

Topological Predicates between Vague Spatial Objects

Alejandro Pauly & Markus Schneider*

University of Florida
Department of Computer & Information Science & Engineering
Gainesville, FL 32611, USA
{apauly, mschneid}@cise.ufl.edu

Abstract. Topological predicates are an important element of database systems that allow manipulation of spatial data. Based on the necessity for such systems to handle uncertainty, we introduce a general mechanism that identifies *vague topological predicates*. This definition forms part of a formal data model referred to as *VASA (Vague Spatial Algebra)*, in which the data types *vague regions*, *vague lines*, and *vague points* are defined in terms of existing definition of crisp spatial data types. Following this trend, the mechanism presented here identifies vague topological predicates on the basis of well defined crisp topological predicates. An example implementation of the mechanism for vague regions is given.

1 Introduction

Most, if not all, *man-made* spatial objects such as buildings, roads, pipelines and even political divisions have a clear boundary and extension. The location of the Eiffel tower is well known and certain, the path of the Interamerican highway is well established and North Dakota has certainly defined boundaries and extension. These are in our words, *crisp* spatial objects. Current spatial database models and GIS successfully implement such object types but lack modeling and representation power when handling objects with not such crispness.

Spatial *vagueness* or *indeterminacy* is an inherent property of many objects that are handled in the spatial database context. Point locations may not be exactly known, paths or trails might fade and become uncertain at intervals. The boundary of regions might not be certainly known or simply not be as sharp as that of a building or a highway. Take as examples lakes (and rivers) whose extension (and path) depends on pluvial activity, or take the location of oil fields that in many cases can only be guessed. This inherent uncertainty brings to light the necessity of more comprehensive models that are able to cope with what we will refer to as *vague spatial objects*.

Correctly handling spatial data involves more than a good definition of the data types. It also involves defining a complete set of operations and predicates

* This work was partially supported by the National Science Foundation under grant number NSF-CAREER-IIS-0347574.

that make the data objects useful in their context. Topological predicates are proven to be very important in spatial data applications. In the case of crisp spatial objects, topological predicates are well studied and plenty of approaches exist for their proper definition. But this is not the case for vague spatial objects where a well defined set of topological predicates does not exist currently. The goal of this paper is to provide a comprehensive model for identifying topological predicates between vague spatial objects. The predicates to be identified have the added complexity of dealing with the vagueness present in the objects themselves, thus making the predicates vague in nature. The model we present here enhances the results of preliminary work that has been part of our own research as part of the *VASA* (Vague Spatial Algebra) project. The data types used are our own *vague spatial data types* [5] and include *vague points*, *vague lines* and, *vague regions*. A major benefit of *vague spatial objects* is that their definition is expressed in terms of crisp spatial operations so that they represent executable specifications. The same benefits are sought for topological predicates, and so we define them for vague spatial objects in terms of the already defined topological predicates for crisp spatial objects.

Section 2 presents related work. Section 3 introduces our vague spatial data types upon which the topological predicates are identified. Our enhanced general mechanism for identifying vague topological predicates is explained in Section 4 where we also draw a comparison to our previous approach. In Section 5 we introduce a case study based on the implementation of the identification mechanism on vague regions. Section 6 introduces the notions necessary to implement the newly identified predicates as part of common database query languages. Finally, Section 7 draws some conclusions and addresses future work.

2 Related Work

We refer to spatial vagueness as a natural feature of a spatial object. Vagueness defines object properties as uncertain or indeterminate such that it is not possible to assure whether certain components belong to the object or not. Three main alternatives have been proposed as general design methods. *Models based on fuzzy sets* (e.g., [15]) are all based on fuzzy set theory, allow a fine-grained modeling of vague spatial objects but are computationally rather expensive with respect to data structures and algorithms. *Models based on rough sets* (e.g., [16]) work with lower and upper approximations of spatial objects, which is similar to our approach. But the formal background is rather different. *Models based on exact spatial objects* (e.g., [9, 10]) extend data models, type systems, and concepts for crisp spatial objects to vague spatial objects. A discussion of the differences of these approaches can be found in [15]. Vague spatial data types [5, 7] leveraged in this paper belong to the latter category.

The basis of the latter category are crisp spatial data types (see [14] for a survey). We assume a very general definition of these data types and call them *complex*. Point objects are considered to be finite collection of points. Lines are assumed to be finite collections of disjoint curves which may meet in single

endpoints. Regions are finite collections of disjoint faces except for single common points, and faces may have disjoint holes except for single common points.

Much research on spatial databases has been devoted to topological predicates (like *overlap*, *meet*, or *disjoint*) on crisp spatial data types. The two main solution approaches employ either spatial logic [1] or point set theory and point set topology [12]. Our definition of vague topological predicates rests on a generalization [8, 13] of the latter approach (the well known *9-intersection model* [12]) in the sense that topological predicates are defined on crisp *complex* spatial objects and not on *simple* spatial objects as in the other approaches. The difference between considering simple and complex spatial objects is important, for example, the number of topological predicates for simple regions is 8, whereas the number for complex regions is 33. Topological predicates for simplified vague regions have already been studied in [9, 10]. These approaches, although already quite sophisticated, suffer from two main drawbacks. First, the crisp regions used are simple regions, i.e., they do not allow holes and only consist of a single component. Second, the vague regions defined are regions with “broad boundaries”. That is, one crisp simple region, which represents the area that definitely belongs to the vague region, is located in another larger crisp simple region. The geometric difference between the larger crisp region and the smaller one is considered to be the broad, vague boundary. Operations on such simple vague spatial objects suffer greatly from a lack of closure properties and expressiveness, which are part of our goals. The simple approach they follow is insufficient for our definition and goals, hence we look to define a general method with more expressive power and that is still usable. The works in [10] and [9] have identified 44 and 46 different topological relationships, respectively. The results from these two previous approaches are not applicable to our own vague spatial data types because they represent only the topological relationships between so-called *concentric* regions (i.e. the *kernel* is always surrounded by the *conjecture*) that belong to a special case of our vague regions under which not all vague region topological relationships are covered. The previous models are only defined for vague regions and not for vague points and vague lines.

3 Vague Spatial Data Types

To motivate our definition of vague spatial data types, we illustrate an ecological scenario that is to be considered for the development of a nature preservation program. The program developers need to consider (among many other data) the extension of lakes, the paths followed by rivers, and the refuges of animals as well as their roaming routes. We can notice how the lake and river data can be uncertain due to rain activity within a time period. Roaming routes of some species might be only approximate as these can change slightly or it is not possible to record the exact path for all cases. Animal refuges might be uncertain due to their underground or cave nature. All these are examples of what we refer to as *vague spatial objects*. The animal refuge locations are specifically modeled as a *vague point* object where the precisely known locations are called

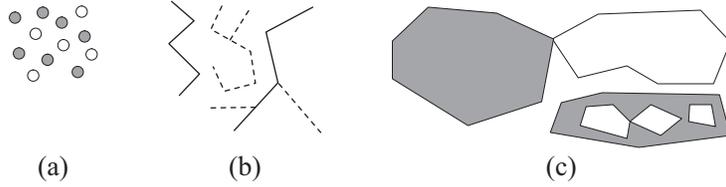


Fig. 1. Examples of a (complex) vague point object (a), a (complex) vague line object (b), and a (complex) vague region object (c). Each collection of components forms a single vague object.

the *kernel point* object and the assumed locations are denoted as the *conjecture point* object. The roaming routes and river paths can be modeled as *vague line* objects. Some routes, called *kernel line* objects, have been definitely identified and are certainly part of the river or route. Other routes can only be assumed and these are denoted as *conjecture line* objects. Knowledge about extension of lakes and other areas within the ecological system can be modeled similarly with *vague regions* formed by *kernel* and *conjecture* parts. Figure 1 gives some illustrations. Grey shaded areas, straight lines, and grey points indicate kernel parts; areas with white interiors, dashed lines, and white points refer to conjecture parts.

For the definition of vague points, vague lines, and vague regions we leverage the well known data types *point* for crisp points, *line* for crisp lines, and *region* for crisp regions [13]. These types are closed under the geometric set operations *union* ($\oplus : \alpha \times \alpha \rightarrow \alpha$), *intersection* ($\otimes : \alpha \times \alpha \rightarrow \alpha$), *difference* ($\ominus : \alpha \times \alpha \rightarrow \alpha$), and *complement* ($\sim : \alpha \rightarrow \alpha$). The use of an exact model for constructing vague spatial data types leads to the benefit that existing definitions, techniques, data structures and algorithms need not be redeveloped but can simply be used or in the worst case slightly modified or extended as necessary.

A vague spatial object is described by a pair of two crisp complex spatial objects. Hence, the same generic definition is applicable to all vague spatial data types. That is, the extension of a crisp spatial data type to a corresponding vague type is given by a type constructor v as follows:

$$v(\alpha) = \alpha \times \alpha \quad \forall \alpha \in \{point, line, region\}$$

This means that for $\alpha = region$ we obtain $v(region) = region \times region$, which we also name *vregion*. Accordingly, the data types *vline* and *vpoint* are defined. For a vague spatial object $w = (w_k, w_c) \in v(\alpha)$, the first crisp spatial object w_k , called the *kernel part*, describes the determinate component of w , that is, the component that definitely belongs to the vague object. The second crisp spatial object w_c , called the *conjecture part*, describes the vague component of w , that is, the component from which we cannot say with any certainty whether it or subparts of it belong to the vague object or not. *Maybe* the conjecture part or subparts of it belong to the vague object, *maybe* this is not the case. Since the kernel part and the conjecture part of the *same* vague spatial object may not share interior points, we define the following as a more general constraint from

the original defined in [5]:

$$\forall \alpha \in \{point, line, region\} \forall w = (w_k, w_c) \in v(\alpha) : w_k^\circ \cap w_c^\circ = \emptyset$$

More details, in particular about the semantics of vague spatial data types as well as the definition of vague spatial operations, can be found in [5].

4 General Mechanism for Vague Topological Predicates

The approach we present here is based on three main goals. The first goal is to develop a formalism that works independently of the data types to which it is applied. It is desired that the formalism can be applied to two vague points equally as it can be applied to two vague lines or to the combination of a vague line and a vague region. Second, we consider important to make use of existing definitions of topological predicates for crisp spatial objects. This goal is a direct result from the definition of vague spatial objects. As noted in Section 3, vague spatial objects are constructed from crisp spatial objects. It is only consistent to let vague topological predicates be constructed from existing crisp topological predicates (see Figure 2). The final goal is to benefit from implementation advantages in such a way that VASA as a whole can make use of a preexisting crisp spatial algebra implementation as a simple executable specification.

The general method we propose, characterizes vague topological predicates on the basis of conjunctions of crisp topological predicates. The crisp topological predicates used as the underlying model are those defined in [13]. For two vague spatial object A, B , we evaluate the conjunction of the crisp topological predicates in the relationships between $(A_k, B_k), (A_k \oplus A_c, B_k), (A_k, B_k \oplus B_c)$, and $(A_k \oplus A_c, B_k \oplus B_c)$. These relationships represent the *smallest* objects which certainly exist (A_k, B_k) , to the biggest possible objects represented with all uncertain features $(A_k \oplus A_c, B_k \oplus B_c)$. Given $\alpha, \beta \in \{point, line, region\}$, let $T_{\alpha, \beta}$ be the set of crisp topological predicates between the types α and β . To identify the vague topological predicates for type-combination $v(\alpha) \times v(\beta)$ we analyze all

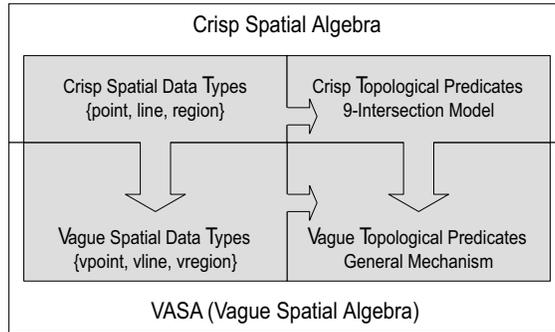


Fig. 2. Relations between crisp and vague spatial data models.

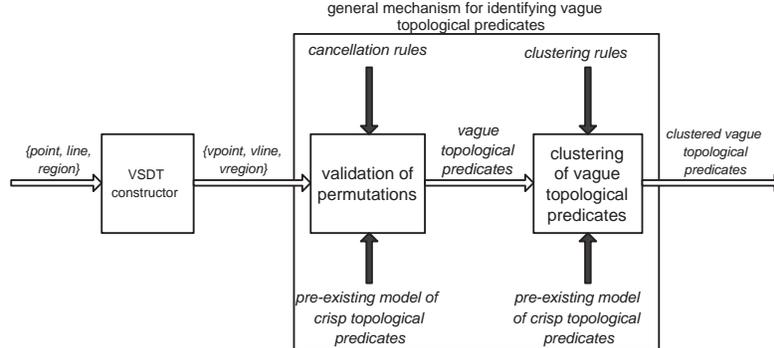


Fig. 3. General Mechanism for Identifying Vague Topological Predicates.

possible $|T_{\alpha,\beta}|^4$ combinations for the four relationships noted above. It is possible that not all combinations are valid due to contradictions between the relationships so we proceed to apply *cancellation rules* that validate each combination (see Figure 3). The cancellation rules are defined on the basis of the point set intersections that define the crisp topological relationships between each of the components involved. This means that we do not refer to the crisp topological predicate by name, but rather specify more general cancellation rules by simply analyzing their point set intersections. Once all invalid combinations have been eliminated, we can refer to the remaining combinations as the vague topological predicates for $v(\alpha) \times v(\beta)$.

It is likely that the identified vague topological predicates will constitute a large set that could prove difficult to handle for the user. To provide for an easier management of such large sets of predicates we implement a step of what we call *clustering*. At this step, we define *clustering rules* which are in charge of grouping single vague topological predicates (as identified in the previous step) into meaningful clusters. Due to the involvement of the *conjecture* in the definition of the topological relationship, it becomes insufficient for the predicates representing these relationships to simply result in either *true* or *false*. As previously proposed in [6], each cluster results in a new data type $vbool = \{true, false, maybe\}$ that extends the regular boolean type and allows handling the inherent uncertainty. This *three-valued* logic can be adapted to boolean logic through a simple conversion that extends the set of available predicates and is detailed in Section 6.

The method we just detailed, is similar to the preliminary approach we expose in [6]. Based on the experiences learned from that preliminary work, we identified two key features that make our current approach better. First, we only take into account four combinations instead of the nine used in the preliminary approach. We eliminated all relationships from the preliminary approach in which the *conjecture* of an object might be considered alone without the *kernel*. The reasoning behind this change rests on the fact that in no situation the *conjecture* of an object will be considered without considering the *kernel*. From the definition of vague spatial objects, the *kernel* is always part of the object and the

conjecture might not be part of it. This reasoning relates to the rough set idea of lower approximation ($A_k, A \in v(\alpha)$) and an upper approximation ($A_k \oplus A_c$), noticing that no consideration of ($A_k \oplus A_c - A_k = A_c$) is made. The second key feature improvement lies in the specification of cancellation and clustering rules. In the preliminary approach, we specified cancellation and clustering rules on the basis of the named crisp topological predicates. This was possible with a small set of easily identified predicates like those for complex points. When dealing with a large set of crisp topological predicates, such as the 33 predicates between complex regions and the 82 predicates between complex lines, it becomes unfeasible to understand the semantics of each unnamed crisp topological predicate and define the rules this way. This is why the rules are now specified on the basis of the point set intersections that represent the individual topological predicates. This removes the necessity of dealing with the large unnamed set, which reduces the probability for errors, and simplifies the set of rules.

We move on in the next section to show how the approach can be implemented. This is done by using a case study involving the identification of topological predicates between vague regions.

5 Topological Predicates between Vague Regions

In this section we use vague regions to illustrate the mechanism described in Section 4. The example implementation shown here identifies the topological predicates between two vague region objects. The underlying crisp topological predicates used are those defined in [13] for two complex crisp regions. A summary of such definition follows.

5.1 Topological Predicates between Complex Spatial Objects

Originally described in [4, 12], the 9-intersection model (4-intersection model previously), defines topological predicates between simple regions on the basis of the intersection between the parts (interior, boundary, exterior) of the regions involved. Later the model is extended in [3] to account for simple regions with holes and in [11] to work with regions made up of multiple components. Finally, in [13], a comprehensive definition of topological predicates for complex regions is proposed. Complex regions can contain both holes and multiple components. The proposed method works by simply applying the 9-intersection model to the point sets belonging to the complex regions.

The topological predicate definition from [13] initially analyzes all possible 3×3 matrices (for a total of 512 matrices). Each matrix entry contains either a 1 or a 0 that represent whether that intersection is *non-empty* or *empty* respectively. A type-combination dependent set of *constraint rules* is applied to the original set of 512 matrices. The constraint rules are in charge of eliminating all non-constraint satisfying matrices that represent invalid scenes. Once all invalid matrices are eliminated, the remaining ones are considered the topological predicates between objects of the type-combination in question. For the purpose of

this case study it is only necessary to consider complex regions, but the reference identifies topological predicates between all type combinations of complex points, complex lines and complex regions.

In the case of complex regions, a total of nine constraint rules result in 33 possible topological predicates between two complex regions. Such a large number of predicates presents problems of manageability for the user. This is the reason for the *clustering* of topological predicates presented in [13]. So-called *topological cluster predicates* are defined by means of clustering rules that are some kind of relaxed constraint rules. The clustering rules define a cluster by taking into account not all the nine intersections in the matrix causing a slight generalization that results in possibly more than one of the original predicates to form part of a single cluster. The authors define eight clustered predicates with semantics similar to the original topological predicates for simple regions identified by the 9-intersection model.

5.2 Cancellation Rules

Now we proceed to identify the topological predicates between vague regions. The first step is the definition of the cancellation rules that eliminate all contradictory information within the combinations explored as part of the general mechanism detailed in Section 4. The underlying set of topological predicates used is, as mentioned before, as shown in [13] and identifies 33 topological predicates between complex regions. We originally deal with a total of $33^4 = 1185921$ combinations. To make the formal rules more compact, we make use of variables $w \in \{B_k, B_k \oplus B_c\}$, $v \in \{A_k, A_k \oplus A_c\}$.

Lemma 1. *Any part (interior, boundary, exterior) of a single component (i.e. kernel or conjecture) or the union of components of the first object that intersects the interior of at least one component from the second object, must also intersect the interior of the union of components from the second object, i.e.,*

Lemma 1.1 $\forall p, q \in T_{\alpha, \beta} :$
 $\neg(p(A_k, w) \wedge q(A_k \oplus A_c, w))$ s.t.
 $\exists r \in \{\circ, \partial, -\} :$
 $(p(A_k, w) \Rightarrow A_k^\circ \cap w^r \neq \emptyset \wedge$
 $q(A_k \oplus A_c, w) \Rightarrow$
 $(A_k \oplus A_c)^\circ \cap w^r = \emptyset)$

Lemma 1.2 $\forall p, q \in T_{\alpha, \beta} :$
 $\neg(p(v, B_k) \wedge q(v, B_k \oplus B_c))$ s.t.
 $\exists r \in \{\circ, \partial, -\} :$
 $(p(v, B_k) \Rightarrow B_k^\circ \cap v^r \neq \emptyset \wedge$
 $q(v, B_k \oplus B_c) \Rightarrow$
 $(B_k \oplus B_c)^\circ \cap v^r = \emptyset)$

Proof. We know that for any vague region C , $C_k^\circ \subseteq (C_k \oplus C_c)^\circ$. From this fact, we can derive that for any point set x , it is always true that $C_k^\circ \cap x \neq \emptyset \Rightarrow (C_k \oplus C_c)^\circ \cap x \neq \emptyset$. Thus, Lemma 1 takes care of those combinations in which this implication is contradicted. \square

Lemma 2. *Any part of a single component or the union of components of the first object that does not intersect the exterior of at least one component from the second object must also not intersect the exterior of the union of components from the second object, i.e.,*

Lemma 2.1 $\forall p, q \in T_{\alpha, \beta} :$
 $\neg(p(A_k, w) \wedge q(A_k \oplus A_c, w))$ s.t.
 $\exists r \in \{\circ, \partial, -\} :$
 $(p(A_k, w) \Rightarrow A_k^- \cap w^r = \emptyset \wedge$
 $q(A_k \oplus A_c, w) \Rightarrow$
 $(A_k \oplus A_c)^- \cap w^r \neq \emptyset)$

Lemma 2.2 $\forall p, q \in T_{\alpha, \beta} :$
 $\neg(p(v, B_k) \wedge q(v, B_k \oplus B_c))$ s.t.
 $\exists r \in \{\circ, \partial, -\} :$
 $(p(v, B_k) \Rightarrow B_k^- \cap v^r = \emptyset \wedge$
 $q(v, B_k \oplus B_c) \Rightarrow$
 $(B_k \oplus B_c)^- \cap v^r \neq \emptyset)$

Proof. We know that for any vague region C , $C_k^- \supseteq (C_k \oplus C_c)^-$. From this fact, we can derive that for any point set x , it is always true that $C_k^- \cap x = \emptyset \Rightarrow (C_k \oplus C_c)^- \cap x = \emptyset$. Thus, Lemma 2 takes care of those combinations in which this implication is contradicted. \square

Lemma 3. *A single component or the union of components of the first object that is not disjoint from at least one component from the second object, must also not be disjoint from the union of both components from the second object, i.e.,*

Lemma 3.1 $\forall p, q \in T_{\alpha, \beta} :$
 $\neg(p(A_k, w) \wedge q(A_k \oplus A_c, w))$ s.t.
 $\exists r, s \in \{\circ, \partial\} : (p(A_k, w) \Rightarrow$
 $A_k^r \cap w^s \neq \emptyset) \wedge$
 $\forall t, u \in \{\circ, \partial\} :$
 $(q(A_k \oplus A_c, w) \Rightarrow$
 $(A_k \oplus A_c)^t \cap w^u = \emptyset))$

Lemma 3.2 $\forall p, q \in T_{\alpha, \beta} :$
 $\neg(p(v, B_k) \wedge q(v, B_k \oplus B_c))$ s.t.
 $\exists r, s \in \{\circ, \partial\} : (p(v, B_k) \Rightarrow$
 $B_k^r \cap v^s \neq \emptyset) \wedge$
 $\forall t, u \in \{\circ, \partial\} :$
 $(q(v, B_k \oplus B_c) \Rightarrow$
 $(B_k \oplus B_c)^t \cap v^u = \emptyset))$

Proof. Due to the nature of the \oplus (geometric union) operation, any intersection of the interiors of the complex regions a and b will remain untouched, and any intersection between boundaries or between an interior and a boundary will either remain untouched or be replaced by an interior-interior intersection when the union operation is applied and instead the intersections between $a \oplus d, b$ are analyzed. This means that there is no possibility for any such intersection to disappear or be replaced by an intersection with an exterior that could result in disjointment of the objects involved. Thus, we can imply that $\neg(\text{disjoint}(a, b)) \Rightarrow \neg(\text{disjoint}((a \oplus d), b))$ where *disjoint* refers to the clustered predicate as defined in [13]. \square

Lemma 4. *If we assume that some τ representing a component or the union of components of the first object is not contained by a single component of the second object but is contained by the union of components of the second object, then τ must not contain the interior of the union of the components from the second object, i.e.,*

Lemma 4.1 $\forall p, q \in T_{\alpha, \beta} :$
 $\neg(p(A_k, w) \wedge q(A_k \oplus A_c, w))$ s.t.
 $(p(A_k, w) \Rightarrow \partial A_k \cap \partial w \neq \emptyset) \wedge$
 $(q(A_k \oplus A_c, w) \Rightarrow ((A_k \oplus A_c)^\circ \subseteq w^\circ)$
 $\wedge (\partial(A_k \oplus A_c) \cap \partial w = \emptyset))$

Lemma 4.2 $\forall p, q \in T_{\alpha, \beta} :$
 $\neg(p(v, B_k) \wedge q(v, B_k \oplus B_c))$ s.t.
 $(p(v, B_k) \Rightarrow \partial B_k \cap \partial v \neq \emptyset) \wedge$
 $(q(v, B_k \oplus B_c) \Rightarrow ((B_k \oplus B_c)^\circ \subseteq v^\circ)$
 $\wedge (\partial(B_k \oplus B_c) \cap \partial v = \emptyset))$

Proof. We show the first sublemma; the proof for the second sublemma is similar. Given $(A_k \oplus A_c)^\circ \subseteq w^\circ \Rightarrow A_k^\circ \subseteq w^\circ$, then the rule represents a contradiction because, for it to be true that $\partial A_k \cap \partial w \neq \emptyset \wedge \partial(A_k \oplus A_c) \cap \partial w = \emptyset$, A_c must share the same boundary with A_k as A_k shares with w , i.e. ($\partial A_k \cap \partial A_c = \partial A_k \cap \partial w$). This means that A_c makes A_k grow towards the exterior of w which in turn makes it impossible for $(A_k \oplus A_c)^\circ \subseteq w^\circ$ to hold. \square

Lemma 5. *If some τ which represents a single component or the union of components from the first object, is completely contained within the interior of a single component from the second object, then the boundary of τ must not intersect the boundary of the union of components from the second object, i.e.,*

$$\begin{array}{ll}
\textbf{Lemma 5.1} \forall p, q \in T_{\alpha, \beta} : & \textbf{Lemma 5.2} \forall p, q \in T_{\alpha, \beta} : \\
\neg(p(A_k, w) \wedge q(A_k \oplus A_c, w)) \text{ s.t.} & \neg(p(v, B_k) \wedge q(v, B_k \oplus B_c)) \text{ s.t.} \\
(p(A_k, w) \Rightarrow \partial A_k \cap \partial w = \emptyset \wedge & (p(v, B_k) \Rightarrow \partial B_k \cap \partial v = \emptyset) \wedge \\
(A_k \oplus A_c)^\circ \supseteq w^\circ) \wedge & (B_k \oplus B_c)^\circ \supseteq v^\circ \wedge \\
(q(A_k \oplus A_c, w) \Rightarrow \partial(A_k \oplus A_c) \cap \partial w \neq \emptyset)) & (q(v, B_k \oplus B_c) \Rightarrow (\partial(B_k \oplus B_c) \cap \partial v \neq \emptyset))
\end{array}$$

Proof. This case represents a similar situation to that in Lemma 4. The difference here is that the boundaries of the subsets do not intersect but, when the conjecture is added, the boundaries intersect. Such a situation is impossible when the first object is completely contained in the second that is being expanded by the conjecture. The reason is that the conjecture would expand the object towards the inside of its own region's kernel which is in direct contradiction with the definition of vague regions. \square

Lemma 6. *If it can be inferred that the conjecture of the first object is empty (\perp), then it must be true that the predicates defined by the relationships between the kernel of the first object and any component or union of the components of the second object, and between the union of the components of the first object and the same component or whole of the second object, are the same, i.e.,*

$$\begin{array}{ll}
\textbf{Lemma 6.1} \forall p, q, r, s \in T_{\alpha, \beta} : & \textbf{Lemma 6.2} \forall p, q, r, s \in T_{\alpha, \beta} : \\
\neg(p(A_k, B_k) \wedge q(A_k \oplus A_c, B_k) \wedge & \neg(p(A_k, B_k) \wedge q(A_k \oplus A_c, B_k) \wedge \\
r(A_k, B_k \oplus B_c) \wedge & r(A_k, B_k \oplus B_c) \wedge \\
s(A_k \oplus A_c, B_k \oplus B_c)) \text{ s.t.} & s(A_k \oplus A_c, B_k \oplus B_c)) \text{ s.t.} \\
(p(A_k, B_k) \wedge q(A_k \oplus A_c, B_k) \Rightarrow & (p(A_k, B_k) \wedge r(A_k, B_k \oplus B_c) \Rightarrow \\
A_c = \perp \wedge & B_c = \perp \wedge \\
(r(A_k, B_k \oplus B_c) \neq s(A_k \oplus A_c, B_k \oplus B_c)) & (q(A_k \oplus A_c, B_k) \neq s(A_k \oplus A_c, B_k \oplus B_c))
\end{array}$$

Proof. If a conjecture is known to be empty, then it does not add any features to the kernel of the object, in other words ($A_k \oplus A_c = A_k$) for some vague region A . Thus, any crisp topological predicates p and q of the whole object and the kernel by itself, respectively, with some other region w must be the same ($p(A_k \oplus A_c, w) = q(A_k, w)$). \square

After applying all rules to the over 1.1 million original combinations, only 69682 remain valid. Such a large number is difficult to manage at any level of usability, thus we present clustering rules to reduce the predicates to a workable

set. We consider important to note that we have not provided a formal proof that all the remaining combinations are valid. It is our opinion that this is a weakness of the case study presented here but does not take any merit away from the general mechanism for identifying vague topological predicates.

5.3 Clustering Rules

Being able to identify topological predicates by name is necessary for the user. The original names of the eight topological predicates between simple regions (*disjoint*, *meet*, *inside*, *contains*, *coveredBy*, *covers*, *equal* and *overlap*) as detailed in [12] seem to be appropriate names thus we name the clustered vague topological predicates alike. We simply capitalize their names to differentiate them from the original. The clusters presented here represent only one of many ways in which the clustering of the predicates can be performed. We attempt to define each cluster in a way so that it has similar semantics as those that identified the topological predicates between simple regions in the 9-intersection model. It is important that the clusters are mutually exclusive in terms of the *true* results. This means that, after the cancellation rules are applied, each of the resulting combinations should result in *true* for one and only one of the following clusters.

Disjoint Two vague regions as truly disjoint if none of their components have intersections of interior or boundaries between each other. The vague regions are truly not disjoint if the interiors or boundaries of their kernel parts intersect. Any other situation leaves the topological relationship uncertain; thus the predicate result is *maybe*. Formally:

$$\begin{aligned}
- \text{Disjoint}(A, B) = \text{true} &\Leftrightarrow ((A_k \oplus A_c)^\circ \cap (B_k \oplus B_c)^\circ = \emptyset) \wedge ((A_k \oplus A_c)^\circ \cap \partial(B_k \oplus B_c) = \emptyset) \wedge \\
&\quad (\partial(A_k \oplus A_c) \cap (B_k \oplus B_c)^\circ = \emptyset) \wedge (\partial(A_k \oplus A_c) \cap \partial(B_k \oplus B_c) = \emptyset) \\
- \text{Disjoint}(A, B) = \text{false} &\Leftrightarrow (A_k^\circ \cap B_k^\circ \neq \emptyset) \vee (A_k^\circ \cap \partial B_k \neq \emptyset) \vee \\
&\quad (\partial A_k \cap B_k^\circ \neq \emptyset) \vee (\partial A_k \cap \partial B_k \neq \emptyset) \\
- \text{Disjoint}(A, B) = \text{maybe} &\Leftrightarrow \neg(\text{Disjoint}(A, B) = \text{true} \vee \text{Disjoint}(A, B) = \text{false})
\end{aligned}$$

Meet Two vague regions certainly meet when the boundaries of their kernels intersect but the interiors of all components do not intersect. They certainly do not meet when the interiors of their kernels intersect or when they are certainly *Disjoint*. Formally:

$$\begin{aligned}
- \text{Meet}(A, B) = \text{true} &\Leftrightarrow (\partial A_k \cap \partial B_k \neq \emptyset) \wedge ((A_k \oplus A_c)^\circ \cap (B_k \oplus B_c)^\circ = \emptyset) \\
- \text{Meet}(A, B) = \text{false} &\Leftrightarrow (A_k^\circ \cap B_k^\circ \neq \emptyset) \vee \\
&\quad (((A_k \oplus A_c)^\circ \cap (B_k \oplus B_c)^\circ = \emptyset) \wedge ((A_k \oplus A_c)^\circ \cap \partial(B_k \oplus B_c) = \emptyset) \wedge \\
&\quad (\partial(A_k \oplus A_c) \cap (B_k \oplus B_c)^\circ = \emptyset) \wedge (\partial(A_k \oplus A_c) \cap \partial(B_k \oplus B_c) = \emptyset)) \\
- \text{Meet}(A, B) = \text{maybe} &\Leftrightarrow \neg(\text{Meet}(A, B) = \text{true} \vee \text{Meet}(A, B) = \text{false})
\end{aligned}$$

Inside A vague region is certainly inside another one if all parts from all components of the first vague region are inside the kernel component of the second vague region. On the other hand, a vague region is certainly not inside another one when the interior of its kernel intersects the exterior of the second region or their boundaries intersect. Formally:

- $Inside(A, B) = true \Leftrightarrow ((A_k \oplus A_c)^\circ \cap B_k^\circ \neq \emptyset) \wedge (\partial(A_k \oplus A_c) \cap B_k^\circ \neq \emptyset) \wedge ((A_k \oplus A_c)^\circ \cap B_k^- = \emptyset) \wedge$
 $(\partial(A_k \oplus A_c) \cap B_k^- = \emptyset) \wedge ((A_k \oplus A_c)^\circ \cap \partial B_k = \emptyset) \wedge (\partial(A_k \oplus A_c) \cap \partial B_k = \emptyset)$
- $Inside(A, B) = false \Leftrightarrow (A_k^\circ \cap (B_k \oplus B_c)^- \neq \emptyset) \vee (\partial A_k \cap \partial(B_k \oplus B_c) \neq \emptyset)$
- $Inside(A, B) = maybe \Leftrightarrow \neg(Inside(A, B) = true \vee Inside(A, B) = false)$

Contains The result for the *Contains* cluster is symmetric to *Inside*. Formally, $Contains(A, B) \Leftrightarrow Inside(B, A)$.

CoveredBy A vague region is certainly covered by another one if all parts from all components of the first vague region are inside the kernel component of the second vague region and the boundary of the kernel of the first region intersects the boundary of the kernel of the second region. On the other hand, a vague region is certainly not covered by another one if the interior of its kernel intersects the exterior of the second region or their boundaries do not intersect. Formally:

- $CoveredBy(A, B) = true \Leftrightarrow ((A_k \oplus A_c)^\circ \cap B_k^\circ \neq \emptyset) \wedge$
 $(\partial(A_k \oplus A_c) \cap B_k^\circ \neq \emptyset) \wedge ((A_k \oplus A_c)^\circ \cap B_k^- = \emptyset) \wedge$
 $(\partial(A_k \oplus A_c) \cap B_k^- = \emptyset) \wedge ((A_k \oplus A_c)^\circ \cap \partial B_k = \emptyset) \wedge (\partial(A_k \oplus A_c) \cap \partial B_k \neq \emptyset)$
 $\wedge (\partial(A_k \oplus A_c) \cap \partial(B_k \oplus B_c) \neq \emptyset) \wedge (\partial A_k \cap \partial B_k \neq \emptyset) \wedge (\partial A_k \cap \partial(B_k \oplus B_c) \neq \emptyset)$
- $CoveredBy(A, B) = false \Leftrightarrow (A_k^\circ \cap (B_k \oplus B_c)^- \neq \emptyset) \vee (Inside(A, B) = true)$
- $CoveredBy(A, B) = maybe \Leftrightarrow \neg(CoveredBy(A, B) = true \vee CoveredBy(A, B) = false)$

Covers The result of *Covers* cluster is symmetric to *CoveredBy*. Formally, $Covers(A, B) \Leftrightarrow CoveredBy(B, A)$.

Equal The only way two vague regions are certainly equal is if their kernels are equal and their conjectures are empty. They are definitely not equal when the interior of one kernel touches the exterior of the other region or if one region is contained inside the kernel of another. Formally:

- $Equal(A, B) = true \Leftrightarrow (A_k^\circ \cap \partial B_k = \emptyset) \wedge (A_k^\circ \cap B_k^- = \emptyset) \wedge$
 $(\partial A_k \cap B_k^\circ = \emptyset) \wedge (\partial A_k \cap B_k^- = \emptyset) \wedge$
 $(A_k^- \cap \partial B_k = \emptyset) \wedge (A_k^- \cap B_k^\circ = \emptyset) \wedge$
 $(A_k^\circ \cap \partial(B_k \oplus B_c) = \emptyset) \wedge (A_k^\circ \cap (B_k \oplus B_c)^- = \emptyset) \wedge$
 $(\partial A_k \cap (B_k \oplus B_c)^\circ = \emptyset) \wedge (\partial A_k \cap (B_k \oplus B_c)^- = \emptyset) \wedge$
 $(A_k^- \cap \partial(B_k \oplus B_c) = \emptyset) \wedge (A_k^- \cap (B_k \oplus B_c)^\circ = \emptyset) \wedge$
 $((A_k \oplus A_c)^\circ \cap \partial B_k = \emptyset) \wedge ((A_k \oplus A_c)^\circ \cap B_k^- = \emptyset) \wedge$
 $(\partial(A_k \oplus A_c) \cap B_k^\circ = \emptyset) \wedge (\partial(A_k \oplus A_c) \cap B_k^- = \emptyset) \wedge$

$$\begin{aligned}
& ((A_k \oplus A_c)^- \cap \partial B_k = \emptyset) \wedge ((A_k \oplus A_c)^- \cap B_k^\circ = \emptyset) \wedge \\
& ((A_k \oplus A_c)^\circ \cap \partial(B_k \oplus B_c) = \emptyset) \wedge ((A_k \oplus A_c)^\circ \cap (B_k \oplus B_c)^- = \emptyset) \wedge \\
& (\partial(A_k \oplus A_c) \cap (B_k \oplus B_c)^\circ = \emptyset) \wedge (\partial(A_k \oplus A_c) \cap (B_k \oplus B_c)^- = \emptyset) \wedge \\
& ((A_k \oplus A_c)^- \cap \partial(B_k \oplus B_c) = \emptyset) \wedge ((A_k \oplus A_c)^- \cap \int(B_k \oplus B_c) = \emptyset) \\
- \text{ } & \text{Equal}(A, B) = \text{false} \Leftrightarrow (A_k^\circ \cap (B_k \oplus B_c)^- \neq \emptyset) \vee ((A_k \oplus A_c)^- \cap B_k^\circ \neq \emptyset) \vee \\
& (((A_k \oplus A_c)^\circ \cap B_k^\circ \neq \emptyset) \wedge (\partial(A_k \oplus A_c) \cap B_k^\circ \neq \emptyset) \wedge ((A_k \oplus A_c)^\circ \cap B_k^- = \emptyset) \wedge \\
& (\partial(A_k \oplus A_c) \cap B_k^- = \emptyset) \wedge ((A_k \oplus A_c)^\circ \cap \partial B_k = \emptyset)) \vee \\
& ((A_k^\circ \cap (B_k \oplus B_c)^\circ \neq \emptyset) \wedge (A_k^\circ \cap \partial(B_k \oplus B_c) \neq \emptyset) \wedge (A_k^- \cap (B_k \oplus B_c)^\circ = \emptyset) \wedge \\
& (A_k^- \cap \partial(B_k \oplus B_c) = \emptyset) \wedge (\partial A_k \cap (B_k \oplus B_c)^\circ = \emptyset)) \\
- \text{ } & \text{Equal}(A, B) = \text{maybe} \Leftrightarrow \neg(\text{Equal}(A, B) = \text{true} \vee \text{Equal}(A, B) = \text{false})
\end{aligned}$$

Overlap Two vague regions surely overlap if their kernel interiors intersect each other and also intersect their whole exteriors. We can certainly say the vague regions do not overlap if any of the other 7 clusters holds *true*, which leaves a large number of possibilities for the regions to maybe overlap. Formally:

$$\begin{aligned}
- \text{ } & \text{Overlap}(A, B) = \text{true} \Leftrightarrow (A_k^\circ \cap B_k^\circ \neq \emptyset) \wedge (A_k^\circ \cap B_k \oplus B_c^- \neq \emptyset) \wedge (A_k \oplus \\
& A_c^- \cap B_k^\circ \neq \emptyset) \\
- \text{ } & \text{Overlap}(A, B) = \text{false} \Leftrightarrow (\text{Disjoint}(A, B) = \text{true}) \vee (\text{Meet}(A, B) = \text{true}) \vee \\
& (\text{Inside}(A, B) = \text{true}) \vee \\
& (\text{Contains}(A, B) = \text{true}) \vee (\text{CoveredBy}(A, B) = \text{true}) \vee (\text{Covers}(A, B) = \\
& \text{true}) \vee \\
& (\text{Equal}(A, B) = \text{true}) \\
- \text{ } & \text{Overlap}(A, B) = \text{maybe} \Leftrightarrow \neg(\text{Overlap}(A, B) = \text{true} \vee \text{Overlap}(A, B) = \\
& \text{false})
\end{aligned}$$

In the next section, we show by using examples, how these clusters can be used in common database queries.

6 Querying

Popular database query language such as SQL understand predicates as boolean expressions. This means that the three values resulting from the clusters must be translated to boolean logic in order to use the vague topological predicates in SQL-like query languages. The translation is done by the following general definition for any clustered predicate P :

$$\begin{aligned}
\text{True_}P(A, B) = \text{true} & \Rightarrow P(A, B) = \text{true} \\
\text{True_}P(A, B) = \text{false} & \Rightarrow P(A, B) = \text{maybe} \vee P(A, B) = \text{false} \\
\text{Maybe_}P(A, B) = \text{true} & \Rightarrow P(A, B) = \text{maybe} \\
\text{Maybe_}P(A, B) = \text{false} & \Rightarrow P(A, B) = \text{true} \vee P(A, B) = \text{false} \\
\text{False_}P(A, B) = \text{true} & \Rightarrow P(A, B) = \text{false} \\
\text{False_}P(A, B) = \text{false} & \Rightarrow P(A, B) = \text{true} \vee P(A, B) = \text{maybe}
\end{aligned}$$

For the sample queries, we assume a scenario (similar to that in Section 3) of an ecological application. We define roaming areas for species as vague regions

with sections that definitely delimit the area in which some species lives. The representation also includes some areas for which we are not sure whether or not the species roam around.

In the first query, we are interested in the roaming areas of all groups of an imaginary endangered species *pastuzo* whose main predator is the tiger. We need to know of all roaming areas for pastuzos, that *might* be contained inside roaming areas of tigers. This would help establish whether or not the majority of pastuzos left are in danger of being eaten by tigers.

```
SELECT p.id
FROM   groups p, groups t
WHERE  p.species = "pastuzo"
and    t.species = "tiger"
and    Maybe_Inside(p.area,t.area);
```

In the next query, we want to establish all pastuzo groups that *certainly* live in areas outside those of the tigers. The purpose of this is to allocate resources and provide reproductive help to these pastuzo groups.

```
SELECT p.id
FROM   groups p, groups t
WHERE  p.species = "pastuzo"
and    t.species = "tiger"
and    True_Disjoint(p.area,t.area);
```

7 Conclusions and Future Work

We provide a mechanism that is able to identify topological predicates between vague spatial objects. We believe the strength of the mechanism stems from having accomplished all three goals imposed on its design. First, the mechanism described is type-independent. That is, given as input a set of cancellation rules and a set of clustering rules the mechanism identifies all vague topological predicates for the respective vague spatial data type combination. Second, the mechanism makes use of existing definitions for crisp topological predicates by defining vague topological predicates on their basis. Third, the definition of vague spatial data types, operations and predicates can be regarded as an executable specification based on crisp concepts. The accomplishment of these goals is improved by the lessons learned from a preliminary approach resulting in a mechanism that is both powerful and simple. We also showed how using the vague topological predicates in common query languages requires only a simple conversion that allows for common sense handling of uncertainty in spatial data through querying.

The obvious step to follow now is the identification of topological predicates between all type combinations of vague spatial data types. Also we consider important to finalize this approach by a full implementation of the concepts that are part of *VASA*.

References

1. Z. Cui and A. G. Cohn and D. A. Randell. Qualitative and Topological Relationships. In *3rd Int. Symp. on Advances in Spatial Databases*, LNCS 692, pages 296–315, 1993.
2. P. A. Burrough and A. U. Frank, editor. *Geographic Objects with Indeterminate Boundaries*. GISDATA Series, vol. 2. Taylor & Francis, 1996.
3. M.J. Egenhofer and E. Clementini and P. Di Felice. Topological Relations between Regions with Holes. *Int. Journal of Geographical Information Systems*, 8(2):128–142, 1994.
4. M. J. Egenhofer and J. Herring. Categorizing Binary Topological Relations Between Regions, Lines, and Points in Geographic Databases. Technical Report 90-12, National Center for Geographic Information and Analysis, University of California, Santa Barbara, 1990.
5. A. Pauly and M. Schneider. Vague Spatial Data Types, Set Operations, and Predicates. In *8th East-European Conf. on Advances in Databases and Information Systems*, pages 379–392, 2004.
6. A. Pauly and M. Schneider. Identifying Topological Predicates for Vague Spatial Objects. In *20th ACM Symp. for Applied Computing*, pages 587–591, 2005.
7. M. Erwig and M. Schneider. Vague Regions. In *5th Int. Symp. on Advances in Spatial Databases*, LNCS 1262, pages 298–320. Springer-Verlag, 1997.
8. T. Behr and M. Schneider. Topological Relationships of Complex Points and Complex Regions. In *Int. Conf. on Conceptual Modeling*, pages 56–69, 2001.
9. A. G. Cohn and N. M. Gotts. *The ‘Egg-Yolk’ Representation of Regions with Indeterminate Boundaries*, pages 171–187. In and A. U. Frank [2], 1996.
10. E. Clementini and P. Di Felice. *An Algebraic Model for Spatial Objects with Indeterminate Boundaries*, pages 153–169. In and A. U. Frank [2], 1996.
11. E. Clementini and P. Di Felice and G. Califano. Composite Regions in Topological Queries. *Information Systems*, 20(7):579–594, 1995.
12. M. J. Egenhofer and R. D. Franzosa. Point-Set Topological Spatial Relations. *Int. Journal of Geographical Information Systems*, 5(2):161–174, 1991.
13. M. Schneider and T. Behr. Topological Relationships between Complex Spatial Objects. Technical Report, CISE Dept., University of Florida, 2004.
14. M. Schneider. *Spatial Data Types for Database Systems - Finite Resolution Geometry for Geographic Information Systems*, volume LNCS 1288. Springer-Verlag, Berlin Heidelberg, 1997.
15. M. Schneider. Uncertainty Management for Spatial Data in Databases: Fuzzy Spatial Data Types. In *6th Int. Symp. on Advances in Spatial Databases*, LNCS 1651, pages 330–351. Springer-Verlag, 1999.
16. M. Worboys. Computation with Imprecise Geospatial Data. *Computational, Environmental and Urban Systems*, 22(2):85–106, 1998.