A Description Logic with Concrete Domains and a Role-forming Predicate Operator*

Volker Haarslev
University of Hamburg, Computer Science Department
Vogt-Kölln-Str. 30, 22527 Hamburg, Germany
http://kogs-www.informatik.uni-hamburg.de/~haarslev/

Carsten Lutz
RWTH Aachen, LuFG Theoretical Computer Science
Ahornstr. 55, 52074 Aachen, Germany
lutz@cantor.informatik.rwth-aachen.de

Ralf Möller
University of Hamburg, Computer Science Department
Vogt-Kölln-Str. 30, 22527 Hamburg, Germany
http://kogs-www.informatik.uni-hamburg.de/~moeller/

March 25, 1999

Abstract

This article presents the description logic $\mathcal{ALCRP}(\mathcal{D})$ with concrete domains and a role-forming predicate operator as its prominent aspects. We demonstrate the feasibility of $\mathcal{ALCRP}(\mathcal{D})$ for reasoning about spatial objects and their qualitative spatial relationships and provide an appropriate concrete domain for spatial objects. The general significance of $\mathcal{ALCRP}(\mathcal{D})$ is demonstrated by adding temporal reasoning to spatial and terminological reasoning using a combined concrete domain. The theory is motivated as a basis for knowledge representation and query processing in the domain of geographic information systems. In contrast to existing work in this domain, which mainly focuses either on conceptual reasoning or on reasoning about qualitative spatial relations, we integrate reasoning about spatial information with terminological reasoning.

Keywords: description logic, spatial reasoning, spatiotemporal reasoning, theoretical foundations for GIS.

*To appear in: *Journal of Logic and Computation*, Vol. 9 No. 3, 1999

1 Introduction

We have developed a new description logic called $\mathcal{ALCRP}(\mathcal{D})$ in order to provide a foundation for integrating reasoning about qualitative relations with terminological reasoning using description logics [26, 18]. Our research is strongly motivated by spatial domains such as geographic information systems (GIS). In this context, inferences about qualitative relations should not be considered in isolation but should be integrated with inferences about descriptions of domain objects (e.g. automatic consistency checking and classification). The main idea of our approach is to deal with knowledge about abstract domain objects using description logic theory and to deal with spatial objects and their qualitative *topological* relations using predicates defined over these objects. In our previous work on spatial and terminological reasoning [20] as well as in other related work [16, 21, 17] we used topological relations only as primitives in the sense of logic. Although some inferences about spatial objects are integrated into ABox reasoning, not all implicit information is exploited for terminological reasoning.

With the help of $\mathcal{ALCRP}(\mathcal{D})$ we can extend the treatment of qualitative relations –and topological relations in particular– especially with respect to TBox reasoning. Thus, the theory presented in this article allows one to detect both inconsistencies and implicit information in formal conceptual models for spatial domain objects. Only the combination of terminological and topological reasoning ensures that this can be achieved according to the intended semantics of the spatial domain objects. The combination of formal conceptual and qualitative reasoning can serve as a theoretical basis for knowledge representation in the domains mentioned above and can be used to solve important application problems [19].

$\mathcal{ALCRP}(\mathcal{D})$ is based on the description logic $\mathcal{ALC}(\mathcal{D})$ which is defined in [4]. $\mathcal{ALC}(\mathcal{D})$ divides the set of logical objects into two disjoint sets, the *abstract* and the *concrete* objects, e.g. real numbers. Abstract objects can be related to concrete objects via features (functional roles). Relationships between concrete objects are described with a set of domain-specific predicates. Referring to these predicates, properties of abstract objects can also be expressed using a *concept-forming* predicate operator. In $\mathcal{ALC}(\mathcal{D})$, the pair consisting of a set of concrete objects and a set of predicates defined over these objects is called a *concrete domain*. $\mathcal{ALCRP}(\mathcal{D})$ extends $\mathcal{ALC}(\mathcal{D})$ by introducing *defined roles* that are based on a *role-forming* predicate operator.

In [18] we have shown that the inference problem of checking the satisfiability of $\mathcal{ALCRP}(\mathcal{D})$ concept terms is undecidable in general. One of the main contribution of this article is the development of a restrictedness criterion such that the termination of the calculus can be guaranteed. We prove that testing the satisfiability of so-called restricted $\mathcal{ALCRP}(\mathcal{D})$ concept terms is decidable. Furthermore, this article formally defines a concrete domain for representing spatial objects and for dealing with their (qualitative) topological relationships. Considering an application context such as a GIS we base spatial relations on a set of topological relations in accordance to RCC-8 [32] (also similar to [13]). The article contains a proof for the admissibility of the spatial concrete domain. In addition, we sketch the definition of a temporal concrete domain based on Allen's Interval Algebra [1] and discuss the combination of concrete domains. We develop an extended combination operator for concrete domains and show that with a role-forming predicate operator and the combination of concrete

domains additional expressive power is gained in comparison to $\mathcal{ALC}(\mathcal{D})$. Although, due to the restrictedness criteria, modeling with $\mathcal{ALCRP}(\mathcal{D})$ can be hard, the examples presented in this article indicate that many interesting spatioterminological phenomena can now be represented and reasoned about.

The remainder of this article is organized as follows. The next section introduces the syntax and semantics of $\mathcal{ALCRP}(\mathcal{D})$ and the syntactic restrictions for concept terms. An algorithm for deciding the consistency problem for $\mathcal{ALCRP}(\mathcal{D})$-Aboxes is devised and its soundness and completeness proven. Section 3 demonstrates that $\mathcal{ALCRP}(\mathcal{D})$ is an appropriate formalism for spatial reasoning by presenting our GIS example domain with the help of a concrete domain for *two-dimensional* spatial objects. We conclude this article with a short discussion of related work and point out topics for future research.

2 The Description Logic $\mathcal{ALCRP}(\mathcal{D})$

In this section, the description logic $\mathcal{ALCRP}(\mathcal{D})$ is introduced by giving a formal syntax and semantics. Afterwards, a syntactically restricted version of the logic is formulated for which a calculus for deciding the ABox consistency problem is given. Soundness and completeness of this calculus are proved.

2.1 The Formalism

The concept language $\mathcal{ALCRP}(\mathcal{D})$ is a descendant of $\mathcal{ALC}(\mathcal{D})$ (see [4]) and thus incorporates concrete domains. For the sake of completeness, the formal definition of concrete domains is given first.

Definition 1 [4] A *concrete domain* $\mathcal{D}$ is a pair $(\Delta_\mathcal{D}, \Phi_\mathcal{D})$, where $\Delta_\mathcal{D}$ is a set called the domain, and $\Phi_\mathcal{D}$ is a set of predicate names. Each predicate name P from $\Phi_\mathcal{D}$ is associated with an arity n, and an n-ary predicate $P^\mathcal{D} \subseteq \Delta_\mathcal{D}^n$. A concrete domain $\mathcal{D}$ is called *admissible* iff (1) the set of its predicate names is closed under negation and contains a name for $\Delta_\mathcal{D}$, (2) the satisfiability problem for finite conjunctions of predicates is decidable.

For any predicate name $P \in \Phi_\mathcal{D}$, $\overline{P}$ denotes the *negation* of P for which we have $\overline{P}^\mathcal{D} := (\Delta_\mathcal{D})^n \setminus P^\mathcal{D}$, where n is the arity associated with P. Specific concrete domains will be investigated more thoroughly in Section 3. Now the role terms of $\mathcal{ALCRP}(\mathcal{D})$ can be defined.

Definition 2 Let R and F be disjoint sets of role and feature names, respectively. Any element of $\mathsf{R} \cup \mathsf{F}$ is an *atomic* role term. A composition of features (written $f_1 f_2 \cdots f_k$) is called a feature chain. A simple feature can be viewed as a feature chain of length 1. If $P \in \Phi_\mathcal{D}$ is a predicate name with arity $n + m$ and $u_1, \ldots, u_n$ as well as $v_1, \ldots, v_m$ are feature chains, then the expression $\exists(u_1, \ldots, u_n)(v_1, \ldots, v_m).P$ (*role-forming predicate operator*) is a *complex* role term. Let S be a role name and let T be a role term. Then $S \doteq T$ is a *terminological axiom*. This type of terminological axiom is also called *role definition*.

Next, the set of $\mathcal{ALCRP}(\mathcal{D})$ concept terms is defined. It will be discussed later that, in order to obtain a formalism for which reasoning is decidable, some combinations of operators have to be prohibited.

Definition 3 Let C be a set of concept names which is disjoint from R and F. Any element of C is a *concept term* (*atomic* concept term). If C and D are concept terms, R is a role term[1], $P \in \Phi_{\mathcal{D}}$ is a predicate name with arity n, and $u_1, \ldots, u_n$ are feature chains, then the following expressions are also concept terms:

- $\neg C, \quad C \sqcap D, \quad C \sqcup D, \quad \forall R.C, \quad \exists R.C,$
- $\exists u_1, \ldots, u_n.P.$

For all kinds of exists and value restrictions, the role term or list of feature chains may be written in parentheses. Let A be a concept name and let D be a concept term. Then $A \doteq D$ and $A \sqsubseteq D$ are terminological axioms as well. Axioms of these types are also called *concept definitions*. A finite set of terminological axioms $\mathcal{T}$ is called a *terminology* or *TBox* if the left-hand sides of all terminological axioms in $\mathcal{T}$ are unique and, furthermore, all concept definitions are acyclic.

Please refer e.g. to [30] for an investigation of terminological cycles. Now, a meaning can be assigned to $\mathcal{ALCRP}(\mathcal{D})$ concept terms by giving a set-theoretic semantics as usual.

Definition 4 An *interpretation* $\mathcal{I} = (\Delta_{\mathcal{I}}, \cdot^{\mathcal{I}})$ consists of a set $\Delta_{\mathcal{I}}$ (the abstract domain) and an interpretation function $\cdot^{\mathcal{I}}$. The sets $\Delta_{\mathcal{D}}$ and $\Delta_{\mathcal{I}}$ must be disjoint. The interpretation function maps each concept name C to a subset $C^{\mathcal{I}}$ of $\Delta_{\mathcal{I}}$, each role name R to a subset $R^{\mathcal{I}}$ of $\Delta_{\mathcal{I}} \times \Delta_{\mathcal{I}}$, and each feature name f to a partial function $f^{\mathcal{I}}$ from $\Delta_{\mathcal{I}}$ to $\Delta_{\mathcal{D}} \cup \Delta_{\mathcal{I}}$, where $f^{\mathcal{I}}(a) = x$ will be written as $(a, x) \in f^{\mathcal{I}}$. If $u = f_1 \cdots f_n$ is a feature chain, then $u^{\mathcal{I}}$ denotes the composition $f_1^{\mathcal{I}} \circ \ldots \circ f_n^{\mathcal{I}}$ of the partial functions $f_1^{\mathcal{I}}, \ldots, f_n^{\mathcal{I}}$. Let the symbols C, D, R, P, $u_1, \ldots, u_m$, and $v_1, \ldots, v_m$ be defined as in Definition 2 and 3, respectively. Then the interpretation function can be extended to arbitrary concept and role terms as follows:

[1] Please note that features are also role terms.

$$
\begin{aligned}
(C \sqcap D)^{\mathcal{I}} &:= C^{\mathcal{I}} \cap D^{\mathcal{I}} \\
(C \sqcup D)^{\mathcal{I}} &:= C^{\mathcal{I}} \cup D^{\mathcal{I}} \\
(\neg C)^{\mathcal{I}} &:= \Delta_{\mathcal{I}} \setminus C^{\mathcal{I}} \\
(\exists R.C)^{\mathcal{I}} &:= \{a \in \Delta_{\mathcal{I}} \mid \exists b \in \Delta_{\mathcal{I}} \colon (a,b) \in R^{\mathcal{I}} \wedge b \in C^{\mathcal{I}}\} \\
(\forall R.C)^{\mathcal{I}} &:= \{a \in \Delta_{\mathcal{I}} \mid \forall b \colon (a,b) \in R^{\mathcal{I}} \rightarrow b \in C^{\mathcal{I}}\} \\
(\exists u_1, \ldots, u_n.P)^{\mathcal{I}} &:= \{a \in \Delta_{\mathcal{I}} \mid \exists x_1, \ldots, x_n \in \Delta_{\mathcal{D}} \colon \\
&\qquad (a, x_1) \in u_1^{\mathcal{I}} \wedge \cdots \wedge (a, x_n) \in u_n^{\mathcal{I}} \wedge (x_1, \ldots, x_n) \in P^{\mathcal{D}}\} \\
(\exists (u_1, \ldots, u_n)(v_1, \ldots, v_m).P)^{\mathcal{I}} &:= \\
&\qquad \{(a,b) \in \Delta_{\mathcal{I}} \times \Delta_{\mathcal{I}} \mid \exists x_1, \ldots, x_n, y_1, \ldots, y_m \in \Delta_{\mathcal{D}} \colon \\
&\qquad\quad (a, x_1) \in u_1^{\mathcal{I}} \wedge \cdots \wedge (a, x_n) \in u_n^{\mathcal{I}} \wedge \\
&\qquad\quad (b, y_1) \in v_1^{\mathcal{I}} \wedge \cdots \wedge (b, y_m) \in v_m^{\mathcal{I}} \wedge \\
&\qquad\quad (x_1, \ldots, x_n, y_1, \ldots, y_m) \in P^{\mathcal{D}}\}
\end{aligned}
$$

An interpretation $\mathcal{I}$ is a *model* of a TBox $\mathcal{T}$ iff it satisfies $A^{\mathcal{I}} = D^{\mathcal{I}}$ for all terminological axioms $A \doteq D$ and $A^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for all $A \sqsubseteq D$ in $\mathcal{T}$. A concept term C *subsumes* a concept term D w.r.t. a TBox $\mathcal{T}$ (written $D \preceq_{\mathcal{T}} C$), iff $D^{\mathcal{I}} \subseteq C^{\mathcal{I}}$ for all models $\mathcal{I}$ of $\mathcal{T}$. A concept term C is *satisfiable* w.r.t. a TBox $\mathcal{T}$ iff there exists a model $\mathcal{I}$ of $\mathcal{T}$ such that $C^{\mathcal{I}} \neq \emptyset$. An object b is called a *role filler* of an object a in an interpretation $\mathcal{I}$ if $(a,b) \in R^{\mathcal{I}}$ for some role R.

Note that a universally quantifying concept-forming operator based on predicates $\forall f_1, \ldots, f_m.P$ can be expressed as $\exists f_1, \ldots, f_m.P \sqcup \forall f_1.\bot \sqcup \ldots \sqcup \forall f_m.\bot$, where $\bot^{\mathcal{I}} = \emptyset$.[2] A universally quantifying, role-forming predicate-based operator $\forall (u_1, \ldots, u_n)(v_1, \ldots, v_m).P$ cannot be expressed in the given logic. To obtain the semantics of this operator, change the existential quantification in the semantics of the role-forming operator of $\mathcal{ALCRP}(\mathcal{D})$ to a universal quantification, and, furthermore, change the last conjunction symbol to an implication symbol. The fact that the described operator is missing does not seem to be a serious lack since it is very hard to think of any cases in which such an operator could be useful. The only difference to its existentially quantifying counterpart is that its extension does also include those pairs of abstract objects (a,b) where either a does not have concrete fillers for all feature chains $u_1, \ldots, u_n$ or where b does not have concrete fillers for all feature chains $v_1, \ldots, v_m$.

One of the basic reasoning services for a description logic formalism is computing the subsumption relationship. This inference is needed in the TBox to build a hierarchy of concepts w.r.t. specificity. Satisfiability and subsumption can be mutually reduced to each other since $C \preceq_{\mathcal{T}} D$ iff $C \sqcap \neg D$ is not satisfiable w.r.t. $\mathcal{T}$ and C is unsatisfiable w.r.t. $\mathcal{T}$ iff $C \preceq_{\mathcal{T}} \bot$.

The following definition introduces the assertional language of $\mathcal{ALCRP}(\mathcal{D})$, which can be used to represent knowledge about individual worlds.

[2] $\bot$ can be expressed as $A \sqcap \neg A$.

Definition 5 Let O_D and O_A be two disjoint sets of object names. If C is a concept term, R a role term, f a feature name, P a predicate name with arity n, a and b are elements of O_A and x, and $x_1, \ldots, x_n$ are elements of O_D, then the following expressions are *assertional axioms*:

$$a : C, \quad (a, b) : R, \quad (a, x) : f, \quad (x_1, \ldots, x_n) : P$$

A finite set of assertional axioms is called an *ABox*. An *interpretation* for the concept language can be extended to the assertional language by additionally mapping every object name from O_A to a single element of $\Delta_{\mathcal{I}}$ and every object name from O_D to a single element from $\Delta_{\mathcal{D}}$. The unique name assumption is not imposed, that is $a^{\mathcal{I}} = b^{\mathcal{I}}$ may hold even if $a \neq b$. An interpretation satisfies an assertional axiom

$$\begin{aligned}
&a : C \ \text{ iff } \ a^{\mathcal{I}} \in C^{\mathcal{I}}, \qquad (a, b) : R \ \text{ iff } \ (a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}, \\
&(a, x) : f \ \text{ iff } \ f^{\mathcal{I}}(a^{\mathcal{I}}) = x^{\mathcal{I}}, \\
&(x_1, \ldots, x_n) : P \ \text{ iff } \ (x_1^{\mathcal{I}}, \ldots, x_n^{\mathcal{I}}) \in P^{\mathcal{D}}
\end{aligned}$$

An interpretation is a *model* of an ABox $\mathcal{A}$ w.r.t. a TBox $\mathcal{T}$, iff it is a model of $\mathcal{T}$ and furthermore satisfies all assertional axioms in $\mathcal{A}$. An ABox is *consistent* w.r.t. a TBox $\mathcal{T}$ iff it has a model w.r.t. $\mathcal{T}$.

The ABox consistency problem is to decide whether a given ABox $\mathcal{A}$ is consistent w.r.t. a TBox $\mathcal{T}$. Satisfiability of concept terms can be reduced to ABox consistency as follows: A concept term C is satisfiable iff the ABox $\{a : C\}$ is consistent. In [27], it was proved that reasoning with $\mathcal{ALCRP}(\mathcal{D})$ will be undecidable if the full expressive power is used. Therefore, in Section 2.3 syntactic restrictions on the concept language are introduced. But first, a remark about concrete domains is appropriate.

2.2 A Note on Concrete Domains

In [5], it is shown that any two admissible concrete domains $\mathcal{D}_1$ and $\mathcal{D}_2$, where $\Delta_{\mathcal{D}_1}$ and $\Delta_{\mathcal{D}_2}$ are disjoint, can be combined into a concrete domain $\mathcal{D}_1 \oplus \mathcal{D}_2$ which is also admissible. Thus, the fact that $\mathcal{ALC}(\mathcal{D})$ can be parameterized only by a single concrete domain is not a limitation. In case of $\mathcal{ALCRP}(\mathcal{D})$, the combination technique proposed by Baader and Hanschke can also be used. Nevertheless, it can be sensibly extended to include additional predicates to be used inside the role-forming predicate operator.

Baader and Hanschke combine two concrete domains $\mathcal{D}_1$ and $\mathcal{D}_2$ which have disjoint domain sets $\Delta_{\mathcal{D}_1}$ and $\Delta_{\mathcal{D}_2}$ into a new concrete domain $\mathcal{D}_1 \oplus \mathcal{D}_2$ as follows: (1) The domain $\Delta_{\mathcal{D}_1 \oplus \mathcal{D}_2}$ is the union of $\Delta_{\mathcal{D}_1}$ and $\Delta_{\mathcal{D}_2}$ and (2) the set of predicates $\Phi_{\mathcal{D}_1 \oplus \mathcal{D}_2}$ is set to $\Phi_{\mathcal{D}_1} \cup \Phi_{\mathcal{D}_2}$ plus additional negation predicates. Please recall that a concrete domain is required to be closed under negation to be admissible. This requires the additional negation predicates to be defined since in $\mathcal{D}_1 \oplus \mathcal{D}_2$ the complements of predicates have to be defined with respect to $\Delta_{\mathcal{D}_1} \cup \Delta_{\mathcal{D}_2}$ and not just with respect to $\Delta_{\mathcal{D}_1}$ or $\Delta_{\mathcal{D}_2}$ alone.

Let $\mathcal{D}_1$ and $\mathcal{D}_2$ be concrete domains. Using $\mathcal{ALCRP}(\mathcal{D}_1 \oplus \mathcal{D}_2)$, it is possible to describe the set of abstract objects that fulfill both of the two concepts $\exists u_1, \ldots, u_n.P_1$ and $\exists u'_1, \ldots, u'_m.P_2$, where $P_1 \in \Phi_{\mathcal{D}_1}$ and $P_2 \in \Phi_{\mathcal{D}_2}$, by simply using the conjunction $\exists u_1, \ldots, u_n.P_1 \sqcap \exists u'_1, \ldots, u'_m.P_2$. On the other hand, it is not possible to describe the set of pairs of abstract objects that are in the extension of both $\exists(u_1, \ldots, u_n)(v_1, \ldots, v_m).P_3$ and $\exists(u'_1, \ldots, u'_k)(v'_1, \ldots, v'_l).P_4$, where $P_3 \in \Phi_{\mathcal{D}_1}$ and $P_4 \in \Phi_{\mathcal{D}_2}$. In Section 3.3, it is demonstrated that this kind of expressivity is very useful in the context of spatiotemporal terminological reasoning. One solution would of course be to add a role conjunction operator to the logic. In the following, it will be shown that this is not necessary since it is possible to add more predicates to $\Phi_{\mathcal{D}_1 \oplus \mathcal{D}_2}$ (without losing admissibility) which allows one to specify exactly the kind of relationship described above.

Definition 6 Let $\mathcal{D}_1$ and $\mathcal{D}_2$ be two concrete domains with $\Phi_{\mathcal{D}_1} \cap \Phi_{\mathcal{D}_2} = \emptyset$. The *extended combination* of $\mathcal{D}_1$ and $\mathcal{D}_2$ (denoted by $\mathcal{D}_1 \oplus' \mathcal{D}_2$) is defined as follows. The domain $\Delta_{\mathcal{D}_1 \oplus' \mathcal{D}_2}$ is the union of $\Delta_{\mathcal{D}_1}$ and $\Delta_{\mathcal{D}_2}$. The set $\Phi_{\mathcal{D}_1 \oplus' \mathcal{D}_2}$ consists of the following predicates:

- For any predicate name P from $\Phi_{\mathcal{D}_1}$ or $\Phi_{\mathcal{D}_2}$ associated with an arity n, there are two predicate names Q and $\widehat{Q}$ in $\Phi_{\mathcal{D}_1 \oplus' \mathcal{D}_2}$, which are also both associated with arity n. Let $\mathcal{D}(P)$ denote the concrete domain that the predicate P stems from. The corresponding predicates are defined as follows:

$$
\begin{aligned}
&(x_1, \ldots, x_n) \in Q^{\mathcal{D}_1 \oplus' \mathcal{D}_2} \text{ iff } (x_1, \ldots, x_n) \in P^{\mathcal{D}(P)}, \\
&(x_1, \ldots, x_n) \in \widehat{Q}^{\mathcal{D}_1 \oplus' \mathcal{D}_2} \text{ iff } (x_1, \ldots, x_n) \in \overline{P}^{\mathcal{D}(P)} \\
&\qquad\qquad \vee\ x_1 \notin \Delta_{\mathcal{D}(P)} \ \vee\ \cdots\ \vee x_n \notin \Delta_{\mathcal{D}(P)}.
\end{aligned}
$$

- For any pair of predicate names P and Q associated with an arity of at least 2, where P is from $\Phi_{\mathcal{D}_1}$ and Q is from $\Phi_{\mathcal{D}_2}$ or vice versa, a number of *combined predicates* is in $\Phi_{\mathcal{D}_1 \oplus' \mathcal{D}_2}$. Let n be the arity associated with P and let m be the arity associated with Q. Then for all $i = 1, \ldots, n-1$ and for all $j = 1, \ldots, m-1$, new predicates $P\text{-}Q_{i,j}$ and $\widehat{P\text{-}Q_{i,j}}$ are defined:

$$
\begin{aligned}
(x_1, \ldots, x_{n+m}) \in P\text{-}Q_{i,j}{}^{\mathcal{D}_1 \oplus' \mathcal{D}_2} \text{ iff } &(x_1, \ldots, x_i, x_{i+j+1}, \ldots, x_{n+j}) \in P^{\mathcal{D}(P)} \\
&\wedge\ (x_{i+1}, \ldots, x_{i+j}, x_{n+j+1}, \ldots, x_{n+m}) \in Q^{\mathcal{D}(Q)},
\end{aligned}
$$

$$
\begin{aligned}
(x_1, \ldots, x_{n+m}) \in \widehat{P\text{-}Q_{i,j}}^{\mathcal{D}_1 \oplus' \mathcal{D}_2} \text{ iff } &(x_1, \ldots, x_i, x_{i+j+1}, \ldots, x_{n+j}) \in \overline{P}^{\mathcal{D}(P)} \\
&\vee\ (x_{i+1}, \ldots, x_{i+j}, x_{n+j+1}, \ldots, x_{n+m}) \in \overline{Q}^{\mathcal{D}(Q)} \\
&\vee\ x_1 \in \Delta_{\mathcal{D}(Q)} \ \vee\ \cdots\ \vee x_n \in \Delta_{\mathcal{D}(Q)} \\
&\vee\ x_{n+1} \in \Delta_{\mathcal{D}(P)} \ \vee\ \cdots\ \vee x_{n+m} \in \Delta_{\mathcal{D}(P)}.
\end{aligned}
$$

  When both predicates P and Q are binary or i and j are derivable from the context, the index i, j may be omitted.

It can be shown that the extended combination of two admissible concrete domains is again an admissible concrete domain.

Lemma 7 If $\mathcal{D}_1$ and $\mathcal{D}_2$ are admissible concrete domains then $\mathcal{D}_1 \oplus' \mathcal{D}_2$ is also an admissible concrete domain.

PROOF. Since it is easy to verify that $\Phi_{\mathcal{D}_1 \oplus' \mathcal{D}_2}$ is closed under negation and contains a name for $\Delta_{\mathcal{D}_1 \oplus' \mathcal{D}_2}$, it remains to be shown that the satisfiability of any finite conjunction of predicates from $\mathcal{D}_1 \oplus' \mathcal{D}_2$ is decidable. The proof is identical to the one used for the non-extended combination as given in [5] and will not be repeated in detail here. The idea is to replace combined predicates and negated predicates by conjunctions and disjunctions corresponding to their definitions given in Definition 6 and to convert the resulting formula to disjunctive normal form. Then, each disjunct can be treated separately. If a variable occurs both inside a predicate from $\Phi_{\mathcal{D}_1}$ and inside a predicate from $\Phi_{\mathcal{D}_2}$, then the disjunct is not satisfiable. Otherwise, the conjunction can be split into two parts: one part contains only predicates from $\Phi_{\mathcal{D}_1}$ and the other contains only predicates from $\Phi_{\mathcal{D}_2}$. Using the satisfiability tests for $\mathcal{D}_1$ and $\mathcal{D}_2$, the two parts can be separately tested for satisfiability. ∎

As already noted, the usefulness of extended combinations of concrete domains will be demonstrated in Section 3.3.

2.3 Restricting $\mathcal{ALCRP}(\mathcal{D})$

In [18], we show that the satisfiability problem for $\mathcal{ALCRP}(\mathcal{D})$ concept terms is undecidable by using a reduction from Post's Correspondence Problem. From this it directly follows that the subsumption and ABox consistency problems are also undecidable. In fact, even for a logic comprised only of conjunction, value restriction, the top concept, and the role-forming predicate operator, the standard reasoning problems are undecidable (see [27]).

A more thorough analysis (see [27, 25]) reveals that $\mathcal{ALCRP}(\mathcal{D})$ does have neither the finite model property nor the tree model property if instantiated with an appropriate concrete domain.[3] This does not imply, but is a strong indication for the undecidability of the formalism. And in fact, for the PCP-reduction a concept is employed that is only satisfied by infinite models.

There are at least two options to overcome the deficiency of undecidability. First, restrictions could be posed on the structure of the concrete domain predicates. This is not very promising since all interesting applications, e.g. those involving topological relations, require fairly complex predicate definitions that are very likely to cause undecidability. Second, the ability to combine some critical operators could be restricted. We will pursue the second approach. Before the structurally restricted terminologies can be introduced, some technical definitions have to be made.

A concept term C is said to be *unfolded* w.r.t. a TBox $\mathcal{T}$ iff none of the concept and role names used in the concept term occur on the left side of a terminological axiom in $\mathcal{T}$. Any concept term can be transformed into an unfolded form by iteratively

[3] A logic has the *finite model property* if any satisfiable concept is satisfied by a finite model. A logic has the *tree model property* if any satisfiable concept is satisfied by a model which has the form of a tree, where the nodes correspond to domain objects and the edges to role filler relationships.

replacing concept and role names by their defining terms. This algorithm terminates since the terminology is required to be acyclic. Any unfolded concept term can then be transformed to an equivalent one in *negation normal form* (NNF). An unfolded concept term is said to be in NNF iff negation occurs only in front of concept names. The transformation to NNF can be done by iteratively applying transformation rules that propagate the negations "down" to the atomic concepts. For example, $\neg(C \sqcap D)$ has to be transformed to $\neg C \sqcup \neg D$. Details are omitted since the transformation rules are the same as for $\mathcal{ALC}(\mathcal{D})$ [4]. Now, the notion of a restricted terminology can be defined.

Definition 8 A concept term X is called *restricted* w.r.t. a TBox $\mathcal{T}$ iff its equivalent X' —that is unfolded w.r.t. $\mathcal{T}$ and in NNF— fulfills the following conditions:[4]

1. For any subconcept term C of X' that is of the form $\forall R_1.D$ where R_1 is a complex role term, D does not contain any terms of the form $\exists R_2.E$ where R_2 is also a complex role term.

2. For any subconcept term C of X' that is of the form $\exists R_1.D$ where R_1 is a complex role term, D does not contain any terms of the form $\forall R_2.E$ where R_2 is also a complex role term.

3. For any subconcept term C of X' that is of the form $\forall R.D$ or $\exists R.D$ where R is a complex role term, D contains only predicate exists restrictions that (i) quantify over feature chains of length 1 and (ii) are not contained inside any value and exists restrictions that are also contained in D.

A terminology is called restricted iff all concept terms occurring on the right-hand side of terminological axioms in $\mathcal{T}$ are restricted w.r.t. $\mathcal{T}$. An ABox $\mathcal{A}$ is called restricted w.r.t. a TBox $\mathcal{T}$ iff $\mathcal{T}$ is restricted and all concept terms used in $\mathcal{A}$ are restricted w.r.t. the terminology $\mathcal{T}$.

For a specification of restricted concept terms using EBNF notation see [26]. For demonstration purposes, the following six very simple terminologies are given. The terminologies with even numbers are restricted while the ones with odd numbers are not restricted because they violate one of the above conditions. Let C and D be concept names, R_a be an atomic role term, R_c be a complex role term, f be a feature, and u be a feature chain with a length greater than 1.

$$\mathcal{T}_1 : \{C \doteq \forall R_c.\exists R_c.D\}, \qquad \mathcal{T}_2 : \{C \doteq \forall R_a.\exists R_c.D\},$$
$$\mathcal{T}_3 : \{C \doteq \exists R_c.\exists u.P\}, \qquad \mathcal{T}_4 : \{C \doteq \exists R_c.\exists f.P\},$$
$$\mathcal{T}_5 : \{C \doteq \forall R_c.\forall R_a.\exists f.P\}, \quad \mathcal{T}_6 : \{C \doteq \forall R_a.\forall R_c.\exists f.P\}$$

The syntactical restrictions ensure that the finite model property holds in the restricted formalism.[5] This means that "dangerous" concepts such as the one needed for the PCP-reduction can no longer be constructed. In fact, when only considering restricted concept terms, the standard reasoning problems are decidable.

[4] For technical reasons, it is assumed that each concept term is a subconcept term of itself.
[5] However, the tree model property still does not hold.

Theorem 1 The ABox consistency problem for restricted $\mathcal{ALCRP}(\mathcal{D})$ ABoxes is decidable.

In the next section, a sound and complete algorithm is given which decides the consistency problem for restricted $\mathcal{ALCRP}(\mathcal{D})$ ABoxes. The existence of such an algorithm proves the above theorem. It is important that this result also implies the decidability of the subsumption and satisfiability problems.

Corollary 9 The subsumption problem and the satisfiability problem for $\mathcal{ALCRP}(\mathcal{D})$ concept terms are decidable w.r.t. terminologies for which the considered concept terms are restricted.

PROOF. This follows from Theorem 1 together with the reduction of subsumption to satisfiability and, in turn, of satisfiability to ABox consistency (see Section 2.1). Please note that if the concept terms C and D are restricted w.r.t. a terminology $\mathcal{T}$, then the concept term $C \sqcap \neg D$ is also restricted w.r.t. $\mathcal{T}$ since (1) the conjunction of two restricted concept terms again yields a restricted concept term and (2) the set of restricted concept terms is closed under negation. To see the second point it is important to note that the restrictions in Definition 8 are symmetric with respect to the duality of operators. ∎

2.4 The Calculus

The algorithm is a standard tableau-based one as it is used for first order or modal logics as well as other description logics. To decide the satisfiability of an ABox $\mathcal{A}_0$ w.r.t. a TBox $\mathcal{T}$, every concept term in $\mathcal{A}_0$ has to be both unfolded w.r.t. $\mathcal{T}$ and converted to negation normal form. It was already noted in Section 2.3 that this is always possible.[6] The algorithm works as follows.

The assertional axioms are viewed as constraints on individual objects. Starting with the initial ABox $\mathcal{A}_0$, the algorithm iteratively applies completion rules to transform a given ABox into one or more descendent ABoxes. The descendent ABoxes are constructed by adding at least one additional axiom to the original ABox. The new axioms explicitly represent knowledge which was only "contained" implicitly in the ABox before the application of the rule. Sometimes there is more than one possible way to extend the original ABox. In this case, more than one descendent ABox is created by the application of a single rule. By iteratively applying rules, the algorithm thus produces a tree of ABoxes Υ. Iterative rule application can be understood as an attempt to build a model for the initial ABox $\mathcal{A}_0$. The iteration continues until either an ABox is produced that is complete, i.e. to which no more rules are applicable, or all leaf ABoxes in the tree Υ are contradictory. In the former case, the complete ABox defines a model for the initial ABox $\mathcal{A}_0$, in the latter case, no model for $\mathcal{A}_0$ exists.

Before the completion rules can be defined, some technical terms are introduced. Let $\mathcal{A}$ be an ABox, R be a role term, a and b be object names from O_A, γ be a symbol that is not element of O_D, u be a feature chain $f_1 \cdots f_k$, and let $u_1, \ldots, u_n$ and $v_1, \ldots, v_m$ (possibly with index) be arbitrary feature chains. For convenience three functions are defined as follows:

[6] In general, however, unfolding has exponential complexity (see [29]).

$$\mathit{filler}_{\mathcal{A}}(a,u) := \begin{cases} x \text{ where } x \in \mathsf{O}_D \text{ such that} \\ \quad \exists b_1,\ldots,b_{k-1} \in \mathsf{O}_A : \\ \quad\quad ((a,b_1):f_1 \in \mathcal{A},\ldots,(b_{k-1},x):f_k \in \mathcal{A}) \\ \gamma \text{ if no such } x \text{ exists.} \end{cases}$$

$$\mathit{chain}_{\mathcal{A}}(a,x,u) := \{(a,c_1):f_1,\ldots,(c_{k-1},x):f_k\} \\ \text{where } c_1,\ldots,c_{k-1} \in \mathsf{O}_A \text{ are not used in } \mathcal{A}.$$

$$\mathit{filler?}_{\mathcal{A}}(a,b,R) := \begin{cases} \mathit{true} \text{ if } (a,b):R \in \mathcal{A} \\ \mathit{true} \text{ if } R \text{ is of the form } \exists(u_1,\ldots,u_n)(v_1,\ldots,v_m).P, \\ \quad\quad \text{and } \exists x_1,\ldots,x_n,y_1,\ldots,y_m \in \mathsf{O}_D \text{ such that} \\ \quad\quad \mathit{filler}_{\mathcal{A}}(a,u_1)=x_1,\ \ldots\ ,\mathit{filler}_{\mathcal{A}}(a,u_n)=x_n, \\ \quad\quad \mathit{filler}_{\mathcal{A}}(b,v_1)=y_1,\ \ldots\ ,\mathit{filler}_{\mathcal{A}}(b,v_m)=y_m, \\ \quad\quad (x_1,\ldots,x_n,y_1,\ldots,y_m):P \in \mathcal{A} \\ \mathit{false} \text{ otherwise} \end{cases}$$

An ABox $\mathcal{A}$ is said to contain a *fork* (for a feature f) if it contains the two axioms $(a,b):f$ and $(a,c):f$, where a and b are either both from O_A or O_D. A fork can be eliminated by replacing all occurrences of c in $\mathcal{A}$ with b. Before any rules are applied to the initial ABox $\mathcal{A}_0$, fork elimination takes place. Forks may be introduced if completion rules generate new objects. In this case, fork elimination is immediately applied with the proviso that each occurrence of the newly generated object is replaced by the older one and not vice versa. The completion rules can now be defined.

Definition 10 The following *completion rules* will replace an ABox $\mathcal{A}$ by an ABox $\mathcal{A}'$ or by two ABoxes $\mathcal{A}'$ and $\mathcal{A}''$ (*descendants* of $\mathcal{A}$). In the following C and D denote concept terms, R denotes a role term, and P denotes a predicate name from $\Phi_{\mathcal{D}}$. Let $f_1, \ldots, f_n$ as well as $g_1, \ldots, g_n$ denote feature names, and $u_1, \ldots, u_m$ denote feature chains. a and b denote object names from O_A.

R⊓ The conjunction rule.
Premise: $a : C \sqcap D \in \mathcal{A}, \quad a : C \notin \mathcal{A} \ \vee\ a : D \notin \mathcal{A}$
Consequence: $\mathcal{A}' = \mathcal{A} \cup \{a : C,\ a : D\}$

R⊔ The disjunction rule.
Premise: $a : C \sqcup D \in \mathcal{A}, \quad a : C \notin \mathcal{A} \ \wedge\ a : D \notin \mathcal{A}$
Consequence: $\mathcal{A}' = \mathcal{A} \cup \{a : C\},\ \mathcal{A}'' = \mathcal{A} \cup \{a : D\}$

R∃C The exists restriction rule.
Premise: $a : \exists R.C \in \mathcal{A}, \quad \neg\exists b \in \mathsf{O}_A : (\mathit{filler?}_{\mathcal{A}}(a,b,R) \ \wedge\ b : C \in \mathcal{A})$
Consequence: $\mathcal{A}' = \mathcal{A} \cup \{(a,b) : R\ ,\ b : C\}$ where $b \in \mathsf{O}_A$ is not used in $\mathcal{A}$.
This rule may create a fork if R is a feature.

R∀C The value restriction rule.
Premise: $a : \forall R.C \in \mathcal{A}, \quad \exists b \in \mathsf{O}_A : (\mathit{filler?}_{\mathcal{A}}(a,b,R) \ \wedge\ b : C \notin \mathcal{A})$
Consequence: $\mathcal{A}' = \mathcal{A} \cup \{b : C\}$

R∃P The predicate exists restriction rule.

Premise: $a : \exists u_1, \ldots, u_n.P \in \mathcal{A}, \neg\exists x_1, \ldots, x_n \in \mathsf{O}_D :$
$(\mathit{filler}_{\mathcal{A}}(a, u_1) = x_1 \wedge \ \ldots \ \wedge \mathit{filler}_{\mathcal{A}}(a, u_n) = x_n \wedge$
$(x_1, \ldots, x_n) : P \in \mathcal{A})$
Consequence: $\mathcal{C}_0 := \mathcal{A} \cup \{(x_1, \ldots, x_n) : P\}$
where the $x_i \in \mathsf{O}_D$ are not used in $\mathcal{A}$.
$\mathcal{C}_1 := \mathit{chain}_{\mathcal{C}_0}(a, x_1, u_1), \ \ldots \ , \mathcal{C}_n := \mathit{chain}_{\mathcal{C}_{n-1}}(a, x_n, u_n)$
$\mathcal{A}' = \bigcup_{i=0..n} \mathcal{C}_n$
This rule may create forks.

Rr∃P The complex role rule.
Premise: $(a, b) : \exists(u_1, \ldots, u_n)(v_1, \ldots, v_m).P \in \mathcal{A},$
$\neg\exists x_1, \ldots, x_n, y_1, \ldots, y_m \in \mathsf{O}_D :$
$(\mathit{filler}_{\mathcal{A}}(a, u_1) = x_1 \wedge \ \ldots \ \wedge \mathit{filler}_{\mathcal{A}}(a, u_n) = x_n \wedge$
$\mathit{filler}_{\mathcal{A}}(b, v_1) = y_1 \wedge \ \ldots \ \wedge \mathit{filler}_{\mathcal{A}}(b, v_m) = y_m \wedge$
$(x_1, \ldots, x_n, y_1, \ldots, y_m) : P \in \mathcal{A})$
Consequence: $\mathcal{C}_0 := \mathcal{D}_0 := \mathcal{A} \cup \{(x_1, \ldots, x_n, y_1, \ldots, y_m) : P\}$
where the $x_i \in \mathsf{O}_D$ and $y_i \in \mathsf{O}_D$ are not used in $\mathcal{A}$.
$\mathcal{C}_1 := \mathit{chain}_{\mathcal{C}_0}(a, x_1, u_1), \ \ldots \ , \mathcal{C}_n := \mathit{chain}_{\mathcal{C}_{n-1}}(a, x_n, u_n),$
$\mathcal{D}_1 := \mathit{chain}_{\mathcal{D}_0}(b, y_1, v_1), \ \ldots \ , \mathcal{D}_m := \mathit{chain}_{\mathcal{D}_{m-1}}(b, y_m, v_m)$
$\mathcal{A}' = \bigcup_{i=0..n} \mathcal{C}_n \ \cup \ \bigcup_{i=1..n} \mathcal{D}_m$
This rule may create forks.

RChoose The choose rule.
Premise: $a : \forall(\exists(u_1, \ldots, u_n)(v_1, \ldots, v_m).P).C \in \mathcal{A},$
$\exists b \in \mathsf{O}_A, \ x_1, \ldots, x_n, y_1, \ldots, y_m \in \mathsf{O}_D :$
$(\mathit{filler}_{\mathcal{A}}(a, u_1) = x_1 \wedge \ \ldots \ \wedge \mathit{filler}_{\mathcal{A}}(a, u_n) = x_n \wedge$
$\mathit{filler}_{\mathcal{A}}(b, v_1) = y_1 \wedge \ \ldots \ \wedge \mathit{filler}_{\mathcal{A}}(b, v_m) = y_m \wedge$
$(x_1, \ldots, x_n, y_1, \ldots, y_m) : P \notin \mathcal{A} \wedge$
$(x_1, \ldots, x_n, y_1, \ldots, y_m) : \overline{P} \notin \mathcal{A})$
Consequence: $\mathcal{A}' = \mathcal{A} \cup \{(x_1, \ldots, x_n, y_1, \ldots, y_m) : P\},$
$\mathcal{A}'' = \mathcal{A} \cup \{(x_1, \ldots, x_n, y_1, \ldots, y_m) : \overline{P}\}$

The notion of a complete and a contradictory ABox still needs to be formally defined.

Definition 11 Let the same naming conventions be given as in Definition 10. Additionally, let f be a feature. An ABox $\mathcal{A}$ is called *contradictory* if any of the following *clash triggers* are applicable:

- Primitive Clash: $a : C \in \mathcal{A}, \quad a : \neg C \in \mathcal{A}$
- Feature Domain Clash: $(a, x) : f \in \mathcal{A}, \quad (a, b) : f \in \mathcal{A}$
- All Domain Clash: $(a, x) : f \in \mathcal{A}, \quad a : \forall f.C \in \mathcal{A}$
- Concrete Domain Clash:
$(x_1^{(1)}, \ldots, x_{n_1}^{(1)}) : P_1 \in \mathcal{A}, \ldots, (x_1^{(k)}, \ldots, x_{n_k}^{(k)}) : P_k \in \mathcal{A}$ and the corresponding conjunction $\bigwedge_{i=1}^{k} P_i(x^{(i)})$ is not satisfiable in $\mathcal{D}$. This can be decided because $\mathcal{D}$ is required to be admissible.

An ABox $\mathcal{A}$ is called *complete* if $\mathcal{A}$ is not contradictory and if, furthermore, none of the completion rules given in Definition 10 is applicable to $\mathcal{A}$.

The clash rules are identical to those used for $\mathcal{ALC}(\mathcal{D})$ (see [5]). Most of the completion rules also appear in the algorithm for deciding the consistency of $\mathcal{ALC}(\mathcal{D})$ ABoxes. The new rules are Rr$\exists$P and RChoose. The use of the *filler?* function in the R$\exists$C and R$\forall$C rules is also new and was necessary because the new role-forming operator is introduced. In the following a detailed explanation of all novelties as compared to the completion rules needed for $\mathcal{ALC}(\mathcal{D})$ is given. First consider the rules R$\exists$C and R$\forall$C. The employment of the *filler?* function can be understood as follows. In $\mathcal{ALCRP}(\mathcal{D})$, there are two kinds of role filler relationships. On the one hand, there are the usual explicit relationships expressed by axioms of the form $(a, b) : R$. On the other hand, there are implicit relationships which are not explicitly represented by axioms. For instance, let a complex role R be defined as $\exists(f_1)(f_2).P$. Furthermore, let there be an ABox $\mathcal{A}$ that contains only the axioms $\{(a, x) : f_1;\ \ (b, y) : f_2;\ \ (x, y) : P\}$. Then the object b is an R-role filler of the object a in $\mathcal{A}$, although $\mathcal{A}$ does not contain the axiom $(a, b) : R$. The *filler?* function captures both kinds of role filler relationships.

The meaning of the rule Rr$\exists$P is straightforward. If it is known that two objects are related via a complex role, concrete objects can immediately be created as fillers of those feature chains being used inside the complex role operator. It is then also known that the predicate used in the definition of the role holds over the newly created concrete objects and thus appropriate axioms can be added. This is what Rr$\exists$P does.

The meaning of RChoose can be understood as follows. If a set of feature chains is used in the definition of a complex role R and (loosely spoken) there are the appropriate concrete fillers for two objects a and b for all the feature chains in this set, then b might be an R role filler of a. But unless there is an explicit axiom which states that the predicate P used in the definition of the role R holds (or does not hold), it is not known if this is really the case. So, if there is no such axiom, both alternatives have to be tried. This is done by creating two successor ABoxes $\mathcal{A}'$ and $\mathcal{A}''$. In $\mathcal{A}'$, it is asserted that P holds and in $\mathcal{A}''$ it is asserted that $\overline{P}$ holds. If any of these two alternatives is the wrong one, a concrete domain clash will be encountered in the corresponding branch of the ABox tree Υ. Like R$\sqcup$, an application of RChoose creates a branch in Υ. The presence of branching rules like R$\sqcup$ and RChoose has been identified as a major source of complexity (see [12]).

It remains to give a formal description of rule application.

Definition 12 The following procedure takes an ABox $\mathcal{A}_0$ as input and returns *consistent* if $\mathcal{A}_0$ has a model and *inconsistent* if $\mathcal{A}_0$ has no model.

define procedure $\mathcal{ALCRP}(\mathcal{D})$-*ABox-cons*$(\mathcal{A}_0)$
  eliminate forks in $\mathcal{A}_0$
  $\mathcal{S} := \{\mathcal{A}_0\}$
  while a completion rule is applicable to an ABox $\mathcal{A} \in \mathcal{S}$ which is not contradictory **do**
    $\mathcal{S} := (\mathcal{S} \setminus \{\mathcal{A}\}) \cup$ *apply-a-completion-rule*$(\mathcal{A})$
  if there is an ABox $\mathcal{A}_c \in \mathcal{S}$ that is not contradictory
    then return *consistent*
    else return *inconsistent*

Please note that if an ABox $\mathcal{A}_c$ is found as described above, then this ABox is complete since it is not contradictory and no completion rule is applicable. The tree Υ is defined as follows: The nodes in the tree are all ABoxes ever generated by $\mathcal{ALCRP}(\mathcal{D})$-*ABox-cons* plus $\mathcal{A}_0$. The node $\mathcal{A}$ is the parent node of a node $\mathcal{A}'$ in Υ iff $\mathcal{A}'$ has been obtained from $\mathcal{A}$ by the application of exactly one completion rule. By definition of the completion rules it is clear that Υ is a tree.

2.5 Termination

Before soundness and completeness of the algorithm given in Definition 12 are shown, termination is proven. First, the motivation for the definition of the restricted terminologies is explained because the restrictions are reflected in the termination proof.

As already noted, in full $\mathcal{ALCRP}(\mathcal{D})$ neither the finite model property nor the tree model property holds. Thus, by using an appropriate concrete domain, concepts can be defined that are fulfilled by infinite models only. This property makes proving termination of decision algorithms extremely difficult or, in case of undecidability, impossible. In fact, the tableau algorithm given in Definition 12 is, in principle, sound and complete for full $\mathcal{ALCRP}(\mathcal{D})$, but termination cannot be guaranteed. By restricting the syntactic structure of concept terms, the finite model property is gained. The drawback of this approach is limited expressivity, i.e. modeling becomes harder.

As a simple example for the mechanism that leads to non-termination in full $\mathcal{ALCRP}(\mathcal{D})$ consider the following initial ABox: $\{o_0 : \forall R_c.\exists R.D,\ (o_0, o_1) : R_c\}$. Using the value restriction over the complex role R_c with the rule R$\forall$C, the concept term $\exists R.D$ is "propagated" to the object o_1. Then, the application of the R$\exists$C rule creates an object o_2 along with the axiom $(o_1, o_2) : R$. Now assume that the concept term D has the form $\exists u_1, \ldots, u_n.P$. This means that concrete objects are generated as fillers of some features of o_2 which may lead to the inference of a new, implicit role filler relationship between o_0 and the newly created object o_2. Again, the value restriction can be used to propagate the exists restriction along the role filler relationship just inferred to the object o_2. Thus, a cycle is obtained. To eliminate this possibility, one has to prevent the generation of concrete fillers for the features of o_2. Concrete objects are only created by the rules R$\exists$P and Rr$\exists$P. The rule R$\exists$P can only be applied if there is a concept-forming predicate operator $(\exists u_1, \ldots, u_n.P)$ inside the concept term D. The rule Rr$\exists$P can only be applied if the concept term C contains an exists restriction quantifying over a complex role $(\exists(\exists(u_1, \ldots, u_n)(v_1, \ldots, v_m).P).X)$. Summarizing, concept-forming predicate operators inside of value restrictions quantifying over complex roles and also nestings of value and exists restrictions which both quantify over complex roles have to be prohibited. The following termination proof depends on these restrictions.

Proposition 13 The $\mathcal{ALCRP}(\mathcal{D})$-*ABox-cons* algorithm terminates on any input that is restricted.

Consider the tree of ABoxes Υ. Since any completion rule is generating only finitely many successors, the tree is branching finitely. If it can additionally be shown that any branch of Υ has finite length, then the termination follows by König's Lemma.

Lemma 14 The tree of ABoxes Υ computed by the $\mathcal{ALCRP}(\mathcal{D})$-*ABox-cons* algorithm has no infinite branch.

The rest of Section 2.5 contains the proof of Lemma 14. It will be shown that there can be no infinite sequence of ABoxes $\mathcal{A}_0, \mathcal{A}_1, \ldots$ where $\mathcal{A}_{k+1}$ is obtained from $\mathcal{A}_k$ by the application of a completion rule. Such a sequence corresponds to a branch in Υ that begins at the root of the tree. Before the proof starts, some definitions are needed.

An axiom that is already present in the initial ABox $\mathcal{A}_0$ is called *old*. All other axioms are called *new*. We need some upper bounds for the number of concept terms and other entities that can appear during the computation. Let $C_{\mathcal{A}_0}$ be the set of subconcept terms of all concepts appearing in $\mathcal{A}_0$ and let NC be the cardinality of the set $C_{\mathcal{A}_0}$. Let NF be the number of distinct features and $NP(n)$ be twice the number of n-ary distinct predicates occurring in $\mathcal{A}_0$. Twice because the negations may also be used during the computation. Let NR be NF plus the number of distinct atomic roles occurring in $\mathcal{A}_0$ plus the number of distinct role-forming operators in the initial ABox.

The rules R$\exists$C, R$\exists$P, and Rr$\exists$P are called *generating* rules. All other rules are called *non-generating*. The application of any of the generating rules may create forks. It may even be the case that fork elimination identifies all new objects (abstract or concrete) with existing ones. In this case the above rules will, without loss of generality, be considered as non-generating ones.

For every ABox $\mathcal{A}$, a graph $G_{\mathcal{A}} = (V_{\mathcal{A}}, E_{\mathcal{A}})$ is defined as follows. For each abstract or concrete object o in $\mathcal{A}$, the graph contains a vertex $\nu(o)$. The graph contains an edge for each *new* axiom ax that is of the form $(a, b) : R$. Please note that edges in the graph correspond to axioms of the above form where R may be atomic or complex. In the following the term edge will be used to refer to such an axiom. There is a sequence of graphs $G_{\mathcal{A}_0}, G_{\mathcal{A}_1}, \ldots$ corresponding to the sequence of ABoxes $\mathcal{A}_0, \mathcal{A}_1, \ldots$. The graph $G_{\mathcal{A}_0}$ for the ABox $\mathcal{A}_0$ does not contain any edges because there are no new axioms at all in this ABox.

It can be shown that the graphs $G_{\mathcal{A}_i}$ in the sequence have the form of a forest (i.e. a collection of unconnected trees). These trees must not be confused with the tree of ABoxes Υ. The application of a generating rule creates a new object with exactly one incoming edge.[7] No rule creates incoming edges for objects already existing. From these two observations it follows that the graphs $G_{\mathcal{A}_0}, G_{\mathcal{A}_1}, \ldots$ are forests and that, furthermore, all forests contain the same number of trees. Each object in the initial ABox $\mathcal{A}_0$ is the root of one of the trees.

Lemma 15 If the sequence of ABoxes $\mathcal{A}_1, \mathcal{A}_2, \ldots$ is infinite, then at least one of the trees in the corresponding sequence of forests $G_{\mathcal{A}_0}, G_{\mathcal{A}_1}, \ldots$ grows infinitely.

Proof. An application of one of the generating rules R$\exists$C and R$\exists$P leads to the growth of exactly one tree. An application of the generating rule Rr$\exists$P leads to the growth of one or two trees (the latter is the case if the rule is applied to an old axiom). Thus, the application of a generating rule leads to the growth of at least one tree. Because the number of trees is the same in all forests $G_{\mathcal{A}_i}$, infinitely many applications of generating rules lead to the infinite growth of at least one of the trees. Hence, an

[7] An incoming edge of an object o is an axiom of the form $(a, o) : R$.

infinite sequence of ABoxes that corresponds to a sequence of graphs in which no tree grows infinitely can only exist if it is possible to apply the non-generating rules an infinite number of times without applying a generating rule. Assume that this is the case for a given computation. Note that none of the non-generating rules generates objects (neither abstract nor concrete). Furthermore, every rule application adds an axiom to the ABox which was not already present. On the other hand, only finitely many distinct axioms can be asserted for a given, finite number of objects: For any single abstract object, no more than NC distinct axioms can be asserted. For each pair of abstract objects, no more than NR distinct axioms can be asserted. For any pair of an abstract and a concrete object, no more than NF distinct axioms can be asserted. And last, for any n concrete objects, no more than $NP(n)$ distinct axioms can be asserted. This shows that there can be only finitely many applications of non-generating rules without adding new objects. That is a contradiction to the assumption and thus completes the proof of Lemma 15. ∎

In the following it will be shown that none of the trees in the forest sequence $G_{\mathcal{A}_0}, G_{\mathcal{A}_1}, \ldots$ can grow infinitely. From this and Lemma 15 it follows that the algorithm terminates on any restricted input.

Proposition 16 In the sequence of forests $G_{\mathcal{A}_0}, G_{\mathcal{A}_1}, \ldots$, none of the trees grows infinitely.

Proof. It will be shown that for each of the trees there are upper bounds for the branching factor and the depth. Therefore, the trees have an upper bound for the number of nodes and thus cannot grow infinitely. In the following, concrete objects can be safely ignored since an examination of the completion rules reveals that the nodes corresponding to these objects cannot have any successors. ∎

Lemma 17 There is an upper bound for the branching factor of each of the trees in each of the graphs $G_{\mathcal{A}_i}$.

Proof. It has to be shown that each of the generating rules can only be applied finitely many times to a single object. First consider the R∃C rule. Let NE be the number of terms in $C_{\mathcal{A}_0}$ (the set of subconcept terms of all concepts appearing in $\mathcal{A}_0$, see above) that are of the form $\exists R.C$ (with R atomic or complex). For each of these terms the R∃C rule may be applied at most once to a single object because its premise is not fulfilled afterwards. Thus, there can exist at most NE successors of each node generated by the R∃C rule.

Now consider the rules R∃P and Rr∃P. Both rules create feature chains that consist of zero or more abstract objects and one concrete object which is the filler of the last feature. However, features may have at most one filler. If there is more than one filler, fork elimination immediately occurs and the R∃P or Rr∃P rule is considered as non-generating for this particular application. Since the number of features is limited by NF, for each abstract object at most NF successors may be created by applications of the R∃P or Rr∃P rule.

Summing up, any node in the tree has at most $NE + NF$ successors. ∎

Lemma 18 There is an upper bound for the depth of each of the trees in each of the graphs $G_{\mathcal{A}_i}$.

PROOF. First, some definitions are needed. There exists a *phantom edge* between two abstract objects in an ABox $\mathcal{A}$ iff there exists a complex role R in the initial ABox $\mathcal{A}_0$ such that $(a, b) : R$ is not in $\mathcal{A}$ but nevertheless $\mathit{filler?}_{\mathcal{A}}(a, b, R)$ holds (see Section 2.4). Thus, the notion phantom edge resembles the informal notion of an implicit role filler relationship. An application of the R∀C rule to an axiom $\forall R.C$ together with a phantom edge is called *phantom application*. The *level* of an object o is denoted by $\lambda(o)$ and defined as the distance between the root of the tree containing $\nu(o)$ and the node $\nu(o)$ itself.[8] Let RD be the maximum *role depth* of any concept term in $\mathcal{A}_0$, where for each concept term C, the role depth $rd(C)$ is the maximum nesting depth of exists and value restrictions in C. Let CL be the maximum length of a feature chain present in $\mathcal{A}_0$.

Using induction over the number of phantom applications of R∀C, it is proved that the maximum level of any object created by rule application is $2 * RD + CL - 2$, and that, furthermore, the maximum level of abstract objects that have concrete objects attached via feature chains is $RD + CL - 1$. From this, Lemma 18 immediately follows. For the induction start, assume that no phantom applications of R∀C take place. By definition, all objects in the initial ABox $\mathcal{A}_0$ are on level 0. A case distinction according to the rule introducing the new object is made.

R∃C It can be shown that for any axiom of the form $o : C$, which is introduced by rule application, we have $\lambda(o) + rd(C) \leq RD$. The proof is by induction over the number of rule applications and makes a case analysis according to the rule applied. From this it directly follows that the maximum level of objects introduced by R∃C is RD.

Rr∃P The Rr∃P rule is applied to axioms $(a, b) : \exists(u_1, \ldots, u_n)(v_1, \ldots, v_m).P$, which are introduced by the R∃C rule, only. Because of the upper bound established for the level of objects introduced by R∃C, we have $\lambda(a) \leq RD$ and $\lambda(b) \leq RD$. By application of the Rr∃P rule, feature chains attached to the objects a and b may be generated. Their maximum length is, however, CL.

R∃P The R∃P rule is applied to axioms of the form $a : \exists u_1, \ldots, u_n.P$. Just as for the Rr∃P rule, we can safely assume $\lambda(a) \leq RD$. By rule application, feature chains with a maximum length of CL attached to the object a may be created.

From these considerations it follows that the maximum level of any abstract object is $RD + CL$ if no phantom edges are involved. The maximum level of any abstract object having concrete objects attached via feature chains is $RD + CL - 1$.

For the induction step, assume that an ABox is given for which the induction hypothesis holds. This means that a phantom edge may exist between any two objects o_0 and o_1 with $\lambda(o_0) < \lambda(o_1) \leq RD + CL - 1$.[9] This is the case because such objects may have concrete objects attached via feature chains. Consider a single phantom application of the R∀C rule to an axiom $o_0 : \forall R.C$. This leads to the generation of an axiom $o_1 : C$. We examine the application of generating rules after the application of the R∀C rule.

[8] Please note that phantom edges are not part of any tree and hence do not contribute to the level.

[9] For phantom edges between two objects on the same level, the arguments from the last paragraph remain valid.

R∃C From $\lambda(o_1) \leq RD + CL - 1$ and $rd(C) \leq (RD - \lambda(o_0)) - 1 \leq RD - 1$, it follows that $\lambda(o_1) + rd(C) \leq 2 * RD + CL - 2$. Generalizing the induction argument from the previous paragraph, it can be shown that if all axioms $o : D$ in an ABox $\mathcal{A}$ fulfill the equation $\lambda(o) + rd(D) \leq RD + n$, then all axioms of the same form that are generated by rule application do also fulfill this equation. This means that the maximum level of objects newly generated by applications of R∃C is $RD + n$. In the given case, $n = RD + CL - 2$. Hence, the maximum level of any object introduced by the application of the R∃C rule is $2 * RD + CL - 2$. No concrete objects are generated.

R∃P The concept term C may have subconcept terms $\exists u_1, \ldots, u_n.P$. Since $\forall R.C$ has to be in restricted form, the predicate operators in C are not in the scope of any exists or value restrictions. From this fact together with the considerations from the last paragraph, it follows that if there is an axiom that has the form $o : \exists u_1, \ldots, u_n.P$, then the maximum level of o is $max(RD, \lambda(o_1)) \leq RD + CL - 1$. For the case $\lambda(o_1) \leq RD$, see the argument for the induction start. The case $RD < \lambda(o_1) \leq RD + CL - 1$ remains to be treated. From the restrictedness of $\forall R.C$, it follows that in this case all feature chains in the predicate operator have length 1. Hence, no abstract objects are created and the maximum level of all concrete objects created is $RD + CL$.

Rr∃P The presence of phantom edges does not lead to additional applications of the Rr∃P rule, since, because of the restrictedness of $\forall R.C$, the concept term C may not contain any subconcept terms of the form $\exists(\exists(u_1, \ldots, u_n)(v_1, \ldots, v_m).P).X$.

It follows that all ABoxes obtained by rule applications after a single phantom application of the R∀C rule also fulfill the induction hypothesis.

We have just proved that the maximum level of all objects generated by rule application is $2 * RD + CL - 2$. This number is hence also the upper bound of the depth of any of the trees in the graph sequence $G_{\mathcal{A}_0}, G_{\mathcal{A}_1}, \ldots$. ∎

This completes the proof of Proposition 13. This result will be needed in the next section to prove soundness and completeness.

2.6 Soundness and Completeness

The proofs of soundness and completeness given in this section are extensions of those given for $\mathcal{ALC}(\mathcal{D})$ in [5].

Proposition 19 The algorithm presented in Definition 12 is sound and complete.

The purpose of our algorithm is to find inconsistencies in a given ABox $\mathcal{A}$. In order to prove soundness, it has to be shown that an ABox which contains a clash cannot have a model. To prove completeness, it has to be shown that if the algorithm terminates because a complete ABox was computed, then $\mathcal{A}$ has a model. First, local correctness of the rules is established.

Lemma 20 (Local Correctness)

1. If an ABox $\mathcal{A}'$ is obtained from an ABox $\mathcal{A}$ by a rule generating only a single descendent ABox, then $\mathcal{A}$ has a model iff $\mathcal{A}'$ has a model.

2. If two ABoxes $\mathcal{A}'$ and $\mathcal{A}''$ are obtained from an ABox $\mathcal{A}$ by a rule which generates two descendent ABoxes, then $\mathcal{A}$ has a model iff at least one of $\mathcal{A}'$ and $\mathcal{A}''$ has a model.

PROOF. This lemma has to be proven for each completion rule separately. The proof will only be given in detail for the RChoose rule. All other rules can be treated similarly.[10] The two directions are proved separately.

($\Leftarrow$) Let $\mathcal{A}'$ and $\mathcal{A}''$ be obtained from $\mathcal{A}$ by application of the RChoose rule. Let $\mathcal{I}$ be a model for either $\mathcal{A}'$ or $\mathcal{A}''$. Then $\mathcal{I}$ is also a model of $\mathcal{A}$ since we have $\mathcal{A} \subseteq \mathcal{A}'$ and $\mathcal{A} \subseteq \mathcal{A}''$.

($\Rightarrow$) Let $\mathcal{A}'$ and $\mathcal{A}''$ be obtained from $\mathcal{A}$ by application of the RChoose rule to an axiom $ax = a : \forall(\exists(u_1, \ldots, u_n)(v_1, \ldots, v_m).P).C$. $\mathcal{A}'$ differs from $\mathcal{A}$ in having the additional axiom $(x_1, \ldots, x_n, y_1, \ldots, y_m) : P$. $\mathcal{A}''$ differs from $\mathcal{A}$ in having the additional axiom $(x_1, \ldots, x_n, y_1, \ldots, y_m) : \overline{P}$. Let $\mathcal{I}$ be a model of $\mathcal{A}$. Since RChoose was applicable to ax in $\mathcal{A}$, $\mathcal{I}$ maps each of the objects $x_1, \ldots, x_n, y_1, \ldots, y_m$ into $\Delta_\mathcal{D}$. By definition of concrete domain predicates, we must have either $x_1, \ldots, x_n, y_1, \ldots, y_m \in P^\mathcal{D}$ or $x_1, \ldots, x_n, y_1, \ldots, y_m \in \overline{P}^\mathcal{D}$. Hence, $\mathcal{I}$ is either a model for $\mathcal{A}'$ or $\mathcal{A}''$. ∎

Now, the soundness and completeness of the tableau calculus can be proved.

Proposition 21 (Soundness) If the algorithm $\mathcal{ALCRP}(\mathcal{D})$-*ABox-cons* terminates because every leaf in the tree of ABoxes Υ contains a clash, then the initial ABox $\mathcal{A}_0$ does not have a model.

PROOF. It has to be shown that the initial A-Box $\mathcal{A}_0$ does not have a model if all leaves in the tree Υ contain a clash. There are two possible cases. First, $\mathcal{A}_0$ itself could contain a clash which means that no descendent ABoxes have been computed. In this case, a look at the definition of the clash triggers reveals that $\mathcal{A}_0$ cannot have a model. In the second case, $\mathcal{A}_0$ does not contain a clash itself but all the leaf ABoxes in Υ do. Like $\mathcal{A}_0$ in the first case, none of the leaf ABoxes can have a model. In this case, using Lemma 20 it can easily be proven by induction that $\mathcal{A}_0$ cannot have a model as well. ∎

Proposition 22 (Completeness) If the algorithm $\mathcal{ALCRP}(\mathcal{D})$-*ABox-cons* does not terminate with the reason that every leaf in Υ contains a clash, then it terminates because a complete ABox $\mathcal{A}_c$ has been derived. In this case the initial ABox $\mathcal{A}_0$ has a model.

PROOF. It has already been proven that the algorithm terminates on any input. It remains to be shown that $\mathcal{A}_0$ has a model if a complete ABox $\mathcal{A}_c$ was computed. For this purpose, the ABox $\mathcal{A}_c$ is used to define an interpretation $\mathcal{I} = (\Delta_\mathcal{I}, \cdot^\mathcal{I})$ that is also a model:

[10] In fact, for all rules except R∀C, Rr∃P and RChoose, the local correctness follows from the soundness and completeness proofs given in [5].

1. $\Delta_{\mathcal{I}}$ consists of all the objects of O_A which occur in $\mathcal{A}_c$.
2. If C is a concept name then $a \in C^{\mathcal{I}}$ iff $a : C \in \mathcal{A}_c$.
3. If R is a role or feature name then $(a, b) \in R^{\mathcal{I}}$ iff $(a, b) : R \in \mathcal{A}_c$.
4. Because there is no concrete domain clash in $\mathcal{A}_c$, there is a variable assignment α that satisfies the conjunction of all occurring axioms $(x_1, \ldots, x_n) : P$. So we set $x^{\mathcal{I}} = \alpha(x)$ iff $x \in \mathsf{O}_D$.

Since $\mathcal{I}$ is obviously an interpretation function, it can be extended to arbitrary concept and role terms as defined in Definition 4. It has to be shown that $\mathcal{I}$ is a model of the ABox $\mathcal{A}_c$, i.e. it satisfies all assertional axioms in $\mathcal{A}_c$ in the sense of Definition 5. It is then an immediate consequence of Lemma 20 that $\mathcal{I}$ is also a model for $\mathcal{A}_0$.

First consider axioms ax of the form $(a, b) : \exists(u_1, \ldots, u_n)(v_1, \ldots, v_m).P$. Since $\mathcal{A}_c$ is complete, the rule Rr$\exists$P has been applied to ax. Hence, the various axioms of the form $(x, y) : f$ generated by Rr$\exists$P are in $\mathcal{A}_c$, and, by definition of $\mathcal{I}$, this means that we have $u_i^{\mathcal{I}}(a) = x_i^{\mathcal{I}}$ as well as $v_j^{\mathcal{I}}(b) = y_j^{\mathcal{I}}$ for $i = 1, \ldots, n$ and $j = 1, \ldots, m$. Furthermore, the axiom $(x_1, \ldots, x_n) : P$ is in $\mathcal{A}_c$, and, by definition of α, it is satisfied by $\mathcal{I}$. From this it follows that $\mathcal{I}$ satisfies ax.

All remaining axioms from $\mathcal{A}_c$ that $\mathcal{I}$ needs to satisfy are of the form $a : C$. The rest of the proof can be done by induction on the size of these axioms in $\mathcal{A}_c$, where the size of a concept term X is simply the number of atomic concepts and operators used in X. A case analysis according to the topmost operator in C has to be made. Most cases are already treated in [5]. The only case that is different is that the considered axiom ax is of the form $\forall R.C$.

Let ax be of the form $a : \forall R.C$. First assume that R is an atomic or complex role and $(a, b) : R$ is in $\mathcal{A}_c$. Then b is in O_A because there is no all domain clash in $\mathcal{A}_c$. Then, for ax together with $(a, b) : R$ the rule R$\forall$C has been applied. By induction, $\mathcal{I}$ satisfies $b : C$ for all these b and hence ax is satisfied. Now assume R is of the form $\exists(u_1, \ldots, u_n)(v_1, \ldots, v_m).P$ and $(a, b) : R$ is not in $\mathcal{A}_c$ but there exist objects x_1, $\ldots$,x_n, y_1, $\ldots$,y_m in O_D such that $u_i^{\mathcal{I}}(a) = x_i^{\mathcal{I}}$ and $v_j^{\mathcal{I}}(b) = y_j^{\mathcal{I}}$ holds for $i = 1, \ldots, n$ and $j = 1, \ldots, m$, respectively (i.e. possibly there exists a phantom edge between a and b). Then b is in O_A because there are axioms of the form $(b, x) : f$. Furthermore, there has to be exactly one of the two axioms $(x_1, \ldots, x_n, y_1, \ldots, y_m) : P$ and $(x_1, \ldots, x_n, y_1, \ldots, y_m) : \overline{P}$ in $\mathcal{A}_c$, since the rule RChoose is not applicable and there is no concrete domain clash in $\mathcal{A}_c$. If the first of these two axioms is in $\mathcal{A}_c$, the rule R$\forall$C has been applied. Like above, the axiom ax is satisfied by $\mathcal{I}$. In the other case, b is not a filler of the role R for a. ∎

This finishes the completeness proof of the *$\mathcal{ALCRP}(\mathcal{D})$-ABox-cons* algorithm. In order to use $\mathcal{ALCRP}(\mathcal{D})$ for knowledge representation in general and for spatial and temporal reasoning in particular, an admissible concrete domain must be defined. This will be discussed in the next section.

3 Spatial and Temporal Reasoning

In this section we demonstrate that $\mathcal{ALCRP}(\mathcal{D})$ is an appropriate formalism for spatial and spatiotemporal reasoning when instantiated with an appropriate concrete domain. First, such a concrete domain is defined and its admissibility is proved. Then, the combined formalism is motivated as a sound basis for knowledge representation and spatial query processing in GIS. We conclude this section with a proposal for additionally representing spatiotemporal phenomena.

3.1 Representing Spatial Information

This section introduces the concrete domain $\mathcal{S}_2$ which can be used for the representation of two-dimensional spatial objects. $\mathcal{S}_2$ provides predicates that allow one to describe qualitative spatial relationships between spatial objects. We prove that the concrete domain $\mathcal{S}_2$ is admissible and hence can be used for integrated spatial and conceptual reasoning in connection with the description logic $\mathcal{ALCRP}(\mathcal{D})$.

3.1.1 The Concrete Domain $\mathcal{S}_2$

Before the concrete domain $\mathcal{S}_2$ can be formally defined, some notions from point set topology and qualitative spatial reasoning need to be introduced. First, we define the basic notion "topology" (see, e.g., [39]).

Definition 23 Let $\mathcal{U}$ be a set. A *topology* on $\mathcal{U}$ is a family T of subsets of $\mathcal{U}$, with

1. if $O_1, O_2 \in T$, then $O_1 \cap O_2 \in T$,
2. if $O_i \in T$ for $i \in I$, then $\bigcup O_i \in T$,
3. $\emptyset, \mathcal{U} \in T$.

The pair $\langle \mathcal{U}, T \rangle$ is called a *topological space*. The elements of T are called *open subsets* of $\mathcal{U}$.

Some more definitions from point set topology are needed, although we will only consider a particular topology on $\mathbb{R}^2$ for the definition of $\mathcal{S}_2$.

Definition 24 Let $\langle \mathcal{U}, T \rangle$ be a topological space, let M be a subset of $\mathcal{U}$, and let x be a point in $\mathcal{U}$.

- M is *closed* if $\mathcal{U} \setminus M$ is open.
- A set $N \subset \mathcal{U}$ is called *neighborhood* of x if there is an open subset $O \subset \mathcal{U}$ such that $x \in O \subset N$.
- x is called an *interior point* of M if there is a neighborhood N of x contained in M. The set of all interior points of M is called the *interior* of M and is denoted by $i(M)$.
- x is called an *exterior point* of M if there is a neighborhood N of x that contains no point in M. The set of all exterior points of M is called the *exterior* of M and is denoted by $e(M)$.

- x is called a *boundary point* of M if every neighborhood N of x contains at least one point in M and one point not in M. The set of all boundary points of M is called the *boundary* of M and is denoted by $b(M)$.
- The *closure* of M is the smallest closed set which contains M (i.e., $M \cup b(M)$), and is denoted by $\widehat{M}$.
- M is *regular closed* if $\widehat{i(M)}$ is equal to M.

The RCC theory is an axiomatic theory for qualitative spatial reasoning [32]. As part of RCC theory, a set of 8 "topological" relations called RCC-8 has been defined.[11] These relations describe all possible relationships that may hold between any two subsets of a topological space. In the following, the RCC-8 relations are introduced w.r.t. topological spaces.

Definition 25 Let $\langle \mathcal{U}, T \rangle$ be a topological space. A *spatial relation* R is a subset of $\mathcal{U} \times \mathcal{U}$. Let M_1 and M_2 be subsets of $\mathcal{U}$. RCC-8 is a set of 8 spatial relations defined as follows:

$$(M_1, M_2) \in DC \text{ iff } \widehat{M_1} \cap \widehat{M_2} = \emptyset$$
$$(M_1, M_2) \in EC \text{ iff } i(M_1) \cap i(M_2) = \emptyset \ \wedge \ \widehat{M_1} \cap \widehat{M_2} \neq \emptyset$$
$$(M_1, M_2) \in PO \text{ iff } i(M_1) \cap i(M_2) \neq \emptyset \ \wedge \ \widehat{M_1} \setminus \widehat{M_2} \neq \emptyset \ \wedge \ \widehat{M_2} \setminus \widehat{M_1} \neq \emptyset$$
$$(M_1, M_2) \in EQ \text{ iff } \widehat{M_1} = \widehat{M_2}$$

$$(M_1, M_2) \in TPP \text{ iff } \widehat{M_1} \cap \widehat{M_2} = \widehat{M_1} \ \wedge \ \widehat{M_1} \neq \widehat{M_2} \ \wedge \ b(M_1) \cap b(M_2) \neq \emptyset$$
$$(M_1, M_2) \in NTPP \text{ iff } \widehat{M_1} \cap \widehat{M_2} = \widehat{M_1} \ \wedge \ \widehat{M_1} \neq \widehat{M_2} \ \wedge \ b(M_1) \cap b(M_2) = \emptyset$$
$$(M_1, M_2) \in TPPI \text{ iff } (M_2, M_1) \in TPP$$
$$(M_1, M_2) \in NTPPI \text{ iff } (M_2, M_1) \in NTPP$$

An *RCC-8 formula* is an expression of the form XRY where R is one of the RCC-8 relations or a disjunction of RCC-8 relations. A set of RCC-8 formulas N is called *RCC-8 network*. The set of variables used in an RCC-8 network N is denoted by $Var(N)$. An RCC-8 network N is *satisfiable* iff there exists a topological space $\langle \mathcal{U}, T \rangle$ and a mapping δ from $Var(N)$ to the set of all non-empty, regular closed subsets of $\mathcal{U}$ such that for all RCC-8 formulas XRY in N, we have $(\delta(X), \delta(Y)) \in R$.

Figure 1 gives examples of the RCC-8 relations in the plane. Based on the notions just introduced, the concrete domain $\mathcal{S}_2$ can be defined.

Definition 26 Consider the topological space $\langle \mathbb{R}^2, 2^{\mathbb{R}^2} \rangle$, i.e. the topology is the set of all subsets of $\mathbb{R}^2$. The concrete domain $\mathcal{S}_2$ is defined w.r.t. this topological space. The domain $\Delta_{\mathcal{S}_2}$ contains all non-empty, regular closed subsets of $\mathbb{R}^2$. The elements of $\Delta_{\mathcal{S}_2}$ are called *regions*. The set $\Phi_{\mathcal{S}_2}$ contains predicates which are defined as follows:

[11] Egenhofer defines relations with exactly the same properties [13] in a more limited framework.

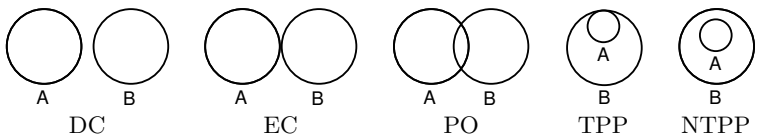

Figure 1: Elementary relations between two regions A and B. The inverses of TPP and NTPP as well as the relation EQ have been omitted.

- A unary predicate is-region with $\mathsf{is\text{-}region}^{\mathcal{S}_2} = \Delta_{\mathcal{S}_2}$ and its negation is-no-region with $\mathsf{is\text{-}no\text{-}region}^{\mathcal{S}_2} = \emptyset$.
- A binary predicate inconsistent-relation with $\mathsf{inconsistent\text{-}relation}^{\mathcal{S}_2} = \emptyset$.
- The 8 basic predicates dc, ec, po, tpp, ntpp, tppi, ntppi and eq correspond to the RCC-8 relations and are defined as follows. Let r_1 and r_2 be two regions. We have $(r_1, r_2) \in \mathsf{dc}^{\mathcal{S}_2}$ iff $(r_1, r_2) \in DC$, $(r_1, r_2) \in \mathsf{ec}^{\mathcal{S}_2}$ iff $(r_1, r_2) \in EC$, ... , $(r_1, r_2) \in \mathsf{ntppi}^{\mathcal{S}_2}$ iff $(r_1, r_2) \in NTPPI$, ...
- For each distinct set $\{p_1, \ldots, p_n\}$ of basic predicates, where $n \geq 2$, an additional predicate of arity 2 is defined. The predicate has the name $\mathsf{p_1\text{-}\cdots\text{-}p_n}$ and we have $(r_1, r_2) \in \mathsf{p_1\text{-}\cdots\text{-}p_n}^{\mathcal{S}_2}$ iff $(r_1, r_2) \in \mathsf{p_1}^{\mathcal{S}_2}$ or ... or $(r_1, r_2) \in \mathsf{p_n}^{\mathcal{S}_2}$ is true. In total, there are $2^8 - 9$ of these combined predicates. We impose the following canonical order on the basic predicate names (i.e. dc, ec, po, tpp, ntpp, tppi, ntppi, eq) that is applied to form unique predicate names for these combined binary predicates.

Please note that the elements of $\Delta_{\mathcal{S}_2}$ are regions that are not necessarily internally connected, i.e. for any region $r \in \Delta_{\mathcal{S}_2}$, there may exist two regions s_1 and s_2 such that $r = s_1 \cup s_2$ and $s_1 \cap s_2 = \emptyset$. As an example for a combined predicate consider tppi-ntppi-eq. For two regions r_1 and r_2, we have $(r_1, r_2) \in \mathsf{tppi\text{-}ntppi\text{-}eq}^{\mathcal{S}_2}$ if and only if $(r_1, r_2) \in \mathsf{tppi}^{\mathcal{S}_2}$, or $(r_1, r_2) \in \mathsf{ntppi}^{\mathcal{S}_2}$, or $(r_1, r_2) \in \mathsf{eq}^{\mathcal{S}_2}$.

3.1.2 Admissibility of $\mathcal{S}_2$

In this section, we prove the admissibility of $\mathcal{S}_2$. It needs to be shown that the set of predicates $\Phi_{\mathcal{S}_2}$ is closed under negation and that the satisfiability of finite predicate conjunctions is decidable.

Lemma 27 The satisfiability of finite conjunctions of predicates from $\Phi_{\mathcal{S}_2}$ is decidable.

Proof. Checking the satisfiability of finite conjunctions of predicates can be reduced to checking the consistency of RCC-8 networks. Let a finite conjunction $C = p_1(x_1^1, \ldots, x_{n_1}^1) \wedge \cdots \wedge p_k(x_1^k, \ldots, x_{n_k}^k)$ of predicates from $\Phi_{\mathcal{S}_2}$ be given. Its satisfiability can be decided as follows.

- If for any $i = 1..k$, p_i is either is-no-region or $p_i =$ inconsistent-relation, then return *unsatisfiable*.

- Remove any conjunct with $p_i = \mathsf{is\text{-}region}$. All predicates in the remaining conjunction have arity 2.
- Translate the remaining conjunction C' into an RCC-8 network N as follows: The variables $Var(N)$ are exactly the variables occurring in C'. Consider each conjunct $p_i(x_1^i, x_2^i)$ from C' separately. The predicate p_i has the form $\mathsf{p_1\text{-}\cdots\text{-}p_n}$ with $n \geq 1$. Let $R = R_1 \vee \cdots \vee R_n$ be the corresponding disjunction of RCC-8 relations. Make a case distinction as follows: (i) if there is no RCC-8 formula $x_1^i S x_2^i$ in N, then add $x_1^i R x_2^i$; (ii) let there be a RCC-8 formula $x_1^i S x_2^i$ in N, where S is a disjunction of RCC-8 relations. Let $(R \cap S)$ denote the disjunction of those relations that appear in both R and S. If there is no such relation, return *inconsistent*. Otherwise, remove the formula $x_1^i S x_2^i$ from N and add the new formula $x_1^i (R \cap S) x_2^i$.
- For all pairs $(r_1, r_2) \in Var(N)$, for which there is no formula $r_1 R r_2$ in N, add the formula $r_1 \ sr \ r_2$, where sr ("spatially related") is the disjunction of all RCC-8 relations.
- Check the satisfiability of the set N and return the result.

In [34], it is proved that deciding the consistency of RCC-8 networks is an NP-complete problem. Efficient methods for this task are described in [35]. It still needs to be shown that C is satisfiable if and only if N is satisfiable. The *only if* direction is trivial by the definition of satisfiability of RCC-8 networks together with the definition of the predicates in $\Phi_{\mathcal{S}_2}$. The *if* direction follows directly from a theorem given by Renz:

> Every consistent set of spatial formula[12] can be realized in any dimension $d \geq 1$ where regions are (sets of) d-dimensional polytopes [33, Theorem 5.5].

Please note that two-dimensional polytopes are polygons and that $\Delta_{\mathcal{S}_2}$ is a superset of all regions that are describable by a finite set of polygons. ∎

Using this lemma, the main result of this section can be established.

Proposition 28 The concrete domain $\mathcal{S}_2$ is admissible.

PROOF. It will be shown that $\Phi_{\mathcal{S}_2}$ is closed under negation. Together with Lemma 27, this implies the proposition. It is well-known that the RCC-8 relations are mutually exclusive and exhaustive [32]. Given this, it is easy to verify that $\Phi_{\mathcal{S}_2}$ is closed under negation: Let P be the set of basic predicates. For any set of predicates $\{p_1, \ldots, p_n\} \subseteq P$ with $1 \leq n < 8$, we have that $\overline{\mathsf{p_1\text{-}\cdots\text{-}p_n}} = \mathsf{s_1\text{-}\cdots\text{-}s_k}$, where $\{s_1, \ldots, s_k\}$ is defined as $P \setminus \{p_1, \ldots, p_n\}$. The predicate $\mathsf{s_1\text{-}\cdots\text{-}s_k}$ is obviously in $\Phi_{\mathcal{S}_2}$. Note that the negation of the disjunctive combination of all 8 basic predicates is $\mathsf{inconsistent\text{-}relation}$ and vice versa. ∎

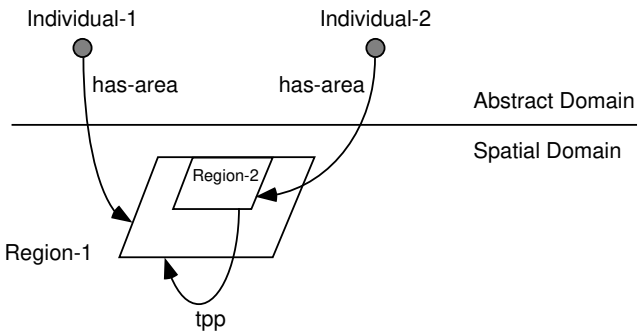

Figure 2: References from the abstract domain into the concrete domain $\mathcal{S}_2$.

3.2 Spatial and Terminological Reasoning

How can predicates over the concrete domain $\mathcal{S}_2$ be used to support spatioterminological inferences with the description logic $\mathcal{ALCRP}(\mathcal{S}_2)$? First of all, as an ontological commitment, we assume that each abstract domain object is associated with its spatial representation via a feature has-area (see Figure 2). We define the roles ntppi and dc for relating individuals in the abstract domain based on the corresponding predicates in $\Phi_{\mathcal{S}_2}$. Then, we give terminological axioms for two different kinds of cities: city-1 and city-2.

$\mathbf{ntppi} \doteq \exists\,(\mathsf{has\text{-}area})(\mathsf{has\text{-}area})\,.\,\mathsf{ntppi}$

$\mathbf{dc} \doteq \exists\,(\mathsf{has\text{-}area})(\mathsf{has\text{-}area})\,.\,\mathsf{dc}$

$\mathbf{city\text{-}1} \doteq \mathsf{city} \sqcap (\exists\,\mathsf{ntppi}\,.\,\mathsf{center}) \sqcap (\exists\,\mathsf{dc}\,.\,\mathsf{suburb})$

$\mathbf{city\text{-}2} \doteq \mathsf{city} \sqcap \exists\,\mathsf{ntppi}\,.\,(\mathsf{center} \sqcap \exists\,\mathsf{dc}\,.\,\mathsf{suburb})$

Both kinds of cities contain a center (topological relation ntppi). In addition, in the former concept definition there exists a suburb which is disconnected (topological relation dc) from the city. In the second definition the existing suburb is disconnected only from the center (but not to the city as a whole). In Figure 3, models for the two concepts are given using a graph notation. Edges drawn with full lines correspond to existential quantification over roles in the concept definitions. Dashed lines between abstract objects represent predicates holding between the associated has-area features of the objects. If we require the suburb to be disconnected from the city, it is certainly disconnected from a region inside the city (in this case the center region). However, if a concept requires the existence of a suburb which is only disconnected from the center, then the suburb can be in any of the indicated relations to the city as a whole (see the lower parts in Figure 3). This set of possible topological relations in the example city-2 can easily be verified using an RCC-8 relation composition table (see e.g. [10]). By considering Figure 3, we see that the concept city-2 subsumes the concept city-1 because the predicate dc-ec-po-tppi-ntppi represents a disjunction

[12] "Spatial formula" is another notion for RCC-8 formula.

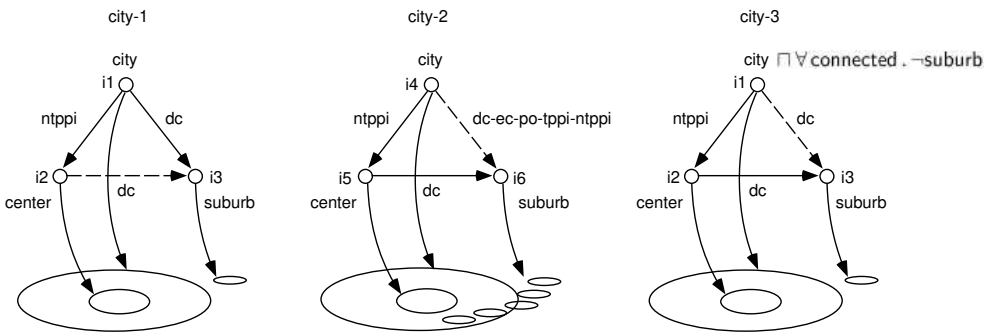

Figure 3: Models for concepts city-1 and city-2 with inferred role filler relationships (see text). In the lower part, examples for the spatial regions of the corresponding objects in the upper part are given. The regions are assumed to be associated with the corresponding abstract object with the feature has-area.

of spatial relations including dc. As desired, the tableau algorithm for $\mathcal{ALCRP}(\mathcal{S}_2)$ applied to the concept city-1 $\wedge \neg$city-2 returns "inconsistent."

The next example demonstrates the use of value restrictions over defined roles. We define another city concept as a specialization of city-2 with the additional restriction that all individuals which are connected are no suburbs.

connected $\doteq \exists\,$(has-area)(has-area) . ec-po-tpp-ntpp-tppi-ntppi-eq

city-3 $\doteq$ city-2 $\sqcap \forall$ connected . $\neg$suburb

We define a role connected using a predicate which models the disjunction of all topological base relations except dc (disconnected). The concept city-3 is defined as a city-2 with an additional value restriction for the role connected. Applying the tableau calculus reveals that the concept city-3 is also more specific than city-1. Due to the value restriction for the role connected (see the concept definition of city-3) we can infer that only dc can hold between the city and the suburb. This and the additional restriction for the city (see Figure 3) are the reasons that city-3 is more specific than city-1.

With defined roles —and value restrictions over defined roles in particular— $\mathcal{ALCRP}(\mathcal{D})$ is more expressive than $\mathcal{ALC}(\mathcal{D})$. The additional expressiveness is required for adequately modeling terminological and spatial knowledge using concept terms. The importance of capturing aspects of space with concept terms was demonstrated by considering the non-obvious subsumption relationships which are detected by TBox inference algorithms based on a sound and complete concept consistency test.

3.3 Representing Spatiotemporal Phenomena

Considering the general mechanism for integrating concrete domains, it becomes clear that another instance of $\mathcal{ALCRP}(\mathcal{D})$ can deal with qualitative temporal relations between time intervals according to [1]. The idea is to define a concrete domain $\mathcal{T}_{IA}$

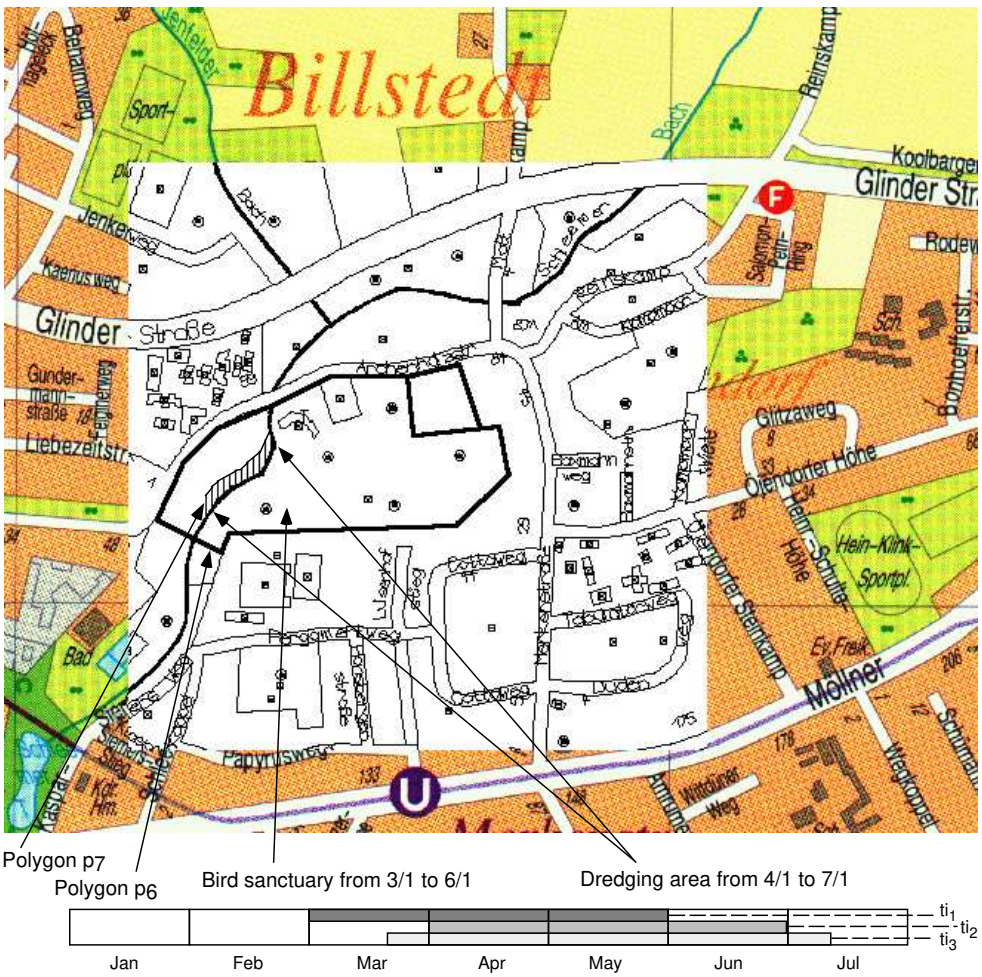

Figure 4: A clip from a city map (see text).

for representing time intervals with two-place predicates representing Allen's Interval Algebra (before, after, meets, met-by, overlaps, overlapped-by, during, contains, starts, started-by, finishes, finished-by, equal and disjunctions of these basic predicates). Constraint satisfaction algorithms known from the literature (see e.g. [15] for an overview) can then be employed to check the satisfiability of conjunctions of predicates. Furthermore, in Section 2.2 it is shown that, from a technical perspective, any two disjoint admissible concrete domains can be combined to form a single admissible concrete domain. The combination $\mathcal{S}_2 \oplus' \mathcal{T}_{IA}$ (see Definition 6) also defines predicates as conjunctions of predicates from the component concrete domains, i.e. the spatial and the temporal concrete domain.

The use of the temporal domain will be illustrated with an example from the GIS application introduced in Figure 4. We assume that, in the GIS database, a certain area is defined to be a bird sanctuary from the beginning of March to the end of

May (see the dark-gray bar in Figure 4). Now, as part of a planning scenario, let us assume a hypothetical dredging operation is to be scheduled from April to June inclusive (middle-gray bar). In Figure 4 the affected area of the creek "Schleemer Bach" is indicated by hatching. In addition, we assume that dredging a creek involves handling trucks and dredging machinery. Thus, we assume the existence of a so-called "support area" touching the real dredging area in the creek (the topological relation is ec). Furthermore, background knowledge should indicate that the "support process" starts earlier and lasts longer than the real dredging operation because the dredging machinery has to be installed and deinstalled (see the light-gray bar in Figure 4 which is used as an example here). Obviously, as the reader might expect, the spatiotemporal constraints, the conceptual knowledge and the knowledge about individuals in this example suggest that dredging in an "active" bird sanctuary should not be sanctioned by the planning module of the GIS. The question is: How can this planning problem using GIS facilities be solved by knowledge representation techniques and logical inferences? In the following we describe a solution with the description logic $\mathcal{ALCRP}(\mathcal{S}_2 \oplus' \mathcal{T}_{IA})$.

Both domain objects, the bird sanctuary and the dredging operation, are represented as ABox objects. The idea is to show that the corresponding ABox can be proven to be inconsistent given the constraints modeling the knowledge informally introduced in the previous section.

As an ontological decision, let us assume that temporal intervals are associated with individuals via the feature has-duration. We can define a spatiotemporal-process as a process for which an interval and a region are existing as fillers for the corresponding features. The predicate is-interval is assumed to check membership in the corresponding concrete domain $\mathcal{T}_{IA}$ (see the admissibility criterion in Definition 1).

spatiotemporal-process $\doteq$ process $\sqcap \exists$ has-duration . is-interval $\sqcap \exists$ has-area . is-region

noisy-process $\sqsubseteq$ process

dredging-support-process $\sqsubseteq$ spatiotemporal-process $\sqcap$ noisy-process

In addition, several auxiliary concepts are introduced to capture the noisiness of a dredging support process. These concepts are only partially defined with inclusion axioms. The concept spatiotemporal-process is used in subsequent terminological inclusion axioms for more specific spatiotemporal processes. The interaction of space and time in the example can be represented by introducing two roles ec-during and ntppi-overlaps which relate fillers of corresponding has-area and has-duration features.

ec-during $\doteq \exists$ (has-area, has-duration)(has-area, has-duration) . ec-during

ntppi-overlaps $\doteq \exists$ (has-area, has-duration)(has-area, has-duration) . ntppi-overlaps

connected-ends-during-overlaps $\doteq$
$\exists$ (has-area, has-duration)(has-area, has-duration) . connected-ends-during-overlaps

The predicate ec-during ensures that (i) the constraint ec is imposed on the fillers of the has-area features and (ii) the constraint during describes a relation between the fillers of the has-duration features. With ntppi-overlaps the corresponding constraints are established between the has-area and has-duration features, respectively

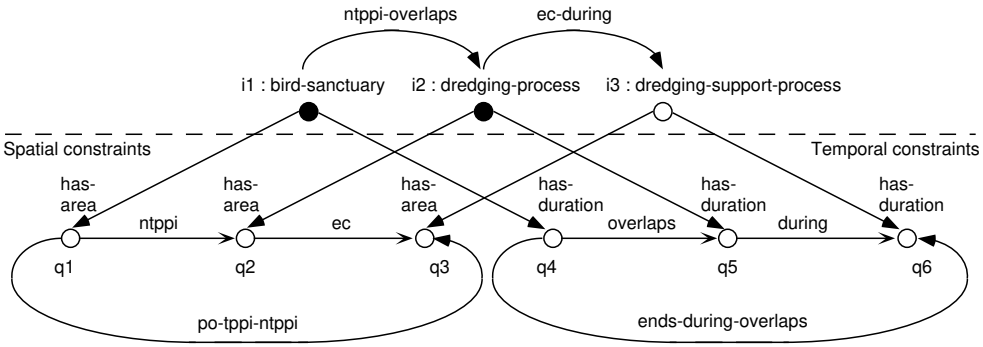

Figure 5: ABox for testing the satisfiability of the dredging process query with implicit spatial and temporal information shown in two constraint nets (see text).

(see Section 2.2). In a similar way, we also define a role connected-ends-during-overlaps combining spatial connectedness and generalized temporal overlapping. Thus, the predicates from both domains simultaneously hold for a pair of abstract individuals. Now, we can (partially) define two concepts dredging-process and bird-sanctuary.

$$\textbf{dredging-process} \sqsubseteq \mathsf{spatiotemporal\text{-}process} \sqcap \exists\, \mathsf{ec\text{-}during} \,.\, \mathsf{dredging\text{-}support\text{-}process}$$

$$\textbf{bird-sanctuary} \sqsubseteq$$
$$\mathsf{spatiotemporal\text{-}process} \sqcap \neg \mathsf{noisy\text{-}process} \sqcap$$
$$\forall\, \mathsf{connected\text{-}ends\text{-}during\text{-}overlaps} \,.\, \forall\, \mathsf{connected} \,.\, \neg \mathsf{noisy\text{-}process}$$

The spatiotemporal role ec-during is used to relate the dredging-support-process in the appropriate way. A bird sanctuary is modeled as a non-noisy spatiotemporal-process with all fillers of the spatiotemporal role connected-ends-during-overlaps being objects which, in turn, are connected to non-noisy processes. The idea behind this definition is that temporally overlapping connected objects must not be spatially connected to noisy processes because this would cause an inconsistency. The $\mathcal{ALCRP}(\mathcal{D})$ restrictedness criterion is fulfilled for this terminology.

In order to represent GIS information concerning the bird sanctuary we simply add the following ABox axiom for an individual i1 which is assumed here to represent the available information about a particular bird sanctuary stored in the GIS. As a hypothesis for the dredging process, additional ABox axioms for the individual i2 and its spatiotemporal relationship with i1 (see also Figure 4) are used.[13]

$$\mathcal{A}_0 := \big\{ \mathsf{i1} : \mathsf{bird\text{-}sanctuary},\ \mathsf{i2} : \mathsf{dredging\text{-}process},\ (\mathsf{i1}, \mathsf{i2}) : \mathsf{ntppi\text{-}overlaps} \big\}$$

The ABox $\mathcal{A}_0$ is checked for consistency. In Figure 5 a graphical representation of the ABox $\mathcal{A}_0$ is shown. Due to the constraints given in ABox $\mathcal{A}_0$, an individual i3 is generated. For this individual, the constraint dredging-support-process is asserted.

[13] Due to space constraints, we neglect the technical details about the GIS implementation and its relation to the ABox of the DL system.

Furthermore, additional constraints resulting from the given spatiotemporal relationship between i1 and i2 are derived (see the role ec-during and the spatial and temporal constraints in Figure 4). The filler of i1 for connected-ends-during-overlaps is i2 in this ABox (see Figure 4 and 5). Due to the value restriction for this role in the concept bird-sanctuary, new constraints are added to i2. The value restriction over connected is asserted. The fillers of the connected role with respect to i2 are i1, i3 and i2 itself.[14] We consider i3. The constraint ¬noisy-process is added to i3. However, this causes a clash because i3 is a dredging-support-process which is a noisy-process by definition. Thus, due to the spatiotemporal constraints, there is no way to find a consistent tableau and, therefore, ABox $\mathcal{A}_0$ is inconsistent. The reader can easily verify that the existence of i3 (the dredging-support-process) is indeed required to make the ABox inconsistent because, in our example, the dredging process itself is not assumed to be a noisy process. If this were the case, there would be another inconsistency concerning i2, of course.

The example shows that the interaction between spatiotemporal and conceptual knowledge is very important for GIS systems. The inconsistency of the ABox $\mathcal{A}_0$ can be interpreted by the application system in such a way that the dredging operation should better be planned in another period of the year.[15] Note that ABox reasoning cannot easily be replaced by model-checking because the implicit processes (see the exists constraint in the definition of dredging-process) are not given as part of the input to the GIS system. For the application system, the knowledge about a necessary dredging-support-process need not be directly available. Furthermore, in $\mathcal{ALCRP}(\mathcal{D})$ we have seen that with defined roles, the interaction between different concrete domains provides a powerful modeling technique which is not available in $\mathcal{ALC}(\mathcal{D})$.

In the example ABox $\mathcal{A}_0$ we have explicitly added the spatiotemporal constraint ntppi-overlaps between i1, the bird-sanctuary, and the dredging-process i2. This is required because the corresponding ABox assertion cannot be inferred. In principle, we have to add similar constraints for every pair of spatiotemporal processes. So, could we find a way to avoid this? In Figure 4 "explicit" regions and time intervals are indicated. We could use the concept-forming predicate operator together with additional predicates from an extended concrete domain in order to represent quantitative knowledge about regions and time intervals with explicit coordinates and time points, respectively. As we will see in the next section, the satisfiability algorithm for conjunctions of predicates from this kind of concrete domain must be extended as well.

3.4 An Extension of $\mathcal{S}_2$

In this section, an extension to the concrete domain $\mathcal{S}_2$ is developed which allows for the integration of qualitative and quantitative spatial reasoning. Polygons may be used to describe actual regions in the plane. Additional predicates are defined that allow one to integrate polygons into the description of spatial situations. First, polylines need to be defined.

[14] The predicate of the role connected subsumes the predicates ntppi, ec and po-tppi-ntppi.

[15] It would be attractive to compute alternatives for the has-duration intervals but this is beyond the scope of this article.

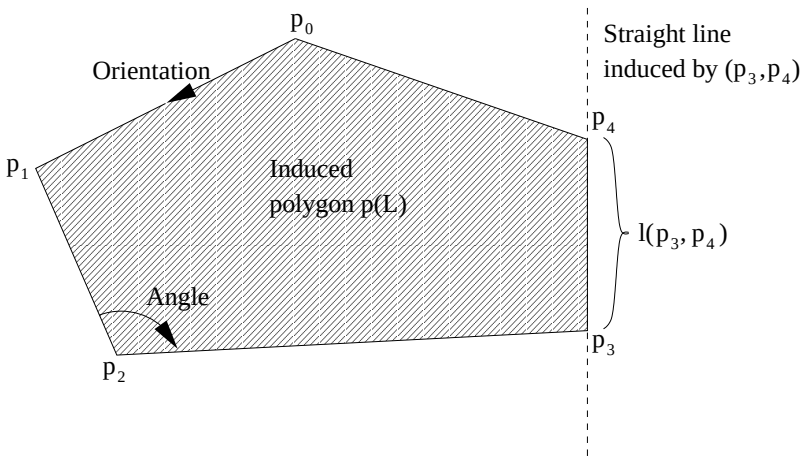

Figure 6: A polygon described by a polyline $L = p_0, \ldots, p_4$.

Definition 29 A pair (p_1, p_2) of two points $p_1, p_2 \in \mathbb{R}^2$ is called an *edge*. An edge $e = (p_1, p_2)$ uniquely describes a straight line in $\mathbb{R}^2$. Let $l(e)$ be the set of all points that are on the straight line induced by e, and, furthermore, are located between p_1 and p_2. The set $l(e)$ does also include the points p_1 and p_2 themselves.

Two edges e_1 and e_2 are *crossing* iff $l(e_1) \cap l(e_2) \neq \emptyset$. For two edges e_1 and e_2, $\varphi(e_1, e_2)$ denotes the (counterclockwise) angle between the two straight lines described by e_1 and e_2.

A sequence of points $S = p_1, \ldots, p_n$ corresponds to an edge sequence $e_1, \ldots, e_n$ as follows: $e_1 := (p_1, p_2), \ldots, e_{n-1} := (p_{n-1}, p_n), e_n := (p_n, p_1)$. S is called a *polyline* if no two edges e_i and e_j with $2 \leq |i - j| < n - 1$ from this sequence are crossing.

Each polyline uniquely describes a polygon. This is defined next.

Definition 30 Let $L = p_1, \ldots, p_n$ be a polyline with the corresponding edge sequence $e_1, \ldots, e_n$.

- We set $e_{n+1} := e_1$. L is called *convex* iff for all pairs of edges (e_i, e_{i+1}) with $1 \leq i \leq n$, the angle $\alpha(e_1, e_2)$ is strictly smaller than 180 degrees.[16]
- Each of the edges $e_i = (p_1, p_2)$ is thought to be oriented from p_1 to p_2. For each edge e_i, let $h(e_i)$ denote the half plane that is located left of e_i w.r.t. its orientation, such that $h(e_i)$ includes the straight line described by e_i itself. L uniquely describes a subset $p(L)$ of $\mathbb{R}^2$ called *polygon* as follows: $p(L) := \bigcap_{i=1..n} h(e_i)$.

Figure 6 shows an example polygon and illustrates the notions just introduced. Polygons play a central rôle in the extension of the concrete domain $\mathcal{S}_2$.

[16] This definition of convex is more restrictive than the usual definition since we demand the angle to be *strictly* smaller than 180 degrees.

Definition 31 The concrete domain $\mathcal{S}_2\mathcal{P}$ is defined as $\mathcal{S}_2$ augmented by a set of additional predicates. Let $\mathcal{P}$ be the set of all polygons. For each $P \in \mathcal{P}$, and each distinct set of RCC-8 relations $\{R_1, \ldots, R_n\}$, $\Phi_{\mathcal{S}_2\mathcal{P}}$ contains an additional unary predicate $\mathsf{r_1}\text{-}\cdots\text{-}\mathsf{r_n}\text{-}\mathsf{P}$ defined as follows: For all $r \in \Delta_{\mathcal{S}_2\mathcal{P}}$, $r \in \mathsf{r_1}\text{-}\cdots\text{-}\mathsf{r_n}\text{-}\mathsf{P}^{\mathcal{S}_2\mathcal{P}}$ iff $(r, P) \in R_1 \vee \cdots \vee R_n$.

There are strong indications that the concrete domain $\mathcal{S}_2\mathcal{P}$ is admissible. A formal proof and a decision algorithm will be part of future work. As things are now, it is important to note that $\mathcal{S}_2\mathcal{P}$ is a useful and natural extension of $\mathcal{S}_2$.

The following ABox $\mathcal{A}_1$ uses facilities provided by $\mathcal{S}_2\mathcal{P}$. The fillers of has-area are restricted by predicates referring to quantitative values (see Figure 4 for the exact location of the polygons p6 and p7).

$$\mathcal{A}_1 := \left\{ \begin{array}{l} \mathsf{i1} : \mathsf{bird\text{-}sanctuary} \sqcap \exists\, \mathsf{has\text{-}area} \,.\, \mathsf{eq\text{-}p6}, \\ \mathsf{i2} : \mathsf{dredging\text{-}process} \sqcap \exists\, \mathsf{has\text{-}area} \,.\, \mathsf{eq\text{-}p7} \end{array} \right\}$$

The algorithm for testing the satisfiability of conjunctions of concrete domain predicates has to be extended in such a way that all implicit topological relationships between regions given as polygons are computed. However, the relationship between qualitative and quantitative constraints is complex. First results in this area have been published for temporal reasoning [24, 28]. Thus, in a similar way as for $\mathcal{S}_2\mathcal{P}$ we can define a temporal concrete domain with metric constraints for time intervals. In ABox $\mathcal{A}_2$ this extended temporal concrete domain is used to represent spatiotemporal knowledge about the bird sanctuary and the dredging process (see Figure 4 for the relation of the time intervals ti1 and ti2).

$$\mathcal{A}_2 := \left\{ \begin{array}{l} \mathsf{i1} : \mathsf{bird\text{-}sanctuary} \sqcap \exists\, \mathsf{has\text{-}area} \,.\, \mathsf{eq\text{-}p6} \sqcap \exists\, \mathsf{has\text{-}duration} \,.\, \mathsf{equal\text{-}ti1}, \\ \mathsf{i2} : \mathsf{dredging\text{-}process} \sqcap \exists\, \mathsf{has\text{-}area} \,.\, \mathsf{eq\text{-}p7} \sqcap \exists\, \mathsf{has\text{-}duration} \,.\, \mathsf{equal\text{-}ti2} \end{array} \right\}$$

Using the extended concrete domain we can derive a combined spatiotemporal concrete domain (see Section 2.2) with metric (i.e. quantitative) and qualitative predicates which can be used to infer qualitative spatiotemporal constraints between processes. In this case there is no need to use external processes to explicitly add ABox assertions such as $(\mathsf{i1}, \mathsf{i2}) : \mathsf{ntppi\text{-}overlaps}$ as in ABox $\mathcal{A}_0$. However, the details about the admissibility proofs and the decision algorithms have to be explored in future work.

4 Related Work

$\mathcal{ALCRP}(\mathcal{D})$ is a generic description logic which incorporates concrete domains. Two of its major application areas are spatial and temporal reasoning, as was demonstrated in the previous sections of this article. Accordingly, related work from all of these three research areas is presented in this section. We start by examining the relationship of $\mathcal{ALCRP}(\mathcal{D})$ to other description logics, especially those that offer reasoning with concrete domains.

The basic description logic for reasoning with concrete domains as introduced in [5] is the logic $\mathcal{ALC}(\mathcal{D})$. In Section 3.2, the connection of $\mathcal{ALC}(\mathcal{D})$ and $\mathcal{ALCRP}(\mathcal{D})$

has already been briefly analyzed. The most obvious difference is that $\mathcal{ALC}(\mathcal{D})$ does only allow the definition of concepts based on concrete domain predicates whereas $\mathcal{ALCRP}(\mathcal{D})$ is equipped with an additional role-forming concrete domain predicate operator. With its concept-forming operator, $\mathcal{ALC}(\mathcal{D})$ provides a limited means for the existential specification of relations between abstract objects using concrete domain predicates (see [26] for modelling examples exploiting this fact). $\mathcal{ALC}(\mathcal{D})$ is, however, truly less expressive than $\mathcal{ALCRP}(\mathcal{D})$. Especially for the universal quantification over defined roles, which can be expressed in $\mathcal{ALCRP}(\mathcal{D})$, no equivalent or similarly expressive means exists in $\mathcal{ALC}(\mathcal{D})$.

In the literature, several other extensions of $\mathcal{ALC}(\mathcal{D})$ can be found. None of these extensions allows the definition of roles based on predicates. In [23], a generalization of the concept-forming predicate operator of $\mathcal{ALC}(\mathcal{D})$ is proposed and the decidability of the resulting language is proved. A recent extension of $\mathcal{ALC}(\mathcal{D})$ intended for data aggregation in the context of extended entity relationship diagrams extends $\mathcal{ALC}(\mathcal{D})$ by defined features based on aggregation predicates [6]. In order to obtain a decidable formalism, syntactic restrictions are posed on the concept language. Interestingly, similar to the case of $\mathcal{ALCRP}(\mathcal{D})$, the use of the value restriction operator and negation need to be constrained.

Qualitative spatial reasoning is one of the most important applications of the description logic $\mathcal{ALCRP}(\mathcal{D})$. For reasoning about spatial structures many specific approaches have been published, see e.g. [9] or [40] for an overview. Spatial reasoning with $\mathcal{ALCRP}(\mathcal{D})$ as proposed in this article is most closely related to approaches which are based on the RCC theory or use the RCC-8 relations. Research on the RCC theory is summarized in [10]. The RCC theory itself [32] is an axiomatic theory based on a many-sorted first-order logic which provides a high expressive power. Unfortunately, the theory (like many other spatial theories, e.g. [31], as discussed in [9]) was shown to be undecidable.

Recently, the RCC-8 relations have been defined in terms of intuitionistic logic and propositional modal logic [7, 34]. These logics are decidable and can be used for reasoning about RCC-8 networks as introduced in Definition 25. However, the logics are designed to support reasoning about spatial regions, only. Integrated spatial and conceptual reasoning is not supported. In Bennett's formalism, the RCC-8 relations are encoded using a modal operator that can be understood to represent "connectedness" of spatial regions. The RCC-8 relations themselves are not represented by modal operators (in contrast to the defined roles of $\mathcal{ALCRP}(\mathcal{D})$)[17] and, hence, universal and existential quantification over spatial relations is not possible. The approach of using $\mathcal{ALCRP}(\mathcal{D})$ as basis for qualitative spatial reasoning seems to be unique in that the integration of spatial and conceptual reasoning in a decidable formalism is considered.

Finally, we present related work from temporal reasoning. In this article, we have examined the applicability of $\mathcal{ALCRP}(\mathcal{D})$ to integrated spatial and temporal reasoning. In [26], $\mathcal{ALCRP}(\mathcal{D})$ has been considered as a tool for temporal reasoning, only. As opposed to spatial reasoning, which has rarely been considered in the context of description logics, there exist a number of description logics for temporal reasoning (see e.g. [38], [37] and [8]). An overview can be found in [3]. As a representative, we

[17] Many description logics can be seen as a notational variant of modal logics [36]. If this view is taken, roles correspond to modal operators.

discuss the temporal description logic $\mathcal{TL}$-$\mathcal{ALCF}$ as proposed in [2].

The logic $\mathcal{TL}$-$\mathcal{ALCF}$ is closely related to temporal logics. $\mathcal{TL}$-$\mathcal{ALCF}$ has a "temporal semantics", i.e., the structure of time is an integral part of the semantics. With $\mathcal{ALCRP}(\mathcal{D})$ the structure of time has to be modeled in the formalism itself. This has been called an internal representation of time whereas $\mathcal{TL}$-$\mathcal{ALCF}$ provides an external representation of time [14]. Furthermore, there is a fundamental ontological difference between the two logics: The logic $\mathcal{TL}$-$\mathcal{ALCF}$ is designed to represent eternal objects that have properties which are changing over time. In contrast, temporal objects in $\mathcal{ALCRP}(\mathcal{D})$ have static properties but a unique temporal extension. In $\mathcal{ALCRP}(\mathcal{D})$, it is rather straightforward to specify temporal relations between two distinct logical objects. $\mathcal{TL}$-$\mathcal{ALCF}$ allows only very restricted specifications of temporal relations between different objects but instead focuses on temporal relations between different states of the *same* logical object. The two logics $\mathcal{TL}$-$\mathcal{ALCF}$ and $\mathcal{ALCRP}(\mathcal{D})$ cover different aspects of temporal reasoning and are in many aspects orthogonal formalisms. A combination that provides the advantages of both logics would be a very powerful tool.

Just as $\mathcal{ALC}(\mathcal{D})$, $\mathcal{TL}$-$\mathcal{ALCF}$ does allow existential but not universal quantification over Allen's relations. The only formalism we know of that supports both existential and universal quantification is (besides $\mathcal{ALCRP}(\mathcal{D})$) an interval modal logic introduced in [22]. The logic has a strong correspondence to the unrestricted version of $\mathcal{ALCRP}(\mathcal{D})$ instantiated with a temporal concrete domain as proposed in [26]. In the mentioned paper, Halpern and Shoham were able to prove the undecidability of their logic.

5 Conclusion

Based on the description logic language $\mathcal{ALCRP}(\mathcal{D})$, we have shown how spatial and terminological reasoning can be combined in the TBox and ABox. Thus, the fruitful research on description logics has been extended to cope with qualitative spatial relations between spatial objects. One of the main ideas is to introduce constructors for roles whose definitions are based on properties of concrete objects. The abstract domain is used to represent terminological knowledge about spatial (and temporal) domains on an abstract logical level. The concrete domain (space domain or temporal domain) extends the abstract domain and provides access to spatial reasoning services. We have shown that the concrete domain $\mathcal{S}_2$ with $\Delta_{\mathcal{S}_2}$ being all non-empty, regular closed subsets of $\mathbb{R}^2$ and the set of predicates $\Phi_{\mathcal{S}_2}$ representing RCC-8 formulas is admissible. Our approach for testing satisfiability of finite conjunctions relies on current work in qualitative spatial reasoning theory (see e.g. [34]). We have provided several application examples demonstrating the necessity of terminological and spatial reasoning (even in the TBox) in order to avoid unintended models.

In a similar way, a temporal domain can be defined with predicates based on interval relations. A decision procedure for conjunctions of these predicates has been sketched. An application example (bird_sanctuary) demonstrates the use of this domain in combination with the spatial domain to represent spatiotemporal phenomena. In comparison with $\mathcal{ALC}(\mathcal{D})$, the combination of two concrete domains in $\mathcal{ALCRP}(\mathcal{D})$ really extends the expressiveness of the language.

We admit that the $\mathcal{ALCRP}(\mathcal{D})$ restrictedness criterion for terminologies does impose tight constraints on modeling spatiotemporal terminological structures. However, in a specific application, many interesting concepts can be represented in a TBox with the additional advantage of having a decidable satisfiability algorithm. The discussion of related work reveals that other approaches introduce similar (if not even more severe) syntactical restrictions on the concept terms that can be handled.

The $\mathcal{ALCRP}(\mathcal{D})$ approach presented in this article demonstrates how constraint reasoning and description logics complement each other. The examples presented in this article indicate that conceptual knowledge formalized with a description logic and spatial knowledge concerning topological relations provides solutions for important problems, for instance in GIS applications. An important application of description logic theory in this context is, for instance, schema reasoning for data integration in GIS systems. Defined relations provide a bridge from spatial to conceptual knowledge and support more extensive reasoning services to be exploited for solving practical problems.

References

[1] J. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, 1983.

[2] A. Artale and E. Franconi. A computational account for a description logic of time and actions. In J. Doyle, E. Sandewall, and P. Torasso, editors, *Fourth International Conference on Principles of Knowledge Representation, Bonn, Germany, May 24-27, 1994*, pages 1–14, May 1994.

[3] Alessandro Artale and Enrico Franconi. Temporal description logics. In L. Vila, P. van Beek, M. Boddy, M. Fisher, D.M. Gabbay, A. Galton, and R. Morris, editors, *Handbook of Time and Temporal Reasoning in Artificial Intelligence*. MIT Press, To appear.

[4] F. Baader and P. Hanschke. A scheme for integrating concrete domains into concept languages. In *Twelfth International Conference on Artificial Intelligence, Darling Harbour, Sydney, Australia, Aug. 24-30, 1991*, pages 452–457, August 1991.

[5] F. Baader and P. Hanschke. A scheme for integrating concrete domains into concept languages. Technical Report DFKI-RR-91-10, German Center for AI (DFKI), 1991.

[6] F. Baader and U. Sattler. Description logics with aggregates and concrete domains, Part II. Technical Report 98-02, LuFG Theoretical Computer Science, RWTH Aachen, 1998.

[7] B. Bennett. Modal logics for qualitative spatial reasoning. *Bull. of the IGPL*, 3:1–22, 1995.

[8] C. Bettini. A family of temporal terminological logics. In P. Torasso, editor, *Advances in Artificial Intelligence, Third Congress of the Italian Association for*

*Artificial Intelligence, AI*IA '93*, volume 728 of *LNAI*, pages 120–131, Torino, Italy, October 26–28, 1993. Springer-Verlag 1993.

[9] A.G. Cohn. Qualitative spatial representation and reasoning techniques. In G. Brewka, C. Habel, and B. Nebel, editors, *Proceedings, KI-97: Advances in Artificial Intelligence, 21st Annual German Conference on Artificial Intelligence, Freiburg, Germany*, volume 1303 of *Lecture Notes in Artificial Intelligence*, pages 1–30. Springer Verlag, Berlin, September 1997.

[10] A.G. Cohn, B. Bennett, J.M. Gooday, and N.M. Gotts. Representing and reasoning with qualitative spatial relations. In Stock [40], pages 97–134.

[11] T. Cohn, L. Schubert, and S. Shapiro, editors. *Proceedings of Sixth International Conference on Principles of Knowledge Representation and Reasoning (KR'98), Trento, Italy, June 2-5, 1998*, June 1998.

[12] F.M. Donini, M. Lenzerini, D. Nardi, and W. Nutt. The complexity of concept languages. *Information and Computation*, 134(1):1–58, 10 April 1997.

[13] M.J. Egenhofer. 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.

[14] M. Finger and D. M. Gabbay. Adding a temporal dimension to a logic system. *Journal of Logic Language and Information*, 1:203–233, 1992.

[15] A. Gerevini. Reasoning about time and actions in artificial intelligence: Major issues. In Stock [40], pages 43–70.

[16] V. Haarslev. Formal semantics of visual languages using spatial reasoning. In *1995 IEEE Symposium on Visual Languages, Darmstadt, Germany, Sep. 5-9*, pages 156–163. IEEE Computer Society Press, Los Alamitos, September 1995.

[17] V. Haarslev. A fully formalized theory for describing visual notations. In K. Marriott and B. Meyer, editors, *Visual Language Theory*, pages 261–292. Springer Verlag, Berlin, 1998.

[18] V. Haarslev, C. Lutz, and R. Möller. Foundations of spatioterminological reasoning with description logics. In Cohn et al. [11], pages 112–123.

[19] V. Haarslev and R. Möller. SBox: A qualitative spatial reasoner—progress report. In L. Ironi, editor, *11th International Workshop on Qualitative Reasoning, Cortona, Tuscany, Italy, June 3-6, 1997, Pubblicazioni N. 1036, Istituto di Analisi Numerica C.N.R. Pavia (Italy)*, pages 105–113, June 1997.

[20] V. Haarslev, R. Möller, and C. Schröder. Combining spatial and terminological reasoning. In B. Nebel and L. Dreschler-Fischer, editors, *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, September 1994.

[21] V. Haarslev and M. Wessel. GenEd—an editor with generic semantics for formal reasoning about visual notations. In *1996 IEEE Symposium on Visual Languages, Boulder, Colorado, USA, Sep. 3-6*, pages 204–211. IEEE Computer Society Press, Los Alamitos, September 1996.

[22] J.Y. Halpern and Y. Shoham. A propositional model logic of time intervals. *Journal of the Association of Computing Machinery*, 38(4):935–962, 1991.

[23] P. Hanschke. Specifying role interaction in concept languages. In B. Nebel, C. Rich, and W. Swartout, editors, *Third International Conference on Principles of Knowledge Representation and Reasoning (KR'92)*. Morgan Kaufmann, 1992.

[24] H. A. Kautz and P. B. Ladkin. Integrating metric and qualitative temporal reasoning. In *Proc. of AAAI-91*, pages 241–246, Anaheim, CA, 1991.

[25] C. Lutz. Representation of Topological Information in Description Logics (in German). Master's thesis, University of Hamburg, Computer Science Department, February 1998.

[26] C. Lutz, V. Haarslev, and R. Möller. A concept language with role-forming predicate restrictions. Technical Report FBI-HH-M-276/97, University of Hamburg, Computer Science Department, 1997.

[27] C. Lutz and R. Möller. Defined topological relations in description logics. In M.-C. Rousset et al., editor, *Proceedings of the International Workshop on Description Logics, DL'97, Sep. 27-29, 1997, Gif sur Yvette, France*, pages 15–19. Universite Paris-Sud, Paris, September 1997.

[28] I. Meiri. Combining qualitative and quantitative constraints in temporal reasoning. *Artificial Intelligence*, 87 (1-2):343–385, 1996.

[29] B. Nebel. Terminological reasoning is inherently intractable. *Artificial Intelligence*, 43:235–249, 1990.

[30] B. Nebel. Terminological cycles: Semantics and computational properties. In J.F. Sowa, editor, *Principles of Semantic Networks: Explorations in the Representation of Knowledge*, pages 331–361. Morgan Kaufmann Publishers, San Mateo, 1991.

[31] I. Pratt and D. Schoop. A complete axiom system for polygonal mereotopology of the real plane. Technical Report UMCS-97-2-2, University of Manchester, Department of Computer Science, 1997.

[32] D.A. Randell, Z. Cui, and A.G. Cohn. A spatial logic based on regions and connections. In B. Nebel, C. Rich, and W. Swartout, editors, *Principles of Knowledge Representation and Reasoning, Cambridge, Mass., Oct. 25-29, 1992*, pages 165–176, October 1992.

[33] J. Renz. A canonical model of the region connection calculus. In Cohn et al. [11], pages 330–341.

[34] J. Renz and B. Nebel. On the complexity of qualitative spatial reasoning: A maximal tractable fragment of the region connection calculus. In *Proceedings of the 15th International Joint Conference on Artificial Intelligence (IJCAI'97)*, pages 522–527, August 1997.

[35] J. Renz and B. Nebel. Efficient methods for qualitative spatial reasoning. In H. Prade, editor, *Proceedings of the 13th European Conference on Artificial Intelligence (ECAI'98), Aug. 23-28, Brighton, UK*, pages 562–571, August 1998.

[36] K. Schild. A correspondence theory for terminological logics: Preliminary report. In *Twelfth International Conference on Artificial Intelligence, Darling Harbour, Sydney, Australia, Aug. 24-30, 1991*, pages 466–471, August 1991.

[37] K. Schild. Combining terminological logics with tense logic. In M. Filgueiras and L. Damas, editors, *Progress in Artificial Intelligence, 6th Portuguese Conference on AI, EPIA '93*, volume 727 of *LNAI*, pages 105–120, Porto, Portugal, October 6–8, 1993. Springer-Verlag 1993.

[38] A. Schmiedel. A temporal terminological logic. In *Proceedings of the Tenth National Conference on Artificial Intelligence AAAI-90*, volume 2, pages 640–645, Boston, Mass., July 29 – August 3, 1990. AAAI Press 1990.

[39] E. Spanier. *Algebraic Topology.* McGraw-Hill Book Company, New York, 1966.

[40] O. Stock, editor. *Spatial and Temporal Reasoning.* Kluwer Academic Publishers, Dordrecht, 1997.