To appear in: Proceedings, 1999 IEEE Symposium on Visual Languages, Tokyo, Japan, Sep. 13-16, IEEE Computer Society Press, 1999.

On Specifying Semantics of Visual Spatial Query Languages

Volker Haarslev and Ralf Möller and Michael Wessel
University of Hamburg, Computer Science Department,
Vogt-Kölln-Str. 30, 22527 Hamburg, Germany
http://kogs-www.informatik.uni-hamburg.de/{~haarslev/|~moeller/|~mwessel/}

Abstract

We present a first treatment dealing with semantics of visual *spatial* query languages for GIS using a suitable description logic. This decidable space logic is described and its usefulness for GIS exemplified. The logic supports the specification of semantics, reasoning about query subsumption and about applying default knowledge.

1 Introduction

For accessing spatial databases or geographic information systems (GIS), different query specification techniques have been proposed. For instance, the visual spatial query system VISCO developed in our group [1, 2] can be used to query a spatial database (GIS) in a *visual* way. In contrast to conventional textual query systems the user is not required to learn a complicated textual query language in order to effectively use an information system. Users can query the database by drawing diagrammatic representations of what is to be retrieved from the spatial information system. However, experiences with the current VISCO system indicate that in the context of VISCO (and query systems in general), the specification of queries in a GIS still could be made easier by advances in research areas combining spatial and terminological reasoning with visual language theory.

In this paper we discuss the application of a new logic-based formalism to specifying semantics of visual spatial queries. To the best of our knowledge this is the first proposal utilizing an expressive and decidable spatial logic for this task. The formalism can be used to define the semantics of visual spatial queries, to reason about query subsumption, and to deal with multiple worlds or query completion with the help of default reasoning. Examples for these kinds of reasoning are discussed in this paper. Our formalism is based on the description logic $\mathcal{ALCRP}(\mathcal{D})$ [3, 4] offering mechanisms for integrating so-called *concrete domains* and on a recent extension for default reasoning [5].

We like to emphasize that the work on VL theory presented in this paper truly extends our previous research as summarized in [6], where we used a logic that is more expressive than $\mathcal{ALCRP}(\mathcal{D})$ since it allows *qualified number restrictions* but also less expressive than $\mathcal{ALCRP}(\mathcal{D})$ since it has no *defined roles*. In [7] we made the first proposal for using $\mathcal{ALCRP}(\mathcal{D})$. This paper extends this proposal by considering semantics of visual spatial query languages and by integrating default reasoning. The use of $\mathcal{ALCRP}(\mathcal{D})$ as a tool for VL theory, especially for formally describing semantics of visual spatial query languages, is rather innovative and unprecedented.

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

This section gives a brief introduction to the description logic $\mathcal{ALCRP}(\mathcal{D})$ and to description logic (DL) theory in general summarizing the notions important for this paper. We refer to [8, 9] for more complete information about description logic theory.

Many DL theories can be viewed as subsets of first-order predicate logic. However it is important to note that particular DL theories are only considered as practical if they are based on *sound and complete* reasoning algorithms, i.e. the *decidability* of a DL is of utmost importance. Of course, this is a major distinction to reasoning with general first-order predicate logic.

DL theories are based on the ideas of structured inheritance networks [10]. A DL has similarities to a term rewriting language usually (but not necessarily) restricting the left-hand side of equations to single unique term names. In a DL a factual world consists of named individuals and their relationships that are asserted through binary relations. Hierarchical descriptions about sets of individuals form the terminological knowledge. Descriptions (or terms) about sets of individuals are called *concepts* and binary relations are called *roles*. Descriptions consist of identifiers denoting concepts, roles, and individuals, and of description constructors. For any individual x the set $\{\mathsf{y}|\mathsf{r}(\mathsf{x},\mathsf{y})\}$ is called the set of *fillers* of role r. A role which may have at most one filler is referred to as *feature*.

For instance, consider the following description used in our GIS scenario with the intended meaning "a cottage that is enclosed by a forest" that contains concept names (e.g. cottage), role names (e.g. is_g_inside), and constructors (e.g. $\sqcap$ and $\exists$).

$$\textbf{cottage\_in\_forest} \doteq \mathsf{cottage} \sqcap \exists\, \mathsf{is\_g\_inside} \,.\, \mathsf{forest}$$

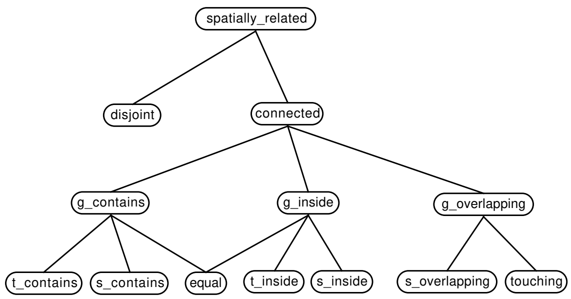

Figure 1: Subsumption hierarchy of spatial predicates.

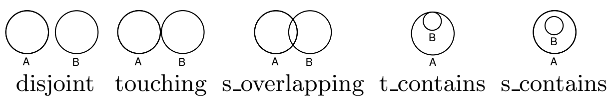

Figure 2: Spatial relations between A and B. The inverses of t_contains and s_contains as well as the relation equal have been omitted.

The expressiveness and computational complexity of a particular DL depends on the variety of employed description constructors. Various complexity results for subsumption algorithms for specific description logics are summarized in [8]. Recent findings for $\mathcal{ALCRP}(\mathcal{D})$ suggest that deciding satisfiability is at least in EXPTIME.

2.1 Terminologies

In this section, the language (syntax and semantics) for defining concepts and roles in $\mathcal{ALCRP}(\mathcal{D})$ is presented. $\mathcal{ALCRP}(\mathcal{D})$ is parameterized with a concrete domain which consists of a set of concrete objects and a set of predicates.

Concrete Domains: 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 $\mathsf{P}^\mathcal{D} \subseteq \Delta_\mathcal{D}^n$. A concrete domain $\mathcal{D}$ is called *admissible* iff (1) the set of its predicate names is closed under negation (i.e. for any $\mathsf{P} \in \Phi_\mathcal{D}$ there exists a $\overline{\mathsf{P}} \in \Phi_\mathcal{D}$ denoting the negation of P) and contains a name $\top_\mathcal{D}$ for $\Delta_\mathcal{D}$ and (2) the satisfiability problem for finite conjunctions of predicates is decidable.

A concrete domain can be understood as a device providing a bridge between conceptual reasoning with abstract entities and (qualitative) constraint reasoning with concrete or symbolic data. In this paper we use the admissible concrete domain $\mathcal{RS}_2$. It is the union of the domains $\mathcal{R}$ (over the set $\mathbb{R}$ of all real numbers with predicates built by first order means from (in)equalities between integer polynomials in several indeterminates, see [11]) and $\mathcal{S}_2$ (over the set of all two-dimensional polygons with topological relations from Figure 1 and 2 as predicates, see [4]). The name 'concrete domain' is in some sense misleading since it suggests that a concrete domain realizes reasoning about 'concrete' (e.g. numeric) data. This kind of reasoning is sometimes supported (e.g. in the domain $\mathcal{R}$) but in our application we mainly use concrete domains for reasoning about the satisfiability of finite conjunctions of predicates. For instance, the domain $\mathcal{S}_2$ qualitatively decides the satisfiability of conjunctions such as $\mathsf{touching}(I_1, I_2) \wedge \mathsf{contains}(I_2, I_3) \wedge \mathsf{touching}(I_1, I_3)$ without any notion for 'concrete' polygons. This is a well-known example for a constraint satisfaction problem.

Without loss of generality we introduce a λ-like notation for anonymous predicates for the domain $\mathcal{R}$. Formally, each anonymous predicate and its negation could be replaced by unique names for the λ-term and its negated counterpart and, moreover, the the negation sign in front of a λ-term can be safely moved inside of this term.

Role Terms: Let R and F be disjoint sets of role and feature names, respectively. Any element of R and any element of F is an *atomic role term*. The elements of F are also called *features*. A composition of features (written $\mathsf{f}_1\mathsf{f}_2\cdots$) is called a feature chain. A feature chain of length one is also a feature chain. If $\mathsf{P} \in \Phi_\mathcal{D}$ is a predicate name with arity $n+m$ and u_1, … ,u_n as well as v_1, … ,v_m are $n+m$ feature chains, then the expression $\exists\,(\mathsf{u}_1,\ldots,\mathsf{u}_n)(\mathsf{v}_1,\ldots,\mathsf{v}_m)\,.\,\mathsf{P}$ (*role-forming predicate restriction*) is a *complex role term*. Let S be a role name and let T be a role term. Then $\mathsf{S} \doteq \mathsf{T}$ is a terminological axiom.

An example for using a role-forming predicate operator is the definition of a role is_g_inside for a corresponding topological predicate g_inside (see Section 2.1 for an explanation of the semantics). Intuitively spoken, this role holds for any pair of individuals (I_1, I_2) iff the associated spatial area (via the feature has_area) of I_1 is generally inside of the area of I_2.

is_g_inside $\doteq\ \exists\,(\mathsf{has\_area})(\mathsf{has\_area})\,.\,\mathsf{g\_inside}$

Concept Terms: Let C be a set of concept names which is disjoint from R and F. Any element of C is an *atomic concept term*. If C and D are concept terms, R is an arbitrary role term or a feature, $\mathsf{P} \in \Phi_\mathcal{D}$ is a predicate name with arity n, and $\mathsf{u}_1,\ldots,\mathsf{u}_n$ are feature chains, then the following expressions are also concept terms: $\mathsf{C} \sqcap \mathsf{D}$ (*conjunction*), $\mathsf{C} \sqcup \mathsf{D}$ (*disjunction*), $\neg\mathsf{C}$ (*negation*), $\exists\,\mathsf{R}\,.\,\mathsf{C}$ (*concept exists restriction*), $\forall\,\mathsf{R}\,.\,\mathsf{C}$ (*concept value restriction*), and $\exists\,\mathsf{u_1},\ldots,\mathsf{u_n}\,.\,P$ (*predicate exists restriction*).

We illustrate the notion of concept and role terms by extending the cottage example mentioned above.

normal_cottage_in_forest $\doteq$

cottage_in_forest $\sqcap$

$\exists\,\mathsf{has\_space}\,.\,\lambda_{\mathcal{R}} x\,.\,(x \geq 30 \wedge x < 70)$

The definition of normal_cottage_in_forest roughly has the intended meaning "something is a standard cottage in a forest if and only if it is a cottage located in a forest with 30-70 square meters of total space for the cottage." This definition also gives an example for a predicate exists restriction for the domain $\mathcal{R}$ using a feature has_space.

In order to ensure the decidability, we had to restrict possible combinations of concepts terms w.r.t. defined roles (e.g. a nested concept term with defined roles such as $\forall\,\mathsf{is\_touching}\,.\,\exists\,\mathsf{is\_g\_inside}\,.\,\mathsf{cottage}$ is not allowed). Note that all examples in this paper are restricted. However, this restrictedness criterion is beyond the scope of this paper and is fully explained elsewhere [4].

Terminology: For any exists and value restrictions, the role term or list of feature chains may be written in parentheses. Let A be a concept name and D be a concept term. Then $\mathsf{A} \doteq \mathsf{D}$ and $\mathsf{A} \sqsubseteq \mathsf{D}$ are terminological axioms as well. A finite set of terminological axioms $\mathcal{T}$ is called a *terminology* or *TBox* if no concept or role name in $\mathcal{T}$ appears more than once on the left-hand side of a definition and, furthermore, if no cyclic definitions are present.

The previous examples already informally introduced concept axioms for defined concepts using the $\doteq$ operator. For convenience, we also allow the $\sqsubseteq$ operator for the definition of primitive concepts, i.e. their definition consists only of necessary conditions. The concept cottage is a good candidate for a primitive definition documenting that we omitted in our terminology other conditions that are not relevant for this modeling task. For instance, a cottage has to be at least a building.

cottage $\sqsubseteq$ building

Of course, there exist other description logics that allow more than one axiom for a particular concept name or even support generalized concept inclusions (implications) with arbitrary concept terms on the left and right side of terminological axioms. These axioms can be used as a powerful modeling tool but are currently not supported in $\mathcal{ALCRP}(\mathcal{D})$ w.r.t. decidability.

Semantics: 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 $\Delta_{\mathcal{D}}$ (see above) and $\Delta_{\mathcal{I}}$ must be disjoint. The interpretation function maps each concept name C to a subset $\mathsf{C}^{\mathcal{I}}$ of $\Delta_{\mathcal{I}}$, each role name R to a subset $\mathsf{R}^{\mathcal{I}}$ of $\Delta_{\mathcal{I}} \times \Delta_{\mathcal{I}}$, and each feature name f to a partial function $\mathsf{f}^{\mathcal{I}}$ from $\Delta_{\mathcal{I}}$ to $\Delta_{\mathcal{D}} \cup \Delta_{\mathcal{I}}$, where $\mathsf{f}^{\mathcal{I}}(a) = x$ will be written as $(a, x) \in f^{\mathcal{I}}$. If $u = \mathsf{f}_1 \cdots \mathsf{f}_n$ is a feature chain, then $u^{\mathcal{I}}$ denotes the composition $\mathsf{f}_1^{\mathcal{I}} \circ \cdots \circ \mathsf{f}_n^{\mathcal{I}}$ of the partial functions $\mathsf{f}_1^{\mathcal{I}}, \ldots, \mathsf{f}_n^{\mathcal{I}}$. Let the symbols C, D, R, P, u_1, ... ,u_m, and v_1, ... ,v_m be defined as above. Then the interpretation function can be extended to arbitrary concept and role terms as follows:

$$
\begin{aligned}
(\mathsf{C} \sqcap \mathsf{D})^{\mathcal{I}} &:= \mathsf{C}^{\mathcal{I}} \cap \mathsf{D}^{\mathcal{I}}\\
(\mathsf{C} \sqcup \mathsf{D})^{\mathcal{I}} &:= \mathsf{C}^{\mathcal{I}} \cup \mathsf{D}^{\mathcal{I}}\\
(\neg\mathsf{C})^{\mathcal{I}} &:= \Delta_{\mathcal{I}} \setminus \mathsf{C}^{\mathcal{I}}\\
(\exists\,\mathsf{R}\,.\,\mathsf{C})^{\mathcal{I}} &:= \{a \in \Delta_{\mathcal{I}} \mid \exists\, b \in \Delta_{\mathcal{I}} : (a, b) \in \mathsf{R}^{\mathcal{I}}, b \in \mathsf{C}^{\mathcal{I}}\}\\
(\forall\,\mathsf{R}\,.\,\mathsf{C})^{\mathcal{I}} &:= \{a \in \Delta_{\mathcal{I}} \mid \forall\, b \in \Delta_{\mathcal{I}} : (a, b) \in \mathsf{R}^{\mathcal{I}} \Rightarrow b \in \mathsf{C}^{\mathcal{I}}\}
\end{aligned}
$$

$$
\begin{aligned}
&(\exists\,\mathsf{u_1},\ldots,\mathsf{u_n}\,.\,\mathsf{P})^{\mathcal{I}} :=\\
&\qquad \{a \in \Delta_{\mathcal{I}} \mid \exists\, \mathsf{x}_1,\ldots,\mathsf{x}_n \in \Delta_{\mathcal{D}} :\\
&\qquad\quad (a, \mathsf{x}_1) \in \mathsf{u}_1^{\mathcal{I}},\ldots,(a, \mathsf{x}_n) \in \mathsf{u}_n^{\mathcal{I}}, (\mathsf{x}_1,\ldots,\mathsf{x}_n) \in \mathsf{P}^{\mathcal{D}}\}\\
&(\exists\,(\mathsf{u_1},\ldots,\mathsf{u_n})(\mathsf{v_1},\ldots,\mathsf{v_m})\,.\,\mathsf{P})^{\mathcal{I}} :=\\
&\qquad \{(a, b) \in \Delta_{\mathcal{I}} \times \Delta_{\mathcal{I}} \mid \exists\, \mathsf{x}_1,\ldots,\mathsf{x}_n, \mathsf{y}_1,\ldots,\mathsf{y}_m \in \Delta_{\mathcal{D}} :\\
&\qquad\quad (a, \mathsf{x}_1) \in \mathsf{u}_1^{\mathcal{I}},\ldots,(a, \mathsf{x}_n) \in \mathsf{u}_n^{\mathcal{I}},\\
&\qquad\quad (b, \mathsf{y}_1) \in \mathsf{v}_1^{\mathcal{I}},\ldots,(b, \mathsf{y}_m) \in \mathsf{v}_m^{\mathcal{I}},\\
&\qquad\quad (\mathsf{x}_1,\ldots,\mathsf{x}_n, \mathsf{y}_1,\ldots,\mathsf{y}_m) \in \mathsf{P}^{\mathcal{D}}\}
\end{aligned}
$$

An interpretation $\mathcal{I}$ is a *model* of a TBox $\mathcal{T}$ iff it satisfies $\mathsf{A}^{\mathcal{I}} = \mathsf{C}^{\mathcal{I}}$ for all terminological axioms $\mathsf{A} \doteq \mathsf{C}$ in $\mathcal{T}$, and $\mathsf{A}^{\mathcal{I}} \subseteq \mathsf{C}^{\mathcal{I}}$ for $\mathsf{A} \sqsubseteq \mathsf{C}$ respectively.

Figure 3 illustrates for the domain $\mathcal{S}_2$ the idea behind the semantics of the role-forming predicate operator. The spatial predicates (e.g. g_inside) operate on concrete domain values (e.g. polygon descriptions) that are attached via features to corresponding abstract individuals. If a role (e.g. is_g_inside) is *defined* by a predicate (e.g. g_inside), then every pair $(\mathsf{p}_1, \mathsf{p}_2)$ of polygons that are fillers of has_area for two abstract individuals i_1 and i_2 is tested whether the binary predicate $\mathsf{g\_inside}(\mathsf{p}_2, \mathsf{p}_1)$ is fulfilled. In case of a successful test the role membership (e.g. is_g_inside) is also established for the abstract individuals i_1 and i_2, i.e. it holds $\mathsf{is\_g\_inside}(\mathsf{i}_2, \mathsf{i}_1)$. This also applies for the opposite direction. If a role membership is asserted for a pair of abstract individuals, their associated concrete feature fillers are either established with the correspond-

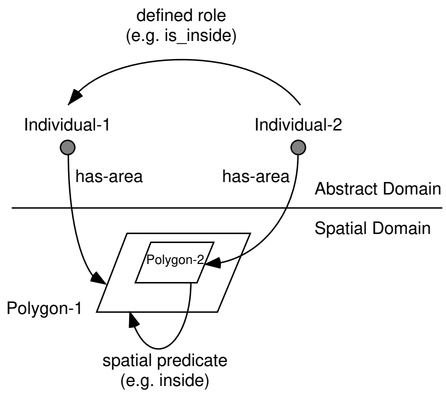

Figure 3: Relationship between abstract and spatial domain.

ing predicate or verified if concrete feature fillers already exist.

2.2 The Assertional Language

The *assertional language* of a DL is designed for stating concept or role memberships of named individuals that are used to describe the factual world. With respect to concrete domains we distinguish *abstract* and *concrete* individuals. Abstract individuals are elements of the abstract domain and have to be members of concepts. Concrete individuals are elements of the concrete domain and are used as parameters for predicates. Both types of individuals can be feature fillers while only abstract individuals can be role fillers. For instance, in the VL domain abstract individuals could represent geometric figures such as circles, rectangles, etc. that represent syntactic elements of a query language and the semantic entities (e.g. lake, estate, etc.) that correspond to syntactic elements. The descriptions might be associated via features with reals defining geometric properties such as the diameter of a circle (lake), the width and height of a rectangle (estate), etc.

The set of assertions (ABox) has to comply to the definitions declared in the TBox. An *ABox* of $\mathcal{ALCRP}(\mathcal{D})$ is a finite set of assertions defined as follows.

Syntax: Let I_A and I_D be two disjoint sets of individual names for the abstract and concrete domain. If C is a concept term, R an atomic or complex role term, f a feature name, P a predicate name with arity n, a and b are elements of I_A and x, x_1, ... ,x_n are elements of I_D, then the following expressions are *assertional axioms*: $\mathsf{a} : \mathsf{C}$ (*concept membership*), $(\mathsf{a},\mathsf{b}) : \mathsf{R}$ (*role filler*), $(\mathsf{a},\mathsf{x}) : \mathsf{f}$ (*feature filler*), $(\mathsf{x}_1,\ldots,\mathsf{x}_\mathsf{n}) : \mathsf{P}$ (*predicate membership*).

Semantics: For specifying the semantics of ABox assertions we have to extend the interpretation function $\mathcal{I}$. An *interpretation* for the assertional language is an interpretation for the concept language which additionally maps every individual name from I_A to a single element of $\Delta_\mathcal{I}$ and every individual name from I_D to a single element from $\Delta_\mathcal{D}$. We assume that the unique name assumption does not hold, that is $\mathsf{a}^\mathcal{I} = \mathsf{b}^\mathcal{I}$ may hold even if $\mathsf{a} \neq \mathsf{b}$.

$$\begin{aligned} \mathsf{a} : \mathsf{C} &\text{ iff } \mathsf{a}^\mathcal{I} \in \mathsf{C}^\mathcal{I} \\ (\mathsf{a},\mathsf{b}) : \mathsf{R} &\text{ iff } (\mathsf{a}^\mathcal{I},\mathsf{b}^\mathcal{I}) \in \mathsf{R}^\mathcal{I} \\ (\mathsf{a},\mathsf{x}) : \mathsf{f} &\text{ iff } \mathsf{f}^\mathcal{I}(\mathsf{a}^\mathcal{I}) = \mathsf{x}^\mathcal{I} \\ (\mathsf{x}_1,\ldots,\mathsf{x}_\mathsf{n}) : \mathsf{P} &\text{ iff } \mathsf{x}_1^\mathcal{I},\ldots,\mathsf{x}_n^\mathcal{I} \in \mathsf{P}^\mathcal{D} \end{aligned}$$

ABox Example: The following assertions illustrate the four different types of ABox assertions using the cottage scenario. Based on the semantics given above a $\mathcal{ALCRP}(\mathcal{D})$ reasoner will infer that c is a member of $\mathsf{normal\_cottage\_in\_forest}$. $\mathsf{S_c}$ and $\mathsf{S_f}$ denote the associated area polygon of c and f.

$\mathsf{c} : \mathsf{cottage}$, $(\mathsf{c}, 60) : \mathsf{has\_space}$, $(\mathsf{c}, \mathsf{S_c}) : \mathsf{has\_area}$

$\mathsf{f} : \mathsf{forest}$, $(\mathsf{f}, \mathsf{S_f}) : \mathsf{has\_area}$, $(\mathsf{S_c}, \mathsf{S_f}) : \mathsf{g\_inside}$

2.3 Reasoning Services

The notion of a *model* (see above) is used to define the reasoning services that a DL inference engine has to provide, i.e. it proves for every concept specification whether the following conditions hold:

- a term A *subsumes* another term B if and only if for every model $\mathcal{I}$, $\mathsf{B}^\mathcal{I} \subseteq \mathsf{A}^\mathcal{I}$;
- a term A is *coherent/satisfiable* if and only if there exists at least one model $\mathcal{I}$ such that $\mathsf{A}^\mathcal{I} \neq \emptyset$;
- terms A and B are *disjoint* if and only if for every model $\mathcal{I}$, $\mathsf{A}^\mathcal{I} \cap \mathsf{B}^\mathcal{I} = \emptyset$;
- terms A and B are *equivalent* if and only if for every model $\mathcal{I}$, $\mathsf{A}^\mathcal{I} = \mathsf{B}^\mathcal{I}$.
- An *ABox* $\mathcal{A}$ is *consistent* if and only if there exists a model $\mathcal{I}$ of $\mathcal{A}$.

The incoherence of a concept is illustrated by the following situation. We define a paradise cottage as a fishing cottage located in a mosquito-free forest, i.e. the forest is not spatially connected with a river.

$\mathbf{fishing\_cottage} \doteq \mathsf{cottage} \sqcap \exists\, \mathsf{is\_touching} \,.\, \mathsf{river}$

$\mathbf{mosquito\_free\_forest} \doteq \mathsf{forest} \sqcap \forall\, \mathsf{is\_connected} \,.\, \neg \mathsf{river}$

$\mathbf{paradise\_cottage} \doteq$

$\mathsf{fishing\_cottage} \sqcap \exists\, \mathsf{is\_g\_inside} \,.\, \mathsf{forest} \sqcap$

$\forall\, \mathsf{is\_g\_inside} \,.\, \mathsf{mosquito\_free\_forest}$

However, a fishing cottage is defined as a cottage that touches a river. It follows that the forest containing a fishing cottage must also be spatially connected with this river. Obviously, the paradise cottage is only a dream that can not exist in the real world. This is due to the intended semantics of the underlying spatial relations: A situation where a region r_1 (cottage) is *g_inside* another region r_2 (forest) and this region r_1 is also *touching* a third region r_3 (river) implies that r_2 is *connected* to r_3, i.e. $\mathsf{g\_inside}(\mathsf{r_1}, \mathsf{r_2}) \land \mathsf{touching}(\mathsf{r_1}, \mathsf{r_3}) \Rightarrow \mathsf{connected}(\mathsf{r_2}, \mathsf{r_3})$

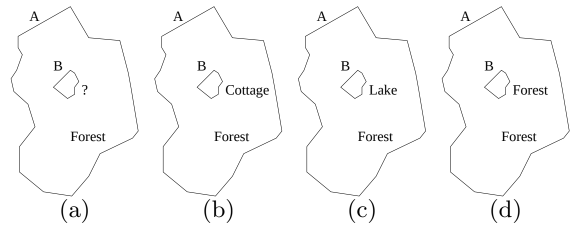

Figure 4: Automatic completion of visual queries by application of default rules (see text).

3 Semantics of Spatial Queries

The previous sections defined the description logic $\mathcal{ALCRP}(\mathcal{D})$ and demonstrated its usefulness for spatial reasoning. We introduced semantic entities such as buildings, cottages, forests, etc. These entities are suitable candidates for elements of visual spatial queries. In VISCO we assume that these and other basic map objects are predefined in a GIS. Furthermore, spatial areas are defined by polygons. Map elements (e.g. polylines, polygons) are annotated with labels such as "forest", "building", "river" etc. that directly correspond to the semantic entities characterized above.

VISCO's queries are basically considered as spatial constellations based on topological and geometric relationships. The syntax and static semantics of VISCO's visual query language can be easily specified with our description logic framework described in [6]. Therefore, we omit any discussions about specifying the syntax of visual spatial query languages and refer the reader to [6]. However, an important contribution of this paper is the attempt to specify the semantics of visual spatial query languages with the help of $\mathcal{ALCRP}(\mathcal{D})$.

We imagine a VISCO application scenario for querying a GIS as follows. Instead of textually writing a complicated SQL query, a user simply draws (sketches) a constellation of spatial entities that resemble the intended constellation of interest. The user also has to assign the intended semantics to drawing elements (e.g. this polygon represents a forest, etc) using the basic vocabulary provided by the GIS. The parser of VISCO analyzes the drawing and creates as semantic representation a corresponding ABox. In case of semantic ambiguities or underspecified information VISCO's reasoner applies default rules for further specialization. The next subsections describe the usefulness of spatial default reasoning and the query processing and reasoning process.

3.1 Completion of Queries

Default knowledge is used to make queries more precise if it can be applied in a consistent way. Due to space limitations we omit any discussions about the formal representation of default knowledge and its rules of inference. This is discussed elsewhere [5]. First of all, the process of formulating (visual) queries can be facilitated by *automatically completing* queries in a meaningful way, therefore reducing the number of mouse interactions. The process of selecting semantic concept descriptors for objects involved in a query (e.g. cottage, river, forest) can partly be automated by interpreting a partially specified query. For instance, in its current development stage, VISCO users can select concept descriptors from a list of over 300 predefined concepts. Thus, even a situation-adapted reduction of the complete list of possibilities to a suitable subset or an order relation for sorting groups of possible concept candidates would be very appropriate.

In order to analyze the modeling problems in this context, we begin with a more detailed discussion of a visual query example. Let us assume a person is interested in buying a cottage located in a forest. In Figure 4(a) the user just started to formulate the query. After (s)he has specified that the type of the surrounding polygon A should be a forest, the type of the small polygon B must be specified. A smart interface should use formal derivation processes for computing plausible candidates for object "type" specifications. For narrowing the set of possibilities we assume that two default rules are applicable: one is saying that the interior small polygon B could be a cottage (Figure 4(b)) and another is stating that B could be a lake (Figure 4(c)) if this does not lead to inconsistencies. Since an object can be either a lake or a cottage, there is no way to believe in both possibilities at a time. This kind of default rule interaction is a simple example demonstrating the necessity of considering different *possible worlds* which must be maintained by the reasoning system. Depending on the default rule being used to conclude new knowledge, different subsequent conclusions might be possible.

Other potentially active default rules might produce inconsistencies with the set of current assertions without providing a possibility of using multiple worlds to avoid inconsistencies. For example, if there had been a default applied indicating that the small polygon B might as well be a forest (Figure 4(d)), we would get a contradiction if we had an axiom (as part of our conceptual background knowledge) requiring that a forest can never contain another forest. Thus, in our query context, the latter default cannot be applied and, as a consequence of computing and appropriately interpreting the set of possible worlds, we can compose a situation-adapted menu for the graphical user interface and the user can select between meaningful concepts for object B. In our specific example, the menu will contain items for cottage and lake but not for forest.

If more than one possible world is computed, an intuitive criterion would be to select the world originating from a default with the more specific precondition or conclusion. E.g., in the query shown in Figure 5(a) we would prefer a default concluding that the thin graphical object might be a 'river flowing into a lake' (which might be a useful concept in our scenario) instead of a more general default concluding only that the object is an ordinary river.

The automatic augmentation of visual queries by conclusions of applied default rules can be seen as a specialization process. Therefore, this process might not only be useful during the construction of a visual query, but also useful as a tool for query refinement after a query has been executed that yields too many results. In addition, not only conceptual information is important. In our GIS context we also have to consider the spatial relations between domain objects

In the context of sketch-based visual querying, on the one hand it is sometimes useful to leave some spatial relations between graphical objects unspecified because they are unknown or simply because the user is not willing to specify them. On the other hand, in order to actually draw a picture, the user *must* specify each spatial relation, even if it is just one of several possible (base) relations. The problem of how to specify "don't care relations" or "example relations" is well known and inherent in diagrammatic representations. It is similar to the problem of visualizing visual disjunctions.

For example, in the query shown in Figure 5(b), we have a visible disjoint relation between the river and the lake. If we intended the river to be disjoint from the lake, the query answering system would not find any rivers flowing into this lake. The problem is how can we specify that the river should be strictly inside the forest but leave the relation to the lake unspecified. As a possible solution to this problem, we could simply *ignore* each visible disjoint relation. But, with this interpretation, we can now no longer state a query searching for rivers *not* flowing into this specific lake, which might be a very useful concept. We propose the following solution. For objects like the river that are drawn with a specific drawing attribute such as dashing, the universal spatial relation to other objects (disjunction of all base relations) is asserted. Dashed objects introduce no spatial query constraints. However, in some cases this would usually not match the users intention as there will be too many matches, i.e. the answer set will be too large. With the help of default knowledge we can automatically refine the query in a way that is appropriate according to the semantics of the objects involved in a query. So, we can guide the interpretation of spatial aspects by the help of conceptual background knowledge and application of defaults, yielding different hypotheses as possible worlds. A river flows into a lake or not, i.e. graphically both objects are either touching, see also Figure 5(c)) or or they are disjoint (see Figure 5(d)). With respect to a lake, there are no other possibilities. In our world model a river never overlaps with a lake (see also Figure 5(e)). This is assumed to be stated as an axiom as part of our general conceptual background knowledge. Besides defaults involving concept constraints we also have to take care of default rules with conclusions yielding new *relation* constraints (see [5] for a discussion).

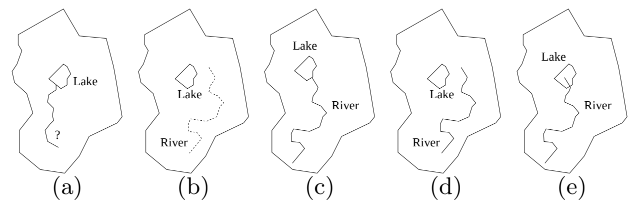

Figure 5: Scenarios for situation-adapted completion of queries (see text).

3.2 Reasoning about Visual Spatial Queries

We flesh out the scenario for the GIS query introduced above. The cottage should be located in a forest with a river in the immediate vicinity. The buyer also want a cottage that provides about 60 m^2 floor space. The estate itself should have about 400 m^2. Having these requirements in mind we sketch a query (see Figure 6) reflecting the intended spatial and geometric constraints.[1] The parser translates the sketch to an equivalent ABox on the basis of a taxonomy containing concept descriptions for the spatial vocabulary of this GIS

[1] We are aware of the scaling problems with drawings and offer with VISCO's query language first solutions. However, in this paper we deliberately ignore these aspects.

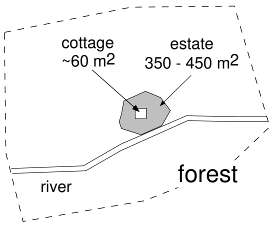

Figure 6: Spatial sketch for first query

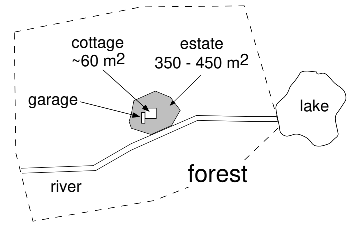

Figure 7: Spatial sketch for second query

domain. We get the following Abox $\mathcal{A}_0$.

$\mathsf{c} : \mathsf{cottage} \sqcap \exists\, \mathsf{has\_space} \,.\, \lambda_{\mathcal{R}} x \,.\, (x > 40 \wedge x < 70)$

$\mathsf{e} : \mathsf{estate\_area} \sqcap \exists\, \mathsf{has\_space} \,.\, \lambda_{\mathcal{R}} x \,.\, (x > 350 \wedge x < 450)$

$\mathsf{r} : \mathsf{river},\ (\mathsf{r}, \mathsf{e}) : \mathsf{is\_touching},\ (\mathsf{c}, \mathsf{e}) : \mathsf{is\_g\_inside}$

$\mathsf{f} : \mathsf{forest},\ (\mathsf{e}, \mathsf{f}) : \mathsf{is\_g\_inside}$

We use concept and role expressions as defined in the previous examples. The cottage is described by the individual c with a predicate-exists restriction asserting a floor space between 40 and 70 m^2. The cottage c has to be inside of an estate with a size between 350 and 450 m^2. As a simplification we assume that the river r has to touch the estate e that is inside of a forest f. Additionally, we assume the following new or revised concept definitions.

$\mathbf{estate} \sqsubseteq \mathsf{spatial\_area} \sqcap \exists\, \mathsf{has\_space} \,.\, \lambda_{\mathcal{R}} x \,.\, (\top_{\mathcal{R}}(x))$

$\mathbf{estate\_in\_forest} \doteq \mathsf{estate} \sqcap \exists\, \mathsf{is\_g\_inside} \,.\, \mathsf{forest}$

$\mathbf{fishing\_cottage} \doteq$

$\quad \mathsf{cottage} \sqcap \exists\, \mathsf{is\_g\_inside} \,.\, (\mathsf{estate} \sqcap \exists\, \mathsf{is\_touching} \,.\, \mathsf{river})$

The realizing component of the $\mathcal{ALCRP}(\mathcal{D})$ reasoner will compute the following parents (i.e. the most specific subsuming concepts) of the cottage c: normal_cottage_in_forest and fishing_cottage. The parents of the estate e will be estate_in_forest. The other individuals r and f keep their asserted concepts as parents.

With the help of an abstraction process we can replace Abox $\mathcal{A}_0$ by an Abox $\mathcal{A}_1$ containing a single assertion for c with the synthesized concept description $\mathsf{cottage}_{\mathsf{c}_1}$. The other concept definition is only used to enhance the readability of $\mathsf{cottage}_{\mathsf{c}_1}$.

$\mathbf{estate}_{\mathbf{e}_1} \doteq$

$\quad \mathsf{estate} \sqcap \exists\, \mathsf{is\_g\_inside} \,.\, \mathsf{forest} \sqcap \exists\, \mathsf{is\_touching} \,.\, \mathsf{river}$

$\mathbf{cottage}_{\mathbf{c}_1} \doteq$

$\quad \mathsf{cottage} \sqcap \exists\, \mathsf{is\_g\_inside} \,.\, \mathsf{estate}_{\mathsf{e}_1} \sqcap$

$\quad \exists\, \mathsf{has\_space} \,.\, \lambda_{\mathcal{R}} x \,.\, (x > 40 \wedge x < 70)$

The revised ABox $\mathcal{A}_1$ now consists only of the assertion $\mathsf{c} : \mathsf{cottage}_{\mathsf{c}_1}$. The newly created concept $\mathsf{cottage}_{\mathsf{c}_1}$ is classified by the reasoner and integrated into the concept taxonomy. The semantic validity of this query is automatically verified during classification, i.e. to check whether the concept is coherent (see Section 2.3). For instance, if the forest f were required to be 'mosquito-free' (see above), the $\mathcal{ALCRP}(\mathcal{D})$ reasoner would immediately recognize the incoherence of $\mathsf{cottage}_{\mathsf{c}_1}$. This information could be used by the spatial parser for generating an explanation to the user and for identifying the source of the contradiction.

Let us assume that the executed query $\mathsf{c} : \mathsf{cottage}_{\mathsf{c}_1}$ returns more than 100 matches. The next step for the user might be to refine the query by adding more constraints.[2] One could add more requirements to the estate, e.g. we ask for a garage connected to the cottage. The extended sketch (see Figure 7) corresponds to the ABox $\mathcal{A}_2$ (ignoring the lake) that results from adding to ABox $\mathcal{A}_0$ the following new assertions.

$\mathsf{g} : \mathsf{garage},\ (\mathsf{c}, \mathsf{g}) : \mathsf{is\_touching}$

The abstraction process reduces ABox $\mathcal{A}_2$ to ABox $\mathcal{A}_3$ consisting only of the assertion $\mathsf{c} : \mathsf{cottage}_{\mathsf{c}_2}$ using the following synthesized concept description.

$\mathbf{cottage}_{\mathbf{c}_2} \doteq$

$\quad \mathsf{cottage} \sqcap \exists\, \mathsf{has\_space} \,.\, \lambda_{\mathcal{R}} x \,.\, (x > 40 \wedge x < 70) \sqcap$

$\quad \exists\, \mathsf{is\_g\_inside} \,.\, \mathsf{estate}_{\mathsf{e}_1} \sqcap \exists\, \mathsf{is\_touching} \,.\, \mathsf{garage}$

The $\mathcal{ALCRP}(\mathcal{D})$ reasoner recognizes the relationship in the taxonomy that $\mathsf{cottage}_{\mathsf{c}_1}$ subsumes $\mathsf{cottage}_{\mathsf{c}_2}$. It can be rewritten as $\mathsf{cottage}_{\mathsf{c}_3}$ that even textually demonstrates the subsumption relationship.

$\mathbf{cottage}_{\mathbf{c}_3} \doteq \mathsf{cottage}_{\mathsf{c}_1} \sqcap \exists\, \mathsf{is\_touching} \,.\, \mathsf{garage}$

For executing the refined query the optimizer can benefit from the detected query subsumption and reduce the search space to the set of query matches already computed for ABox $\mathcal{A}_1$. Note that these query matches are members of the concept $\mathsf{cottage}_{\mathsf{c}_1}$. This type of query optimization is an important aspect in applying description logics to database theory (see [9] for an introduction to these topics).

[2] Of course, one of the most important criteria is the price of the estate. This is neglected due to the non-spatial nature of this part of the query.

The benefits of computing a concept subsumption taxonomy can be even more subtle. Imagine a query from another user looking for a cottage located in a forest that is connected to a river. The ABox $\mathcal{A}_4$ derived from the sketch might be structured as follows.

$\mathsf{c} : \mathsf{cottage},\ \mathsf{e} : \mathsf{estate\_area},\ (\mathsf{c}, \mathsf{e}) : \mathsf{is\_g\_inside}$

$\mathsf{r} : \mathsf{river},\ \mathsf{f} : \mathsf{forest},\ (\mathsf{f}, \mathsf{r}) : \mathsf{is\_connected},\ (\mathsf{e}, \mathsf{f}) : \mathsf{is\_g\_inside}$

The abstraction process creates the following concept definitions.

$$\mathbf{forest}_{\mathsf{f_2}} \doteq \mathsf{forest} \sqcap \exists\, \mathsf{is\_connected} \,.\, \mathsf{river}$$

$$\mathbf{estate}_{\mathsf{e_2}} \doteq \mathsf{estate} \sqcap \exists\, \mathsf{is\_g\_inside} \,.\, \mathsf{forest}_{\mathsf{f_2}}$$

$$\mathbf{cottage}_{\mathsf{c_4}} \doteq \mathsf{cottage} \sqcap \exists\, \mathsf{is\_g\_inside} \,.\, \mathsf{estate}_{\mathsf{e_2}}$$

The resulting ABox $\mathcal{A}_4$ consists only of the assertion $\mathsf{c} : \mathsf{cottage}_{\mathsf{c_4}}$. It turns out that the concept $\mathsf{cottage}_{\mathsf{c_4}}$ subsumes the other concepts $\mathsf{cottage}_{\mathsf{c}_i}$ although the concept descriptions are textually different. This is a rather complex proof based on the interaction between the spatial relations: $\mathsf{g\_inside}(\mathsf{e}, \mathsf{f}) \wedge \mathsf{touching}(\mathsf{e}, \mathsf{r}) \Rightarrow \mathsf{connected}(\mathsf{f}, \mathsf{r})$.

The abstraction process works rather well for ABoxes containing no joins or cycles, i.e. the same individual is a filler of several roles or even related to itself through a cycle of role assertions. If joins or cycles are present in an ABox, it depends on the expressiveness of the description logic whether an ABox can be reduced to a single concept membership assertion. For instance, joins can be expressed by restricting the number of possible role fillers or by equality restrictions for feature fillers. As mentioned above, other DLs also support the definition of cyclic concepts that might be required to fully reduce some ABoxes. Due to unknown decidability results $\mathcal{ALCRP}(\mathcal{D})$ currently does not allow cyclic concepts or number restrictions. Therefore, in case of ABoxes with joins or cycles, we can only partially reduce these ABoxes. This is illustrated in Figure 7 by adding a lake. The river has to flow into the lake and the same lake is touching the forest. This is an example for a join in a corresponding ABox. However, the reasoning with $\mathcal{ALCRP}(\mathcal{D})$ as described above is still valid and usable for query processing. Only the subsumption between ABox queries requires a more sophisticated approach.

4 Conclusion and Future Work

The formalism presented in this paper can be used to define the semantics of visual spatial queries, to reason about query subsumption, and to deal with multiple worlds or query completion with the help of default reasoning. The open problem with joins or cycles in ABoxes will be addressed by additionally utilizing graph matching techniques for Aboxes. This is work in progress.

References

[1] V. Haarslev and M. Wessel, "Querying GIS with animated spatial sketches", in *1997 IEEE Symposium on Visual Languages, Capri, Italy, Sep. 23-26.* Sept. 1997, pp. 197–204, IEEE Computer Society Press, Los Alamitos.

[2] M. Wessel and V. Haarslev, "VISCO: Bringing visual spatial querying to reality", in *1998 IEEE Symposium on Visual Languages, Halifax, Canada, Sep. 1-4.* Sept. 1998, pp. 170–177, IEEE Computer Society Press, Los Alamitos.

[3] V. Haarslev, C. Lutz, and R. Möller, "Foundations of spatioterminological reasoning with description logics", in *Proceedings of Sixth International Conference on Principles of Knowledge Representation and Reasoning (KR'98), Trento, Italy, June 2-5, 1998*, T. Cohn, L. Schubert, and S. Shapiro, Eds., June 1998, pp. 112–123.

[4] V. Haarslev, C. Lutz, and R. Möller, "A description logic with concrete domains and a role-forming predicate operator.", *Journal of Logic and Computation*, vol. 9, no. 3, 1999, In press.

[5] R. Möller and M. Wessel, "Terminological default reasoning about spatial information: A first step", in *Proc. of COSIT'99, International Conference on Spatial Information Theory, Stade.* 1999, Springer Verlag, Berlin, In press.

[6] V. Haarslev, "A fully formalized theory for describing visual notations", in *Visual Language Theory*, K. Marriott and B. Meyer, Eds., pp. 261–292. Springer Verlag, Berlin, 1998.

[7] V. Haarslev, "A logic-based formalism for reasoning about visual representations (extended abstract)", in *Proceedings, AAAI Workshop on Formalizing Reasoning with Visual and Diagrammatic Representations, AAAI Fall Symposium Series 1998, October 23-25, Orlando, Florida/USA. Technical Report FS-98-04.* 1998, pp. 57–66, AAAI Press.

[8] W.A. Woods and J.G. Schmolze, "The KL-ONE family", in *Semantic Networks in Artificial Intelligence*, F. Lehmann, Ed., pp. 133–177. Pergamon Press, Oxford, 1992.

[9] A. Borgida, "Description logics in data management", *IEEE Transactions on Knowledge and Data Engineering*, vol. 7, no. 5, pp. 671–682, 1995.

[10] R.J. Brachman and J.G. Schmolze, "An overview of the KL-ONE knowledge representation system", *Cognitive Science*, pp. 171–216, Aug. 1985.

[11] A. Tarski, *A Decision Method for Elementary Algebra and Geometry*, University of California Press, Berkeley, CA, 1951.