Published in: Bernhard Nebel and Leonie Dreschler-Fischer (eds.), *KI-94: Advances in Artificial Intelligence – Proc. 18th German Annual Conference on Artificial Intelligence*, Saarbrücken, Sept. 18–23, 1994, volume 861 of *Lecture Notes in Artificial Intelligence*, pages 142–153. Springer-Verlag, Berlin – Heidelberg – New York, 1994.

Combining Spatial and Terminological Reasoning

Volker Haarslev, Ralf Möller, and Carsten Schröder

University of Hamburg, Computer Science Department,
Vogt-Kölln-Str. 30, D-22527 Hamburg, Germany
{haarslev,moeller,schroed}@informatik.uni-hamburg.de

Abstract The paper presents a method for terminological reasoning about spatial objects on the basis of a KL-ONE-like framework (LOOM). We apply this method to the domain of deductive geographic information systems and parsing of visual languages. In contrast to existing work, which mainly focusses on reasoning about qualitative spatial relations, we integrate quantitative information with conceptual or terminological reasoning by the use of "generative" qualitative relations. These relations allow a modularization of systems for terminological reasoning and domain-specific storage and indexing of, e.g., spatial data. Qualitative relations are computed on demand from quantitative data during forward-chaining assertional reasoning.

1 Introduction

A lot of inference processes of knowledge-based systems are based on different kinds of spatial reasoning. In this paper we present an inference scheme which combines terminological reasoning with inferences about spatial data. This scheme is useful for interpreting spatial data in different application domains. In terms of terminological reasoning an *interpretation* is defined as the most specialized classification of the objects of the domain. Here, classification of spatial objects depends on the specific relations found in a concrete spatial "constellation".

1.1 Spatio-Terminological Inferences

Current research about spatial reasoning mostly concentrates on inference processes about qualitative spatial relations and how they can be combined to model spatial reasoning [10, 22]. Calculi for qualitative relations are proposed to represent intrinsic properties of space (like neighborhood). Two-level representations have been proposed to integrate logical representations for qualitative spatial relations (like upon, over, above) and coordinate-oriented, i.e., quantitative information [6, 20].

However, since qualitative relations are considered as the basis for reasoning processes, in current proposals there is no adequate transition from quantitative information to terminological or conceptual reasoning via, e.g., qualitative relations. Thus, terminological reasoning is not integrated with spatial reasoning in a well-formalized way. Besides concept classification (in the TBox), we are especially interested in using spatial information during forward-chaining object classification processes (ABox reasoning). We would like to propose the term "spatio-terminological inferences" for a three-level view of inference processes combining quantitative, qualitative and conceptual representations.

1.2 Overview

In order to demonstrate how spatial reasoning can be efficiently combined with terminological reasoning, Sect. 2 presents two examples from two different domains: image interpretation and parsing of visual programming languages. Section 3 compares our appoach with a proposal for combining propositional and analogical representations and the work concerning deductive databases for Geographic Information Systems. After discussing open problems we conclude with a summary.

2 Spatio-Terminological Reasoning for Interpretation Tasks

In the first example, which deals with aerial image interpretation, object classification is used to model recognition of meaningful "constellations" of concrete spatial objects. With this example we discuss what patterns of inferences are useful for modeling spatio-terminological reasoning. The second example extends this work and uses a complete set of spatial relations for declaratively specifying knowledge for parsing "constellations" of graphical objects (rectangles, lines, etc.) as part of a visual programming language.

In our examples, the basic representation of spatial objects is quantitative in nature. For simplification purposes we use a two-dimensional representation where objects are represented by bounding boxes. Furthermore, we assume that these spatial data are stored in a special database that provides adequate indexing mechanisms for the retrieval of geometric objects (e.g., R-Trees [11], see also [2]). These databases are called spatial databases from now on. Spatial databases have been developed for Geographic Information Systems (GIS). Similar systems have also been implemented to support spatial reasoning for diagnostic purposes [23]. However, none of these systems implements models that combine spatial reasoning with terminological reasoning. The examples in this paper show how this can be exploited for interpretation tasks.

Each of the examples we would like to discuss introduces problems that have to be dealt with when terminological reasoning is to be used in practical applications:

- How can the assertional knowledge base (ABox) be coupled with a spatial database?
- How to deal with a huge number of spatial objects?
- How to control the computation of qualitative spatial relations during forward-chaining?

2.1 Image Interpretation by Object Classification

In the context of an aerial image interpretation system we assume that there exists a two-dimensional model of a geographic scene stored in a GIS. In this system multi-spectral images are interpreted, for example, in order to detect changes in the scene. To speak of an *interpretation* of an image, e.g. a change has to be described not only at the geometrical level but also at the conceptual level. For a description of an aerial image interpretation project where reconstructions of, e.g., airports are to be recognized see [8].

We use the terminological knowledge representation system LOOM (actually LOOM 2.1, [19]) to represent a model of an example world. Types of spatial objects are

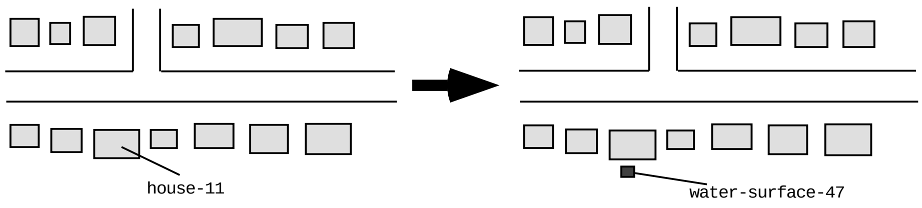

Figure1. A configuration of houses in a village.

represented by concepts using the TBox of LOOM. For illustration purposes we consider a small example where houses, villas, water surfaces and swimming pools are represented:[1]

$$
\begin{aligned}
\texttt{spatial-object} &\sqsubseteq_C \top \\
\texttt{spatial-relation} &\sqsubseteq_R \texttt{spatial-object} \times \texttt{spatial-object} \\
\texttt{water-surface} &\sqsubseteq_C \texttt{spatial-object} \\
\texttt{house} &\sqsubseteq_C \texttt{spatial-object} \\
\texttt{swimming-pool} &\doteq_C \texttt{water-surface} \sqcap (\geq 1\, \texttt{near} : \texttt{house}) \\
\texttt{villa} &\doteq_C \texttt{house} \sqcap (\geq 1\, \texttt{near} : \texttt{swimming-pool})
\end{aligned}
$$

In our application, concepts like `house` and `water-surface` can be adequately defined as primitive concepts, because we assume that a rectangle can definitely be asserted to be a water surface or a house. Our example presupposes that the underlying image interpretation process is not only able to detect rectangular areas but will also determine the material of any object found in an image (e.g., by using spectral analysis).

The definitions can be paraphrased using natural language. A `swimming-pool` is a `water-surface` with at least one `house` in the neighborhood (relation `near`). A `villa` is a special `house` with at least one `swimming-pool` in the neighborhood. The definitions of `villa` and `swimming-pool` use *defined* relations (i.e., relations with sufficient as well as necessary conditions) with value and range restrictions for `near`.

Figure 1 provides a sketch of a village where an initial scene with several houses (light-gray rectangles) is shown in the left part. The objects are assumed to be asserted using LOOM's tell facility. Furthermore, we assume that houses are stored in a spatial database with appropriately defined coordinates.

When interpreting an image of this scene, a new object might be found in the neigborhood of a certain `house` (`house-11`). The image interpretation process detects a water surface (`water-surface-47`, small dark-gray rectangle in the right part of Fig. 1) which is then asserted as an instance in LOOM's ABox. For "interpreting" the image the following deductions are needed. When a `water-surface` is found near a `house`, this `water-surface` will be considered as a `swimming-pool`.[2] However, when

[1] Whenever possible, we use the common abstract syntax instead of the concrete LOOM syntax. See [3] for a definition of the syntax and semantics of the complete abstract language.

[2] In a full-size knowledge-base there might also be a garden pond concept, but this is another topic.

it is known to be a `swimming-pool`, the `house` will have to be classified (or recognized) as a `villa`.

Note, that we are interested in the most specialized classification of any object visible in the scene, so the object classification must be performed in a forward-chaining manner. However, the recognition of every relation holding between any pair of objects might not necessarily be of interest. It therefore suffices to compute them in a backward-chaining way only when needed for classifying an object.

How can this simple pattern of inference be modeled using a KL-ONE system and what extensions are advantageous? Consider the following declaration of `near` as a defined relation:[3]

```
(defrelation near
  :is (:and spatial-relation (:satisfies (?x ?y) ...)))
```

According to our definition of `spatial-relation`, `near` can only hold between spatial objects.

There are several problems with this declaration. First, the definition of `near` requires every pair of spatial objects to be explicitly declared as a member of `spatial-relation` because `spatial-relation` is primitive. Second, in order to determine for a given object whether there exists any other object in the relation `near` to it, a standard ABox inference system (like LOOM) would have to check the predicate specified in the `:satisfies`-form for every other spatial object. In a real application there will be far too many objects for this complex operation to be efficient. Even worse, when forward-chaining is not only used for object classification but for the recognition of defined relations as well, this has to be done for each pair of spatial objects.

However, in our example application, spatial indexing mechanisms can be used which provide quick access to the objects in the neighborhood of, for instance, `water-surface-47`. It would be unfortunate if the services of a spatial database could not be made available to the reasoning system.

We propose the following view. The spatial database can be interpreted as a surrogate for a set of ABox terms (or assertions) for spatial relations. In LOOM terminology, this can be paraphrased as a set of implicit assertions like `(tell (near water-surface-47 house-11))`. Thus, during the forward-chaining process of object classification we need a mechanism that accesses these statements on demand and makes them available to the ABox inference processes. Or, put in another way, while considering object specialization possibilities, the ABox reasoner must be able to use an efficient *candidate generator*.

The necessary integration of the ABox reasoning mechanisms with the functionality of a spatial database can be achieved by a special LOOM feature called *functional relation*. The following definition shows the use of a function as a generator for the tuples of the relation `near`.

```
(defrelation near
  :function ((x) (compute-nearby-objects *spatial-database* x))
  :characteristics (:multiple-values :symmetric))
```

[3] The `:satisfies`-form allows a concept to be defined using a query formula.

The function is used as follows. LOOM specializes an instance if there still exists a *defined* subconcept and the sufficient conditions for this subconcept can be proven to hold (process of forward-chaining). Since we defined `swimming-pool` as a subconcept of `water-surface` with at least one house in the `near`-relation, the ABox inference engine tries to find a corresponding tuple of `near` with `water-surface-47` in the domain. The function `compute-nearby-objects` will be evaluated.[4] In our example from Fig. 1 the function returns `house-11`. This `house` can, in turn, be specialized when there is a `swimming-pool` found to be in the `near`-relation. Thus, `compute-nearby-objects` is evaluated again,[5] now with `house-11` as an input parameter. In this case, `swimming-pool-47` is returned and `house-11` is specialized to a `villa`.

This example shows that spatio-terminological domain-level inferences can be carried out using the definitions of a terminological knowledge base. Furthermore, qualitative spatial relations can be computed on demand and with access to efficient storage and computation systems. If generative relations (via functions) were not available in terminological description languages, we had to define relations like `near` as given in our first `defrelation`-form. During forward-chaining object classification, e.g. of `water-surface-47`, the ABox inferencer will have to use backward-chaining to prove whether a corresponding object is found to exist in the range of the `near`-relation. Thus, `near` had to be a *defined* relation using the query-defining term `:satisfies`. The disadvantage is that the ABox inferencer can hardly exploit domain knowledge to preselect candidates. In the worst case the inferencer tries to prove whether the predicate `near` holds for every spatial object in the knowledge base.

In this example we used the relation `near`. The same problems occur with other spatial relations. The next section introduces the definition of other qualitative spatial relations and demonstrates how the same pattern of inference can be applied to other domains.

2.2 Visual Parsing by Object Classification

This section presents the application of spatio-terminological reasoning to parsing of visual programming languages. These languages are represented by (at least 2D) graphics instead of text. Syntax and semantics of "pure" visual languages are mostly expressed by pictorial relationships between (graphical) language elements[6]. Programs expressed in a visual language are graphically represented and are defined as meaningful "constellations" of *abstract* spatial objects. Therefore, we consider parsing of visual languages as an image interpretation or object classification process.

Pictorial Janus as Example Domain. The example described in this section has been fully implemented using LOOM. It is based on a full treatment of the visual programming language Pictorial Janus (PJ) [16]. PJ is a language for the domain of flat guarded horn

[4] In this paper we do not consider the details of this function.

[5] Since `near` is declared to be symmetric a cache might also be used. Note however, that a linguistic analysis reveals that in general the relation `near` is not symmetric [21].

[6] Also simply referred to as *objects*.

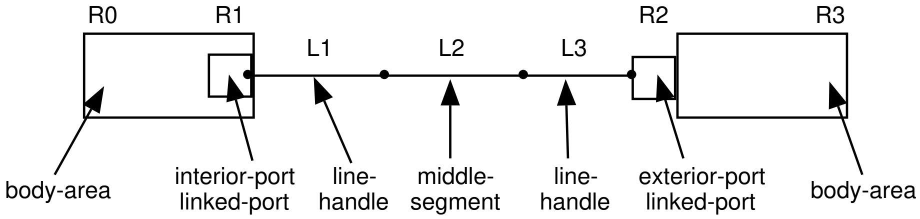

Figure2. Linking situation in a Pictorial Janus program.

clauses. PJ's syntax and semantics are defined through topological relations which have to hold between language elements. Nearly all basic language elements (agents, rules, ports, primitive functions, constraints, constants, arrays, bags) are represented as closed contours, others are represented as (directed) lines (links, channels, call arrows). We refer to Haarslev [12] for more details about this approach.

In the following we assume the existence of a spatial database storing information about elements of visual programs. Programs might be entered into the database by using appropriate techniques (e.g. via graphic editors, by scanning (hand)drawings, etc.) For sake of simplicity we ignore any low-level vision processes and assume an object-oriented representation based on rectangles, straight line segments, and arrows. Figure 2 gives an example for a typical subpart of PJ programs. This subpart consists of four rectangles `R0`, ..., `R3` and three line segments `L1`, ..., `L3`. The rectangles `R1` and `R2` are connected via a chain of line segments.

A Small Knowledge Base. We present a simplified subpart of a knowledge base developed for Pictorial Janus. A lot of (more complex) concept, relation and rule definitions are left out for sake of simplicity. The concept definitions make use of primitive generative relations that model qualitative, topological relationships between language elements. Due to space limitations, however, we left out many explanations for number and value restrictions of roles.

We define two concepts `line-segment` and `area` as a `spatial-object` with dimension 1 or 2, respectively. `Dim` is a primitive generative relation which associates spatial objects with their geometric dimension.

```
(defrelation dim
  :function ((x) (compute-dimension-of x)))
```

$$\texttt{line-segment} \doteq_C \texttt{spatial-object} \sqcap (\texttt{dim} : 1)$$
$$\texttt{area} \doteq_C \texttt{spatial-object} \sqcap (\texttt{dim} : 2)$$

Areas—either classified as *port* or *body-area*—are basic building blocks of PJ programs. Ports are used for denoting data-item handles, for connecting list elements, and as arguments of other entities. A body area is used to represent PJ language elements such as messages, rules, and agents.

$$\begin{aligned}
\texttt{empty-area} &\doteq_C \texttt{area} \sqcap (\leq 0\,\texttt{containing}) \sqcap (\leq 0\,\texttt{covering})\\
\texttt{port} &\doteq_C \texttt{empty-area} \sqcap (\leq 3\,\texttt{touching}) \sqcap (\leq 1\,\texttt{touching} : \texttt{area}) \sqcap\\
&\quad (\leq 2\,\texttt{touching} : \texttt{line-segment})\\
\texttt{body-area} &\doteq_C \texttt{area} \sqcap (\leq 0\,\texttt{covered-by}) \sqcap (\forall\,\texttt{touching} : \texttt{empty-area}) \sqcap\\
&\quad (\leq 0\,\texttt{touching} : \texttt{line-segment})
\end{aligned}$$

A port can be further classified as *interior* (`R1`) or *exterior* (`R2`), *empty* or *linked* (`R1`,`R2`). An interior port (`R1`) denotes the object (`R0`) that covers this port. Ports usually serve as "docking place" for connecting lines. See Fig. 2 for the examples.

$$\begin{aligned}
\texttt{interior-port} &\doteq_C \texttt{port} \sqcap (1\,\texttt{covered-by}) \sqcap (\forall\,\texttt{covered-by} : \texttt{body-area}) \sqcap\\
&\quad (\leq 1\,\texttt{touching}) \sqcap (\exists\,\texttt{touching} : \texttt{line-handle}) \sqcap\\
&\quad (\leq 1\,\texttt{touching} : \texttt{segment}) \sqcap (\leq 0\,\texttt{touching} : \texttt{area})\\
\texttt{exterior-port} &\doteq_C \texttt{port} \sqcap (\exists\,\texttt{touching} : \texttt{body-area}) \sqcap (\leq 0\,\texttt{covered-by})\\
\texttt{empty-port} &\doteq_C \texttt{port} \sqcap (1\,\texttt{touching} : \texttt{body-area}) \sqcap (\leq 0\,\texttt{touching} : \texttt{segment})\\
\texttt{linked-port} &\doteq_C \texttt{port} \sqcap (\exists\,\texttt{touching} : \texttt{line-handle})
\end{aligned}$$

A `true-segment` is defined as a `line-segment` which satisfies some restrictions on the number of objects `touching` it. The concepts `end-segment` and `middle-segment` are specializations of `true-segment` and specify corresponding geometric situations.

$$\begin{aligned}
\texttt{true-segment} &\doteq_C \texttt{line-segment} \sqcap (\geq 1\,\texttt{touching}) \sqcap (\leq 4\,\texttt{touching})\\
\texttt{end-segment} &\doteq_C \texttt{true-segment} \sqcap (\leq 3\,\texttt{touching}) \sqcap\\
&\quad (\leq 1\,\texttt{touching} : \texttt{line-segment}) \sqcap (\geq 1\,\texttt{touching} : (\texttt{dim} : \{1,2\}))\\
\texttt{middle-segment} &\doteq_C \texttt{true-segment} \sqcap (\forall\,\texttt{touching} : (\texttt{dim} : \{0,1\})) \sqcap\\
&\quad (2\,\texttt{touching} : \texttt{line-segment})
\end{aligned}$$

`Touching` is defined as a primitive generative relation which associates touching objects. The other relations `crossing`, `covering`, `covered-by` are analogously defined.

```
(defrelation touching
  :function ((x) (compute-touching-of x))
  :characteristics :multiple-values)
```

`Touching` as well as the other basic topological relations are defined as generative relations in the same way as `near` in the introductory example. The formal definition of these relations is based on a proposal by Clementini et al. [7]. The advantage of their approach compared to similar proposals (e.g. see [9]) is its ability to deal with intersections of lines *and* regions. It is also important to note that their relations are complete and mutually exclusive.

There are several constraints on the types of geometric objects: Areas have to be convex, connected and without holes, lines must not be self-intersecting, are either circular or directed, and have exactly two end points.

The possible relationships are defined by the dimension of intersections between mathematical point-sets representing the geometric objects mentioned above. Every object is composed of a *boundary* and an *interior*. The boundary of a region is a circular line. The boundary of a line is either an empty point-set (circular line) or a point-set consisting of two end points (non-circular line), the boundary of a point is an empty point-set. The interior of an object is the object without its boundary. In case of circular lines and points the interior is identical to the object itself.

Using these definitions six binary topological relations can be defined (see also [7]). The boundary of λ_n is formally denoted by $\partial\lambda_n$, its interior by λ_n^o.

- **touching:** $touching(\lambda_1, \lambda_2) \Leftrightarrow (\lambda_1 \cap \lambda_2 \neq \emptyset) \wedge (\lambda_1^o \cap \lambda_2^o = \emptyset)$
  Only the boundaries are intersecting; touching is symmetric and applies to every situation except point/point.
- **overlapping:** $overlapping(\lambda_1, \lambda_2) \Leftrightarrow (\lambda_1 \cap \lambda_2 \neq \lambda_1) \wedge (\lambda_1 \cap \lambda_2 \neq \lambda_2) \wedge (\dim(\lambda_1^o \cap \lambda_2^o) = \dim(\lambda_1^o) = \dim(\lambda_2^o))$
  The intersection is either a line or a point which has to be different to both objects; overlapping is symmetric and applies only to area/area and line/line situations.
- **crossing:** $crossing(\lambda_1, \lambda_2) \Leftrightarrow (\lambda_1 \cap \lambda_2 \neq \lambda_1) \wedge (\lambda_1 \cap \lambda_2 \neq \lambda_2) \wedge \dim(\lambda_1^o \cap \lambda_2^o) = (\max(\dim(\lambda_1^o), \dim(\lambda_2^o)) - 1)$
  Two lines are crossing if their intersection is an internal point. A line crosses a region if the line is partly inside and outside of this region; crossing is symmetric and applies only to line/line and line/area situations.
- **containing/inside:** $containing(\lambda_1, \lambda_2) \Leftrightarrow (\lambda_1 \cap \lambda_2 = \lambda_2) \wedge (\lambda_1^o \cap \lambda_2^o \neq \emptyset)$
  An object λ_1 contains an object λ_2 if the intersection between λ_1's and λ_2's regions is equal to λ_2 and the interiors of their regions intersect; the inverse containing is inside. They are transitive and apply to every situation.
- **equal:** $equal(\lambda_1, \lambda_2) \Leftrightarrow \lambda_1 \cap \lambda_2 = \lambda_1 = \lambda_2$
  The intersection is equal to both objects; equal is symmetric, transitive and applies to every situation.
- **disjoint:** $disjoint(\lambda_1, \lambda_2) \Leftrightarrow \lambda_1 \cap \lambda_2 = \emptyset$
  The intersection is empty; disjoint is symmetric and applies to every situation.

With respect to our application domain, we defined a seventh relation which is a specialization of 'containing' and 'inside':

- **covering/covered-by:** $covering(\lambda_1, \lambda_2) \Leftrightarrow containing(\lambda_1, \lambda_2) \wedge \dim(\partial\lambda_1 \cap \partial\lambda_2) = \dim(\partial\lambda_2)$
  An object λ_1 covers an object λ_2 if λ_1's region contains λ_2's region and the intersection of their boundaries has a dimension equal to the dimension of λ_2's boundary; the inverse of covering is covered-by. They apply to every situation except point/point.

These seven relations may hold between geometric objects *and* their boundary and interior. Additionally, we defined a relation *dimension* (also called dim) which applies to any object.

Parsing PJ Programs. Rectangles and line segments as presented in Fig. 2 define the input to the assertional reasoning process. For example, as the result of this reasoning, a rectangle like `R1` is specialized to an `interior-port` since it is an `empty-area` and can be proven to be in the `covered-by`-relation to `R0` which, in turn, is a `body-region`.

Similar deductions will be performed for other graphical objects. Figure 2 illustrates the complete result of this reasoning process (denoted by arrows).

2.3 Summary of the Examples

The last example shows how spatio-terminological reasoning can be applied to parsing problems. A complete set of topological spatial relations together with appropriately defined concepts is used for interpreting constellations of geometrical objects. A subset of visual languages is completely specifiable by terminological definitions, and parsing is reduced to assertional reasoning. Thus, spatio-terminological reasoning is applicable not only to toy problems but scales up to be the basic reasoning mechanism for various applications.

3 Related Work

In the AI literature representations with domain-specific indexing mechanisms are also discussed in the context of "analogical" representations. In the project "LILOG" there have been attempts to couple a propositional representation and reasoning system with a special system for representing spatial information ("depictional system", [13, 5]) using cell matrices. The depictional system supports special "imagination" and "inspection" processes in the sense of Kosslyn [18]. The propositional representation system of LILOG integrates some aspects of terminological description languages with a theorem prover for a sorted logic. The connection to the depictional (non-logical) component is done by explicit switching statements declared inside of LILOG rules (see [17], p. 38). In LILOG, the spatial (depictional) representation system is not used for object classfication.

Recently, research in spatial databases has concentrated on providing databases with deductive capabilities. Especially, work in geographic data handling proposes the use of extended Prolog systems to describe domain knowledge on the basis of primitive spatial relations [1]. In our group, we used CLP(R) [15] to model spatial domain knowledge using Horn clauses and constraints. The results were that adequate indexing mechanisms for spatial data beyond the standard Prolog (and CLP(R)) resolution mechanisms are necessary due to severe performance problems. However, the main problem with Prolog-like backward-chaining systems is that in order to drive inference processes, goals have to be set up. Therefore, in our first image interpretation example, we have to directly ask whether a given house is a villa. Unfortunately, in an image interpretation domain the "right" questions (or queries) are almost never known in advance. In addition to object classification by forward-chaining, terminological description systems (like LOOM) also support backward-chaining inferences as a by-product.

4 Open Problems and Future Work

In the examples of Sect. 2 we have shown how a generator function using a proper indexing mechanism can be used for classifying objects in a forward-chaining manner based on quantitative spatial information. However, this approach has some problems with the revision of information.

4.1 Reason Maintenance

What happens when the quantitative spatial information is revised? Consider the case were `water-surface-47` near `house-11` is removed. Then, the condition (≥ 1 `near` : `swimming-pool`) for `house-11` being a `villa` is no longer fulfilled, and the classification as a `villa` must be retracted. Unfortunately, by using a function generating the tuples of the `near`-relation in an extra-logical way, the ABox reasoning mechanism has no information about the dependencies and, therefore, an automatic retraction is not possible. In order to allow for reason maintenance of the `near`-relation a mechanism must be provided for declaring the information the relation depends on.

What is the information a relation tuple depends on? In our case the `near`-relation of a `house` and a `water-surface` depends on the existence of both the `house` and the `water-surface` and on their respective location. In general, all the ABox-objects and all the relation-tuples used in a generator function must be "marked" to allow for a correct reason maintenance.

However, the use of generator functions in combination with a means for marking the information a relation depends on, has one major disadvantage: the user is responsible for consistently and correctly using these mechanisms. More desirable would be an intra-logical, declarative construct for defining spatial relations were the information needed for reason maintenance could be acquired automatically. This would require a means for declaring *defined* relations and powerful, expressive constructs for declaring spatial contraints. Recently, Baader & Hanschke [4, 14] proposed a very general scheme for integrating special domains with their own reasoning mechanisms into a description logic. However, their language $\mathcal{ALCFP}(\mathcal{D})$ does not allow for the declaration of defined roles and, therefore, cannot be directly used for our purposes. Whether this approach can be appropriately extended, has to be explored in the future.

4.2 Intrinsic Properties of the Domain

We have shown how terminological inferences can be drawn on the basis of certain spatial relations and concepts. Other inferences, however, most notably those which are based on intrinsic properties of space, are still missing. For example, non-penetrable objects cannot overlap in space. This fact can be used for ABox- as well as TBox-reasoning. Trying to assert two different objects located at the same place must result in an inconsistency, and a concept requiring at least 2 different spatial objects at overlapping locations must be detected as being inconsistent.

For another example, assume that the spatial region under consideration in an application is completely covered or filled by non-penetrable objects. This might give rise to another line of reasoning: asserting an object as being a `villa` and assuming there is only one object `near` to it which is classified as a `spatial-object`, then this `spatial-object` would have to be specialized to a `swimming-pool`.

For these kinds of reasoning the intrinsic properties of the domain must be axiomatisized in a description logic. When the reasoning services of a real system are based on a consistency or satisfiability test as proposed by Schmidt-Schauß & Smolka [24] this might be done by extending the rule sets of the satisfiability test by certain rules realizing the spatial axioms. Again, this has to be explored by future work.

5 Conclusion

Our proposal shows that non-logical representations can directly and consistently be combined with the ABox resoning services of a KL-ONE-like description logic. The examples indicate that for an interesting subset of problem domains this hybrid combination is sufficient to realize the necessary inference processes. We discussed an augmentation of the ABox of LOOM with a system that provides efficient access functions for spatial data. We also showed that (distributed) *explicit* switching statements (placed in rules like in LILOG) are not necessary if we use the concept of functional or generative relations. In our proposal the services of the external database system can even be described in terms of ABox (`tell`) statements. Furthermore, for our applications, the discussion of a canonical analogical representation system (in the context of the "imagery debate") is not important. Our point of view is that we want to provide the most suitable representation for application-specific computational processes. Terminological description languages are suitable for deductive reasoning involving classification while, e.g., R-Trees are suitable for efficiently accessing spatial data. Both systems are coupled in a well-formalized way. However, our current approach requires the reasoning to be monotonous.

Acknowledgements

We thank our colleague Ralf Röhrig for thoughtful comments and interesting discussions.

References

1. A. I. Abdelmoty, M. H. Williams, and N. W. Paton. Deduction and Deductive Databases for Geographic Data Handling. In Abel and Ooi [2], pages 443–464.
2. D. Abel and B. C. Ooi, editors. *Advances in Spatial Databases*, Singapore, June 23–25 1993, volume 692 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin – Heidelberg – New York, 1993.
3. F. Baader et al. Terminological Knowledge Representation: A Proposal for a Terminological Logic. In Bernhard Nebel, Christof Peltason, and Kai von Luck, editors, *International Workshop on Terminological Logics*, Schloß Dagstuhl, Germany, May 6–8, 1991, KIT-Report 89, pages 120–128. TU Berlin, Fachbereich Informatik.
4. F. Baader and Ph. Hanschke. A Scheme for Integrating Concrete Domains into Concept Languages. In John Mylopoulos and Ray Reiter, editors, *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence IJCAI-91*, Sydney, Australia, August 24–30, 1991, pages 452–457. Morgan Kaufmann Publ. Inc., San Mateo, CA, 1991. Extended version published as DFKI Research Report RR-91-10, Kaiserslautern.
5. T. Bollinger and U. Pletat. The LILOG Knowledge Representation System. IWBS Report 156, IBM Wissenschaftliches Zentrum, Institut für Wissensbasierte Systeme, Stuttgart, 1991.
6. P. Breuer and J. Müller. A Two Level Representation for Spatial Relations, Part I. DFKI Research Report RR-91-14, Deutsches Forschungszentrum für Künstliche Intelligenz, Kaiserslautern, July 1991.
7. E. Clementini, P. Di Felice, and P. van Oesterom. A Small Set of Formal Topological Relationships Suitable for End-User Interaction. In Abel and Ooi [2], pages 277–295.

8. L. Dreschler-Fischer, Ch. Drewniok, H. Lange, and C. Schröder. A Knowledge-Based Approach to the Detection and Interpretation of Changes in Aerial Images. In Sadao Fujimura, editor, *1993 International Geoscience and Remote Sensing Symposium (IGARSS'93)*, Tokyo, August 1993, volume I, pages 159–161. Institute of Electrical and Electronics Engineers, 1993.
9. M. J. Egenhofer. Reasoning about Binary Topological Relations. In Oliver Günther and H.-J. Scheck, editors, *Advances in Spatial Databases*, Zürich, August 28–30 1991, volume 525 of *Lecture Notes in Computer Science*, pages 143–160. Springer-Verlag, Berlin – Heidelberg – New York, 1991.
10. Ch. Freksa. Qualitative Spatial Reasoning. In David M. Mark and Andrew U. Frank, editors, *Cognitive and Linguistic Aspects of Geographic Space*, NATO ASI Series. Kluwer Academic Publ., Boston – Dortrecht – Lancaster, 1991.
11. A. Guttman. R-Trees: A Dynamic Index Structure for Spatial Searching. In *Proc. ACM SIGMOD International Conference on Management of Data*, page 47ff, 1984.
12. V. Haarslev. Knowledge-based Parsing and Formal Specification of Visual Languages. Bericht, Fachbereich Informatik, Universität Hamburg, 1994. in preparation.
13. Ch. Habel. Repräsentation räumlichen Wissens. Mitteilung FBI-HH-M-153/87, Fachbereich Informatik, Universität Hamburg, 1987.
14. Ph. Hanschke. Specifying Role Interactions in Concept Languages. In Bernhard Nebel, Charles Rich, and William Swartout, editors, *Principles of Knowledge Representation and Reasoning – Proc. of the Third International Conference KR'92*, Cambridge, Mass., October 25–29, 1992, pages 318–329. Morgan Kaufmann Publ. Inc., San Mateo, CA, 1992.
15. J. Jaffar, S. Michaylov, P. J. Stuckey, and R. H. C. Yap. The CLP(R) Language and System. *ACM Trans. on Programming Languages and Systems* **14** (3), 339–395, 1992.
16. K. M. Kahn, V. A. Saraswat, and V. Haarslev. Pictorial Janus: Eine vollständig visuelle Programmiersprache und ihre Umgebung. In J. Encarnacao, editor, *GI-Fachgespräch Programmieren multimedialer Anwendungen der GI-Jahrestagung 1991*, Darmstadt, Oktober 1991, pages 427–436. Springer-Verlag.
17. K. Klabunde. Erweiterung der Wissensrepräsentationssprache L-LILOG um Konstrukte zu Spezifikation von Kontrollinformation. IWBS Report 92, IBM Wissenschaftliches Zentrum, Institut für Wissensbasierte Systeme, Stuttgart, 1989.
18. S. Kosslyn. *Image and Mind*. Harvard University Press, 1980.
19. R. M. MacGregor. The Evolving Technology of Classification-Based Knowledge Representation Systems. In John F. Sowa, editor, *Principles of Semantic Networks – Explorations in the Representation of Knowledge*, chapter 13, pages 385–400. Morgan Kaufmann Publ. Inc., San Mateo, CA, 1991.
20. J.-P. Mohren and J. Müller. Representing Spatial Relations (Part II): The Geometrical Approach. DFKI Research Report RR-92-21, Deutsches Forschungszentrum für Künstliche Intelligenz, Kaiserslautern, April 1992.
21. S. Pribbenow. *Räumliche Konzepte in Wissens- und Sprachverarbeitung – Hybride Verarbeitung von Lokalisierung*. Deutscher Universitäts Verlag, Wiesbaden, 1993.
22. R. Röhrig. A Theory for Qualitative Spatial Reasoning Based on Order Relations. In *Proceedings of the Twelfth National Conference on Artificial Intelligence AAAI-94*, Seattle, Washington, July 31 – August 4, 1994. AAAI-Press/The MIT Press, Menlo Park – Cambridge – London, 1994.
23. M. Schick, S. Kockskämper, and B. Neumann. Konstellationserkennung auf Basis einer hybriden Raumrepräsentation. Behavior Memo 03-93, Fachbereich Informatik, Universität Hamburg, 1993.
24. M. Schmidt-Schauß and G. Smolka. Attributive concept descriptions with complements. *Artificial Intelligence* **48** (1), 1–26, 1991.