Title
Algorithms for Hierarchical Spatial Reasoning (96-2)

Authors
Papadias, Dimitris
Egenhofer, Max

Publication Date
1996-02-01

NCGIA

National Center for Geographic Information and Analysis

Algorithms for Hierarchical Spatial Reasoning

by

Dirnitris Papadias
and
Max Egenhofer

University of Maine, Orono

Technical Report 96-2

February 1996

Simonett Center for Spatial Analysis
University of California
35 10 Phelps Hall
Santa Barbara, CA 93106-4060
Office (805) 893-8224
Fax (805) 893-8617
ncgia@ncgia.ucsb.edu

State University of New York
301 Wilkeson Quad, Box 610023
Buffalo NY 14261-0001
Office (716) 645-2545
Fax (716) 645-5957
ncgia@ubvms.cc.buffalo.edu

University of Maine
348 Boardman Hall
Orono ME 04469-5711
Office (207) 581-2149
Fax (207) 581-2206
ncgia@spatial.maine.edu

Algorithms for Hierarchical Spatial Reasoning

Dimitris Papadias, Max Egenhofer

National Center for Geographic Information and Analysis &
Department of Spatial Information Science and Engineering
University of Maine, ME 04469-5711
e-mail:{dimitris, max}@spatial.maine.edu
fax: 207- 5812206
tel: 207-5812127

Abstract In several applications, there is the need to reason about spatial relations using multiple local frames of reference organized in aggregation hierarchies. In this paper we deal with *direction relations*, a special class of spatial relations that describe order in space (e.g., north, northeast). We assume a spatial database of points and regions. Points belong to regions, which may be parts of larger regions and so on. The direction relations between points in the same region are explicitly represented. Inference mechanisms are applied to extract the relation between points in different regions and detect inconsistencies. We study two complementary types of inference. The first one derives the relation between two points that exist in different regions through chains of common points using path consistency. The second type of inference uses the relation between ancestor regions to infer the relation between the points. The paper describes algorithms for both types of inference and discusses their computational complexity.

Keywords: Spatial Reasoning, Geographic Databases, Direction Relations, Path Consistency

Acknowledgements: The authors would like to thank Christos Papadimitriou and Timos Sellis for providing useful comments.

This work was partially supported by the National Science Foundation under grant number No. SES 88-10917 for the NCGIA. Max Egenhofer's work is further supported by the National Science Foundation under grant number IRI-9309230, Rome Laboratories under grant number F30602-95-1-0042, the Scientific and Environmental Division of the North Atlantic Treaty Organization, and by Massive Digital Data Systems.

1. INTRODUCTION

Direction relations constitute a special class of spatial relations that deal with order in space (*left*, *northeast*). The large availability of spatial data from various sources and in various forms (e.g., satellite images, video), in combination with Progress in Multimedia, Image, and Spatial Databases, and Geographic Information Systems (GIS), created the need to answer queries involving direction (and other spatial) relations (e.g. "find all major cities northeast of Boston in New England"). This has motivated a significant amount of research on Reasoning (Smith and Park, 1992; Egenhofer and Sharma, 1993), Query Processing (Clementini et al., 1994; Papadias et al., 1995) and Spatial Query Languages (Roussopoulos et al., 1988; Egenhofer, 1994, Papadias and Sellis, 1995).

A number of *relation-based* systems have been proposed for the representation of direction relations. Chang et al., (1987) designed the *2D strings* for iconic indexing in Image Databases. A 2D string is a pair of one dimensional strings that represent the symbolic projections of the objects on the x and y axis. Glasgow and Papadias (1992) developed *symbolic arrays*, nested array structures that preserve directions relations among the distinct parts of complex spatial entities at different levels. Most of previous work, however, has focused on the representation and processing of explicit relations; the proposed systems do not include mechanisms for inference and inconsistency checking. Although some approaches have dealt with hierarchical direction reasoning, these mainly concern specific types of hierarchies captured by certain types of representations (Glasgow, 1994).

This paper studies hierarchical reasoning about direction relations in spatial databases of points and regions. The objects in the database form aggregation hierarchies: a point belongs to a region, which may be part of another region and so on. As an example, consider that cities are points in states represented by regions. The states are grouped together to form larger geographic entities (e.g., countries) in the next level and so on. We assume the existence of multiple hierarchies, that is, a spatial entity (region or point) could belong to more than one regions in the next level of hierarchy.

Direction relations between objects in the same spatial entity are explicitly represented and consistent. Although relations within the same entity are consistent, inconsistencies may occur by combining spatial knowledge from different sources. For example, consider a database of maps. If a city is northeast of another city in the map of region A, and the second city is north of a third city in the map of region B, then the existence of the first city south of the third in region C would yield an inconsistency. Such inconsistencies may occur when data about the same or overlapping areas are collected form different sources such as images, topographic surveys, verbal descriptions etc. (for an extended discussion see Frank, 1992). Spatial inference mechanisms are essential for explicating relations and enforcing consistency in the database.

The rest of the paper is organized as follows: Section 2 defines direction relations between points and regions, and describes spatial databases preserving directions. Section 3 presents an algorithm for the inference of the relation between points using the relations of their ancestor regions. Section 4 describes a complementary form of inference that uses chains of common points and achieves path consistency for the whole database. Section 5 studies the complexity of the algorithms and discusses the efficiency of hierarchical representations. Section 6 proposes extensions for alternative types of direction relations, and Section 7 concludes with comments.

2. DIRECTION RELATIONS

There have been two approaches in defining direction relations (Hernandez, 1994). According to the *cone-shaped* approach, direction relations are defined using angular regions between objects (Peuquet, Ci-Xiang, 1987; Dutta, 1989). Our method is *projection based*, that is, direction relations are defined using projection lines vertical to the coordinate axes (Mukerjee and Joe, 1991; Sistla et al., 1994). For the following discussion, P, P_1, P_2 ... denote points, A, A_1, A_2 ... regions, and X, X_1, X_2 ... objects (points or regions). In this paper we are concerned with spatial DBMSs that store only direction relations between distinct objects and not absolute coordinates.

2.1 Direction Relations Between Points and Regions

We use the notation $A \vdash NW(P_1,P_2)$ to express that point P_1 is NorthWest of P_2 in the map of region A. There are nine "primitive" direction relations between points if we assume projection-based definitions (Freksa, 1992; Papadias and Sellis, 1994). Figure 1 illustrates these relations depending on the position of a primary point with respect to a reference point (denoted by *) in region A. In addition to the eight relations of Figure 1, there is also SamePosition which means that the points are at the same location.

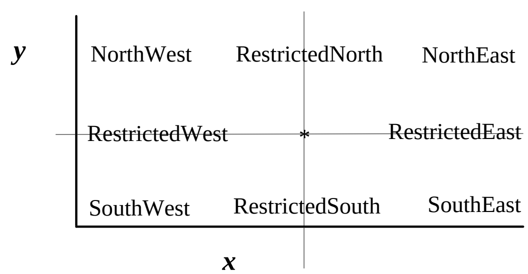


Figure 1 Primitive direction relations between points

Exactly one of the previous relations holds true between any pair of point objects in a region. The primitive relations are transitive and *SP* is also symmetric. The rest form four pairs of converse relations (e.g., $A \vdash NW(P_1,P_2) \Leftrightarrow A \vdash SE(P_2,P_1)$). *U* denotes the *universal* relation, the disjunction of all primitive relations. The relation $\varnothing$ denotes the *empty* relation (the relation that arises during inconsistencies). The above relations form a relation algebra and can be used for relation-based reasoning. They constitute the set of *high resolution* relations; we also define a set of *low resolution* relations using disjunctions:

$A \vdash N(P_1,P_2) \equiv A \vdash NW(P_1,P_2) \vee A \vdash RN(P_1,P_2) \vee A \vdash NE(P_1,P_2)$ *(North)*

$A \vdash E(P_1,P_2) \equiv A \vdash NE(P_1,P_2) \vee A \vdash RE(P_1,P_2) \vee A \vdash SE(P_1,P_2)$ *(East)*

$A \vdash S(P_1,P_2) \quad \equiv \quad A \vdash SW(P_1,P_2) \vee A \vdash RS(P_1,P_2) \vee A \vdash SE(P_1,P_2)$ *(South)*

$A \vdash W(P_1,P_2) \quad \equiv \quad A \vdash NW(P_1,P_2) \vee A \vdash RW(P_1,P_2) \vee A \vdash SW(P_1,P_2)$ *(West)*

$A \vdash SL(P_1,P_2) \quad \equiv \quad A \vdash RW(P_1,P_2) \vee A \vdash SP(P_1,P_2) \vee A \vdash RE(P_1,P_2)$ *(SameLevel)*

$A \vdash SH(P_1,P_2) \quad \equiv \quad A \vdash RN(P_1,P_2) \vee A \vdash SP(P_1,P_2) \vee A \vdash RS(P_1,P_2)$ *(SamewidtH)*

In order to define direction relations between regions we use projections on the x and y axis. There are 13 mutually exclusive relations between intervals in 1D space (Allen, 1983). If we extend Allen's relations to 2D space we get the 169 primitive relations between region projections of Figure 2. $A \vdash P_{1-1}(A_1,A_2)$ means that A_1 and A_2 are related by projection relation P_{1-1} in the map of area A that contains A_1 and A_2. Previous structures aimed at the representation of direction relations, such as the 2D Strings and Symbolic Arrays, preserve only the above type of relations and discard other forms of spatial information, such as shape, distance and topological relations.

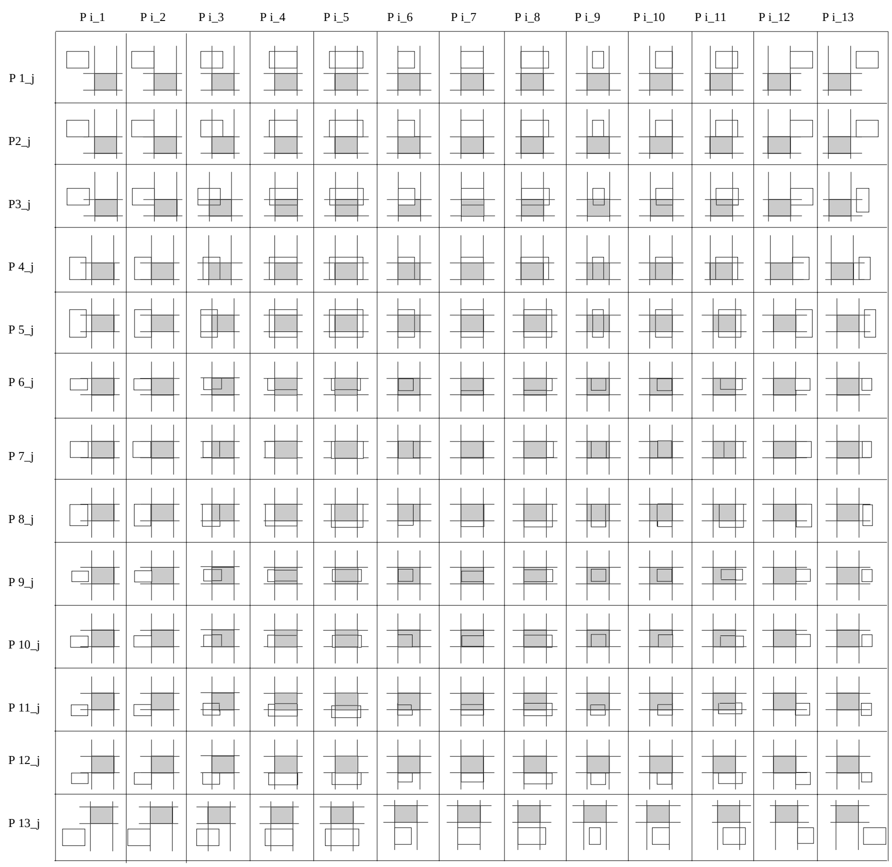


Figure 2 Projection relations in 2D space

2.2 Retrieval of Direction Relations in Spatial Databases

Let *DB* be a spatial database of maps each corresponding to a distinct region. For every map there is a relation-based representation (2D string, symbolic array, a relational table or a set of binary predicates) that stores the relations between all pairs of objects in the region. The objects in the map can be either points or regions but not both (the regions that contain points are called *leaf regions*). Each pair of objects in a map is related by a primitive direction relation explicitly represented. The relation of each point with itself is SP and the projection relation of each region with itself is $P_{7\text{-}7}$. Converse pairs of objects are related by converse relations.

The hierarchy is represented by pointers to next-level areas (IN relation). $DB \vdash IN(X_i,A_j)$ denotes that object (point or region) X_i is a part of (therefore, totally contained in) the next level region A_j. IN* is the transitive closure of IN. $DB \vdash IN^*(X_i,A_j)$ denotes that the relation IN* between objects X_i and A_j is satisfied in the database: $DB \vdash IN^*(X_i,A_j) \equiv DB \vdash IN(X_i,A_j) \vee \exists A_k\ [DB \vdash IN^*(X_i,A_k) \wedge DB \vdash IN^*(A_k,A_j)]$. Object X_i is IN* A_j in database *DB* if: X_i belongs to the next-level region A_j, or, there is a region A_k such that X_i is IN* A_k and A_k is IN* A_j. IN* is not explicitly represented, but is computed by the function *mark* that traverses the hierarchy bottom-up in a depth first manner, and marks all the ancestors of a point in the hierarchy:

```
function IN*(P,A_i)
DB├ IN*(P,A_i)=True;
for each region A_k such that DB├ IN(A_i,A_k)
        if not (DB├ IN*(P,A_k)) then IN*(P,A_k)
end-for

function Mark(P)
for each region A such that DB├ IN(P,A)
        IN*(P,A);
```

For demonstration, we use the example of Figure 3a. At the lower level (level 2) we have four points (cities) that belong to three regions (Greece, Balkan Peninsula and U.K.) which are parts of a top region (Europe). Figure 3b illustrates the information that is actually stored in the database (we omit the relations between the irrelevant European regions, the relations SP for points, the projection $P_{7\text{-}7}$ for regions, and the converse relations for points and regions).

level 0 - top region

Europe ├ $P_{1\text{-}1}$ (UK, Balkan Peninsula)
Europe ├ $P_{1\text{-}1}$ (UK, Greece)
Europe ├ $P_{11\text{-}11}$ (Greece, Balkan Peninsula)
additional relations between the regions of Europe .....

level 1 - leaf regions

DB ├ IN(Greece, Europe)
Greece ├ NW(Athens,Heraklion)

DB ├ IN(Balkan_Peninsula, Europe)
Balkan_Peninsula ├ NW(Belgrade,Athens)

DB ├ IN(UK, Europe)

level 2 - points

DB ├ IN(London, UK)
DB ├ IN(Athens, Balkan_Peninsula)
DB ├ IN(Athens, Greece)
DB ├ IN(Heraklion, Greece)

Figure 3 A geographic example

R, R_1, R_2 ... denote relation variables between points, and r, r_1, r_2 ... between regions. The general problem is to retrieve, the explicit, or implicit, relation between any pair of points P_i and P_j in the database: $DB \vdash R(P_i,P_j)$. There are three cases regarding the direction relations between points. The first case is *explicit retrieval*, that is, there is a region A, such that A contains P_i and P_j: $\exists A\ (A \vdash R(P_i,P_j)) \Rightarrow DB \vdash R(P_i,P_j)$. The relation between two points in the database is R, if the points are related by R in some region. In the example of Figure 3, the relation NW between Belgrade and Athens is explicitly represented in the Balkan Peninsula. Inconsistencies during explicit retrieval arise when P_i and P_j exist together in multiple maps and their relations in these maps are different (e.g., $A_1 \vdash NW(P_1,P_2)$ and $A_2 \vdash NE(P_1,P_2)$). The following algorithm performs explicit retrieval by retrieving all leaf region and examining the relations between all pairs of points in them[1].

```
Explicit_retrieval

// initialization
for each point Pi
        for each point Pj
                if i=j then R(Pi,Pj)= SP;
                        else R(Pi,Pj)=U;
        end-for
end-for

// retrieval from leaf regions
for each (leaf) region Ak
        retrieve Ak;
        for each point Pi such that DB├ IN(Pi,Ak)
                        for each point Pj such that DB├ IN(Pj,Ak) and i<j
                                get the relation R' : Ak ├ R'(Pi,Pj);
                                R(Pi,Pj)= R(Pi,Pj)∩R';
                                if R=∅ then return INCONSISTENCY DUE TO EXPLICIT RETRIEVAL;
                                        else R(Pj,Pi)=converse(R(Pi,Pj));
                        end-for
        end-for
end-for
```

In the second case, *inference through regions*, P_i and P_j do not exist in the same region, but their relation can be inferred using the relations between their ancestor regions. The notation $r \rightarrow R$ means that when the relation r holds true between two regions, then the relation R holds between all pairs of points in the regions. For example, $P_{1\text{-}1} \rightarrow$ NW, since if two regions (e.g., U.K., Greece) are related by projection relation $P_{1\text{-}1}$, the relation between any two points (e.g., London, Athens), each belonging to one region, is NW. Inference through regions can be described as: $[\exists A_k\ \exists A_l\ (DB \vdash IN^*(P_i,A_k) \wedge DB \vdash IN^*(P_j,A_l) \wedge DB$

---

[1] All the algorithms assume that information in each region is arc consistent: $A \vdash R(P_i, P_j) \Leftrightarrow A \vdash converse(R(P_j, P_i))$ and work only on the pairs (P_i, P_j) for which i<j.

$\models$ r(A_k,A_l)) ∧ (r → R)] ⇒ *DB* $\models$ R(P_i,P_j). That is, the relation between P_i and P_j is R, if: P_i and P_j are IN* regions A_k and A_l which are related by projection relation r, and r implies the relation R for all pairs of points in A_k and A_l. Inconsistencies during inference through regions arise when the relation between P_i and P_j in some map is not consistent with the relation between some of their ancestors regions. As an example consider: A $\models$ RN(P_1,P_2) and DB $\models$ IN*(P_1,A_1) and DB $\models$ IN*(P_2,A_2) and DB $\models$ P_{1-1}(A_1,A_2). This is an inconsistency because P_{1-1}→ NW (and not RN).

In the third case (*inference through points*), the relation between P_i and P_j that belong to different maps is inferred by a chain of common points using composition[2] of spatial relations: [∃P (*DB* $\models$ R_k(P_i,P) ∧ *DB* $\models$ R_l(P, P_j)) ∧ (R_k ∗ R_l = R)] ⇒ *DB* $\models$ R(P_i,P_j). That is, *DB* satisfies the direction relation R between P_i and P_j if: there is a point P such that the relation R_k between P_i and P, and the relation R_l between P and P_j is satisfied in the database, and there is a composition rule R_k∗ R_l = R. The relation NW between Heraklion and Belgrade can be inferred because: Balkan_Peninsula $\models$ NW(Belgrade, Athens) and Greece $\models$ NW(Athens, Heraklion), and NW*NW=NW. Inconsistencies in this case arise when different relations are inferred by different chains of points, or when the inferred relation contradicts the results of explicit retrieval or inference through regions (e.g., A_1 $\models$ NW(P_1,P) and A_2 $\models$ NW(P,P_2) and A_3 $\models$ RS(P_1,P_2)).

Unlike explicit retrieval which is straightforward, the other two cases require inference mechanisms that potentially search large parts of the database. In the next sections we discuss algorithms that extract the relation between all pairs of points and detect inconsistencies. For each case we provide rules of inference, we describe extensive examples, and we obtain formulas for the cost.

3. HIERARCHICAL SPATIAL INFERENCE THROUGH REGIONS

This type of inference usually provides good results with minimum computational overhead under the condition that the points belong to "ancestor" regions with non-overlapping projections on at least one axis. The same result is not always achievable using inference through common points (even if such common points exist). As an example consider the configuration of Figure 4a. Figure 4b illustrates the hierarchical structure and the explicit relations for the objects that co-exist in some map. Inference through the chain P_1, P_2, P_3, and P_4 yields the universal relation between P_1 and P_4. On the other hand, the projection relation P_{1-13} between A_1 and A_4 yields the relation NE. Therefore, in some case inference through regions provides more accurate results than inference through points (while some other times the opposite happens).

---

[2]The problem of composition can be defined as "if the spatial relation between P_i and P, and between P and P_j is known, what are the possible relations between P_i and P_j?". The symbol * denotes *path composition* (Frank, 1992): R_1 ∗ R_2 = R means that (R_1 (X,Z) ∧R_2 (Z,Y)) ⇒ R(X,Y)

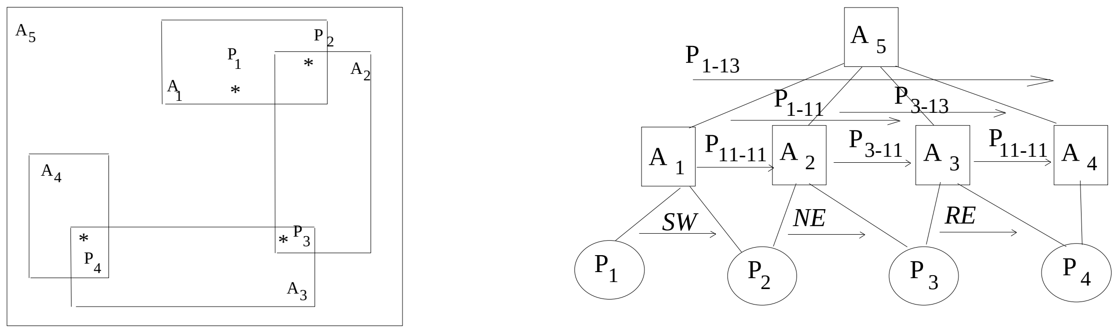


Figure 4 Inference through regions that yields better results than inference through common points

3.1 Rules of Inference

In the case that the projections of two regions are disjoint on both axes (projections P_{1-1}, P_{1-13}, P_{13-13}, and P_{13-1} in Figure 2), then high resolution information can be inferred for both south-north and west-east directions. However, not all projections allow such inferences regarding the relations between points. When the projections of the regions are disjoint on only one axis, low resolution relations about this axis can be derived, but information on the other axis is lost. For example, all cities of Germany are *North* of all cities of Italy (Figure 5a). No conclusion can be drawn about the relation of the cities on the x axis: a city in Germany can be *NE*, *RN* or *NW* of a city in Italy.

In the case that the projections on some axis are not disjoint, but meet at the boundary then some information can still be inferred on this projection. In Figure 5b we can infer that all cities of the U.K are *N* of or *SL* of all cities of France. If regions overlap on both projections, no direction relation can be inferred between the points of the regions (Figure 5c). Depending on the shapes of the regions, any relation between points may be allowable.

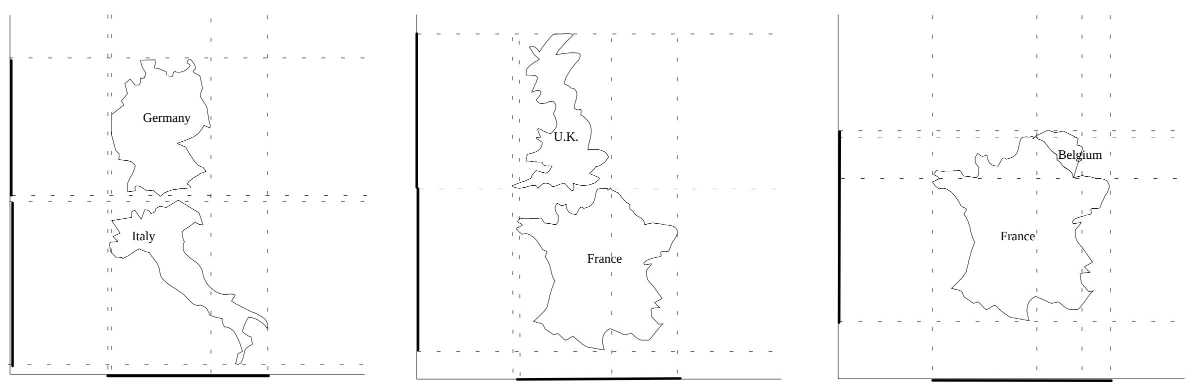


Figure 5 Non disjoint projections

Figure 6 summarises the relations that can be derived about points given the projection relation between regions. High resolution relations can only be inferred in the case that we have disjoint projections on both axes. Non-overlapping projections on one axis imply low resolution relations regarding this axis. Projections that meet, imply relations such a $N \vee SL$, or $NW \vee RN$. The entries of

Figure 6 with *U*, correspond to overlapping projections on both axes (therefore no conclusion can be drawn about the relations between points).

P	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**
1	NW	NW∨RN	N	N	N	N	N	N	N	N	N	NE∨RN	NE
2	NW∨RW	NW∨RN∨RW∨SP	N∨SL	N∨SL	N∨SL	N∨SL	N∨SL	N∨SL	N∨SL	N∨SL	N∨SL	NE∨RN∨RE∨SP	NE∨RE
3	W	W∨SH	U	U	U	U	U	U	U	U	U	E∨SH	E
4	W	W∨SH	U	U	U	U	U	U	U	U	U	E∨SH	E
5	W	W∨SH	U	U	U	U	U	U	U	U	U	E∨SH	E
6	W	W∨SH	U	U	U	U	U	U	U	U	U	E∨SH	E
7	W	W∨SH	U	U	U	U	U	U	U	U	U	E∨SH	E
8	W	W∨SH	U	U	U	U	U	U	U	U	U	E∨SH	E
9	W	W∨SH	U	U	U	U	U	U	U	U	U	E∨SH	E
10	W	W∨SH	U	U	U	U	U	U	U	U	U	E∨SH	E
11	W	W∨SH	U	U	U	U	U	U	U	U	U	E∨SH	E
12	SW∨RW	SW∨RS∨RW∨SP	S∨SL	S∨SL	S∨SL	S∨SL	S∨SL	S∨SL	S∨SL	S∨SL	S∨SL	SE∨RS∨RE∨SP	SE∨RE
13	SW	SW∨RS	S	S	S	S	S	S	S	S	S	SE∨RS	SE

Figure 6 Direction information conveyed by projections

In general, there is significant information loss in this type of inference when there is a large number of objects with overlapping projections. However, experiments with spatial access methods for Geographic Information Systems have shown that for usual values of data density (sum of all region areas divided by the area of the global space) the vast majority of projections are disjoint (Papadias and Theodoridis, to appear). For verification, we created 10,000 regions of various sizes, randomly distributed over the global space. The percentage of regions that have disjoint projections on both axes with respect to a reference region object varied from 99.9% to 99.5%. Similar numbers are produced by real geographic data sets used as standard benchmarks for databases (see Faloutsos and Kamel, 1994).

The above observations refer to "flat" representations; the hierarchical organization in multiple levels results in increased information loss (for a discussion see Section 5). Nevertheless, chances are that inference through regions will produce a high resolution relation. Traditional spatial data structures, such as the R-trees (Guttman, 1984), take advantage of this fact for the efficient retrieval of overlap queries by hierarchical decomposition of space. However, spatial data structures assume that global coordinates are stored, and cannot be used when only the relative positions of objects in the same spatial entity are known (as happens here).

3.2 The Algorithm

Initially the relation between any pair of points is given by explicit retrieval. Inference through regions starts with the procedure *mark* which traverses the hierarchy bottom-up and computes the IN* relation for all points. Then the algorithm retrieves one by one all non-leaf regions A and gets the relation r: A ├

$r(A_k,A_l)$ for all pairs of regions IN A. Let R' be the relation implied by r: r→R' (according to the rules of Figure 6). If R' ≠ U the relation between all pairs of points P_i such that DB├IN*(P_i,A_k), and P_j such that DB├IN*(P_j,A_l) is updated to $R(P_i,P_j)=R(P_i,P_j)\cap R'$.

```
Inference_through_regions

// Bottom up traversal - marking of nodes
for each point Pi
               mark(Pi);

// Extraction of relations
for each non-leaf region A
  retrieve A;
  for each region Ak such that DB├IN(Ak,A)
     for each region Al such that DB├IN(Al,A) and i<j
        get the relation r : A ├ r(Ak,Al);
        lookup R’: r → R’;
        if R’≠ U then
           for each point Pi such that DB├IN*(Pi,Ak)
             for each point Pj such that DB├IN*(Pj,Al)
               R(Pi,Pj)=R(Pi,Pj)∩R’ ;
               if R(Pi,Pj)=∅ then return INCONSISTENCY DUE TO INFERENCE THROUGH REGIONS;
                              else R(Pj,Pi)=converse(R(Pi,Pj));
             end-for
           end-for
     end-for
  end-for
end-for
```

In order to demonstrate how the algorithm works, we use the configuration of Figure 7. There are five points and eight regions organized hierarchically. Each object belongs to the next level region that fully contains it.

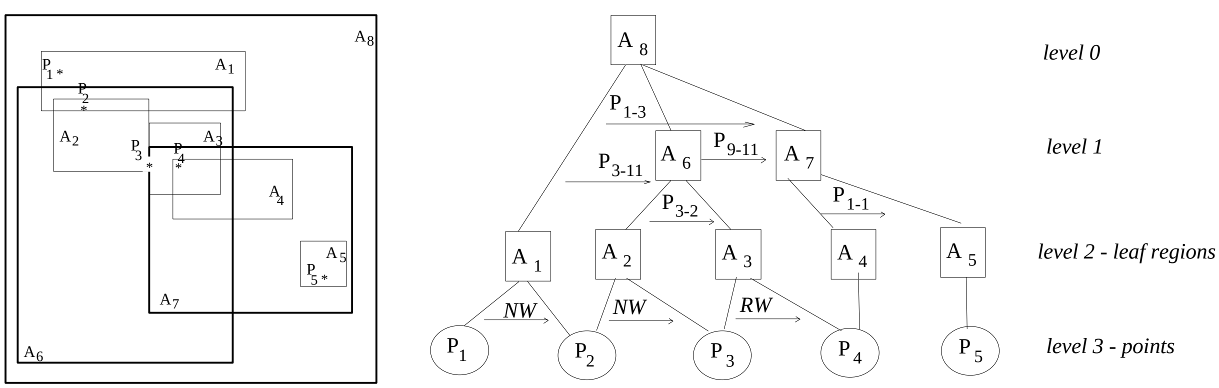


Figure 7 Example of inference through regions

The initial relation between each pair of points is given by explicit retrieval. Figure 8 illustrates the explicit relations between all pairs of points in the form of a constraint network. First A_6 is retrieved and the relation between A_2 and A_3 is found to be $P_{3\text{-}2}$. Since $P_{3\text{-}2} \rightarrow W \vee SH$, the relation between P_2 (that belongs to A_2) and P_4 (that belongs to A_3) is refined to $U \cap (W \vee SH) = W \vee SH$ (Figure 8b). The relation between P_3 and the other points of A_2 and A_3 remains unchanged because $NW \cap (W \vee SH) = NW$ and $RW \cap (W \vee SH) = RW$ (for (P_2,P_3) and (P_3,P_4) respectively). Then A_7 is retrieved and the relation NW between P_4 and P_5 is inferred because the ancestor regions of the two points (A_4 and A_5) are related by $P_{1\text{-}1}$ and, $P_{1\text{-}1} \rightarrow NW$ (Figure 8c). After the retrieval of A_8 (the last non-leaf region) the network takes its final form of Figure 8d. Because $A_8 \vDash P_{1\text{-}3}(A_1,A_7)$, and $P_{1\text{-}3} \rightarrow N$, the relation North is inferred between all points of A_1 and the ones IN* A_7, resulting in $N(P_1,P_4)$, $N(P_1,P_5)$, $N(P_2,P_5)$ and $NW(P_2,P_4) \vee RN(P_2,P_4)$ (the last relation is obtained by $N \cap (W \vee SH)$). The relations $P_{3\text{-}11}(A_1,A_6)$ and $P_{9\text{-}11}(A_6,A_7)$ do not allow any inferences because $P_{3\text{-}11} \rightarrow U$ and $P_{9\text{-}11} \rightarrow U$.

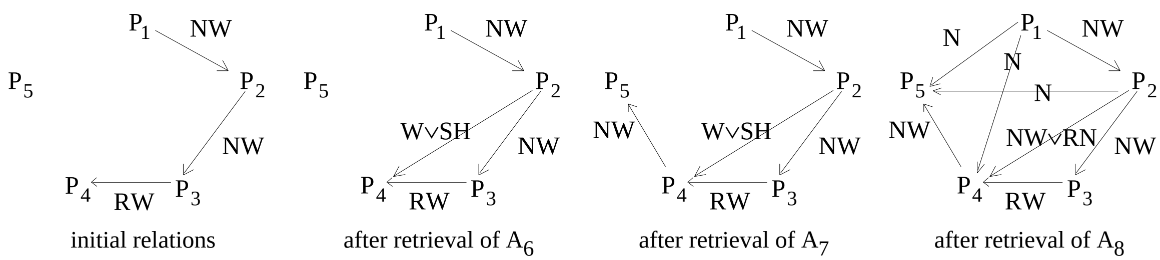


Figure 8 Illustration of the algorithm

Since the algorithm generates the permitted relations for all pairs of points, it needs to be performed only once and its results can be stored for future use. The above algorithm produces fast (an analysis is given later in the paper) and high resolution relations in many situations. However, in cases where we have overlapping projections with multiple common points (as in the example of Figure 7) further refinements are possible by using the common points.

4. Hierarchical Spatial Inference Through Points

Inference using common points, can be formulated as path consistency problem in a network of binary direction constraints. Each constraint in the network is a disjunction of primitive relations and represents the permitted relations between a pair of points after explicit retrieval and inference through regions have taken place (e.g., Figure 8d). Path consistency uses the relative positions of common points to derive the relation between any two points as they are implied by the given constraints. Inference is achieved by excluding relations that cause inconsistencies and maintaining only the ones that could participate in a solution of the network.

4.1 Rules of Inference

In order to apply some path consistency algorithm we need a set of composition rules for direction relations. Figure 9 describes the rules that are applied in order to produce the possible direction relations between P_i and P_j when their relation with respect to a third point P is known. Frank (in press) describes composition of direction relations based on the concepts of *projections* and *cone-shaped directions*. Unlike Frank who uses the notion of *Euclidean approximate* to deal with uncertainty, our system generates a disjunction of the potential primitive relations (which are expressed by the low resolution relations).

	NW (P,P_j)	**RN (P,P_j)**	**NE (P,P_j)**	**RW (P,P_j)**	**SP (P,P_j)**	**RE (P,P_j)**	**SW (P,P_j)**	**RS (P,P_j)**	**SE (P,P_j)**	**N (P,P_j)**	**E (P,P_j)**	**S (P,P_j)**	**W (P,P_j)**	**SL (P,P_j)**	**SH (P,P_j)**
NW(P_i,P)	NW	NW	N	NW	NW	N	W	W	U	N	U	U	W	N	W
RN(P_i,P)	NW	RN	NE	NW	RN	NE	W	SH	E	N	E	U	W	N	SH
NE(P_i,P)	N	NE	NE	N	NE	NE	U	E	E	N	E	U	U	N	E
RW(P_i,P)	NW	NW	N	RW	RW	SL	SW	SW	S	N	U	S	W	SL	W
SP(P_i,P)	NW	RN	NE	RW	SP	RE	SW	RS	SE	N	E	S	W	SL	SH
RE(P_i,P)	N	NE	NE	SL	RE	RE	S	SE	SE	N	E	S	U	SL	E
SW(P_i,P)	W	W	U	SW	SW	S	SW	SW	S	U	U	S	W	S	W
RS(P_i,P)	W	SH	E	SW	RS	SE	SW	RS	SE	U	E	S	W	S	SH
SE(P_i,P)	U	E	E	S	SE	SE	S	SE	SE	U	E	S	U	S	E
N(P_i,P)	N	N	N	N	N	N	U	U	U	N	U	U	U	N	U
E(P_i,P)	U	E	E	U	E	E	U	E	E	U	E	U	U	U	E
S(P_i,P)	U	U	U	S	S	S	S	S	S	U	U	S	U	S	U
W(P_i,P)	W	W	U	W	W	U	W	W	U	U	U	U	W	U	W
SL(P_i,P)	N	N	N	SL	SL	SL	S	S	S	N	U	S	U	SL	U
SH(P_i,P)	W	U	E	W	SH	E	W	SH	E	U	E	U	W	U	SH

Figure 9 Composition table for low and high resolution relations

The composition constraint $R_k * R_l$ is computed by forming the cross products of the primitive constraints that comprise R_k and R_l, composing each resulting ordered pair by looking up the results in the composition table, and taking the union of the resulting sets. Besides the primitive relations, the table of Figure 9 contains the low resolution relations (U is not included, because the composition of U with any relation is U). Notice that this set of relations is closed under composition; in a network where all the initial constraints belong to this set, the final constraint (after path consistency) between each pair of objects is also a high, or low resolution relation, or U.

In addition to the above 16 relations, the type of networks that result after inference through regions, may contain another 16 relations such as NW∨RN (see Figure 6). Composition is also closed under all 32 relations. Figure 10 illustrates the compositions of the five new relations at the upper-left corner of the table of Figure 6 with all relations. The remaining relations of Figure 6 produce the symmetrical relations on the corresponding axes. The result of composition is always one of the 32 relations[3]; therefore, the constraint between each pair of points in the network is always one of the above relations and arbitrary disjunctions do not appear at any phase of inference through points. As we discuss later, this fact is important for the consistency of the relations in the final network, and for the cost of execution.

	$NW(P,P_j) \vee RN(P,P_j)$	$NW(P,P_j) \vee RW(P,P_j)$	$NW(P,P_j) \vee RN(P,P_j) \vee RW(P,P_j) \vee SP(P,P_j)$	$N(P,P_j) \vee SL(P,P_j)$	$W(P,P_j) \vee SH(P,P_j)$
$NW(P_i,P)$	NW	NW	NW	N	W
$RN(P_i,P)$	NW∨RN	NW	NW∨RN	N	W∨SH
$NE(P_i,P)$	N	N	N	N	U
$RW(P_i,P)$	NW	NW∨RW	NW∨RW	N∨SL	W
$SP(P_i,P)$	NW∨RN	NW∨RW	NW∨RN∨RW∨SP	N∨SL	W∨SH
$RE(P_i,P)$	N	N∨SL	N∨SL	N∨SL	U
$SW(P_i,P)$	W	W	W	U	W
$RS(P_i,P)$	W∨SH	W	W∨SH	U	W∨SH
$SE(P_i,P)$	U	U	U	U	U
$N(P_i,P)$	N	N	N	N	U
$E(P_i,P)$	U	U	U	U	U
$S(P_i,P)$	U	U	U	U	U
$W(P_i,P)$	W	W	W	U	W
$SL(P_i,P)$	N	N∨SL	N∨SL	N∨SL	U
$SH(P_i,P)$	W∨SH	W	W∨SH	U	W∨SH
$NW(P_i,P) \vee RN(P_i,P)$	NW∨RN	NW	NW∨RN	N	W∨SH
$NW(P_i,P) \vee RW(P_i,P)$	NW	NW∨RW	NW∨ RW	N∨SL	W
$NW(P_i,P) \vee RN(P_i,P) \vee RW(P_i,P) \vee SP(P_i,P)$	NW∨RN	NW∨RW	NW∨RN∨RW∨SP	N∨SL	W
$N(P_i,P) \vee SL(P_i,P)$	N	N∨SL	N∨SL	N∨SL	U
$W(P_i,P) \vee SH(P_i,P)$	W∨SH	W	W∨SH	U	W∨SH
$SW(P_i,P) \vee RW(P_i,P)$	W	W	W	U	W
$SW(P_i,P) \vee RS(P_i,P)$	W∨SH	W	W∨SH	U	W∨SH
$SW(P_i,P) \vee RS(P_i,P) \vee RW(P_i,P) \vee SP(P_i,P)$	W∨SH	W	W∨SH	U	W
$S(P_i,P) \vee SL(P_i,P)$	U	U	U	U	U
$SE(P_i,P) \vee RS(P_i,P)$	U	U	U	U	U
$SE(P_i,P) \vee RE(P_i,P)$	U	U	U	U	U
$SE(P_i,P) \vee RS(P_i,P) \vee RE(P_i,P) \vee SP(P_i,P)$	U	U	U	U	U
$E(P_i,P) \vee SH(P_i,P)$	U	U	U	U	U
$NE(P_i,P) \vee RE(P_i,P)$	N	N∨SL	N∨SL	N∨SL	U
$NE(P_i,P) \vee RN(P_i,P)$	N	N	N	N	U
$NE(P_i,P) \vee RE(P_i,P) \vee RN(P_i,P) \vee SP(P_i,P)$	N	N∨SL	N∨SL	N∨SL	U

Figure 10 Composition table for relations generated after inference through regions

4.2 The Algorithm

A number of path consistency algorithms have been proposed (Allen, 1983; Macworth and Freuder, 1985). The following one is a variation modified for the current problem. Initially the network is derived

---

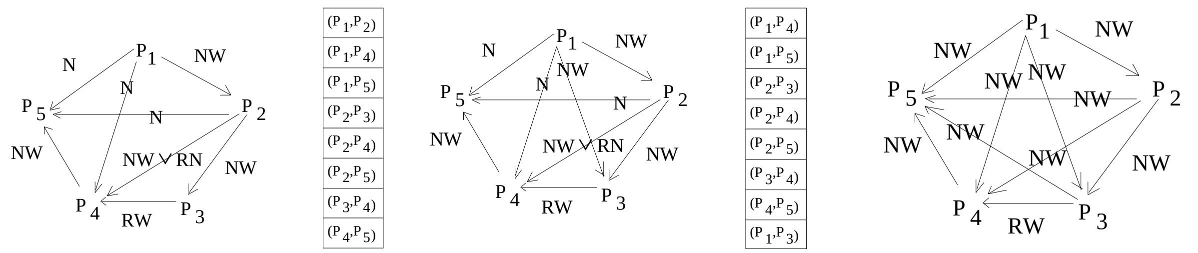
[3] Sharma (1995), makes similar observations regarding the closure of compositions in 2D space.

from explicit retrieval and inference through regions. All pairs of points whose relation is not U are inserted into a queue. Then every pair is popped from the queue and the corresponding relation is used to refine the relation between the popped points and all the other points that co-exist with them in some region. The pairs of points whose relation is refined are pushed in the queue for propagation of the update through the network.

```
Inference_through_points

for each point P_i
        for each point P_j such that i<j
                        if R(P_i,P_j)≠U then push-queue(P_i,P_j);
while not-empty-queue
    pop-queue(P_i,P_j);
        for each (leaf) region A_l such that DB├ IN(P_i,A_l)
                retrieve A_l;
                for each point P_k such that DB├ IN(P_k,A_l) and k ≠ i and k ≠ j
                        R_t(P_k,P_j)=R(P_k,P_j) ∩ (R(P_k,P_i)*R(P_i,P_j));
                        if R_t=∅ then return INCONSISTENCY DUE TO PATH CONSISTENCY;
                        else if R_t(P_k,P_j)⊂ R(P_k,P_j) then
                                        R(P_k,P_j)= R_t(P_k,P_j);
                                        R(P_j,P_k)=converse(R(P_k,P_j));
                                        if not in-queue(P_k,P_j) then push-queue(P_k,P_j);
                end-for
        end-for
        for each (leaf) region A_m such that DB├ IN(P_j,A_m)
                retrieve A_m;
                for each point P_k such that DB├ IN(P_k,A_m) and k ≠ i and k ≠ i
                        R_t(P_i,P_k)=R(P_i,P_k) ∩ (R(P_i,P_j)* R(P_j,P_k));
                        if R_t=∅ then return INCONSISTENCY DUE TO PATH CONSISTENCY;
                        else if R_t(P_i,P_k)⊂ R(P_i,P_k) then
                                        R(P_i,P_k)= R_t(P_i,P_k);
                                        R(P_k,P_i)=converse(R(P_i,P_k));
                                        if not in-queue(P_i,P_k) then push-queue(P_i,P_k);
                end-for
        end-for
end-while
```

In order to demonstrate the algorithm, we use the configuration of Figure 7 and the network of Figure 8d. After explicit retrieval and inference through regions have been applied, the pairs of points whose relation is not U are pushed into a queue. Here we assume the order of Figure 11a, but the order is not important. First the pair (P_1,P_2) is popped and all the regions that contain these points are retrieved. P_3 co-exists with P_2 in region A_2 and its relation with P_1 is updated according to: $R(P_1,P_3)=R(P_1,P_3)\cap(R(P_1,P_2)*R(P_2,P_3))=U\cap(NW*NW)=NW$. Because the new relation is a refinement of the previous one (NW⊂U) the pair (P_1,P_3) is pushed into the queue for propagation. The new network and the state of the queue at this phase are illustrated in Figure 11b. Then the pair (P_1,P_4) is popped from the queue, the regions A_1, A_3, and A_4 are retrieved, and the relations between the points (P_2,P_4), and (P_1,P_3) are updated. However the network does not change at this stage because: $R(P_2,P_4) = R(P_2,P_4) \cap (R(P_2,P_1)*R(P_1,P_4)) = (NW \vee RN) \cap (SE*N) = NW \vee RN$, and $R(P_1,P_3) = R(P_1,P_3) \cap (R(P_1,P_4)*R(P_4,P_3)) = NW \cap (N*RE)=NW$. Similarly the pair ($P_1$,$P_5$) will not alter the network, while the pair (P_2,P_3) will produce: $R(P_2,P_4)$ =NW. The remaining pairs update the network in the same fashion; the final state after the termination of the algorithm is illustrated in Figure 11c.



initial network and queue after (P_1,P_2) has been popped final network

Figure 11 Illustration of the algorithm

Path consistency refines the constraints between each pair of points by pruning from the constraints the relations that cause inconsistencies (relations that are not consistent with the explicit relations between some other pairs). However, path consistency does not achieve global consistency (does not remove all inconsistencies) from general constraint networks[4]. Van Beek and Cohen (1991) have proven that any path consistent, point algebra network that contains inconsistent relations, has a subgraph of four vertices isomorphic to the network of Figure 12a. This network is path consistent because every primitive relation that appears in a constraint participates in at least one solution of each triangle (Figures 12b-12e). Still, the relation SL (*SameLevel*) between P_1 and P_4 causes inconsistency because it will enforce SL between P_2 and P_3, which is not allowed by the initial constraints. Van Beek (1992) demonstrates that such problems are created in networks that contain inequality (N or S but not SL - in our context North can be substituted by >, SameLevel by = and, South by <).

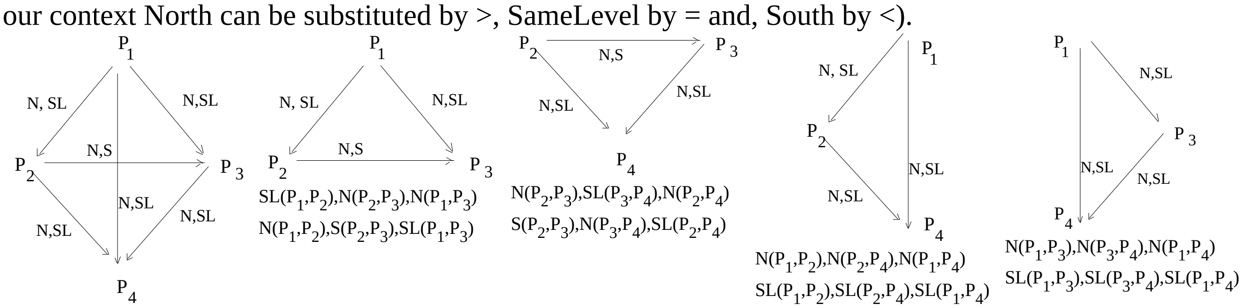


Figure 12 Path consistent spatial constraint network with inconsistent relations

[4] Constraint satisfaction problems are in general exponential in nature, while path consistency is polynomial. Grigni et al., (1995) have proven that constraint satisfaction in networks of topological relations is NP-Complete, and realizability in space is NP-Hard.

Nevertheless, in the type of problem we study here, we start with the set of constraints imposed after inference through regions, and not with arbitrary disjunctions. This set is closed under composition. Therefore, path consistency does not produce inequality (e.g., North $\vee$ South) on any axis, and the network does not contain inconsistent relations after the application of the above algorithm.

5. On The Efficiency of Hierarchical Representations

In this section we discuss a unified framework for hierarchical spatial inference, we describe the computational complexity of the algorithms and we study the efficiency of hierarchical representations.

5.1 A Unified Framework for Hierarchical Spatial Inference

In the previous sections we demonstrated three different ways of generating the relations between pairs of points. We argued that first explicit retrieval obtains the relations between pairs of points that exist in the same region, then inference through regions generates additional constraints imposed by the relations between the ancestor regions, and finally inference through points takes advantage of common points to produce further refinements. The order in which explicit retrieval and inference though regions are performed is not important. As long as the content of the database remains unaltered they will generate the same result independently on which is performed first. On the other hand, inference through points has always to be performed at the end, otherwise it may not produce all relations.

Assume, for example, that path consistency is applied before inference through regions to the configuration of Figure 7. The explicit relations are illustrated in Figure 13a, and the path consistent network in Figure 13b. The subsequent application of inference through regions will refine some relations (in particular the relations between P_5 and P_1, P_2 and P_4) resulting in the network of Figure 13c which lacks some relations with respect to the network of Figure 11c (e.g., the relation between P_5 and P_3 is *U*). The problem is created by isolated regions (regions, such as A_5, that do not contain common points with other regions). Inference through points has to be applied again in order propagate the new relations and generate the network of Figure 11c.

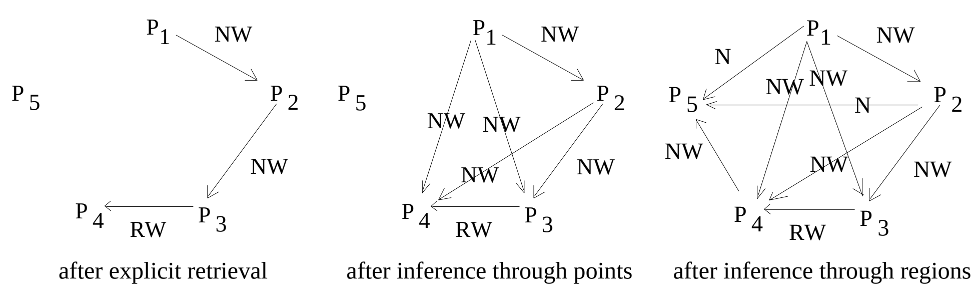


Figure 13 Permutation of the functions

Figure 14 illustrates a unified framework for inference and inconsistency checking in spatial databases of points and objects involving the previous mechanisms. Either explicit retrieval, or inference through regions can be applied after the initialization (which is included as part of the explicit retrieval in Section 2 but can be an independent function). Inference through points should be the final phase. This framework explicates the relations between all pairs of points and its results can be stored and used to answer future queries involving direction relations. It should be executed either when there is an update in the database, at periodical time intervals as batch processes, or after the number of modifications in the database becomes larger than a specified threshold.

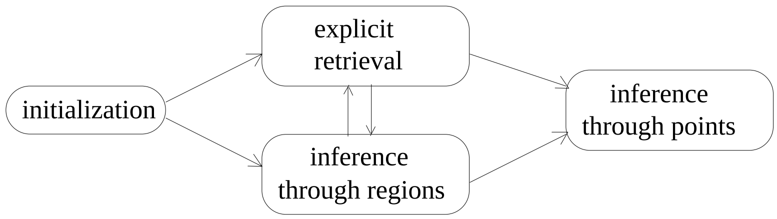


Figure 14 A Unified framework

5.2 Cost of Inference

In order to obtain formulas for the cost of the algorithms we make the following simplifications (although such simplifications may not apply for real applications, they provide a good measure for the expected cost in most cases). Each region contains *k* objects (points or other regions). Each object belongs to *m* regions in the upper hierarchy level, except for the region at the top (0 level) that does not belong to any region, and the objects at level 1 that belong only to the top-level region. It is always the case that $k/m>1$ and in regular applications $k/m>>1$. *N* is the total number of points in the database. We assume that there is a buffer that stores the $N(N-1)/2$ relations between all pairs of points.

For demonstration we use Figure 15, where $k=4$ and $m=2$. The objects are represented as nodes in a hierarchy of height *h*. For each object (except for the ones at levels 0 and 1) there are *m* copies (illustrated as overlapping nodes), each corresponding to an instance of the object in a parent node (this data replication also exists in the database because each object is represented in all parent regions).

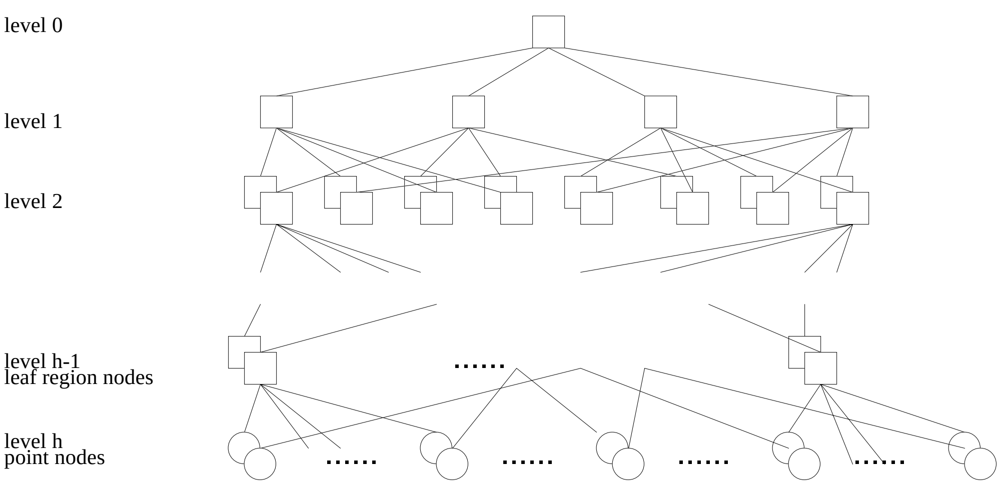


Figure 15 Hierarchical structure for k=4 and m=2

The cost is a function of the number of map retrievals because such operations require access to secondary storage (i.e., retrieval of the disk pages that contain the map). This is common practice in database literature where indexing methods are compared on the number of accessed pages from the disk (Guttman 1984; Faloutsos and Kamel, 1994). In the case of explicit retrieval, for example, we have to retrieve all leaf regions. Due to the fact that leaf regions store all points and their copies, their number is *mN/k*. Therefore, explicit retrieval performs *mN/k* map retrievals.

In order to measure the cost of inference through regions we need to calculate the number of non-leaf regions, because all these regions are retrieved. There is only one node at level 0, *k* nodes at level 1, and k^2 at level 2. Out of these k^2 nodes, k^2/m correspond to objects and the rest to copies. Level 3 contains *k* nodes for each original node of the previous level resulting in a total of k^3/m nodes out of which only k^3/m^2 are original and represented at level 4. Similarly, at level h-1 there are k^{h-1}/m^{h-2} nodes that correspond to actual leaf regions. Since the number of leaf regions is *mN/k*, we have the following equation that provides a formula for *h*:

$$\frac{k^{h-1}}{m^{h-2}} = \frac{m \cdot N}{k} \Rightarrow h = \left\lceil \log_{(k\ /\ m)}\left(\frac{N}{m}\right) \right\rceil \qquad (1)$$

The number of non-leaf regions (and therefore the number of map retrievals during inference through regions) is the sum of original regions from level 0 to level h-2. Substituting the height of equation 1 we get the following approximation for the cost of inference though regions:

$$m^2 \frac{N-k}{k(k-m)} \qquad (2)$$

In order to find the cost of inference through points we start with the observation that composition is closed under the initial constraints. A constraint imposed by inference through regions or explicit retrieval may be refined a number of times until it reaches its final state at the end of path consistency. Each time a refinement happens the corresponding pair of points is pushed to the queue. Figure 16 illustrates the possible refinements for the 32 constraints that may appear in the network. A constraint at any level may only be refined to a constraint of a lower level. For example, a constraint between two points may initially be U and become N∨SL, then NW∨RN∨RW∨SP, then NW∨RN and finally NW. The links in Figure 16 connect each constraint with the constraints of the immediately lower level that it can be refined. The maximum number of refinements for any constraint is four.

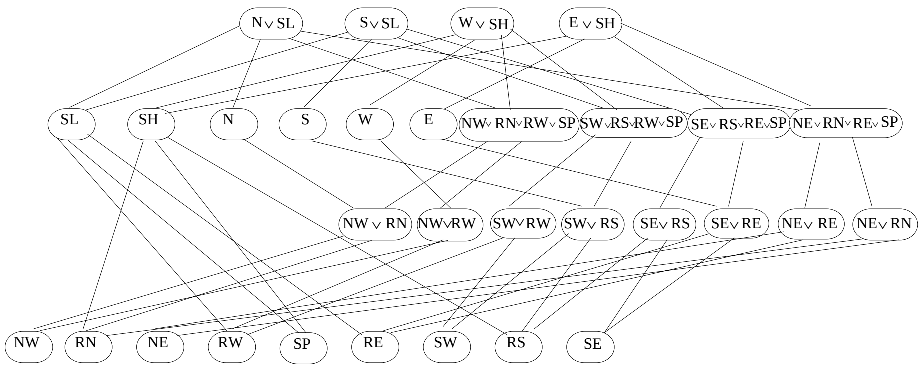


Figure 16 Refinement of direction constraints

There exist *N(N-1)/2* distinct pairs of points in the database and each may be pushed into the queue a maximum of four times. Each time a pair is popped from the queue, *2m* map retrievals are performed to retrieve the points that are related with the popped points in some region. Therefore, inference through points requires *4mN(N-1)* map retrievals in the worst case.

5. 3 Storage Efficiency and Information Loss

The hierarchical representation of space reduces storage requirements significantly because only spatial information within the same region is explicitly represented. All the other relations are extracted by the inference mechanisms. Assume a relational model where the binary relations between all pairs of points are stored as tupples in a large "flat" table, containing a total of *N(N-1)/2* tupples. The hierarchical variation of the same database would contain a number of "small" tables each standing for a distinct region. The number of such tables equals the number of original regions in the hierarchy:

$$1 + k + \ldots + \frac{k^{h-1}}{m^{h-2}} \approx m\frac{N-m}{k-m} \qquad (3)$$

Each small table contains *k(k-1)/2* tupples representing the relations between the objects in the corresponding region. In addition, we need an extra table representing the IN relations. For each point and region there exist *m* pointers (tupples) to their father regions. By adding the relation and the IN tupples we get the total number of tupples of in the hierarchical version.

$$\frac{k(k-1)}{2} \cdot \frac{m(N-m)}{k-m} + m \cdot \left(m\frac{N-m}{k-m} + N \right) \qquad (4)$$

Assume an implementation where the average capacity *k* of each region equals the usual disk page capacity in points (*k=100* points). Each point (or region) belongs to *m=4* regions. The database contains $N=10^6$ points (a moderate number given the sizes of existing spatial databases). Substituting these numbers to equation 4 we get approximately $55*10^6$ tupples for the hierarchical version as opposed to $5*10^{11}$ for the flat database.

In addition to storage efficiency, the hierarchical decomposition of space in natural geographic entities optimizes queries that involve objects within the same entity ("find all major cities of France northeast of Paris"). For such queries only the map(s) corresponding to the entity (i.e., France) needs to be retrieved, while in a flat representation the same query could involve searching the whole database in the worst case. The trade-off that hierarchical representations pay for efficiency is information loss regarding the relations between points that exist in different regions, when such relations cannot be inferred.

Consider the example of Figure 17 where P_1 belongs to A_1 which in turn belongs to A_3 and so on. The grey zones show the areas of information loss, that is, areas that correspond to projections of P_1's ancestor at the current level. A high resolution relation between P_1 and some other point can be inferred only if an ancestor of the second point is disjoint with the corresponding grey zones of its level. When a region overlaps one or more of these zones, information about one or both axes is lost. For example, the inferred relation between P_1 and P_2 is North; no east-west information can be extracted because A_2 overlaps the grey area corresponding to A_1's projection on the x axis. Similarly, no relation can be inferred between P_1 and P_3, because A_3 and A_5 have overlapping projections on both axes (although the leaf regions A_2 and A_4 have disjoint projections their relation is not explicitly represented in some parent region). In general, the information loss increases as the distance of the points in the hierarchy increases.

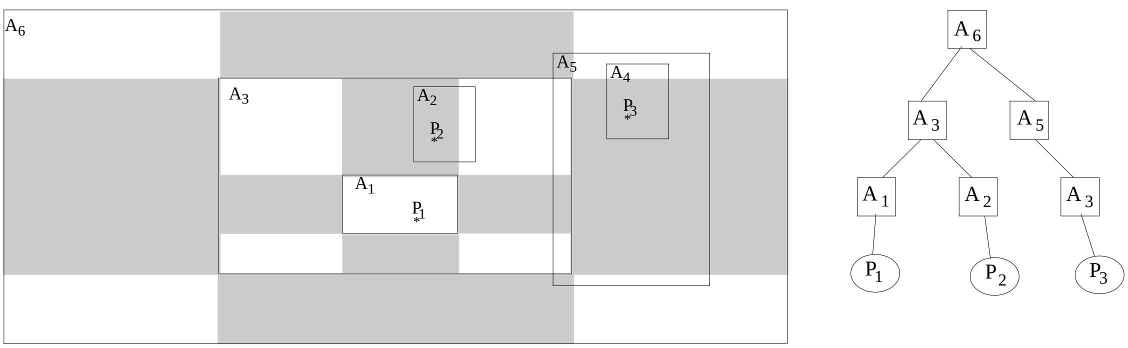


Figure 17 Information loss in hierarchical representations

Using the example of Figure 3, all relations between cities in the same European country are represented but relations between some cities that belong to different countries with overlapping projections are lost. Extending the example, we would not be able to derive the relation between a city in Europe and a city in Asia by inference through regions because Europe and Asia have overlapping projections on both axes. Such relations may be possible only by inference through common points if such points exist. However, in practical applications this information loss is not very important because the vast majority of spatial queries refers to objects within the same geographic entity. Queries of the form "find all cities of France north of Africa" are not common, and if they are imposed, chances are that there is enough information to infer the answer.

6. EXTENSIONS

The algorithms of this paper can be applied to different sets of direction relations under the condition that the inference rules are modified for the specific case. In this section we use as our basis an alternative set of projection-based direction relations, called *direction relations with neutral area* first defined in (Frank, 1992). According to these definitions the restricted relations are not line segments, but areas that extend *d/2* from each side of the reference point (where *d* is determined by the application requirements). SamePosition is a square of side d. Figure 18 illustrates the direction relations with neutral area between points. Such relations are useful in cases that there is uncertainty of location.

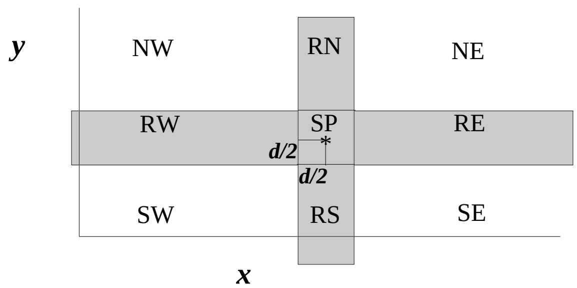


Figure 18 Direction relations with neutral area between points

Projection relations with neutral area can be defined accordingly for regions. There exist 15 (instead of 13) relations on each axis, because there are two new relations for the cases where the primary region is totally contained within *d/2* distance from some edge point. Figure 19 illustrates the first 45 of the 225 possible relations between region projections. Topaloglou (1994) defined similar projections relations to model objects with fuzzy boundaries, an application domain unsuitable for the direction relations of the previous sections.

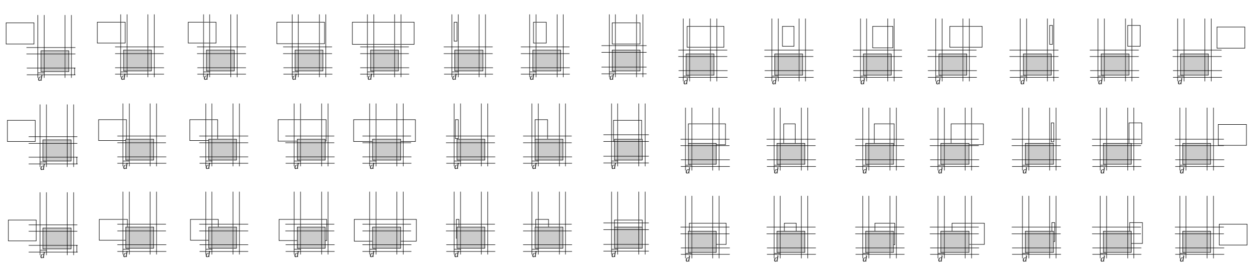


Figure 19 Direction relations with neutral area between regions

Inference through regions can be achieved if we assume the region relations of Section 2 (i.e., relations between points are defined according to Figure 18, and relations between regions according to Figure 2). However, in this case there is significant information loss even for disjoint projections on both axes. For example, $P_{1\text{-}1}$ would imply NW∨RN∨RW∨SP instead of NW. On the other hand, assuming the relations of Figure 19, the table for inference through regions is identical to the one of Figure 6 (with the inclusion of two extra rows and columns for the additional relations). The table for inference through points is significantly different from the one in Figure 9 (e.g., the restricted relations are not transitive anymore). Figure 20 illustrates the composition table for directions with neutral area.

	NW(P,P$_j$)	RN(P,P$_j$)	NE(P,P$_j$)	RW(P,P$_j$)	SP(P,P$_j$)	RE(P,P$_j$)	SW(P,P$_j$)	RS(P,P$_j$)	SE(P,P$_j$)
NW(P$_i$,P)	NW	NW∨RN	N	NW∨RW	NW∨RN∨RW∨SP	N∨SL	W	W∨SH	U
RN(P$_i$,P)	NW∨RN	N	NE∨RN	NW∨RN∨RW∨SP	N∨SL	NE∨RN∨RE∨SP	W∨SH	U	E∨SH
NE(P$_i$,P)	N	NE∨RN	NE	N∨SL	NE∨RN∨RE∨SP	NE∨RE	U	E∨SH	E
RW(P$_i$,P)	NW∨RW	NW∨RN∨RW∨SP	N∨SL	W	W∨SH	U	SW∨RW	RW∨SP∨SW∨RS	S∨SL
SP(P$_i$,P)	NW∨RN∨RW∨SP	N∨SL	NE∨RN∨RE∨SP	W∨SH	U	E∨SH	SW∨RS∨RW∨SP	S∨SL	SE∨RS∨RE∨SP
RE(P$_i$,P)	N∨SL	NE∨RN∨RE∨SP	NE∨RE	U	E∨SH	E	S∨SL	SE∨RS∨RE∨SP	SE∨RE
SW(P$_i$,P)	W	W∨SH	U	SW∨RW	SW∨RS∨RW∨SP	S∨SL	SW	SW∨RS	S
RS(P$_i$,P)	W∨SH	U	E∨SH	RW∨SP∨SW∨RS	S∨SL	SE∨RS∨RE∨SP	SW∨RS	S	SE∨RS
SE(P$_i$,P)	U	E∨SH	E	S∨SL	SE∨RS∨RE∨SP	SE∨RE	S	SE∨RS	SE

Figure 20 Composition table for directions with neutral area

Unlike topological relations where the *intersection model* (Egenhofer and Franzosa, 1991) has become a standard in both research literature and commercial products, there are not widely accepted definitions for direction relations. People have used different types of direction relations to match different needs that range from cognitive modelling (Herskovits, 1986) to image similarity retrieval (Lee et al., 1992) and from robot navigation (Holmes and Jungert, 1992) to user interfaces (Roussopoulos et al., 1988). Although, in this paper, we have used a specific set of relations, the algorithms for hierarchical reasoning can be applied to any set of direction relations with the corresponding inference rules.

7. CONCLUSIONS

The hierarchical representation of space has a strong psychological motivation (Hirtle and Jonides, 1985) and numerous computational advantages that have been exploited in a number of areas such as Data Structures (Guttman, 1984) and Wayfinding (Car and Frank, 1994). In this paper we focus on hierarchical spatial reasoning involving direction relations in 2D space. Although we have dealt with a set of projection-based direction relations often found in the literature, the methods of the paper are not relation-specific. They could be applied to alternative sets of relations with the appropriate rules of inference.

We present two complementary algorithms for spatial inference and inconsistency detection in hierarchically structured spatial databases: (1) the first achieves inference of direction relations between points through their ancestor regions, and, (2) the second performs inference through chains of common points and path consistency. For both algorithms we provide the corresponding inference rules and formulas for their cost. Because the algorithms generate the relations between all pairs of points they don't need to be executed for each individual query, but only after the contents of the database are modified.

Hierarchical representations result in information loss with respect to flat representations. Some relations between points in different regions cannot be derived by inference. On the other hand, they have significant storage advantages and facilitate query processing for queries involving objects within the same entity. Furthermore, in some cases, hierarchical representations are not just an option but a necessity. Even in a single system, data about the same or overlapping areas but from different sources are stored separately. This information may be incomplete or inconsistent, and inference mechanisms are required to explicate relations and remove inconsistencies.

As interoperability issues are solved, heterogeneous spatial databases and open GIS will soon become a reality. Such systems will store huge amounts of spatial data in various formats and of variable quality. Users will query the systems requiring fast and accurate results (and not answers of the form "A is north and south of B"). Spatial inference mechanisms will play an important role for the detection of inconsistencies in the data and the integration of the different systems.

REFERENCES

Allen, J.F. "Maintaining Knowledge about Temporal Intervals". CACM 26 (11), pp. 832-843, 1983.

Car, A., Frank, A. "General Principles of Hierarchical Spatial Reasoning: The Case of Wayfinding". In the Proceedings of the 6th International Symposium on Spatial Data Handling, 1994.

Chang, S.K., Shi, Q.Y. Yan C.W. "Iconic Indexing by 2-D String". IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-9, no 3, pp. 413-428, 1987.

Clementini, E., Sharma, J., Egenhofer, M. "Modeling Topological Spatial Relations: Strategies for Query Processing". International Journal of Computer and Graphics, vol. 18 (6), pp. 815-822, 1994.

Dutta, S. "Qualitative Spatial Reasoning: A Semi-Qualitative Approach using Fuzzy Logic". Proceedings of the First Symposium on the Design and Implementation of Large Spatial Databases 1989.

Egenhofer, M., Franzosa R. (1991) "Point Set Topological Relations". International Journal of Geographic Information Systems, 5, pp. 161-174, 1991.

Egenhofer, M., Sharma, J. "Assessing the Consistency of Complete and Incomplete Topological Information". Geographical Systems, Vol. 1, No 1, pp. 47-68, 1993.

Egenhofer, M., "Spatial SQL: A Query and Presentation Language", IEEE Transactions on Knowledge and Data Engineering, Vol. 6(1), pp. 86-95, 1994.

Faloutsos, C., Kamel, I., "Beyond Uniformity and Independence: Analysis of R-trees Using the Concept of Fractal Dimension", Proceedings of the 13th ACM PODS Symposium on Principles of Database Systems, 1994.

Frank, A.U., “Qualitative Spatial Reasoning about Distances and Directions in Geographic Space”, Journal of Visual Languages and Computing, Vol. 3, pp. 343-371, 1992.

Frank, A. U. “Qualitative Spatial Reasoning: Cardinal Directions as an Example”. To appear in the International Journal of Geographic Information Systems.

Freksa, C. “Using Orientation Information for Qualitative Spatial Reasoning”. Proceedings of the International Conference GIS - From Space to Territory: Theories and Methods of Spatio-Temporal Reasoning in Geographic Space, Pisa, Italy. Springer Verlag LNCS.

Glasgow, J.I. “Spatial Reasoning in Indeterminate Worlds”. Proceedings of the National Conference on Artificial Intelligence, 1994.

Glasgow, J.I., Papadias, D. “Computational Imagery”. Cognitive Science, 16, pp. 355-394, 1992.

Grigni, M., Papadias, D., Papadimitriou, C., “Topological Inference”, Proceedings of the International Joint Conference of Artificial Intelligence, 1995.

Guttman, A., “R-trees: a Dynamic Index Structure for Spatial Searching”, Proceedings of ACM SIGMOD International Conference on Management of Data, 1984.

Hernandez, D., “Qualitative Representation of Spatial Knowledge”, Springer Verlag LNAI, 1994.

Herskovits, A. “Language and Spatial Cognition”. Cambridge University Press, 1986.

Hirtle, S, Jonides, J. “Evidence of Hierarchies in Cognitive Maps”. Memory and Cognition 13, pp. 208-217, 1985.

Holmes, P.D., Jungert, E. “Symbolic and Geometric Connectivity Graph Methods for Route Planning In Digitized Maps”. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-14, no 5, pp. 549-565, 1993

Lee, S-Y., Yang, M-C., Chen, J-W. “Signature File as a Spatial Filter for Iconic Image Database”. Journal of Visual Languages and Computing, 3, pp. 373-397, 1992.

Mackworth, A, Freuder, E. “The Complexity of some Polynomial Network Consistency Algorithms for Constraint Satisfaction Problems”. Artificial Intelligence, 25, pp. 65-74, 1985.

Mukerjee, A., Joe, G., “A Qualitative Model for Space”. Proceedings of the National Conference on Artificial Intelligence, 1991.

Papadias, D., Sellis, T., “Qualitative Representation of Spatial Knowledge in Two-Dimensional Space”, Very Large Data Bases Journal, Special Issue on Spatial Databases, Vol 3(4), pp. 479-516, 1994.

Papadias, D., Sellis, T. “A Pictorial Query-By-Example Language”. Journal of Visual Languages and Computing, Special Issue on Visual Query Systems, Vol. 6, No. 1, 53 -72, 1995.

Papadias, D., Theodoridis, Y., Sellis, T., Egenhofer, M. “Topological Relations in the World of Minimum Bounding Rectangles: a Study with R-trees”. Proceedings of ACM SIGMOD International Conference on Management of Data, 1995.

Papadias, D., Theodoridis, Y. “Spatial Relations, Minimum Bounding Rectangles, and Spatial Data Structures”, To appear in the International Journal of Geographic Information Systems.

Peuquet, D., Ci-Xiang, Z. “An Algorithm to Determine the Directional Relationship between Arbitrarily-Shaped Polygons in the Plane”. Pattern Recognition, Vol 20, No 1, pp. 65-74, 1987

Roussopoulos, N., Faloutsos,C., Sellis, T. “An Efficient Pictorial Database System for PSQL”. IEEE Transactions on Software Engineering, Special issue on Image Databases, pp. 639-650, 1988.

Sharma, J., “Heterogeneous Spatial Reasoning in Geographic Information Systems”. Ph.D. Thesis, Department of Spatial Information Science and Engineering, University of Maine, 1995.

Sistla, P., Yu, C., Haddad, R. “Reasoning about Spatial Relationships in Picture Retrieval Systems”. In the Proceedings of the 20th VLDB Conference, 1994.

Smith, T., Park, K., “Algebraic Approach to Spatial Reasoning”. International Journal of Geographic Information Systems, Vol. 6, No. 3, pp. 177-192, 1992.

Topaloglou, T., “Qualitative Reasoning about Imprecise Spatial information”. Proceedings of the 2nd ACM Workshop on GIS, 1994.

van Beek, P., Cohen, R. “Exact and Approximate Reasoning about Temporal Relations”. Computational Intelligence 6, 132-144, 1990.

van Beek, P. “Reasoning About Qualitative Temporal Information”. Artificial Intelligence, 58, pp. 297-326, 1992.