Modeling Spatial Objects with Undetermined Boundaries Using the Realm/ROSE Approach[1]

Markus Schneider

Praktische Informatik IV
FernUniversität Hagen
Postfach 940
D-58084 Hagen
Germany
schneide@fernuni-hagen.de

Abstract: The purposes of this paper are twofold, namely first to present general criteria for the design of spatial data types that are necessary and valid for the modeling of spatial objects regardless whether we consider objects with sharp or undetermined boundaries, and second to show how the concepts of a type system for spatial objects with sharp boundaries can be suitably transferred and extended to the modeling of spatial objects with undetermined boundaries. The most relevant design criteria of such a type system comprise generality, closure properties, rigorous definition, finite resolution, numerical robustness, topological correctness, geometric consistency, extensibility and data model independence of spatial data types. The Realm/ROSE approach allows for these design criteria and offers an appropriate definition of a type system for spatial objects with sharp boundaries. An extension of the Realm/ROSE model is proposed that shows how general region objects with undetermined boundaries can be derived from this model which have nice closure properties and which obey the design criteria. The idea is to consider determined zones surrounding the undetermined borders of the object and expressing its minimal and maximal extension.

Keywords: Spatial data types, algebra, realm, ROSE, generality, closure properties, rigorous definition, finite resolution, numerical robustness, topological correctness, geometric consistency, extensibility, data model independence, vagueness, uncertainty, fuzziness, vague regions

[1] This work was supported by the DFG (Deutsche Forschungsgemeinschaft) under grant Gu 293/1-2

1 Introduction

The diversity of geometric applications has led to a lot of proposals both for the modeling of spatial data and for the design of new data models and query languages integrating traditional alphanumeric data as well as geometric data. In literature by now the general opinion prevails that special data types are necessary to model geometry and to enable geometric data to be efficiently represented in database systems. These data types are commonly denoted as *spatial* or *geometric data types* (SDT), such as, for example, *point*, *line*, or *region*. We speak of *spatial objects* as occurrences of spatial data types. Thus, we take an entity-oriented view of spatial phenomena. The definition of spatial data types and operations expressing the spatial semantics visible at the user level and the mechanisms for providing them to the user are to a high degree responsible for the design of a spatial data model and the performance of a spatial database system being the basis of GIS and have a great influence on the expressiveness of spatial query languages. This is true regardless whether we consider spatial objects with sharp or undetermined boundaries and whether a DBMS uses a relational, complex object, object-oriented, or some other data model. *Hence, the definition and implementation of spatial data types is probably the most fundamental issue in the development of spatial database systems.*

Spatial data types modeling objects with sharp boundaries are used routinely in the description of spatial query languages (e.g. [Eg89, Gü88, JC88, LN87, SH91]) and have been implemented in some prototype systems (e.g. [Gü89, OM88, RFS88]) even if only a few formal definitions have been given for them [Gü88, GS93, GS95, GNT91, SV89]. For spatial objects with undetermined boundaries analogous approaches, especially formal ones, are unknown to the author.

The treatment of spatial objects with undetermined, vague, and blurred boundaries is especially problematic for computer scientists who are confronted with the difficulties how to model such objects in their database systems, how to finitely represent them in a computer format, how to develop spatial index structures for them, and how to draw them. They are accustomed to the abstraction process of simplifying spatial phenomena of the real world to simply-structured, manageable, and sharply-bounded objects of Euclidian geometry like points, lines, and regions. On the other hand this abstraction process itself mapping reality onto a mathematical model introduces a certain kind of vagueness and imprecision.

Spatial objects with undetermined boundaries are difficult to model and so far rarely or not supported in spatial database systems. Two categories of vagueness and indeterminacy concerning spatial objects can be distinguished. *Uncertainty* relates either to a lack of knowledge about the position and shape of an object with an existing, real border or to the inability of measuring such an object precisely. *Fuzziness* describes the vagueness of objects which certainly have an extension but which inherently do not have a precisely definable border.

At least three alternatives are conceivable as general design methods for the modeling of spatial objects with undetermined boundaries: (a) fuzzy models [Ba93, Du91, HB93, LGL92, LL93], (b) probabilistic models [Fi93], and (c) transfer and extension of data models, methods, and concepts for spatial objects with sharp boundaries to spatial objects without clear boundaries. In this paper we pursue the third approach and extend the Realm/ROSE model [GS93, GS95] as an algebraic model for handling spatial objects with sharp boundaries to a model for spatial objects with undetermined boundaries which contemporaneously obeys general criteria for the design of spatial data types and which preserves the properties of the Realm/ROSE model. The idea is to consider determined zones surrounding the undetermined borders of the object and expressing its minimal and maximal extension. The zones serve as a description and separation of the space that certainly belongs to the region object and the space that is certainly outside. Similar approaches have been pursued in [CF94] and in [CG94] which both presuppose some kind of zone concept for the modeling of vague spatial objects. But in contrast to this paper they are mainly interested in classifications of topological relationships between vague spatial objects and not in a precise formal modeling of the objects themselves.

Section 2 introduces general criteria for the design of spatial data types regardless of the determinacy or indeterminacy of its objects. Section 3 informally sketches the Realm/ROSE model, and section 4 shows how this concept can be used to formally model general region objects with undetermined boundaries having nice closure properties.

2 General Criteria for the Design of Spatial Data Types

General design criteria for spatial data types are stated which are considered to be relevant for the modeling of spatial objects and which are valid regardless whether we consider objects with sharp or undetermined boundaries. The current modeling approaches for spatial objects with determined boundaries only partially follow these criteria. Within the framework of the Realm/ROSE model we have attempted to take all these criteria into account and to offer a satisfactory solution in a single model. The design criteria are in detail:

- *Generality.* It should be feasible to model spatial objects being the occurrences of SDTs as general as possible. A line object should be able to model the ramification of the Nile delta. A region object should be able to represent a collection of disjoint areas each of which may have holes. This allows, for instance, to model the German state of Niedersachsen enclosing the state of Bremen and having offshore islands in the North Sea as one object.

- *Closure properties*. The domains of spatial data types like *point*, *line*, and *region* must be closed under union, intersection, and difference of their underlying point sets. This allows the definition of powerful data type operations with nice closure properties. When observing this criterion geometric anomalies are avoided which can occur when for instance conventional operations in set theory and point set topology are carried out, a problem which for this case has been solved by regularized operations [Ti80].

- *Rigorous definition*. The semantics of SDTs, that is, the possible values for the types and the functions associated with the operations, require a *formal*, clear, and unique definition to avoid ambiguities both for the user and the implementor.

- *Finite resolution, numerical robustness, and topological correctness*. The formal definitions must take into account *the finite representations available in computers*. This has so far been neglected in definitions of SDTs. It is left to the programmer to close the gap between theory and practice which leads rather inevitably not only to numerical but also to topological errors.

- *Geometric consistency.* Distinct spatial objects may be related through geometric consistency constraints (e.g. adjacent regions have a common boundary, or two lines meet in a point). The definition of SDTs must offer facilities to enforce such consistency.

- *Extensibility*. Even though the designer of a spatial database system may provide a good collection of spatial data types and operations, there will always be applications requiring further operations on existing types or requiring new types with new operations. A type system should therefore be extensible for new data types.
- *Data model independence*. Spatial data types as such are rather useless; they need to be integrated into a DBMS data model and query language. However, a definition of SDTs should be valid regardless of a particular DBMS data model and therefore not depend on it. Instead, the SDT definition should be based on a general abstract interface to the DBMS data model.

These design criteria have to be transferred to and realized at the implementation level when constructing spatial database systems.

3 The Realm/ROSE Model : An Informal Overview

In this section we present a short, intuitive and informal overview of the realm concept and the ROSE (*RO*bust *S*patial *E*xtension) algebra which both support an entity-oriented view of spatial reality and which were originally only planned for spatial objects with sharp boundaries. Any formal definitions are omitted here, and the reader interested in details is referred to [GS93, GS95].

A *realm* used as a basis for spatial data types is essentially a finite set of points and *non-intersecting* line segments over a discrete domain (Figure 1) and can from a graph-theoretical point of view be viewed as a planar graph over a finite resolution grid. Intuitively, it describes the

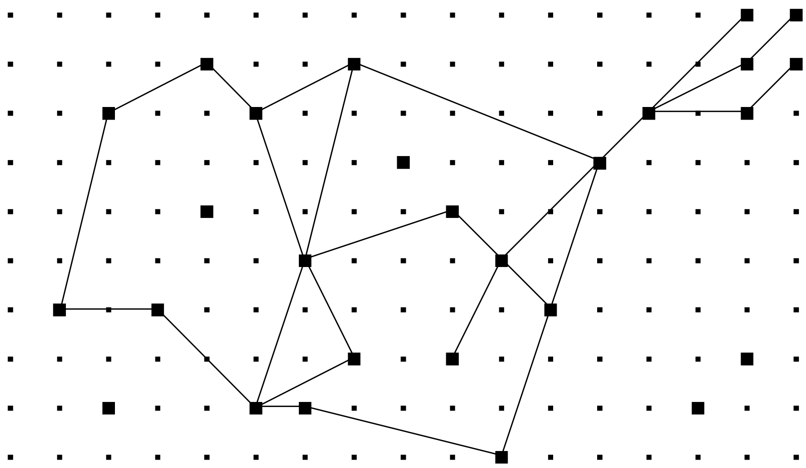


Figure 1

complete underlying geometry of an application. All spatial objects like points, lines and regions can be defined in terms of points and line segments present in the realm. In fact, in such a database spatial objects are never created directly but only by suitably selecting some realm objects and composing them to spatial objects. They are never updated directly. Instead, updates are performed on the realm and from there propagated to the dependent spatial objects. Hence, all spatial objects occuring in a database are *realm-based*.

Figure 2 shows some spatial objects definable over the realm of Figure 1. The realm-based spatial data types are called <u>*points*</u>, <u>*lines*</u>, and <u>*regions*</u> and are the sorts (types) of the ROSE algebra.

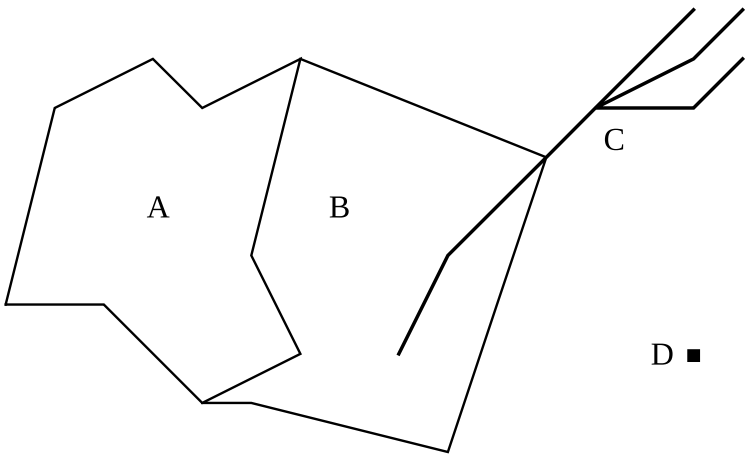


Figure 2

Hence, A and B represent *regions* objects, C is a *lines* object, and D a *points* object. One can imagine A and B to belong to two adjacent countries, C to represent a river, and D a city.

The underlying grid of a realm arises simply from the fact that numbers have a finite representation in computer memory. In practice, these representations will be of fixed length and correspond to INTEGER or REAL data types available in programming languages. Of course, the resolution selected for a concrete application will be much finer than could be shown in Figure 1.

The realm concept as a basis of spatial data types serves the following purposes:

- It guarantees nice *closure properties* for the computation with spatial data types above the realm. The algebraic operations for the spatial data types are defined in a way to construct only spatial objects that are realm-based as well. For example, the intersection of region B with line C (the part of river C lying within country B) is also a realm-based *lines* object. So the spatial algebra is closed with respect to a given realm. This means in particular that no two objects of spatial data types occurring in geometric computation have “proper” intersections of line segments. Instead, two initially intersecting segments have already been split at the intersection point when they were entered into the realm. One could say that any two intersecting SDT objects “have become acquainted” already when they were entered into the realm. This is a crucial property for the correct and efficient implementation of geometric operations.

- It shields geometric computation in query processing from numeric correctness and robustness problems. This is because such problems arise essentially from the computation of intersection points of line segments which normally do not lie on the grid[2]. With realm-based SDTs, there are *never any new intersection points computed* in query processing. Instead, the numeric problems are treated *below and within* the realm level, namely, whenever updates are made to a realm.

---

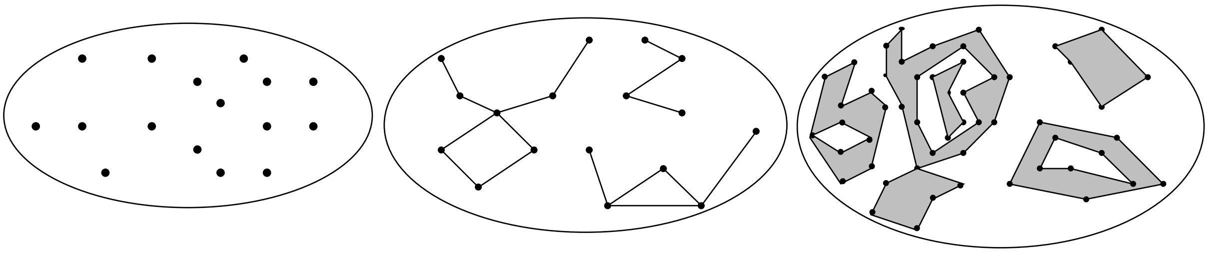
[2] The methods for the treatment of numeric correctness problems below and within the realm level and especially the important problem of mapping an application’s set of intersecting line segments into a realm’s set of non-intersecting line segments are an interesting and complex problem but out of the scope of this paper (see [GS93]).

- It provides the programmer with a precise specification on all levels of the model that directly lends itself to a correct implementation. This particularly means that the spatial algebra obeys algebraic laws precisely in theory as well as in practice.

- It enforces *geometric consistency* of related spatial objects. For example, the common part of the borders of countries A and B is exactly the same for both objects.

Certain structures can be constructed in a realm that serve as a basis for the definition of SDTs. Let us view a realm as a planar graph. Then an *R-cycle* is a cycle of this graph. An *R-face* is an *R*-cycle possibly enclosing some other disjoint *R*-cycles corresponding to a region with holes. An *R-unit* is a minimal *R*-face. These three notions support the definition of a *regions* data type. An *R-block* is a maximal connected component of the realm graph; it supports the definition of a *lines* data type. For all of these *realm-based structures* predicates are defined to describe their possible topological relationships.

The ROSE algebra contains very general data types *points*, *lines*, and *regions* (Figure 3). Let *R* be a realm. Then a *points* object is a set of *R*-points. There are two alternative views of *lines* and *regions*. The first “flat” view is that a *lines* object is a set of *R*-segments and a *regions* object a set of *R*-units. The other “structured” view is equivalent but “semantically richer”: A *lines* object is a set of disjoint *R*-blocks and a *regions* object a set of (edge-) disjoint *R*-faces. For example, it is now possible to represent the whole area of a state including islands or separate land areas in a single *regions* object, or a complete highway network in a single *lines* object. Especially, the modeling of “regions with holes” is now possible.



a *points* object a *lines* object a *regions* object

Figure 3

The definition of these data types guarantees very nice closure properties. They are closed under the geometric operations *union*, *intersection*, and *difference* with regard to the same realm. That is, the result of such an operation is a realm-based object as well and corresponds to the definitions of the spatial data types informally given above. The validity of the closure properties is based on the reduction of the geometric operations to the corresponding set-theoretic ones.

The spatial operations of the ROSE algebra [GS95] are divided into four classes. Note that the last group of operations manipulates not only spatial objects but also the geographical objects they are associated with. The classes are:

- spatial predicates expressing topological relationships (e.g. **inside**, **adjacent**, **disjoint**)
- operations returning atomic spatial objects (e.g. **intersection**, **contour**, **plus**, **minus**)
- operations returning numbers (e.g. **length**, **dist**, **diameter**, **area**)
- operations on sets of geographical objects (e.g. **overlay**, **fusion**, **closest**, **decompose**).

4 Using the Realm/ROSE Approach for Modeling Spatial Objects with Undetermined Boundaries

An extension of the Realm/ROSE model to spatial objects with undetermined boundaries leads to very general data types <u>*vpoints*</u>, <u>*vlines*</u>, and <u>*vregions*</u> where the prefix "v" stands for the term "vague" unifying the two categories of uncertain and fuzzy spatial objects. These vague objects are to be defined by "sharp" means using some concepts and definitions of the Realm/ROSE model. Within the framework of this paper we confine ourselves to the formal treatment of general *regions with undetermined boundaries* or *vague regions* with possibly existing *vague holes* and to the treatment of their closure properties.

The central idea is to approximate each of the undetermined boundaries of a region object, that is, its outer boundary line and the boundary lines of each of its possibly existing holes, by *zones* modeling a kind of "irregular spatial intervals" which we call *border zone* and *hole zone*, respectively (Figure 4). A border zone is modeled by two or more simple cycles, one representing its *outer border* and one or more representing its *inner border(s)*. A hole zone is modeled by two simple cycles representing its *inner border* and its *outer border*. Matching inner and outer borders surround the undetermined borders of the outer boundary line and the holes on both sides so that for a vague region border zone and hole zones express the vagueness of the real, undetermined boundary lines which lie somewhere between the outer border and the inner border(s) of the zones. For a zone the area of its inner border(s) (the "safe" area) is always contained in

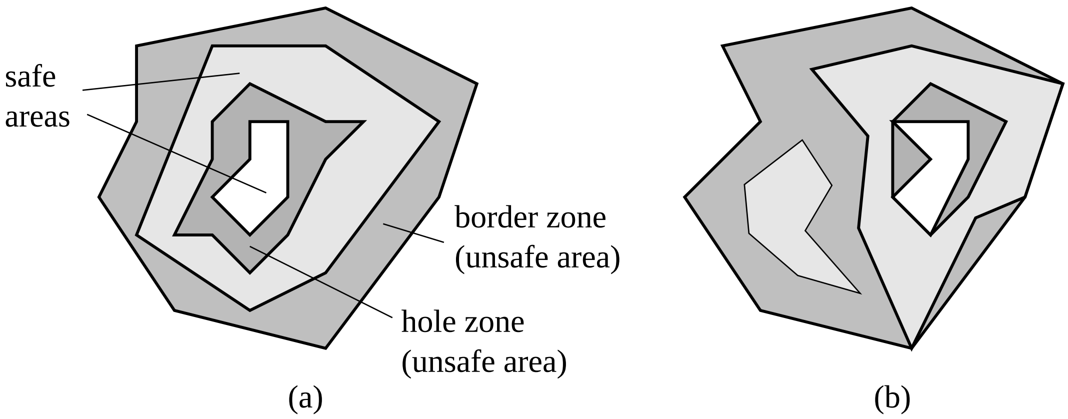


Figure 4: (a) A vague region with a vague hole and the corresponding zones. (b) Note that the inner border of a border or hole zone may have common segments and common points with its outer border and that there can be more than one inner border for border zones.

the area of its outer border. All the zones together serve as a description and separation of the space that certainly belongs to the vague region and the space that is certainly outside. Hence, the maximal extension of a vague region is given by the outer border of its outer boundary line and the inner borders of its holes. The minimal extension is given by the inner border(s) of its outer boundary line and the outer borders of its holes.

As an example, we can consider a lake which has a minimal water level in dry and a maximal water level in rainy periods. Dry periods can entail puddles where is no water. Small islands in the lake which are less flooded by water in dry and more flooded in rainy periods can then be modeled as vague holes. Even in rainy periods an island is never flooded completely. The example underpins that this modeling approach is suitable for describing vague regions and that it corresponds to the user’s conceptual view and intuition of spatial vagueness.

We now take a more formal view of vague regions and use concepts of the Realm/ROSE model. Let $N = \{0, ..., n-1\} \subseteq \mathbf{N}$. An *N-point* is defined as a pair $(x, y) \in N \times N$. An *N-segment* is a pair of distinct *N*-points (p, q). P_N denotes the set of all *N*-points and S_N the set of all *N*-segments. Two *N*-segments <u>meet</u> if they have exactly one end point in common.

An *R-cycle c* is a cycle in the graph interpretation of a realm, defined by a set of *R*-segments $S(c) = \{s_0, ..., s_{m-1}\}$, such that

(i) $\forall\, i \in \{0, ..., m\text{-}1\} : s_i$ <u>meets</u> $s_{(i+1) \bmod m}$

(ii) No more than two segments from $S(c)$ <u>meet</u> in any point p.

Cycle c partitions the set P_N into three subsets $P_{in}(c)$, $P_{on}(c)$, and $P_{out}(c)$ of *R*-points lying inside, on, and outside c. Let $P(c) := P_{on}(c) \cup P_{in}(c)$. Cycles are interesting because they are the basic entities over realms for the definition of objects with a spatial extension. The relationships that may be distinguished between two *R*-cycles c_1 and c_2 are shown in Figure 5.

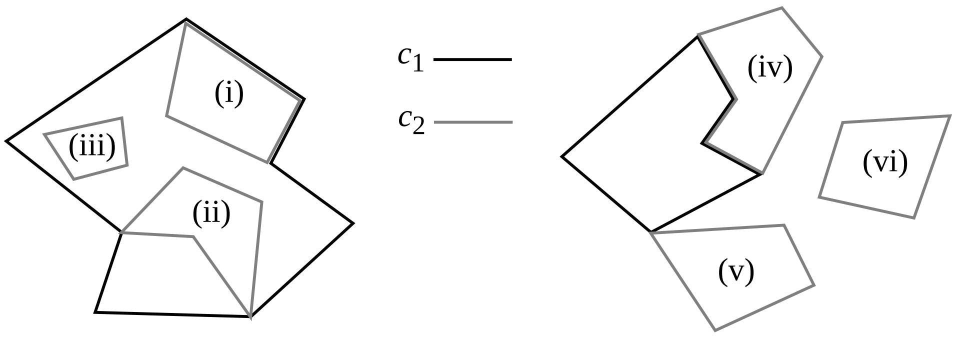


Figure 5

The following terminology is used for these configurations: c_2 is (<u>*area-*</u>)<u>*inside*</u> (i, ii, iii), <u>*edge-inside*</u> (ii, iii), or <u>*vertex-inside*</u> (iii) c_2. c_1 and c_2 are <u>*area-disjoint*</u> (iv, v, vi), <u>*edge-disjoint*</u> (v, vi), or (<u>*vertex-*</u>)<u>*disjoint*</u> (vi).

The meaning is that (i) c_2 is (w.r.t. *area*) *inside* c_1, (ii) additionally has no common edges with c_1, (iii) has not even common vertices with c_1. Similarly (iv) c_2 is *disjoint* w.r.t. *area* with c_1, (v) additionally has no common edges with c_1, (vi) additionally has not even common vertices with c_1. *area-inside* is the standard interpretation of the term *inside*, *vertex-disjoint* the standard interpretation of the term *disjoint*. Formally these predicates are defined as follows:

c_1 (<u>*area-*</u>)<u>*inside*</u> c_2	$:\Leftrightarrow$	$P(c_1) \subseteq P(c_2)$
c_1 <u>*edge-inside*</u> c_2	$:\Leftrightarrow$	c_1 <u>*area-inside*</u> $c_2 \wedge S(c_1) \cap S(c_2) = \varnothing$
c_1 <u>*vertex-inside*</u> c_2	$:\Leftrightarrow$	c_1 <u>*edge-inside*</u> $c_2 \wedge P_{on}(c_1) \cap P_{on}(c_2) = \varnothing$
c_1 and c_2 are <u>*area-disjoint*</u>	$:\Leftrightarrow$	$P_{in}(c_1) \cap P(c_2) = \varnothing \wedge P_{in}(c_2) \cap P(c_1) = \varnothing$
c_1 and c_2 are <u>*edge-disjoint*</u>	$:\Leftrightarrow$	c_1 and c_2 are <u>*area-disjoint*</u> $\wedge S(c_1) \cap S(c_2) = \varnothing$
c_1 and c_2 are (<u>*vertex-*</u>)<u>*disjoint*</u>	$:\Leftrightarrow$	c_1 and c_2 are <u>*edge-disjoint*</u> $\wedge P_{on}(c_1) \cap P_{on}(c_2) = \varnothing$

Based on the concept of *R*-cycles, the notions *R-face* and *R-unit* are introduced which describe regions from two different perspectives and which are used equivalently. Both of them essentially define polygonal regions with holes. An *R*-unit is a “minimal” *R*-face in the sense that any

R-face within the *R*-unit is equal to the *R*-unit. Hence *R*-units are the smallest region entities that exist over a realm.

An *R-face* f is a pair (c, H) where c is an *R*-cycle and $H = \{h_1, ..., h_m\}$ is a (possibly empty) set of *R*-cycles such that the following conditions hold (let $S(f)$ denote the set of segments of all cycles of f):

(i) $\forall\, i \in \{1, ..., m\} : h_i$ edge-inside c

(ii) $\forall\, i, j \in \{1, ..., m\}, i \neq j : h_i$ and h_j are edge-disjoint

(iii) Each cycle in $S(f)$ is either equal to c or to one of the cycles in H (no other cycle can be formed from the segments of f)

The last condition ensures uniqueness of representation, that is, there are no two different interpretations of a set of segments as sets of faces. The grid points belonging to an *R*-face f are defined as $P(f) := P(c) \setminus \bigcup_{i=1}^{m} P_{in}(h_i)$.

Let $F(R)$ denote the set of all possible *R*-faces and $U(R)$ denote the set of all *R*-units for a realm R. In [GS93] the equivalence of two representations of a region over a realm is formally established, namely, as a set of (pairwise) edge-disjoint *R*-faces, and as a set of area-disjoint *R*-units. Operations called *faces* and *units* are defined to convert between the two formal representations. Hence the equivalence can be expressed as: $\forall\, F \subseteq F(R)$: $faces(units(F)) = F$. The operation *units* is defined as $units(F) := \{u \in U(R) \mid \exists\, f \in F\colon u$ area-inside $f\}$. The operation *faces* basically works as follows: From a given set of area-disjoint *R*-units, its multiset of boundary segments is formed. Then, all segments occurring twice are removed. The remaining set of segments defines uniquely a set of edge-disjoint *R*-faces.

Now we are able to formally define vague regions. Two equivalent definitions are conceivable expressing slightly different conceptual views. The first definition supports the zone concept whereas the second definition emphasizes the maximal and minimal extension of a vague region.

Let $C = (c^{out}, C^{in})$ denote a pair of a single *R*-cycle c^{out} and a non-empty set of *R*-cycles $C^{in} = \{c_1^{in}, ..., c_n^{in}\}$, and let $H = (H^{out}, H^{in})$ denote a pair of (possibly empty) sets of *R*-cycles $H^{out} = \{h_1^{out}, ..., h_m^{out}\}$ and $H^{in} = \{h_1^{in}, ..., h_m^{in}\}$. Then a *vague region vr* is a pair (C, H) so that the following conditions are satisfied:

(i) $\forall\, i \in \{1, ..., n\} : c_i^{in}$ area-inside c^{out}

(ii) $\forall\, k, l \in \{1, ..., n\}, k \neq l : c_k^{in}$ edge-disjoint c_l^{in}

(iii) $\forall\, k \in \{1, ..., m\}\ \exists\, l \in \{1, ..., n\} : h_k^{out}$ edge-inside c_l^{in}

(iv) $\forall\, k, l \in \{1, ..., m\}, k \neq l : h_k^{out}$ edge-disjoint h_l^{out}

(v) There exist two bijective functions $f : \{1, ..., m\} \to H^{out}$ and $g : \{1, ..., m\} \to H^{in}$ such that $\forall\, i \in \{1, ..., m\} : g(i)$ area-inside $f(i)$

These conditions reflect the informal zone model of a vague region presented above. C models the unsafe border zone consisting of the outer border c^{out} and a non-empty set C^{in} of inner borders. H models the unsafe hole zones consisting of a set H^{out} of outer borders and a corresponding set H^{in} of inner borders. The conditions describe the inclusion relationships between the four kinds of cycles that occur in unsafe border and hole zones, the disjointedness relationship between each pair of inner borders of the unsafe border zone, and the disjointedness relationship between each pair of outer borders of the unsafe hole zones. Condition (v) requires that exactly one inner border of H^{in} lies inside exactly one outer border of H^{out}. Note that f and g are total functions, since domain and codomain of each function have the same cardinality. All conditions together prevent a proper intersection of borders (cycles).

Based on these five conditions, an equivalent definition for *vr* as a pair of *regions* objects (R^{out}, R^{in}) can be given. The first component of this pair is defined as $R^{out} := \{(c^{out}, H^{in})\}$ which describes a set with a single *R*-face. R^{out} is a *regions* object, since all inner borders h_i^{in} of unsafe hole zones are edge-inside to the outer border c^{out} of the unsafe border zone (follows from conditions (v), (iii), and (i)) and since they are pairwise edge-disjoint (follows from conditions (v) and (iv)). R^{in} represents a set of edge-disjoint *R*-faces and is defined as $R^{in} := \{(c_i^{in}, H_i^{out}) \mid i \in \{1, ..., n\}\}$ with $H_i^{out} := \{h \in H^{out} \mid h \textit{ edge-inside } c_i^{in}\}$. R^{in} is a *regions* object, since all inner borders of the unsafe border zone are pairwise edge-disjoint (follows from condition (ii)) and since for an inner border c_i^{in} all cycles of H_i^{out} are pairwise edge-disjoint (follows from condition (iv)). It is obvious that $\bigcup_{i=1}^{n} H_i^{out} = H^{out}$ and that $\forall\, k, l \in \{1, ..., n\}, k \neq l : H_k^{out} \cap H_l^{out} = \emptyset$. Intuitively, R^{out} represents the maximal and R^{in} the minimal extent of *vr*. We will assume this latter definition as the formal definition of a vague region.

Let us now define what it means that two vague regions $vr_1 = (R_1^{out}, R_1^{in})$ and $vr_2 = (R_2^{out}, R_2^{in})$ are edge-disjoint (for the definition of the predicate <u>*edge-disjoint*</u> between <u>*regions*</u> objects/ *R*-faces see [GS93]):

$$vr_1 \ \underline{\textit{edge-disjoint}}\ vr_2 \quad :\Leftrightarrow \quad R_1^{out} \ \underline{\textit{edge-disjoint}}\ R_2^{out}$$

The realm-based structure of a vague region forms the basis for the definition of the spatial data type <u>*vregions*</u>.

> *For a given realm R, a value of type <u>vregions</u> is a set of pairwise edge-disjoint vague regions.*

We now have to show the closure properties of the data type <u>*vregions*</u>, that is, it must be closed under the geometric operations union, intersection, and difference with regard to the same realm. Let w.l.o.g. $VR_1 = \{(R_1^{out}, R_1^{in})\}$ and $VR_2 = \{(R_2^{out}, R_2^{in})\}$ be two (one-component) <u>*vregions*</u> objects. Then

$$\textbf{\textit{union}}\ (VR_1, VR_2) := \textit{decompose}(\textbf{\textit{union}}(R_1^{out}, R_2^{out}), \textbf{\textit{union}}(R_1^{in}, R_2^{in})).$$

For ***intersection*** and ***difference*** the definitions are analogous. Since each of the three geometric operations when applied to the <u>*regions*</u> objects R_1^{out} and R_2^{out} normally leads to a <u>*regions*</u> object, say R^{out}, with more than one edge-disjoint *R*-face, R^{out} must be decomposed into its *R*-faces, and the uniquely matching set of edge-disjoint *R*-faces from the same geometric operation applied to R_1^{in} and R_2^{in} must be assigned to each such *R*-face in order to form a set of edge-disjoint vague regions. This is the task of the operation *decompose* whose formal definition is omitted here. The definitions of the geometric operations can be simply generalized to many-component <u>*vregions*</u> objects. Due to the underlying realms, these operations both in theory *and* in practice obey the usual algebraic laws, for instance, commutative, associative, and distributive laws.

5 Conclusions

The first part of the paper enumerated relevant design criteria for the modeling of spatial data types that are valid regardless whether we consider objects with sharp or undetermined boundaries. The second part showed how general vague region objects with nice closure properties can be defined on the basis of the Realm/ROSE approach, an algebraic model for constructing sharply-bounded spatial objects. Using and extending a "sharp" model can lead to success and meet the user's conceptual view and intuition of spatial vagueness.

Future work will have to relate to the formal definition of the data types *vpoints* and *vlines* and to the formal definition of vague spatial operations like vague topological relationships.

References

[Ba93] R. Banai. Fuzziness in Geographical Information Systems: Contributions from the Analytic Hierarchy Process. *Int. Journal of Geographical Information Systems*, vol. 7, no, 4, pp. 315-329, 1993.

[CF94] E. Clementini & P. d. Felice. An Algebraic Model for Spatial Objects with Undetermined Boundaries. *GISDATA Specialist Meeting on Spatial Conceptual Models for Geographic Objects with Undetermined Boundaries*, Baden (Austria), 1994.

[CG94] A.G. Cohn & N.M. Gotts. Expressing Spatial Vagueness in Terms of Connection. *GISDATA Specialist Meeting on Spatial Conceptual Models for Geographic Objects with Undetermined Boundaries*, Baden (Austria), 1994.

[Du91] S. Dutta. Topological Constraints: A Representational Framework for Approximate Spatial and Temporal Reasoning. *Second Symp. on Advances in Spatial Databases*, pp. 161-180,1991.

[Eg89] M.J. Egenhofer. Spatial SQL: A Spatial Query Language. Report 103, Dept. of Surveying Engineering, University of Maine, 1989.

[Fi93] J.T. Finn. Use of the Average Mutual Information Index in Evaluating Classification Error and Consistency. *Int. Journal of Geographical Information Systems*, vol. 7, no, 4, pp. 349-366, 1993.

[GNT91] M. Gargano, E. Nardelli & M. Talamo. Abstract Data Types for the Logical Modeling of Complex Data. *Information Systems*, vol. 16, no. 5, 1991.

[Gü88] R.H. Güting. Geo-Relational Algebra: A Model and Query Language for Geometric Database Systems. *Proc. of the Intl. Conf. on Extending Database Technology*, pp. 506-527, 1988.

[Gü89] R.H. Güting. Gral: An Extensible Relational Database System for Geometric Applications. *Proc. of the 15th Intl. Conf. on Very Large Databases*, pp. 33-44, 1989.

[GS93] R.H. Güting & M. Schneider. Realms: A Foundation for Spatial Data Types in Database Systems. *Proc. 3rd Intl. Symposium on Large Spatial Databases*, pp. 14-35, 1993.

[GS95] R.H. Güting & M. Schneider. Realm-Based Spatial Data Types: The ROSE Algebra. *VLDB Journal*, vol.4, pp. 100-143, 1995.

[HB93] G.B.M. Heuvelink & P.A. Burrough. Error Propagation in Cartographic Modelling Using Boolean Logic and Continuous Classification. *Int. Journal of Geographical Information Systems*, vol. 7, no, 3, pp. 231-246, 1993.

[JC88] T. Joseph & A. Cardenas. PICQUERY: A High Level Query Language for Pictorial Database Management. *IEEE Transactions on Software Engineering*, vol. 14, pp. 630-638, 1988.

[LGL92] Y. Leung, M. Goodchild & C.-C. Lin. Visualization of Fuzzy Scenes and Probability Fields. *Fifth Int. Symp. on Spatial Data Handling*, pp. 480-490, 1992.

[LL93] Y. Leung & K.S. Leung. An Intelligent Expert System Shell for Knowledge-Based Geographical Information Systems: 1. The Tools. *Int. Journal of Geographical Information Systems*, vol. 7, no, 3, pp. 189-199, 1993.

[LN87] U. Lipeck & K. Neumann. Modelling and Manipulating Objects in Geoscientific Databases. *Proc. of the 5th Intl. Conf on the Entity-Relationship Approach*, pp. 67-86, 1987.

[SH91] P. Svensson & Z. Huang. Geo-SAL: A Query Language for Spatial Data Analysis. *Proc. of the 2nd Int. Symposium on Large Spatial Databases*, pp. 119-140, 1991.

[SV89] M. Scholl & A. Voisard. Thematic Map Modeling. *Proc. of the 1st Int. Symposium on Large Spatial Databases*, pp. 167-190, 1989.

[Ti80] R.B. Tilove. Set Membership Classification: A Unified Approach to Geometric Intersection Problems. *IEEE Transactions on Computers*, vol. C-29, pp. 874-883, 1980.