\xi[\texttt{classic-thing}] &= \mathcal{D} \\
\xi[\texttt{(and } C_1 \ldots C_n\texttt{)}] &= \cap_{i=1}^{n} \xi[C_i] \\
\xi[\texttt{(at-least } n\ R\texttt{)}] &= \{x \mid \|\{(x,y) \mid (x,y) \in \xi[R]\}\| \geq n\} \\
\xi[\texttt{(at-most } n\ R\texttt{)}] &= \{x \mid \|\{(x,y) \mid (x,y) \in \xi[R]\}\| \leq n\} \\
\xi[\texttt{(all } R\ C\texttt{)}] &= \{x \mid \forall y : (x,y) \in \xi[R] \Rightarrow y \in \xi[C]\} \\
\xi[\texttt{(equal (compose } F_1 F_2 \ldots F_k\texttt{) (compose } G_1 G_2 \ldots G_h\texttt{))}] &= \\
&\{x \mid \xi[F_k](\ldots \xi[F_1](x)) = \xi[G_h](\ldots \xi[G_1](x))\}
\end{aligned}$$

The meaning of terminological axioms is defined by the notion of *satisfiability*. The terminological axioms `(define-primitive-concept` A C `)` and `(define-concept` A C `)` are *satisfied* by an interpretation ξ if $\xi[\mathsf{A}] \subseteq \xi[C]$ and $\xi[\mathsf{A}] = \xi[C]$, respectively. An interpretation that satisfies all axioms in a terminology is called a *model*. The semantics for assertions about individual objects is defined analogously: `(state (instance` I C `))` is satisfied if $\xi[I] \in \xi[C]$, `(state (related` I_1 I_2 R `))` is satisfied if $(\xi[I_1], \xi[I_2]) \in \xi[R]$. An interpretation is called a model for an ABox if all TBox and ABox axioms are satisfied. The notion of a model is used to define the reasoning services provided by a DL inference engine.

2.3 Inference services

The main services are subsumption and consistency checking which are closely related. A term C is *consistent* iff there exists a model $\langle \mathcal{D}, \xi \rangle$ such that $\xi[C] \neq \emptyset$. A term C_1 *subsumes* another term C_2 iff for every model $\langle \mathcal{D}, \xi \rangle$ it holds that $\xi[C_2] \subseteq \xi[C_1]$. A concept C_1 is defined to be a *superconcept* of a concept C_2 iff C_1 subsumes C_2. Two concept terms are *equivalent* iff they subsume each other. An individual I is subsumed by a concept C (given a specific TBox and ABox) iff `(state` I C `)` is satisfied in all models.

The task of a DL inference engine is (i) to check whether the given axioms are consistent and (ii) to derive implicit subsumption and equivalence relations that follow from the declared definitions. It can be shown that the Basic-CLASSIC reasoner provides a sound and complete inference engine with polynomial time complexity [2]. The facilities provided by description logic reasoners comprise services to validate the data model by classifying concepts and relations as well as consistency checking of the whole model. The next subsection discusses some examples where this is important. The derivation of implicit information is a prerequisite for applying description logics in our information retrieval scenario.

2.3.1 Example: Concept classification

As we have seen, grounding the inference engine on a formal semantics ensures that implicit subsumption relations between defined concepts can be automatically detected provided that sound and complete inference services are available. For example, in our scenario the TBox could be extended by the following concept definitions:

```
(define-concept ship-with-captain (and ship (at-least 1 has-captain)))
(define-concept ship-with-cargo
   (and ship (at-least 1 has-cargo-object) (at-least 1 has-captain)))
```

Though not explicitly stated, due to the semantics given above, it is evident that `ship-with-cargo` is also a subconcept of `ship-with-captain` and the TBox reasoner recognizes `ship-with-captain` as a superconcept of `ship-with-cargo`.

2.3.2 Example: Instance classification and queries

In some circumstances a concept should only subsume an instance when the instance is set into relation to another instance. Thus, a `ship` should only be subsumed by the concept `ship-in-shipyard` when it is set into relation to a `shipyard`. A similar situation occurs when "persons" become "customers" if they are set into relation to a "bank". This dynamic classification is no problem when DL concepts are used. The ship example is continued with the following declarations and assertions.

```
(define-primitive-concept ship-in-shipyard ship)
(define-primitive-role has-ship-in-repair-dock nil)
(define-primitive-concept shipyard (all has-ship-in-repair-dock ship-in-shipyard))

(state (instance s1 ship))
(state (instance yard1 shipyard))
(state (related yard1 s1 has-ship-in-repair-dock))
```

Even though `s1` is created as a ship, the dynamic reclassification mechanism forces `s1` to be also an instance of `ship-in-shipyard` just because `s1` is set into relation `has-ship-in-repair-dock` to `yard1` and, by definition, *all* fillers are subsumed by `ship-in-shipyard`. The classification

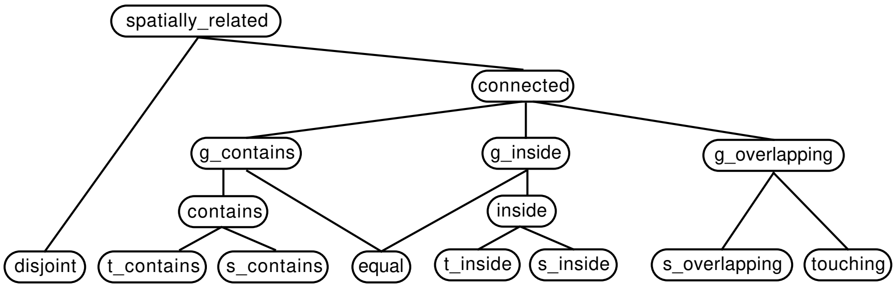
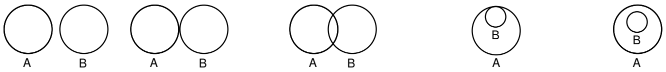


Figure 2: Subsumption hierarchy of spatial relations.



disjoint touching s_overlapping t_contains s_contains

Figure 3: Elementary spatial relations between A and B.

is dynamic because `s1` will no longer be a `ship-in-shipyard` when the related statement is retracted. After asserting the ABox statement `(state (related s1 c1 has-captain))`, the ship `s1` is automatically classified as a `ship-with-captain` because the sufficient conditions for `ship-with-captain` are fulfilled (see also the definition of `ship` presented above).

In addition to dynamic classification, the ABox reasoner of the CLASSIC system can retrieve all instances that are subsumed by a certain concept. In other words, concepts can be used as queries and the DL language is also a query language. For implementing an intelligent information retrieval system, the dynamic computation of query concepts is the key idea. As we will see later, concepts can also be used to represent the commonalities between domain objects. Before we will discuss concept-computing operations in detail, we extend the DL formalism in order to adequately represent spatial information.

2.4 Support for reasoning about spatial objects

In the TV-Assistant domain, spatial information plays an important role. For instance, there might be a user of the TV-Assistant who lives in Hamburg and has special interest in films about objects or events related to ships (sailing ship documentations, movies with pirates, etc.). This section introduces extensions to the basic description logic which support inference services over spatial regions used in concept terms.

We define fifteen binary topological relations that are organized in a subsumption hierarchy (see Figure 2). The leaves of this hierarchy represent eight mutually exclusive relations (*elementary* relations) that are equivalent to the set of relations defined in [5] or [13]. The non-elementary relations are defined to be a disjunction of the corresponding subrelations (see Figure 2). Figure 3 illustrates five elementary relations (the inverses and the relation 'equal' have been omitted). Due to lack of space we refer to [11] for a formal definition of these relations.

In order to support spatial inferences, we extended CLASSIC with new concept constructors based on the spatial relations [7]. Our semantics assumes that each domain object is associated with its spatial representation (i.e. a polygon) via a predefined attribute `has-area`. Concepts for spatial objects are denoted as `(exists-area` *sr p*`)` where *sr* is a relation from Figure 2

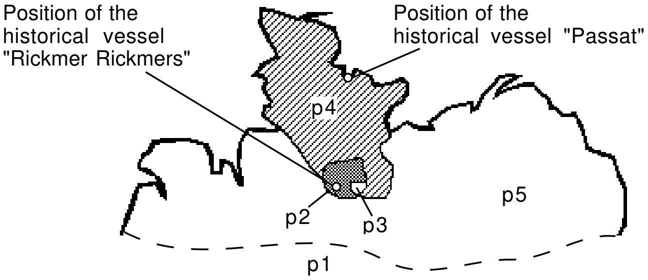


Figure 4: A sketch of the northern part of Germany with polygons for Germany (p1), Northern Germany (p5), the federal states Schleswig-Holstein (p4) and Hamburg (p2) as well as a small district of Hamburg (p3). Polygon p3 is assumed to be inside p2 but p2 is not inside p4.

and p is a name for a polygon constant. We extend the range of the DL interpretation function ξ to the set of polygons $\mathcal{P}$ where each polygon $p \in \mathcal{P}$ defines a subset of $\mathbb{R}^2$. The operator (exists-area sr p) has the following semantics.

$$\xi[\texttt{(exists-area } sr\ p\texttt{)}] = \{x |\ \exists\, q \in \mathcal{P} : (x, q) \in \xi[\texttt{has}-\texttt{area}], (q, p) \in \xi[sr])\}$$
$$\text{with } \xi[sr] \subseteq \mathcal{P} \times \mathcal{P}$$

We can use the constructor to define concepts for a region in Northern Germany and for a district of the city of Hamburg (see Figure 4 for a sketch of the polygons):

```
(define-concept northern-german-region (exists-area g_inside p5))
(define-concept district-of-hh (exists-area inside p2))
```

The construct (exists-area g_inside p5) subsumes every region of Northern Germany whose associated polygon is g_inside of p5 (see Figure 4). We also define concepts for the federal states Hamburg and Schleswig-Holstein.

```
(define-concept federal-state-hh (exists-area equal p2))
(define-concept federal-state-sh (exists-area equal p4))
```

For instance, federal-state-hh is subsumed by northern-german-region because the set $\xi[$(exists-area equal p2)$]$ is a subset of $\xi[$(exists-area g_inside p5)$]$.

In many cases, restrictions about spatial relations have to be combined with terminological restrictions. For example, how can we define a concept that describes a district of Hamburg that touches the federal state Hamburg from the inside? This requires some kind of qualified existential quantification with concept restrictions. Therefore, we propose the concept-forming operator (exists-so sr C)[4] with the following semantics (let sr denote a spatial relation and C be a concept term):

$$\xi[\texttt{(exists-so } sr\ C\texttt{)}] = \{x |\ \exists y_1, y_2, z :\ (x, y_1) \in \xi[\texttt{has}-\texttt{area}], (z, y_2) \in \xi[\texttt{has}-\texttt{area}],$$
$$(y_1, y_2) \in \xi[sr],\ x \neq z,\ z \in \xi[C]\}$$

With this new operator we define the following two concepts which are used to define an instance referring to p3 in Figure 4.

[4]The suffix so stands for "spatial object".

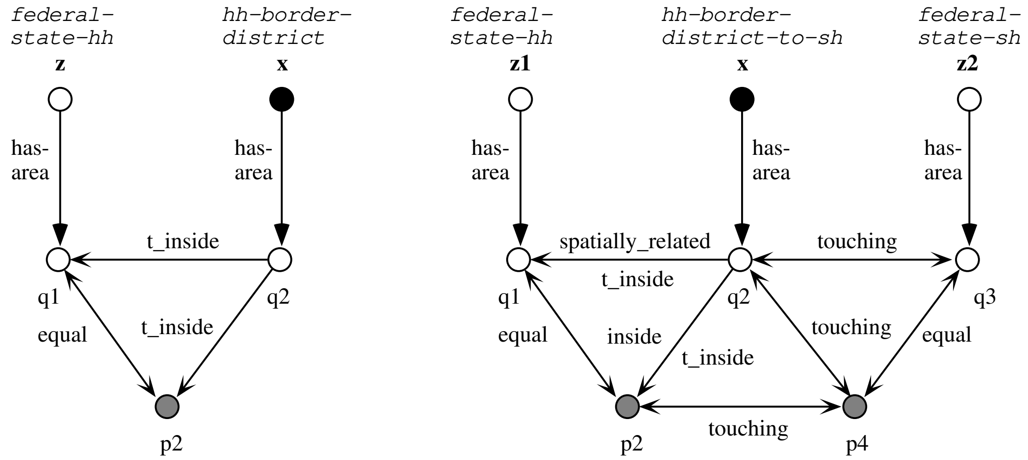


Figure 5: Spacenet representing the restrictions imposed by the concept `hh-border-district` (to the left) and `hh-district-touching-sh` (to the right).

```
(define-concept hh-border-district
   (and district-of-hh (exists-so t_inside federal-state-hh)))
(define-concept hh-district-touching-sh
   (and district-of-hh
        (exists-so touching federal-state-sh)
        (exists-so spatially_related federal-state-hh)))
```

It can be proven that `hh-district-touching-sh` is subsumed by `hh-border-district`. The constraints given in the definition of `hh-district-touching-sh` have to be "normalized" in order to make the implicit subsumption relationship apparent, i.e. constraints given in different conjuncts have to be "accumulated." The constraints imposed by `hh-border-district` and `hh-district-touching-sh` are shown as so-called spacenets in Figure 5. The topological relation `touching` between `p2` and `p4` is automatically computed on the basis of concrete geometric data associated with the corresponding polygons. After computing relations between polygons given as constants, constraint propagation techniques are used to compute implicit relations from the initial information given directly in the concept terms. In Figure 5 we see that `spatially_related` and `inside` are both restricted to `t_inside` by this process. The spacenet for `hh-border-district` is a subgraph of the spacenet for `hh-district-touching-sh` and, therefore, the former subsumes the latter. Details about the algorithms for deciding subsumption of concepts containing `(exists-area` *sr p*`)` and `(exists-so` *sr C*`)` are explained in [11].

The examples presented in this paper demonstrate that we cannot directly use relational modeling as known from database technology for representing spatial relations. Unintuitive models would not be detected if the semantics of space was not fully captured in the formalism and if implicit spatial relations were not derived by spatioterminological reasoning processes. Thus, description logics (and our extensions for representing spatial information) offer more than standard database representation systems (and GIS systems that rely on relational databases).

2.5 Operations on concept terms: Least Common Subsumer

Concept terms can be combined in various ways. For instance, the approach presented in [4] introduces the notion of a "least common subsumer" (LCS) to represent the commonalities between two instances which are described by respective concepts. The LCS operation is used to compute a new concept term representing the least common constraints of two input concept terms. Thus, a concept `C` is a least common subsumer of `D1` and `D2` iff `C` subsumes both `D1` and `D2` and there is no other common subsumer of `D1` and `D2` that is subsumed by `C` (see [4]). In the following, the LCS of the most common concept constructors is defined:

- `(lcs (and c11 ... c1k) (and c21 ... c2l))` = `(and (lcs c11 c21) ... (lcs c1k c2l))`
- `(lcs (all r1 c) (all r2 d))` = if `r1` = `r2` then `(all r1 (lcs c d))` else `classic-thing`
- `(lcs (at-least n r1) (at-least n r2))` =
  if `r1` = `r2` then `(at-least (min n m) r1)` else `classic-thing`
- `(lcs (at-most n r1) (at-most n r2))` =
  if `r1` = `r2` then `(at-most (max n m) r1)` else `classic-thing`

It should be noted that LCS is a semantic operation, i.e. it is not always possible to compute the LCS of a complex concept by pairwise considering its parts. For the `equal` construct, a complex graph matching operation has to be implemented for computing the LCS [4, 12].

In order to cope with our extensions to CLASSIC for representing spatial information, the LCS operator defined in [4] had to be extended. Basically, the LCS of so-called qualified existential restrictions had to be defined. These operators are not expressible in the CLASSIC language but are indirectly introduced in the semantics of the new operators introduced in the previous section. For instance, `(exists-so` R C `)` ensures that there exists an object which is subsumed by C in the (topological) relation R. In a similar way, the operator `exists-area` can be treated (the implicit relation is called `has-area`). In order to deal with `exists-area`, the LCS operation has to be defined for spatial relations and polygons, too. This operation can be defined in an application-dependent way. In our case we use a part-of hierarchy for declared polygons and select the “smallest” common superpart. For instance, `(lcs p2 p4)` is defined as `p5`.

- `(lcs (exists-so r c) (exists-so s d))` = `(exists-so (lcs r s) (lcs c d))`
- `(lcs (exists-area sr1 p) (exists-area sr2 q))` = `(exists-area (lcs sr1 sr2) (lcs p q))`

Please note again, that `LCS` is a semantic operation. Thus, before the `LCS` can be applied to complex conjunctions, corresponding spacenets have to be constructed and all implicit relations must be computed in each spacenet. Afterwards, the relations and predicates mentioned directly in the syntactic terms are properly restricted, corresponding `exists-area` and `exists-so` terms can be reconstructed and only then the `LCS` reduction rules can be applied (see [12] for details).

3 Semantics-based information retrieval with DLs

In this section we discuss how the theory presented in the previous section is used to build information retrieval applications such as the TV-Assistant shown in Figure 1. Let us assume that a certain user has selected a prototype film being presented in the program display and wants the system to retrieve other movies which are “related” to the prototype. Furthermore, the pinboard (see Figure 1) allows the user to collect films and use them as exemplar-based

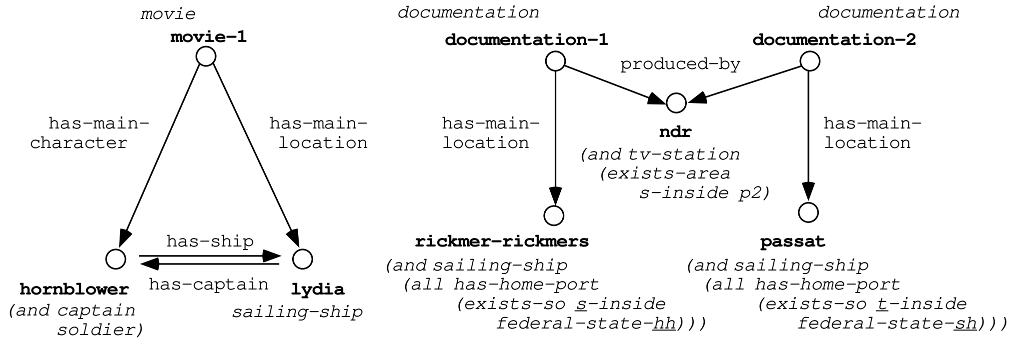


Figure 6: A small ABox about objects in our movie domain and their relations (shown as arrows with labels). Assertions about conceptual restrictions for individuals are attached to the corresponding nodes.

queries to find related films ("Find films like these."). In order to support these kinds of queries the commonalities of sets of films must be computed and "matches" against possible candidates in the program database must be found. In this section, the notion of "commonalities" and of "being related" will be formally defined with a logical semantics.

We assume that the domain model of our example application is extended with definitions for `movie`, `documentation` and `tv-station` as direct subconcepts of `classic-thing`. Some new concepts `sailing-ship` and `titanic` are defined as primitive subconcepts of `ship`.[5] In addition, we assume that `soldier` and `pirate` are declared as primitive subconcepts of `person`. Furthermore, the model is extended with concepts for movies. Important *attributes* for movies are `has-main-character` and `has-main-location`. The concepts `pirate-movie` and `titanic-movie` are defined as follows:

```
(define-concept pirate-movie
   (and movie
        (all has-main-character (and pirate captain))
        (all has-main-location sailing-ship)
        (equal has-main-character (compose has-main-location has-captain))))

(define-concept titanic-movie
   (and movie
        (all has-main-character captain)
        (all has-main-location titanic)
        (equal has-main-location (compose has-main-character has-ship))))
```

The structure shown in Figure 6 represents a small excerpt of the domain model for objects (ABox) in our TV-Assistant application. In the current version, this model is acquired by hand but in future versions the model might be automatically computed from background knowledge, textual descriptions and other database information provided directly by TV stations. Let us assume that our ABox is extended with a pirate movie (`movie-2`) and a Titanic movie (`movie-3`).

Let us further assume `movie-2` and `movie-3` have been placed on a TV-Assistant user's pin-board where they can be used for an exemplar-based query. The LCS operation is applied to the concept descriptions of both movies and returns the following concept:

[5]We model `titanic` as a concept rather than as an instance because, in films, different individual ships might be used during the shooting phases.

```
(and movie (equal (compose has-main-character has-ship) has-main-location)
     (all has-main-character captain) (all has-main-location ship))
```

We can see that an abstraction of the original movies has been computed. From `pirate-movie` the concept `sailing-ship` has been "reduced" to `ship` and from `titanic-movie` the concept `titanic` has been abstracted to `ship`, too. The `equal` construct is used to represent the commonalities concerning the main character and locations in the input concepts. Considering the different way of using attribute chain agreements in `pirate-movie` (focus is on the main character) and `titanic-movie` (focus is on the main location), it becomes clear that the LCS is not a simple syntactic operation but an operation that has to be carefully defined with respect to the semantics of the representation constructs of the modeling language. The resulting LCS concept presented above can be used as a query for retrieving instances from a database (ABox). In our simplified example from Figure 6 the movie `movie-1` is subsumed by the LCS concept and, therefore, it is returned as a query answer (possibly among others).

In some cases, the concept being returned by the LCS operation might be too specific because incidental commonalities are present between the input objects (or concepts). In the example presented in the right part of Figure 6, two other films are shown with their conceptual restrictions. Both are documentation films produced by a TV station in Hamburg (individual `ndr`). The left one presents information about a historical vessel in Hamburg, the other one deals with another historical vessel in Schleswig-Holstein (see also the map in Figure 4). If the LCS is applied to both documentation films which might be on the pinboard of some user, the following concept is returned (let us call it `LCS0`):

```
LCS0 = (and documentation
          (all produced-by (and tv-station (exists-area s-inside p5)))
            (all has-main-location
              (and sailing-ship (all has-home-port
                                    (exists-so inside (exists-area equal p5))))))
```

Both films are documentation films produced by a TV station located (strictly) inside Northern Germany. The main location the films talk about is a sailing ship whose home port is inside Northern Germany, too. If this concept is used as a query, possibly no documentation might be found that is produced by a Northern German (`p5`) TV station. However, in this case the conceptual restrictions for the TV station that produced the film are likely to be irrelevant for the user of the TV-Assistant. Thus, we need a mechanism for systematically generalizing LCS results (i.e. database queries). This is where terminological background knowledge comes into play. The domain model for the TV-Assistant contains several "strategic" concept definitions which represent the most frequent commonalities encountered in the TV domain. The idea is to use the LCS first to identify one of these predefined commonalities and then use this commonality for database (or ABox) retrievals. For instance, in our domain model (TBox) there is a declaration for a Northern German ship documentation film:

```
(define-concept northern-german-ship-documentation-film
  (and documentation
    (all has-main-location
      (and ship (all has-home-port (exists-so g-inside northern-german-region))))))
```

The concept `LCS0` is inserted into the TBox. In this process the implicit subsumption relationships are automatically detected. Thus, `northern-german-ship-documentation-film` is computed as a subsumer of `LCS0`. If no instances are found for `LCS0`, the most specific (named) superconcepts are used as queries and so forth. The role of terminological knowledge is important here. Predefined anchor points for commonalities can be automatically found.

In the brief examples presented in this section we have neglected any information about broadcasting times. Relying on standard database techniques, a "time window" can be used as an additional filter for candidates, of course. Other attributes that relate films: e.g. "recorded with Technicolor" can be handled as well. However, even the simplified examples clearly demonstrate the potential of semantics-based information retrieval. We have seen that the description logic CLASSIC, extended with constructs for spatial reasoning, can be used as an adequate representation formalism for information systems. Detecting implicit subsumption relationships (one of the main inference services) is very important. Let us assume that applying the LCS returns the concept expression used in the definition of `hh-district-touching-sh`, i.e. the concept is not already present in the TBox but computed on the fly. If we insert this concept into the TBox, the subsumption relation to `hh-border-district` must be found in order to actually find films which are related to the specific query concept. Sound and complete inference algorithms are absolutely necessary in this case.

4 Summary and conclusion

Information systems are one of the most important applications of database theory. The large amount of literature in this area cannot be cited here. In this paper we have demonstrated the potential of description logics for information access (see also [3]). The examples show that DL theory extends the expressive power and reasoning facilities offered by current database systems. In fact, description logic theory and database theory are now converging.

The theory presented in this paper has been evaluated by implementing a working prototype of a Web-based TV-Assistant which offers novel search facilities and interaction techniques that can be handled even by non-experts (exemplar-based queries can be easily composed with VRML-based Perspective Wall interaction techniques).

Applications of description logics for information retrieval (and CLASSIC in particular) are also reported in [10] and [14]. We have contributed to extending DL theory by increasing the expressive power of DLs concerning reasoning about space (see [7, 11]). The LCS operation initially introduced in [4] has been extended in order to adequately deal with the spatial representation requirements for the TV-Assistant application. Thus, the theory presented in this paper works in practice. However, the costs for modeling the required domain knowledge must not be neglected.

We are considering extensions for user-adaptive query narrowing with default logic (for LCS results that are too unspecific). The technique is also applied in [8] but, in our case, the theory has to be extended in order to cope with the extensions for spatial reasoning that we have developed. An important extension to the TV-Assistant we are currently investigating is a systematic way of modeling plot structures for film genres and specific films. In [6] we have presented the theory for a more expressive description logic that can be used to adequately represent plot schemes and to reason about commonalities in the temporal structures of related films (e.g. films with happy-end).

Acknowledgments

We would like to thank the members of the BAND project from Lavielle GmbH and the AI Laboratory: Kay Hidde, Rainer Joswig and Thomas Mantay. Furthermore, we also acknowledge the work of our students Irena Kortas, Carsten Lutz, Claas Prien and Michael Wessel.

References

[1] Borgida, A., Brachman, R.J., McGuiness, D.L., Resnick, L.A., CLASSIC: A Structural Data Model for Objects, In: Proc. of the 1989 ACM SIGMOD International Conference on Management of Data, Portland, Oregon, May-June, 1989.

[2] Borgida, A., Patel-Schneider, P.F., A Semantics and Complete Algorithm for Subsumption in the CLASSIC description logic, Journal of Artificial Intelligence Research, No. 1, Morgan Kaufmann Publ., 1994, pages 277–308.

[3] Bresciani, P., Franconi, E., Description Logics for Information Access. AI*IA 1996 Workshop on Access, Extraction and Integration of Knowledge, September 1996.

[4] Cohen, W.W., Borgida, A., Hirsh, H., Computing Least Common Subsumers in Description Logics, In: Proc. AAAI-92, AAAI Press/The MIT Press, 1992, pages 754–760.

[5] Egenhofer, M.J., Reasoning about Binary Topological Relations. In: O. Günther and H.-J. Schek, editors, Advances in Spatial Databases, Second Symposium, SSD'91, Zurich, Aug. 28-30, 1991, volume 525 of Lecture Notes in Computer Science, pages 143–160. Springer-Verlag, Berlin, August 1991.

[6] Haarslev, V., Lutz, C., Möller, R., Foundations of Spatioterminological Reasoning with Description Logics. In: Principles of Knowledge Representation and Reasoning: Proc. of the Sixth International Conference (KR'98), A.G. Cohn, L.K. Schubert, S.C.Shapiro, editors, Morgan-Kaufmann Publishers, 1998.

[7] Haarslev, V., Möller, R., Spatioterminological Reasoning: Subsumption Based On Geometric Reasoning. In: Rousset et al., editors, 11th International Workshop on Description Logics DL'97, Gif sur Yvette, Universite Paris Sud, Laboratoire de Recherche en Informatique (LRI), CNRS, 1997., pages 74–76, Sept. 27–29.

[8] Lambrix, P., Shahmehri, N., Wahlöf, N., A Default Extension to Description Logics for Use in an Intelligent Search Engine, Proc. of the 1st Hawaiian Int. Conf. on System Science, 1998.

[9] Mackinlay, J.D., Robertson, G.G., Card, S.K., The Perspective Wall: Detail and Context Smoothly Integrated, In: Proc. CHI'91, Computer-Human Interaction, ACM-Press 1991, pages 173–179.

[10] McGuinness, D.L., Manning, H., Beattie, T., Knowledge Assisted Search. In: Proc. of the International Joint Conference on Artificial Intelligence Workshop on The Future of AI and the Internet, Nagoya, Japan, August, 1997.

[11] Möller, R., Haarslev, V., Lutz, C., Spatioterminological Reasoning Based on Geometric Inferences: The $\mathcal{ALCRP}(\mathcal{D})$ Approach. Technical Report FBI-HH-M-277/97, University of Hamburg, Computer Science Department, 1997.

[12] Prien C.P., Application of the Least-Common Subsumer Method for Determining Similarities in Information Retrieval Systems (in German), Diploma Thesis, forthcoming, 1998.

[13] Randell, D., Cui, Z., and Cohn, A., A Spatial Logic Based on Regions and Connections. In: B. Nebel, C. Rich, and W. Swartout (Eds.), Principles of Knowledge Representation and Reasoning, Cambridge, Mass., Oct. 25-29, 1992, pages 165–176.

[14] Salotti, S., Ventos, V., Study and Formalization of a Case-Based Reasoning System with a Description Logic. In: Rousset et al., editors, 11th International Workshop on Description Logics DL'97, Gif sur Yvette, Universite Paris Sud, Laboratoire de Recherche en Informatique (LRI), CNRS, 1997., pages 114–118, Sept. 27–29.

[15] Woods, W.A., Schmolze, J.G., The KL-ONE Family, In: Semantic Networks in Artificial Intelligence, Lehmann, F. (Ed.), Pergamon Press, 1992, pages 133–177.