Computing and Managing Cardinal Direction Relations

Spiros Skiadopoulos, Christos Giannoukos, Nikos Sarkas, Panos Vassiliadis, Timos Sellis, and Manolis Koubarakis

Abstract—Qualitative spatial reasoning forms an important part of the commonsense reasoning required for building intelligent Geographical Information Systems (GIS). Previous research has come up with models to capture cardinal direction relations for typical GIS data. In this paper, we target the problem of efficiently computing the cardinal direction relations between regions that are composed of sets of polygons and present two algorithms for this task. The first of the proposed algorithms is purely qualitative and computes, in linear time, the cardinal direction relations between the input regions. The second has a quantitative aspect and computes, also in linear time, the cardinal direction relations with percentages between the input regions. Our experimental evaluation indicates that the proposed algorithms outperform existing methodologies. The algorithms have been implemented and embedded in an actual system, CARDIRECT, that allows the user to 1) specify and annotate regions of interest in an image or a map, 2) compute cardinal direction relations between them, and 3) pose queries in order to retrieve combinations of interesting regions.

Index Terms—Spatial databases and GIS, cardinal direction relations, computing spatial relations.

1 Introduction

Recent developments in the fields of mobile and collaborative computing call for two particularly important features that intelligent Geographical Information Systems (GIS) should support: real-time response to complex queries and interoperability. Related research in the field of spatiotemporal data management and reasoning has provided several results toward the aforementioned problems. Among these research topics, qualitative spatial reasoning has received a lot of attention in the areas of Geographic Information Systems [3], [5], Artificial Intelligence [2], [17], Databases [14], and Multimedia [19]. Several kinds of useful spatial relations have been studied so far, e.g., topological relations [2], [3], [17], directional relations [6], [9], [12], [24], [25], and qualitative distance relations [4], [30]. The uttermost aim in these lines of research is to define new categories of spatial operators as well as to build efficient algorithms for the automatic processing of queries using such operators.

The present paper concentrates on *cardinal direction relations* [6], [9], [12]. Cardinal direction relations are qualitative spatial relations characterizing the relative position of a region with respect to another (e.g., region a is *north of* region b). Our starting point is the cardinal direction framework presented in [6], [24], [25]. The underlining idea of this framework is quite simple. To express the cardinal direction relation between a primary region and a reference region, the aforementioned model uses the lines of the minimum bounding box (MBB) of the reference region to partition the space into a number of areas and records the areas where the primary region falls in. For instance, region a lies *partly northeast and partly east* of region b is a cardinal direction relation. In other words, this model approximates only the reference region (using its MBB) while it uses the exact shape of the primary region. This offers a more precise and expressive model than previous approaches that approximate both extended regions using points or MBBs [5], [9], [12], [14]. In this paper, we employ the cardinal direction model presented in [23], [25] because it is formally defined and can be applied to a wide set of regions (e.g., disconnected regions and regions with holes). Additionally, we also study cardinal direction relations with percentages [6]. This extension offers the option to record the percentage of the primary region that falls into each area of the reference region. For instance, region a lies *50 percent northeast and 50 percent east* of region b is a cardinal direction relation with percentages.

The goal of this paper is to address the problem of efficiently computing the cardinal direction relations [6], [24], [25] between regions that are composed of sets of polygons and to present an implemented system, CARDIRECT that encapsulates this functionality in order to answer interesting user queries.

Algorithms for computing interesting and useful spatial relations are in the heart of Spatial Databases and GISs. Thus, they have attracted the interest of many researchers. For some spatial relations (like topological relations [2], [3]), these algorithms can be found in the Computational Geometry literature [13], [16] while for other spatial relations (like cardinal direction relations [6], [9], [24]), there exist more direct methods. For instance, Peuquet and Ci-Xiang [15] capture cardinal direction on polygons using points and MBB's approximations and present linear time algorithms that compute the relative direction.

- *S. Skiadopoulos, C. Giannoukos, N. Sarkas, and T. Sellis are with the School of Electrical and Computer Engineering, Division of Computer Science, National Technical University of Athens, 9 Iroon Polytechniou Street, Zographou 157 73, Athens, Hellas (Greece). E-mail: {spiros, chgian, sarkas, timos}@dblab.ece.ntua.gr.*
- *P. Vassiliades is with the Department of Computer Science, University of Ioannina, PO Box 1186, GR 45110 Ioannina. E-mail: pvassil@cs.uoi.gr.*
- *M. Koubarakis is with the Department of Electronic and Computer Engineering, Technical University of Crete, University Capus, Kounoupidiana 73100 Chania, Crete, Greece. E-mail: manolis@intelligence.tuc.gr.*

Manuscript received 27 Jan. 2005; revised 16 June 2005; accepted 27 June 2005; published online 19 Oct. 2005.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-0037-0105.

For the cardinal direction relations model that we will employ [6], [24], [25], a naive method to solve the computation of cardinal direction relations problem is to use polygon clipping methods from Computational Geometry [7], [8], [11], [26], [28]. Given two polygons a and b, a polygon clipping algorithm returns a new polygon that represents the part of polygon a that falls inside polygon b. Thus, we can use such algorithms to check if (and in what percentage) a primary region falls in each area of the reference region and solve the cardinal direction relations problem.

In this paper, we move a step forward and solve the problem of computing cardinal direction relations in a more direct way that does not use polygon clipping. To this end, we present two linear time algorithms (with respect to the number of input polygons' edges). The input of both algorithms is two sets of polygons representing the two involved regions. The first of the proposed algorithms computes the cardinal direction relation between the input regions, while the second computes the cardinal direction relations with percentages between the input regions. The latter algorithm is based on a novel technique for the computation of areas of polygons. To the best of our knowledge, these are the first algorithms that *directly* handle the aforementioned problem for the cardinal direction relations that can be expressed in [6], [24], [25], [21].

We also perform an experimental evaluation of the algorithms' performance. Our algorithms are compared with methods based on well-studied polygon clipping algorithms from Computational Geometry. Particularly, we employ the clipping methods of Sutherland and Hodgman [26] and Liang and Barsky [8]. We demonstrate that our algorithms outperform these methods by taking advantage of the specific properties of the cardinal direction relations computation problem.

The proposed algorithms have been implemented and embedded in an actual system, CARDIRECT. The scenario for CARDIRECT usage is based on a simple scheme, where raw digital data (e.g., astronomical images, city maps, etc.,) are examined to identify possibly interesting areas. CARDIRECT allows the user to specify, edit, and annotate these regions of interest. Then, the tool automatically computes the cardinal direction relations between these regions using the aforementioned linear algorithms. The information concerning the underlying image, the introduced regions, their composing polygons, and their relations form the metainformation for the overall configuration. This metainformation is persistently stored using an XML description. Additionally, the user is allowed to query the stored XML description of the image and retrieve combinations of interesting regions. A preliminary version of this paper appears in *Proceedings of the of Extending Database Technology Conf.* [21].

The rest of the paper is organized as follows: Section 2 presents the cardinal direction relations model. In Section 3, we present two algorithms for the problem of computing cardinal direction relations and prove their correctness. Section 4 experimentally evaluates the performance of the aforementioned algorithms. In Section 5, we present the architecture and functionality of the CARDIRECT tool. Finally, Section 6 offers conclusions and lists topics of future research.

2 A Formal Model for Cardinal Direction Information

Cardinal direction relations, for various types of regions, have been defined in [6], [24], [25]. Goyal [6] first presented a set of cardinal direction relations for connected regions. Skiadopoulos and Koubarakis [22], [24] formally define the above cardinal direction relations, propose composition algorithms, and prove that these algorithms are correct. Moreover, Skiadopoulos and Koubarakis [23], [25] have presented an extension that handles disconnected regions and regions with holes, and study the consistency problem for a given set of cardinal direction constraints. In this paper, we start with the cardinal direction relations for the composite regions presented in [23], [25] and then we present an extension with percentages in the style of [6].

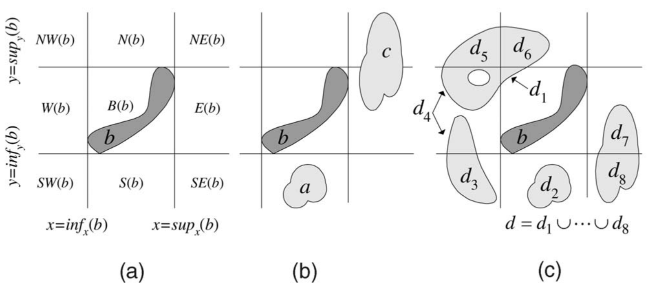


Fig. 1. Tiles, relations, and component variables *and* $a = a_1 \cup \cdots \cup a_k$.

We consider the Euclidean space $\Re^2$. *Regions* are defined as nonempty and bounded sets of points in $\Re^2$. Let a be a region. The *greatest lower bound* or the *infimum* [10] of the *projection* of region a on the x-axis (respectively, y-axis) is denoted by $inf_x(a)$ (respectively, $inf_y(a)$). The *least upper bound* or the *supremum* of the *projection* of region a on the x-axis (respectively, y-axis) is denoted by $sup_x(a)$ (respectively, $sup_y(a)$). We will often refer to sup and inf as *endpoints*.

The *minimum bounding box* of a region b, denoted by $mbb(b)$, is the rectangular region formed by the straight lines $x = inf_x(b)$, $x = sup_x(b)$, $y = inf_y(b)$, and $y = sup_y(b)$ (see Fig. 1a). Obviously, the projections on the x-axis (respectively, y-axis) of a region and its minimum bounding box have the same endpoints.

Throughout this paper, we will consider regions that are formed by finite unions of regions that are homeomorphic to the *closed unit disk* [24]. This set of regions is denoted by REG^*. Regions in REG^* can be *disconnected* and have *holes*. However, class REG^* excludes points, lines, and regions with emanating lines. Regions in REG^* are very common in GISs, Multimedia, and Image Databases [1], [19]. For example, countries are made up of separations (islands, exclaves, external territories) and holes (enclaves) [1]. In Fig. 1, regions a, b, c, and $d = d_1 \cup \cdots \cup d_8$ are in REG^*. Notice that region d is disconnected and has a hole.

Let us now consider two arbitrary regions a and b in REG^*. Let region a be related to region b through a cardinal direction relation (e.g., a is north of b). Region b will be called the *reference* region (i.e., the region which the relation refers to), while region a will be called the *primary* region (i.e., the region for which the relation is introduced). The axes forming the minimum bounding box of the reference region b divide the space into nine areas which we call *tiles* (Fig. 1a). The peripheral tiles correspond to the eight cardinal direction relations $south$, $southwest$, $west$, $northwest$, $north$, $northeast$, $east$, and $southeast$. These tiles will be denoted by $S(b)$, $SW(b)$, $W(b)$, $NW(b)$, $N(b)$, $NE(b)$, $E(b)$, and $SE(b)$, respectively. The central area corresponds to the region's minimum bounding box and is denoted by $B(b)$. By definition, each one of these tiles includes the parts of the axes forming it. The union of all nine tiles is $\Re^2$.

If a primary region a is included (in the set-theoretic sense) in tile $S(b)$ of some reference region b (Fig. 1b), then we say that a *is south of* b and we write $a\ S\ b$. Similarly, we can define *southwest* (SW), *west* (W), *northwest* (NW), *north* (N), *northeast* (NE), *east* (E), *southeast* (SE), and *bounding box* (B) relations.

If a primary region c lies partly in the area $NE(b)$ and partly in the area $E(b)$ of some reference region b (Fig. 1b), then we say that c *is partly northeast and partly east of* b and we write $c\ NE{:}E\ b$.

The general definition of a cardinal direction relation in our framework is as follows:

Definition 1. *A* cardinal direction relation *is an expression* $R_1 : \cdots : R_k$*, where* $1 \leq k \leq 9$,

$$R_1, \ldots, R_k \in \{B, S, SW, W, NW, N, NE, E, SE\},$$

and $R_i \neq R_j$ *for every* i, j *such that* $1 \leq i, j \leq k$*, and* $i \neq j$*. A cardinal direction relation* $R_1 : \cdots : R_k$ *is called* single-tile *if* $k = 1$*; otherwise, it is called multitile.*

To formally define cardinal direction relations, we first define single-tile relations and then multitile relations. Let a *and* b *be two regions in* REG^**. Single-tile cardinal direction relations are defined as follows:*

- $a\ B\ b$ $inf_x(b) \leq inf_x(a)$, $sup_x(a) \leq sup_x(b)$,
$$inf_y(b) \leq inf_y(a),$$
and $sup_y(a) \leq sup_y(b)$.
- $a\ S\ b$ $sup_y(a) \leq inf_y(b)$, $inf_x(b) \leq inf_x(a)$, *and*
$$sup_x(a) \leq sup_x(b).$$
- $a\ SW\ b$ $sup_x(a) \leq inf_x(b)$ *and* $sup_y(a) \leq inf_y(b)$.
- $a\ W\ b$ $sup_x(a) \leq inf_x(b)$, $inf_y(b) \leq inf_y(a)$, *and*
$$sup_y(a) \leq sup_y(b).$$
- $a\ NW\ b$ $sup_x(a) \leq inf_x(b)$ *and* $sup_y(b) \leq inf_y(a)$.
- $a\ N\ b$ $sup_y(b) \leq inf_y(a)$, $inf_x(b) \leq inf_x(a)$, *and*
$$sup_x(a) \leq sup_x(b).$$
- $a\ NE\ b$ $sup_x(b) \leq inf_x(a)$ *and* $sup_y(b) \leq inf_y(a)$.
- $a\ E\ b$ $sup_x(b) \leq inf_x(a)$, $inf_y(b) \leq inf_y(a)$, *and*
$$sup_y(a) \leq sup_y(b).$$
- $a\ SE\ b$ $sup_x(b) \leq inf_x(a)$ *and* $sup_y(a) \leq inf_y(b)$.

Multitile ($2 \leq k \leq 9$) *relations are defined as follows:*

- $a\ R_1 : \cdots : R_k\ b$ *iff there exist regions* $a_1, \ldots, a_k \in REG^*$ *such that* $a_1\ R_1\ b, \ldots, a_k\ R_k\ b$ *and*
$$a = a_1 \cup \cdots \cup a_k.$$

In Definition 1, notice that for every i, j such that $1 \leq i$, $j \leq k$, and $i \neq j$, a_i, and a_j have disjoint interiors but may share points aolong their boundaries.

Example 1. Expressions S, $NE{:}E$ and $B{:}S{:}SW{:}W{:}NW{:}N{:}E{:}SE$ are cardinal direction relations. The first relation is single-tile while the others are multitile. In Fig. 1, we have $a\ S\ b$, $c\ NE : E\ b$, and $d\ B{:}S{:}SW{:}W{:}NW{:}N{:}E{:}\ SE\ b$. For instance, in Fig. 1c, we have $d\ B{:}S{:}SW{:}W{:}NW{:}N{:}E{:}SE\ b$ because there exist regions $d_1, \ldots, d_8$ in REG^* such that $d = d_1 \cup \cdots \cup d_8$, $d_1\ B\ b$, $d_2\ S\ b$, $d_3\ SW\ b$, $d_4\ W\ b$, $d_5\ NW\ b$, $d_6\ N\ b$, $d_7\ E\ b$, and $d_8\ SE\ b$.

In order to avoid confusion, we will write the single-tile elements of a cardinal direction relation according to the following order: B, S, SW, W, NW, N, NE, E, and SE. Thus, we always write $B{:}S{:}W$ instead of $W{:}B{:}S$ or $S{:}B{:}W$. Moreover, for a relation such as $B{:}S{:}W$ we will often refer to B, S, and W as its *tiles*.

The set of cardinal direction relations for regions in REG^* contains $\sum_{i=1}^{9} \binom{9}{i} = 511$ elements. We will use $\mathcal{D}^*$ to denote this set. Relations in $\mathcal{D}^*$ are jointly exhaustive and pairwise disjoint, and can be used to represent *definite information* about cardinal directions, e.g., $a\ N{:}W\ b$ denotes that region a is *partly north and partly west* of region b. Using the relations of $\mathcal{D}^*$ as our basis, we can define the *powerset* $2^{\mathcal{D}^*}$ of $\mathcal{D}^*$ which contains 2^{511} relations. Elements of $2^{\mathcal{D}^*}$ are called *disjunctive cardinal direction relations* and can be used to represent not only definite, but also *indefinite information* about cardinal directions, e.g., $a\ \{N, W\}\ b$ denotes that region a is *either* north *or* west of region b. A detailed study of cardinal direction relations is presented in [24], [25].

Goyal [6] uses *direction relation matrices* to represent cardinal direction relations. At a finer level of granularity, the notation of [6] also offers the option to record how much of the a region falls into each tile. Such relations are called *cardinal direction relations with percentages* and can be represented with *cardinal direction matrices with percentages*. Let a and b be two regions in REG^*. The cardinal direction matrices with percentages can be defined as follows:

$$a\ \frac{100\%}{area(a)} \cdot \begin{bmatrix} area(NW(b) \cap a) & area(N(b) \cap a) & area(NE(b) \cap a) \\ area(W(b) \cap a) & area(B(b) \cap a) & area(E(b) \cap a) \\ area(SW(b) \cap a) & area(S(b) \cap a) & area(SE(b) \cap a) \end{bmatrix} b,$$

where $area(a)$ denotes the area of region a.

Consider, for example, regions c and b in Fig. 1b; region c is 50 percent northeast and 50 percent east of region b. This relation is captured by the following cardinal direction matrix with percentages:

$$c \begin{bmatrix} 0\% & 0\% & 50\% \\ 0\% & 0\% & 50\% \\ 0\% & 0\% & 0\% \end{bmatrix} b.$$

In this paper, we will use simple assertions (e.g., S, $B{:}S{:}SW$) to capture cardinal direction relations [22], [23], [24], [25] and direction relations matrices to capture cardinal direction relations with percentages [6].

3 Computing Cardinal Direction Relations

Typically, in Geographical Information Systems and Spatial Databases, the composite regions in REG^* are represented using sets of polygons [18], [29]. In this paper, the edges of polygons are taken in a clockwise order. For instance, in Fig. 2, region $a \in REG^*$ is represented using polygon $(M_1 \cdots M_9)$. Notice that using sets of polygons, we can even represent regions with holes. For instance, in Fig. 2, region

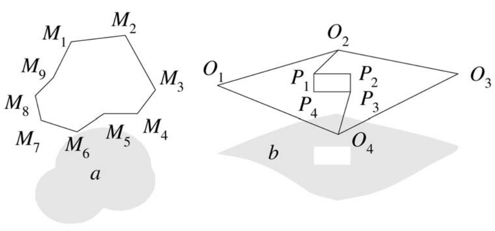


Fig. 2. Using polygons to represent regions.

$b \in REG^*$ is represented using polygons $(O_2O_3O_4P_3P_2P_1)$ and $(O_1O_2P_1P_4P_3O_4)$.

Given the polygon representations of a primary region a and a reference region b, the *computation of cardinal direction relations problem* lies in the calculation of the cardinal direction relation R, such that $a\,R\,b$ holds. Similarly, we can define the *computation of cardinal direction relations with percentages problem*.

Let us consider a primary region a and a reference region b. According to Definition 1, in order to calculate the cardinal direction relation between regions a and b, we have to divide the primary region a into segments such that each segment falls exactly into one tile of b. Furthermore, in order to calculate the cardinal direction relation with percentages we also have to measure the area of each segment. Segmenting polygons using *bounded boxes* is a well-studied topic of Computational Geometry called *polygon clipping* [7], [8], [11], [26], [28]. A polygon clipping algorithm can be extended to handle *unbounded boxes* (such as the tiles of reference region b) as well. Since polygon clipping algorithms are very efficient (linear in the number of polygon edges), someone would be tempted to use them for the calculation of cardinal direction relations and cardinal direction relations with percentages. Let us briefly discuss the disadvantages of such an approach.

Let us consider regions a and b presented in Fig. 3a. Region a is formed by a quadrangle (i.e., a total of four edges). To achieve the desired segmentation, polygon clipping algorithms introduce new edges to a [26], [8]. After the clipping algorithms are performed (Fig. 3b), region a is formed by four quadrangles (i.e., a total of 16 edges). The worst case that we can think (illustrated in Fig. 3c) starts with three edges (a triangle) and ends with 35 edges (two triangles, six quadrangles, and one pentagon).

Specifically, the problem is focused on the new edges that lie on the lines forming the bounding box of the reference region. For example, in Fig. 3b, these lines are O_1B_1, B_1O_3, O_2B_1, and B_1O_4. These new edges are only used for the calculation of cardinal direction relations and are discarded afterwards. Thus, it would be important to minimize their number. Moreover, in order to clip the primary region a, its edges must be scanned nine times (one time for every tile of the reference region b). In real GIS applications, we expect that the average number of edges is quite high. Thus, each scan of the edges of a polygon can be time consuming. Finally, polygon clipping algorithms sometimes require complex floating point operations which are costly.

In Sections 3.1 and 3.2, we consider the problem of calculating cardinal direction relations and cardinal direction relations with percentages, respectively. We provide algorithms specifically tailored for this task, which avoid the drawbacks of polygon clipping-based methods. Our proposal does not segment polygons; instead, it only divides some of the polygon edges. In Example 2, we show that such a division is necessary for the correct calculation. Interestingly,

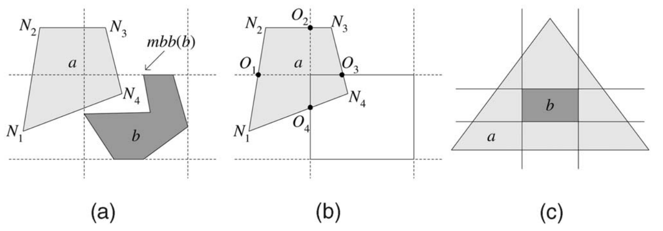


Fig. 3. Polygon clipping.

the resulting number of introduced edges is significantly smaller than the respective number of polygon clipping methods. Furthermore, the complexity of our algorithms is not only linear in the number of polygon edges, but it can be performed with a single pass. The efficiency of these algorithms is experimentally verified in Section 4.

3.1 Cardinal Direction Relations

We start by considering the calculation of cardinal direction relations problem. First, we need the following definition.

Definition 2. *Let $R_1, \ldots, R_k$ be cardinal direction relations. The* tile-union *of $R_1, \ldots, R_k$, denoted by tile-union$(R_1, \ldots, R_k)$, is a relation formed from the union of the tiles of $R_1, \ldots, R_k$.*

For instance, if $R_1 = S{:}SW$, $R_2 = S{:}E{:}SE$, and $R_3 = W$, then we have *tile-union*$(R_1, R_2) = S{:}SW{:}E{:}SE$ and *tile-union*$(R_1, R_2, R_3) = S{:}SW{:}W{:}E{:}SE$.

Let $S_a = \{p_1, \ldots, p_k\}$ and $S_b = \{q_1, \ldots, q_l\}$ be sets of polygons representing a primary region a and a reference region b. To calculate the cardinal direction R between the primary region a and the reference region b, we can record the tiles of region b where the points forming the edges of the polygons $p_1, \ldots, p_k$ fall in. Unfortunately, as the following example presents, this is not enough.

Example 2. Let us consider the region a (formed by the single polygon $(N_1N_2N_3N_4)$) and the region b presented in Fig. 4a. Clearly, points N_1, N_2, N_3, and N_4 lie in $W(b)$, $NW(b)$, $NW(b)$ and $NE(b)$, respectively, but the relation between p and b is $B{:}W{:}NW{:}N{:}NE$ and not $W{:}NW{:}NE$.

The problem of Example 2 arises because there exist edges of polygon $(N_1N_2N_3N_4)$ that expand over three tiles of the reference region b. For instance, N_3N_4 expands over tiles $NW(b)$, $N(b)$, and $NE(b)$. In order to handle such situations, we use the lines forming the minimum bounding box of the reference region b to divide the edges of the polygons representing the primary region a and create new edges such that 1) region a does not change and 2) every new edge lies in exactly one tile. To this end, for every edge AB of region a such that A and B lie in different tiles of b, we compute the set of intersection points $\mathcal{I}$ of AB with the lines forming the minimum bounding box of b. We use the intersection points of $\mathcal{I}$ to divide AB into a number of segments $\{AO_1, \ldots, O_kB\}$. Each segment $AO_1, \ldots, O_kB$ lies in only one tile of b and the union of all segments is AB. Thus, we can safely replace edge AB with $AO_1, \ldots, O_kB$ without affecting region a. Finally, to compute the cardinal direction between regions a and b, we only have to record the tile of b where each new segment lies. Choosing a single

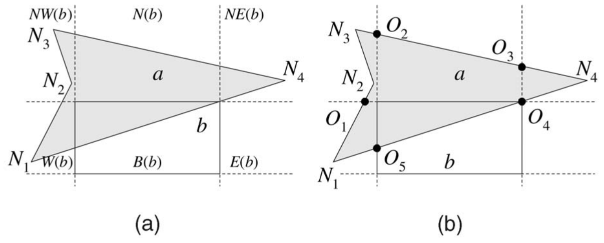


Fig. 4. Illustration of Examples 2 and 3.

point from each segment is sufficient for this purpose; we choose to pick the middle of the segment as a representative point. Thus, the tile where the middle point lies gives us the tile of the segment, too. The above procedure is captured in Algorithm COMPUTE-CDR (Fig. 5) and is illustrated in the following example.

Example 3. Let us continue with the regions of Example 2 (see also Fig. 4). Algorithm COMPUTE-CDR considers every edge of region a (polygon $(N_1N_2N_3N_4)$) in turn and performs the replacements presented in the following table.

Edge	*Replace with (new edges)*
N_1N_2	$N_1O_1,\ O_1N_2$
N_2N_3	N_2N_3
N_3N_4	$N_3O_2,\ O_2O_3,\ O_3N_4$
N_4N_1	$N_4O_4,\ O_4O_5,\ O_5N_1$

It is easy to verify that every new edge lies in exactly one tile of b (Fig. 4b). The middle points of the new edges lie in $B(b)$, $W(b)$, $NW(b)$, $N(b)$, $NE(b)$, and $E(b)$. Therefore, Algorithm COMPUTE-CDR returns $B{:}W{:}NW{:}N{:}NE{:}E$, which precisely captures the cardinal direction relation between regions a and b.

Notice that in Example 3, Algorithm COMPUTE-CDR takes as input a quadrangle (four edges) and returns nine edges. This should be contrasted with the polygon clipping method that would have resulted in 19 edges (two triangles, two quadrangles, and one pentagon). Similarly, for the shapes in Figs. 3b and 3c, Algorithm COMPUTE-CDR introduces eight and 11 edges, respectively, while polygon clipping methods introduce 16 and 34 edges, respectively.

The following theorem captures the correctness of Algorithm COMPUTE-CDR and measures its complexity.

Theorem 1. *Algorithm COMPUTE-CDR is correct, i.e., it returns the cardinal direction relation between two regions a and b in REG^* that are represented using two sets of polygons S_a and S_b, respectively. The running time of Algorithm COMPUTE-CDR is $\mathcal{O}(k_a + k_b)$, where k_a (respectively, k_b) is the total number of edges of all polygons in S_a (respectively, S_b).*

Proof. Initially, Algorithm COMPUTE-CDR computes the minimum bounding box of b. Then, the algorithm considers every polygon p in S_a (the outer For loop) and calculates the cardinal direction relation of p with respect to region b. To this end, the algorithm considers every edge AB of polygon p (the inner For loop). Each edge, is divided such that every new edge lies in exactly one tile of the reference region b. To find in which tile of the region b a particular edge AB lies, we just have to check the middle point of AB. For instance, consider Fig. 6a. Edge AB is divided into three segments AO_1, O_1O_2, and O_2B. The middle points of these segments lie in NW, N, and NE tiles of b which are exactly the tiles where edge AB lies.

Degenerated cases arise when an edge AB (and, of course, its middle point) lies entirely on one of the lines forming the minimum bounding box of the reference region b. Such edges are called *degenerated edges*. The middle point of such an edge cannot be used to determine its relative position with respect to a reference region. For instance, consider Fig. 6b, where the degenerated edge AB lies on the east vertical line of the minimum bounding box of b. Using the middle point O of AB, we cannot tell whether the polygon p lies in $N(b)$ or $NW(b)$. We distinguish the following cases:

```
Algorithm COMPUTE-CDR
Input: Two sets of polygons $S_a$ and $S_b$ representing regions $a, b \in REG^*$ resp.
Output: The cardinal direction relation $R$ such that $a\ R\ b$ holds.
Method:
Use $S_b$ to compute the minimum bounding box $mbb(b)$ of $b$.
Let $R$ be the empty relation.
For every polygon $p$ of $S_a$
    For every edge $AB$ of polygon $p$
        If $A$ and $B$ lie in the same tile $T$ of $b$ Then $R = \textit{tile-union}(\ R,\ T\ )$
        Else
            Let $\mathcal{I}$ be the set of intersection points of $AB$ with the lines forming box $b$.
            Let $\{AO_1, \ldots, O_kB\}$ be the segments that points in $\mathcal{I}$ divide $AB$.
            Replace $AB$ with $\{AO_1, \ldots, O_kB\}$ in the representation of polygon $p$.
            Let $T_1, \ldots, T_k$ be the tiles of $b$ in which the middle points of
            edges $\{AO_1, \ldots, O_kB\}$ lie.
            $R = \textit{tile-union}(\ R,\ T_1, \ldots, T_k\ )$
        EndIf
    EndFor
    If the center of $mbb(b)$ is in $p$ Then $R = \textit{tile-union}(\ R,\ B\ )$
EndFor
Return $R$
```

Fig. 5. Algorithm COMPUTE-CDR.

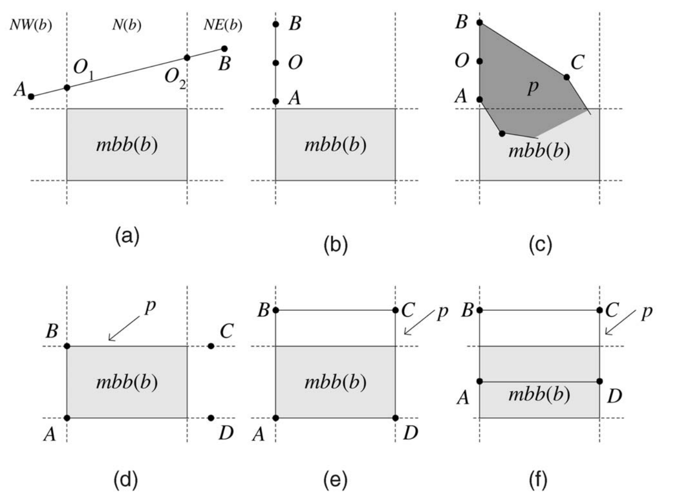


Fig. 6. Cases considered by the proof of Theorem 1.

1. *Polygon p has only one degenerated edge.* Let AB be the degenerated edge of polygon p. In this case, we can resolve the uncertainty introduced by edge AB by looking at the edge that precedes or the edge that follows AB. For instance, let us consider Fig. 6c and the degenerated edge AB of p. The fact that polygon p lies in $N(b)$ cannot be derived by edge AB, but it can be derived by edge BC that follows AB. Summarizing, we can safely ignore the degenerated edge AB of a polygon p without affecting the correct computation of the cardinal direction relation of polygon p with respect to region b.
2. *Polygon p has only degenerated edges.* This may happen only when $p = mbb(b)$ (see Fig. 6d). In this case, none of the middle points determine the cardinal direction relation. This case is handled by the If statement of Algorithm COMPUTE-CDR. For instance, in Fig. 6d, the center of $mbb(b)$ lies inside polygon p, thus the algorithm correctly returns $p\ B\ b$. Thus, in this case, we can ignore the degenerated edges, too. Notice that the If statement is sound, so it will not affect the correct computation of any other (nondegenerated) case.
3. *Polygon p has k, $1 < k$, degenerated edges.* It is easy to see that to correctly compute the direction relation between polygon p and region b, we have to ignore the degenerate edges,consider only nondegenerated edges, and, finally, check whether the center of $mbb(b)$ lies inside polygon p, exactly as it is done in Algorithm COMPUTE-CDR.
   See, for instance, Fig. 6e and Fig. 6f, where the algorithm correctly returns $p\ B:N\ b$.

Let us now calculate the running time of this algorithm. The minimum bounding box of b can be calculated in $\mathcal{O}(k_b)$ time. Following, Algorithm COMPUTE-CDR performs an outer loop. This loop examines every polygon p in S_a and performs an inner loop and a check (the If statement). The inner loop examines every edge of polygon p once. The condition of the If statement also requires an examination of every edge of polygon p [16]. Summarizing, the outer loop can be executed in $\mathcal{O}(k_a)$ time. Finally, the total complexity of Algorithm COMPUTE-CDR is $\mathcal{O}(k_a + k_b)$ time. □

Notice that Algorithm COMPUTE-CDR can be easily changed so that the inner loop and the If statement are merged and performed with a single pass on the edges of region a (instead of the two passes that are required now). The new algorithm has the same complexity, but it has improved performance especially in real GIS applications where the number of edges is high.

Summarizing this section, we can use Algorithm COMPUTE-CDR to compute the cardinal direction relation between two sets of polygons representing two regions a and b in REG^*. The following section considers the case of cardinal direction relations with percentages.

3.2 Cardinal Direction Relations with Percentages

In order to compute cardinal direction relations with percentages, we have to calculate the area of the primary region that falls in each tile of the reference region. A naive way for this task is to segment the polygons that form the primary region (using polygon clipping algorithms) so that every polygon lies in exactly one tile of the reference region. Then, for each tile of the reference region, we find the polygon segments of the primary region that lie inside it and compute their area. In this section, we will propose an alternative method that is based on Algorithm COMPUTE-CDR. This method simply computes the area between the edges of the polygons that represent the primary region and an appropriate reference line without segmenting these polygons. Specifically, in the rest of this section, we proceed with the following steps:

1. We present two expressions to compute the areas that lie between a certain edge and two axes (Definition 4) and we employ these expressions to compute the area of a polygon (Lemma 1).
2. Depending on the tile T that an edge of a polygon lies in, we present a simple method to decide which of the two aforementioned expressions should be used for the correct computation of the area that lies within tile T (Lemma 2).
3. We present Algorithm COMPUTE-CDR% that computes the cardinal direction relation with percentages of two regions (Fig. 10).

We will now present a method to compute the area between a line and an edge. We will first need the following definition.

Definition 3. *Let AB be an edge and e be a line. Line e* does not cross *AB if and only if one of the following holds: 1) AB and e do not intersect, 2) AB and e intersect only at point A or B, or 3) AB completely lies on e.*

For example, in Fig. 7, line $y = l$ does not cross edge AB. Let us now calculate the area between an edge and a line.

Definition 4. *Let $A(x_A, y_A)$ and $B(x_B, y_B)$ be two points forming edge AB, $y = l$, and $x = m$ be two lines that do not cross AB. Also, let L_A and L_B (respectively, M_A and M_B) be the projections of points A, B to line $y = l$ (respectively, $x = m$)—see also Fig. 7. We define the expressions $E_l(AB)$ and $E'_m(AB)$ as follows:*

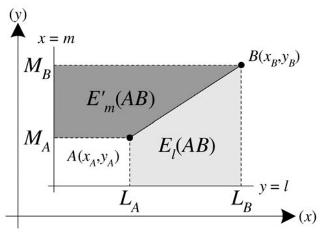


Fig. 7. Area between an edge and a line.

$$E_l(AB) = \frac{(x_A - x_B)(y_A + y_B - 2l)}{2} \; and$$
$$E'_m(AB) = \frac{(y_A - y_B)(x_A + x_B - 2m)}{2}.$$

Expressions $E_l(AB)$ and $E'_m(AB)$ can be positive or negative depending on the direction of vector $\overrightarrow{AB}$. It is easy to verify that $E_l(AB) = -E_l(BA)$ and $E'_m(AB) = -E'_m(BA)$ holds. The absolute value of $E_l(AB)$ equals to the area between edge AB and line $y = l$, i.e., the area of polygon (ABL_BL_A). In other words, we have

$$area(\ (ABL_BL_A)\ ) = |E_l(AB)| = \left| \frac{(x_A - x_B)(y_A + y_B - 2l)}{2} \right|.$$

Symmetrically, the area between edge AB and line $x = m$, i.e., the area of polygon (AM_BM_AB), equals to the absolute value of $E'_m(AB)$, thus we have

$$area(\ (AM_AM_BB)\ ) = |E'_m(AB)| = \left| \frac{(y_A - y_B)(x_A + x_B - 2m)}{2} \right|.$$

Expressions E_l and E'_m can be used to calculate the area of polygons. Consider the following lemma.

Lemma 1. *Let $p = (N_1 \cdots N_k)$ be a polygon and $y = l$ and $x = m$ be two lines that do not cross with any edge of polygon p. The area of polygon p, denoted by $area(p)$, can be calculated as follows:*

$$area(p) = |\ E_l(N_1N_2) + \cdots + E_l(N_kN_1)\ | = |\ E'_m(N_1N_2) + \cdots + E'_m(N_kN_1)\ |.$$

Proof. Let O be an arbitrary point. The area of polygon $p = (N_1 \cdots N_k)$ can be computed using expression

$$area(p) = |\ E(O, N_1, N_2) + E(O, N_2, N_3) + \cdots + E(O, N_k, N_1)\ |,$$

where $E(A, B, C)$ is the area of triangle with corners A, B, and C [13], [16]. If $N_i = (x_i, y_i)$, $1 \le i \le k$, then equivalently we have:

$$area(p) = \left| \frac{1}{2} \left(\sum_{i=1}^{k-1} (x_iy_{i+1} - y_ix_{i+1}) + (x_ky_1 - y_kx_1) \right) \right|.$$

Let us now consider expressions $|\ E_l(N_1N_2) + \cdots + E_l(N_kN_1)\ |$ and $|\ E'_m(N_1N_2) + \cdots + E'_m(N_kN_1)\ |$. It is easy to verify that by replacing expressions $E_l(N_iN_j) = \frac{(x_i - x_j)(y_i + y_j - 2l)}{2}$ and $E'_m(N_iN_j) = \frac{(y_i - y_j)(x_j + x_j - 2m)}{2}$, we have that:

$$area(p) = |\ E_l(N_1N_2) + \cdots + E_l(N_kN_1)\ | = |\ E'_m(N_1N_2) + \cdots + E'_m(N_kN_1)\ |.$$

□

Notice that, in order to calculate the area of a polygon p, Computational Geometry algorithms use a similar method that is based on a reference point (instead of a line) [13], [16]. This point-based method is not appropriate for our case because it requires to segment the primary region using polygon clipping algorithms (see also the discussion at the beginning of Section 3). In the rest of this section, we will present a method that utilizes expressions E_l and E'_m and does not require polygon clipping.

Example 4. Let us consider polygon $p = (N_1N_2N_3N_4)$ and line $y = l$ presented in Fig. 8d. The area of polygon p can be calculated using formula

$$area(p) = |\ E_l(N_1N_2) + E_l(N_2N_3) + E_l(N_3N_4) + E_l(N_4N_1)\ |.$$

All the intermediate expressions $E_l(N_1N_2)$,

$$E_l(N_1N_2) + E_l(N_2N_3), E_l(N_1N_2) + E_l(N_2N_3) + E_l(N_3N_4),$$

and $E_l(N_1N_2) + E_l(N_2N_3) + E_l(N_3N_4) + E_l(N_4N_1)$ are presented as the gray areas of Figs. 8a, 8b, 8c, and 8d, respectively.

We will use expressions E_l and E'_m to compute the percentage of the area of the primary region that falls in each tile of the reference region. Let us consider region a presented in Fig. 9. Region a is formed by polygons $(N_1N_2N_3N_4)$ and $(M_1M_2M_3)$. Similarly to Algorithm COMPUTE-CDR, to compute the cardinal direction relation with percentages of a with b we first use the $mbb(b)$ to divide the edges of region a. Region b is not depicted in Fig. 9, in order to avoid overloading the figure. Let $x = m_1$, $x = m_2$, $y = l_1$, and $y = l_2$ be the lines forming $mbb(b)$. These lines divide the edges of polygons $(N_1N_2N_3N_4)$ and $(M_1M_2M_3)$ as shown in Fig. 9.

Let us now compute the area of a that lies in the NW tile of b (i.e., $area(NW(b) \cap a)$). Notice that

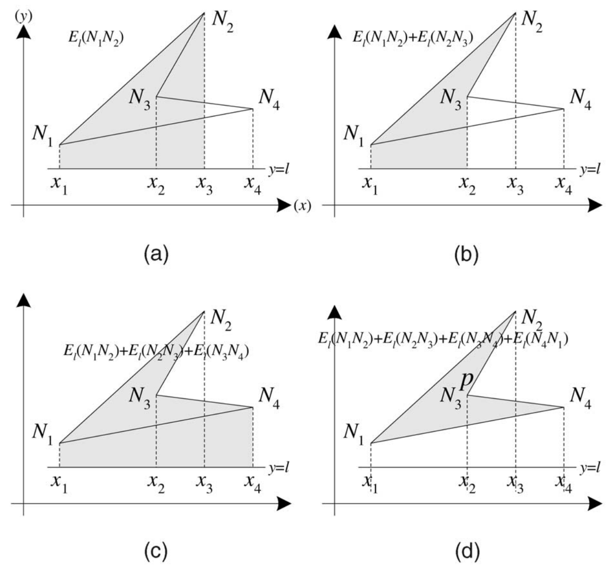
$$area(NW(b) \cap a) = area((O_1N_2O_2B_1)).$$

To compute the area of polygon $(O_1N_2O_2B_1)$ it is convenient to use the reference line $x = m_1$. Doing so, we do not have to compute edges B_1O_1 and O_2B_1 because $E'_{m_1}(B_1O_1) = 0$ and $E'_{m_1}(O_2B_1) = 0$ hold and, thus, the area we are looking for can be calculated with formula

$$area(NW(b) \cap a) = area((O_1N_2O_2B_1)) = |\ E'_{m_1}(O_1N_2) + E'_{m_1}(N_2O_2)\ |.$$

In other words, to compute the area of a that lies in $NW(b)$ $(area(NW(b) \cap a))$ we calculate the area between the west line of $mbb(b)$ $(x = m_1)$ and every edge of a that lies in $NW(b)$, i.e., we have $area(NW(b) \cap a) = |\ \sum_{AB \in NW(b)} E'_{m_1}(AB)\ |$.

Similarly, to calculate the area of a that lies in the $W(b)$ and $SW(b)$, we can use the expressions: $area(W(b) \cap a) = |\ \sum_{AB \in W(b)} E'_{m_1}(AB)\ |$ and



(y)
(x)
$E_l(N_1N_2)$
$E_l(N_1N_2)+E_l(N_2N_3)$
$E_l(N_1N_2)+E_l(N_2N_3)+E_l(N_3N_4)$
$E_l(N_1N_2)+E_l(N_2N_3)+E_l(N_3N_4)+E_l(N_4N_1)$
N_1
N_2
N_3
N_4
p
$y=l$
x_1
x_2
x_3
x_4
(a)
(b)
(c)
(d)

Fig. 8. Using expression E_l to calculate the area of a polygon.

$$area(SW(b) \cap a) = \left| \sum_{AB \in SW(b)} E'_{m_1}(AB) \right|.$$

For instance, in Fig. 9, we have $area(W(b) \cap a) = |E'_{m_1}(N_1O_1) + E'_{m_1}(O_4N_1)|$ and $area(SW(b) \cap a) = 0$.

To calculate the area of a that lies in $NE(b)$, $E(b)$, $SE(b)$, $S(b)$, and $N(b)$ we simply have to change the line of reference that we use. In the first three cases, we use the east line of $mbb(b)$ (i.e., $x = m_2$ in Fig. 9), in the fourth case, we use the south line of $mbb(b)$ ($y = l_1$) and, in the last case, we use the north line of $mbb(b)$ ($y = l_2$). In all cases, we use the edges of a that fall in the tile of b that we are interested in. Thus, we have: $area(T(b) \cap a) = | \sum_{AB \in T(b)} E'_{m_2}(AB) |$ if $T \in \{NE, E, SE\}$, $area(S(b) \cap a) = | \sum_{AB \in S(b)} E_{l_1}(AB) |$, and

$$area(N(b) \cap a) = \left| \sum_{AB \in N(b)} E_{l_2}(AB) \right|.$$

For instance, in Fig. 9, we have $area(N(b) \cap a) = | E_{l_2}(O_2N_3) + E_{l_2}(N_3O_3) |$ and

$$area(E(b) \cap a) = | E'_{m_2}(Q_1M_2) + E'_{m_2}(M_2Q_2) |.$$

Let us now consider the area of a that lies in $B(b)$. None of the lines of $mbb(b)$ can help us compute $area(B(b) \cap a)$ without segmenting the polygons that represent region a. For instance, in Fig. 9, using line $y = l_1$, we have

$$area(B(b) \cap a) = | E_{l_1}(Q_4M_1) + E_{l_1}(M_1Q_1) + E_{l_1}(O_3N_4) + E_{l_1}(N_4O_4) + E_{l_1}(B_1O_3) |.$$

Edge B_1O_3 is not an edge of any of the polygons representing a. To handle such situations, we employ the following method. We use the south line of $mbb(b)$ ($y = l_1$) as the reference line and calculate the areas between $y = l_1$ and all edges that lie *both* in $N(b)$ and $B(b)$. This area will be denoted by $area((B+N)(b) \cap a)$ and is practically the area of a that lies on $N(b)$ and $B(b)$, i.e., $area((B+N)(b) \cap a) = area(N(b) \cap a) + area(B(b) \cap a)$.

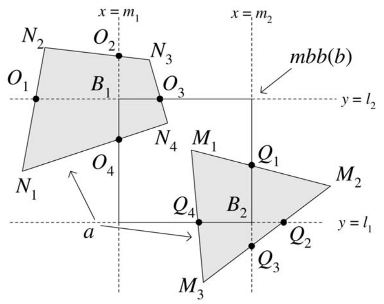


Fig. 9. Computing cardinal direction relations with percentages.

Since $area(N(b) \cap a)$ has been previously computed, we just have to subtract it from $area((B+N)(b) \cap a)$ in order to derive $area(B(b) \cap a)$. For instance, in Fig. 9, we have

$$\begin{aligned} area((B+N)(b) \cap a) &= \left| \sum_{AB \in B(b) \cup N(b)} E_{l_1}(AB) \right| \\ &= | E_{l_1}(O_2N_3) + E_{l_1}(N_3O_3) + E_{l_1}(O_3N_4) \\ &\quad + E_{l_1}(N_4O_4) + E_{l_1}(M_1Q_1) \\ &\quad + E_{l_1}(Q_4M_1) | \\ &= area(\ (O_2N_3O_3N_4O_4) \\ &\quad + (M_1Q_1B_2Q_4)\ ) \end{aligned}$$

and

$$\begin{aligned} area(N(b) \cap a) &= \left| \sum_{AB \in N(b)} E_{l_2}(AB) \right| \\ &= | E_{l_2}(O_2N_3) + E_{l_2}(N_3O_3) | \\ &= area(\ (O_2N_3O_3B_1)\ ). \end{aligned}$$

Therefore,

$$area(B(b) \cap a) = area((B+N)(b) \cap a) - area(N(b) \cap a)$$

holds.

The above method is summarized in the following lemma.

Lemma 2. *Let a and b be two polygons. Let us also assume that every edge of AB of a lies in exactly one tile of b. Then, the area of polygon a that falls in every tile of polygon b can be computed as follows:*

$$area(T(b) \cap a) = \left| \sum_{AB \in T(b)} E'_{m_1}(AB) \right| \ if\ T \in \{NW, W, SW\}$$

$$area(T(b) \cap a) = \left| \sum_{AB \in T(b)} E'_{m_2}(AB) \right| \ if\ T \in \{NE, E, SE\}$$

$$area(S(b) \cap a) = \left| \sum_{AB \in S(b)} E_{l_1}(AB) \right|$$

$$area(N(b) \cap a) = \left| \sum_{AB \in N(b)} E_{l_2}(AB) \right|$$

$$area(B(b) \cap a) = \underbrace{\left| \sum_{AB \in N(b) \cup B(b)} E_{l_1}(AB) \right|}_{area(B(b) \cup N(b))} - \underbrace{\left| \sum_{AB \in N(b)} E_{l_2}(AB) \right|}_{area(N(b))},$$

```
Algorithm COMPUTE-CDR%
Input: Two sets of polygons $S_a$ and $S_b$ representing regions $a, b \in REG^*$ resp.
Output: The cardinal direction relation with percentages $R$, such that $a\ R\ b$ holds.
Method:
Use $S_b$ to compute the minimum bounding box $mbb(b)$ of $b$.
Divide all the edges in $S_a$ so that every new edge lies in exactly one tile of $b$.
Let $y = l_1$, $y = l_2$, $x = m_1$ and $x = m_2$ where $l_1 < l_2$ and $m_1 < m_2$ be the lines forming $mbb(b)$.
$a_{B+N} = a_S = a_{SW} = a_W = a_{NW} = a_N = a_{NE} = a_E = a_{SE} = 0$
For every edge $AB$ of $S_a$
    Let $t$ be the tile of $b$ that $AB$ falls in.
    Case $t$    // Expressions $E'$ and $E$ are defined in Definition 4
        $NW$,$W$,$SW$: $a_t = a_t + E'_{m_1}(AB)$
        $NE$,$E$,$SE$:  $a_t = a_t + E'_{m_2}(AB)$
        $S$:            $a_t = a_t + E_{l_1}(AB)$
        $N$:            $a_t = a_t + E_{l_2}(AB)$; $a_{B+N} = a_{B+N} + E_{l_1}(AB)$
        $B$:            $a_{B+N} = a_{B+N} + E_{l_1}(AB)$
    EndCase
EndFor
$a_B = |a_{B+N}| - |a_N|$
$totalArea = |a_B| + |a_S| + |a_{SW}| + |a_W| + |a_{NW}| + |a_N| + |a_{NE}| + |a_E| + |a_{SE}|$
Return $\frac{100\%}{totalArea} \cdot \begin{bmatrix} |a_{NW}| & |a_N| & |a_{NE}| \\ |a_S| & |a_B| & |a_E| \\ |a_{SW}| & |a_S| & |a_{SE}| \end{bmatrix}$
```

Fig. 10. Algorithm COMPUTE-CDR%.

where $AB \in T(b)$ denotes all the edges of polygon a that fall in the T tile of polygon b.

Algorithm COMPUTE-CDR%, presented in Fig. 10, utilizes Lemma 2 to compute cardinal direction relations with percentages. The following theorem captures the correctness of Algorithm COMPUTE-CDR% and measures its complexity.

Theorem 2. *Algorithm COMPUTE-CDR% is correct, i.e., it returns the cardinal direction relation with percentages between two regions a and b in REG^* that are represented using two sets of polygons S_a and S_b, respectively. The running time of Algorithm COMPUTE-CDR% is $\mathcal{O}(k_a + k_b)$, where k_a (respectively, k_b) is the total number of edges of all polygons in S_a (respectively, S_b).*

Proof. The correctness of Algorithm COMPUTE-CDR% follows from the correctness of Algorithm COMPUTE-CDR and Lemmas 1 and 2. With respect to the running time, Algorithm COMPUTE-CDR starts by dividing the edges of region a using the lines of the minimum bounding box of region b. This can be done in $\mathcal{O}(k_a + k_b)$ time (similarly to Algorithm COMPUTE-CDR). Then, Algorithm COMPUTE-CDR% performs a For loop which examines every edge of S_a. All the checks inside this loop can be done in constant time, thus the complexity of the For loop is $\mathcal{O}(k_a)$ time. Summarizing, the total complexity of Algorithm COMPUTE-CDR% is $\mathcal{O}(k_a + k_b)$ time. □

4 Experimental Evaluation

In Section 3, we have presented Algorithms COMPUTE-CDR and COMPUTE-CDR% that compute cardinal direction relations and cardinal direction relations with percentages, respectively. Moreover, we have briefly discussed the advantages of using Algorithms COMPUTE-CDR and COMPUTE-CDR% over methods that compute cardinal direction relations and are based on polygon clipping. In this section, we experimentally evaluate the performance of our algorithms to support the aforementioned theoretical analysis.

A preliminary evaluation analysis has indicated that Algorithm COMPUTE-CDR% is marginally slower than Algorithm COMPUTE-CDR. Therefore, in our experiments, we have only used Algorithm COMPUTE-CDR% since it provides more informative relations than Algorithm COMPUTE-CDR. We compare Algorithm COMPUTE-CDR% with two algorithms that are based on the popular Sutherland and Hodgman [26] and Liang and Barsky [8] polygon clipping methods. These algorithms solve a special case of the polygon clipping problem. They take as input a polygon a and a *rectangle* b and return a new polygon that represents the part of polygon a that fall inside b. There are also some more recent and more sophisticated algorithms for polygon clipping in the Computational Geometry literature [7], [28]. We do not consider these algorithms since they solve the general polygon clipping problem (i.e., where both a and b are arbitrary polygons) and this functionality is not required to solve the problem of computing cardinal direction relations.

Particularly, the Sutherland and Hodgman method clips the polygon a against a rectangle b as follows: Every edge of rectangle b, if extended to a line, divides the plane into two half-planes. Clipping a polygon against a line involves computing the part of the polygon a that lies on the half-plane that contains b. Based on this observation, the Sutherland and Hodgman method clips polygon a consecutively against each edge of rectangle b. The Liang and Barsky method treats the edges of polygon a and rectangle b as lines of infinite extent. Then, the intersection points of these extended lines are computed and these points are used to determine the segment of each edge that falls inside rectangle b, as well as any additional edges that need to be included in the output polygon.

The three algorithms that we compare were programmed in C and all experiments were performed on an AMD Sempron 2400+ with 512MB of memory running Windows XP. In our experiments, we have used both real and synthetic data. The synthetic data is comprised of four different groups of artificially produced configurations of polygons. Each group contains 20 configurations that are formed by randomly generated polygons all having three,

Data Set	COMPUTE CDR%	Sutherland Hodgman	Liang Barsky
Polygons with 3 edges	10.6%	24.5%	28.2%
Polygons with 4 edges	10.5%	24.2%	30.8%
Polygons with 5 edges	10.5%	24.1%	34.3%
Polygons with 10 edges	10.4%	23.7%	42.5%
Prefectures of Hellas	0.9%	2.0%	2.9%
Land parcels	1.6%	3.4%	4.6%

(a)

Data Set	COMPUTE CDR%	Sutherland Hodgman	Liang Barsky
Polygons with 3 edges	1.0	7.4	14.7
Polygons with 4 edges	1.0	7.8	15.8
Polygons with 5 edges	1.0	7.8	16.1
Polygons with 10 edges	1.0	8.4	18.2
Prefectures of Hellas	1.0	12.3	28.6
Land parcels	1.0	12.0	26.6

(b)

Fig. 11. Summary of results. (a) Additional edges. (b) Normalized time.

four, five, and 10 edges, respectively. Additionally, the number of polygons in the configurations of each group ranges from 25 to 500.

The real data is comprised of two configurations. The first configuration is a map of the prefectures of Hellas. It is formed by 129 polygons with 53 edges on average. The second configuration is a map of land parcels from the Hellenic Cadastre and it is comprised of 1,995 polygons with 17 edges on average.

For every map, we compute the cardinal direction relation between every possible pair of polygons (in this particular map). As we have mentioned in Section 3, both Algorithm COMPUTE-CDR% and polygon clipping-based algorithms introduce additional edges in order to compute cardinal direction relations. Therefore, we calculate the total time needed by the algorithms for every image, as well as the total number of new edges that are introduced. Our results are summarized in Fig. 11. Fig. 11a presents the increase in the total number of polygon edges of an image, as a percentage of the original total number. We can see that Algorithm COMPUTE-CDR% introduces at least two times (respectively, three times) fewer edges than the algorithm based on Sutherland and Hodgman (respectively, Liang and Barsky) clipping method.

Fig. 11b illustrates the running time of the three algorithms that we compare. Algorithm COMPUTE-CDR% is between 7 and 12 times (respectively, 14 and 28) faster than the method based on Sutherland and Hodgman (respectively, Liang and Barsky) clipping algorithm.

For the synthetic data, we also present more detailed results (Fig. 12). For each of the four groups of maps we present a pair of graphs. The left-hand side graphs illustrate the number of additional edges that the three algorithms introduce (in thousands). The right-hand side graphs illustrate the running time of the three algorithms (in milliseconds). For instance, the second pair of graphs presents the additional edges and the running time of the three algorithms for the group of configurations that contains polygons with four edges.

Summarizing, in this section we have compared Algorithm COMPUTE-CDR% and two algorithms that use polygon clipping in order to compute cardinal direction relations. We have experimentally demonstrated that Algorithm COMPUTE-CDR% outperforms the clipping-based methods. In the following section, we present an actual system, CARDIRECT, that incorporates Algorithms COMPUTE-CDR and COMPUTE-CDR%.

5 A Tool for the Manipulation of Cardinal Direction Information

In this section, we will present a tool that implements the aforementioned algorithms for the computation of cardinal direction relations among regions. The tool, CARDIRECT, has been implemented in C++ over the Microsoft Visual Studio toolkit.[1] Using CARDIRECT, the user can specify, edit, and annotate regions of interest over some underlying image (astronomical images, city maps, etc.). Then, the tool automatically computes the cardinal direction relations (with and without percentages) between these regions using the linear algorithms of Section 3. The information concerning the underlying image, the introduced regions, their composing polygons and their relations form the metainformation for the overall configuration. The configuration of the image and the introduced regions is persistently stored in an ASCII file using a simple XML description. Such a representation of data is economical in terms of data size (only a few kilobytes, without any compression). Moreover, the user is allowed to query the stored XML description of the image and retrieve combinations of interesting regions. The queries are expressions in conjunctive normal form, with variables ranging either over regions or other meaningful properties of the polygons of the configuration (e.g., color). The results are regions that fulfill the criteria of the submitted query.

The architecture of CARDIRECT involves

1. a graphical front-end component, through which the user manipulates his configurations and poses queries,
2. a reasoner component, used to compute all the combinations of cardinal direction relations, and
3. a query engine to answer queries.

It is also possible to export the constructed configurations in XML descriptions and to pose queries over them, though the respective components.

The XML description of the exported scenarios is quite simple. A configuration (`Image`) is defined upon an image file (e.g., a map) and is comprised of a set of regions and a set of relations among them. Each region is comprised of a set of polygons of the same color and each polygon is comprised of a set of edges (defined by x and y-coordinates). The direction relations among the different regions are all stored in the XML description of the configuration. More details about the used XML description can be found in [20].

Observe Fig. 13a. In this configuration, the user has opened a map of Ancient Greece at the time of the Peloponnesian war as the underlying image. Then, the user defined three sets of regions:

1. the "Athenean Alliance" in blue, comprising of Attica, the Islands, the regions in the East, Corfu and South Italy,
2. the "Spartan Alliance" in red, comprising of Peloponnesos, Beotia, Crete and Sicily, and
3. the "Pro-Spartan" in black, comprising of Macedonia.

1. See http://www.microsoft.com/visualc/.

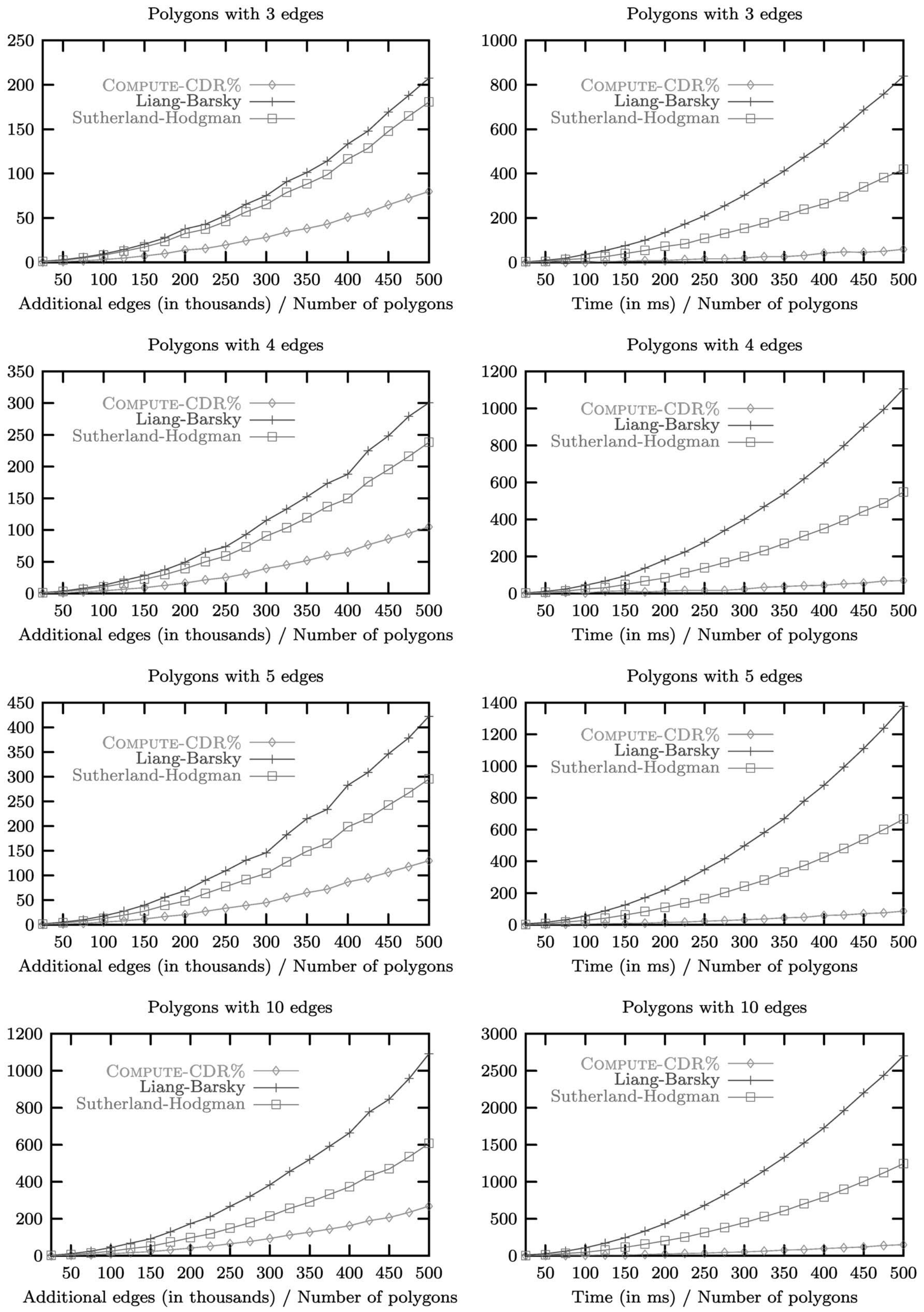
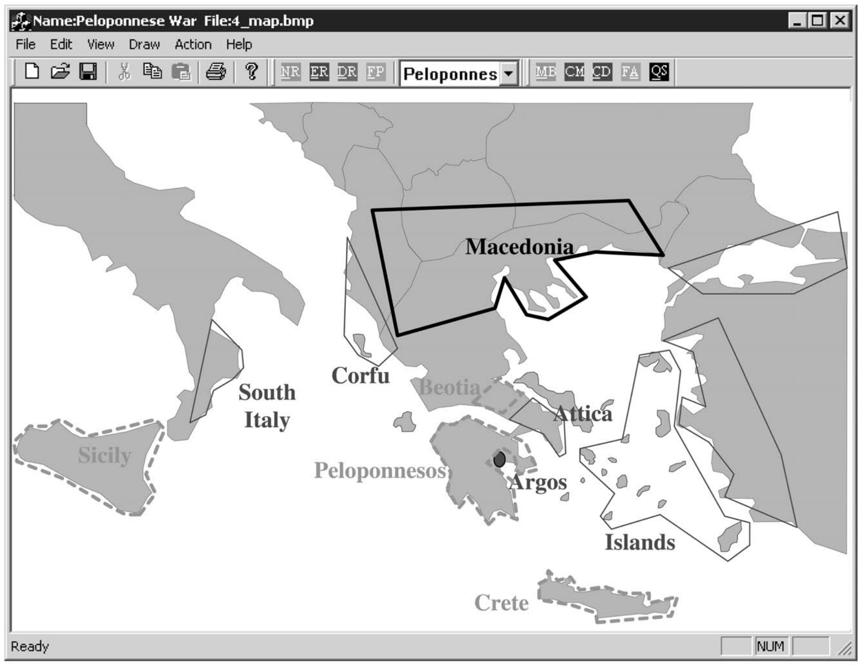


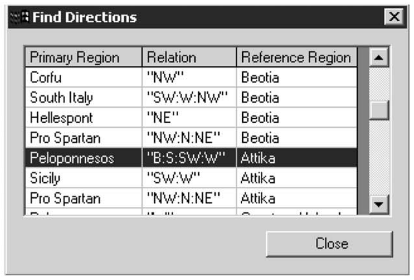
Fig. 12. Detailed results for synthetic data.

To further improve the readability of Fig. 13, gray lines (Spartan Alliance) also appear dashed, black lines (Pro-Spartan) are thicker, while other gray lines (Athenean Alliance) are normal.

Moreover, CARDIRECT can compute the cardinal direction relations and the cardinal direction relations with percentages between the identified regions. In Figs. 13b and 13c, we have calculated the relations between the regions of

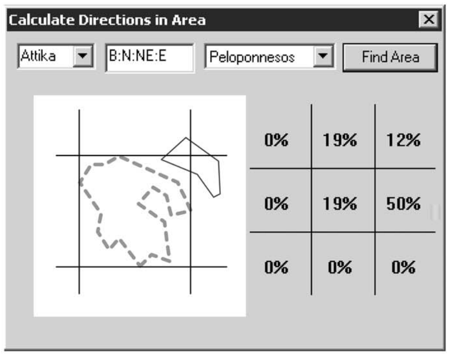


(a)



Primary Region	Relation	Reference Region
Corfu	"NW"	Beotia
South Italy	"SW:W:NW"	Beotia
Hellespont	"NE"	Beotia
Pro Spartan	"NW:N:NE"	Beotia
Peloponnesos	"B:S:SW:W"	Attika
Sicily	"SW:W"	Attika
Pro Spartan	"NW:N:NE"	Attika

(b)



0%	19%	12%
0%	19%	50%
0%	0%	0%

(c)

Fig. 13. Using CARDIRECT to annotate images.

Fig. 13a. For instance, Peloponnesos is $B:S:SW:W$ of Attica (Fig. 13b) while Attica is

$$\begin{bmatrix} 0\% & 19\% & 12\% \\ 0\% & 19\% & 50\% \\ 0\% & 0\% & 0\% \end{bmatrix}$$

of Peloponnesos (Fig. 13c).

The query language that we employ is based on the following simple model. Let A be a set of regions in REG^* over a configuration. Let C be a finite set of thematic attributes for the regions of REG^* (e.g., the color of each region) and f a function, $f: REG^* \rightarrow dom(C)$, practically relating each of the regions with a value over the domain of C (e.g., the fact that the Spartan Alliance is colored).

A *query condition* over variables $x_1, \dots, x_k$ is a conjunction of the following types of formulae $x_i = a$, $f(x_i) = c$, $x_i \, R \, x_j$, where $1 \le i, j \le k$, $a \in A$ is a region of the configuration, $c \in dom(C)$ is a value of a thematic attribute, and $R \in 2^{\mathcal{D}*}$ is a (possibly disjunctive) cardinal direction relation. A *query* q over variables $x_1, \dots, x_k$ is a formula of the form $q = \{(x_1, \dots, x_k) \mid \phi(x_1, \dots, x_k)\}$, where $\phi(x_1, \dots, x_k)$ is a query condition.

Intuitively, the query returns a set of regions in the configuration of an image that satisfy the query condition, which can take the form of:

1. a cardinal direction constraint between the query variables (e.g., $x_1 \; B{:}SE{:}S \; x_2$),
2. a restriction in the thematic attributes of a variable (e.g., $color(x_1) = blue$), and
3. direct reference of a particular region (e.g., $x_1 = Attica$).

For instance, over the configuration of Fig. 13a, we can pose the following query: "Find all regions of the Athenean Alliance which are surrounded by a region in the Spartan Alliance." This query can be expressed as

$$q = \{(a, b) \mid color(a) = red,\; color(b) = blue,\; a\; S{:}SW{:}W{:}NW{:}N{:}NE{:}E{:}SE\; b\}.$$

We can express and execute the above query using CARDIRECT. Moreover, CARDIRECT allows the user to store the query using XML. For instance, the above query can be expressed in XML as follows:

```
<?query version="1.0" encoding="UTF-8"?>
<Image id="image1" >
 <Variables>
  <name="a" color="Red"/>
  <name="b" color="Blue"/>
 </Variables>
 <Relations>
  <primary="a" reference="b"
   relations="S:SW:W:NW:N:NE:E:SE"/>
 </Relations>
</Image>
```

For the regions of Fig. 13a, only Peloponnesos and Argos satisfy the query. Peloponnesos, which belongs to the Spartan Alliance presented in dashed lines, is $S{:}SW{:}W{:}NW{:}N{:}NE{:}E{:}SE$ of Argos, which belongs to the Athenean Alliance presented in normal lines.

6 CONCLUSIONS AND FUTURE WORK

In this paper, we have addressed the problem of efficiently computing the cardinal direction relations between regions that are composed of sets of polygons 1) by presenting two linear algorithms for this task and 2) by explaining their incorporation into an actual system. The input of both algorithms is two sets of polygons representing the two involved regions. The first of the proposed algorithms computes the cardinal direction relations between the input regions while the second computes the cardinal direction relations with percentages between the input regions. To the best of our knowledge, these are the first algorithms that address the aforementioned problem. Moreover, we have experimentally evaluated the proposed algorithms and demonstrated their improved performance over algorithms based on polygon clipping methodologies. The algorithms have been implemented and embedded in an actual system, CARDIRECT, which allows the user to specify, edit, and annotate regions of interest in an image. Then, CARDIRECT automatically computes the cardinal direction relations between these regions. The configuration of the image and the introduced regions are persistently stored using a simple XML description. The user is allowed to query the stored XML description of the image and retrieve combinations of interesting regions on the basis of the query.

Although this part of our research addresses the problem of relation computation to a sufficient extent, there are still open issues for future research. An interesting topic is the possibility of combining the presented cardinal direction relations model with topological [3] and distance relations [4]. Another issue is the possibility of combining the underlying model with extra thematic information and the enrichment of the employed query language on the basis of this combination. Finally, a long-term goal would be the integration of CARDIRECT with image segmentation software [27], which would assist the user to specify interesting regions in an image, thus, providing a more automated environment for the management of image configurations.

ACKNOWLEDGMENTS

The authors would like to thank the Hellenic Cadastre (http://www.ktimatologio.gr/) for providing us with sample data. They would also like to thank Kostas Patroumpas for helping us cope with these data.

REFERENCES

[1] E. Clementini, P. Di Felice, and G. Califano, "Composite Regions in Topological Queries," *Information Systems,* vol. 7, pp. 759-594, 1995.

[2] Z. Cui, A.G. Cohn, and D.A. Randell, "Qualitative and Topological Relationships in Spatial Databases," *Proc. Int'l Symp. Advances in Spatial Databases (SSD '93),* pp. 296-315, 1993.

[3] M.J. Egenhofer, "Reasoning about Binary Topological Relationships," *Proc. Int'l Symp. Advances in Spatial Databases (SSD '91),* pp. 143-160, 1991.

[4] A.U. Frank, "Qualitative Spatial Reasoning about Distances and Directions in Geographic Space," *J. Visual Languages and Computing,* vol. 3, pp. 343-371, 1992.

[5] A.U. Frank, "Qualitative Spatial Reasoning: Cardinal Directions as an Example," *Int'l J. GIS,* vol. 10, no. 3, pp. 269-290, 1996.

[6] R. Goyal, "Similarity Assessment for Cardinal Directions between Extended Spatial Objects," PhD thesis, Dept. of Spatial Information Science and Eng., Univ. of Maine, Apr. 2000.

[7] G. Greiner and K. Hormann, "Efficient Clipping of Arbitrary Polygons," *ACM Trans. Graphics,* vol. 17, no. 2, pp. 71-83, 1998.

[8] Y.-D. Liang and B.A. Barsky, "A New Concept and Method for Line Clipping," *ACM Trans. Graphics,* vol. 3, no. 1, pp. 868-877, 1984.

[9] G. Ligozat, "Reasoning about Cardinal Directions," *J. Visual Languages and Computing,* vol. 9, pp. 23-44, 1998.

[10] S. Lipschutz, *Set Theory and Related Topics.* McGraw Hill, 1998.

[11] P.-G. Maillot, "A New, Fast Method For 2D Polygon Clipping: Analysis and Software Implementation," *ACM Trans. Graphics,* vol. 11, no. 3, pp. 276-290, 1992.

[12] A. Mukerjee and G. Joe, "A Qualitative Model for Space," *Proc. AAAI '90,* pp. 721-727, 1990.

[13] J. O'Rourke, *Computational Geometry in C.* Cambridge Univ. Press, 1994.

[14] D. Papadias, Y. Theodoridis, T. Sellis, and M.J. Egenhofer, "Topological Relations in the World of Minimum Bounding Rectangles: A Study with R-Trees," *Proc. ACM SIGMOD Conf.,* pp. 92-103, 1995.

[15] D.J. Peuquet and Z. Ci-Xiang, "An Algorithm to Determine the Directional Relationship between Arbitrarily-Shaped Polygons in the Plane," *Pattern Recognition,* vol. 20, no. 1, pp. 65-74, 1987.

[16] F. Preparata and M. Shamos, *Computational Geometry: An Introduction.* Springer Verlag, 1985.

[17] J. Renz and B. Nebel, "On the Complexity of Qualitative Spatial Reasoning: A Maximal Tractable Fragment of the Region Connection Calculus," *Artificial Intelligence,* vols. 1-2, pp. 95-149, 1999.

[18] P. Rigaux, M. Scholl, and A. Voisard, *Spatial Data Bases.* Morgan Kaufman, 2001.

[19] A.P. Sistla, C. Yu, and R. Haddad, "Reasoning about Spatial Relationships in Picture Retrieval Systems," *Proc. Very Large Data Bases Conf.,* pp. 570-581, 1994.

[20] S. Skiadopoulos, C. Giannoukos, N. Sarkas, P. Vassiliadis, T. Sellis, and M. Koubarakis, "Computing and Managing Cardinal Direction Relations (Extended Report)," Technical Report TR-2004-10, Nat'l Technical Univ. of Athens, 2003, http://www.dblab.ece.ntua.gr/pubs.

[21] S. Skiadopoulos, C. Giannoukos, P. Vassiliadis, T. Sellis, and M. Koubarakis, "Computing and Handling Cardinal Direction Information," *Proc. Int'l Conf. Extending Database Technology,* pp. 329-347, 2004.

[22] S. Skiadopoulos and M. Koubarakis, "Composing Cardinal Directions Relations," *Proc. Int'l Symp. Spatial and Temporal Databases,* pp. 299-317, July 2001.

[23] S. Skiadopoulos and M. Koubarakis, "Qualitative Spatial Reasoning with Cardinal Directions," *Proc. Int'l Conf. Principles & Practice of Constraint Programming,* pp. 341-355, Sept. 2002.

[24] S. Skiadopoulos and M. Koubarakis, "Composing Cardinal Direction Relations," *Artificial Intelligence,* vol. 152, no. 2, pp. 143-171, 2004.

[25] S. Skiadopoulos and M. Koubarakis, "On the Consistency of Cardinal Directions Constraints," *Artificial Intelligence,* vol. 163, no. 1, pp. 91-135, 2005.

[26] I. Sutherland and G. Hodgman, "Reentrant Polygon Clipping," *Comm. ACM,* vol. 17, no. 1, pp. 32-42, 1974.

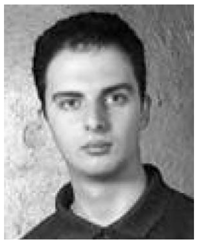
[27] A.S. Szalay, J. Gray, A. Thakar, P.Z. Kunszt, T. Malik, J. Raddick, C. Stoughton, and J. van den Berg, "The SDSS Skyserver: Public Access to the Sloan Digital Sky Server Data," *Proc. ACM SIGMOD Conf.,* pp. 570-581, June 2002.

[28] B.R. Vatti, "A Generic Solution to Polygon Clipping," *Comm. ACM,* vol. 35, no. 7, pp. 56-63, 1992.

[29] M. Zeiler, *Modelling Our World. The ESRI Guide to Geodatabase Design.* ESRI-Press, 1999.

[30] K. Zimmermann, "Enhancing Qualitative Spatial Reasoning—Combining Orientation and Distance," *Proc. Conf. Spatial Information Theory (COSIT '93),* pp. 69-76, 1993.

Spiros Skiadopoulos received the diploma and PhD degree from the National Technical University of Athens and the MSc degree from UMIST. His research interests include spatial and temporal databases, constraint databases, query evaluation and optimization, and data warehouses. He has published more than 25 papers in international refereed journals and conferences.



Christos Giannoukos received the diploma from the National Technical University of Athens. His research interests include spatial databases, and spatial representation and reasoning. He is currently working at a technical company concerning mobile telephony.

Nikos Sarkas received the diploma from the National Technical University of Athens and he is currently working toward a PhD degree. His research interests include spatial databases, spatial reasoning, and modeling information with constrains.

Panos Vassiliadis received the PhD degree from the National Technical University of Athens in 2000. He is a lecturer at the University of Ioannina. His research interests include data warehousing, Web services, and database design and modeling. He has published more than 25 papers in refereed journals and international conferences in the above areas.

Timos Sellis received the diploma degree in electrical engineering in 1982 from the National Technical University of Athens (NTUA), Greece. In 1983, he received the MSc degree from Harvard University and the PhD degree from the University of California at Berkeley in 1986, where he was a member of the INGRES group, both in computer science. In 1986, he joined the Department of Computer Science at the University of Maryland, College Park, as an assistant professor, and became an associate professor in 1992. Between 1992 and 1996, he was an associate professor in the Computer Science Division of NTUA, where he is currently a full professor. He is also the head of the Knowledge and Database Systems Laboratory at NTUA. His research interests include peer-to-peer database systems, data warehouses, the integration of Web and databases, and spatial database systems. He has published more than 120 articles in refereed journals and international conferences in the above areas and has been an invited speaker in major international events. He was the organization chair for the Very Large Data Bases Conference in 1997 and pc chair for ACM SIGMOD 2001. Dr. Sellis is a recipient of the prestigious Presidential Young Investigator (PYI) award given by the President of USA to the most talented new researchers (1990), and of the VLDB 1997 10 Year Paper Award for his work on spatial databases. He was the president of the National Council for Research and Technology of Greece (2001-2003) and a member of the VLDB Endowment (1996-2000).

Manolis Koubarakis received a degree in mathematics from the University of Crete, the MSc degree in computer science from the University of Toronto, and the PhD degree in computer science from the National Technical University of Athens. He joined the Department of Electronic and Computer Engineering, Technical University of Crete in January 1999. He is currently an associate professor and director of the Intelligent Systems Laboratory (www.intelligence.tuc.gr). Before coming to Crete, he held positions at Imperial College, London (research associate) and UMIST, Manchester (lecturer). Dr. Koubarakis has published papers in the areas of database and knowledge-base systems, constraint programming, intelligent agents, semantic Web, and peer-to-peer computing. More information is available at www.intelligence.tuc.gr/~manolis.