On topological elementary equivalence of spatial databases

Bart Kuijpers[1], Jan Paredaens[1] and Jan Van den Bussche[2]

[1] University of Antwerp (UIA), Dept. Math. & Computer Sci.,
Universiteitsplein 1, B-2610 Antwerp, Belgium
Email: {kuijpers, pareda}@uia.ua.ac.be
[2] University of Limburg (LUC), Dept. WNI
B-3590 Diepenbeek, Belgium
Email: vdbuss@luc.ac.be

Abstract. We consider spatial databases and queries definable using first-order logic and real polynomial inequalities. We are interested in topological queries: queries whose result only depends on the topological aspects of the spatial data. Two spatial databases are called topologically elementary equivalent if they cannot be distinguished by such topological first-order queries. Our contribution is a natural and effective characterization of topological elementary equivalence of closed databases in the real plane. As far as topological elementary equivalence is concerned, it does not matter whether we use first-order logic with full polynomial inequalities, or first-order logic with simple order comparisons only.

1 Introduction and summary

Spatial database systems [5, 12, 1, 9] are concerned with the representation and manipulation of data that have a geometrical or topological interpretation. In this paper, we are interested in *planar* spatial databases; the conceptual view of such a database is that of a possibly infinite set of points in the real plane.

Of course one needs a spatial data model that describes exactly which kinds of sets of points can be handled by the system, and how they can be queried. A rather general spatial data model is the *semi-algebraic* one. A semi-algebraic set is defined by a finite union of systems of polynomial equations and inequalities. This semi-algebraic model is an instance of the framework of constraint databases [14] (see also [17]).

By viewing the spatial database, a set of points in the real plane, as a binary relation over the real numbers, we can use the relational calculus, augmented with polynomial comparisons, as a spatial query language. For example, to verify whether the database S is contained within a straight line, one would write

$$(\exists a)(\exists b)(\exists c)(\forall x)(\forall y)(S(x,y) \rightarrow ax+by+c=0), \qquad (*)$$

or to verify whether the database contains a disk as a subset, one would write

$$(\exists x_0)(\exists y_0)(\exists r \neq 0)(\forall x)(\forall y)((x-x_0)^2+(y-y_0)^2<r^2 \rightarrow S(x,y)). \qquad (\dagger)$$

Naturally, quantifiers range over the real numbers. Such *spatial calculus queries* can be effectively answered on semi-algebraic databases [14], but a discussion of how this is done is outside the scope of the present paper. In this paper, we focus on Boolean queries (returning true or false) only.

Some queries are concerned with the topological properties of the spatial data only. Query (†) above is an example of such a query; a database contains a disk if and only if its topological interior is non-empty, and this is a purely topological property of the database. In contrast, query ($*$) is not a topological query; from a purely topological point of view, a straight line is no different from a curved line. Technically, two databases are considered topologically the same if one can be mapped into the other by a topological transformation of the real plane (mathematically, a *homeomorphism* of $\mathbf{R}^2$). A query is then called *topological* if it has the same result on databases that are topologically the same.

There is not much understanding yet of the class of spatial calculus queries that are topological in the above sense. One of the natural questions that arise in this respect is that of understanding *topological elementary equivalence*. In mathematical logic, two structures are called elementary equivalent if they cannot be distinguished logically, i.e., if every first-order formula has the same truth value on them. We study the corresponding notion in the spatial context: two spatial databases A and B are called topologically elementary equivalent if for every topological spatial calculus query Q, $Q(A)$ is true if and only if $Q(B)$ is.

We have been able to find a natural characterization of topological elementary equivalence in the case of *closed* databases. A spatial database is called closed if it is a closed subset of the real plane in the ordinary topological sense. This simply means that there are no points lying infinitely close to the database without actually lying in the database. Spatial databases occurring in practical applications, such as geographic information systems, are closed almost by default.

In the course of proving our result, we will also deal with a more relaxed notion of topological query, interesting in itself, which considers two databases to be the same topologically if one can be mapped into another by an *isotopy* rather than a general homeomorphism. Isotopies are homeomorphisms that preserve orientation.

Our characterization is based on a known topological property of semi-algebraic sets [7], namely that locally around each point they are "conical". We partition the points in the database according to the types of their cones. Roughly, our characterization then says that two databases are topologically elementary equivalent if and only if the cardinalities of the equivalence classes of their partitions match. A corollary of our characterization is that topological elementary equivalence of spatial databases is decidable. Another corollary is that as far as topological elementary equivalence is concerned, it does not matter whether we use first-order logic with full polynomial inequalities, or first-order logic with simple order comparisons only.

Our proof of the characterization involves various techniques. For the if-direction we show that it is expressible in the spatial calculus that the cone around a point has some specific type. For the only-if direction we show that two databases with matching equivalence classes can be transformed into topologically one and the same database. The transformation rules used in this transformation are shown to produce topologically elementary equivalent databases. The proof of the latter uses a recent collapse theorem on the expressiveness of first-order logic with polynomial inequalities by Benedikt, Dong, Libkin and Wong [3], and involves reduction techniques inspired by those introduced by Grumbach and Su [11].

A variation of topological queries was studied extensively by Egenhofer and his collaborators under the term "topological spatial relationships" (e.g., [8]). Their work is concerned with the topological relationships that can exist between two given individual spatial objects. Papadimitriou, Suciu and Vianu [16] significantly extended this work to collections of named spatial objects. As explained above, our work is different in scope, being concerned with the intrinsic topological properties of single spatial databases. In fact, our work is more in the spirit of topological model theory (e.g., [13, 10, 18]).

This paper is organized as follows. Definitions are given in Section 2. The partition of a database according to the cone types of its points is described in Section 3. The main results are formulated in Section 4. The main proof is sketched in Section 5. Concluding remarks are presented in Section 6.

2 Preliminaries

In this section we give the basic definitions we will be using concerning spatial databases and topological queries.

Spatial databases. We denote the real numbers by $\mathbf{R}$, so $\mathbf{R}^2$ denotes the real plane. A semi-algebraic set in $\mathbf{R}^2$ is a set of points that can be defined as

$$\{(x,y) \in \mathbf{R}^2 \mid \varphi(x,y)\},$$

where $\varphi(x,y)$ is a formula built using the Boolean connectives $\wedge$, $\vee$, and $\neg$, from atoms of the form $p(x,y) > 0$, where $p(x,y)$ is a polynomial in the variables x and y with integer coefficients. Observe that $p = 0$ is equivalent to $\neg(p > 0) \wedge \neg(-p > 0)$, so equations can be used as well as inequalities.

A *closed semi-algebraic database in the real plane* (or simply *database*) is a semi-algebraic set in $\mathbf{R}^2$ that is closed in the ordinary topological sense. It is known [4] that these are precisely the finite unions of sets of points that can be defined as

$$\{(x,y) \in \mathbf{R}^2 \mid p_1(x,y) \geq 0 \wedge \ldots \wedge p_m(x,y) \geq 0\}.$$

In other words, we disallow the essential use of strict inequalities in the definition of a database.

Queries. A *spatial calculus formula* is a first-order logic formula built using the logical connectives and quantifiers from two kinds of atomic formulas: $S(x, y)$ and $p(x_1, \ldots, x_k) > 0$, where S is the binary relation name representing the spatial database and $p(x_1, \ldots, x_k)$ is a polynomial in the variables $x_1, \ldots, x_k$ with integer coefficients.

A *spatial calculus query* (or simply *query*) is a spatial calculus formula without free variables. A query Q can be evaluated on a database A, yielding true or false, by letting variables range over $\mathbf{R}$ and by interpreting $S(x, y)$ to mean that the point (x, y) is in A. That Q evaluates to true on A would be denoted in model-theoretic notation as $(\mathbf{R}, A) \models Q(S)$, but we will simply write $A \models Q$.

Homeomorphism-invariance and -equivalence. Two databases A and B are called *homeomorphic* if there is a homeomorphism h of $\mathbf{R}^2$ such that $h(A) = B$.[3] A query Q is called *invariant under homeomorphisms* (abbreviated as *$\mathcal{H}$-invariant*) if whenever databases A and B are homeomorphic, then $A \models Q$ if and only if $B \models Q$. Finally, two databases A and B are called *$\mathcal{H}$-equivalent* if for each $\mathcal{H}$-invariant query Q, $A \models Q$ if and only if $B \models Q$.

Of course, databases that are homeomorphic are also $\mathcal{H}$-equivalent, but the converse does not hold. For example, we will see later that if A consists of a single disk, and B consists of two separate disks, then A and B are $\mathcal{H}$-equivalent.

Isotopy-invariance and -equivalence. An *isotopy* is a homeomorphism that is isotopic to the identity.[4] Two databases A and B are called *isotopic* if there is an isotopy h such that $h(A) = B$. Intuitively, this means that A can be continuously deformed into B without leaving the plane. The prototypical example of a homeomorphism that is not an isotopy is a reflection. As a matter of fact, it is known [15] that every homeomorphism either is an isotopy, or is isotopic to a reflection. Hence, when two databases A and B are homeomorphic, either A is actually isotopic to B, or A is isotopic to the mirror image of B. For example, Figure 1 shows two databases that are mirror-images of each other but that are not isotopic. They can be thought of as a left hand and a right hand, where the arm and the thumb have thickness and the wrist and the other fingers have no thickness.

A query Q is called *invariant under isotopies* (abbreviated as *$\mathcal{I}$-invariant*) if whenever databases A and B are isotopic, then $A \models Q$ if and only if $B \models Q$. Finally, two databases A and B are called *$\mathcal{I}$-equivalent* if for each $\mathcal{I}$-invariant query Q, $A \models Q$ if and only if $B \models Q$.

---

[3] A homeomorphism of $\mathbf{R}^2$ is a continuous bijection from $\mathbf{R}^2$ to itself whose inverse is also continuous.

[4] Two homeomorphisms f and g are *isotopic* if there is a function $F : \mathbf{R}^2 \times [0, 1] \to \mathbf{R}^2$ such that

1. for each $t \in [0, 1]$, the function $F_t : \mathbf{R}^2 \to \mathbf{R}^2 : p \mapsto F(p, t)$ is a homeomorphism;
2. F_0 is f and F_1 is g; and
3. $F(p, t)$ is continuous in t.

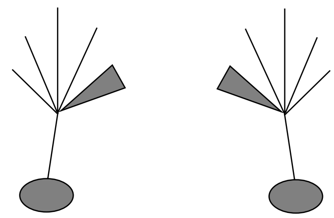


Fig. 1. Two homeomorphic, but not isotopic databases.

Clearly, $\mathcal{H}$-invariance implies $\mathcal{I}$-invariance and $\mathcal{I}$-equivalence implies $\mathcal{H}$-equivalence.

Examples. The query $(*)$ mentioned in the Introduction is a typical example of a query that is neither $\mathcal{I}$-invariant nor $\mathcal{H}$-invariant. The following queries are all $\mathcal{H}$-invariant (and hence also $\mathcal{I}$-invariant):

- The query $(\dagger)$ mentioned in the Introduction. In other words this query expresses that the database contains 2-dimensional parts.
- The query
$$(\exists r)(\forall x)(\forall y)(S(x,y) \rightarrow x^2 + y^2 \leq r^2)$$
expresses that the database is bounded.
- The query
$$(\exists x)(\exists y)\big(S(x,y) \wedge (\exists \varepsilon \neq 0) \\ (\forall x')(\forall y')((x-x')^2 + (y-y')^2 < \varepsilon^2 \wedge S(x',y')) \rightarrow (x' = x \wedge y' = y)\big)$$
expresses that the database contains isolated points.
- The query expressing that the database consists exclusively of lines that do not intersect. (We omit the formula; informally, this formula states that for each point p in the database all sufficiently small circles around p intersect the database either in one or in two points.)

Another natural topological property of databases one might want to test for, is topological connectivity; however, this is not expressible in the spatial calculus [3, 11] and therefore outside the scope of the discussion in the present paper.

3 Point-structures

In this section, we define the "point-structure" of a database. This definition is based on a known topological property of semi-algebraic sets [7], namely that locally around each point they are conical. This is illustrated in Figure 2. More precisely, for every non-isolated point p of a semi-algebraic set A there exists an $\varepsilon > 0$ such that $D(p, \varepsilon) \cap A$ is isotopic to the planar cone with top p and base

$C(p,\varepsilon) \cap A$.[5] More precisely, $D(p,\varepsilon) \cap A$ is isotopic to $\{\lambda \cdot (x,y) + (1-\lambda) \cdot (x',y') \mid (x',y') \in C(p,\varepsilon) \cap A \wedge 0 \leq \lambda \leq 1\}$, where $p = (x,y)$.

For every isolated point p of a database, there exists an $\varepsilon > 0$ such that $D(p,\varepsilon) \cap A = \{p\}$, so in this case we could say we have a cone with an empty base. We can thus refer in both cases to *the cone of p in A*; this cone is defined up to isotopy.

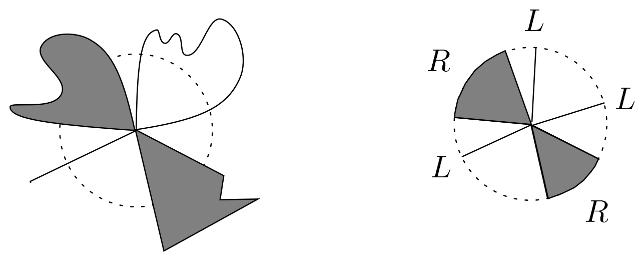


Fig. 2. A database and the cone of one of its points.

A database is also conical around the point at infinity.[6] More precisely, there exists an $\varepsilon > 0$ such that $\{(x,y) \mid x^2 + y^2 \geq \varepsilon^2\} \cap A$ is isotopic to $\{\lambda \cdot (x,y) \mid (x,y) \in C((0,0),\varepsilon) \cap A \wedge \lambda \geq 1\}$. We can indeed view the latter set as the cone with top ∞ and base $C((0,0),\varepsilon) \cap A$, and call it *the cone of ∞ in A* (again defined up to isotopy).

We use the following finite representation for cones. The cone having a full circle as its base is represented by the letter F (for "full"). Any other cone can be represented by a circular list of L's and R's (for "line" and "region" resp.) which describes the cone in a complete clockwise turn around the top. For example, the cone of Figure 2 is represented by $(LLRLR)$. The cone with empty base is represented by the empty circular list $()$.

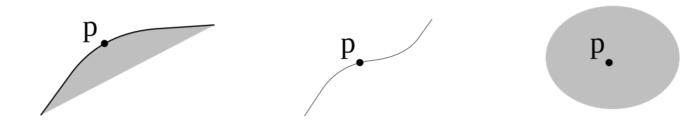
Let $\mathcal{C}$ be the set of cones represented in this way. We define:

Definition 1. Let A be a database. The *point-structure of A* is the function Π_A from $A \cup \{\infty\}$ to $\mathcal{C}$ that maps each point to its cone.

It can be shown that $\Pi_A^{-1}(c)$ is empty on all but a finite number of cones c. Moreover, as illustrated in Figure 3, there are only three cones c where $\Pi_A^{-1}(c)$ can be infinite: (R), (LL), and F. It can indeed be shown that in each database, the points with a cone different from these three are finite in number. We will refer to these points as the *singular* points of the database. Non-singular points are also called *regular* points.

[5] $D(p,\varepsilon)$ is the closed disk with center p and radius ε, $C(p,\varepsilon)$ is its bordering circle.

[6] If we project $\mathbf{R^2}$ stereographically onto a sphere, the point at infinity corresponds to the missing point on the sphere.



$\Pi_A(p) = (R)$ $\quad$ $\Pi_A(p) = (LL)$ $\quad$ $\Pi_A(p) = F$

Fig. 3. Regular points.

Definition 2. Let A and B be databases. We say that Π_A is *isomorphic* to Π_B (denoted by $\Pi_A \cong \Pi_B$) if there is a bijection f from $A \cup \{\infty\}$ to $B \cup \{\infty\}$ with $f(\infty) = \infty$, such that $\Pi_A = \Pi_B \circ f$.

4 The main results

Our main technical result is a characterization of $\mathcal{I}$-equivalence in terms of point-structure isomorphism:

Theorem 3. *Two databases A and B are $\mathcal{I}$-equivalent if and only if $\Pi_A \cong \Pi_B$.*

The proof will be sketched in the next section. A corollary of the theorem is a similar characterization of $\mathcal{H}$-equivalence:

Theorem 4. *Let σ be some fixed reflection of $\mathbf{R}^2$. Two databases A and B are $\mathcal{H}$-equivalent if and only if $\Pi_A \cong \Pi_B$ or $\Pi_A \cong \Pi_{\sigma(B)}$.*

Proof. Assuming Theorem 3, we have to prove that A and B are $\mathcal{H}$-equivalent if and only if A and B are $\mathcal{I}$-equivalent or A and $\sigma(B)$ are $\mathcal{I}$-equivalent. The if-implication follows from the fact, already mentioned in the Introduction, that every homeomorphism is either an isotopy or is isotopic to σ. For the only-if-implication, assume on the contrary that A and B are $\mathcal{H}$-equivalent and that there exist $\mathcal{I}$-invariant queries Q and Q' such that $A \models Q$, $B \not\models Q$, $A \models Q'$, and $\sigma(B) \not\models Q'$. Then the query $\bar{Q}$ defined by

$$\bar{Q}(S) = (Q(S) \vee Q'(\sigma(S))) \wedge (Q'(S) \vee Q(\sigma(S))),$$

is $\mathcal{H}$-invariant, and $A \models \bar{Q}$ but $B \not\models \bar{Q}$. This contradicts the assumption. $\square$

Examples. One disk and two separate disks have isomorphic point-structures: the points on the border have (R) as cone, and the points in the interior have F as cone. As a consequence of Theorem 3, they are $\mathcal{I}$-equivalent, hence also $\mathcal{H}$-equivalent.

Although all points on the unit circle and all points on the x-axis have the same cone (namely, (LL)), these databases do not have isomorphic point-structures. Indeed, in the former database the cone of ∞ is $()$ (in other words, this database is bounded), while in the latter database the cone of ∞ is (LL).

Figure 4 shows two databases that are not $\mathcal{I}$-equivalent. Indeed, the cone of the center point in the database on the left is $(LLLRLLRLR)$, while that on the right is $(LRLLRLLLR)$, which is a different circular list. The two database are of course $\mathcal{H}$-equivalent since they are mirror images of each other. We point out that it is possible that two mirror images are still $\mathcal{I}$-equivalent; for instance, the two databases shown in Figure 1 have isomorphic point-structures.

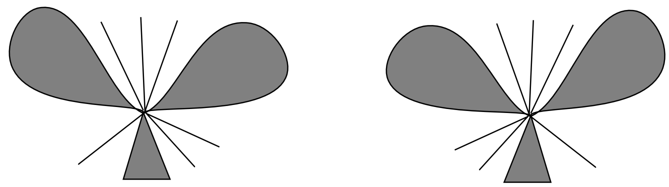


Fig. 4. Two databases that are $\mathcal{H}$-equivalent but not $\mathcal{I}$-equivalent.

Remark: In classical logic, if A and A' are elementary equivalent and B and B' are elementary equivalent, then the disjoint union of A and B and the disjoint union of A' and B' are also elementary equivalent. Theorem 3 implies that this is also true for $\mathcal{I}$-equivalence. However, it is not true for $\mathcal{H}$-equivalence; for instance, take A, A', and B to be the database on the left of Figure 4, and take B' to be the one on the right.

5 The proof

In this section we sketch the proof of Theorem 3.

Transformation rules. The crucial tool in our proof consists of the following three transformations rules that locally change databases:

Strip-cut: The *strip-cut* transformation, shown by the left-to-right arrow in Figure 5(a), locally cuts a strip in the database in two.

Strip-paste: The *strip-paste* transformation, shown by the right-to-left arrow in Figure 5(a), is the inverse of strip-cut.

Line-cut&paste: The *line-cut&paste* transformation, shown in Figure 5(b), locally cuts two parallel lines in the database and connects the corresponding loose ends. An isolated part D of the database may be present between the lines, which will come free after the line-cut&paste.

Note that the line-cut&paste transformation is its own inverse.

A fundamental property of the transformation rules is:

Proposition 5. *If a database B is obtained from a database A by a strip-cut, a strip-paste, or a line-cut&paste transformation, then A and B are $\mathcal{I}$-equivalent.*

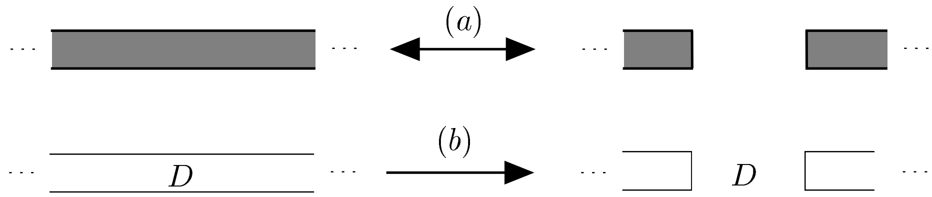


Fig. 5. The transformation rules: (a) strip-cut and strip-paste, (b) line-cut&paste.

Proposition 5 is proven in a number of steps:

1. First, a variation of the proposition is proven for weak versions of the three transformation rules (Lemma 6). These are illustrated in Figure 6. Line-cut&paste is split in a weak line-cut (arrow (b) from left to right) and a weak line-paste (arrow (b) from right to left).
2. The gap between the weak and the original rules is then closed via the notions of *2-regular* and *1-regular* database (Lemmas 8 and 9).

Due to space limitations, the proofs of the Lemmas have been omitted.

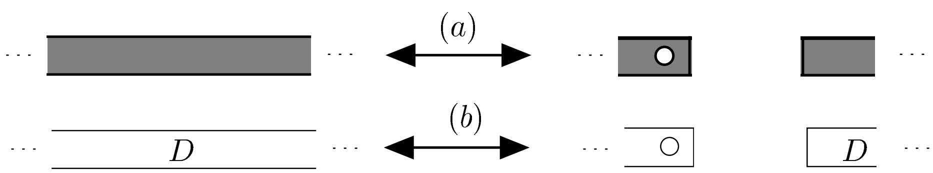


Fig. 6. Weak forms of the three transformation rules.

Lemma 6. *If a database B is obtained from a database A by a weak transformation rule, then A and B are $\mathcal{I}$-equivalent.*

Definition 7. A database is called *2-regular* if is not empty and all of its points have either F or (R) as cone. A database is called *1-regular* if it is not empty and all of its points have (LL) as cone.

Lemma 8. *Let A be a database and O be an open disk in $\mathbf{R}^2$.*

1. *If $A \cap O$ is a 2-regular database, then replacing $A \cap O$ by any other 2-regular database, yields a database that is $\mathcal{I}$-equivalent to A.*
2. *The same holds for 1-regular databases.*

Lemma 9. *Let A be a database and O be an open disk in $\mathbf{R}^2$. If $A \cap O$ consists of a part D surrounded by a circle, then replacing $A \cap O$ by a circle and D outside this circle, yields a database that is $\mathcal{I}$-equivalent to A.*

Proof of Proposition 5. (Sketch) We only sketch the proof for the line-cut&paste transformation. It is illustrated in Figure 7. First, the database is isotopically deformed. Second, weak line-cut&past is applied. Third, Lemma 9 is applied. Fourth, Lemma 8 is applied. The last arrow in the figure comprises three applications of weak line-cut&paste. □

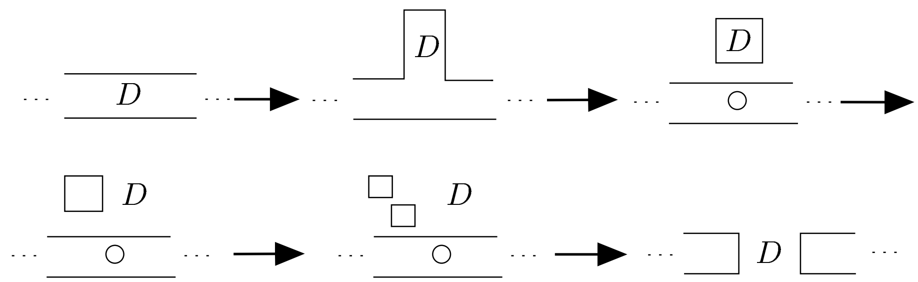


Fig. 7. The proof of the line-cut&paste transformation.

The transformation process. Having our tools, as furnished by Proposition 5, in place, we now show:

Proposition 10. *Let A and B be databases such that $\Pi_A \cong \Pi_B$. Then A and B can be transformed, by a finite sequence of strip-cut, strip-paste and line-cut&paste transformations and isotopies, into one and the same database.*

Proof. (Sketch) Since $\Pi_A \cong \Pi_B$, the singular points in A and B can be put in a one-one correspondence, with matching cones. For each singular point p in A and its corresponding point q in B, as well as for the point at infinity, we proceed as follows. From each region in the cone of p we cut out a bounded lobe coming out of p, using strip-cut. We do the same for q. Then, we choose a line l in the cone of p and then use line-cut&paste to connect l and the next line (in the clockwise order around p) into a loop starting and ending in p. We continue this process in the clockwise order around p until all lines, or all lines but one, form loops. We hereby make sure that no isolated parts of the database become trapped in these loops. Similarly, we cut loops in q starting from a line corresponding to l. If the number of lines in the point p is even, we obtain a "flower" around p. If the number is odd, we obtain a "flower with a stem". This stem is necessarily connected to another flower. Note that a flower can just be a single point.

After this process, all connected components are situated in the same area of $\mathbf{R}^2$. As "residual material" of the process we get isolated, bounded, regular parts, that are isotopic to circles, disks, or disks with holes. This material can be transformed to a single closed disk, a single circle, or the disjoint union of a closed disk and a circle. If a flower with points of type (LL) (resp. (R)) is present, the circle (resp. the disk) can even be absorbed by this flower.

The only way in which the resulting databases can still differ is that stems can connect different flowers. We can interchange stems by isotopically bringing them into a parallel position (this is possible since all stems are in the same area) and by then using a line-cut&paste transformation. This finally yields isotopic databases.

An illustration of the transformation process is given in Figure 8. □

We are finally ready for:

Proof of Theorem 3. (Sketch) The if-implication is immediate from Propositions 5 and 10.

For the only-if implication, assume $\Pi_A \ncong \Pi_B$. Then there exists at least one cone for which A has a different number of points than B. It is therefore sufficient to show that we can express in the spatial calculus that the database has a certain number of points having some fixed cone. Since the cone of a point in a database is invariant under isotopies, such a query is certainly $\mathcal{I}$-invariant. Query (†) in the Introduction is a trivial example of such a query; it expresses that there is a point whose cone is F.

The expressibility of such queries is based on the following refinement of the property that databases are locally conical around each point, including ∞ (proof omitted):

Lemma 11. *Let p be a point in a database A. There exists an $\varepsilon_0 > 0$ such that for every ε, with $0 < \varepsilon < \varepsilon_0$, $C(p,\varepsilon) \cap A$ is isotopic to $C(p,\varepsilon_0) \cap A$. There also exists an $\varepsilon_0 > 0$ such that for every ε, with $\varepsilon > \varepsilon_0$, $C((0,0),\varepsilon) \cap A$ is isotopic to $C((0,0),\varepsilon_0) \cap A$.*

By this lemma, it is sufficient to show that we can describe the intersection of such circles with the database by means of a spatial calculus formula. The points on such an intersection are of three types: interior points of the database, border-points of the interior of the database, and border-points of the database that are not on the border of the interior. The latter being points on lines. All three types of points can be described by a spatial calculus formula. There is also a spatial calculus formula that expresses that a point is between an ordered pair of points on a fixed circle in clockwise order. This completes the proof. □

6 Concluding remarks

Theorems 3 and 4 have two interesting corollaries. The first one is:

Corollary 12. *$\mathcal{I}$- and $\mathcal{H}$-equivalence are decidable.*

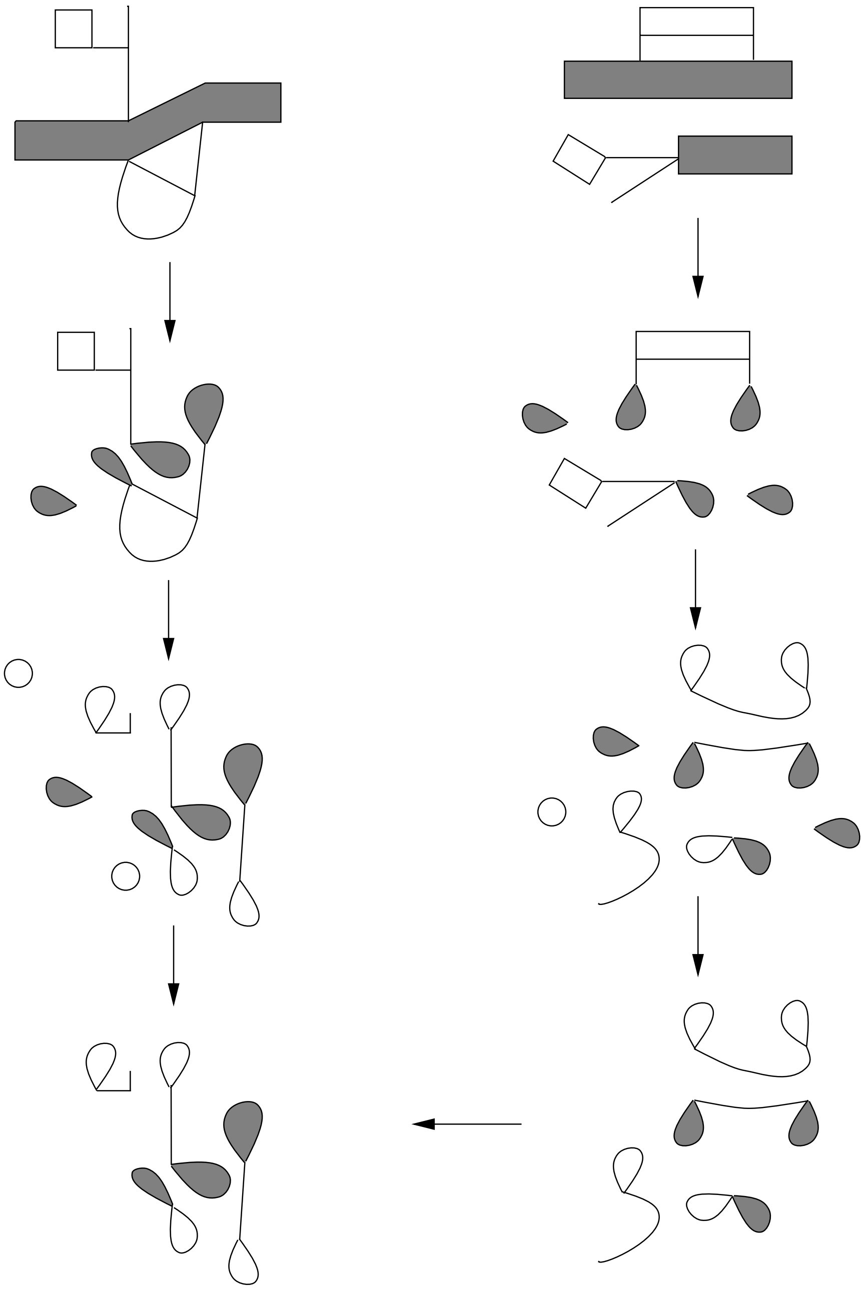


Fig. 8. Two databases with isomorphic point-structures and their transformation into the same database. After the first step, lobes have been cut out. After the second step, loops have been formed. After the third step, the residue has been absorbed. In the fourth step, stems are interchanged in the bottom-right database to obtain a database isotopic to the bottom-left database.

Proof. A decision algorithm for $\mathcal{I}$-equivalence is as follows:

$V_A := \emptyset$; $V_B := \emptyset$;
for each cone c do
  $A_{\mathbf{c}} := \{p \in A \cup \{\infty\} \mid \Pi_A(p) = \mathbf{c}\}$;
  $B_{\mathbf{c}} := \{p \in B \cup \{\infty\} \mid \Pi_B(p) = \mathbf{c}\}$;
  if $|A_{\mathbf{c}}| \neq |B_{\mathbf{c}}|$ **then return** *false*;
  $V_A := V_A \cup A_{\mathbf{c}}$; $V_B := V_B \cup B_{\mathbf{c}}$;
  if $V_A = A \cup \{\infty\} \wedge V_B = B \cup \{\infty\}$ **then return** *true*
od.

The algorithm tests whether $\Pi_A \cong \Pi_B$. The test $|A_{\mathbf{c}}| \neq |B_{\mathbf{c}}|$ can be performed as follows. Either $\mathbf{c}$ is F, (R), or (LL), in which case the test amounts to testing that one of $A_{\mathbf{c}}$ and $B_{\mathbf{c}}$ is empty and the other is not. This is a test expressible in first-order logic with polynomial inequalities and therefore decidable by Tarski's theorem. If $\mathbf{c}$ is another kind of cone, then both $A_{\mathbf{c}}$ and $B_{\mathbf{c}}$ are finite and symbolic algorithms for the first-order theory of the reals (e.g., Collins's algorithm [6, 2]) can effectively enumerate them. All other steps in the algorithm are also expressible in first-order logic with polynomial inequalities. The for-loop always terminates since in each database only a finite number of cones can appear. □

In order to formulate the second corollary, we define CALC($<$) to be the fragment of the spatial calculus in which as polynomial inequalities only those of the form $x-y > 0$ (i.e., $x > y$) are allowed. Call two databases $\mathcal{I}$-equivalent *under* CALC($<$) if they cannot be distinguished by $\mathcal{I}$-invariant queries of CALC($<$). We have:

Corollary 13. *Two databases are $\mathcal{I}$-equivalent if and only if they are $\mathcal{I}$-equivalent under* CALC($<$). *The same holds for $\mathcal{H}$-equivalence.*

Proof. (Sketch) The only-if implication is trivial. For the if-implication, assume A and B are not $\mathcal{I}$-equivalent. Then Π_A and Π_B are not isomorphic, and thus there exists at least one cone for which A has a different number of points than B. In the proof of Theorem 3 we saw that it is expressible in the spatial calculus that a database has a certain number of points having some fixed cone. The crucial observation is that this can actually be done in CALC($<$). Indeed, as base of the cone of a point we can take a rectangle instead of a circle (this is isotopic). All needed properties of rectangles can be expressed in CALC($<$). Also, that a point is an interior point (or a border point) is expressible in terms of rectangles instead of circles as well. Hence, CALC($<$) is sufficient. □

In this paper, we have focused on *closed* databases. However, the notion of point-structure, fundamental to our development, can also be defined for general semi-algebraic sets in $\mathbf{R}^2$. Unfortunately, due to the possible presence of components of the interior of semi-algebraic sets with mixed borders (open and closed), our transformation-based proof (in particular Proposition 10) does not carry over to this more general setting in a straightforward way.

We are also looking at other dimensions. In dimension one, the notions of $\mathcal{I}$-equivalence and $\mathcal{H}$-equivalence coincide with isotopic and homeomorphic. Generalizations to higher dimensions seem feasible. Indeed, the local cone structure around points in a semi-algebraic set, which provided the main inspiration for our work, also holds there. It remains to be investigated how the cut and paste transformations can be generalized.

References

1. D. Abel and B.C. Ooi, editors. *Advances in spatial databases—3rd Symposium SSD'93*, volume 692 of *Lecture Notes in Computer Science*. Springer-Verlag, 1993.
2. D.S. Arnon. Geometric reasoning with logic and algebra. *Artificial Intelligence*, 37:37–60, 1988.
3. M. Benedikt, G. Dong, L. Libkin, and L. Wong. Relational expressive power of constraint query languages. In *Proceedings 15th ACM Symposium on Principles of Database Systems*, pages 5–16. ACM Press, 1996.
4. J. Bochnak, M. Coste, and M.-F. Roy. *Géométrie algébrique réelle*. Springer-Verlag, 1987.
5. A. Buchmann, editor. *Design and implementation of large spatial databases—First Symposium SSD'89*, volume 409 of *Lecture Notes in Computer Science*. Springer-Verlag, 1989.
6. G.E. Collins. Quantifier elimination for real closed fields by cylindrical algebraic decomposition. *Lecture Notes in Computer Science*, 33:134–183, 1975.
7. M. Coste. Ensembles semi-algébriques. In *Géometrie algébrique réelle et formes quadratiques*, volume 959 of *Lecture Notes in Mathematics*, pages 109–138. Springer, 1982.
8. M.J. Egenhofer and R.D. Franzosa. Point-set topological spatial relations. *Int. J. Geographical Information Systems*, 5(2):161–174, 1991.
9. M.J. Egenhofer and J.R. Herring, editors. *Advances in Spatial Databases*, volume 951 of *Lecture Notes in Computer Science*. Springer, 1995.
10. J. Flum and M. Ziegler. *Topological Model Theory*, volume 769 of *Lecture Notes in Mathematics*. Springer-Verlag, 1980.
11. S. Grumbach and J. Su. First-order definability over constraint databases. In U. Montanari and F. Rossi, editors, *Principles and practice of constraint programming*, volume 976 of *Lecture Notes in Computer Science*, pages 121–136. Springer, 1995.
12. O. Gunther and H.-J. Schek, editors. *Advances in spatial databases—2nd Symposium SSD'91*, volume 525 of *Lecture Notes in Computer Science*. Springer-Verlag, 1991.
13. C.W. Henson, C.G. Jockusch, Jr., L.A. Rubel, and G. Takeuti. *First order topology*, volume CXLIII of *Dissertationes Mathematicae*. 1977.
14. P.C. Kanellakis, G.M. Kuper, and P.Z. Revesz. Constraint query languages. *Journal of Computer and System Sciences*, 51(1):26–52, August 1995.
15. E.E. Moise. *Geometric topology in dimensions 2 and 3*, volume 47 of *Graduate Texts in Mathematics*. Springer, 1977.
16. C.H. Papadimitriou, D. Suciu, and V. Vianu. Topological queries in spatial databases. In *Proceedings 15th ACM Symposium on Principles of Database Systems*, pages 81–92. ACM Press, 1996.

17. J. Paredaens, J. Van den Bussche, and D. Van Gucht. Towards a theory of spatial database queries. In *Proceedings 13th ACM Symposium on Principles of Database Systems*, pages 279–288. ACM Press, 1994.
18. A. Pillay. First order topological structures and theories. *Journal of Symbolic Logic*, 52(3), September 1987.