Available online at www.sciencedirect.com
SCIENCE DIRECT®
Artificial Intelligence 163 (2005) 91–135
Artificial Intelligence
www.elsevier.com/locate/artint

On the consistency of cardinal direction constraints ☆

Spiros Skiadopoulos [a,*], Manolis Koubarakis [b]

[a] *Knowledge and Database Systems Laboratory, School of Electrical and Computer Engineering, National Technical University of Athens, Zographou, 157 73 Athens, Greece*
[b] *Intelligent Systems Laboratory, Department of Electronic and Computer Engineering, Technical University of Crete, Chania, 731 00 Crete, Greece*

Received 3 December 2003; accepted 18 October 2004

Available online 15 December 2004

Abstract

We present a formal model for qualitative spatial reasoning with cardinal directions utilizing a coordinate system. Then, we study the problem of checking the consistency of a set of cardinal direction constraints. We introduce the first algorithm for this problem, prove its correctness and analyze its computational complexity. Utilizing the above algorithm, we prove that the consistency checking of a set of basic (i.e., non-disjunctive) cardinal direction constraints can be performed in $\mathcal{O}(n^5)$ time. We also show that the consistency checking of a set of unrestricted (i.e., disjunctive and non-disjunctive) cardinal direction constraints is NP-complete. Finally, we briefly discuss an extension to the basic model and outline an algorithm for the consistency checking problem of this extension.

Keywords: Cardinal direction relations; Spatial constraints; Consistency checking; Qualitative spatial reasoning

☆ This is a greatly revised and extended version of a paper which appears in Proc. of CP-02, Lecture Notes in Comput. Sci., vol. 2470, Springer, Berlin, 2002, pp. 341–355.

* Corresponding author.
E-mail addresses: spiros@dblab.ece.ntua.gr (S. Skiadopoulos), manolis@intelligence.tuc.gr (M. Koubarakis).
URLs: http://www.dblab.ece.ntua.gr/~spiros (S. Skiadopoulos), http://www.intelligence.tuc.gr/~manolis (M. Koubarakis).

0004-3702/$ – see front matter
doi:10.1016/j.artint.2004.10.010

1. Introduction

Qualitative spatial reasoning has received a lot of attention in the areas of Geographic Information Systems [13–15], Artificial Intelligence [6,8,13,29,36–38], Databases [32] and Multimedia [44]. Qualitative spatial reasoning problems have recently been posed as constraint satisfaction problems and solved using traditional algorithms, e.g., path-consistency [38]. One of the most important problems in this area is the identification of useful and tractable classes of spatial constraints and the study of efficient algorithms for consistency checking, minimal network computation and so on [38]. Several kinds of useful spatial constraints have been studied so far, e.g., topological constraints [5,6,12,13,36–38], cardinal direction constraints [17,25,41] and qualitative distance constraints [14,48].

In this paper, we concentrate on *cardinal direction constraints* [17,25,32]. Cardinal direction constraints describe how regions of space are placed relative to one another utilizing a co-ordinate system (e.g., region a is *north of* region b). Currently, the model of Goyal and Egenhofer [16,17] and Skiadopoulos and Koubarakis [40,42] is one of the most expressive models for qualitative reasoning with cardinal directions. The model that we will present in this paper is closely related to the above model but there is a significant difference. The model of [16,17,40,42] basically deals with extended regions that are connected and have connected boundaries while our approach allows regions to be disconnected and have holes. The regions that we consider are very common in Geography, Multimedia and Image Databases [4,7,44]. For example, countries are made up of separations (islands, exclaves, external territories) and holes (enclaves) [7].

We will study the problem of checking the consistency of a given set of cardinal direction constraints in our model. Checking the consistency of a set of constraints in a model of spatial information is a fundamental problem and has received a lot of attention in the literature [25,32,38]. Algorithms for consistency checking are of immediate use in various situations including:

- Propagating relations and detecting inconsistencies in a given set of spatial relations [25,38].
- Preprocessing spatial queries so that inconsistent queries are detected or the search space is pruned [31].

The technical contributions of this paper can be summarized as follows:

1. We present a formal model for qualitative reasoning about cardinal directions. This model is related to the model of [16,17,40,42] and is currently one of the most expressive models for qualitative reasoning with cardinal directions. The proposed model formally defines cardinal direction relations on extended regions that can be disconnected and have holes. The definition of a cardinal direction relation uses two types of constraints: order constraints (e.g., $a < b$) and set-union constraints (e.g., $a = a_1 \cup a_2$).
2. We use our formal framework to study the problem of checking the consistency of a given set of cardinal direction constraints in the proposed model. We present the first algorithm for this problem and prove its correctness. The algorithm is interesting and has a non-trivial step where we show how to avoid using explicitly the obvious

but computational costly set-union constraints resulting from the definition of cardinal direction relations.

3. We present an analysis of the computational complexity of the consistency checking problem for cardinal direction constraints. We show that the aforementioned problem for a given set of *basic* (i.e., non-disjunctive) cardinal direction constraints in n variables can be solved in $\mathcal{O}(n^5)$ time. Moreover, we prove that checking the consistency of a set of unrestricted (i.e., disjunctive and non-disjunctive) cardinal direction constraints is NP-complete.
4. Finally, we consider the consistency checking problem of a set of cardinal direction constraints expressed in an interesting extension of the basic model and outline an algorithm for this task. This extension considers not only extended regions but also points and lines.

The rest of the paper is organized as follows. In Section 2, we survey related work. Section 3 presents the cardinal direction relations and constraints of our model. In Section 4, we discuss the consistency checking of a set of basic cardinal direction constraints (expressed in the model of Section 3) and we present the first algorithm for this task. Section 5 studies the computational complexity of the consistency checking problem of basic and unrestricted sets of cardinal directions constraints. In Section 6, we outline algorithms for the consistency checking for an interesting extension of the basic cardinal direction model that we have already completed. Finally, Section 7 offers conclusions and proposes future directions.

2. Related work

Qualitative spatial reasoning forms an important part of the commonsense reasoning required for building successful intelligent systems [10]. Most researchers in qualitative spatial reasoning have dealt with three main classes of spatial information: topological, directional and distance. *Topological* constraints describe how the boundaries, the interiors and the exteriors of two regions relate [5,6,12,36–38]. For instance, if a and b are regions then *a includes b* and *a externally connects with b* are topological constraints. *Directional* (or *orientation*) constraints describe where regions are placed relative to one another [1,13, 15,17,25,32,40,41]. For instance, *a north b* and *a southeast b* are directional constraints. Finally, *distance* constraints describe the relative distance of two regions [14,48]. For instance, *a is far from b* and *a is close to b* are distance constraints.

In this paper, we concentrate on *cardinal direction constraints* [17,25,32,40]. Earlier qualitative models for cardinal direction relations approximate a spatial region by a representative point (most commonly the centroid) or by a representative box (most commonly the minimum bounding box) [14,15,20,25,29,32].

Depending on the particular spatial configuration these approximations may be too crude [16,17]. Thus, expressing direction relations on these approximations can be misleading and contradictory (related observations are made in [28,34,45]). For instance, with respect to the point-based approximation Spain is northeast of Portugal. Most people would agree that “northeast” does not describe accurately the relation between Spain and Portu-

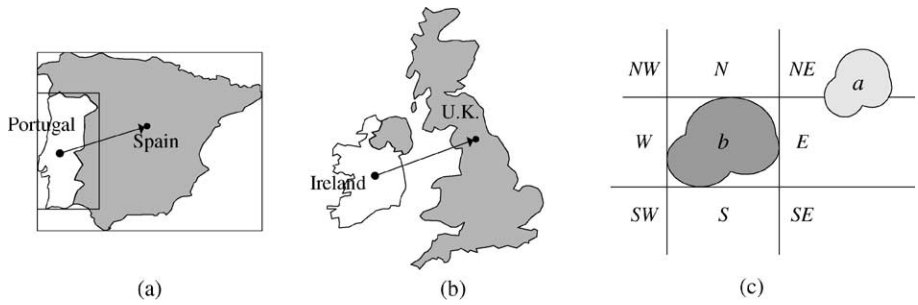

Fig. 1. Problems with point and minimum bounding box approximations.

gal on a map (see Fig. 1(a)). Similar examples are very common in geography. Consider also the direction relation between Ireland and the UK (Fig. 1(b)). Summarizing, there is a demand for the formulation of a model that expresses direction relations between extended objects that overcomes the limitations of the point-based and box-based approximation models.

With the above problem in mind, Goyal and Egenhofer [16,17] and Skiadopoulos and Koubarakis [40,42] presented a model in which we can express the cardinal direction relation of a region a with respect to a region b, by approximating b (using its minimum bounding box) while using the exact shape of a. Informally, the above model divides the space around the reference region b, using its minimum bounding box, into nine areas and records the areas where the primary region a falls into (Fig. 1(c)). This gives a direction relation between the primary and the reference region. Relations in the above model are clearly more expressive than point and box-based models. The model of [17,40] deals with connected regions with a connected boundary. The model that we will present in Section 3, is a variation of the original model of [17,40] that allows regions to be disconnected and have holes. Such regions are very common in Geography, Multimedia and Image Databases [4,7,44]. For instance, the UK is made up from two separated territories: Northern Ireland and Great Britain (Fig. 1(b)).

The consistency checking problem has been studied in detail for all the above classes of spatial constraints. For instance, consistency checking has been examined in great extent for topological constraints [19,38], point-based direction constraints [25] and box-based direction constraints [3,20]. Apart form spatial relations, consistency checking has also been studied for other qualitative relations. For example, Nebel and Bürckert [30] consider this problem for the 13 interval relations of Allen [2] and van Beek [46] for the relations of point algebra.

Typically, for the above cases, the consistency checking problem has been posed as a constraint satisfaction problem and solved using traditional algorithms like path-consistency [38]. For all the above qualitative (spatial and temporal) models, checking the consistency of a set of constraints can be done with a path-consistency method based on composition. Unfortunately, in the case of cardinal direction relations studied in this paper, composition cannot be used to decide consistency [42]. In the following section, we will first formally define cardinal direction constraints and then present a direct algorithm that checks the consistency of a given set of cardinal direction constraints.

3. A formal model for cardinal direction information

We consider the Euclidean space $\mathbb{R}^2$. *Regions* are defined as non-empty and bounded sets of points in $\mathbb{R}^2$. Let a be a region. The *projection* of region a on the x-axis, denoted by $\Pi_x(a)$, is defined as the set of the x-coordinates of all the points in a. Similarly, we can define the projection of region a on the y-axis, denoted by $\Pi_y(a)$. The projection on the x-axis (or y-axis) of a disconnected region is, in general, a bounded set of real numbers. If a region is connected then its projection on the x-axis (respectively y-axis) forms a single interval on the x-axis (respectively y-axis).

A real number $m \in \mathbb{R}$, is a *lower bound* of a set of real numbers I iff $m \leqslant x$ for all $x \in I$. If some lower bound of I is greater than every other lower bound of I, then it is called the *greatest lower bound* or the *infimum* and is denoted by $inf(I)$. Similarly, we can define the *least upper bound* or the *supremum* of a set of real numbers I, denoted by $sup(I)$ [26]. The infimum and the supremum of a set of real numbers are called its *endpoints*.

For clarity, we will denote the greatest lower bound of the projection of region a on the x-axis (i.e., $inf(\Pi_x(a))$) by $inf_x(a)$. Similarly, $sup_x(a)$, $inf_y(a)$ and $sup_y(a)$ are shortcuts for $sup(\Pi_x(a))$, $inf(\Pi_y(a))$ and $sup(\Pi_y(a))$ respectively.

Let a be a region. We say that a is a *box* iff a is a rectangular region formed by the straight lines $x = c_1$, $x = c_2$, $y = c_3$ and $y = c_4$ where c_1, c_2, c_3 and c_4 are real constants such that $c_1 \leqslant c_2$ and $c_3 \leqslant c_4$. Moreover, iff $c_1 < c_2$ and $c_3 < c_4$ hold, we say that a is a *non-trivial box*. A box is *trivial* if it is a point or a vertical line segment or a horizontal line segment.

The *minimum bounding box* of a region a, denoted by $mbb(a)$, is the box formed by the straight lines $x = inf_x(a)$, $x = sup_x(a)$, $y = inf_y(a)$ and $y = sup_y(a)$ (see Fig. 2). Obviously, the projections on the x-axis (respectively y-axis) of a region and its minimum bounding box have the same endpoints.

We will consider throughout the paper the following types of regions:

- Regions that are homeomorphic to the *closed unit disk* (i.e., the set $\{(x, y)\colon x^2 + y^2 \leqslant 1\}$). The set of these regions will be denoted by *REG*. Regions in *REG* are *closed*, *connected* and have *connected boundaries* (for definitions see [9,27]). Class *REG* excludes

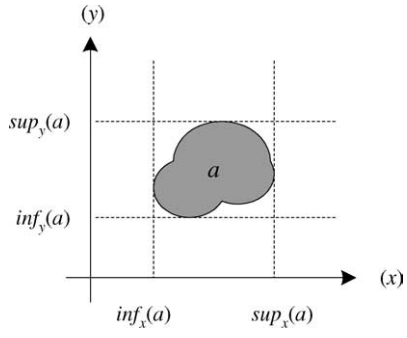

Fig. 2. A region and its bounding box.

disconnected regions, regions with holes, points, lines and regions with emanating lines. Connected regions have been previously studied in [17,35,42].

- Regions in *REG* cannot model the variety and complexity of geographic entities [7]. Thus, we extend class *REG* in order to accommodate *disconnected regions* and *regions with holes*. The set of these regions will be denoted by REG^*. A region a belongs to set REG^* iff there exists a finite set of regions $a_1, \ldots, a_n \in REG$ such that $a = a_1 \cup \cdots \cup a_n$, i.e., set REG^* contains all regions that can be formed by a finite union of regions in *REG*. Set REG^* is a natural extension of *REG* which is useful to model (possibly disconnected) land parcels and countries in Geographic Information Systems [7,12,14] or areas of an image containing similar chromatic arrangements [4]. Notice that the results of Sections 4, 5 and 6 are not affected if we consider regions that are homeomorphic to the *open unit disk* (as in [35]).
- The last class of regions that we consider is an extension that covers arbitrary shapes of $\mathbb{R}^2$. Regions in $\mathbb{R}^2$ can be regions in REG^* but can also be *points*, *lines* and *regions with emanating lines*.

In Fig. 3, regions a, b_1, b_2 and b_3 are in *REG* (also in REG^*) and region $b = b_1 \cup b_2 \cup b_3$ is in REG^* but not in *REG*. Notice that region b is disconnected and has a hole. Fig. 4 presents regions that are not in *REG* and REG^*. Points (Fig. 4(a)), lines (Fig. 4(b)) and regions with emanating lines (Figs. 4(c)–(d)) are not homeomorphic to the closed unit disk. All regions of Fig. 4 are naturally in $\mathbb{R}^2$.

In the rest of this section, we will define a model that expresses cardinal direction relations between regions in REG^*. Then in Sections 4 and 5, we will study the problem of checking the consistency of a given set of cardinal direction constraint in this model. The aforementioned problem for regions in $\mathbb{R}^2$ will be discussed in Section 6.

The following straightforward proposition expresses an important property of regions in REG^*.

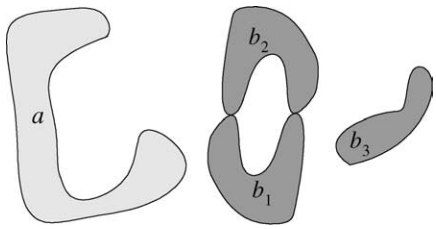

Fig. 3. Regions.

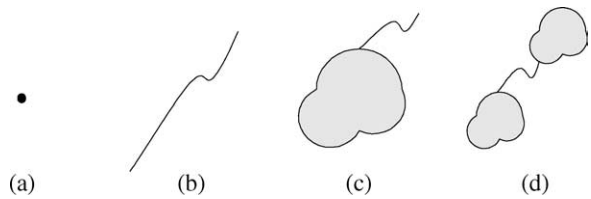

Fig. 4. Regions not in REG^*.

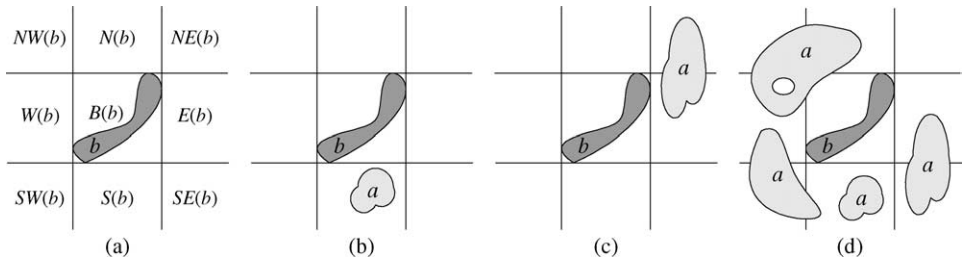

Fig. 5. Reference tiles and relations.

Proposition 1. *If $a \in REG^*$ then $mbb(a)$ is a non-trivial box. Equivalently, the following inequalities hold*:

$$inf_x(a) < sup_x(a) \quad \textit{and} \quad inf_y(a) < sup_y(a).$$

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 *to* which the relation is described) while region a will be called the *primary* region (i.e., the region *from* which the relation is described) [17]. The axes forming the minimum bounding box of the reference region b divide the space into 9 tiles (Fig. 5(a)). 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. Notice that

- all tiles are closed,
- all tiles but $B(b)$ are unbounded,
- the union of all 9 tiles is $\mathbb{R}^2$, and
- two distinct tiles have disjoint interiors but may share point in their boundaries, for instance, $W(b)$ and $B(b)$ share the left-side of the minimum bounding box of b.

If a primary region a is included (in the set-theoretic sense) in tile $S(b)$ of some reference region b (Fig. 5(b)) 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. Notice that, despite the fact that some tiles have common boundaries, we can always determine the tile of the reference region b that a given primary region $a \in REG^*$ falls in because class REG^* does not include points and lines. This is so because only points and lines can be in two tiles (e.g., W and B) at the same time, thus by excluding these regions from our domain REG^* we achieve disjointness of relations.

If a primary region a lies partly in the area $NE(b)$ and partly in the area $E(b)$ of some reference region b (Fig. 5(c)) then we say that *a is partly northeast and partly east of b* and we write $a\ NE{:}E\ b$.

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

Definition 1. A *basic cardinal direction relation* is an expression $R_1{:}\cdots{:}R_k$ where

(i) $1 \leqslant k \leqslant 9$,
(ii) $R_1, \ldots, R_k \in \{B,\ S,\ SW,\ W,\ NW,\ N,\ NE,\ E,\ SE\}$, and
(iii) $R_i \neq R_j$ for every i, j such that $1 \leqslant i, j \leqslant k$ and $i \neq j$.

A basic cardinal direction relation $R_1{:}\cdots{:}R_k$ is called *single-tile* if $k = 1$; otherwise it is called *multi-tile*.

Example 1. The following are basic cardinal direction relations:

$$S, \quad NE{:}E \quad \text{and} \quad B{:}S{:}SW{:}W{:}NW{:}N{:}E{:}SE.$$

The first relation is single-tile while the others are multi-tile. Regions involved in these relations are shown in Figs. 5(b), 5(c) and 5(d) respectively.

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 $S{:}B{:}W$. We avoid using set-theoretic notation for basic relation and reserve this for disjunctive ones (see next section). The readers should also be aware that for a basic relation such as $B{:}S{:}W$, we will often refer to B, S and W as its *tiles*.

3.1. Defining basic cardinal direction relations formally

Let us first start by formally defining the single-tile cardinal direction relations of our model.

Definition 2. Let a and b be two regions in REG^*. Relations B, S, SW, W, NW, N, NE, E and SE are defined as follows:

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

Using the above single-tile relations and set-union, we can define all multi-tile ones. For instance, relation $NE{:}E$ (Fig. 6(a)) and relation $B{:}S{:}SW{:}W{:}NW{:}N{:}E{:}SE$ (Fig. 6(b)) are defined as follows:

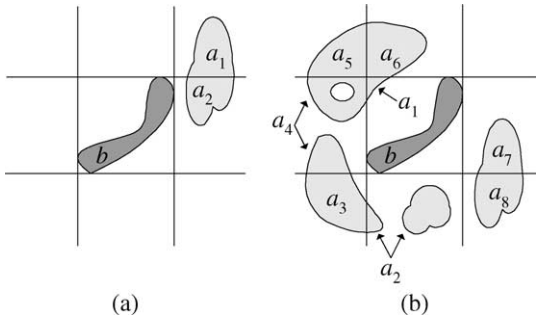

Fig. 6. Relations and component variables.

a *NE*:*E* b	iff	there exist regions a_1 and a_2 in REG^* such that $a = a_1 \cup a_2$, a_1 *NE* b and a_2 *E* b.
a *B*:*S*:*SW*:*W*:*NW*:*N*:*SE*:*E* b	iff	there exist regions $a_1, \ldots, a_8$ in REG^* such that $a = a_1 \cup a_2 \cup a_3 \cup a_4 \cup a_5 \cup a_6 \cup a_7 \cup a_8$, a_1 *B* b, a_2 *S* b, a_3 *SW* b, a_4 *W* b, a_5 *NW* b, a_6 *N* b, a_7 *SE* b and a_8 *E* b.

In general, each multi-tile cardinal direction relation is defined as follows.

Definition 3. Let a and b be two regions in REG^*. For $2 \leqslant k \leqslant 9$, $a\ R_1{:}\cdots{:}R_k\ b$ holds 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 $a = a_1 \cup \cdots \cup a_k$.

The variables $a_1, \ldots, a_k$ in any equivalence such as the above (which defines a basic cardinal direction relation) will be called the *component variables* corresponding to variable a. Notice that for every i, j such that $1 \leqslant i, j \leqslant k$ and $i \neq j$, regions a_i and a_j have disjoint interiors but may share points in their boundaries (see Fig. 6).

The set of basic cardinal direction relations in our model 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. Elements of $\mathcal{D}^*$ can be used to represent *definite information* about cardinal directions, e.g., $a\ N\ b$. Notice the difference between the model presented in this section and the proposal of [17,42] that deals only with the connected regions of *REG*. Our approach accommodates a wider set of region (i.e., regions in REG^*) that also allows regions to be disconnected and have holes. As a result we have 511 relations while the model of [17,42] has 218. This enables us to express several natural spatial arrangements (e.g., $a\ S{:}W\ b$ or $a\ S{:}N\ b$) that are not possible in [17,42].

Using the 511 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 *cardinal direction relations* and can be used to represent not only definite but also *indefinite information* about cardinal directions, e.g., $a\ \{S,\ W\}\ b$ denotes that region a is south *or* west of region b, i.e., $(a\ S\ b) \vee (a\ W\ b)$.

Notice the difference between the basic cardinal direction relation $S{:}W$ and the disjunctive cardinal direction relation $\{S, W\}$. Expression $a\ S{:}W\ b$ denotes that region a lies partly in $S(b)$ and partly in $W(b)$ tile of b (definite information). On the other hand, expression

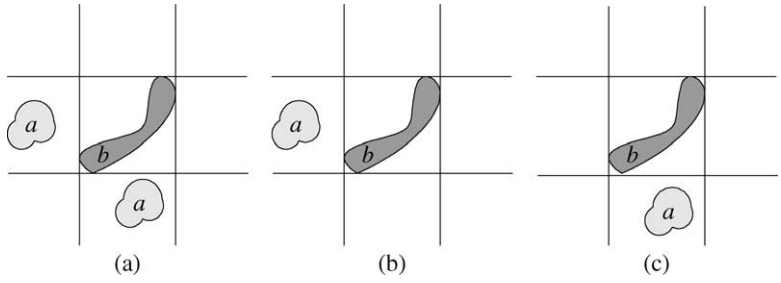

Fig. 7. Constraints $a\ S{:}W\ b$ and $a\ \{S, W\}\ b$.

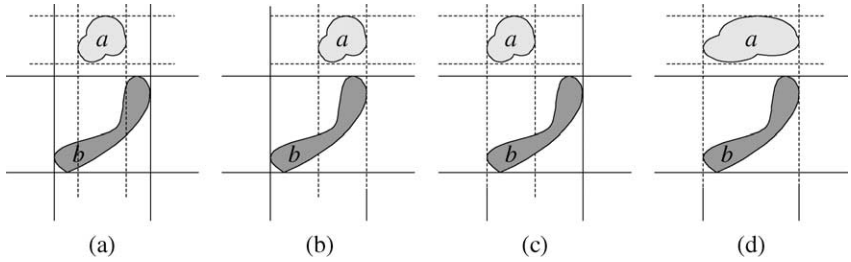

Fig. 8. Members of $inv(N)$.

$a\ \{S, W\}\ b$ denotes that region a lies entirely either in $S(b)$ or $W(b)$ tile of b. For instance, among the spatial configurations of Fig. 7, only regions a and b in Fig. 7(a) satisfy relation $S{:}W$. Relation $\{S, W\}$ is satisfied by regions a and b in Figs. 7(b) and 7(c) but it is not satisfied by the respective regions in Fig. 7(a).

Definition 4. Let $R \in 2^{\mathcal{D}^*}$. The *inverse* of relation R, denoted by $inv(R)$, is another cardinal direction relation which satisfies the following. For arbitrary regions $a, b \in REG^*$ the constraint $a\ inv(R)\ b$ holds, iff $b\ R\ a$ holds.

Let us consider two regions a and b and assume that $a\ R\ b$ holds, where R is a basic relation. Then, relation $inv(R)$ is not necessarily a basic cardinal direction relation but it can also be a disjunction of basic relations. For instance, if $a\ N\ b$ then it is possible that $b\ SE{:}S{:}SW\ a$ or $b\ SE{:}S\ a$ or $b\ S{:}SW\ a$ or $b\ S\ a$ (see Fig. 8). Therefore, we have:

$$inv(N) = \{S{:}SW{:}SE,\ S{:}SW,\ SE{:}S,\ S\}.$$

In other words, to describe the relative position of two regions a and b using cardinal direction relations we need to specify *both* the relation of a with respect to b *and* the relation of b with respect to a. Summarizing, the relative position of two regions a and b is given by the pair (R_1, R_2), where R_1 and R_2 are cardinal directions such that $a\ R_1\ b$, $b\ R_2\ a$, R_1 is a disjunct of $inv(R_2)$ and R_2 is a disjunct of $inv(R_1)$. An algorithm for computing the inverse relation is discussed at the end of Section 4.

In a previous line of work, we have studied the composition problem for cardinal direction relations [40,42]. In the following sections, we will study the consistency checking of

a given set of cardinal direction constraints and present an algorithm for this task. Let us first formally define cardinal direction constraints.

Definition 5. A *cardinal direction constraint* is a formula $a\ R\ b$ where a, b are variables ranging over regions in REG^* and R is a cardinal direction relation from the set $2^{\mathcal{D}^*}$. Moreover, a cardinal direction constraint is called *single-tile* (respectively *multi-tile, basic*) if R is a single-tile (respectively multi-tile, basic) cardinal direction relation.

Obviously, a basic cardinal direction constraint is *non-disjunctive* while, in general, a cardinal direction constraint can either be *disjunctive* or non-disjunctive.

Example 2. The following are cardinal direction constraints:

$$a_1\ S\ b_1, \quad a_2\ NE{:}E\ b_2 \quad \text{and} \quad a_3\ \{B, S\}\ b_3.$$

The constraint $a_1\ S\ b_1$ is single-tile. The constraint $a_2\ NE{:}E\ b_2$ is multi-tile. The first two constraints are basic (non-disjunctive) while the third one is not.

Definition 6. Let C be a set of cardinal direction constraints in variables $a_1, \ldots, a_n$. The *solution set* of C, denoted by $Sol(C)$, is defined as:

$$\big\{(\alpha_1, \ldots, \alpha_n)\colon\ \alpha_1, \ldots, \alpha_n \in REG^* \text{ and the constraints in } C \text{ are satisfied by assigning } \alpha_1 \text{ to } a_1, \ldots, \alpha_n \text{ to } a_n\big\}.$$

Each member of $Sol(C)$ is called a *solution* of C. A set of cardinal direction constraints is called *consistent* iff its solution set is non-empty.

In this paper, we will also be interested in special kinds of order constraints which are defined below.

Definition 7. An *order constraint* is a formula in any of the following forms:

$$inf_x(a) \sim inf_x(b), \quad sup_x(a) \sim sup_x(b), \quad inf_x(a) \sim sup_x(b),$$
$$inf_y(a) \sim inf_y(b), \quad sup_y(a) \sim sup_y(b), \quad inf_y(a) \sim sup_y(b)$$

where a and b are variables ranging over regions in REG^* and $\sim$ can be any operator from the set $\{<, >, \leqslant, \geqslant, =\}$.

The above order constraints express all possible relations between the endpoints of the projections on the x- and y-axis of regions a and b.

Definition 8. A set of order constraints is called *canonical* iff it includes the constraints $inf_x(a) < sup_x(a)$ and $inf_y(a) < sup_y(a)$ for every region variable a referenced in the set.

Definition 9. Let O be a canonical set of order constraints in region variables $a_1, \ldots, a_n$. The *solution set* of O, denoted by $Sol(O)$, is the set of n-tuples

$$(\alpha_1, \ldots, \alpha_n) \in (REG^*)^n$$

such that the constraints in O are satisfied by assigning

$$inf_x(\alpha_i) \text{ to } inf_x(a_i), \quad sup_x(\alpha_i) \text{ to } sup_x(a_i),$$
$$inf_y(\alpha_i) \text{ to } inf_y(a_i), \quad sup_y(\alpha_i) \text{ to } sup_y(a_i)$$

for every i, $1 \leqslant i \leqslant n$.

As with cardinal direction constraints, a set of order constraints is called *consistent* iff its solution set is non-empty. In this paper, we will use letters from the Latin alphabet (e.g., $a, b, c, r, \ldots$) to denote variables and letters from the Greek alphabet (e.g., $\alpha, \beta, \gamma, \rho, \ldots$) to denote values of the respective variable (similarly to Definition 9). Let us now consider the following proposition.

Proposition 2. *Let O be a canonical set of order constraints in region variables $a_1, \ldots, a_n$. Set O has a solution $(\alpha_1, \ldots, \alpha_n) \in (REG^*)^n$ iff it has a solution $(\beta_1, \ldots, \beta_n)$ where $\beta_1, \ldots, \beta_n$ are non-trivial boxes.*

Proof. (If) Obvious.

(Only if) Let $(\alpha_1, \ldots, \alpha_n) \in (REG^*)^n$ be a solution of O. In the definition of the solution of O (Definition 9), we are only interested in the endpoints of the projections on the x- and y-axis of regions $\alpha_1, \ldots, \alpha_n$. Notice that regions $\alpha_1, \ldots, \alpha_n$ have the same endpoints with their bounding boxes $mbb(\alpha_1), \ldots, mbb(\alpha_n)$, thus it follows that

$$\big(mbb(\alpha_1), \ldots, mbb(\alpha_n)\big)$$

is also a solution of O. □

Using Proposition 2, we can assume without loss of generality that if $\alpha_1, \ldots, \alpha_n$ is a solution of a canonical set of order constraints O in variables $a_1, \ldots, a_n$ then all $\alpha_1, \ldots, \alpha_n$ are non-trivial boxes. Proposition 2 will be very useful in Section 4 and in the proof of Theorem 3.

4. Consistency of basic cardinal direction constraints

We will now consider the consistency checking problem for a given set of cardinal direction constraints involving only basic relations and present an algorithm for solving it. In [40,42], we have studied the composition operator for cardinal direction relations and we have shown that we cannot use composition to decide consistency (as defined in Definition 6). As a result, this section does not use composition in any way.

Let us first consider Definition 3 that defines cardinal direction relations. The "iff" definitions of Definition 3 can be used to map a set of arbitrary basic cardinal direction constraints C to a set $\mathcal{S}$ consisting of the following two types of constraints:

- *single-tile* cardinal direction constraints;
- *set-union* constraints of the form $r = r_1 \cup \cdots \cup r_n$ where $r, r_1, \ldots, r_n$ are variables representing regions in REG^*.

If we had an algorithm for deciding the consistency of sets like $\mathcal{S}$, we could use it to solve the consistency problem. Given the unavailability of such an algorithm, below we develop from first principles Algorithm CONSISTENCY that checks whether C (equivalently $\mathcal{S}$) is consistent. CONSISTENCY is a rather long algorithm with a non-trivial step where we avoid having to deal with the set-union constraints of $\mathcal{S}$.

To check the consistency of a given set of basic cardinal direction constraints C in variables $a_1, \ldots, a_n$, Algorithm CONSISTENCY proceeds as follows:

1. Initially, Algorithm CONSISTENCY uses Algorithm TRANSFORM (Fig. 10) to translate the cardinal direction constraints of C into order constraints. Algorithm TRANSFORM considers every constraint of C in turn and maps the single-tile cardinal direction constraints of its definition into a set of order constraints O. Set O contains order constraints involving the projections on the x- and y-axis of region variables $a_1, \ldots, a_n$ and the component variables that correspond to $a_1, \ldots, a_n$. To achieve this mapping, Algorithm TRANSFORM uses Definitions 2 and 3 (Section 3.1). Moreover, the algorithm introduces into set O additional order constraints that are implied by the cardinal direction constraint under consideration. These constraints will be discussed in Section 4.1.
2. Then, Algorithm CONSISTENCY finds a solution of the set of order constraints O (any solution serves our purpose). To this end, we use the algorithms of [11,46]. If a solution of O exists, it assigns non-trivial boxes to region variables $a_1, \ldots, a_n$ and the component variables that correspond to $a_1, \ldots, a_n$ (see also Proposition 2). Using this solution, the second step of the algorithm constructs a *maximal solution* by enlarging appropriately the regions that correspond to the component variables of regions $a_1, \ldots, a_n$ (Section 4.2). This solution is called maximal in the sense that any further enlargement results in an assignment that is not a solution of O. The construction of a maximal solution is necessary in order to perform the third and last step of the algorithm.
3. The first two steps of Algorithm CONSISTENCY have considered for each constraint in set C *only* the single-tile cardinal direction constraints of its definition (see the discussion at the beginning of Section 4). This final step of the algorithm deals with the set-union constraints that correspond to every constraint in C. Currently, there does not exist an efficient algorithm for handling theories consisting of set constraints and order constraints. Thus, we go a step further and map set-union constraint into a complex expression involving order constraints. Then, we prove that to solve the consistency problem we just have to check whether the derived complex expression is satisfied by the maximal solution of Step 2. This checking can be efficiently performed using Algorithm GLOBALCHECKCONSTRAINTNTB (Section 4.3). This step is very interesting since it avoids using the computational costly set-union constraints.

Let us now describe the three steps of Algorithm CONSISTENCY. Throughout our presentation, we will use the set of constraints C of the following example to illustrate the details of the algorithm.

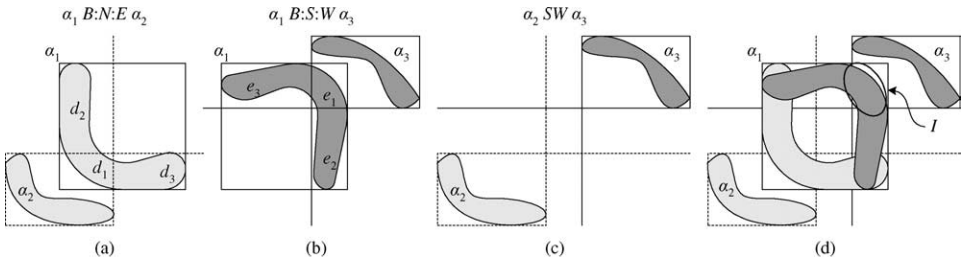

Fig. 9. Illustration of Example 3.

Example 3. Let C be the following set of basic cardinal direction constraints on region variables a_1, a_2 and a_3:

$$C = \{a_1\ B{:}N{:}E\ a_2,\ a_1\ B{:}S{:}W\ a_3,\ a_2\ SW\ a_3\}.$$

In Figs. 9(a), 9(b) and 9(c), we illustrate regions that satisfy constraints $a_1\ B{:}N{:}E\ a_2$, $a_1\ B{:}S{:}W\ a_3$ and $a_2\ SW\ a_3$ respectively. Unfortunately, there does not exist an assignment that satisfies all constraints of set C. In other words, set C is inconsistent. To see this consider Fig. 9(d). The first constraint of C forces region a_1 to be shaped like the *light* grey region of Fig. 9(d) while the second one forces region a_1 to be shaped like the *dark* grey region. Now let us consider the circled area I of Fig. 3(d). Constraint $a_1\ B{:}N{:}E\ a_2$ requires that region I *does not belong* to a_1 while constraint $a_1\ B{:}S{:}W\ a_3$ requires that region I *belongs* to a_1 resulting in an inconsistency.

4.1. Step 1—Algorithm TRANSFORM

Let C be a set of basic cardinal direction constraints on region variables $a_1, \ldots, a_n$. The first step of Algorithm CONSISTENCY uses Algorithm TRANSFORM (Fig. 10) to translate set C into a set of order constraints O. Set O contains order constraints involving the projections on the x- and y-axis of region variables $a_1, \ldots, a_n$ and the component variables that correspond to $a_1, \ldots, a_n$. For every constraint in the input set C, Algorithm TRANSFORM repeats Steps T1–T4. Let us consider an arbitrary constraint $a_i\ R_1{:}\cdots{:}R_k\ a_j$ in C.

In Step T1 of Algorithm TRANSFORM, we consult the definition constraint $a_i\ R_1{:}\cdots{:}R_k\ a_j$ (Section 3.1), and introduce order constraints encoding all single-tile cardinal direction relations between the reference region a_j and the component variables corresponding to the primary region a_i. More specifically, Step T1 distinguishes two cases.

1. If $k = 1$ then $a_i\ R_1{:}\cdots{:}R_k\ a_j$ is a single-tile constraint and Step T1 introduces the corresponding equivalent order constraints of Definition 2.
2. If $k > 1$ then $a_i\ R_1{:}\cdots{:}R_k\ a_j$ is a multi-tile constraint and Step T1 introduces new region variables $a^j_{i1}, \ldots, a^j_{ik}$ which denote the component variables corresponding to variable a_i and constraint $a_i\ R_1{:}\cdots{:}R_k\ a_j$ (see Definition 3). Then, Step T1 introduces order constraints equivalent to single-tile constraints $a^j_{i1}\ R_1\ a_j, \ldots, a^j_{ik}\ R_k\ a_j$ by consulting Definition 2.

Algorithm TRANSFORM

Input: A set of basic cardinal direction constraints C in variables $a_1, \ldots, a_n$.
Output: A set O of order constraints involving variables $a_1, \ldots, a_n$ and the component variables corresponding to $a_1, \ldots, a_n$.
Method:
$O = \emptyset$;
For every constraint $a_i\ R_1{:}\cdots{:}R_k\ a_j$ in C Do

Step T1: Introduce constraints from the definitions of Section 3.1

If $k = 1$ Then
Add to O the order constraints defining the single-tile cardinal direction constraint $a^i\ R_1\ a_j$ (use definitions of Section 3.1).
ElseIf $k > 1$ Then
Introduce new region variables $a^j_{i1}, \ldots, a^j_{ik}$. These are component variables corresponding to a_i.
For $t = 1$ To k Do
Add to O the order constraints defining the single-tile cardinal direction constraint $a^j_{it}\ R_t\ a_j$ (use definitions of Section 3.1).
EndFor
EndIf

Step T2: Enforce that regions $\{a_i, a_j, a^j_{i1}, \ldots, a^j_{ik}\}$ are in REG^*.

For every region variable r in $\{a_i, a_j, a^j_{i1}, \ldots, a^j_{ik}\}$ Do
$O = O \cup \{inf_x(r) < sup_x(r),\ inf_y(r) < sup_y(r)\}$
EndFor

Step T3: Enforce that regions $\{a^j_{i1}, \ldots, a^j_{ik}\}$ are subregions of a_i.

For every region variable r in $\{a^j_{i1}, \ldots, a^j_{ik}\}$ Do
$O = O \cup \{inf_x(a_i) \leqslant inf_x(r),\ sup_x(r) \leqslant sup_x(a_i),\ inf_y(a_i) \leqslant inf_y(r),\ sup_y(r) \leqslant sup_y(a_i)\}$
EndFor

Step T4: Strictest relation between a_i and a_j

If $k > 1$ Then

If	$\{R_1, \ldots, R_k\} \subseteq \{NE, E, SE\}$	Then	$O = O \cup \{sup_x(a_j) \leqslant inf_x(a_i)\}$
Else If	$\{R_1, \ldots, R_k\} \subseteq \{NE, E, SE, N, B, S\}$	Then	$O = O \cup \{inf_x(a_j) \leqslant inf_x(a_i)\}$
If	$\{R_1, \ldots, R_k\} \subseteq \{NW, W, SW\}$	Then	$O = O \cup \{sup_x(a_i) \leqslant inf_x(a_j)\}$
Else If	$\{R_1, \ldots, R_k\} \subseteq \{NW, W, SW, N, B, S\}$	Then	$O = O \cup \{sup_x(a_i) \leqslant sup_x(a_j)\}$
If	$\{R_1, \ldots, R_k\} \subseteq \{NW, N, NE\}$	Then	$O = O \cup \{sup_y(a_j) \leqslant inf_y(a_i)\}$
Else If	$\{R_1, \ldots, R_k\} \subseteq \{NW, N, NE, W, B, E\}$	Then	$O = O \cup \{inf_y(a_j) \leqslant inf_y(a_i)\}$
If	$\{R_1, \ldots, R_k\} \subseteq \{SW, S, SE\}$	Then	$O = O \cup \{sup_y(a_i) \leqslant inf_y(a_j)\}$
Else If	$\{R_1, \ldots, R_k\} \subseteq \{SW, S, SE, W, B, E\}$	Then	$O = O \cup \{sup_y(a_i) \leqslant sup_y(a_j)\}$

EndIf

EndFor
Return O

Fig. 10. Algorithm TRANSFORM.

Example 4. Let us continue with the set $C = \{a_1\ B{:}N{:}E\ a_2,\ a_1\ B{:}S{:}W\ a_3,\ a_2\ SW\ a_3\}$ of Example 3. Let O be the input of Algorithm TRANSFORM. This algorithm considers constraint $a_1\ B{:}N{:}E\ a_2$ first. Step T1 introduces component variables[1] d_1, d_2 and d_3 representing subregions of a_1 such that $d_1\ B\ a_2$, $d_2\ N\ a_2$, and $d_3\ E\ a_2$ hold. Then, the definitions of relations B, N and E are consulted and Step T1 adds to O the following order constraints (see also Fig. 9(a)):

$$\begin{aligned}
&\text{Constraint } d_1\ B\ a_2\text{:} && inf_x(a_2) \leqslant inf_x(d_1),\ sup_x(d_1) \leqslant sup_x(a_2),\\
& && inf_y(a_2) \leqslant inf_y(d_1),\ sup_y(d_1) \leqslant sup_y(a_2).\\
&\text{Constraint } d_2\ N\ a_2\text{:} && inf_x(a_2) \leqslant inf_x(d_2),\ sup_x(d_2) \leqslant sup_x(a_2),\\
& && sup_y(a_2) \leqslant inf_y(d_2).\\
&\text{Constraint } d_3\ E\ a_2\text{:} && sup_x(a_2) \leqslant inf_x(d_3),\ inf_y(a_2) \leqslant inf_y(d_3),\\
& && sup_y(d_3) \leqslant sup_y(a_2).
\end{aligned}$$

Then, Step T1 considers constraint $a_1\ B{:}S{:}W\ a_3$ and introduces component variables e_1, e_2 and e_3 representing subregions of a_1 such that $e_1\ B\ a_3$, $e_2\ S\ a_3$, and $e_3\ W\ a_3$ hold. Then, Step T1 adds to O the following order constraints (see also Fig. 9(b)):

$$\begin{aligned}
&\text{Constraint } e_1\ B\ a_3\text{:} && inf_x(a_3) \leqslant inf_x(e_1),\ sup_x(e_1) \leqslant sup_x(a_3),\\
& && inf_y(a_3) \leqslant inf_y(e_1),\ sup_y(e_1) \leqslant sup_y(a_3).\\
&\text{Constraint } e_2\ S\ a_3\text{:} && inf_x(a_3) \leqslant inf_x(e_2),\ sup_x(e_2) \leqslant sup_x(a_3),\\
& && sup_y(e_2) \leqslant inf_y(a_3).\\
&\text{Constraint } e_3\ W\ a_3\text{:} && sup_x(e_3) \leqslant inf_x(a_3),\ inf_y(a_3) \leqslant inf_y(e_3),\\
& && sup_y(e_3) \leqslant sup_y(a_3).
\end{aligned}$$

Finally, Step T1 considers constraint $a_2\ SW\ a_3$ and adds to O the following order constraints (see also Fig. 9(c)):

$$\text{Constraint } a_2\ SW\ a_3\text{:}\quad sup_x(a_2) \leqslant inf_x(a_3),\ sup_y(a_2) \leqslant inf_y(a_3).$$

In Step T2 of Algorithm TRANSFORM, we introduce for regions a_i, a_j and the component variables $a_{i1}^j, \ldots, a_{ik}^j$ corresponding to a_i, the obvious order constraints relating the endpoints of their projections (Proposition 1). These constraints make the set of order constraints O canonical.

Example 5. Let us continue with the set C of Example 3. The constraints of set C are expressed on region variables $\{a_1, a_2, a_3\}$. Moreover, Step T1 of Algorithm TRANSFORM has introduced region variables $\{d_1, d_2, d_3, e_1, e_2, e_3\}$ representing component variables corresponding to a_1 (see also Example 4). Thus, for every region variable $r \in \{a_1, a_2, a_3, d_1, d_2, d_3, e_1, e_2, e_3\}$, Step T2 of Algorithm TRANSFORM adds to O the following constraint (see also Fig. 9):

$$inf_x(r) < sup_x(r) \quad \text{and} \quad inf_y(r) < sup_y(r).$$

[1] Algorithm TRANSFORM introduces component variables a_{11}^2, a_{12}^2 and a_{13}^2. In order to simplify the expressions, these variables are denoted by d_1, d_2 and d_3 respectively.

Step T3 of Algorithm TRANSFORM deals with the component variables $a_{i1}^j, \ldots, a_{ik}^j$ corresponding to variable a_i. The fact that a variable r is a component variable representing a subregion of a_i, implies that the following constraints hold:

$$inf_x(a_i) \leqslant inf_x(r), \quad sup_x(r) \leqslant sup_x(a_i),$$
$$inf_y(a_i) \leqslant inf_y(r) \quad \text{and} \quad sup_y(r) \leqslant sup_y(a_i).$$

The above constraints are introduced by Step T3 for all component variables $a_{i1}^j, \ldots, a_{ik}^j$ corresponding to region variable a_i.

Example 6. Let us continue with the set C of Example 3. Notice that regions $\{d_1, d_2, d_3, e_1, e_2, e_3\}$ are all subregions of a_1. Thus, for every region variable $r \in \{d_1, d_2, d_3, e_1, e_2, e_3\}$, Step T3 adds to O the following constraints (see also Fig. 9):

$$inf_x(a_1) \leqslant inf_x(r), \quad sup_x(r) \leqslant sup_x(a_1),$$
$$inf_y(a_1) \leqslant inf_y(r) \quad \text{and} \quad sup_y(r) \leqslant sup_x(a_1).$$

Given a constraint $a_i\ R\ a_j$, the constraints introduced by Steps T1, T2 and T3 of Algorithm TRANSFORM establish relations between (a) the component variables corresponding to a_i and (b) region variables a_i and a_j. Unfortunately, if R is a multi-tile cardinal direction relation, these constraints are not enough to establish the *strictest possible* order relation between the endpoints of the projections of regions a_i and a_j on the x- and y-axis implied by the definitions of Section 3.1. For instance, consider regions a_1 and a_2 of Fig. 9(a). For these regions constraints, $inf_x(a_2) \leqslant inf_x(a_1)$ and $inf_y(a_2) \leqslant inf_y(a_1)$ hold. These constraints could not have been introduced by Steps T1, T2 and T3 or implied by constraints introduced by these steps (see also Examples 4, 5 and 6).

Step T4 of Algorithm TRANSFORM examines all multi-tiles constraints of the given set C and introduces additional order constraints that establish the aforementioned strictest relation between the endpoints of the projections of regions a_i and a_j on the x- and y-axis. Using a simple case analysis, we can verify that we need to consider 8 possible cases. These cases correspond to checking, for every constraint $a_i\ R_1{:}\cdots{:}R_k\ a_j$ of C, whether the set of relations $\{R_1, \ldots, R_k\}$ is a subset of one or more of the following sets:

$\{NE, E, SE\}$, $\{NE, E, SE, N, B, S\}$,
$\{NW, W, SW\}$, $\{NW, W, SW, N, B, S\}$,
$\{NW, N, NE\}$, $\{NW, N, NE, W, B, E\}$,
$\{SW, S, SE\}$, $\{SW, S, SE, W, B, E\}$.

For example, if the set of relations $\{R_1, \ldots, R_k\}$ is a subset of $\{NE, E, SE\}$, the constraint

$$sup_x(a_j) \leqslant inf_x(a_i)$$

is introduced in O by the first If statement of Step T4 because the primary region denoted by a_i is included in the region defined by the tiles $NE(a_j) \cup E(a_j) \cup SE(a_j)$ of the reference region denoted by a_j (Fig. 11(a)). On the other hand, if the set of relations $\{R_1, \ldots, R_k\}$ is

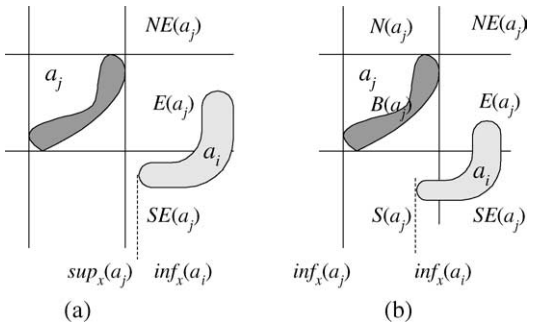

Fig. 11. Example of constraints introduced by Step T4 of Algorithm TRANSFORM.

a subset of $\{NE, E, SE, N, B, S\}$, the first ElseIf statement of Step T4 adds the following constraint to O (see also Fig. 11(b)):

$$inf_x(a_j) \leqslant inf_x(a_i).$$

Similar comments are in order for the other statements of Step T4.

Example 7. Let us continue with the set C of Example 3 and consider constraint $a_1\ B{:}N{:}E\ a_2$. We notice that relations $\{B, N, E\}$ are members of

$$\{NE, E, SE, N, B, S\} \quad \text{and} \quad \{NW, N, NE, W, B, E\}.$$

As a result Step T4 adds to O the following constraints (see also Fig. 9(a)):

$$inf_x(a_2) \leqslant inf_x(a_1), \quad \text{and} \quad inf_y(a_2) \leqslant inf_y(a_1).$$

When we consider constraint $a_1\ B{:}S{:}W\ a_3$, we notice that relations $\{B, S, W\}$ are members of $\{NW, W, SW, N, B, S\}$ and $\{SW, S, SE, W, B, E\}$. Thus, Step T4 adds to O the following constraints (see also Fig. 9(b)):

$$sup_x(a_1) \leqslant sup_x(a_3), \quad \text{and} \quad sup_y(a_1) \leqslant sup_y(a_3).$$

4.2. Step 2—maximal solution

The second step of Algorithm CONSISTENCY checks the consistency of the set of order constraints O produced by Algorithm TRANSFORM. To this end, we use the algorithms of [11,46]. It follows from the discussion of Section 4.1 that all constraints introduced in set O by Step 1 are logically implied by the spatial configuration expressed by the cardinal direction constraints of the original set C. Thus, if set O is inconsistent then set C is also inconsistent and Algorithm CONSISTENCY exits returning 'Inconsistent'.

Let us now assume that a solution of O exists. Such a solution assigns non-trivial boxes (see Proposition 2) to the region variables of set C and their corresponding component variables. Let us consider the following example.

Example 8. Let us continue with the set C of Example 3. In Examples 4–7, we have used Algorithm TRANSFORM to map C into a set of order constraints O.

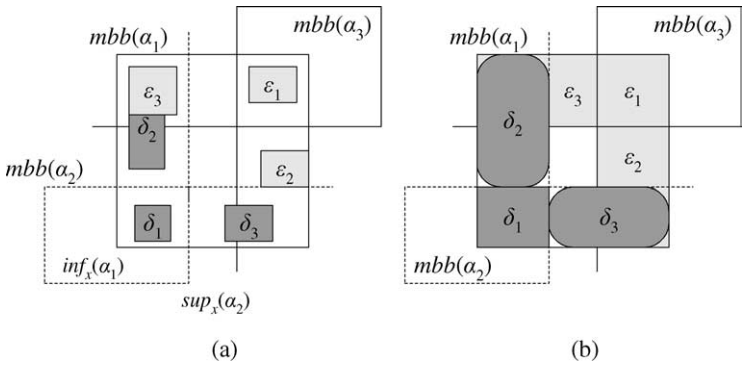

Fig. 12. Solutions of O.

Set O is consistent. For instance, a solution of O can be constructed if we assign to region variables a_1, a_2, a_3, d_1, d_2, d_3, e_1, e_2 and e_3 the non-trivial boxes α_1, α_2, α_3, δ_1, δ_2, δ_3, ε_1, ε_2 and ε_3 depicted in Fig. 12(a) respectively. It easy to verify that the regions of Fig. 12(a) satisfy all constraints of O introduced in Examples 4–7.

We can now extend boxes δ_1, δ_2, δ_3, ε_1, ε_2 and ε_3 in all directions until they touch whatever line is closer to them from the ones forming boxes α_1 and α_2. For instance, we can extend δ_1 to the west to touch the vertical line $y = inf_x(\alpha_1)$ and to the east to touch the vertical line $y = sup_x(\alpha_2)$. Fig. 12(b) illustrates this idea for all regions δ_1, δ_2, δ_3, ε_1, ε_2 and ε_3 (the corners of δ_2 and δ_3 have been curved to show that these regions overlap with regions ε_3 and ε_2 respectively). Now notice that the new regions still satisfy all constraints in O.

The following lemma captures the observation of Example 8 in its full generality.

Lemma 1. *Let O be the output of Algorithm* TRANSFORM *when it is called with input a set of cardinal direction constraints C in variables $a_1, \ldots, a_n$. For each variable a_i $(1 \leqslant i \leqslant n)$, let $a_{i1}, \ldots, a_{il}$ be the component variables corresponding to a_i that have been generated by Algorithm* TRANSFORM *while considering various constraints $a_i\ R_1{:}\cdots{:}R_k\ a_j$ where $1 \leqslant j \leqslant n$ and $1 \leqslant k \leqslant 9$.*

Let s^0 be a solution of O. Let also α_i, α_j, $\alpha_{i1}, \ldots, \alpha_{il}$ be the non-trivial boxes that s^0 assigns to region variables a_i, a_j, $a_{i1}, \ldots, a_{il}$ respectively. Then, a new solution v^0 of O can be constructed as follows. For every constraint $a_i\ R_1{:}\cdots{:}R_k\ a_j$ in C we consider each component variable a_{im} $(1 \leqslant m \leqslant l)$ in turn.

- *If $\alpha_{im}\ B\ \alpha_j$ holds, perform the following substitutions*:

$$
\begin{aligned}
&inf_x(\alpha_{im}) \quad \text{by } max\{inf_x(\alpha_i),\ inf_x(\alpha_j)\},\\
&sup_x(\alpha_{im}) \quad \text{by } min\{sup_x(\alpha_i),\ sup_x(\alpha_j)\},\\
&inf_y(\alpha_{im}) \quad \text{by } max\{inf_y(\alpha_i),\ inf_y(\alpha_j)\},\\
&sup_y(\alpha_{im}) \quad \text{by } min\{sup_y(\alpha_i),\ sup_y(\alpha_j)\}.
\end{aligned}
$$

- *If* α_{im} *S* α_j *holds, perform the following substitutions*:

$$\begin{aligned}
&inf_x(\alpha_{im}) \quad by\ max\{\ inf_x(\alpha_i),\ inf_x(\alpha_j)\},\\
&sup_x(\alpha_{im}) \quad by\ min\{sup_x(\alpha_i),\ sup_x(\alpha_j)\},\\
&inf_y(\alpha_{im}) \quad by\ inf_y(\alpha_i), \quad sup_y(\alpha_{im}) \quad by\ min\{sup_y(\alpha_i),\ inf_y(\alpha_j)\}.
\end{aligned}$$

- *If* α_{im} *SW* α_j *holds, perform the following substitutions*:

$$\begin{aligned}
&inf_x(\alpha_{im}) \quad by\ inf_x(\alpha_i), \quad sup_x(\alpha_{im}) \quad by\ min\{sup_x(\alpha_i),\ inf_x(\alpha_j)\},\\
&inf_y(\alpha_{im}) \quad by\ inf_y(\alpha_i), \quad sup_y(\alpha_{im}) \quad by\ min\{sup_y(\alpha_i),\ inf_y(\alpha_j)\}.
\end{aligned}$$

- *If* α_{im} *W* α_j *holds, perform the following substitutions*:

$$\begin{aligned}
&inf_x(\alpha_{im}) \quad by\ inf_x(\alpha_i), \quad sup_x(\alpha_{im}) \quad by\ min\{sup_x(\alpha_i),\ inf_x(\alpha_j)\},\\
&inf_y(\alpha_{im}) \quad by\ max\{inf_y(\alpha_i),\ inf_y(\alpha_j)\},\\
&sup_y(\alpha_{im}) \quad by\ min\{sup_y(\alpha_i),\ sup_y(\alpha_j)\}.
\end{aligned}$$

- *If* α_{im} *NW* α_j *holds, perform the following substitutions*:

$$\begin{aligned}
&inf_x(\alpha_{im}) \quad by\ inf_x(\alpha_i), \quad sup_x(\alpha_{im}) \quad by\ min\{sup_x(\alpha_i),\ inf_x(\alpha_j)\},\\
&inf_y(\alpha_{im}) \quad by\ max\{inf_y(\alpha_i),\ sup_y(\alpha_j)\}, \quad sup_y(\alpha_{im}) \quad by\ sup_y(\alpha_i).
\end{aligned}$$

- *If* α_{im} *N* α_j *holds, perform the following substitutions*:

$$\begin{aligned}
&inf_x(\alpha_{im}) \quad by\ max\{inf_x(\alpha_i),\ inf_x(\alpha_j)\},\\
&sup_x(\alpha_{im}) \quad by\ min\{sup_x(\alpha_i),\ sup_x(\alpha_j)\},\\
&inf_y(\alpha_{im}) \quad by\ max\{\ inf_y(\alpha_i),\ sup_y(\alpha_j)\}, \quad sup_y(\alpha_{im}) \quad by\ sup_y(\alpha_i).
\end{aligned}$$

- *If* α_{im} *NE* α_j *holds, perform the following substitutions*:

$$\begin{aligned}
&inf_x(\alpha_{im}) \quad by\ max\{inf_x(\alpha_i),\ sup_x(\alpha_j)\}, \quad sup_x(\alpha_{im}) \quad by\ sup_x(\alpha_i),\\
&inf_y(\alpha_{im}) \quad by\ max\{inf_y(\alpha_i),\ sup_y(\alpha_j)\}, \quad sup_y(\alpha_{im}) \quad by\ sup_y(\alpha_i).
\end{aligned}$$

- *If* α_{im} *E* α_j *holds, perform the following substitutions*:

$$\begin{aligned}
&inf_x(\alpha_{im}) \quad by\ max\{inf_x(\alpha_i),\ sup_x(\alpha_j)\}, \quad sup_x(\alpha_{im}) \quad by\ sup_x(\alpha_i),\\
&inf_y(\alpha_{im}) \quad by\ max\{inf_y(\alpha_i),\ inf_y(\alpha_j)\},\\
&sup_y(\alpha_{im}) \quad by\ min\{\ sup_y(\alpha_i),\ sup_y(\alpha_j)\}.
\end{aligned}$$

- *If* α_{im} *SE* α_j *holds, perform the following substitutions*:

$$\begin{aligned}
&inf_x(\alpha_{im}) \quad by\ max\{inf_x(\alpha_i),\ sup_x(\alpha_j)\}, \quad sup_x(\alpha_{im}) \quad by\ sup_x(\alpha_i),\\
&inf_y(\alpha_{im}) \quad by\ inf_y(\alpha_i), \quad sup_y(\alpha_{im}) \quad by\ min\{sup_y(\alpha_i),\ inf_y(\alpha_j)\}.
\end{aligned}$$

Proof. See Appendix A. □

Example 9. Let us continue with the set C of Example 3 and let O be the output of Algorithm TRANSFORM with input set C. In Example 8, we have seen that the regions of Figs. 12(a) and 12(b) form solutions of set O. We can verify that the regions of Fig. 12(b) are formed by applying to the regions of Fig. 12(a) the substitutions of Lemma 1. For instance, consider the component δ_1 corresponding to region α_1 of Fig. 12(a) and notice that $\delta_1\ B\ \alpha_2$ holds. Region δ_1 of Fig. 12(b) results after performing the following substitutions:

$$
\begin{aligned}
&inf_x(\delta_1) \quad \text{by } inf_x(\alpha_1) = max\{inf_x(\alpha_1),\ inf_x(\alpha_2)\},\\
&sup_x(\delta_1) \quad \text{by } sup_x(\alpha_2) = min\{sup_x(\alpha_1),\ sup_x(\alpha_2)\},\\
&inf_y(\delta_1) \quad \text{by } inf_y(\alpha_1) = max\{inf_y(\alpha_1),\ inf_y(\alpha_2)\},\\
&sup_y(\delta_1) \quad \text{by } sup_y(\alpha_2) = min\{sup_y(\alpha_1),\ sup_y(\alpha_2)\}.
\end{aligned}
$$

Definition 10. Let C be a set of basic cardinal direction constraints and O be the set returned by Algorithm TRANSFORM when it is called with input C. A solution v^0 of O is called *maximal* iff v^0 is not affected by the substitutions of Lemma 1.

Example 10. Continuing with set O of Example 9, we notice that the regions of Fig. 12(b) form a maximal solution of O.

Using the Definition 10 and Lemma 1, we can prove the following theorem.

Theorem 1. *Let C be a set of basic cardinal direction relations and O be the set returned by Algorithm* TRANSFORM *when it is called with input C. Set O is consistent iff it has a maximal solution.*

Proof. (If) Obvious.

(Only if) If O is consistent it follows that it has at least a solution s^0. Applying the substitutions of Lemma 1 to s^0 we can form a maximal solution of O. □

4.3. Step 3—the non-trivial box constraint

At the beginning of Section 4, we have explained how one can transform a given set of cardinal direction constraints C into a set $\mathcal{S}$ containing only single-tile cardinal direction constraints and set-union constraints. Up to this point, Algorithm CONSISTENCY has dealt with the order constraints produced by Algorithm TRANSFORM. These order constraints encode *only* the single-tile cardinal direction constraints envisaged in set $\mathcal{S}$ (Step T1 of Algorithm TRANSFORM) together with some other useful constraints (Steps T2–T4). It is now time to introduce a special constraint capturing only the *essential facts following from the set-union constraints of set $\mathcal{S}$*.

Let us assume that $C_{a_i} = \{c_1, \ldots, c_m\}$ contains all constraints of the input set C with region a_i as the primary region. Let

$$
\begin{aligned}
c_1 \text{ be } a_i\ & R_1^1{:}\cdots{:}R_{k_1}^1\ a_{j_1}\\
&\ \vdots\\
c_m \text{ be } a_i\ & R_1^m{:}\cdots{:}R_{k_m}^m\ a_{j_m}
\end{aligned}
$$

where $R_1^1{:}\cdots{:}R_{k_1}^1, \ldots, R_1^m{:}\cdots{:}R_{k_m}^m$ are cardinal direction relations and $a_{j_1}, \ldots, a_{j_m}$ are region variables of set C.

Let $S_1 = \{a_{i1}^1, \ldots, a_{ik_1}^1\}, \ldots, S_m = \{a_{i1}^m, \ldots, a_{ik_m}^m\}$ be the sets of component variables corresponding to a_i due to constraints $c_1, \ldots, c_m$ respectively. According to the definitions of Section 3.1, set $\mathcal{S}$ would contain the following set-union constraints:

$$\begin{aligned} a_i &= a_{i1}^1 \cup \cdots \cup a_{ik_1}^1 \\ &\vdots \\ a_i &= a_{i1}^m \cup \cdots \cup a_{ik_m}^m. \end{aligned}$$

Let us consider an arbitrary component variable $s \in S_1 \cup \cdots \cup S_m$. The above set-union constraints imply that there is an important relationship between component variable s and the component variables from sets $S_1, \ldots, S_m$. This relationship is described by the above set-union constraints but unfortunately it cannot straightforwardly be mapped into order constraints. Lemma 2 expresses conditions that captures the relations following for the set-union constraints. These relations are established on the minimum bounding boxes of regions; thus, they can be expressed using order constraints.

Lemma 2. *Let a be a region in REG*. Let $S_1, \ldots, S_m$ be finite sets of subregions of a such that*:

1. *Every region in S_i, $1 \leqslant i \leqslant n$, is in REG*, i.e.,*

$$(\forall r \in S_i)(r \in REG^*).$$

2. *The union of the members of each S_i, $1 \leqslant i \leqslant n$, is region a, i.e.,*

$$a = \bigcup_{r \in S_1} r = \cdots = \bigcup_{r \in S_m} r.$$

Then, the following constraint also holds.

> **Non-Trivial Box Constraint (NTB)**: *For all $s \in S_1 \cup \cdots \cup S_m$ there exists a tuple $(s_1, \ldots, s_m) \in S_1 \times \cdots \times S_m$ such that $mbb(s) \cap mbb(s_1) \cap \cdots \cap mbb(s_m)$ is a non-trivial box.*

Proof. We will use induction on m. For $m = 1$, Constraint NTB trivially holds. For $m = 2$, Constraint NTB also holds. By contradiction, let us assume that there exists a region $s \in S_1$ such that $mbb(s) \cap mbb(s_1) \cap mbb(s_2)$ is a trivial box for every $s_1 \in S_1$ and $s_2 \in S_2$ (the case where $s \in S_2$ is similar). Since s and s_1 are in S_1 and our assumption holds for every $s_1 \in S_1$, we can choose s_1 to be s. Then, for all subregions $s_2 \in S_2$ of a, $mbb(s) \cap mbb(s_2)$ would be either empty or a point or a vertical line segment or a horizontal line segment (see Fig. 13(a)). Since region a is the union of all regions in S_2 (i.e., $a = \bigcup_{r \in S_2} r$ holds), it follows that region a will not have any points in the interior of the area covered by the minimum bounding box of region s and thus in the interior of region s itself. This contradicts our initial assumption that region s is a subregion of a.

Let us now assume that Constraint NTB holds for $m = \mu - 1$, i.e.,

Constraint NTB$_{\mu-1}$: For all $s \in S_1 \cup \cdots \cup S_{\mu-1}$ there exists a tuple $(s_1, \ldots, s_{\mu-1}) \in S_1 \times \cdots \times S_{\mu-1}$ such that $mbb(s) \cap mbb(s_1) \cap \cdots \cap mbb(s_{\mu-1})$ is a non-trivial box.

We will prove that the constraint holds for $m = \mu$ as well, i.e.,

Constraint NTB$_{\mu}$: For all $s \in S_1 \cup \cdots \cup S_{\mu}$ there exists a tuple $(s_1, \ldots, s_{\mu}) \in S_1 \times \cdots \times S_{\mu}$ such that $mbb(s) \cap mbb(s_1) \cap \cdots \cap mbb(s_{\mu})$ is a non-trivial box.

We will first prove that Constraint NTB$_{\mu}$ holds for all $s \in S_1 \cup \cdots \cup S_{\mu-1}$. Since Constraint NTB$_{\mu-1}$ holds, there exist regions $(s_1, \ldots, s_{\mu-1}) \in S_1 \times \cdots \times S_{\mu-1}$, such that $mbb(\sigma) \cap mbb(s_1) \cap \cdots \cap mbb(s_{\mu-1})$ is a non-trivial box (Fig. 13(b)). It is easy to see that no matter how S_μ divides a into subregions there would be a subregion $s_\mu \in S_\mu$ such that $mbb(\sigma) \cap mbb(s_1) \cap \cdots \cap mbb(s_{\mu-1}) \cap mbb(s_\mu)$, is a non-trivial box (see Fig. 13(c)).

Similarly, we can prove that Constraint NTB$_{\mu}$ holds for all $s \in S_\mu$; which concludes our proof. □

Example 11. Let us consider regions α_1, α_2, α_3 and α_4 of Fig. 14. For these regions we have:

$$\{\alpha_1\ S{:}SW\ \alpha_2,\ \alpha_1\ NW{:}N{:}NE\ \alpha_3,\ \alpha_1\ S{:}SW{:}W\ \alpha_4\}.$$

Let $\Sigma_1 = \{\delta_1,\ \delta_2\}$, $\Sigma_2 = \{\varepsilon_1,\ \varepsilon_2,\ \varepsilon_3\}$, and $\Sigma_3 = \{\zeta_1,\ \zeta_2,\ \zeta_3\}$ be the set of components corresponding to a_1 due to constraints $\alpha_1\ S{:}SW\ \alpha_2$, $\alpha_1\ NW{:}N{:}NE\ \alpha_3$ and $\alpha_1\ S{:}SW{:}W\ \alpha_4$ respectively.

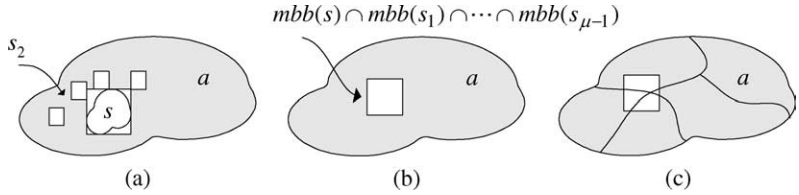

Fig. 13. Region a and its subregions.

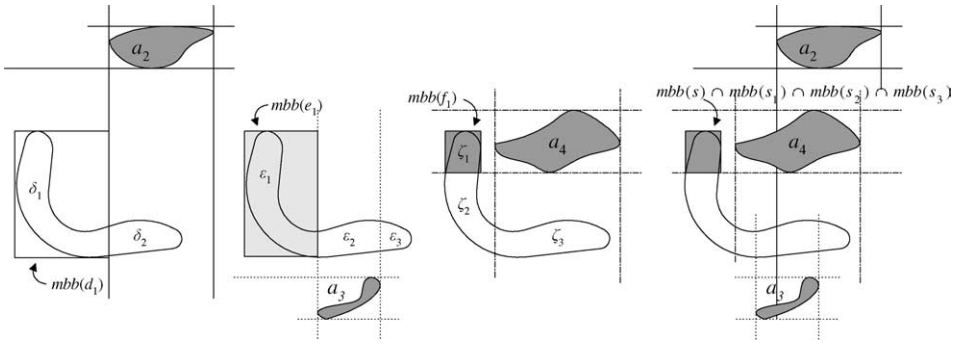

Fig. 14. Example of Constraint NTB.

Observing Fig. 14, it is not hard to see that Constraint NTB holds. For instance, for component $s = \delta_1 \in S_1$ there exists a region $s_1 \in S_1$, namely δ_1, a region in $s_2 \in S_2$, namely ε_1, and a region in $s_3 \in S_3$, namely ζ_1, such that $mbb(s) \cap mbb(s_1) \cap mbb(s_2) \cap mbb(s_3)$ is a non-trivial box (the corresponding bounding boxes are depicted in Fig. 14).

Constraint NTB requires that the intersection of $m+1$ non-trivial boxes $(s, s_1, \ldots, s_m)$ is a non-trivial box. This condition can be mapped into order constraints. For instance, given two non-trivial boxes a and b their intersection $c = a \cap b$ is a box defined as follows.

$$inf_x(c) = max\{inf_x(a), inf_x(b)\}, \qquad sup_x(c) = min\{sup_x(a), sup_x(b)\},$$
$$inf_y(c) = max\{inf_y(a), inf_y(b)\}, \qquad sup_y(c) = min\{sup_y(a), sup_y(b)\}.$$

Box c is non-trivial iff $inf_x(c) < sup_x(c)$ and $inf_y(c) < sup_y(c)$ (see also Proposition 1).

Lemma 2 is very important since it provides us with a method (using Constraint NTB) to map the set-union constraints of the definition of a cardinal direction relation (Definition 3) into order constraints. Therefore, a solution of a given set of cardinal direction constraints should not only satisfy the order constraint introduced by Algorithm TRANSFORM but also the order constraints that correspond to Constraint NTB. Since the former constraints are enforced by Step 1 of Algorithm CONSISTENCY, at this point one might wonder whether to solve the consistency problem, it suffices to introduce expressions that *enforce* Constraint NTB. Indeed, this is correct but unfortunately it results in an inefficient algorithm. Let us briefly discuss why.[2] Constraint NTB can be equivalently written as follows.

$$\bigwedge_{s \in S_1 \cup \cdots \cup S_m} \Bigg(\bigvee_{(s_1,\ldots,s_m) \in S_1 \times \cdots \times S_m} (mbb(s) \cap mbb(s_1) \cap \cdots \cap mbb(s_m) \text{ is a non-trivial box}) \Bigg).$$

Notice that $m = \mathcal{O}(n)$ and $|S_t| \leqslant 9$ $(1 \leqslant t \leqslant m)$ hold. The above expression contains $|S_1| + \cdots + |S_m| = \mathcal{O}(n)$ conjunctions each containing a disjunction with $|S_1| \cdots |S_m| = \mathcal{O}(9^n)$ disjuncts. Thus, in order to enforce Constraint NTB we have to write down (and solve!) an expression exponential to the size of our initial problem.

Summarizing, enforcing Constraint NTB is an inefficient procedure. We will now investigate the problem of checking whether a given assignment satisfies Constraint NTB. As we will later see in this section, a solution to this problem will help us tackle the consistency checking problem. Let $\Sigma_1, \ldots, \Sigma_m$ be sets of regions representing components of a given region α. In order to check whether $\Sigma_1, \ldots, \Sigma_m$ satisfy Constraint NTB, we use Algorithm CHECKCONSTRAINTNTB (Fig. 15). For every component variable s in $\Sigma_1 \cup \cdots \cup \Sigma_m$, the algorithm checks if there exists a tuple $(s_1, \ldots, s_m) \in \Sigma_1 \times \cdots \times \Sigma_m$ such that $mbb(s) \cap mbb(s_1) \cap \cdots \cap mbb(s_m)$ is a non-trivial box. To this end, it utilizes sets $\mathcal{Q}$ and $\mathcal{Q}'$. Initially, $\mathcal{Q}$ contains only the component variable s. Then, the algorithm considers every set of component variables Σ' in $\Sigma_1, \ldots, \Sigma_m$ in turn.

Let us now suppose that the algorithm has processed sets Σ_1 to $\Sigma_{\mu-1}$ where $1 \leqslant \mu - 1 < m$. In this case, set Q contains all non-trivial boxes of the form $s \cap \sigma_1 \cap \cdots \cap \sigma_{\mu-1}$

[2] The interested reader is referred to [39,43] for more information about this approach.

Algorithm CHECKCONSTRAINTNTB
Input: Sets of component variables $\Sigma_1, \ldots, \Sigma_m$ that correspond to a certain region variable.
Output: 'True' if sets $\Sigma_1, \ldots, \Sigma_m$ satisfy Constraint NTB; 'False' otherwise.
Method:
For every s in $\Sigma_1 \cup \cdots \cup \Sigma_m$
  $\mathcal{Q} = \{s\}$
  For every Σ' in $\{\Sigma_1, \ldots, \Sigma_m\}$ Do
    $\mathcal{Q}' = \emptyset$
    For every s' in Σ' and every q in $\mathcal{Q}$ Do
      If $mbb(s') \cap mbb(q)$ in a non-trivial box Then $\mathcal{Q}' = \mathcal{Q}' \cup \{mbb(s') \cap mbb(q)\}$
    EndFor
    If $\mathcal{Q}' = \emptyset$ Then Return 'False'
    $\mathcal{Q} = \mathcal{Q}'$
  EndFor
EndFor
Return 'True'

Fig. 15. Algorithm CHECKCONSTRAINTNTB.

where $\sigma_i \in \Sigma_i$. Then, the algorithm considers the component variables of Σ_μ and the non-trivial boxes of Q. Algorithm CHECKCONSTRAINTNTB finds all regions $s' \in \Sigma_\mu$ and $q \in Q$ such that $s' \cap q$ is a non-trivial box and puts them into a new set Q'. In other words, set Q' contains all non-trivial boxes of the form $s \cap \sigma_1 \cap \cdots \cap \sigma_\mu$ where $\sigma_i \in \Sigma_i$. Hence, if Q' is empty, Constraint NTB is violated and Algorithm CHECKCONSTRAINTNTB returns 'Inconsistent'. Otherwise Q' is assigned to Q and the algorithm continues with the remaining sets of non-trivial boxes $\Sigma_{\mu+1}, \ldots, \Sigma_m$ that correspond to variable a.

Theorem 2. *Let $\Sigma_1, \ldots, \Sigma_m$ be sets of regions representing components of a given region. Algorithm* CHECKCONSTRAINTNTB *correctly decides whether $\Sigma_1, \ldots, \Sigma_m$ satisfy Constraint NTB.*

Proof. Let $\sigma \in \Sigma_1 \cup \cdots \cup \Sigma_m$. We can verify, from the above discussion, that when Algorithm CHECKCONSTRAINTNTB has processed sets $\Sigma_1, \ldots \Sigma_\mu$, $1 \leqslant \mu \leqslant m$, set Q contains all tuples $(s_1, \ldots, s_\mu) \in \Sigma_1 \times \cdots \times \Sigma_\mu$ such that $mbb(\sigma) \cap mbb(s_1) \cap \cdots \cap mbb(s_\mu)$ is a non-trivial box.

Therefore, if the algorithm returns 'False' for set $\Sigma_{\mu+1}$ then it means that $mbb(\sigma) \cap mbb(s_1) \cap \cdots \cap mbb(s_{\mu+1})$ is a non-trivial box for every $s_{\mu+1} \in \Sigma_{\mu+1}$, i.e., Constraint NTB is not satisfied.

If Algorithm CHECKCONSTRAINTNTB returns 'True' then Constraint NTB is satisfied because set Q is non-empty and contains all tuples $(s_1, \ldots, s_m) \in \Sigma_1 \times \cdots \times \Sigma_m$ such that $mbb(\sigma) \cap mbb(s_1) \cap \cdots \cap mbb(s_m)$ is a non-trivial box. □

Example 12. Let us continue with Example 11. For sets $\Sigma_1 = \{\delta_1, \delta_2\}$, $\Sigma_2 = \{\varepsilon_1, \varepsilon_2, \varepsilon_3\}$ and $\Sigma_3 = \{\zeta_1, \zeta_2, \zeta_3\}$, Constraint NTB holds (see also Fig. 14). It is not hard to verify that Algorithm CHECKCONSTRAINTNTB returns 'True' when it is called with input sets Σ_1, Σ_2 and Σ_3.

Algorithm GLOBALCHECKCONSTRAINTNTB
Input: A maximal solution v^0 of set O (produced by Algorithm TRANSFORM with input a set C of cardinal direction constraints in variables $a_1, \ldots, a_n$).
Output: 'True' if assignment v^0 satisfies Constraint NTB; 'False' otherwise.
Method:
For every region variable a in $\{a_1, \ldots, a_n\}$ Do
  Let $S_1, \ldots, S_m$ be the sets of component variables (introduced by Algorithm TRANSFORM) corresponding to a.
  Let $\Sigma_1, \ldots, \Sigma_m$ be the sets of boxes that v^0 assigns to the sets $S_1, \ldots, S_m$ of a respectively.
  If CHECKCONSTRAINTNTB($\Sigma_1, \ldots, \Sigma_m$) returns 'False' Then Return 'False'.
EndFor
Return 'True'

Fig. 16. Algorithm GLOBALCHECKCONSTRAINTNTB.

As another example, let us consider sets $\Sigma_1 = \{\delta_1, \delta_2, \delta_3\}$ and $\Sigma_2 = \{\varepsilon_1, \varepsilon_2, \varepsilon_3\}$ of Example 8 (see also Fig. 12). For these sets Algorithm CHECKCONSTRAINTNTB returns 'False' because Constraint NTB is not satisfied.

The third step of Algorithm CONSISTENCY uses Algorithm GLOBALCHECKCONSTRAINTNTB to check if the maximal solution v^0 of set O produced by Step 2 of Algorithm CONSISTENCY satisfies Constraint NTB. Algorithm GLOBALCHECKCONSTRAINTNTB considers every variable $a \in \{a_1, \ldots, a_n\}$ referenced in set C and forms all sets of component variables $S_1, \ldots, S_m$ that correspond to variable a. Notice that these sets were introduced by the first step of Algorithm CONSISTENCY (Algorithm TRANSFORM). Let now $\Sigma_1, \ldots, \Sigma_m$ be the sets of boxes that v^0 assigns to sets of component variables $S_1, \ldots, S_m$ respectively. Algorithm GLOBALCHECKCONSTRAINTNTB calls Algorithm CHECKCONSTRAINTNTB to check whether $\Sigma_1, \ldots, \Sigma_m$ satisfy Constraint NTB.

Here we finish our detailed discussion of the third and final step of Algorithm CONSISTENCY. The following section summarizes and presents a complete example of Algorithm CONSISTENCY.

4.4. Summary

In Sections 4.1, 4.2 and 4.3, we have presented in detail the three steps of Algorithm CONSISTENCY. This algorithm takes as input a set of basic cardinal direction constraints C in variables $a_1, \ldots, a_n$ and returns 'Consistent' if C is consistent; otherwise it returns 'Inconsistent'. Let us briefly summarize the three steps of Algorithm CONSISTENCY (Fig. 17).

- In the first step, Algorithm CONSISTENCY calls Algorithm TRANSFORM. Let O be the output of Algorithm TRANSFORM. Set O contains order constraints involving the projections on the x- and y-axis of region variables $a_1, \ldots, a_n$ and the component variables that correspond to $a_1, \ldots, a_n$ (see Section 4.1).
- The second step of Algorithm CONSISTENCY uses Algorithm CSPAN of [46] to find a solution s^0 of set O. If no solution exists then O (and also C) is inconsistent, thus Algorithm CONSISTENCY exits returning 'Inconsistent'. Otherwise, Algorithm CON-

Algorithm CONSISTENCY
Input: A set of basic cardinal direction constraints C in variables $a_1, \ldots, a_n$.
Output: 'Consistent' if C is consistent; 'Inconsistent' otherwise.
Method:

Step 1: Map the basic cardinal directions constraints of C into a set of order constraints O.

$O = \text{TRANSFORM}(C)$

Step 2: Find a maximal solution v^0 of O.

Find a solution s^0 of O (using Algorithm CSPAN of [46]).
If CSPAN returns 'Inconsistent' Then Return 'Inconsistent'
Find a maximal solution v^0 of O (using s^0 and Lemma 1).

Step 3: Check whether the maximal solution v^0 satisfies Constraint NTB.

If GLOBALCHECKCONSTRAINTNTB(v^0) returns 'False' Then Return 'Inconsistent'.

Return 'Consistent'

Fig. 17. Algorithm CONSISTENCY.

SISTENCY applies Lemma 1 to s^0 to derive a maximal solution v^0 of O (see also Section 4.2).

- In the third step, Algorithm CONSISTENCY considers Constraint NTB. If solution v^0 does not satisfies Constraint NTB then Algorithm CONSISTENCY exits returning 'Inconsistent'. This checking is performed using Algorithm GLOBALCHECKCONSTRAINTNTB (see also Section 4.3).

The three steps of Algorithm CONSISTENCY are based on the following theorem.

Theorem 3. *Let C be a set of basic cardinal direction relations and O be the set returned by Algorithm* TRANSFORM *when it is called with input C. Set C is consistent iff the following two conditions hold*:

1. *Set O has a maximal solution u^0.*
2. *Solution u^0 satisfies Constraint NTB.*

Proof. See Appendix B. An illustration of the structure of the proof is presented in Fig. 18. The fact that O is consistent iff it has a maximal solution was proven in Theorem 1 (lower-left part of Fig. 18). To prove that if C is consistent then the maximal solution of O satisfies Constraint NTB (only if—part of Theorem 3) we use Lemmata 1 and 2. Finally, to prove the if—part we use the values of the assignment of the maximal solution O and the fact that it satisfies Constraint NTB to construct regions that satisfy the cardinal direction constraints of C. □

The following theorem establishes the correctness of Algorithm CONSISTENCY.

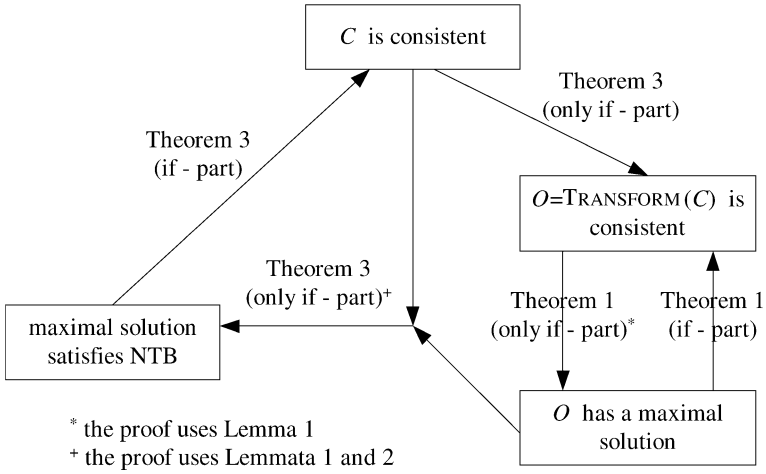

Fig. 18. Summary of the proof of Theorem 3.

Theorem 4. *Let C be a set of basic cardinal direction constraints. Algorithm* CONSISTENCY *correctly decides whether C is consistent.*

Proof. The correctness of Algorithm CONSISTENCY follows from the above discussion and Theorem 3. □

Let us now see an example of Algorithm CONSISTENCY in operation.

Example 13. Let us consider the constraint set

$$C = \{a_1\ B{:}N{:}E\ a_2,\ a_1\ B{:}S{:}W\ a_3,\ a_2\ SW\ a_3\}$$

of Example 3 and examine the three steps of Algorithm CONSISTENCY with input set C.

- In the first step, Algorithm CONSISTENCY calls Algorithm TRANSFORM to produce set O. The order constraints of O are presented in Examples 4–7.
- The second step of Algorithm CONSISTENCY uses Algorithm CSPAN to check the consistency and find a solution of set O. Set O is consistent and a solution is depicted in Fig. 12(a). Then, Algorithm CONSISTENCY applies Lemma 1 to construct a maximal solution of O. The maximal solution of the solution of O that corresponds to Fig. 12(a) is presented in Fig. 12(b) (see also Examples 8–10).
- In the third step, Algorithm CONSISTENCY calls Algorithm CHECKCONSTRAINT-NTB to check whether the maximal solution of the second step satisfies Constraint NTB. Using Fig. 12, we can see that Constraint NTB does not hold. Thus, Algorithm CONSISTENCY exits on Step 3 returning 'Inconsistent' (see also Example 12).

Algorithm CONSISTENCY is interesting in its own right, but it can also be used to compute the transitivity table for all basic cardinal direction relations defined in Section 3. For any pair of basic cardinal direction constraints $a\ R_1\ b$ and $b\ R_2\ c$, a basic cardinal direction

relation R_3, satisfies the cardinal direction constraint $a\ R_3\ c$ if and only if the constraint set $\{a\ R_1\ b,\ b\ R_2\ c,\ a\ R_3\ c\}$ is consistent (a more direct algorithm for this task is discussed in [42]). Similarly, Algorithm CONSISTENCY can be used to calculate the inverse of a given basic cardinal direction relation. For any basic cardinal direction constraints $a\ R_1\ b$, a basic cardinal direction relation R_2, satisfies the cardinal direction constraint $b\ R_2\ a$ if and only if the constraint set $\{a\ R_1\ b,\ b\ R_2\ a\}$ is consistent.

5. Complexity of consistency checking

In this section, we study the computational complexity of the consistency checking problem for cardinal direction constraints. Section 5.1 studies the aforementioned problem for *basic* cardinal direction constraints (i.e., non-disjunctive) while Section 5.2 considers this problem in its generality and studies *unrestricted* cardinal direction constraints (i.e., disjunctive and non-disjunctive).

5.1. Consistency of basic cardinal direction constraints

In Section 4, we have presented Algorithm CONSISTENCY that decides the consistency of a given set of basic cardinal direction constraints. The following theorem calculates the time complexity of Algorithm CONSISTENCY.

Theorem 5. *Deciding the consistency of a set of basic cardinal direction constraints in n region variables can be done using Algorithm* CONSISTENCY *in* $\mathcal{O}(n^5)$ *time.*

Proof. Let the input of Algorithm CONSISTENCY be a set C of basic cardinal direction constraints in n region variables. The number of constraints in C is $\mathcal{O}(n^2)$.

The first step of Algorithm CONSISTENCY calls Algorithm TRANSFORM with input set C. The latter algorithm considers every constraint of C in turn and returns a set of order constraints O. Algorithm TRANSFORM introduces at most 9 new variables each time Step T1 is executed. Hence, the total number of region variables is $\mathcal{O}(n^2)$. Steps T1–T3 of Algorithm TRANSFORM add to O $\mathcal{O}(n^2)$ order constraints. Summarizing, Algorithm TRANSFORM runs in $\mathcal{O}(n^2)$ time and returns a set O containing $\mathcal{O}(n^2)$ order constraints in $\mathcal{O}(n^2)$ variables.

In the second step, Algorithm CONSISTENCY finds a maximal solution of set O. This can be done using Algorithm CSPAN of [46]. Algorithm CSPAN decides the consistency of a set of order constraints in k variables in $\mathcal{O}(k^2)$ time. Thus, using Algorithm CSPAN of [46], we can find a solution of set O in $\mathcal{O}(n^4)$ time. Then, Algorithm CONSISTENCY applies Lemma 1. This can be performed in $\mathcal{O}(n)$ time. Summarizing the second step of Algorithm CONSISTENCY can be done in $\mathcal{O}(n^4)$ time.

The third step of Algorithm CONSISTENCY uses Algorithms GLOBALCHECKCONSTRAINTNTB (Fig. 16) and CHECKCONSTRAINTNTB (Fig. 15) to check whether Constraint NTB is satisfied. The latter algorithm uses a set Q. We first need to measure the size of set Q. For a given variable a_i, $1 \leqslant i \leqslant n$, set C contains $\mathcal{O}(n)$ constraints of the form $a_i\ R\ a_j$. Now since a_i participates in $\mathcal{O}(n)$ constraints of C, it follows that the region

represented by variable a_i is divided by $\mathcal{O}(n)$ horizontal and $\mathcal{O}(n)$ vertical lines. Thus a_i is divided into $\mathcal{O}(n^2)$ pieces. Now notice that set Q cannot contain more members than the possible pieces of a_i, thus the size of Q is $\mathcal{O}(n^2)$.

In order to check whether Constraint NTB is satisfied, Algorithm CONSISTENCY calls Algorithm GLOBALCHECKCONSTRAINTNTB which in turn calls $\mathcal{O}(n)$ times Algorithm CHECKCONSTRAINTNTB. The latter algorithm performs three nested loops. The outer two loop is executed $\mathcal{O}(n)$ times. Both the inner loops are performed at most $\mathcal{O}(n^3)$ times. Thus, checking whether Constraint NTB is satisfied can be done in $\mathcal{O}(n^5)$ time.

Summarizing, the complexity of Algorithm CONSISTENCY is $\mathcal{O}(n^5)$. □

5.2. Consistency of arbitrary cardinal direction constraints

We will now turn our attention to the consistency checking problem of a set of unrestricted cardinal direction relations expressed in the model of Section 3 (i.e., a set that includes both disjunctive and non-disjunctive cardinal direction constraints).

Theorem 6. *Deciding the consistency of a set of cardinal direction constraints is* NP*-complete.*

Proof. Let C be a set of cardinal direction constraints. Deciding the consistency of C is easily seen to be in NP. A nondeterministic algorithm first constructs a new set C' as follows. For every cardinal direction constraint $(a\ \{R_1, \ldots, R_m\}\ b) \in C$, the algorithm guesses a basic cardinal direction relation R_i among $\{R_1, \ldots, R_m\}$ and adds constraints $a\ R_i\ b$, $1 \leqslant i \leqslant m$, to set C'. Then, the nondeterministic algorithm checks to see whether the new set C' is consistent. This can be done with Algorithm CONSISTENCY in $\mathcal{O}(n^5)$ (Theorem 5).

To prove NP-hardness, we will use a reduction from 3SAT [33]. We construct an equivalent, with respect to consistency, mapping from a 3SAT formula to a set of cardinal direction constraints. In the construction, we map each literal of 3SAT to a region variable and each clause of 3SAT to a set of cardinal direction constraints.

This proof borrows some ideas from a proof that appears in [47] (like the use of a center region). It differentiates in the way we use relations and auxiliary regions.

Similarly to [47], we need a region o that denotes our center (Fig. 19). Regions that fall west of o correspond to false values while regions that fall east of o correspond to true values.

For each literal p in the 3SAT formula and its negation $\neg p$ we create a pair of regions s_p and $s_{\neg p}$. These regions are related to the center o using region $e_{p,\neg p}$ as follows:

$$e_{p,\neg p}\ B{:}W{:}E\ o, \qquad s_p\ \{B{:}W, B{:}E\}\ e_{p,\neg p}, \qquad s_{\neg p}\ \{B{:}W, B{:}E\}\ e_{p,\neg p},$$
$$s_p\ \{W, E\}\ o, \qquad s_{\neg p}\ \{W, E\}\ o, \qquad s_p\ \{W, E\}\ s_{\neg p}.$$

Intuitively, we use region $e_{p,\neg p}$ to ensure that regions s_p and $s_{\neg p}$ cannot both be true or both be false (Fig. 20). For instance, if region s_p falls into area A_1 of Fig. 20 (i.e., p is false) then the above constraints guarantee that region $s_{\neg p}$ falls into area A_2 (i.e., $\neg p$ is true).

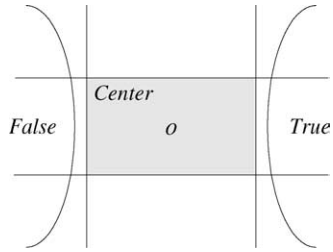

Fig. 19. Region o that denotes the center.

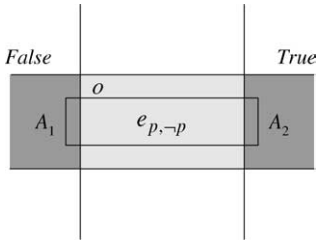

Fig. 20. Region $e_{p,\neg p}$ and its use.

Then, for each clause $p \vee q \vee r$ we create the following constraints.

$s_p \{W, E\} s_q$,
$s_q \{W, E\} s_r$,
$s_r \{W, E\} s_p$.

The above constraints ensure that regions s_p, s_q and s_r are disjoint. Moreover, we introduce the following constraints.

$$b_{p\vee q\vee r}\ B{:}W{:}E\ o, \qquad a_{p\vee q\vee r}\ W\ o, \qquad a_{p\vee q\vee r}\ B{:}W\ b_{p\vee q\vee r},$$
$$s_p \{E, B{:}W, B{:}E\}\ a_{p\vee q\vee r}, \qquad s_p \{W, E, B{:}E\}\ b_{p\vee q\vee r},$$
$$s_q \{E, B{:}W, B{:}E\}\ a_{p\vee q\vee r}, \qquad s_q \{W, E, B{:}E\}\ b_{p\vee q\vee r},$$
$$s_r \{E, B{:}W, B{:}E\}\ a_{p\vee q\vee r}, \qquad s_r \{W, E, B{:}E\}\ b_{p\vee q\vee r}.$$

The key to this encoding is that no more that two of the clauses regions (i.e., s_p, s_q and s_r) are allowed to be in the false area A_1 of Fig. 20. Therefore, at least one region will lie in the true area A_2 of Fig. 20 and thus its corresponding literal will be true. For instance, if both s_p and s_q are in the false area then the above constraints force s_r to be in the true area (see also Fig. 21).

To conclude this proof, we note that the above encoding can be performed in time polynomial in the length of the formula. It follows, from the above discussion, that the 3SAT formula is consistent iff its encoding to cardinal direction constraints is consistent. Moreover, since 3SAT is NP-complete, it follows that checking the consistency of a set of cardinal direction constraints is also NP-complete. □

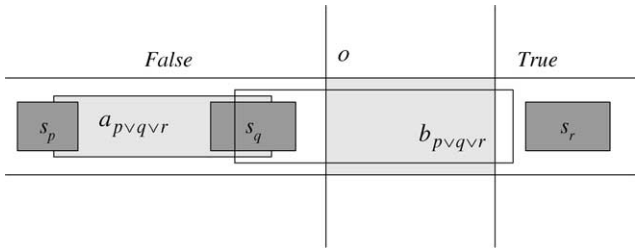

Fig. 21. Regions $a_{p\vee q\vee r}$ and $b_{p\vee q\vee r}$.

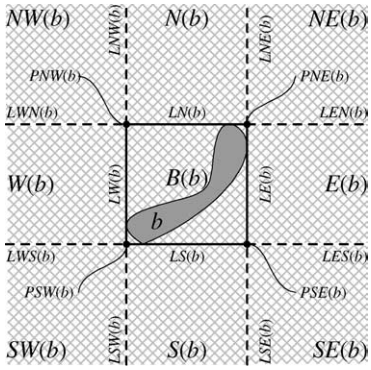

Fig. 22. Including points and lines.

6. Extensions

In Section 3, we have presented a model defining cardinal direction relations for the disconnected regions in REG^*. In this section, we will present an interesting variation that accommodates arbitrary regions in $\mathbb{R}^2$ [18,42]. In other words, this variation also considers points, lines and regions with emanating lines (see Fig. 4). Such regions have been excluded carefully from REG^* (they are not homeomorphic to the unit disk) but they can be easily included by dividing the space around the reference region b into the following 25 areas (see also Fig. 22):

- 9 two-dimensional areas ($B(b)$, $S(b)$, $SW(b)$, $W(b)$, $NW(b)$, $N(b)$, $NE(b)$, $E(b)$, $SE(b)$). These areas are formed by the axis of the bounding box of the reference region b (grey shaded areas of Fig. 22). Notice that each area does not include the parts of the axis forming it (contrary to the model of Section 3). The above areas correspond to the bounding box and the 8 cardinal directions.
- 8 semi-lines ($LSW(b), LWS(b), LWN(b), LNW(b), LNE(b), LEN(b), LES(b), LSE(b)$). These semi-lines are formed by the vertical and horizontal lines that start from the corners of the bounding box of the reference region b (dotted lines of Fig. 22). Notice that each semi-line does not include the corner of the bounding box.

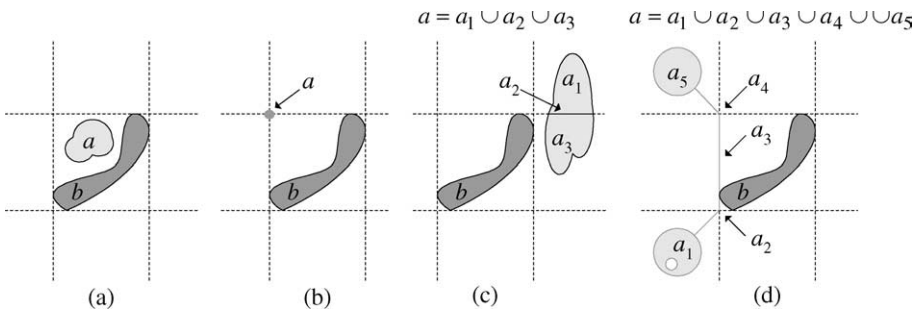

Fig. 23. Regions in $\mathbb{R}^2$ and relations in $\mathcal{D}^{\mathbb{R}^2}$.

- 4 line segments ($LS(b)$, $LW(b)$, $LN(b)$, $LE(b)$). These lines segments correspond to the sides of the bounding box of the reference region b (solid lines of Fig. 22). Notice that each line segment does not include the corners of the bounding box.
- 4 points ($PSW(b)$, $PNW(b)$, $PNE(b)$, $PSE(b)$). These points correspond to the corners of the bounding box of the reference region b.

The above partition of the reference space should be contrasted to the partition of Section 3 that divides the space into 9 areas. The new set, denoted by $\mathcal{D}^{\mathbb{R}^2}$, contains $\sum_{i=1}^{25} \binom{25}{i} = 33{,}554{,}431$ jointly exhaustive and pairwise disjoint cardinal direction relations.

The single-tile cardinal direction relations in $\mathcal{D}^{\mathbb{R}^2}$ are defined analogously to Definition 2. For instance:

$$a\ B\ b \quad \text{iff} \quad inf_x(b) < inf_x(a),\ sup_x(a) < sup_x(b),\ inf_y(b) < inf_y(a), \text{ and } sup_y(a) < sup_y(b).$$
$$a\ PNW\ b \quad \text{iff} \quad inf_x(a) = sup_x(a) = inf_x(b) \text{ and } inf_y(a) = sup_y(a) = sup_y(b).$$

Regions involved in these relations are shown in Figs. 23(a) and 23(b) respectively.

The multi-tile cardinal direction relations in $\mathcal{D}^{\mathbb{R}^2}$ are defined analogously to Definition 3. For instance:

$a\ NE{:}LEN{:}E\ b$ iff there exist regions a_1, a_2 and a_3 in $\mathbb{R}^2$ such that $a = a_1 \cup a_2 \cup a_3$, $a_1\ NE\ b$, $a_2\ LEN\ b$ and $a_3\ E\ b$.

$a\ SW{:}PSW{:}LW{:}PNW{:}NW\ b$ iff there exist regions $a_1, \ldots, a_5$ in $\mathbb{R}^2$ such that $a = a_1 \cup a_2 \cup a_3 \cup a_4 \cup a_5$, $a_1\ SW\ b$, $a_2\ PSW\ b$, $a_3\ PNW\ b$, $a_4\ LW\ b$ and $a_5\ NW\ b$.

Regions involved in these relations are shown in Figs. 23(c) and 23(d) respectively.

Algorithm CONSISTENCY presented in Section 4 can be modified in order to handle the consistency checking of a given set of cardinal direction constraints involving relations of $\mathcal{D}^{\mathbb{R}^2}$. Such modifications take place in Algorithms TRANSFORM and CHECKCONSTRAINTNTB. More specifically, the modifications of Algorithm TRANSFORM (Fig. 10) are as follows:

- Step T1 now takes into account the definition of relations in $\mathbb{R}^2$. These definitions can be derived similarly to the case of cardinal direction relations for regions in REG^* (Section 3).
- The constraint added by Step T2 changes to:

$$O = O \cup \{inf_x(r) \leqslant sup_x(r),\ inf_y(r) \leqslant sup_y(r)\}.$$

  This is so because Proposition 1 does not hold for regions in $\mathbb{R}^2$. For instance, for a point $p(\chi, \psi) \in \mathbb{R}^2$ we have $\chi = inf_x(p) = sup_x(p)$ and $\psi = inf_y(p) = sup_y(p)$.
- Step T4, for regions in REG^*, considers a cardinal direction relation $R_1{:}\cdots{:}R_m$ and checks whether the set of relations $\{R_1, \ldots, R_k\}$ is a subset of 8 sets. In the case of cardinal direction relations for regions in $\mathbb{R}^2$, we have to check a relation R against 39 sets. In order to present these sets, we will first need to define the following sets (see also Fig. 24):

$$\mathbf{A} = \{NW, LWN, W, LSW, SW\},$$
$$\mathbf{B} = \{LNW, PNW, LW, PSW, LSW\},$$
$$\mathbf{C} = \{N, LN, B, LS, S\},$$
$$\mathbf{D} = \{LNE, PNE, LE, PSE, LSE\},$$
$$\mathbf{E} = \{NE, LEN, E, LES, SE\},$$
$$\mathbf{A}' = \{NE, LNE, N, LNW, NW\},$$
$$\mathbf{B}' = \{LEN, PNE, LN, PNW, LWN\}$$
$$\mathbf{C}' = \{E, LE, B, LW, W\},$$
$$\mathbf{D}' = \{LSE, PSE, LS, PSW, LWS\},$$
$$\mathbf{E}' = \{SE, LSE, S, LSW, SW\}.$$

  Using the above sets we can express the 39 sets, we are looking for, as follows:

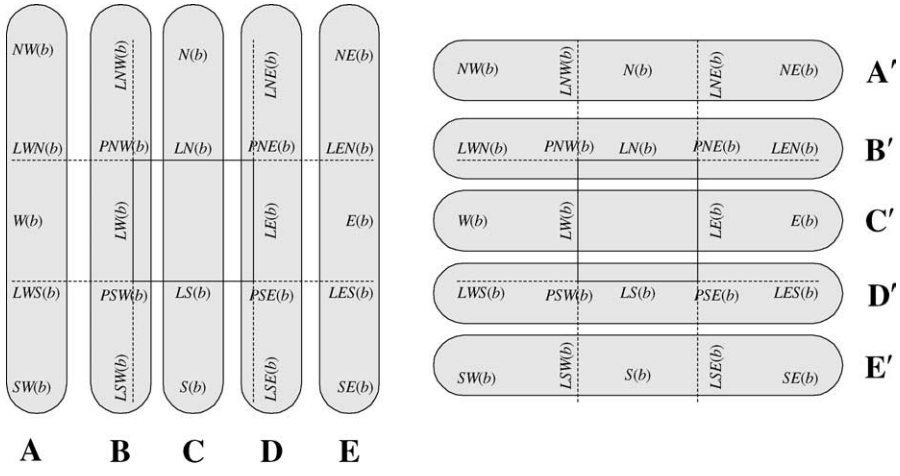

Fig. 24. Groups of basic relation for regions in $\mathbb{R}^2$.

$\mathbf{A}$,	$\mathbf{E}$,
$\mathbf{A} \cup \mathbf{B}$,	$\mathbf{E} \cup \mathbf{D}$,
$\mathbf{A} \cup \mathbf{B} \cup \mathbf{C}$,	$\mathbf{E} \cup \mathbf{D} \cup \mathbf{C}$,
$\mathbf{A} \cup \mathbf{B} \cup \mathbf{C} \cup \mathbf{D}$,	$\mathbf{E} \cup \mathbf{D} \cup \mathbf{C} \cup \mathbf{B}$,
$\mathbf{B}$,	$\mathbf{D}$,
$\mathbf{B} \cup \mathbf{C}$,	$\mathbf{D} \cup \mathbf{C}$,
$\mathbf{B} \cup \mathbf{C} \cup \mathbf{D}$,	$\mathbf{D} \cup \mathbf{C} \cup \mathbf{B}$,
$\mathbf{C}$,	$\mathbf{C}$,
$\mathbf{C} \cup \mathbf{D}$,	$\mathbf{C} \cup \mathbf{B}$,
$\mathbf{A}'$,	$\mathbf{E}'$,
$\mathbf{A}' \cup \mathbf{B}'$,	$\mathbf{E}' \cup \mathbf{D}'$,
$\mathbf{A}' \cup \mathbf{B}' \cup \mathbf{C}'$,	$\mathbf{E}' \cup \mathbf{D}' \cup \mathbf{C}'$,
$\mathbf{A}' \cup \mathbf{B}' \cup \mathbf{C}' \cup \mathbf{D}'$,	$\mathbf{E}' \cup \mathbf{D}' \cup \mathbf{C}' \cup \mathbf{B}'$,
$\mathbf{B}'$,	$\mathbf{D}'$,
$\mathbf{B}' \cup \mathbf{C}'$,	$\mathbf{D}' \cup \mathbf{C}'$,
$\mathbf{B}' \cup \mathbf{C}' \cup \mathbf{D}'$,	$\mathbf{D}' \cup \mathbf{C}' \cup \mathbf{B}'$,
$\mathbf{C}'$,	$\mathbf{C}'$,
$\mathbf{C}' \cup \mathbf{D}'$,	$\mathbf{C}' \cup \mathbf{B}'$.

For brevity we will show only the constraints added by Step T4 for the first 9 cases:

If $k > 1$ Then

If	$\{R_1, \ldots, R_k\} \subseteq \mathbf{A}$	Then	$O = O \cup \{sup_x(a_i) < inf_x(a_j)\}$
Else If	$\{R_1, \ldots, R_k\} \subseteq \mathbf{A} \cup \mathbf{B}$	Then	$O = O \cup \{sup_x(a_i) = inf_x(a_j)\}$
Else If	$\{R_1, \ldots, R_k\} \subseteq \mathbf{A} \cup \mathbf{B} \cup \mathbf{C}$	Then	$O = O \cup \{sup_x(a_i) < sup_x(a_j)\}$
Else If	$\{R_1, \ldots, R_k\} \subseteq \mathbf{A} \cup \mathbf{B} \cup \mathbf{C} \cup \mathbf{D}$	Then	$O = O \cup \{sup_x(a_i) = sup_x(a_j)\}$
Else If	$\{R_1, \ldots, R_k\} \subseteq \mathbf{B}$	Then	$O = O \cup \{inf_x(a_i) = inf_x(a_j), sup_x(a_i) = inf_x(a_i)\}$
Else If	$\{R_1, \ldots, R_k\} \subseteq \mathbf{B} \cup \mathbf{C}$	Then	$O = O \cup \{inf_x(a_i) = inf_x(a_j), sup_x(a_i) < sup_x(a_j)\}$
Else If	$\{R_1, \ldots, R_k\} \subseteq \mathbf{B} \cup \mathbf{C} \cup \mathbf{D}$	Then	$O = O \cup \{inf_x(a_i) = inf_x(a_j), sup_x(a_i) = sup_x(a_j)\}$
Else If	$\{R_1, \ldots, R_k\} \subseteq \mathbf{C}$	Then	$O = O \cup \{inf_x(a_j) < inf_x(a_i), sup_x(a_i) < sup_x(a_j)\}$
Else If	$\{R_1, \ldots, R_k\} \subseteq \mathbf{C} \cup \mathbf{D}$	Then	$O = O \cup \{inf_x(a_j) < inf_x(a_i), sup_x(a_i) = sup_x(a_j)\}$

...

EndIf

Finally, in Algorithm CHECKCONSTRAINTNTB we only have to change line:

$$\underline{\text{If}}\ mbb(s') \cap mbb(q) \text{ in a non-trivial box} \quad \underline{\text{Then}}\ \mathcal{Q}' = \mathcal{Q}' \cup \{mbb(s') \cap mbb(q)\}$$

into:

$$\underline{\text{If}}\ mbb(s') \cap mbb(q) \text{ in non-empty} \quad \underline{\text{Then}}\ \mathcal{Q}' = \mathcal{Q}' \cup \{mbb(s') \cap mbb(q)\}$$

simply because contrary to *REG**, our new domain $\mathbb{R}^2$, contains regions (e.g., points and lines) that can be trivial boxes.

The proof of correctness for the case of cardinal direction relations in $\mathcal{D}^{\mathbb{R}^2}$ is similar to the proof of correctness of Algorithm CONSISTENCY (Theorem 4). We can first generalize Theorem 1 and Lemmata 1 and 2 to handle the case of constraints in $\mathcal{D}^{\mathbb{R}^2}$. Then, the proof is analogous to the proof of Theorem 4 (see also Fig. 18).

7. Conclusions and future work

In this paper, we have presented a formal model for qualitative spatial reasoning with cardinal directions. This model can handle extended regions that might be disconnected or even have holes. Then, we have studied the problem of checking the consistency of a set of cardinal direction constraints that can be expressed in our model. We have presented the first algorithm for this problem, proved its correctness and analyzed its computational complexity. An implementation of this algorithm is available to interested researchers, from the first author of this paper. Moreover, we have outlined a modification of the consistency algorithm that can be used for an interesting extension of the model of Section 3.

With respect to the cardinal direction constraints the following are interesting open problems:

The consistency checking problem for connected regions. In this paper, we have considered the consistency checking problem of a given set C of cardinal direction constraints expressed on region variables $a_1, \ldots, a_n$ ranging over the (possibly disconnected) regions of class *REG**. We also intend to study an interesting restriction of the consistency checking problem that requires all region variables $a_1, \ldots, a_n$ to range over the connected regions of class *REG*. This problem is open. For example, it is not clear how to extend the proof of Theorem 3 so that the solution $(\rho_{a_1}, \ldots, \rho_{a_1})$ that we have constructed is formed only by the connected region of *REG* (see Appendix B, p. 132).

A complete complexity analysis of the consistency checking problem. In Sections 4 and 5, we have presented the first complexity analysis for the consistency problem. Specifically, we have seen than the consistency problem for a set of basic cardinal direction constraints can be solved in PTIME while the consistency problem of an unrestricted set of cardinal direction constraints is NP-complete. Following, the line of research of [3,19,21,25,30,38], we plan to continue and complete this analysis. To this end, one should first investigate the existence of classes of directional constraints (other than the class of non-disjunctive constraints) with a

polynomial consistency checking problem. Another issue that should also be addressed is whether the above classes are *maximal* (informally, a class is maximal if any extension of the class leads to NP-completeness). Answers to these problems will help us to exploit the frontier between tractable and possibly intractable cases.

Introducing cardinal direction constraints into a database model. We also intend to study other interesting problems for cardinal direction relations like *variable elimination, minimal network computation, global consistency enforcement, entailment*, etc. The important practical aspect of this research will be the development of all the required theory that will allow us to integrate cardinal direction constraints into a constraint database model like of instance [22–24].

Unified model for spatial information. Finally, we would like to combine the present model with the topological constraints framework of [12,36] and with the direction relations of [14,48] to devise a unified spatial reasoning formalism which can cope simultaneously with cardinal directions, topology and distance.

Acknowledgements

We would like to thank Timos Sellis for his comments and guidance. We are also grateful to the referees of this paper whose comments and suggestions have led to substantial improvements.

Appendix A. Proof of Lemma 1

We will prove that v^0 is a solution of O. By contradiction, let us assume that v^0 is not a solution of O. This can happen in four cases:

(i) There is a component variable a_{im} such that the value for $inf_x(\alpha_{im})$ in v^0 falsifies one of the constraints of O.
(ii) There is a component variable a_{im} such that the value for $sup_x(\alpha_{im})$ in v^0 falsifies one of the constraints of O.
(iii) There is a component variable a_{im} such that the value for $inf_y(\alpha_{im})$ in v^0 falsifies one of the constraints of O.
(iv) There is a component variable a_{im} such that the value for $sup_y(\alpha_{im})$ in v^0 falsifies one of the constraints of O.

Let us first assume that case (i) is what happens and consider every possible constraint in O that could involve $inf_x(\alpha_{im})$. Such a constraint can possibly belong to the following three categories:

1. The constraint was introduced in Step T1 of the Algorithm TRANSFORM. This happens when TRANSFORM is called with input $a_i\ R_1{:}\cdots{:}R_k\ a_j$ where $1 \leqslant j \leqslant n$.

The form of the constraint of O depends on the single-tile cardinal direction constraint $a_{im}\ R\ a_j$ considered for variable a_{im} by Algorithm TRANSFORM (where R is one of $R_1, \ldots, R_k$ and $1 \leqslant j \leqslant n$). There are nine such possible constraints:

(a) $\alpha_{im}\ NW\ \alpha_j$
The translation of this constraint into order constraints (Section 3.1) is

$$sup_x(\alpha_{im}) \leqslant inf_x(\alpha_j) \quad \text{and} \quad sup_y(\alpha_j) \leqslant inf_y(\alpha_{im}).$$

Since no order constraints involving $inf_x(\alpha_{im})$ are introduced, this case is impossible.

(b) $\alpha_{im}\ W\ \alpha_j$
This case is impossible as well (similar to 1(a), no order constraints involving $inf_x(\alpha_{im})$ are introduced).

(c) $\alpha_{im}\ SW\ \alpha_j$
This case is impossible as well (similar to 1(a), no order constraints involving $inf_x(\alpha_{im})$ are introduced).

(d) $\alpha_{im}\ N\ \alpha_j$
In this case, the constraint involving $inf_x(\alpha_{im})$ introduced by Algorithm TRANSFORM is: $inf_x(\alpha_j) \leqslant inf_x(\alpha_{im})$ (M1). According to the substitutions of Lemma 1, the possible values for $inf_x(\alpha_{im})$ in v^0 are $inf_x(\alpha_j)$ and $inf_x(\alpha_i)$.
- If $inf_x(\alpha_{im}) = inf_x(\alpha_j)$ holds in v^0, obviously, M1 is not falsified.
- If $inf_x(\alpha_{im}) = inf_x(\alpha_i)$ holds in v^0, we will prove that M1 is not falsified.
  Since $\alpha_{im}\ N\ \alpha_j$ and $inf_x(\alpha_{im}) = inf_x(\alpha_i)$ hold, it follows that $\{R_1, \ldots, R_k\} \not\subseteq \{NE, E, SE\}$ and $\{R_1, \ldots, R_k\} \subseteq \{NE, E, SE, N, B, S\}$ (see Fig. A.1). Therefore, when Algorithm TRANSFORM processes constraint $a_i\ R_1{:}\cdots{:}R_k\ a_j$, it will introduce an order constraint $inf_x(\alpha_j) \leqslant inf_x(\alpha_i)$ (in the first ElseIf statement of Step T4). This order constraint will of course end up in O. Notice that this constraint implies M1.

Therefore, this case is impossible as well.

(e) $\alpha_{im}\ B\ \alpha_j$
This case is impossible as well (similar to 1(d), the constraint involving $inf_x(\alpha_{im})$ is M1 which is not falsified by the substitutions of Lemma 1).

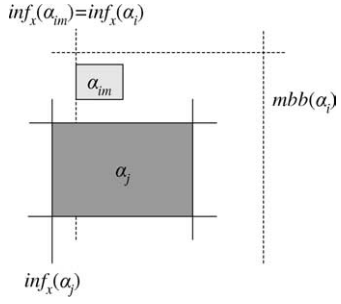

Fig. A.1. Proving Lemma 1 case 1(d).

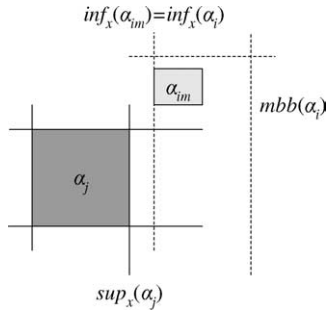

Fig. A.2. Proving Lemma 1 case 1(g).

(f) $\alpha_{im}\ S\ \alpha_j$
This case is impossible as well (similar to 1(d), the constraint involving $inf_x(\alpha_{im})$ is M1 which is not falsified by the substitutions of Lemma 1).

(g) $\alpha_{im}\ NE\ \alpha_j$
In this case, the constraint involving $inf_x(\alpha_{im})$ introduced by Algorithm TRANSFORM is: $sup_x(\alpha_j) \leqslant inf_x(\alpha_{im})$ (M2). According to the substitutions of Lemma 1, the possible values for $inf_x(\alpha_{im})$ in v^0 are $sup_x(\alpha_j)$ and $inf_x(\alpha_i)$.
- If $inf_x(\alpha_{im}) = sup_x(\alpha_j)$ holds in v^0, obviously, M2 is not falsified.
- If $inf_x(\alpha_{im}) = inf_x(\alpha_i)$ holds in v^0, we will prove that M2 is not falsified. Since $\alpha_{im}\ NE\ \alpha_j$ and $inf_x(\alpha_{im}) = inf_x(\alpha_i)$ hold, $\{R_1, \ldots, R_k\} \subseteq \{NE, E, SE\}$ (see Fig. A.2). Therefore, when Algorithm TRANSFORM processes constraint $a_i\ R_1{:}\cdots{:}R_k\ a_j$, it will introduce an order constraint $sup_x(\alpha_j) \leqslant inf_x(\alpha_i)$ (in the first If statement of Step T4). This order constraint will of course end up in O. Notice that this constraint implies M2.

Therefore, this case is impossible as well.

(h) $\alpha_{im}\ E\ \alpha_j$
This case is impossible as well (similar to 1(g), the constraint involving $inf_x(\alpha_{im})$ is M2 which is not falsified by the substitutions of Lemma 1).

(i) $\alpha_{im}\ SE\ \alpha_j$
This case is impossible as well (similar to 1(g), the constraint involving $inf_x(\alpha_{im})$ is M2 which is not falsified by the substitutions of Lemma 1).

2. The constraint was introduced in Step T2 of Algorithm TRANSFORM. In this case, the constraint would be $inf_x(\alpha_{im}) < sup_x(\alpha_{im})$ (M3). According to the substitutions of Lemma 1, the possible values for $inf_x(\alpha_{im})$ in v^0 are $inf_x(\alpha_j)$ and $inf_x(\alpha_i)$.
- If $inf_x(\alpha_{im}) = inf_x(\alpha_j)$ holds in v^0, then it would be either $\alpha_{im}\ N\ \alpha_j$ or $\alpha_{im}\ B\ \alpha_j$ or $\alpha_{im}\ S\ \alpha_j$. For all these cases, Step T1 of Algorithm TRANSFORM introduces constraint $inf_x(\alpha_j) \leqslant inf_x(\alpha_{im})$ (see Fig. A.3(a)).
  Therefore, substituting the value for $inf_x(\alpha_{im})$ in s^0 by a smaller or equal value will not falsify (M3).
- Let us now assume that $inf_x(\alpha_{im}) = inf_x(\alpha_i)$ holds in v^0. Step T3 of Algorithm TRANSFORM introduces constraint $inf_x(\alpha_i) \leqslant inf_x(\alpha_{im})$ (see Fig. A.3(b)).
  Therefore, substituting the value for $inf_x(\alpha_{im})$ in s^0 by a smaller or equal value will not falsify (M3).

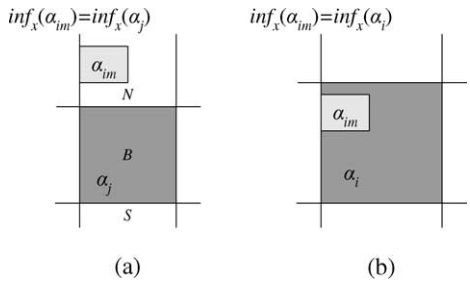

Fig. A.3. Proving Lemma 1 case 2.

3. The constraint was introduced in Step T3 of the Algorithm TRANSFORM. In this case, the constraint would be $inf_x(a_i) \leqslant inf_x(a_{im})$ and it is not falsified by the substitutions that created v^0.

Summarizing, we have proved that case (i) considered at the beginning of the proof is impossible. Similarly, we can show that cases (ii), (iii) and (iv) are also impossible (these cases are symmetric to case (i)). Therefore, our original assumption about v^0 does not hold, i.e., v^0 is a solution of O. □

Appendix B. Proof of Theorem 3

(Only if) Let us assume that the set of cardinal directions constraints C in variable $a_1, \ldots, a_n$ is consistent. Let O be the set returned by Algorithm TRANSFORM when it is called with input C. We will prove that

1. Set O has a maximal solution u^0.
2. Solution u^0 satisfies Constraint NTB.

We will show first that there is an ordinary solution s^0 of O that satisfy Constraint NTB. Then, we will construct from s^0 a maximal solution u^0 of O that still satisfies Constraint NTB.

If set C is consistent then there exist regions of REG^* that satisfies all constraints in C. Let $a\ R_1{:}\cdots{:}R_k\ b$ be an arbitrary constraint in C. Since this constraint is satisfied, there exist regions α, β and subregions $\alpha_1, \ldots, \alpha_k$ of α such that $\alpha = \alpha_1 \cup \cdots \cup \alpha_k$ and $\alpha_1\ R_1\ \beta, \ldots, \alpha_k\ R_k\ \beta$ hold (see definitions in Section 3.1). The existence of all these regions and their subregions implies the existence of their projections on x- and y-axes. The endpoints of these projections satisfy all the order constraints introduced in set O by Algorithm TRANSFORM. This follows from the discussion in Section 4. Such endpoints form a solution s^0 of O. Moreover, it follows from Lemma 2 that solution s^0 also satisfies Constraint NTB.

Now let us use the substitutions of Lemma 1 to construct from s^0 a maximal solution u^0 of O. The maximal solution u^0 satisfies all constraints of set O (Lemma 1), thus we only have to prove that u^0 also satisfies Constraint NTB.

Constraint NTB contains expressions of the form:

$$mbb(s) \cap mbb(s_1) \cap \cdots \cap mbb(s_m) \text{ is a non-trivial box.} \qquad (Z)$$

Let $\sigma, \sigma_1, \ldots, \sigma_m$ be the regions that solution s^0 assigns to region variables $s, s_1, \ldots, s_m$ and $\sigma', \sigma'_1, \ldots, \sigma'_m$ be the regions that the maximal solution u^0 assigns to the same region variables.

Since solution s^0 satisfies Constraint NTB, we have:

$$mbb(\sigma) \cap mbb(\sigma_1) \cap \cdots \cap mbb(\sigma_m) \text{ is a non-trivial box.}$$

Regions $\sigma', \sigma'_1, \ldots, \sigma'_m$ are formed by extending regions $\sigma, \sigma_1, \ldots, \sigma_m$ respectively (using the substitutions of Lemma 1). Thus, we have:

$$\sigma \subseteq \sigma', \ \sigma_1 \subseteq \sigma'_1, \ \ldots, \ \sigma_m \subseteq \sigma'_m.$$

We can verify that regions $\sigma', \sigma'_1, \ldots, \sigma'_m$ also satisfy Expression Z, thus the maximal solution u^0 satisfies Constraint NTB.

(If) Let C be a set of cardinal direction constraints in variables $a_1, \ldots, a_n$ and O be the set returned by Algorithm TRANSFORM when it is called with input C. Let us assume that v^0 is a maximal solution of O and v^0 satisfies Constraint NTB. To prove that C is consistent, we will use v^0 to form regions $(\rho_{a_1}, \ldots, \rho_{a_n})$ that satisfy all constraints in C.

Let $\alpha_1, \ldots, \alpha_n$ be the non-trivial boxes that v^0 assigns to region variables $a_1, \ldots, a_n$ (see also Proposition 2). Let us now consider an arbitrary region variable a_i $(1 \leqslant i \leqslant n)$. Let us also assume that $C_{a_i} = \{c_1, \ldots, c_m\}$ contains all constraints of C with region a_i as the primary region. Let

$$\begin{aligned} c_1 &\equiv (a_i \ R_1^1{:}\cdots{:}R_{k_1}^1 \ a_{j_1}) \\ &\ \ \vdots \\ c_m &\equiv (a_i \ R_1^m{:}\cdots{:}R_{k_m}^m \ a_{j_m}) \end{aligned}$$

where $R_1^1{:}\cdots{:}R_{k_1}^1, \ldots, R_1^m{:}\cdots{:}R_{k_m}^m$ are cardinal direction relations and $a_{j_1}, \ldots, a_{j_m}$, $1 \leqslant j_1, \ldots, j_m \leqslant n$ are region variables. Every time one of these m constraints is processed by Algorithm TRANSFORM, a new set of component variables corresponding to a_i is introduced (Step T1). Let $S_1 = \{a_{i1}^1, \ldots, a_{ik_1}^1\}, \ldots, S_m = \{a_{i1}^m, \ldots, a_{ik_m}^m\}$ be all such sets of component variables.

Let $\alpha_{i1}^1, \ldots, \alpha_{ik_1}^1, \ldots, \alpha_{i1}^m, \ldots, \alpha_{ik_m}^k$ be the non-trivial boxes that v^0 assigns to region variables $a_{i1}^1, \ldots, a_{ik_1}^1, \ldots, a_{i1}^m, \ldots, a_{ik_m}^m$ respectively (see also Proposition 2).

Now let us consider the sets

$$\begin{aligned} &\Xi_1 = \Pi_x(\alpha_{i1}^1) \cup \cdots \cup \Pi_x(\alpha_{ik_1}^1), \qquad \Theta_1 = \Pi_y(\alpha_{i1}^1) \cup \cdots \cup \Pi_y(\alpha_{ik_1}^1) \\ &\ \ \vdots \\ &\Xi_m = \Pi_x(\alpha_{i1}^m) \cup \cdots \cup \Pi_x(\alpha_{ik_m}^m), \qquad \Theta_m = \Pi_y(\alpha_{i1}^m) \cup \cdots \cup \Pi_y(\alpha_{ik_m}^m) \end{aligned}$$

of the x- and y-axis formed by considering sets $S_1, \ldots, S_m$ in turn.

Notice that since v^0 is a maximal solution, all sets $\Xi_1, \ldots, \Xi_m$ have the same endpoints with set $\Pi_x(\alpha_i)$ and all sets $\Theta_1, \ldots, \Theta_m$ have the same endpoints with $\Pi_y(\alpha_i)$ (according to Lemma 1).

Let $\Delta_i^1, \ldots, \Delta_i^m$ be regions formed as follows:

$$\Delta_i^1 = \alpha_{i1}^1 \cup \cdots \cup \alpha_{ik_1}^1, \ \ldots, \ \Delta_i^m = \alpha_{i1}^m \cup \cdots \cup \alpha_{ik_m}^m.$$

Regions $\Delta_i^1, \ldots, \Delta_i^m$ are well-defined regions in REG^* since they are formed by the union of non-trivial boxes. From the above coincidence fact, regions $\alpha_i, \Delta_i^1, \ldots, \Delta_i^m$ have the same bounding box. Moreover, for any t, $1 \leqslant t \leqslant m$, regions Δ_i^t and α_{j_t} satisfy by construction the constraint $c_t \equiv (a_i \ R_1^t{:}\cdots{:}R_{k_t}^t \ a_{j_t})$.

Let us consider region o_{a_i} formed as follows:

$$o_{a_i} = \Delta_i^1 \cap \cdots \cap \Delta_i^m.$$

Equivalently, we have:

$$\begin{aligned} o_{a_i} &= \left(\alpha_{i1}^1 \cup \cdots \cup \alpha_{ik_1}^1\right) \cap \cdots \cap \left(\alpha_{i1}^m \cup \cdots \cup \alpha_{ik_m}^m\right) \\ &= \bigcup_{1 \leqslant s_1 \leqslant k_1, \ \ldots, \ 1 \leqslant s_m \leqslant k_m} \left(\alpha_{is_1}^1 \cap \cdots \cap \alpha_{is_m}^m\right). \end{aligned} \tag{B.1}$$

Each intersection $\alpha_{is_1}^1 \cap \cdots \cap \alpha_{is_m}^m$ $(1 \leqslant s_1 \leqslant k_1, \ldots, 1 \leqslant s_m \leqslant k_m)$ can be either *empty* or a *trivial box* or a *non-trivial* box.

We now form a region ρ_{a_i} defined as the union of all the intersections $\alpha_{is_1}^1 \cap \cdots \cap \alpha_{is_m}^m$ from Eq. (B.1) that are non-trivial boxes.

We can now prove the following:

1. ρ_{a_i} is a non-empty region in REG^*.
   Let us consider Eq. (B.1). Since u^0 satisfies Constraint NTB, that at least one of the intersections forming the union is a non-trivial box, therefore ρ_{a_i} is non-empty. Moreover, by definition, ρ_{a_i} is the union of non-trivial boxes and thus, is a region in REG^*.
2. The assignment $(a_i, a_{j1}, \ldots, a_{jm}) = (\rho_{a_i}, \alpha_{j_1}, \ldots, \alpha_{j_m})$ satisfies all constraints $c_1, \ldots, c_m$ in C_{a_i}.
   Let us consider an arbitrary constraint $c_t \equiv (a_i \ R_1^t{:}\cdots{:}R_{k_t}^t \ a_{j_t})$, $1 \leqslant t \leqslant m$, in C_{a_i}. Let us also consider the component variable $a_{i1}^t \in S_t$ of a_i and the box α_{i1}^t that v^0 assigns to a_{i1}^t. Since v^0 satisfies Constraint NTB there exist regions

   $$\alpha_{is_1}^1 \in \left\{\alpha_{i1}^1, \ldots, \alpha_{ik_1}^1\right\}, \ \ldots, \ \alpha_{is_m}^m \in \left\{\alpha_{i1}^m, \ldots, \alpha_{ik_m}^m\right\}$$

   (i.e., for any $1 \leqslant v \leqslant m$, $\alpha_{is_v}^v$ is a non-trivial box and a subregion of $\Delta_i^v = \alpha_{i1}^v \cup \cdots \cup \alpha_{ik_v}^v$) such that

   $$\mathcal{B} = \alpha_{i1}^t \cap \alpha_{is_1}^1 \cap \cdots \cap \alpha_{is_m}^m$$

   is a non-trivial box. Region $\mathcal{B}$ is a subregion of o_{a_i} (see Eq. (B.1)) and a non-trivial box thus it is also a subregion of ρ_{a_i}. Since non-trivial box α_{i1}^t lies completely in the R_1^t tile of α_{j_t} (from the definition of constraint c_t), box $\mathcal{B}$ also lies completely in the R_1^t tile of α_{j_t}. Thus region ρ_{a_i} has a subregion which is a non-trivial box and lies in the R_1^t tile of α_{j_t}. Similarly, we can prove that region ρ_{a_i} has a subregion which is a non-trivial box and lies in the R_s^t tile of α_{j_t} for any s such that $2 \leqslant s \leqslant k_t$.

Finally, we have to prove that region ρ_{a_i} lies completely in $R_1^t(\alpha_{j_t}) \cup \cdots \cup R_{k_t}^t(\alpha_{j_t})$. In other words we have to prove that for every $p \in \mathbb{R}^2 - (R_1^t(\alpha_{j_t}) \cup \cdots \cup R_{k_t}^t(\alpha_{j_t}))$ we have that $p \notin \rho_{a_i}$ holds. If $p \in \mathbb{R} - (R_1^t(\alpha_{j_t}) \cup \cdots \cup R_{k_t}^t(\alpha_{j_t}))$ holds, then $p \notin \Delta_i^t = a_{i1}^t \cup \cdots \cup a_{ik_t}^t$ also holds for every t such that $1 \leqslant t \leqslant m$. It follows from the definition of region o_{a_i} that $p \notin o_{a_i}$ and thus $p \notin \rho_{a_i}$ holds.
Therefore, $\rho_{a_i}\ R_1^t{:}\cdots{:}R_{k_t}^t\ \alpha_{j_t}$ holds which proves the proposition.

3. Regions ρ_{a_i} and α_i have the same bounding box, i.e., $mbb(\rho_{a_i}) = mbb(\alpha_i)$.
Let $\rho_{a_i}(\gamma_1, \ldots, \gamma_l)$, $l \leqslant m$, be a region formed as ρ_{a_i} but using only constraints $\gamma_1, \ldots, \gamma_l \in \{c_1, \ldots, c_m\}$. Notice that $\rho_{a_i}(c_1, \ldots, c_m) = \rho_{a_i}$, $\rho_{a_i}(c_t) = \Delta_i^t$ for every $1 \leqslant t \leqslant m$ and $\rho_{a_i}(c_1, \ldots, c_l) \cap \rho_{a_i}(c_{l+1}, \ldots, c_{l'}) = \rho_{a_i}(c_1, \ldots, c_l, c_{l+1}, \ldots, c_{l'})$ hold. We will prove that $inf_x(\rho_{a_i}(\gamma_1, \ldots, \gamma_l)) = inf_x(a_i)$, for $l \geqslant 2$. We will use induction on the number of constraints l.
For $l = 2$, let γ_1 and γ_2 be two arbitrary constraints in $\{c_1, \ldots, c_m\}$. When Algorithm TRANSFORM processes constraints γ_1 and γ_2 it introduces sets S_1 and S_2 of component variables corresponding to a_i respectively. Let Σ_1 and Σ_2 be the set of boxes that v^0 assigns to the component variables of S_1 and S_2 respectively. Let also Φ and Ψ be the sets containing all boxes in Σ_1 and Σ_2 respectively such that, for every $s \in \Phi \cup \Psi$, $inf_x(s) = inf_x(a_i)$ holds. Notice that since regions $\rho_{a_i}(\gamma_1)$ and $\rho_{a_i}(\gamma_2)$ have the same minimum bounding box we have $\Phi \neq \emptyset$ and $\Psi \neq \emptyset$.
Let us now assume that for all $\phi \in \Phi$ and $\psi \in \Psi$, the intersection $\phi \cap \psi$ is a trivial box (i.e., it is either empty or a point or a line segment). This contradicts the fact that v^0 satisfies Constraint NTB, thus there exist regions $u \in \Phi$ and $v \in \Psi$ such that $u \cap v$ is a non-trivial box. Moreover, $u \cap v$ is one of the intersections unioned to construct $\rho_{a_i}(\gamma_1, \gamma_2)$. Since $u \in \Phi$ and $v \in \Psi$, it is $inf_x(u) = inf_x(v) = inf_x(\alpha_i)$. Thus, $inf_x(\rho_{a_i}(\gamma_1, \gamma_2)) = inf_x(\alpha_i)$ (see Fig. B.1).
The inductive step is similar.
In a similar way, we can also prove that $sup_x(\rho_{a_i}) = sup_x(a_i)$, $inf_y(\rho_{a_i}) = inf_y(a_i)$ and $sup_y(\rho_{a_i}) = sup_y(a_i)$. Thus $mbb(\rho_{a_i}) = mbb(\alpha_i)$ holds.

Similarly to the construction of ρ_{a_i}, we can also form regions:

$$\rho_{a_1}, \ldots, \rho_{a_{i-1}}, \rho_{a_{i+1}}, \ldots, \rho_{a_n} \in REG^*.$$

Each region ρ_{a_i}, $1 \leqslant i \leqslant n$, is a non-empty, well-defined region in REG^*. Moreover, the assignment

$$(a_1, \ldots, a_n) = (\alpha_1, \ldots, \alpha_{i-1}, \rho_{a_i}, \alpha_{i+1}, \ldots, \alpha_n)$$

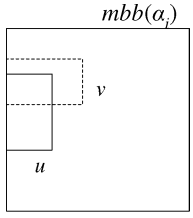

Fig. B.1. Proving that $mbb(\rho_{a_i}) = mbb(\alpha_i)$.

satisfies all constraints in C_{a_i}.

We will now show that the assignment $(a_1, \ldots, a_n) = (\rho_{a_1}, \ldots, \rho_{a_n})$ satisfies all constraints in C. Let $c \equiv (a_i \ R \ a_j)$ be a constraint in C. From the previous discussion, we know that regions ρ_{a_i} and α_j satisfy constraint c. Since $mbb(\alpha_j) = mbb(\rho_{a_j})$ it follows that regions ρ_{a_i} and ρ_{a_j} also satisfy constraint c. Therefore, the n-tuple $(\rho_{a_1}, \ldots, \rho_{a_n})$ is a solution of C. □

References

[1] A.I. Abdelmoty, B.A. El-Geresy, An intersection-based formalism for representing orientation relations in a geographic database, in: Proceedings of 2nd ACM Conference on Advances in GIS Theory, 1994.
[2] J.F. Allen, Maintaining knowledge about temporal intervals, Comm. ACM 26 (11) (1983) 832–843.
[3] P. Balbiani, J-F. Condotta, L.F. del Cerro, A new tractable subclass of the rectangle algebra, in: Proceedings of IJCAI'99, Stockholm, Sweden, 1999, pp. 442–447.
[4] A. Berretti, A. Del Bimbo, E. Vicario, Modeling spatial relationships between color sets, in: Proceedings of IEEE Workshop on Content-based Access of Image and Video Libraries (CVPR-2000), 2000.
[5] B. Bennett, Logical representations for automated reasoning about spatial relations, PhD Thesis, School of Computer Studies, University of Leeds, 1997.
[6] Z. Cui, A.G. Cohn, D.A. Randell, Qualitative and topological relationships in spatial databases, in: Proceedings of SSD'93, 1993, pp. 296–315.
[7] E. Clementini, P. Di Fellice, G. Califano, Composite regions in topological queries, Inform. Syst. 7 (1995) 594–759.
[8] E. Clementini, P. Di Fellice, D. Hernandez, Qualitative representation of positional information, Artificial Intelligence 95 (1997) 317–356.
[9] W.G. Chinn, N.E. Steenrod, First Concepts of Topology, The Mathematical Association of America, 1966.
[10] E. Davis, Representations of Commonsense Knowledge, Morgan Kaufmann, San Mateo, CA, 1990.
[11] J.P. Delgrande, A. Gupta, T. Van Allen, Point based approaches to qualitative temporal reasoning, in: Proceedings of the AAAI'99, Orlando, FL, 1999, pp. 739–744.
[12] M.J. Egenhofer, Reasoning about binary topological relationships, in: Proceedings of SSD'91, 1991, pp. 143–160.
[13] B. Faltings, Qualitative spatial reasoning using algebraic topology, in: Proceedings of COSIT'95, in: Lecture Notes in Comput. Sci., vol. 988, Springer, Berlin, 1995, pp. 17–30.
[14] A.U. Frank, Qualitative spatial reasoning about distances and directions in geographic space, J. Visual Languages Comput. 3 (1992) 343–371.
[15] C. Freksa, Using orientation information for qualitative spatial reasoning, in: Proceedings of COSIT'92, in: Lecture Notes in Comput. Sci., vol. 639, Springer, Berlin, 1992, pp. 162–178.
[16] R. Goyal, M.J. Egenhofer, The direction-relation matrix: a representation for directions relations between extended spatial objects, in: The Annual Assembly and the Summer Retreat of University Consortium for Geographic Information Systems Science, 1997.
[17] R. Goyal, M.J. Egenhofer, Cardinal directions between extended spatial objects, IEEE Trans. Data Knowledge Engrg., in press. Available at http://www.spatial.maine.edu/~max/RJ36.html.
[18] R. Goyal, M.J. Egenhofer, Consistent queries over cardinal directions across different levels of detail, in: Proceedings of the 11th International Workshop on Database and Expert Systems Applications, 2000.
[19] M. Grigni, D. Papadias, C. Papadimitriou, Topological inference, in: Proceedings of IJCAI'95, Montreal, Quebec, 1995.
[20] H.W. Guesgen, Spatial reasoning based on Allen's temporal logic, Technical Report TR-89-049, International Science Institute National Technical, Berkeley, 1989.
[21] P. Jonsson, T. Drakengren, A complete classification of tractability in RCC-5, J. Artificial Intelligence Res. 6 (1997) 211–221.
[22] P.C. Kanellakis, G.M. Kuper, P.Z. Revesz, Constraint query languages, J. Comput. System Sci. 51 (1995) 26–52.

[23] M. Koubarakis, The complexity of query evaluation in indefinite temporal constraint databases, in: L.V.S. Lakshmanan (Ed.), Theoret. Comput. Sci. 171 (1997) 25–60. Special Issue on Uncertainty in Databases and Deductive Systems.
[24] M. Koubarakis, S. Skiadopoulos, Querying temporal and spatial constraint networks in PTIME, Artificial Intelligence 123 (1–2) (2000) 223–263.
[25] G. Ligozat, Reasoning about cardinal directions, J. Visual Languages Comput. 9 (1998) 23–44.
[26] S. Lipschutz, General Topology, McGraw Hill, New York, 1965.
[27] S. Lipschutz, Set Theory and Related Topics, McGraw Hill, New York, 1998.
[28] D.M. Mark, C. Freksa, S.C. Hirtle, R. Lloyd, B. Tversky, Cognitive models of geographic space, Internat. J. Geograph. Inform. Sci. 13 (8) (1999) 747–774.
[29] A. Mukerjee, G. Joe, A qualitative model for space, in: Proceedings of AAAI'90, Boston, MA, 1990, pp. 721–727.
[30] B. Nebel, H.-J. Bürckert, Reasoning about temporal relations: a maximal tractable subclass of Allen's interval algebra, J. ACM 42 (1) (1995) 43–66.
[31] D. Papadias, N. Arkoumanis, N. Karacapilidis, On the retrieval of similar configurations, in: Proceedings of 8th International Symposium on Spatial Data Handling (SDH), 1998.
[32] D. Papadias, Relation-based representation of spatial knowledge, PhD Thesis, Department of Electrical and Computer Engineering, National Technical University of Athens, 1994.
[33] C.H. Papadimitriou, Computational Complexity, Addison Wesley, Reading, MA, 1994.
[34] J. Portugali, I. Omer, Systematic distortions in cognitive maps: the North American west coast vs. the (west) coast of Israel, in: Proceedings of COSIT'03, in: Lecture Notes in Comput. Sci., vol. 2825, Springer, Berlin, 2003, pp. 93–100.
[35] C.H. Papadimitriou, D. Suciu, V. Vianu, Topological queries in spatial databases, J. Comput. System Sci. 58 (1) (1999) 29–53.
[36] D.A. Randell, Z. Cui, A. Cohn, A spatial logic based on regions and connection, in: Principles of Knowledge Representation and Reasoning: Proceedings of the Third International Conference (KR'92), Cambridge, MA, Morgan Kaufmann, San Mateo, CA, 1992.
[37] J. Renz, Maximal tractable fragments of the region connection calculus: a complete analysis, in: Proceedings of IJCAI'99, Stockholm, Sweden, 1999, pp. 448–455.
[38] J. Renz, B. Nebel, On the complexity of qualitative spatial reasoning: a maximal tractable fragment of the region connection calculus, Artificial Intelligence 1–2 (1999) 95–149.
[39] S. Skiadopoulos, M. Koubarakis, Qualitative spatial reasoning with cardinal directions: semantics, algorithms and computational complexity, Technical Report TR-2000-3, National Technical University of Athens, 2000. Available at http://www.dblab.ece.ntua.gr/publications.
[40] S. Skiadopoulos, M. Koubarakis, Composing cardinal directions relations, in: Proceedings of the 7th International Symposium on Spatial and Temporal Databases (SSTD'01), in: Lecture Notes in Comput. Sci., vol. 2121, Springer, Berlin, 2001, pp. 299–317.
[41] S. Skiadopoulos, M. Koubarakis, Qualitative spatial reasoning with cardinal directions, in: Proceedings of the 7th International Conference on Principles and Practice of Constraint Programming (CP'02), in: Lecture Notes in Comput. Sci., vol. 2470, Springer, Berlin, 2002, pp. 341–355.
[42] S. Skiadopoulos, M. Koubarakis, Composing cardinal direction relations, Artificial Intelligence 152 (2) (2004) 143–171.
[43] S. Skiadopoulos, Query evaluation in constraint databases, PhD Thesis, School of Electrical and Computer Engineering, National Technical University of Athens, 2002.
[44] A.P. Sistla, C. Yu, R. Haddad, Reasoning about spatial relationships in picture retrieval systems, in: Proceedings of VLDB'94, 1994, pp. 570–581.
[45] B. Tversky, Distortions in memory for maps, Cognitive Psychology 13 (1981) 407–433.
[46] P. van Beek, Reasoning about qualitative temporal information, Artificial Intelligence 58 (1992) 297–326.
[47] M. Vilain, H. Kautz, P. van Beek, Constraint propagation algorithms for temporal reasoning: a revised report, in: D.S. Weld, J. de Kleer (Eds.), Readings in Qualitative Reasoning about Physical Systems, Morgan Kaufmann, San Mateo, CA, 1989, pp. 73–381.
[48] K. Zimmermannm, Enhancing qualitative spatial reasoning—combining orientation, in: Proceedings of COSIT'93, in: Lecture Notes in Comput. Sci., vol. 716, Springer, Berlin, 1993, pp. 69–76.