Dipartimento di Informatica
Università degli Studi di Verona

Rapporto di ricerca **90**
Research report
September 2013

Robustness of Spatial Relation Evaluation

Alberto Belussi
Sara Migliorini
Mauro Negri
Giuseppe Pelagatti

Questo rapporto è disponibile su Web all'indirizzo:
This report is available on the web at the address:
http://www.di.univr.it/report

Abstract

In the last few years the amount of spatial data available through the network has increased both in volume and in heterogeneity, so that dealing with this huge amount of information has become an interesting new research challenge. In particular, spatial data is usually represented through a vector model upon which several spatial relations have been defined. Such relations represent the basic tools for querying spatial data and their robust evaluation in a distributed heterogeneous environment is an important issue to consider, in order to allow an effective usage of this kind of data. Among all possible spatial relations, this report considers the topological ones, since they are the most widely available in existing systems and represent the building blocks for the implementation of other spatial relations.

The conditions and the operations needed to make a dataset robust w.r.t. topological interpretations strictly depends on the adopted evaluation model. In particular, this report considers an environment where two different evaluation models for topological relations exist, one in which equality is based on identity of geometric primitives, and the other one where a tolerance in equality evaluation is introduced. Given such premises, the report proposes a set of rules for guaranteeing the robustness in both models, and discusses the applicability of available algorithms of the Snap Rounding family, in order to preserve robustness in case of perturbations.

Keywords: Topological relations, Robustness, Spatial data infrastructure, Tolerance equality model, Identity equality model.

1 Introduction

Nowadays the amount of spatial data available through the network has considerably increased. Such data is usually characterized by a great heterogeneity that determines new problems during their management.

In geographical applications spatial data is usually described by means of two aspects: its geometrical position on the Earth surface, and the existing relation with surrounding objects. As regards to the first aspect, the geometry of spatial objects is usually represented through a vector model, upon which several spatial relations have been defined in order to manage the second aspect. Several spatial relations have been described in literature, for instance topological, cardinal-directional relations, and distance relations. This report considers the topological ones, since they are the most widely offered by existing systems and represent the building blocks for the implementation of other spatial relations.

Although many abstract models have been studied in literature [3, 4, 6] for defining the semantics of topological relationships between geometric objects embedded in an Euclidean space, the problems arising when topological relationships are evaluated on real data have been much less explored. In particular, topological relations have been defined by using the 9-intersection matrix approach [6] or other axiomatic approaches [13], while for their evaluation, specific computational geometry algorithms have been implemented in real systems which work on real data represented as vectors in a discrete space.

A consequence of this fact is that the evaluation of topological relations can be non robust, i.e. it can produce different results on the same data in different contexts. The existence of robustness problems in the execution of computational geometry algorithms which use finite numbers (e.g. floating point) for the representation of coordinates in an Euclidean space, instead of the real numbers theoretically required, is well known [2, 10].

This report considers the distributed and heterogeneous context of a Spatial Data Infrastructure (SDI) in which the problems related to the adopted finite number representation are made even worse by the data perturbation occurring during the exchange between different systems. Such exchanges can introduce perturbations in geometric representations as a result of the conversions between different formats and precisions.

In [1] a set of rules is proposed which can be applied to vector datasets in order to increase their robustness w.r.t. topological relation evaluation. More specifically, the authors consider an implementation model in which equality between two geometric primitives requires that they are bitwise identical. This model is called here *identity model*, in contrast with another kind of

model in which equality is evaluated using a tolerance value, called here *tolerance model*.

This report considers a distributed environment in which both implementation models can be adopted and proposes a set of rules for guaranteeing the robustness in both models; finally, it discusses the applicability of available algorithms of the Snap Rounding family in order to preserve robustness in case of perturbations.

The remainder of the report is organized as follows: Sec. 2 summarizes some related work, while Sec. 3 formalizes the problem, and in particular introduces the concepts of non-ambiguous dataset and robust dataset. Sec. 4 defines a set of rules that make a dataset non-ambiguous, while Sec. 5 presents a set of rules for guaranteeing the robustness of a dataset. Sec. 6 discusses the applicability of existing algorithms for establishing and restoring the robustness of a non-ambiguous dataset after a perturbation occurred during a system transfer. Finally, Sec. 7 proposes some experimental results that exemplify the effects of perturbations on the dataset robustness.

2 Related Work

Geometric algorithms typically assume an infinite precision in coordinate representation. This assumption does not fit well with their implementation and raises great difficulties in ensuring robustness. In recent years several techniques have been proposed in order to overcome these issues. For instance, the Exact Geometric Computation model [2] provides a method for making the evaluation of geometric algorithms robust. This can be achieved either by computing every numeric value exactly, or by using some symbolic or implicit numeric representation that allows predicate values to be computed exactly. Exact computation is theoretically possible whenever all numeric values are algebraic, which is the case for most current problems in computational geometry. Another solution requires the application of rounding algorithms that convert an arbitrary-precision arrangement of segments into a fixed-precision representation, such as the Snap Rounding algorithm [10] and its iterative version [9].

In the geographical field, several robustness rules have been proposed in order to solve the mentioned robustness problems, and they are to some extent applied by real systems. The most important one is based on the identification of common geometric primitives between different objects. These common primitives can be either stored once and referred to by the objects (topological structures [5]) or repeated identically in all objects which share them. A GIS topology is a set of rules that models how points, lines and

polygons share coincident geometries, for instance imposing that adjacent features will have a portion of common boundary. A topological data model manages spatial relationships by representing spatial objects as an underlying graph of topological primitives: nodes, faces and edges. A complementary robustness rule, which has been suggested, for instance in [14], consists in ensuring that a minimum distance is kept between all geometric primitives which are not identical.

The identification of coincident geometries can be performed in two distinct ways: requiring the bitwise equality between coordinates (identity model) or considering a tolerance value during the tests (tolerance model). Some available GIS tools, such as PostGIS [11] and JTS Topology Suite [15], uses the first model for implementing topological relations, while other ones, such as ESRI ArcGIS [8], applies the second one for topology construction. In particular, the term cluster tolerance is used to identify the distance range below which all vertices are considered identical or a vertex is considered to belong to a segment. Notice that in ArcGIS the clustering step implies the replacement of coincident vertices with a single representative point, determined considering the position of the original vertices and an assigned weight [7]. Conversely, the tolerance model considered in this report does not include a replacement of original vertices, but only the definition of equality clusters.

3 Problem Formalization

The analysis performed in [1] about the robustness of topological relations starts by considering the robustness of a set of vector predicates that are used in the implementation of topological relations. In particular, the robustness of topological relations is directly derived from the robustness of these predicates, which are called critical since their evaluation can produce different results in different systems.

For the purposes of this paper, three critical vector predicates are of particular interest because their implementation is different in a identity model with respect to a tolerance model:

- *boolean equal*(v_1, v_2): it tests the equality between two vertices v_1 and v_2.
- *boolean belongsTo*(v, s): it tests if vertex v belongs to the segment s, represented in vector format.
- *boolean leftOf*(v, s) (or *rightOf*(v, s)): it returns true if the vertex is contained in the half-plane induced by s on its left (or on its right).

In order to discuss the robustness of these predicates in different implementation models, the report first distinguishes two sources of problems that can affect this evaluation: the first one regards the algorithm implementation; while the second one regards the perturbation in vector data representation due to data exchange or other operations.

Definition 3.1 (Numerical weakness)**.** The *numerical weakness* of a set of algorithm implementations on different machines is the largest distance between two vertices or a vertex and a segment such that the evaluation of the basic predicates can produce different results. □

Techniques like the Exact Geometric Computation model [2], or the Snap Rounding algorithm [10] and its iterative version [9] aim to reduce or eliminate numerical weakness in algorithm implementation. Therefore, in a given context it is possible to assume that the numerical weakness is less than a given value nw.

The following definitions are useful in order to precisely define the concept of robustness that will be discussed in the sequel.

Definition 3.2 (Topological interpretation)**.** The *topological interpretation* of a geometric dataset is the evaluation of all possible topological relationships between the geometries of the dataset. □

Definition 3.3 (Topologically non-ambiguous dataset)**.** A dataset DS is topologically *non-ambiguous* if and only if different algorithm implementations on different machines always produce the same topological interpretation on DS. □

Definition 3.4 (p-perturbation)**.** Given a number $p \in \mathbb{R}$, a *p-perturbation* of a dataset DS is a copy of DS where each coordinate of its geometries is arbitrarily modified by an amount $\varepsilon < p$. □

Definition 3.5 (p-robustness)**.** A dataset DS is *p-robust* if and only if the same topological interpretation is produced by different algorithm implementations on any p-perturbation of DS. □

Observation 3.1. Given a p-robust dataset DS, a generic perturbation performed on it produces a situation in which: (1) the dataset has maintained the same topological interpretation, but (2) the dataset is no longer p-robust. □

Given the above definitions, the aim of the paper is threefold: (1) define a set of conditions for making a dataset DS topologically non-ambiguous in a context characterized by a given numerical weakness, (2) define conditions for making a dataset p-robust, and (3) define the properties of an algorithm

restore_p-robustness for restoring the robustness of the dataset after a perturbation, and of an algorithm *establish_p-robustness* for establishing the robustness of a non-ambiguous dataset.

The need for the first algorithm comes from Observation 3.1: if the dataset robustness is not restored after a perturbation, subsequent perturbations may lead to a loss of the topological content, thus neither the topology nor the robustness can be recovered. A solution to this problem exists, since the dataset was originally robust.

Conversely, relatively to the second algorithm, it is not always possible to establish the robustness for a non-ambiguous dataset without modifying its topological interpretation. Finally, these issues will be analyzed considering two approaches for computing the topological interpretation of a dataset, as described in the following section.

4 Rules for Non Ambiguous Datasets

As mentioned in the previous sections, there are two fundamental approaches to the topological interpretation of a geometric dataset: the *identity model* (IM) and the *tolerance model* (TM). The two approaches differ in the way basic predicates between two geometric primitives are defined. The identity model applies the following predicates definitions:

- Given two vertices a and b, $equal(a, b)$ is true if and only if their coordinates are (bitwise) identical.
- Given a vertex v and a segment s, $belongsTo(v, s)$ is always false, unless v is an endpoint of s. In order to obtain that v is located onto s it is required that s is split in two segments s_1 and s_2, so that v is equal to the end point of s_1 and to the start point of s_2.
- Given a vertex v and a segment s, $leftOf(v, s)$ (or $rightOf(v, s)$) is true if v lies in the left half-plane induced by s (or in the right half-plane induced by s).

Conversely, the tolerance model adopts the following definitions:

- A tolerance value t is established.
- Given two vertices a and b, $equal(a, b)$ is true, if and only if the distance between a and b is less than t.
- The transitive property of the equal predicate is preserved, i.e. given three vertices a, b and c: $equal(a, b) \wedge equal(b, c) = equal(a, c)$

- Given a vertex v and a segment s, $belongsTo(v, s)$ is true if and only if the distance between v and s is less than t.
- Given a vertex v and a segment s, $leftOf(v, s)$ (or $rightOf(v, s)$) is true if and only if v lies in the left half-plane induced by s (or in the right half-plane induced by s) and the distance between v and s is greater than t.

Notice that the tolerance model adds a new basic critical predicate:

$$distance(v_1, v_2) < t \text{ (or } distance(v, s) < t)$$

The two models require different rules for making a dataset non-ambiguous. In particular, let us assume a context characterized by a numerical weakness nw and consider the three predicates: $equal(a, b)$, $belongsTo(v, s)$, and $leftOf(p, s)$ ($rightOf(v, s)$).

In the identity model, the first two predicates, $equal(a, b)$ and $belongsTo(v, s)$ are never ambiguous. The only possible ambiguity refers to the $leftOf(p, s)$ ($rightOf(p, s)$) predicate. In order to make this predicate non-ambiguous only the following rule is needed:

R1. $\forall v \in Vertices \ \forall s \in Segments$
$(\neg belongsTo(v, s) \implies distance(v, s) > nw)$

The tolerance model requires more rules, since its semantics is based on the distance function; the required rules are:

R2. $\forall v_1, v_2 \in Vertices(equal(v_1, v_2) \implies distance(v_1, v_2) < t - nw)$

R3. $\forall v_1, v_2 \in Vertices(\neg equal(v_1, v_2) \implies distance(v_1, v_2) > t + nw)$

R4. $\forall v \in Vertices \ \forall s \in Segments$
$(belongsTo(v, s) \implies distance(v, s) < t - nw)$

R5. $\forall v \in Vertices \ \forall s \in Segments$
$(\neg belongsTo(v, s) \implies distance(v, s) > t + nw)$

Proposition 4.1. Given a dataset DS and a context characterized by a numerical weakness nw, if DS satisfies rule R1, then it is non-ambiguous in the identity model (IM).

Proof. In IM the $equal(v_1, v_2)$ predicate is non-ambiguous by definition. The same is true for the $belongsTo(v, s)$ predicate, indeed if it is true then v is equal to the start or end point of s, thus it is non-ambiguous. Regarding the $leftOf(v, s)$ ($rightOf(v, s)$) predicate, since DS satisfies R1, then the minimum

distance between v and s is greater than nw (numerical weakness of the considered context) and this guarantees that the predicate does not change, i.e. it is non-ambiguous. □

Proposition 4.2. Given a dataset *DS* and a context characterized by a numerical weakness nw, if *DS* satisfies rules R2, R3, R4 and R5, then it is non-ambiguous in the tolerance model (TM).

Proof. Since *DS* satisfies rule R2, if two vertices v_1, v_2 are equal, than it holds that $distance(v_1, v_2) < t - nw$, and this guarantees that in a context of numerical weakness nw, $distance(v_1, v_2) < t$ is non-ambiguous. The satisfaction of rule R3 by *DS* allows us to apply a similar reasoning to the case in which v_1, v_2 are not equal, thus proving that the $equal(v_1, v_2)$ predicate is non-ambiguous. Moreover, since DS satisfies rule R4, if a vertex v belongs to a segment s, then it is true that $distance(v, s) < t - nw$, and this guarantees that in a context of numerical weakness nw, $distance(v, s) < t$ is non-ambiguous. In the same way, by exploiting the satisfaction of R5, it can be proved, for the case in which v does not belong to s, that $distance(v, s) > t$ is non-ambiguous, thus concluding that the $belongsTo(v, s)$ predicate is non-ambiguous. Finally, rule R5 also guarantees that the $leftOf(v, s)$ ($rightOf(v, s)$) predicate is non-ambiguous. □

5 Rules for Robust Datasets

The perturbations considered in this paper can be arbitrarily applied to each primitive coordinate. In particular, two kinds of perturbations can be distinguished: *preservative perturbations*, and *non-preservative perturbations*.

Definition 5.1 (IM preservative perturbation)**.** A perturbation performed in the context of an identity model is said to be preservative if equal changes are performed on all equal coordinates. □

In other words, this kind of perturbation preserves the identity among vertices: namely, after a preservative perturbation the datasets is still represented in an identity model.

Definition 5.2 (TM preservative perturbation)**.** A perturbation performed in the context of a tolerance model is said to be preservative if it ensures that the equality classes induced by clustering are preserved, namely vertices that are equal before a perturbation remains equal also after the perturbation. □

A perturbation is said to be non-preservative if the changes applied to equal vertices can produce vertices that are no longer equal. In the following only preservative perturbations are considered.

In order to make a dataset p-robust, with respect to preservative p-perturbations, two rules have to be defined: one for the equality robustness, and one for the disjointness robustness. This section defines such rules for both the identity and the tolerance model.

Rule 5.1 (Identity model equality rule (IME))**.** The equality rule implemented by the identity model requires that the coordinates of equal primitives (i.e., points and segments) are bitwise identical (identity of coordinates). □

Rule 5.2 (Identity model disjointness rule (IMD))**.** The disjunction rule implemented by the identity model is based on the concept of minimum distance (min_d). In other words, the minimum distance min_d between two points, or between a point and a segment, has to be greater than $2p$: $min_d > 2p$. □

The coefficient 2 is needed because two points, or a point and a segment, can move close to each other in opposite directions.

Proposition 5.1. Given a dataset DS that satisfies IME and IMD rules, then DS is p-robust in the identity model.

Proof. The proof is similar to the one presented for Proposition 4.1, where rule IME preserves the $equal(v_1, v_2)$ predicate and the $belongsTo\ (v, s)$ predicate when they are true, while rule IMD preserves the same predicates when they are false and guarantees the p-robustness of the $leftOf(v, s)$ ($rightOf(v, s)$) predicate. □

Relatively to the tolerance model, the following rules can be defined where t is the tolerance, namely the distance below which two points are considered the same.

Rule 5.3 (Tolerance model equality rule (TME))**.** The equality rule implemented by the tolerance model requires that the maximum distance (max_d) between two equal primitives (two vertices or a vertex and a segment end point) is less than $(t - 2p)$: $max_d < t - 2p$. □

Rule 5.4 (Tolerance model disjointness rule (TMD))**.** The disjointness rule implemented by the tolerance model requires that the minimum distance (min_d) between two points, or between a point and a segment, has to be greater than $t + 2p$: $min_d > t + 2p$. □

As for the previous two rules, the coefficient 2 is needed because two points, or a point and a segment, can move close to each other in opposite directions. Notice that a necessary condition to satisfy TME and TMD is that $t > 2p$. Thus, the tolerance model can guarantee p-robustness only for values of p that satisfy the above condition.

Proposition 5.2. Given a dataset DS that satisfies TME and TMD rules, then it is p-robust in the tolerance model.

Proof. The proof is similar to the one presented for Proposition 4.2, where the rule TME preserves the $equal(v_1, v_2)$ predicate and the $belongsTo(v, s)$ predicate when they are true, while TMD rule preserves the same predicates when they are false and guarantees the p-robustness of the $leftOf(v, s)$ ($rightOf(v, s)$) predicate. □

6 Algorithm for Robust Dataset Management

This section analyses the applicability of existing algorithm for establishing or restoring robustness in the identity and tolerance model, respectively. First, the following assumption is introduced.

Assumption 6.1. This paper assumes that the input dataset is non-ambiguous; therefore, the topological relations to be made robust are the ones that can be derived from the available geometries. □

Notice that Assumption 6.1 implies the following conditions:

- In the identity model, datasets have to be produced by cloning geometric primitives that have to be shared by different geometries.
- In the tolerance model, datasets must be produced by applying a clustering algorithm in order to ensure that the transitivity of the equality relation is satisfied. In other words, groups of vertices have to be identified among which the equality relation can be transitively applied.

Notice that in literature the only available solution for guaranteeing robustness of a set of segments is to apply an algorithm of the Snap Rounding (SR) family. In particular, when also a minimum distance among non intersecting (or touching) segments has to be guaranteed, the Iterated Snap Rounding [9] or the Iterated Snap Rounding with Bounded Drift [12] has to be adopted. This algorithm proceeds as follows:

1. It computes the segment intersections and splits every pair of intersecting segments in four non interior-intersecting segments.

2. Given a grid of pixels covering the reference space in which segments are embedded, it snaps the segment end-points to the center of the pixels. Pixels containing segments end-points are called hot pixels.

3. It splits and snaps to the pixel center also the segments that cross a hot pixel.

4. After this first iteration, other segments can cross some hot pixels, thus the previous step is iterated until no segments cross any hot pixels.

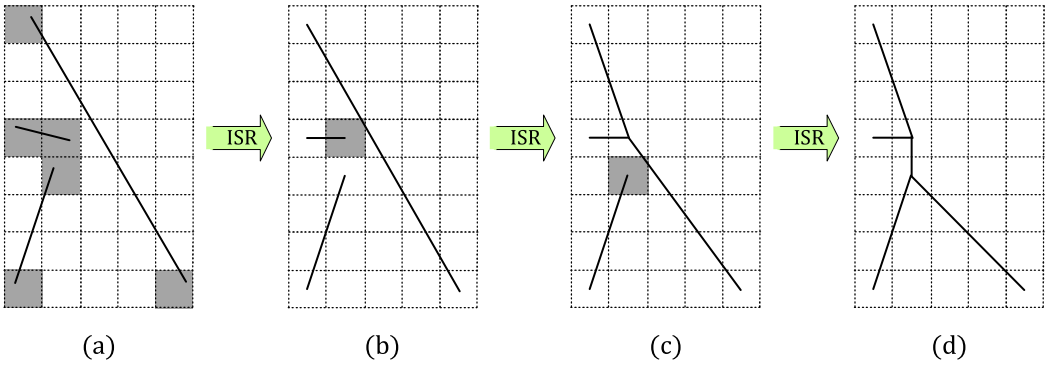

Figure 1: Example of execution of the Iterative Snap Rounding algorithm on an initial arrangement of three segments.

Fig. 1 illustrates an example of execution of the ISR algorithm on an initial arrangement of three segments. Fig. 1(a) shows the initial segment arrangement , while Fig. 1(b) shows the arrangement after steps 1 and 2; and finally, Fig. 1(c) and Fig. 1(d) show two iterations of the step 3. However, in particular for the restoring robustness issue, the application of the ISR approach to the entire dataset has many drawbacks. The following two subsections discuss these drawbacks and present some possible ideas for restoring p-robustness on IM-based and TM-based datasets.

6.1 Algorithms for IM-based Datasets

Since by Assumption 6.1 the original dataset is non-ambiguous, establishing robustness means that the IMD rule has to be satisfied by the whole dataset. In IM-based datasets this operation can be performed by applying the ISR algorithm, but this choice has the following drawbacks:

- After an ISR application, vertices that were initially different might have to become equal, since they are snapped to the same pixel x of the

grid, when they both fall in x. However, the dataset is assumed to be non-ambiguous, so distinct vertices should remain distinct, otherwise the topology interpretation is changed.

- ISR ensures a minimum distance between vertices and segments: after its application, each vertex is at least half a unit away from any non-incident segment. In order to ensure a minimum distance of $2p$, the grid unit has to be set to $4p$, thus generating an approximation that is higher than necessary.

Conversely, as regards to the restoring robustness problem, the application of ISR becomes even more problematic. Indeed, given Assumption 6.1 and starting from a robust condition, the current state is not only non-ambiguous, but there should be less local configurations that violate the IMD rule. Therefore, a different approach can be followed which consists of two steps:

- Check the dataset to identify the local configurations that violate the IMD rule (by performing a minimum distance checking).
- Apply some local adjustments (which can be automatic or manual) at each configuration that violates the rule.

The minimum distance checking can be performed by loading the dataset in a spatial database and executing a set of SQL queries. In order to perform some experiments, this test has been implemented in a PostGIS database: starting from a collection of datasets $D_1, \ldots, D_n$, they have been loaded in n tables $T_1, \ldots, T_n$ each with a spatial attribute AG and an identifier ID, and given them the following set of SQL queries have been executed. In particular, initially the geometries of type *point/multipoint* are loaded in a temporary table called `PNTS`, while the geometries *linestring/multilinestring* are loaded in a temporary table called `SEGS` (the same will be done for *polygon/multipolygon*, by considering the *linestrings* composing their boundary). Subsequently, three queries are executed to identify the violation of the minimum distance rule: (i) among points in `PNTS` (*Q1*), (ii) between points in `PNTS` and segments in `SEGS` (*Q2*), and (iii) among segments in `SEGS` (*Q3*), respectively. According to rule IMD, the minimum distance (MD) is set to $2p$ for IM-based datasets, where p is the maximum admitted perturbation.

Listings 6.1-6.5 shows the queries that can be applied for loading the dataset and performing the checks. The performed experiments on a dataset containing 56,833 points and 176,253 segments show that using envelope and `GIST` indices the time required by the queries is: 3960 ms (11030 ms) for

loading PNTS (+ index creation), 12470 ms (20540 ms) for loading SEGS (+ index creation), 2810 ms, 4410 ms, and 47650 for testing *Q1*, *Q2* and *Q3*, respectively.

Listing 6.1 Query that loads the geometries of type point or multipoint into a temporary table called PTNS.

```
-- LOADING POINTS
INSERT INTO PNTS(id, table, field, vertex)
(SELECT id,Ti,AG,(ST_Dump(AG)).geom
  FROM Ti
  WHERE AG IS NOT NULL AND
        NOT ST_IsEmpty(AG));
UPDATE PNTS SET envelope = PostGIS_AddBBox(ST_Expand(vertex, MD));
CREATE INDEX PNTidx1 ON PNTS USING gist(envelope);
```

Listing 6.2 Query $Q1$ that extracts the violations of the minimum distance rule between points in PTNS.

```
-- QUERY EXTRACTING THE VIOLATION OF MD AMONG POINTS
SELECT p1.id as ID1, p1.table as T1,
       p1.field as F1,
       p2.id as ID2, p2.table as T2,
       p2.field as F2,
       ST_MakeLine(p1.vertex,p2.vertex)
       as connectingLine,
       ST_Distance(p1.vertex,p2.vertex)
       as distance
FROM PNTS p1 JOIN PNTS p2
     ON p1.envelope && p2.envelope AND
        p1.id < p2.id
WHERE ST_Distance(p1.vertex,p2.vertex)>0
  AND ST_Distance(p1.vertex,p2.vertex)<=MD
```

Notice that in Listing 6.5 an auxiliary table has to be loaded with the segments end-points. This is necessary in order to deal with intersecting segments (distance = 0) where the crossing end-point is under the minimum distance from the intersected segment. Finally, also the segment length should be tested, since only segments with a length greater than

Listing 6.3 Query that loads the geometries of type linestring or multiline into a temporary table called SEGS.

```
-- LOADING SEGMENTS
INSERT INTO SEGS(id, table, field, segment)
 (SELECT T.id,Ti,AG,ST_MakeLine(T.s,T.e)
  FROM
   (SELECT L.id,
    ST_PointN(L.geom, generate_series(1,
              ST_NPoints(L.geom)-1)) as s,
    ST_PointN(L.geom, generate_series(2,
              ST_NPoints(L.geom))) as e
    FROM (SELECT id,(ST_Dump(AG)).geom
          FROM Ti
          WHERE AG IS NOT NULL AND
                NOT ST_IsEmpty(AG)) AS L
   ) as T);
CREATE INDEX SEGidx1 ON PNTS
USING gist(segment);
```

Listing 6.4 Query $Q2$ that extracts the violations of the minimum distance rule between points in PTNS and segments in SEGS.

```
-- QUERY EXTRACTING THE VIOLATION OF MD
-- BETWEEN SEGMENTS AND POINTS
SELECT p.id as IDP, s.id as IDL,
       p.table as TP, p.field as FP,
       s.table as TL, s.field as FL,
       p.vertex as point,
       s.segment as segment,
       ST_Distance(s.segment,p.vertex)
       as distance
FROM PNTS p JOIN SEGS s
     ON p.envelope && s.segment
WHERE ST_Dwithin(p.vertex, s.segment, MD)
      AND NOT ST_Touches(p.vertex,s.segment)
```

Listing 6.5 Query $Q3$ that extracts the violations of the minimum distance rule among segments in SEGS.

```
-- LOADING VERTICES
INSERT INTO VRTS(id, table, field, vertex)
 (SELECT S.id,S.table,S.field,
         ST_StartPoint(S.segment)
  FROM SEGS UNION
  SELECT S.id,S.table,S.field,
         ST_EndPoint(S.segment)
  FROM SEGS)
UPDATE VRTS SET envelope = PostGIS_AddBBox(ST_Expand(vertex, MD)));
CREATE INDEX VRTidx1 ON VRTS
       USING gist(envelope);

-- QUERY EXTRACTING THE VIOLATION OF MD AMONG SEGMENTS
SELECT s.id as IDS, v.id as IDV,
       s.table as TS, s.field as FS,
       sv.table as TV, v.field as FV,
       s.segment as seg, v.vertex as ver,
       ST_Distance(s.segment,v.vertex)
       as distance
FROM SEGS s JOIN VRTS v ON v.id < s.id
     AND  v.envelope && s.segment
WHERE ST_Dwithin(s.segment, v.vertex, MD)
  AND NOT ST_Touches(s.segment,v.vertex)
```

the minimum distance have to be accepted, in order to avoid that a relation $In(vertex, linestring)$ changes to a $Touch(vertex, linestring)$. Listing 6.6 shows this additional query (time required 280 ms).

Regarding the local adjustments that can be automatically executed for restoring the minimum distance, a possible solution to be explored is to apply to each pair of geometries violating the rules a spreading operation. Since different adjustments could be necessary for different pairs of geometries, each adjustment can be represented by a vector and their integration can be performed by applying a vector combination, such as the vector sum.

Listing 6.6 Query that extracts the segments in `SEGS` that violate the minimum distance rule, since their length is less than MD.

```
SELECT s.id, s.table, s.field, s.segment,
       ST_Length(s.segment) as length
FROM SEGS s
WHERE ST_Length(s.segment) < MD
```

6.2 Algorithms for TM-based Datasets

Establishing robustness in a TM context means that TME and TMD rules have to be satisfied by the whole dataset. In TM-based datasets this operation cannot be performed by applying the ISR algorithm, since this approach aims to snap close vertices, while the required operation in this case is sometimes a spreading and sometimes a rapprochement. Algorithms for point clustering could be useful [7], but they should have to be extended in order to apply points movements for preserving the transitivity property of equality. This means that, given a point, in its neighborhood of size t there could be only points belonging to its cluster.

Since a global approach is not available, we suggest to apply for both operations, establish and restoring robustness, the same idea, i.e.: first the critical configurations are identified using SQL queries and then a local adjustment is applied.

As regards to the queries for detecting critical configurations, starting from the tables `PNTS` and `SEGS` presented for IM-based datasets, queries $Q1$, $Q2$ and $Q3$ can be used also for evaluating the TME and TMD rules, provided that the test in the `WHERE` clause regarding the distance is changed as follows:

- for TME the critical distance interval is: $[(t - 2p) \dots t]$
- for TMD the critical distance interval is: $[t \dots (t + 2p)]$

Therefore, for detecting the violations of TME and TMD among points the `WHERE` clause of the query $Q1$ has to be modified as follows:

- $Q1$ for TME:
  ```
  ST_Distance(p1.vertex,p2.vertex) > t-2p AND
  ST_Distance(p1.vertex,p2.vertex) <= t
  ```
- $Q1$ for TMD:
  ```
  ST_Distance(p1.vertex,p2.vertex) > t AND
  ST_Distance(p1.vertex,p2.vertex) <= t+2p
  ```

Similarly, since in $Q2$ and $Q3$ the `ST_Dwithin()` function has been used, the `WHERE` clause has to be modified as follows:

- $Q2/Q3$ for TME:
  `ST_Dwithin(p.vertex,s.segment,t) AND`
  `NOT ST_Dwithin(p.vertex,s.segment,t-2p)`
- $Q2/Q3$ for TMD:
  `ST_Dwithin(p.vertex,s.segment,t+2p) AND`
  `NOT ST_Dwithin(p.vertex,s.segment,t)`

Finally, the query in Listing 6.6, regarding segments length, has to converted in a test of TME and TMD rules among points and linestring endpoints. Therefore, a $Q1$ query for TME and a $Q1$ query for TMD have to be executed considering tables `PNTS` and `BNDS`, where `BNDS` table is loaded as shown in Listing 6.7.

Listing 6.7 Query that loads the end points of the linestring.

```
-- LOADING LINESTRING END POINTS
INSERT INTO BNDS
(id, table, field, vertex)
(SELECT L.id::varchar||-S,Ti,AG,
        ST_StartPoint(L.geom)
 FROM (SELECT id,(ST_Dump(AG)).geom
         FROM Ti
         WHERE AG IS NOT NULL AND
               NOT ST_IsEmpty(AG)) AS L
 UNION
 SELECT L.id::varchar||-E,Ti,AG,
        ST_EndPoint(L.geom)
 FROM (SELECT id,(ST_Dump(AG)).geom
         FROM Ti
         WHERE AG IS NOT NULL AND
               NOT ST_IsEmpty(AG)) AS L
UPDATE BNDS SET envelope =
                PostGIS_AddBBox(ST_Expand(vertex, MD)));
CREATE INDEX BNDidx1 ON VRTS
       USING gist(envelope);
```

Regarding the local adjustments that can be automatically applied in order to restore the minimum distance of $(t+2p)$ (or the maximum distance of

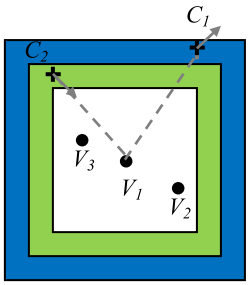

Figure 2: Spreading and nearing operations performed for restoring a cluster robustness.

($t2p$)), a possible solution to be explored is to apply to each pair of geometries violating the rules a spreading (or a nearing) operation for restoring the rule satisfaction. This require to compute the clusters of equal points (which are disjoint from each other) based on a tolerance t and the critical points for each cluster (notice that in many cases in each cluster there will be only one point). Critical points are those identified by queries $Q1$, $Q2$ and $Q3$. The spreading (or nearing operation) can be represented by a vector applied onto the straight line connecting the critical point with the center of gravity of the cluster, thus producing a spreading or nearing movement of the critical point w.r.t. the cluster.

Fig. 2 illustrates the process. In this case the cluster is composed of two points v_1, v_2 and v_3, while the critical points are c_1 and c_2. The arrows show the suggested adjustments.

7 Robustness Test

In order to test the effectiveness of the proposed robustness rules for IM-based and TM-based datasets, some experiments on real datasets have been performed. The road links (RL with 3851 linestrings) and road nodes (RN with 2856 points) of an urban area of Northern Italy have been considered and their p-robustness rules have been tested as follows. These datasets have been considered since we know they are not robust, in particular with respect to topological relations evaluation in the identity model.

First the test for IM-based datasets has been performed through the following steps:

- The topological relations existing among each pair ($link, node$) have been computed.

- A sequence of p-perturbations have been simulated by rounding the geometries using a grid of cell size of $10^{-6}, \ldots, 10^{-1}$ meters.
- The topological relation changes have been detected in order to evaluate the effective robustness of datasets.
- Finally, the satisfaction of the robustness rule IMD with respect to the sequence of perturbations has been tested by applying queries of Listing 6.1-6.5.

The results are shown in Table 1.

Table 1: Perturbation simulation on RL and RN datasets considering the IM approach

Perturbation	#relation changes	#IMD violations
10^{-6}	16	56
10^{-5}	49	115
10^{-4}	76	695
10^{-3}	2257	2340
10^{-2}	2091	2344
10^{-1}	2286	2369

Notice that, the number of IMD violations is always higher with respect to the number of relation changes; this means that:

- The rule works correctly w.r.t. the goal to detect possible configurations that can be sources of robustness violations.
- Not all non-robust configurations give raise to a relation change. Moreover, this experiment shows that the considered datasets are highly non-robust when perturbations size is higher than 10^{-4} meter.

Moreover, this experiment shows that the considered datasets are highly non-robust when perturbations size is higher than 10^{-4} meter.

The test for TM-based datasets has been applied on the same datasets considered in the previous experiment and with the same steps described above, but with a tolerance of one order of magnitude greater than the perturbation, obtaining the results shown in Table 2. In the table the number of violations of both TMI and TMD rules are reported, considering both the distance between points and segments and the distance between points and segment end-points.

Notice that, again the number of TMI and TMD violations is always higher w.r.t. relation changes; this means that the rules work correctly w.r.t. the aim to detect possible source of robustness violation. Moreover, from the experiments we can see that these datasets have a higher level of robustness in the tolerance model w.r.t. the identity model; however, some robustness problems still exist.

Table 2: Perturbation simulation on RL and RN datasets considering the IM approach

Perturbation	#relation changes	#TMI violations pnt/seg (pnt/endpoints)	#TMD violation pnt/seg (pnt/endpoints)
10^{-6}	1	8 (3)	7(0)
10^{-5}	12	49 (16)	64 (26)
10^{-4}	0	0 (0)	0 (0)
10^{-3}	0	0 (0)	0 (0)
10^{-2}	0	0 (4)	1 (4)
10^{-1}	3	0 (8)	2 (0)

8 Conclusion

The execution of computational geometry algorithms, which use finite coordinate representations instead of the theoretically required real numbers, can induce many robustness problems. Such problems are of particular importance in a geographical distributed context, where topological relations can be evaluated in several systems producing different results.

This paper deals with the robustness of topological relations by considering a distributed context in which two kind of implementations can be applied: an identity model in which equality between primitive geometries requires a bitwise identity, and a tolerance model which considers the presence of a predefined tolerance value. Given such context, the paper proposes a set of rules for guaranteeing the robustness of topological relations and analyses the applicability of available algorithms of the Snap Rounding family in order to preserve robustness in case of perturbations. In particular, a set of SQL queries have been defined for determining the situations that do not satisfied the defined robustness rules and discusses the applicability of existing algorithm for locally solving such situations. These queries have been applied to a real dataset in order to evaluate the impact of robustness

problems in real situations and the applicability of the proposed robustness tests. The analysis highlights that existing algorithms cannot be successfully applied to establish or restore the robustness of a non-ambiguous dataset, essentially because they are based only on snap operations that inevitably modify the initial topology. Therefore, a future work will be the study of modified versions of the Snap Rounding based-algorithms, which considers also a spread operation that moves away two vertices or a vertex and a segment, preserving the original topological interpretation.

References

[1] Alberto Belussi, Sara Migliorini, Mauro Negri, and Giuseppe Pelagatti. Robustness of Spatial Relation Evaluation in Data Exchange. In *Proceedings of the 20th International Conference on Advances in Geographic Information Systems (SIGSPATIAL '12)*, pages 446–449, New York, NY, USA, 2012. ACM.

[2] L. Chen. *Exact Geometric Computation: Theory and Applications*. PhD thesis, New York University, Department of Computer Science, 2001.

[3] Eliseo Clementini and Paolino Di Felice. A Comparison of Methods for Representing Topological Relationships. *Inf. Sci. Appl.*, 3(3):149–178, 1995.

[4] Eliseo Clementini, Paolino Di Felice, and Peter van Oosterom. A Small Set of Formal Topological Relationships Suitable for End-User Interaction. In *Proceedings of the Third International Symposium on Advances in Spatial Databases*, pages 277–295. Springer-Verlag, 1993.

[5] M. J. Egenhofer, A. U. Frank, and J. P. Jackson. A topological data model for spatial databases. In *Proceedings of the 1st Symposium on Design and Implementation of Large Spatial Databases (SSD '90)*, pages 271–286, 1990.

[6] M. J. Egenhofer and R. Franzosa. Point-set topological spatial relations. *International Journal of Geographical Information Systems*, 5(2):161–174, 1991.

[7] ESRI. *Understanding Geometric Processing in ArcGIS*, 2010. An ESRI Technical Paper.

[8] ESRI. ArGIS, 2013. `http://www.esri.com/`.

[9] Dan Halperin and Eli Packer. Iterated snap rounding. *Comput. Geom. Theory Appl.*, 23(2):209–225, 2002.

[10] J. Hobby. Practical segment intersection with finite precision output. *Comp. Geometry Theory and App*, 13:Comp. Geometry Th. and App., 1999.

[11] OSGeo. *PostGIS 2.0 Manual*, 2013. `http://postgis.net/stuff/postgis-2.0.pdf`.

[12] Eli Packer. Iterated snap rounding with bounded drift. *Comput. Geom. Theory Appl.*, 40(3):231–251, August 2008.

[13] David A. Randell, Zhan Cui, and Anthony Cohn. A Spatial Logic Based on Regions and Connection. In *Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning (KR'92)*, pages 165–176. Morgan Kaufmann, 1992.

[14] Rodney James Thompson and Peter van Oosterom. Interchange of Spatial Data-Inhibiting Factors. In *Proceeding of the 9th AGILE International Conference on Geographic Information Science*, 2006.

[15] Vivid Solutions Inc. *JTS Topology Suite*, 2013. `http://www.vividsolutions.com/jts/bin/JTS%20Developer%20Guide.pdf`.

University of Verona
Department of Computer Science
Strada Le Grazie, 15
I-37134 Verona
Italy

http://www.di.univr.it