Topological Reasoning Between Complex Regions in Databases with Frequent Updates

Arif Khan & Markus Schneider*
Department of Computer and Information Science and Engineering
University of Florida
Gainesville, FL 32611, USA
{ahkhan, mschneid}@cise.ufl.edu

ABSTRACT

Reasoning about space has been a considerable field of study both in Artificial Intelligence and in spatial information theory. Many applications benefit from the inference of new knowledge about the spatial relationships between spatial objects on the basis of already available and explicit spatial relationship knowledge that we call *spatial* (*relationship*) *facts*. Hence, the task is to derive new spatial facts from known spatial facts. A considerable amount of work has focused on reasoning about *topological relationships* (as a special and important subset of spatial relationships) between *simple* spatial objects like *simple regions*. There is a common consensus in the GIS and spatial database communities that simple regions are insufficient to model spatial reality and that *complex* region objects are needed that allow multiple components and holes. Models for topological relationships between complex regions have already been developed. Hence, as the next logical step, the goal of this paper is to develop a reasoning model for them. Further, no reasoning model considers changes of the spatial fact basis stored in a database between consecutive queries. We show that conventional modeling suffers from performance degradation when the database is frequently changing. Our model does not assume any geometric representation model or data structure for the regions. The model is also backward compatible, i.e., it is also applicable to simple regions.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Spatial databases and GIS; H.2.3 [**Database Management**]: Query languages

General Terms

Design, Languages

*This work was partially supported by the National Science Foundation under grant numbers NSF-CAREER-IIS-0347574 and NSF-IIS-0812194.

ACM GIS '10, November 2-5, 2010. San Jose, CA, USA

Keywords

Topological relationship, complex region, spatial database, GIS

1. INTRODUCTION

Understanding the topological relationships between objects in space has become a multidisciplinary research issue involving AI, CAD/CAM systems, cognitive science, computer vision, image databases, linguistics, robotics, GIS, and spatial databases. From a spatial database and GIS point of view, topological relationships are necessary as filter conditions for spatial selections and spatial joins as well as for spatial data retrieval and analysis. In spatial databases and GIS, we generally deal with a large number of spatial objects. Hence, it is not uncommon that we do not have all possible relationships available between every pair of spatial objects all the time. This situation can arise either due to a lack of information or since it is impossible to get all the relationships. To deal with this problem of a lack of complete knowledge, we need a process through which we can infer the topological relationship between two spatial objects where the relationship does not currently exist in the knowledge base. This process is called reasoning. Hence, reasoning about topological relationship is a method of inferring new topological relationships, called *spatial facts*, between two spatial objects using the other existing spatial facts in the knowledge base. For example, given three objects A, B and C, and given two topological relationships $R_x(A, B)$ and $R_y(B, C)$, reasoning helps us find out the relationship R_z between A and C where R_z does not exist in the knowledge base. This process is called the *composition of relationships* and is the most common method of reasoning.

So far, the main focus of the available reasoning models is to deal with simple regions. But in the real world we often face the situation where real objects cannot be represented by simple regions alone. For example, Italy contains the Vatican as a hole, and the Galapagos islands do not consist of a single island but rather of a collection of many islands. These spatial phenomena cannot be represented by simple regions. The second problem is that the current reasoning models hardly take the changes of spatial facts into account. It is natural that often the information is added, deleted or updated in the databases. Hence, it is important to understand as well as to consider the effect of such changes while designing a reasoning model.

The main goal of this paper is to develop a reasoning model for complex regions. The main challenge is to deal with a large number of possible topological relationships between

two complex regions as well as to deal with a large number of such regions. Our second goal is to derive a set of inference rules by which the inference of relationships is performed. Since the type for simple regions is a subset of the type for complex regions, it is also our goal that the reasoning model is able to handle simple regions without requiring any modification. Finally, we show the effect of the changes of spatial facts on the reasoning process, and we propose an algorithm to handle those changes.

We propose a generalized process to infer new relationships between complex regions which is not restricted by the number of regions as well as changes in the database. The process has two basic steps. In the first step, we perform a reasoning process involving three regions and call it *local inference*. In the second step, we extend this local inference to N regions and hence call it *global inference*.

The remainder of the paper is organized as follows: Section 2 discusses related work regarding reasoning models for simple objects and the topological relationships between complex objects. Section 3 gives a more detailed view of the reasoning process. In Sections 4 and 5, we describe the local and global inference respectively. Section 6 integrates the two steps and gives an algorithm for the overall reasoning process. Section 7 evaluates the performance of the algorithm. Finally, Section 8 draws some conclusions and discusses future work.

2. RELATED WORK

In the past, numerous data models have been proposed with the aim of representing spatial objects in databases and GIS. Spatial objects embedded in the 2D space can be either point objects, line objects, or region objects. In this document we mainly consider complex region objects. Region objects are two-dimensional spatial objects with an extent (i.e., both height and width). Each kind of spatial object can be categorized as either a simple spatial object [13] or a complex spatial object [15, 16]. A simple region is topologically equivalent to a closed disc; it does not have holes. However, a complex region (Figure 1a) may have multiple components, called *faces*, and may have multiple holes. One important aspect is that for the reasoning process the spatial objects are only needed as symbolic terms; their geometries are not required. Spatial relationships are subdivided into directional relationships, topological relationships, and distance relationships. Our focus is on topological relationships which characterize the relative position of two spatial objects (e.g., *overlap*, *meet*). An important approach for characterizing the topological relationships between spatial objects is known as 9-intersection model [7]. By using this model, the authors in [16] have identified the topological relationships between any two complex spatial objects irrespective of their types. Thirty-three relationships have been found for two complex regions.

Numerous studies have been done on topological relationships as well as topological reasoning. The reasoning process tries to infer the unknown relationships from a set of explicitly known relationships which are defined by a particular relationship model. Hence, reasoning models depend on the underlying relationship models. Researchers from different domains such as AI, mathematics, GIS and databases, have been contributing to this field of study. The authors of the papers [18, 12, 3] tackle this problem with algebraic logic approaches. The authors in [7] define spatial objects on the basis of topological set theory and propose the 9-intersection model as a way to characterize them. Based on topological set theory, the authors also propose reasoning models for simple regions [3, 6, 5] and simple regions with holes [20]. In [1] the authors propose a reasoning model taking the concavity of the regions into the account along with their convex hulls. In most cases, the inferred relationship between spatial objects may not be unique, i.e., the inferred relationship can be a disjunction of several basic relationships. Based on this observation, the authors of [9, 10] propose hierarchical models for topological reasoning.

All of the above mentioned studies mainly focus on local inference (i.e., the composition of relationships involving three objects by means of inference rules). It is well understood that local inference is an essential and basic step of the reasoning process but without global inference the process is not complete. The reason behind the larger focus on local inference is that global inference is a constraint satisfaction problem (CSP) [4, 11, 19, 2] which is an extensively studied topic and is independent of the local inference process. The authors of [12, 17, 14] have studied the issues related to constraint satisfaction for spatial objects such as the complexity and the tractability. So far, the lowest complexity of CSP algorithms is $O(n^3)$ [11, 19, 2]. All of these CSP algorithms operate on a static knowledge base. That is, given a BSCN, the algorithm is able to infer relationships between any pair of complex regions. But over time, the existing facts may change and the CSP algorithms are not designed to handle changes. To the best of our knowledge, none of the reasoning models deals with changes of the spatial facts, and our work is motivated by this issue.

3. OVERVIEW OF THE REASONING PROCESS

The first step of the reasoning process is the *local inference* involving three regions in the form of $R_x(A, B)$ and $R_y(B, C)$. Here, R_x and R_y are the spatial facts between the complex regions A and B as well as B and C. The goal is to find the relationship $R_z(A, C)$. This local inference is carried out by a process called *composition of relationships* by means of a set of inference rules. It is important to note that the composition of relationships does not depend on the spatial features (like the extent) of the regions. Therefore, the composition of relationships can be denoted as $R_x \diamond R_y \Rightarrow R_z$. Local inference alone is not enough for inferring relationships between two complex regions. Consider the chain $R_1(A, B)$, $R_2(B, C)$, $R_3(C, D)$, $R_4(D, E)$ of topological relationships among the five regions A, B, C, D, and E. In this situation, local inference alone is not sufficient to infer the relationship between A and E because an intermediate object is required that is in relationship to both A and E. In our example, such an intermediate region does not exist. Thus, *global inference* comes into play which makes use of the composition of relationships to infer relationships between any two regions in the knowledge base.

An important observation is that global inference is orthogonal to local inference. That is, global inference can employ any algorithm to infer relationships globally as long as the composition of relationships is available. Unsurprisingly, global inference is a *constraint satisfaction problem*. A constraint satisfaction problem (CSP) is defined as a triple

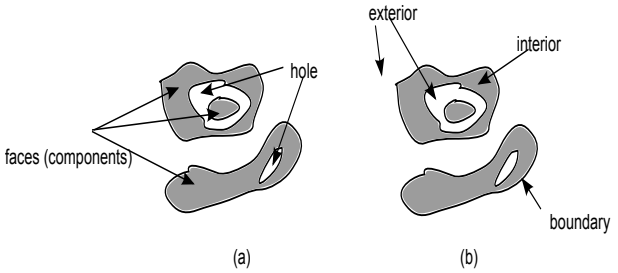


Figure 1: (a) A complex region with its faces and holes, and (b) its interior, boundary, and exterior.

(X, D, C), where X is a set of variables, D is a domain of values, and C is a set of constraints. Every constraint is in turn a pair (t, R), where t is a tuple of variables and R is a relation. The CSP can be viewed as a directed graph, where the nodes are the variables and the edges between two variables are the relations or the constraints. This directed graph is also called *constraint network*. In our case, the relations are all binary topological relationships and the variables are spatial objects (i.e., regions); we call this graph representation *binary spatial constraint network* ($BSCN$). The class of algorithms for global inferencing by using BSCN is based on a *path consistency* procedure. A pair of variables is path consistent with a third variable if each consistent evaluation of the pair can be extended to the other variable in such a way that all binary constraints are satisfied. Formally, the variables A and C are path consistent with B if there is a relation $R_1(A, C)$ that satisfies the binary constraint between A and C and if there are two relations $R_2(A, B)$ and $R_3(B, C)$ that satisfy the constraint between A and B and between B and C, respectively. A simple observation tells us that path consistency can be achieved through composition of relationships. The algorithm applies the path consistency procedure to all combinations of nodes in the BSCN until no new relationships can be inferred. An important point is that, given a partially observed knowledge base, the path consistency algorithms derive the complete knowledge, i.e., the relationships between every pair of objects. That is, after running the global inference algorithm the knowledge base becomes complete and it takes $O(1)$ time to find the relationship(s) between any pair of complex regions.

4. LOCAL INFERENCE

The local inference process (Figure 2) takes the two topological relationships $R_x(A, B)$ and $R_y(B, C)$, composes them, and infers the relationship(s) $R_Z(A, C)$. Since a 9-intersection matrix can uniquely characterize each topological relationship, the input of the local inference consists of the two 9-intersection matrices, and the output is a set of inferred relationships. In a first step, the corresponding set relationships (i.e., subset relationships, empty/nonempty intersections) between the interiors of the regions are evaluated from the 9-intersection matrices. In a second step, the inference rules are applied to find out the 9-intersection predicate values between A and C. In a last step, the inferred relationships are derived from the predicate values.

4.1 Set Relationships between The Interiors

Point set topology characterizes each spatial object by

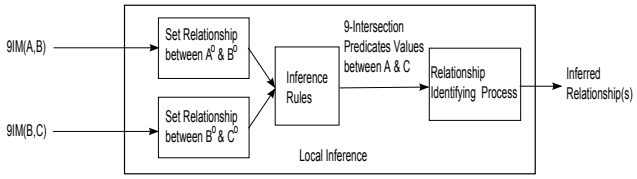


Figure 2: Steps of Local Inference.

three mutually exclusive point sets in the topological space $\mathbb{R}^2$. These sets are the interior (A^o), the boundary (∂A), and the exterior (A^-) for any spatial object A (Figure 1b). The 9-intersection model uses nine predicates to check the nine intersections of these point sets provided by two spatial objects A and B for non-emptiness. Each topological relationship between any two spatial objects is characterized by a unique combination of nine Boolean values. The 9-intersection predicates are arranged in a 9-intersection matrix shown in Figure 3a. For example, the spatial meeting configuration in Figure 3b is represented by the 9-intersection matrix in Figure 3c.

On the other hand, the interior, boundary, and exterior of a spatial object are uniquely defined and disjoint from each other [16]. Therefore, it is sufficient to specify any of these three sets to uniquely characterize a region object. In this document we consider the interior of a complex region to uniquely characterize it. Hence, for each topological relationship, there is a set relation between the interiors of the two complex regions. That is, either the interior of A is a subset of or a superset of or equal to or disjoint to or overlaps the interior of B. In [7] the authors show a way to find out the set relationship between any two components of a region object from the 9-intersection matrix by using the topological properties of the spatial regions. We employ the same technique to find out the set relation between the interiors of the two participating regions of a topological relationship.

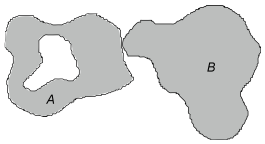
4.2 Inference Rules

From set theory, two non-empty sets X and Y must have one of the following five relations: (i) X is a proper subset of Y, (ii) X is equal to Y, (iii) Y is a proper subset of X, (iv) X and Y have some common and some different elements, and (v) X and Y do not have any common element. The fourth relation, we call it *overlap*, denotes that two sets have common elements but none of them is the proper subset of the other. We extend these five relations to eight by adding special cases to the relations (i), (iii), and (v) using the spatial properties. Consider X and Y as the interiors of two regions A and B respectively. Then relation (i) states that A is completely inside B. There can be two special cases of this scenario: (a) A is inside B and their boundaries touch, and (b) A is inside B and their boundaries do not touch. Similarly, these two special cases also hold for the relations (iii) and (iv).

Let the symbol $\subset$ denote the proper subset relation. The symbol $\diamond$ is to denote the predicate for overlap, e.g., $A^\circ \diamond B^\circ \Leftrightarrow (A^\circ \cap B^\circ \neq \varnothing \ \wedge \ A^\circ - B^\circ \neq \varnothing \ \wedge \ B^\circ - A^\circ \neq \varnothing)$. The predicate for a non-empty intersection, e.g., $A^\circ \cap B^\circ \neq \varnothing$, is denoted by $A^\circ B^\circ$, and the predicate for an empty intersection, e.g., $A^\circ \cap B^\circ = \varnothing$, is denoted by $\neg A^\circ B^\circ$. Hence, the eight relations between the interiors of two region objects are the following:

$$\begin{pmatrix} A^o \cap B^o \neq \varnothing & A^o \cap \partial B \neq \varnothing & A^o \cap B^- \neq \varnothing \\ \partial A \cap B^o \neq \varnothing & \partial A \cap \partial B \neq \varnothing & \partial A \cap B^- \neq \varnothing \\ A^- \cap B^o \neq \varnothing & A^- \cap \partial B \neq \varnothing & A^- \cap B^- \neq \varnothing \end{pmatrix}$$

(a)



(b)

$$\begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

(c)

Figure 3: (a) 9-Intersection Matrix, (b) complex regions A and B meet, (c) $R_{meet}(A, B)$.

1. $A^\circ \subset B^\circ \wedge \neg\partial A\partial B$
2. $A^\circ \subset B^\circ \wedge \partial A\partial B$
3. $A^\circ = B^\circ$
4. $A^\circ \diamond B^\circ$
5. $\neg A^\circ B^\circ \wedge \partial A\partial B$
6. $\neg A^\circ B^\circ \wedge \neg\partial A\partial B$
7. $B^\circ \subset A^\circ \wedge \partial A\partial B$
8. $B^\circ \subset A^\circ \wedge \neg\partial A\partial B$

The relations 1 and 2 are two special cases of the original relation (i). Similarly, the relations 5 and 6 as well as the relations 7 and 8 are special cases of the original relations (iii) and (v). Unsurprisingly, these five basic and eight extended relations correspond to the RCC-5 and RCC-8 approaches [13, 14]. Most importantly, these eight relations hold for any type of region objects (i.e., simple, complex) because a simple region is nothing but a single component, complex region without any hole. On the other hand, since we consider the interior as a whole, which means the interior of a complex region is the union of the interiors of its all faces, it does not matter how many holes and components are contained in a complex region. Since these eight relations completely characterize the relations between the interiors of two complex regions, any relationship between two complex regions A and B must include exactly one of these relations. Therefore, if we have $R_x(A, B)$ and $R_y(B, C)$ then by the transitivity property, the interiors of A and C must belong to exactly one of the $8 \cdot 8 = 64$ configurations of these relations. That is, for each relation between A and B, there are eight possible relations between B and C which gives us 64 configurations.

For each of these 64 configurations, we determine the 9-intersection predicate values between A and C. As an example, for the configuration $A^\circ \subset B^\circ \wedge \neg\partial A\partial B$ and $B^\circ \subset C^\circ \wedge \neg\partial B\partial C$, by applying simple set theory, we get $A^\circ \subset B^\circ \wedge B^\circ \subset C^\circ \Rightarrow A^\circ \subset C^\circ \Rightarrow A^\circ \cap C^\circ \neq \varnothing$. This means for the configuration of A and B and of B and C that the *interior-interior* intersection between A and C is always *true*. Similarly, for the same configuration we can prove that the *interior-exterior* intersection between A and C is always *false*. We know that the three components (i.e., interior, exterior and boundary) of a region object are mutually exclusive (i.e., $C^\circ \cap C^- = \varnothing$). Hence, $A^\circ \subset B^\circ \wedge B^\circ \subset C^\circ \Rightarrow A^\circ \subset C^\circ \wedge (C^\circ \cap C^- = \varnothing) \Rightarrow A^\circ \cap C^- = \varnothing$. On the other hand, for the configuration $A^\circ \diamond B^\circ \wedge B^\circ \diamond C^\circ$, we cannot say certainly whether $A^\circ \cap C^\circ$ is *empty* or *nonempty*, which means the outcome of $A^\circ \cap C^\circ$ is unknown. We can prove this statement by the two scenarios described in Figure 4 where for this same configuration we get different interior-interior intersection values between A and C.

Based on the above observations, for each configuration we can determine the values *true*, *false*, or *unknown* of all 9-intersection predicates between A and C. We do not need to determine the *exterior-exterior* intersection because it is always *true*. Hence, we define the remaining eight 9-intersection predicates by three sets of rules that specify for which configuration the predicate is supposed to yield certainly true, certainly false, and unknown. By applying some simple propositional logic reduction techniques and set theory notations (e.g., by combining $\subset$ and $=$ to $\subseteq$), we obtain the sets of inference rules for all 9-intersection predicates indexed by $P1$ to $P9$ in Figure 5.

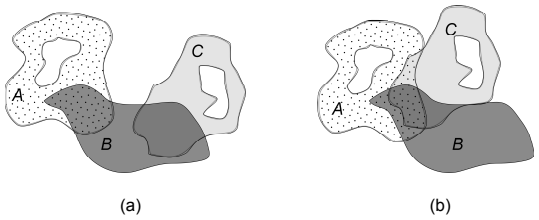


Figure 4: The interiors of A and C: (a) intersects, (b) does not intersect.

The proofs of these rules are performed by simple set theory and by proofs by counter-example and drawing as shown in Figure 4. The proofs of the rules are not given in this document due to space constraints. However, the completeness of this set of rules follows from the formulation of the rules. Two regions must have exactly one of the eight interior-interior set relations for any topological relationship, and after the composition A and C must satisfy one of the 64 configurations. Since the inference rules take each configuration into account, these rules never miss any scenario for which it cannot determine the 9-intersection predicates. Thus, the inference rules are complete by formulation.

4.3 Relationship Identifying Process

We evaluate the 9-intersection predicates (called *evaluated predicates*) of the topological relationship to be inferred by applying the inference rules defined in the previous subsection. These evaluated predicates have slightly different characteristics than the usual 9-intersection predicates because evaluated predicates may have the value *unknown* whereas usual 9-intersection predicates always have determinate values (i.e., either *true* or *false*). This is not surprising since the inferred relationship can be unique (i.e., a single basic relationship) or a disjunction of basic relationships. If the inferred relationship is unique, then all the evaluated predicate values are determinate. On the other hand, if the inferred relationship is a disjunction of basic relationships, then at least one of the evaluated predicates must have the value *unknown*. In fact, the evaluated predicates have determinate values only for those predicates that agree for all the relationships in that disjunction. Since we may have an indeterminate value, we need one more step to identify the relationship(s) from the evaluated predicates.

A simple brute force approach to finding out the inferred relationship is to compare the evaluated matrix against each of the 33 relationship matrices, predicate by predicate. The problem is that it takes too many comparisons. Since the

$P1 : A^\circ C^\circ =$

$$\begin{cases} true & A^\circ = B^\circ \wedge B^\circ = C^\circ \ \vee \\ & A^\circ B^\circ \wedge B^\circ \subset C^\circ \ \vee \\ & B^\circ C^\circ \wedge B^\circ \subset A^\circ \ \vee \\ & \neg A^\circ B^\circ \wedge \partial A \partial B \wedge B^\circ \subset C^\circ \wedge \neg \partial B \partial C \ \vee \\ & \neg B^\circ C^\circ \wedge \partial B \partial C \wedge B^\circ \subset A^\circ \wedge \neg \partial A \partial B \\ false & A^\circ \subseteq B^\circ \wedge \neg B^\circ C^\circ \ \vee \\ & \neg A^\circ B^\circ \wedge C^\circ \subseteq B^\circ \\ unknown & otherwise \end{cases}$$

$P2 : A^\circ \partial C =$

$$\begin{cases} true & (C^\circ \subset B^\circ \vee C^\circ \diamond B^\circ) \wedge B^\circ \subseteq A^\circ \ \vee \\ & C^\circ = B^\circ \wedge B^\circ \subset A^\circ \wedge \neg \partial B \partial A \\ false & C^\circ \subseteq B^\circ \wedge \neg B^\circ A^\circ \ \vee \\ & (\neg C^\circ B^\circ \vee B^\circ \subseteq C^\circ) \wedge A^\circ \subseteq B^\circ \\ unknown & otherwise \end{cases}$$

$P3 : A^\circ C^- =$

$$\begin{cases} true & C^\circ \subseteq B^\circ \wedge \neg (A^\circ \subseteq B^\circ) \ \vee \\ & (C^\circ \subset B^\circ \vee B^\circ \diamond C^\circ) \wedge A^\circ = B^\circ \ \vee \\ & B^\circ \diamond C^\circ \wedge (B^\circ \subset A^\circ \vee A^\circ \subset B^\circ) \ \vee \\ & \neg B^\circ C^\circ \wedge A^\circ B^\circ \ \vee \\ & \neg B^\circ C^\circ \wedge \neg \partial B \partial C \wedge \neg A^\circ B^\circ \wedge \partial A \partial B \ \vee \\ & C^\circ \subset B^\circ \wedge \neg \partial B \partial C \wedge A^\circ \subset B^\circ \wedge \partial A \partial B \ \vee \\ & B^\circ \subset C^\circ \wedge \partial B \partial C \wedge B^\circ \subset A^\circ \wedge \neg \partial A \partial B \\ false & B^\circ \subseteq A^\circ \wedge C^\circ \subseteq B^\circ \\ unknown & otherwise \end{cases}$$

$P4 : \partial A C^\circ =$

$$\begin{cases} true & (A^\circ \subset B^\circ \vee A^\circ \diamond B^\circ) \wedge B^\circ \subseteq C^\circ \ \vee \\ & A^\circ = B^\circ \wedge B^\circ \subset C^\circ \wedge \neg \partial B \partial C \\ false & A^\circ \subseteq B^\circ \wedge \neg B^\circ C^\circ \ \vee \\ & (\neg A^\circ B^\circ \vee B^\circ \subseteq A^\circ) \wedge C^\circ \subseteq B^\circ \\ unknown & otherwise \end{cases}$$

$P5 : \partial A \partial C =$

$$\begin{cases} true & A^\circ = B^\circ \wedge B^\circ = C^\circ \ \vee \\ & A^\circ = B^\circ \wedge (B^\circ \subset C^\circ \vee C^\circ \subset B^\circ \vee \neg B^\circ C^\circ) \wedge \partial B \partial C \ \vee \\ & B^\circ = C^\circ \wedge (B^\circ \subset A^\circ \vee A^\circ \subset B^\circ \vee \neg A^\circ B^\circ) \wedge \partial A \partial B \\ false & A^\circ \subset B^\circ \wedge \neg \partial A \partial B \wedge (B^\circ \subseteq C^\circ \vee \neg B^\circ C^\circ) \ \vee \\ & A^\circ \subset B^\circ \wedge \partial A \partial B \wedge (B^\circ \subseteq C^\circ \vee \neg B^\circ C^\circ) \wedge \neg \partial B \partial C \ \vee \\ & C^\circ \subset B^\circ \wedge \neg \partial B \partial C \wedge (B^\circ \subseteq A^\circ \vee \neg A^\circ B^\circ) \ \vee \\ & C^\circ \subset B^\circ \wedge \partial B \partial C \wedge (B^\circ \subseteq A^\circ \vee \neg A^\circ B^\circ) \wedge \neg \partial A \partial B \ \vee \\ unknown & otherwise \end{cases}$$

$P6 : \partial A C^- =$

$$\begin{cases} true & C^\circ \subseteq B^\circ \wedge \neg (A^\circ \subseteq B^\circ) \ \vee \\ & (A^\circ \subset B^\circ \vee A^\circ \diamond B^\circ \vee (\neg A^\circ B^\circ \wedge \neg \partial A \partial B)) \\ & \quad \wedge B^\circ = C^\circ \ \vee \\ & \neg B^\circ C^\circ \wedge A^\circ \subset B^\circ \ \vee \\ & \neg B^\circ C^\circ \wedge \neg \partial B \partial C \wedge (\neg A^\circ B^\circ \vee B^\circ \subset A^\circ) \\ & \quad \wedge \partial A \partial B \ \vee \\ & C^\circ \subset B^\circ \wedge \neg \partial B \partial C \wedge A^\circ \subset B^\circ \wedge \partial A \partial B \ \vee \\ & B^\circ \subset C^\circ \wedge \partial B \partial C \wedge B^\circ \subset A^\circ \wedge \neg \partial A \partial B \\ false & B^\circ \subseteq A^\circ \wedge C^\circ \subseteq B^\circ \\ unknown & otherwise \end{cases}$$

$P7 : A^- C^\circ =$

$$\begin{cases} true & A^\circ \subseteq B^\circ \wedge \neg (C^\circ \subseteq B^\circ) \ \vee \\ & (A^\circ \subset B^\circ \vee A^\circ \diamond B^\circ) \wedge B^\circ = C^\circ \ \vee \\ & A^\circ \diamond B^\circ \wedge (B^\circ \subset C^\circ \vee C^\circ \subset B^\circ) \ \vee \\ & \neg A^\circ B^\circ \wedge B^\circ C^\circ \ \vee \\ & \neg A^\circ B^\circ \wedge \neg \partial A \partial B \wedge \neg B^\circ C^\circ \wedge \partial B \partial C \ \vee \\ & A^\circ \subset B^\circ \wedge \neg \partial A \partial B \wedge C^\circ \subset B^\circ \wedge \partial B \partial C \ \vee \\ & B^\circ \subset A^\circ \wedge \partial A \partial B \wedge B^\circ \subset C^\circ \wedge \neg \partial B \partial C \\ false & B^\circ \subseteq A^\circ \wedge C^\circ \subseteq B^\circ \\ unknown & otherwise \end{cases}$$

$P8 : A^- \partial C =$

$$\begin{cases} true & A^\circ \subseteq B^\circ \wedge \neg (C^\circ \subseteq B^\circ) \ \vee \\ & (A^\circ \subset B^\circ \vee A^\circ \diamond B^\circ \vee (\neg A^\circ B^\circ \wedge \neg \partial A \partial B)) \\ & \quad \wedge B^\circ = C^\circ \ \vee \\ & \neg A^\circ B^\circ \wedge C^\circ \subset B^\circ \ \vee \\ & \neg A^\circ B^\circ \wedge \neg \partial A \partial B \wedge (\neg B^\circ C^\circ \vee B^\circ \subset C^\circ) \\ & \quad \wedge \partial B \partial C \ \vee \\ & A^\circ \subset B^\circ \wedge \neg \partial A \partial B \wedge C^\circ \subset B^\circ \wedge \partial B \partial C \ \vee \\ & B^\circ \subset A^\circ \wedge \partial A \partial B \wedge B^\circ \subset C^\circ \wedge \neg \partial B \partial C \\ false & B^\circ \subseteq A^\circ \wedge C^\circ \subseteq B^\circ \\ unknown & otherwise \end{cases}$$

P9: $A^-\ C^- = \text{true}$

Figure 5: Inference rules for the predicates of the 9-intersection matrix.

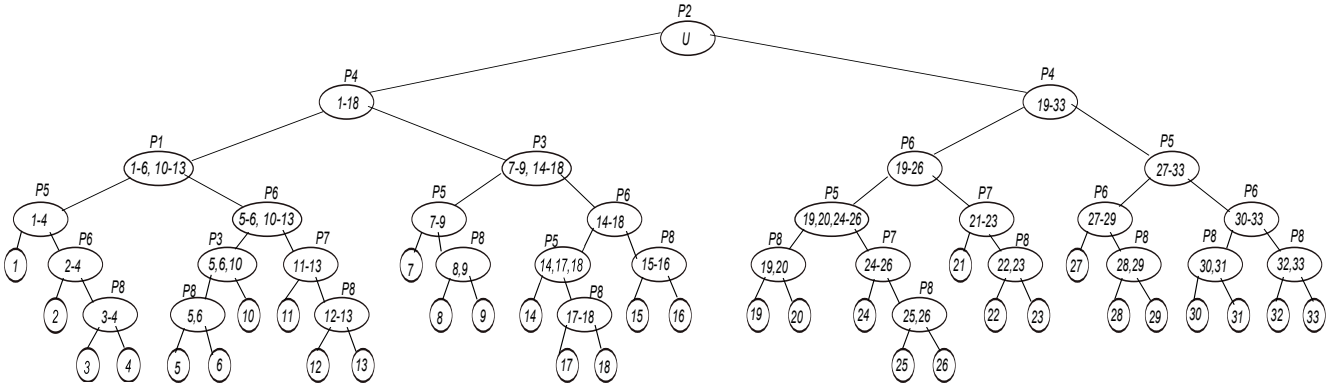


Figure 6: Decision tree of the relationship space for complex regions.

```
     algorithm IdentifyRelationship
(1)    input: Decision tree T := (V, E)
(2)           Intersection matrix IM
(3)    output: Inferred relationship R
(4)    begin
(5)      Step 1: Start with the root ∈ T
(6)      Step 2: At each node check the value of the evaluated predicate.
(7)        Step 2a: If the predicate value is 0, then follow the left subtree.
(8)        Step 2b: If the value is 1, then follow the right subtree.
(9)        Step 2c: If the value is unknown, then follow both subtrees.
(10)     Step 3: Repeat Step 2 until the leaf nodes are reached in all branches.
(11)     Step 4: If a single leaf is found, then return the corresponding relationship
(12)             else return the disjunction of all corresponding relationships.
     end IdentifyRelationship
```

Figure 7: The algorithm *IdentifyRelationship*.

exterior-exterior intersection is always *true*, we have to compare eight of these evaluated predicates for each matching which means $33 \cdot 8 = 264$ comparisons are required in the worst case.

To reduce the number of comparisons we build a *decision tree* of these 33 relationships. Table 1 shows all 33 possible relationship matrices [16]. We recursively divide the relationship space based on a predicate value at each level of the tree until we reach a single relationship. For example, 18 relationships (matrices 1 to 18 in Table 1) have *false* as their *interior-boundary* intersection value. Thus, we divide the relationship space so that the relationships 1 to 18 are on one side and the relationships 19 to 33 are on the other side. Next if we look into the relationships 19 to 33, we find that the relationships 19 to 26 represented by the matrices 19 to 26 have the value *false* for the *boundary-interior* predicate and that the other relationships have the predicate value *true*. Therefore, we again divide the relationship space where relationship 19 to 26 is on one side and relationships 27 to 33 are on the other side. We continue this process until there is only one relationship in a leaf node. At each level, we divide the relationship space into half as close as possible to attain minimum average path length from the root to the leaf nodes. Since, there are 33 relationships a balanced binary tree should have the height $\lceil \log_2 33 \rceil = 6$. Our decision tree also has the height six. Though, this tree is not unique but this tree has the minimum average path length. The complete decision tree is shown in Figure 6. Each inner node has two entries. The entry inside a node describes the current relationship space that is considered, and the entry above the node denotes the predicate that has to be considered to further divide the current relationship space.

With the help of this tree, we design a recursive algorithm *IdentifyRelationship* (Figure 7) for identifying the inferred relationship. The input of the algorithm is the decision tree T and the 9-intersection matrix (IM) which is the evaluated matrix. The output is the inferred relationship (R). At each node, starting from the root, the value of the predicate assigned to that node is retrieved from the evaluated matrix and checked. Depending on the value, we follow either the left, right, or both subtrees. This process recursively follows down to the tree until a leaf node is reached. If all the evaluated predicates have determinate values (i.e., *true* or *false*), only one leaf node is reached. Otherwise, if any predicate has an indeterminate value (i.e., *unknown*), more than one leaf node is found. In this case, the inferred relationship is the disjunction of all the corresponding relationships represented by those leaf nodes. The maximum height of this decision tree is 6. This means if all the evaluated predicates have determinate values, in the worst case it would take 6 comparisons instead of 264 comparisons, which is a 97% improvement. Since evaluated predicates can have indeterminate values, we may end up searching through the whole tree in the worst case. The required number of comparisons to search through the whole tree is equal to the number of the inner nodes. The decision tree that we show in Figure 6 has 32 inner nodes. Consequently, 32 instead of 264 comparisons are sufficient which is an improvement of 88%.

5. GLOBAL INFERENCE

As we have already discussed in Section 3, a well accepted way of carrying out global inference is by means of path consistency algorithms. The first problem of this approach is the high complexity. Since such an algorithm generates a complete knowledge base, it is required to run only once at the beginning. One could argue that the higher runtime can be counted as pre-processing time and that it is a one time overhead. This argument holds when the database is static or changes rarely. If the database changes frequently, the runtime of the algorithm becomes a big overhead. For example, if a new object is added to the database then the algorithm should run again with this new information. The same argument holds if there is a change in any relationship because that change may cause other relationships to adjust. This means the algorithm should run to propagate those updates. In case of the deletion of an object, only the object and the emanating relationships from it have to be deleted. Therefore, the $O(n^3)$ overhead is incurred almost every time when there is a change, and this becomes worse when the database is large (i.e., n is large).

The second problem arises in the case of answering complex queries. For example, assuming there are two regions A and C describing areas affected by two different earthquakes. We want to know if there is any part of state S which was hit by both earthquakes. The answer can be obtained by looking at the topological relationship between the intersection of A and S as well as the intersection of C and S. Let the intersections be denoted by I_1 and I_2 respectively. Our goal is to find the relationship between these two regions. For

Matrix 1	Matrix 2	Matrix 3	Matrix 4	Matrix 5	Matrix 6	Matrix 7
$\begin{pmatrix}0&0&1\\0&0&1\\1&1&1\end{pmatrix}$	$\begin{pmatrix}0&0&1\\0&1&0\\1&1&1\end{pmatrix}$	$\begin{pmatrix}0&0&1\\0&1&1\\1&0&1\end{pmatrix}$	$\begin{pmatrix}0&0&1\\0&1&1\\1&1&1\end{pmatrix}$	$\begin{pmatrix}1&0&0\\0&1&0\\0&0&1\end{pmatrix}$	$\begin{pmatrix}1&0&0\\0&1&0\\1&1&1\end{pmatrix}$	$\begin{pmatrix}1&0&0\\1&0&0\\1&1&1\end{pmatrix}$
Matrix 8	**Matrix 9**	**Matrix 10**	**Matrix 11**	**Matrix 12**	**Matrix 13**	**Matrix 14**
$\begin{pmatrix}1&0&0\\1&1&0\\1&0&1\end{pmatrix}$	$\begin{pmatrix}1&0&0\\1&1&0\\1&1&1\end{pmatrix}$	$\begin{pmatrix}1&0&1\\0&1&0\\1&1&1\end{pmatrix}$	$\begin{pmatrix}1&0&1\\0&1&1\\0&0&1\end{pmatrix}$	$\begin{pmatrix}1&0&1\\0&1&1\\1&0&1\end{pmatrix}$	$\begin{pmatrix}1&0&1\\0&1&1\\1&1&1\end{pmatrix}$	$\begin{pmatrix}1&0&1\\1&0&1\\1&1&1\end{pmatrix}$
Matrix 15	**Matrix 16**	**Matrix 17**	**Matrix 18**	**Matrix 19**	**Matrix 20**	**Matrix 21**
$\begin{pmatrix}1&0&1\\1&1&0\\1&0&1\end{pmatrix}$	$\begin{pmatrix}1&0&1\\1&1&0\\1&1&1\end{pmatrix}$	$\begin{pmatrix}1&0&1\\1&1&1\\1&0&1\end{pmatrix}$	$\begin{pmatrix}1&0&1\\1&1&1\\1&1&1\end{pmatrix}$	$\begin{pmatrix}1&1&1\\0&0&1\\0&0&1\end{pmatrix}$	$\begin{pmatrix}1&1&1\\0&0&1\\1&1&1\end{pmatrix}$	$\begin{pmatrix}1&1&1\\0&1&0\\0&0&1\end{pmatrix}$
Matrix 22	**Matrix 23**	**Matrix 24**	**Matrix 25**	**Matrix 26**	**Matrix 27**	**Matrix 28**
$\begin{pmatrix}1&1&1\\0&1&0\\1&0&1\end{pmatrix}$	$\begin{pmatrix}1&1&1\\0&1&0\\1&1&1\end{pmatrix}$	$\begin{pmatrix}1&1&1\\0&1&1\\0&0&1\end{pmatrix}$	$\begin{pmatrix}1&1&1\\0&1&1\\1&0&1\end{pmatrix}$	$\begin{pmatrix}1&1&1\\0&1&1\\1&1&1\end{pmatrix}$	$\begin{pmatrix}1&1&1\\1&0&0\\1&1&1\end{pmatrix}$	$\begin{pmatrix}1&1&1\\1&0&1\\1&0&1\end{pmatrix}$
Matrix 29	**Matrix 30**	**Matrix 31**	**Matrix 32**	**Matrix 33**		
$\begin{pmatrix}1&1&1\\1&0&1\\1&1&1\end{pmatrix}$	$\begin{pmatrix}1&1&1\\1&1&0\\1&0&1\end{pmatrix}$	$\begin{pmatrix}1&1&1\\1&1&0\\1&1&1\end{pmatrix}$	$\begin{pmatrix}1&1&1\\1&1&1\\1&0&1\end{pmatrix}$	$\begin{pmatrix}1&1&1\\1&1&1\\1&1&1\end{pmatrix}$		

Table 1: 33 possible topological relationships between two complex regions.

this purpose, we need to add these two regions as two nodes in the BSCN and run the path-consistency algorithm. The algorithm gives us not only the relationship between I_1 and I_2 but also the relationships between I_1 and all the other nodes as well as the relationships between I_2 and all the other nodes. But we do not need these extra relationships. Hence, the whole procedure becomes quite inefficient. Moreover, I_1 and I_2 are temporary regions only and are thrown out of the BSCN after the query execution. When those temporary regions are thrown out, the BSCN must revert to its previous state. This means we need to save the previous state of the BSCN when any such complex query is posed. Based on these observations, we can argue that complete knowledge may not be desirable in some cases and that path consistency algorithms are not designed to handle database changes. Hence, our goal is to develop a different runtime strategy to carry out global inference.

Three scenarios can arise when a query is made to find out the topological relationship between two regions: (i) the relationship is already known which means no reasoning is required, (ii) no relationship is available and there are no intermediate nodes through which we can infer the relationship, and (iii) no relationship is available but there are some intermediate nodes through which we can infer the relationship. In terms of a graph, these three scenarios are equivalent of having (i) a direct edge between the two nodes, (ii) no path between the two nodes, and (iii) at least one path between the nodes respectively. The first scenario is straightforward so that we have only to be concerned about the other two scenarios. It is very important to identify whether it is possible to infer knowledge between two given regions. The reasoning procedure is a costly process. If we could anticipate that the inference of new knowledge between two complex regions is impossible before starting the procedure, it would

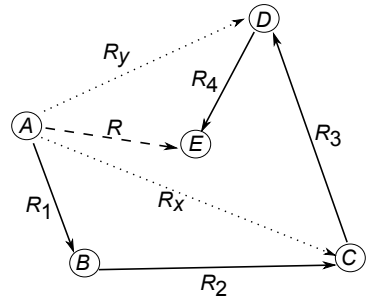


Figure 8: A chain of relationships.

save us time and resources. But surprisingly the solution is straightforward. Since the BSCN is a graph, a simple path finding algorithm that assumes one of the two regions as the source and the other one as the destination can answer this question. A necessary condition for reasoning is that there is a path between the nodes representing the two regions.

Therefore, the first step is to run a path finding algorithm. A path between two target nodes through a set of intermediate nodes corresponds to the chaining example that we described before in the Introduction. Figure 8 describes the scenario where A and E are the target nodes and B, C, and D are the intermediate nodes. The known relationships are $R_1(A, B)$, $R_2(B, C)$, $R_3(C, D)$, and $R_4(D, E)$, and our goal is to infer $R(A, E)$. We can solve this long chain of relationships by simplifying it into a series of compositions of relationships involving three nodes. Referring to Figure 8, we first compose $R_1(A, B)$ and $R_2(B, C)$ to get $R_x(A, C)$. Then we compose $R_x(A, C)$ and $R_3(C, D)$ to obtain $R_y(A, D)$. Finally, by composing $R_y(A, D)$ and $R_4(D, E)$, we get $R(A, E)$. In the AI domain, this process is

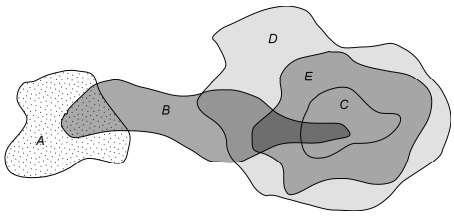


Figure 9: Multiple chains of relationships.

known as *forward chaining*.

Intuitively, *shortest path algorithms* are a good choice for a path finding algorithm because they can give us the path with the minimum number of intermediate nodes; this might ensure a lower processing time. However, let us consider a configuration with two chains (paths). First, we assume that A *overlaps* B and B *overlaps* C. Second, we assume that A *disjoint* D, D *contains* E, and E *contains* C (Figure 9). From the first chain the inferred relationship between A and C is the *universal* relationship, i.e., the disjunction of all possible relationships. But from the second chain the inferred relationship between A and C is *disjoint*. Though both results are correct, the second, longer chain gives us the more specific and thus better answer. A similar example can be shown where the shorter path gives us a more specific and thus better answer. In fact, this shows that there is no relation between the length of the path and the more specific answer. This means that by considering one path, we may not obtain the most specific answer. Hence, we have to consider all possible paths, and the intersections of all inferred relationships obtained through these paths should give us the most specific relationship. The problem is that in the worst case the number of all simple paths between two nodes in a graph is $n!$ when the graph is complete. Interestingly, this worst case scenario is actually good for the reasoning process because we don't need any inference when the knowledge base is complete. If the graph is sparse then the number of paths between any nodes may be much lower depending on the configuration of the graph. Based on this observation, an alternative heuristic solution is to consider k paths instead of all simple paths. If k is large then it is possible that this heuristic may give us the most specific result. We choose a k-shortest path algorithm which is a generalization of the shortest path problem and determines k paths, instead of one, in an increasing order of length. The length, in our case, is measured as the number of hops from source to destination which means the edges of the BSCN are of equal weight which means we treat all the relationships equally. The worst case complexity for the k-shortest simple path algorithm is $O(m + n \log n + k)$ [8] where n is the number of nodes and m is the number of edges. If we choose $k = cn$ where c is a positive integer then the complexity becomes $O(n \log n)$ for $n \log n >= m$.

6. AN ALGORITHM FOR REASONING BETWEEN COMPLEX REGIONS

So far, we have described the two basic steps of the reasoning process. In this section, we integrate these steps which give us a generalized conceptual model for reasoning as well as a complete picture of our work. The algorithm is also the starting point of the implementation of this conceptual model. We employ the *k-shortest simple path* algorithm and assume that k is equal to the number of nodes in the BSCN. The inputs of the algorithm *ReasoningBetweenComplexRegions* (Figure 10) are the BSCN G, a matrix M, which stores the existing relationships, and the two complex regions α and β for which we infer the relationship. The matrix M is indexed by (i, j) which means the topological relationship between the complex objects i and j is stored in the matrix entry $M_{i,j}$. The output of the algorithm is the inferred relationship. There is a simple check (line 7) to find out whether the relationship already exists or not. If the relationship already exists, we simply return this relationship and no reasoning is required. The reasoning procedure has two loops. The outer loop (lines 9 to 20) executes a *k-shortest path algorithm*. Each time when we get a new path (i.e., $p_{\alpha,\beta}$), the inner loop (lines 13 to 17) is executed. This inner loop executes the *forward chaining* process. In this loop, the composition of relationships is performed in three steps. First, the set relations between the interiors of the regions under consideration are being evaluated (line 14). Then, the evaluation of the 9-intersection predicates by means of the inference rules is performed (line 15), and finally the inferred relationship is obtained by passing those evaluated predicates to the relationship identifying process (line 16). In order to find out the most specific result, we take the intersection of all inferred relationships which are obtained through different paths (line 18). The complexity of the inner loop depends on the length of the chain because applying the inference rules and the relationship identifying process requires a constant amount of time. In a graph the maximum path length between any nodes can be $|V| - 1$. Hence, the time complexity of the inner loop is $O(n)$. Since the complexity of the outer loop is $O(n \log n)$, this gives us the total complexity of $O(n^2 \log n)$. This complexity is lower than the complexity $O(n^3)$ of the original BSCN path-consistency algorithm. Though the reduction of the complexity is not dramatic, the main advantage is that we only need to run this algorithm when a query is fired. Therefore, this approach can save a lot of overhead for large dynamic databases. It also solves the complex query problem because it only computes the relationship of the target objects without modifying any other relationships in the database.

7. SIMULATION AND RESULTS

The performance of the heuristic depends on the percent of time the heuristic is able to find the most specific relationship between two regions. Since we consider k paths, instead of all paths, between two nodes representing the two regions, it is possible that we might miss the path which could give us the most specific relationship. Let us assume that the number of paths in a BSCN between any two nodes is E. If $k \geq E$, then we can surely say (i.e., with probability $p = 1$) that the heuristic gives us the most specific result. On the other hand, if $k < E$ then the probability of obtaining the most specific relationship is $p = k/E$ since all the edges have equal weights. We generate a random graph which represents the BSCN. The number of edges of each node is power law distributed between 1 and n, where n is the number of nodes in the graph. The reason is that the edges represent the information available about the nodes. In practice, we have a lot of information for a few regions, a reasonable amount of information for many regions, and less information about

```
algorithm ReasoningBetweenComplexRegions
(1)  input: BSCN G := (V, E)
(2)        Matrix M keeping the topological relationship between two complex regions
(3)        α, β ∈ V representing complex regions
(4)  output: The inferred relationship R(α, β) between nodes (complex regions) α and β
(5)  begin
(6)    if M_{α,β} ≠ null then
(7)      return M_{α,β}
(8)    k := 0
(9)    repeat
(10)     p_{α,β} := find the next best path from α to β in G
(11)     // p_{α,β} is a list of nodes from G that starts with α, ends with β and
(12)     // includes the intermediate nodes
(13)     for each i in intermediate nodes from p_{α,β}
(14)       S_i:=Evaluate the set relations between the interiors from the 9IM of M_{α,i}, M_{i,i+1}
(15)       IM:=Evaluate 9-intersection predicates by means of inference rules(S_i)
(16)       R_t(α, i + 1) := IdentifyRelationship(IM)
(17)     endfor
(18)     R(α, β) := R(α, β) ∩ R_t(α, β)
(19)     k := k + 1
(20)   until there are no paths from α to β or k = |V|
(21)   return R(α, β)
       end ReasoningBetweenComplexRegions
```

Figure 10: The algorithm *ReasoningBetweenComplexRegions*.

the rest of the regions. This phenomenon is captured by the power law distribution. We run the simulation for different sizes of databases and observe the performance of the heuristic by varying k. At each run, the performance is measured by averaging the p for all possible pairs of nodes. The number k of considered paths is a constant multiple of the number of nodes, i.e., $k = cn$ to keep the complexity of the k-shortest path bounded to $O(n \log n)$. Figure 11 shows that the performance of the heuristic decreases with the increase of the database size, which is expected. Figure 11 also shows that for a fixed database size, the performance increases if we consider more paths, i.e., if we increase c. For small databases such as $10 \leq n \leq 50$, the heuristic is able to find the most specific result more than 90% of time which is considered to be good performance by a heuristic. The heuristic performs reasonably well (i.e., above 80%) in case of medium sized databases with $50 \leq n \leq 300$. As the number of nodes grows beyond 300 nodes, the heuristics does not perform well when $c \leq 10$. But we see that significant performance gain can be obtained by considering more paths (e.g., $c = 20$). Though, increasing c does not hurt the overall complexity as long as $n >> c$ but it slows the algorithm by the factor of $c_2 \log c_2 / c_1 \log c_1$ where $c_2 > c_1$. Based on this observation, the value of c can be set by the user based on the size of the database and the requirement of precision.

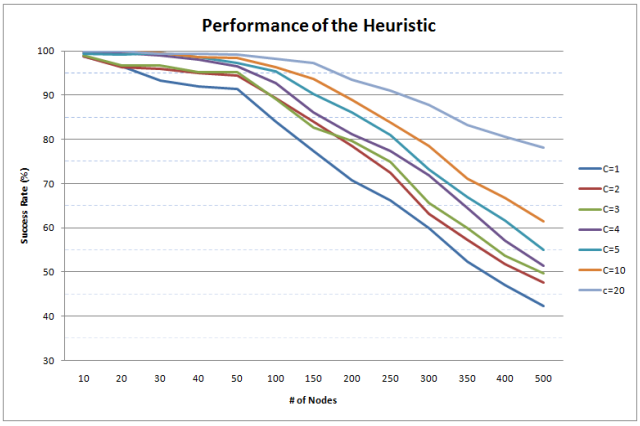


Figure 11: Performance of the heuristic for different database sizes.

8. CONCLUSIONS AND FUTURE WORK

From an application point of view, more complex geometric structures than the simple spatial objects are required to represent real world spatial phenomena. It is often the case that if the database is large and complex, the complete knowledge regarding the participating objects is unavailable. The first contribution of this paper is the design of a complete set of inference rules through which we can infer topological relationship between complex regions. The inference rules are formulated in such a way that they can also be applied to simple regions. Our second contribution is to define an overall conceptual framework for the reasoning process from a database point of view which can handle the typical database issues like updating, adding, and deleting information.

A main topic for future work is to implement the framework in spatial databases. We plan to apply some *algorithmic* (e.g., dynamic programming) and Artificial Intelligence (e.g., forward chaining, decision tree) techniques to implement this conceptual reasoning framework. An important topic for future work is to explore other heuristics for global inference such as using different weights for the edges. In this document we consider equal weights for all relationships. But an observation, in case of simple regions, shows that composing any relationship with the *overlap* relationship always results in a disjunction of relationships. Hence, it is less probable that the most specific result can be found if a chain

contains an *overlap* relationship. We can give higher weight to the edges representing *overlap* so that a chain containing *overlap* is considered later by the k-shortest path algorithm. Another important topic for future work is extending the reasoning model to all combinations of complex objects such as *line-line* and *line-region*.

9. REFERENCES

[1] A. Abdelmoty and B. El-Geresy. A general method for spatial reasoning in spatial databases. In *4th Int. Conf. on Information and Knowledge Management*, pages 312–317, 1995.

[2] P. V. Beek. On the minimality and decomposability of constraint networks. In *In Proc. of the 10th National Conference on Artificial Intelligence*, pages 447–452, 1992.

[3] Z. Cui, A. Cohn, and D. Randell. Qualitative and topological relationships in spatial databases. *Int. Symposium on Advances in Spatial Databases*, pages 296–315, 1993.

[4] R. Dechter. Constraint networks. *Encyclopedia of Artificial Intelligence*, 1:276–285, 1992.

[5] M. Egenhofer. Reasoning about binary topological relations. In *2nd Int. Symp. on Advances in Spatial Databases*, pages 143–160, 1991.

[6] M. Egenhofer. Deriving the composition of binary topological relations. *Journal of Visual Languages and Computing*, 5(2):133–149, 1994.

[7] M. Egenhofer and J. Herring. Categorizing binary topological relations between regions, lines, and points in geographic databases. *Technical Report, Department of Surveying Engineering, University of Maine, Orono, ME*, 1994.

[8] D. Eppstein. Finding the k shortest paths. *SIAM Journal on Computing*, 28(2):652–673, 1999.

[9] M. Grigni, D. Papadias, and C. Papadimitriou. Topological inference. In *14th Int. Joint Conference on Artificial Intelligence*, pages 901–907, 1995.

[10] V. Haarslev and R. Moller. SBox: A qualitative spatial reasoner's progress report. In *11th Int. Workshop on Qualitative Reasoning*, pages 3–6, 1997.

[11] A. K. Mackworth and E. C. Freuder. The complexity of some polynomial network consistency algorithms for constraint satisfaction problems. *Artificial Intelligence*, 25(1):65–74, 1985.

[12] R. Maddux. Some algebras and algorithms for reasoning about time and space. *Internal paper, Department of Mathematics, Iowa State University, Ames, Iowa*, 1989.

[13] Randell, D.A., Z. Cui, and A. G. Cohn. A spatial logic based on regions and connection. In *3rd International Conference on Knowledge Representation and Reasoning*, 1992.

[14] J. Renz and B. Nebel. On the complexity of qualitative spatial reasoning: A maximal tractable fragment of the region connection calculus. *Artificial Intelligence*, 108(1):69–123, 1999.

[15] M. Schneider. *Spatial data types for database systems: finite resolution geometry for geographic information systems*. Springer Verlag, 1997.

[16] M. Schneider and T. Behr. Topological relationships between complex spatial objects. *ACM Transactions on Database Systems*, 31(1):81, 2006.

[17] T. Smith and K. Park. Algebraic approach to spatial reasoning. *Int. Journal of Geographical Information Science*, 6(3):177–192, 1992.

[18] A. Tarski. On the calculus of relations. *The Journal of Symbolic Logic*, 6(3):73–89, 1941.

[19] A. M. University, A. K. Mackworth, and E. C. Freuder. The complexity of constraint satisfaction revisited. *Artificial Intelligence*, 59:57–62, 1993.

[20] M. Vasardani and M. Egenhofer. Comparing relations with a multi-holed region. *Spatial Information Theory*, pages 159–176, 2009.