A characterization of first-order topological properties of planar spatial data

(Extended abstract)

Michael Benedikt
Bell Labs

Christof Löding
Lehrstuhl Informatik VII
RWTH Aachen

Jan Van den Bussche
Limburgs Universitair Centrum

Thomas Wilke
Institut für Informatik und Praktische Mathematik
Christian-Albrechts-Universität zu Kiel

ABSTRACT

Closed semi-algebraic sets in the plane form a powerful model of planar spatial datasets. We establish a characterization of the topological properties of such datasets expressible in the relational calculus with real polynomial constraints. The characterization is in the form of a query language that can only talk about points in the set and the "cones" around these points.

1. INTRODUCTION

A simple yet powerful way of modeling spatial data is using *semi-algebraic sets*. A (possibly infinite) subset of n-dimensional Euclidean space $\mathbf{R}^n$ is called semi-algebraic if it can be defined by a boolean combination of polynomial inequalities. The present paper is particularly concerned with sets in the plane, $\mathbf{R}^2$. First-order logic over the reals with arithmetic, order, and an extra binary predicate S, denoted here by FO[$\mathbf{R}$], then becomes a spatial query language, fitting in the well-known framework of constraint query languages introduced by Kanellakis, Kuper, and Revesz [10, 12]. For example, 'is the set S bounded?' can be expressed in FO[$\mathbf{R}$] as $\exists b \forall x \forall y (S(x, y) \rightarrow (-b < x < b \wedge -b < y < b))$. We will consider only sets that are closed in the ordinary topology on $\mathbf{R}^2$. This assumption is of great help from a technical point of view, and is harmless from a practical point of view.

A property of spatial datasets is called *topological* if it is invariant under topological transformations of the plane. More precisely, whenever the property holds for some A, it must also hold for any other A' that is the image of A under a homeomorphism of the plane (a bijection $f\colon \mathbf{R}^2 \rightarrow \mathbf{R}^2$ such that both f and f^{-1} are continuous). For example, the above-mentioned property 'the set is bounded' is topological, as is 'the set is a plane curve', and 'the set has three connected components'. In contrast, properties like 'the set contains a straight line segment' and 'the set is a perfect circle' are not topological. Apart from our interest in topological properties as a natural and mathematically well-motivated class of properties, they are also practically motivated by geographical information systems [5, 6, 7, 13].

Given the above setup, a natural question is to understand exactly which topological properties are first-order, i.e., expressible in FO[$\mathbf{R}$]. For example, 'the set is a plane curve' is first-order [20], but properties involving topological connectivity are not [2, 9, 12]. It is undecidable whether a given FO[$\mathbf{R}$]-sentence is topological [20]. Yet, this leaves open the possibility to syntactically capture topological FO[$\mathbf{R}$]. Indeed, a syntactic characterization has been a target of earlier work on the topic [18, 19, 11]. This is what we will do in the present paper.

Our starting point is the work by Kuijpers, Paredaens and Van den Bussche [19], which considers the more basic question of understanding topological elementary equivalence: when are two sets indistinguishable by means of topological FO[$\mathbf{R}$]-sentences? A characterization was discovered in terms of the *cone types* occurring in the two sets. Indeed, semi-algebraic sets are topologically well-behaved in that locally around each point they are "conical" [3]. The cone of a point consists of the lines and regions arriving in the point, and can thus be represented as a (circular) string of L's (lines) and R's (regions). The characterization states that two sets are topologically elementary equivalent if and only if they have the same number of occurrences of every cone.

This characterization immediately suggests "Cone Logic" [11]: a topological query language that allows to express boolean combinations of properties of the form 'there are at most k occurrences of cones satisfying property γ'. Here, γ is any first-order property of cones viewed as circular strings. The first-order properties of strings are well-known to be the star-free regular languages [22]. It is tempting to conjecture that Cone Logic exactly captures topological FO[$\mathbf{R}$], but a proof has remained elusive; until now we only knew it for the special case of sets consisting of regions only, i.e., without L's in cones [11].

PODS 2004 June 14–16, 2004, Paris, France.
Copyright 2004 ACM 1-58113-858-X/04/06 ... $5.00.

The main result of the present paper is that Cone Logic indeed captures topological FO[$\mathbf{R}$], over arbitrary closed semi-algebraic sets in the plane. Our proof develops extensively the idea of coding planar sets by finite structures [11]. This coding may well have other applications. Our proof also introduces new invariance arguments. These arguments show that first-order properties of structures enhanced with some form of "decoration", but invariant under the particular choice of decoration, are in fact expressible without referring to the decoration at all. Compare this to the famous example by Gurevich [1, Exercise 17.27], [4, Proposition 2.5.6], where the decoration is a total order. In that example the decoration is shown to be indispensable. In contrast, we will encounter kinds of decorations that are indeed dispensable. Of course, our final theorem can be seen as stating that queries can be dramatically simplified syntactically if they satisfy an invariance assumption. But in the process we develop techniques for doing this kind of elimination in the context of discrete structures. Finally, our proof not surprisingly relies on the collapse theorems for constraint queries on finite structures [16, 2]; as a matter of fact, the characterization we prove can be viewed as a lifting of collapse from finite structures to infinite sets.

Our proof also yields some variations and generalizations of the main result. For example, if one is interested in semi-linear sets only (i.e., sets definable using linear polynomials only), then Cone Logic still captures the first-order topological properties. Also, the result generalizes to o-minimal expansions of the reals [23].

In closing we should also mention previous work on topological properties not of single sets, but of entire collections of sets [17, 21]. This also covers the case of sets not necessarily closed, because such a general set can be represented by two closed ones, namely, its closure, and the closure of its complement. Even in the case of just two sets, Grohe and Segoufin [8] showed that topological elementary equivalence can no longer be characterized by looking at cones only. Yet, they were able to provide a characterization in the special case of collections of sets with "regular" points only. It would be interesting to lift this characterization to the level of queries, just like we have done here for the case of single sets.

2. PRELIMINARIES

Spatial data

In this paper, a *spatial dataset* (or just dataset) is defined as a semi-algebraic set in $\mathbf{R}^2$ that is closed in the ordinary topological sense. More concretely, this is a set that can be defined as a union of sets of the form $\{(x,y) \in \mathbf{R}^2 \mid P_1(x,y) \geqslant 0 \wedge \cdots \wedge P_m(x,y) \geqslant 0\}$, where each P_i is a polynomial in the variables x and y with integer coefficients. When all P_i's are linear, the set is called *semi-linear.*

First-order logic over the vocabulary $(0, 1, +, \times, <, S)$, with S a binary relation symbol, is denoted by FO[$\mathbf{R}$]. An FO[$\mathbf{R}$]-formula φ can be evaluated on a dataset A by letting variables range over $\mathbf{R}$, interpreting the arithmetic symbols in the obvious way, and interpreting $S(x,y)$ to mean that the point (x,y) is in A.

To formalize what it means for two datasets A and B to be topologically the same, we use the notion of *isotopy.* The intuition behind an isotopy is a continuous deformation of the plane. A and B are called *isotopic* if there is an isotopy h such that $h(A) = B$.[1] An FO[$\mathbf{R}$]-sentence φ is now called *topological* if whenever datasets A and B are isotopic, then $\varphi(A)$ is true if and only if $\varphi(B)$ is true.

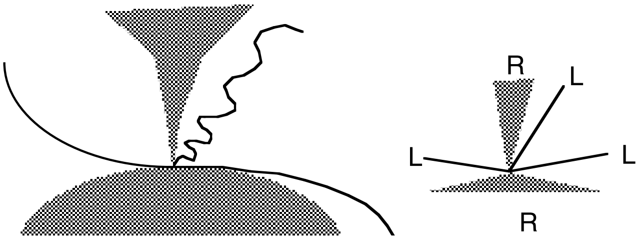

Figure 1: A dataset and the cone of one of its points.

Cones

A known topological property of semi-algebraic sets [3] is that locally around each point they are conical. This is illustrated in Figure 1. Formally, for a point p and a real $\varepsilon > 0$, denote the closed disk with center p and radius ε by $D(p,\varepsilon)$, and denote its bordering circle by $C(p,\varepsilon)$. Then for every point p of a dataset A there exists an $\varepsilon > 0$ such that $D(p,\varepsilon) \cap A$ is isotopic to the planar cone with top p and base $C(p,\varepsilon) \cap A$. We thus refer to *the cone of* p *in* A.

Every dataset A is also conical around infinity. Formally, there exists an $\varepsilon > 0$ such that $\{(x,y) \mid x^2 + y^2 \geqslant \varepsilon^2\} \cap A$ is isotopic to $\{\lambda \cdot (x,y) \mid (x,y) \in C((0,0),\varepsilon) \cap A \wedge \lambda \geqslant 1\}$. 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 at* ∞ *in* A.

We will identify cones with circular lists. The cone having a full circle as its base (which appears around interior points) is represented by the single letter F. Any other cone can be represented by a circular list of L's and R's (for "line" and "region") which describes the cone in a complete clockwise turn around the top. For example, the cone of Figure 1 is represented by $(LLRLR)$. The cone with empty base (which appears around isolated points) is represented by the empty list ( ).

There are only three cones that can occur infinitely often in a dataset: F, (LL) (the cone around points on curves),

[1] Formally, an isotopy is a homeomorphism of the plane that is isotopic to the identity. 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.

A more relaxed notion of "being topologically the same" is to simply require that B is the image of A under a homeomorphism rather than an isotopy. The only difference between the two notions is that the latter considers mirror images to be the same, while the former does not. Indeed, every homeomorphism either is an isotopy itself, or is isotopic to a reflection [15]. All the results we will present under isotopies have close analogues under homeomorphisms.

and (R) (the cone around points on the smooth border of a region). We call these the *regular cones*; all other cones are called *singular*. The points with a singular cone are called the *singular* points of the dataset. Because datasets are semi-algebraic, they can have only a finite number of singular points.

3. THE CHARACTERIZATION

We will characterize the properties of datasets expressible by topological FO[$\mathbf{R}$]-sentences. Our characterization will be in terms of conditions on the cones occurring in the datasets, as well as on the number of such cones.

Given that cones are circular strings over the alphabet $\{L, R\}$ (except for the special cone F), it is convenient to use standard formal language theory to define properties of cones. Specifically, recall that a *star-free regular expression* over a finite alphabet Σ is an expression built up from the atoms Σ^*, ϵ, and a, for $a \in \Sigma$, using the operations union, difference, and concatenation. Such expressions define string languages, i.e., sets of strings over Σ, in the obvious way. If a string s is in the language defined by e, we also say that s *satisfies* e.

But these expressions can also be used to define sets of circular strings. It suffices to agree that a circular string satisfies expression e if it equals the circularization of a normal string satisfying e. For example, the expression LR^*L defines all cones that have only two L's, and these L's must be consecutive.[2]

This leads us to a natural topological query language called "Cone Logic" or $\mathcal{CL}$ for short. A $\mathcal{CL}$-sentence is a boolean combination of atomic conditions of the following possible forms:

1. F, meaning that there exists a point in the dataset with cone F (in which case there will automatically be infinitely many such points).
2. $F(\infty)$, meaning that the cone at infinity is F.
3. $|e| \geqslant n$, with e a star-free regular expression over $\{L, R\}$ and n a natural number, meaning that there are at least n points in the dataset whose cone satisfies e.
4. $e(\infty)$, meaning that the cone at infinity satisfies e.

Note that properties of datasets expressed in $\mathcal{CL}$ are always topological. Every $\mathcal{CL}$-sentence can be equivalently expressed in FO[$\mathbf{R}$], i.e., for each $\mathcal{CL}$-sentence ψ there exists an FO[$\mathbf{R}$]-sentence φ such that $\psi(A) = \varphi(A)$ for each dataset A. Our main result is that $\mathcal{CL}$ actually characterizes the topological FO[$\mathbf{R}$]-sentences:

Main Theorem. *For each topological* FO[$\mathbf{R}$]*-sentence φ there exists a $\mathcal{CL}$-sentence ψ such that $\varphi(A) = \psi(A)$ for each dataset A.*

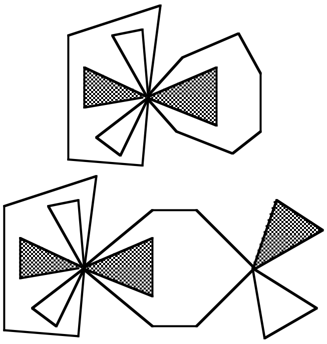

Figure 2: A single flower and a flower pair.

4. FLOWER NORMAL FORM

For simplicity of presentation, in proving our characterization we will restrict attention to bounded datasets, so that the point at infinity can be ignored. Incorporating infinity makes the proof technically more complicated, but it involves no new insights.

A more drastic restriction is to datasets in what we call *flower normal form*. Such a dataset, called a *flower dataset* for short, consists of a number of connected components, of two possible kinds: *single flowers* and *flower pairs*. Both are illustrated in Figure 2.

- A single flower is a connected dataset with exactly one singular point, where every R in the cone is a small "lobe" emanating from the point but meeting no other R's. Necessarily, every line emanating from the point also arrives somewhere else in the point, i.e., all lines are *self-lines*. Note that a self-line is visible as two L's in the cone of the singular point, so a single flower has an even number of L's in the cone.
- A flower pair consists of two single flowers, except that some of the lines cross between the two singular points. These *cross-lines* must be consecutive: between two emanating cross-lines there cannot be a self-line. Note that a cross-line is visible as one L in the cone of each of the two singular points. Paired flowers need not have an even number of L's in their cone.

The justification for flower normal form comes from the notion of *topologically elementary equivalence*, or t.e.e. Two datasets are t.e.e. if no topological FO[$\mathbf{R}$]-sentence distinguishes between them. We recall:

Theorem 1 ([19]). *Two datasets are t.e.e. if and only if they have the same cone at ∞, and every other cone occurs exactly the same number of times in both sets (a finite number for singular cones, or infinitely often for regular cones).*

By this theorem, every bounded dataset is t.e.e. to a flower dataset, provided the set does not have any connected components consisting of regular points only. A simple argument (omitted) shows, however, that if our characterization holds over the datasets without such regular components, then it

[2] The subexpression R^* can be viewed as a shorthand for $\Sigma^* - \Sigma^* L \Sigma^*$ with $\Sigma = \{L, R\}$.

holds over all datasets. So from now on we can focus on flower datasets.

5. OVERVIEW OF THE PROOF

The global outline of our proof is that for any topological FO[$\mathbf{R}$]-sentence φ we can find two natural numbers k and ℓ such that any two flower datasets that are "(k,ℓ) equivalent" are indistinguishable by φ. Hence, φ is a union of (k,ℓ)-equivalence classes. Now (k,ℓ)-equivalence will have two good properties: it will be of finite index, and every equivalence class can be defined in $\mathcal{CL}$. As a consequence, φ can be written as a finite disjunction of $\mathcal{CL}$-sentences, and we will have proven our characterization.

To define (k,ℓ)-equivalence, we need the notion of a *cone structure*. Up to isomorphism, this is a finite structure with domain $\{1,\dots,n\}$, for some natural number n. Every element is labeled with L or R. Moreover, the structure includes a ternary relation B (for "between"). $B(x,y,z)$ holds if y comes before z in the following sequence: x, $x+1$, …, n, 1, 2, …, $x-1$. We say that y is between x and z. So, a cone structure is an explicit representation of a cone. Note that a cone structure is the circular version of what is known as a *word structure* over the alphabet $\{L,R\}$; in word structures we have the total order on $\{1,\dots,n\}$ instead of the betweenness relation.

Cone structures allow us to use first-order logic sentences over the vocabulary (L,R,B) to express properties of cones. In particular, a *k-type* is a maximally consistent conjunction of first-order sentences of quantifier rank k, over the vocabulary (L,R,B) of cone structures. Now two datasets are called *(k,ℓ)-equivalent* if for every k-type τ, they either have precisely the same number of singular cones of type τ, or the two numbers are both at least 3ℓ. That (k,ℓ)-equivalence classes are indeed definable in $\mathcal{CL}$ follows easily from the well-known translation of first-order sentences over word structures to star-free regular expressions [22].

The proof that two (k,ℓ)-equivalent flower datasets are indistinguishable by φ proceeds by transforming one dataset into the other, using a repertoire of transformations that are indistinguishable by φ. They are the following:

Marrying and divorcing: Two single flowers can be married to become a flower pair with the same cones (with, e.g., all lines going across). The inverse of this transformation is allowed as well.

Spouse swapping: Two flower pairs $\{f_1,f_2\}$ and $\{f_3,f_4\}$ can be replaced by two other flower pairs $\{g_1,g_3\}$ and $\{g_2,g_4\}$, such that the cones of f_i and g_i are identical for $i=1,2,3,4$.

Substitution: A single flower can be replaced by any other single flower whose cone has the same k-type. A flower pair can be replaced by any other flower pair, as long as the pair of cone k-types is the same.

Stretching: For any k-type with at least ℓ occurrences of single flowers of that cone type, we can add any number of additional single flowers of the type. For any pair of k-types with at least ℓ occurrences of a flower pair with that pair of cone types, we can add any number of additional such flower pairs.

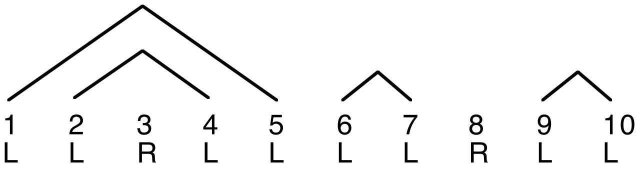

Figure 3: The single flower of Figure 2 is a drawing of the single cycle shown here.

Formally, we will prove:

Lemma 1. *For any topological* FO[$\mathbf{R}$]*-sentence φ there exist k and ℓ such that the above transformations are indistinguishable by φ.*

Lemma 2. *Any two (k,ℓ)-equivalent flower datasets can be transformed into each other by the above transformations.*

Lemma 2 is shown by marrying the cone types in a canonical way, after first stretching their numbers to match; the details are omitted. Proving Lemma 1 is a large enterprise which is taken up in the next two sections. Note that the legitimacy of Marrying, Divorcing, and Spouse Swapping, which do not mention k and ℓ anyway, already follows from Theorem 1.

6. CODES AND DRAWINGS

We are going to represent flower datasets by abstract finite structures, which we call *codes*. A code is a disjoint union of components, of two possible forms: *single cycles* and *cycle pairs*. A single cycle represents a single flower, and a cycle pair represents a flower pair.

Single cycles

A single cycle is a word structure over the alphabet $\{L,R\}$, with two modifications. First, the number of L's must be even. Second, the structure includes a matching[3] G on the L-labeled nodes that is *planar* in the following sense: if $i<j<k<\ell$, then it is forbidden that $G(i,k)$ and $G(j,\ell)$ both hold.

Note that a cycle is not cyclic at all, but we still use the name because cycles will always have a circular interpretation: we will never need to distinguish two cycles that are the same up to rotation. In particular, there is an obvious notion of a single flower Y being a *drawing* of a single cycle C, which we do not define formally, but illustrate in Figure 3. When Y is a drawing of C, then Y is a drawing of every rotation of C as well.

Cycle pairs

A cycle pair is a disjoint union of two single cycles, with two modifications. First, the number of L's in each cycle need not be even. Second, instead of G being a planar matching on the L's of each single cycle separately, a subsequence

[3] A matching on a set X is the symmetric closure of a bijection from one half of X to the other half.

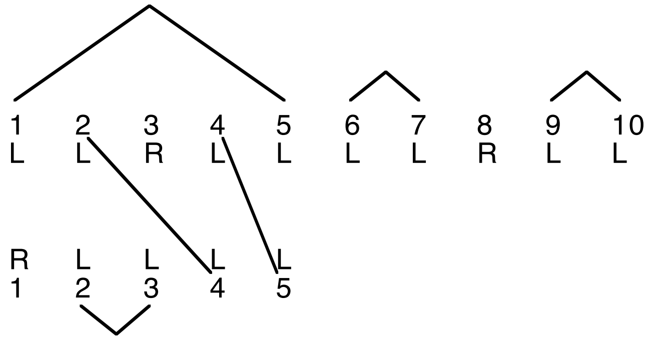

Figure 4: The flower pair of Figure 2 is a drawing of the cycle pair shown here.

of consecutive L's in one cycle is now matched by G to a subsequence of consecutive L's in the other cycle. Here, we call two L's consecutive if there is no other L in between (there may be R's), where "between" has its meaning as in cone structures.

Clearly, the cross-matches by G model the cross-lines in a paired flower, and again there is an obvious notion of a flower pair being a drawing of a cycle pair, illustrated in Figure 4.

Codes

A *code* is now defined as a disjoint union of single cycles and paired cycles. We denote the vocabulary of codes, consisting of $<$, L, R, and G by Γ.

A flower dataset A is called a *drawing* of a code C if A has a separate drawing for every single cycle and every cycle pair of C, and nothing more. In that case we also say that C is a *representation* of A.

The following proposition demonstrates the utility of codes.

Proposition 1. *For any topological* $\mathrm{FO}[\mathbf{R}]$*-sentence* φ *there exists a first-order sentence* ψ *over* Γ *such that for every flower dataset A, and every representation C of A, we have* $\varphi(A) = \psi(C)$.

Proof. An *embedded code* is a code embedded in $\mathbf{R}$, so the abstract nodes happen to be real numbers. An embedded code is called *well embedded* if within each component, the ordering on the nodes as real numbers agrees with the order given by the cycles. Moreover, all nodes belonging to one component must be either all smaller or all larger (in the real order) than all nodes belonging to another component, and in a cycle pair all nodes of one of the cycle are all smaller than all nodes of the other cycle.

Until now, $\mathrm{FO}[\mathbf{R}]$-formulas were always understood to be over the vocabulary of $\mathbf{R}$ $(0, 1, +, \times, <)$ expanded with a binary relation S to address the spatial dataset to be queried. But in the following lemma we use $\mathrm{FO}[\mathbf{R}]$-formulas on embedded codes, where we expand $\mathbf{R}$ not with S but with Γ. We refer to such formulas as $\mathrm{FO}[\mathbf{R}]$-formulas over Γ.

Lemma 3. *There exists an* $\mathrm{FO}[\mathbf{R}]$*-formula* $draw(x, y)$ *over* Γ *such that for any well-embedded code C, the set* $\{(x, y) \in \mathbf{R}^2 \mid (x, y) \in draw(C)\}$ *is t.e.e. to a drawing of C. If C is not a well-embedded code, then* $draw(C)$ *is empty.*

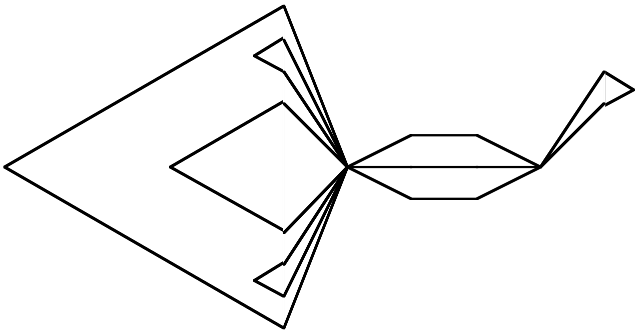

Figure 5: "Proof by picture" of Lemma 3. The figure shows a systematic drawing of an embedded cycle pair (involving only L's).

We omit the proof, but Figure 5 gives an idea of the construction.

Consider now the composed query $\varphi \circ draw$. By the natural-active collapse theorem [2], we can equivalently express $\varphi \circ draw$ by an $\mathrm{FO}[\mathbf{R}]$-sentence χ over Γ in which the quantifiers range over the nodes of C only. Moreover, the query is *order-generic:* for any embedded Γ-structure C, and for any monotone bijection ρ of $\mathbf{R}$, we have $\varphi(draw(C)) = \varphi(draw(\rho(C)))$. Hence, by the generic collapse theorem [16, 2], we can further reduce χ to a first-order sentence ψ just over Γ. In other words, ψ sees just the abstract code, not the actual embedding, were it not that it still sees a total order on *all* nodes, instead of just the partial orders given within the cycles.

Fortunately, we can get rid of the order among cycles using a model-theoretic argument which we just sketch. Observe that ψ is actually invariant under the particular way the different cycles compare to each other. Hence, if r is the quantifier rank of ψ, we may fix an arbitrary order on r-types and assume that the order among the components is according to their r-types. This leaves us only with the order among components of the same r-type, and the order among the two cycles in a cycle pair. These can be dispensed with by an Ehrenfeucht-Fraïssé game argument. □

As we will see in the next section, Proposition 1 opens the door towards proving the remaining Lemma 1. But the proposition is also interesting in itself: it shows that topological $\mathrm{FO}[\mathbf{R}]$-queries can be supported by a standard finite relational database representation of the spatial dataset.

7. INVARIANCE ARGUMENTS

Fix a topological $\mathrm{FO}[\mathbf{R}]$-sentence φ, and let ψ be obtained from Proposition 1. Now observe that ψ must be invariant under rearrangement of the planar matching G in any of the components of a code C. Indeed, in a drawing A of C, this yields a rearrangement of self-lines (and possibly cross-links in a flower pair), and φ cannot notice such rearrangements by t.e.e. Hence, ψ cannot notice them in C either. We say in short that ψ is *planar invariant.* Moreover, ψ is invariant under rotation of the cycles. We say that ψ is *rotation invariant.*

The following lemma, proven by a simple Ehrenfeucht-Fraïssé game argument (omitted), shows that we can "push down" invariance to the level of the separate connected components of a code.

Pushdown Lemma. *Every invariant sentence over codes can be rewritten as a boolean combination of conditions of the form $|\gamma| \geqslant n$, with γ again invariant. Such a condition means that at least n components in the code satisfy γ.*

We are now ready for the

Proof of Lemma 1. As already mentioned, we must deal only with the transformations of Substitution and Stretching.

Substitution revolves around eliminating the major difference between cycles and cones, namely that codes contain the matching relation G. The following crucial lemma allows us to get rid of G. The proof, which is quite involved, is sketched at the end of this section.

Invariance Lemma. *Let ψ be a first-order sentence over word structures equipped with planar matchings G. If ψ is planar-invariant, then it can be equivalently written without G, i.e., as a first-order sentence over words.*

We would like to apply the Invariance Lemma to each component sentence γ from the Pushdown Lemma that works on single cycles, so that it can be rewritten without G. Since γ is also rotation-invariant, it is then an easy matter to rewrite it further to get a cone sentence. If we then take k to be the maximal quantifier rank of all the resulting component cone sentences, we obtain the desired result that Substitution of a single flower is indistinguishable by φ. Moreover, if we take ℓ to be the maximal bound n from the Pushdown Lemma, we obtain the same for Stretching of a single flower.

There is a small problem, however, since the Invariance Lemma deals with word structures equipped with a total planar matching, while cycles have a planar matching only on their L's. We can solve this as follows. Let r be the quantifier rank of γ. A standard Ehrenfeucht-Fraïssé game argument shows that γ cannot distinguish two words that agree if we count blocks of R's only up to 2^r. This allows us to view γ as a sentence on the word consisting of the L's only, but where each L is labeled with one of the new letters P_i for $i = 0, \ldots, 2^r$, where P_i for $i < 2^r$ means that there are exactly i R's following the element, and P_{2^r} means that there are at least 2^r R's. After using this abstraction, we can then apply the Invariance Lemma. Other applications of the Invariance Lemma in the next paragraphs must also be interpreted in this manner.

For flower pairs the argument for Substitution is a bit more complicated. First, observe that by t.e.e. we can always rearrange G so that there is either exactly one cross-match, or none at all. The first case occurs when the number of L's in each cycle is odd; the second when the number is even. Hence, if we restrict a component sentence γ to pairs with even cross-matches, then we can rewrite it as an assertion about cone types; in this case we can arrange for there to be no cross-matches, leaving us with simply two word structures equipped with planar matchings, from which the matching can be removed using the Invariance Lemma. Similarly if we restrict to pairs with odd cross-matches; the cross-matched pair can be replaced in favor of a distinguished label for the ends of the crossing line, and the Invariance Lemma can be applied to these enhanced words. From this we see not only that Substitution is justified for pairs of the same parity, but also that γ can be written as a sentence using only the cone signature, provided that a parity check is permitted in addition to first order logic.

Now on the other hand, again by t.e.e. we can also maximize the number of cross-matches, so that the smaller of the two cycles has all its lines crossing to the larger cycle. Again incorporating blocks of R's into the labels of the L's as before, this yields a view of the cycle pair as a word of pairs of letters, followed by a word comprising the unpaired elements from the larger cycle, on which we still have a planar matching G. Over this view, a component sentence γ can be broken up into sentences γ_1 quantifying over the letter pairs and sentences γ_2 quantifying over the unpaired elements. The planar matching G can be removed from the latter sentence using the Invariance Lemma. Viewing now the unpaired elements as paired with a dummy letter, we obtain a first-order sentence over words over a pair alphabet.

So on the one hand γ expresses a combination of regular conditions on the words in a pair (by the paragraph before the previous one), and on the other hand it expresses a star-free regular condition on the word pair (viewed as a word over the pair alphabet). A simple argument (omitted) then shows that γ must already be a combination of star-free regular conditions on the separate words. Since these conditions are rotation-invariant (here we apply a mini-version of the Pushdown Lemma), we have arrived at the desired combination of cone types, which justifies Substitution in general, as well as Stretching for pairs. □

Proof sketch of the Invariance Lemma. We will use the following notation for vocabularies. For words over an alphabet Σ we use $\mathcal{L}_W\Sigma = \Sigma \cup \{<\}$ and for words equipped with a planar matching we use $\mathcal{L}_{WM}\Sigma = \Sigma \cup \{<, G\}$. From $<$ we can easily define the binary relation $suc(x, y)$ meaning that either y is the successor of x or x is the last position in the word and y is the first position in the word.

For an $\mathcal{L}_{WM}\Sigma$ sentence ϕ we denote by $M(\phi)$ the set of models (words equipped with planar matchings) of ϕ, and by $W(\phi)$ we denote the set of words obtained from $M(\phi)$ by omitting the planar matchings. If ϕ is planar-invariant, then $M(\phi)$ is completely determined by $W(\phi)$. Further, note that $W(\phi)$ only contains words of even length. For an $\mathcal{L}_W\Sigma$ sentence θ we denote by $W(\theta)$ the set of words w with $w \models \theta$. In general, the set $W(\theta)$ can contain words of even and words of odd length.

To prove the invariance lemma we have to show that for each

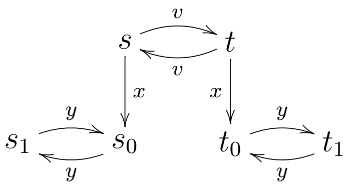

Figure 6: pattern for two-stage counter

planar-invariant $\mathcal{L}_{WM}\Sigma$ sentence ϕ there exists an $\mathcal{L}_W\Sigma$ sentence θ such that $W(\phi) = W(\theta) \cap (\Sigma\Sigma)^*$.

The main idea is to show that $W(\phi)$ for a planar-invariant $\mathcal{L}_{WM}\Sigma$ sentence ϕ is regular and contains counters of a very restricted type only. The proof is then completed by showing that a regular language $W \subseteq (\Sigma\Sigma)^*$ with these restrictions on the occurrence of counters is of the form $W' \cap (\Sigma\Sigma)^*$ for an $\mathcal{L}_W\Sigma$ definable language W'. We start with some terminology on counters.

Let $\leftrightarrow_W$ be the relation defined by $u \leftrightarrow_W v$ if $u \in W$ and $v \in W$, or $u \notin W$ and $v \notin W$. Nerode's congruence for W, denoted $\equiv_W$, is defined by $u \equiv_W v$ if $uw \leftrightarrow_W vw$ for every $w \in \Sigma^*$. A pair (u, v) with $u, v \in \Sigma^*$ is a counter of a language W if there exists $n > 0$ such that $u \not\equiv_W uv$ and $uv^n \equiv_W u$. The smallest such n is called the *period* of (u, v), and $|v|$ is called the *progression* of (u, v). An (n, m)-counter is a counter with period n and progression m. We use the expression $(2, \mathrm{odd})$-counter to denote a $(2, m)$-counter for some odd m.

We say that (u, v, x, y) is a *two-stage counter* of W if (u, v), (ux, y) and (uvx, y) are counters such that $ux \not\equiv_W uvxy$. In the minimal DFA for W a two-stage counter correponds to the pattern depicted in Figure 6, where s is reachable via u from the initial state and s_0 and t_1 are required to be distinct.

We will make intensive use of the well known result that a regular language $W \subseteq \Sigma^*$ is definable in $\mathcal{L}_W\Sigma$ if, and only if, it is counter-free [14].

To show the required restrictions on counters in $W(\phi)$ we introduce two special types of planar matchings. We call G a chain matching if it satisfies

$$\forall xy(G(x, y) \to (suc(x, y) \lor suc(y, x))) \enspace ,$$

and a parenthetical matching if it satisfies

$$\forall x_0x_1x_2x_3(suc(x_0, x_1) \land suc(x_2, x_3) \\ \to (G(x_0, x_3) \leftrightarrow G(x_1, x_2))) \enspace .$$

The main steps in the proof of the invariance lemma are the following.

1. If ϕ is invariant under chain matchings, then $W(\phi)$ is regular and only contains $(2, \mathrm{odd})$-counters.

2. If ϕ is invariant under chain matchings and parenthetical matchings, then $W(\phi)$ does not contain two-stage counters.

3. If W is a regular language containing only $(2, \mathrm{odd})$-counters and no two-stage counter, then there is an $\mathcal{L}_W\Sigma$ sentence θ such that $W = W(\theta) \cap (\Sigma\Sigma)^*$

Step 1 is shown by introducing a copy $\bar{\Sigma}$ of the alphabet and considering the language $\bar{W}(\phi)$ obtained from $W(\phi)$ by replacing in each word every second letter by the corresponding letter from $\bar{\Sigma}$. Now G can be expressed using *suc* and the alternation between letters from Σ and $\bar{\Sigma}$. Hence, $\bar{W}(\phi)$ is $\mathcal{L}_W\bar{\Sigma}$ definable and thus regular and counter-free. By projecting the letters from $\bar{\Sigma}$ to the corresponding letters of Σ we reobtain $W(\phi)$ and one can easily observe that the counters introduced by this operation must all be $(2, \mathrm{odd})$-counters.

To prove step 2 we pass to "folded" words. Let Σ_{fold} be the set of all column vectors over Σ with two rows, called folded letters. Words over this alphabet are called folded words. A word w equipped with a parenthetical cycle bijection G such that the first and the last position of the word are in the relation G corresponds in a natural way to a folded word. This folded word is obtained by reversing the second half of w and writing it below the first half of w. In this way two positions that are in the same column of the folded word are positions that are linked by G in w. If we apply this operation to all words in $W(\phi)$ we obtain the language $W_{\mathrm{fold}}(\phi)$ and ϕ can be rewritten into a $\mathcal{L}_W\Sigma_{\mathrm{fold}}$ sentence defining $W_{\mathrm{fold}}(\phi)$. Now one can show that a two-stage counter in $W(\phi)$ induces a counter in $W_{\mathrm{fold}}(\phi)$ contradicting the fact that $W_{\mathrm{fold}}(\phi)$ is $\mathcal{L}_W\Sigma_{\mathrm{fold}}$ definable.

In step 3 we construct from the minimal DFA $\mathcal{A}$ for W a new DFA $\mathcal{A}'$ accepting a counter-free language W' such that $W = W' \cap (\Sigma\Sigma)^*$. At the beginning, $\mathcal{A}'$ simulates $\mathcal{A}$. As soon as it has processed a word u such that (u, v) is a counter of W, it goes to a pair of states (s, t), where s is the state reached after reading u, and t is the state reached in $\mathcal{A}$ after reading uv. Starting from this state $\mathcal{A}'$ simulates the product automaton $\mathcal{A} \times \mathcal{A}$. A state (s_1, s_2) is accepting in $\mathcal{A}'$ if s_1 or s_2 is accepting in $\mathcal{A}$. The language accepted by this automaton has the desired properties. □

8. DISCUSSION

In Section 3 we already mentioned that $\mathcal{CL}$ can indeed be simulated in FO[**R**]. Actually, this is already possible in FO[<], the fragment of FO[**R**] that does not use arithmetic on **R**, only order. As an immediate corollary of our theorem we thus obtain:

Corollary 1. *Every topological* FO[**R**]*-sentence on closed semi-algebraic sets in the plane can already be expressed in* FO[<].

This is a nice analog of the generic collapse theorem [2] used in the proof of Proposition 1, which says exactly the same for order-generic FO[**R**]-sentences on finite structures over the reals. Thus our theorem can be viewed as a "lifting" of collapse from finite structures to infinite datasets.

We note that the drawing arguments to prove our Main Theorem (as well as the drawings used in the proof of Theorem 1 [19]) all remain within semi-*linear* sets, and hence the entire argument could have taken place there. Hence we have also proved:

Corollary 2. *Every* FO[**R**]*-query that is topological over (closed) semi-linear datasets is equivalent over semi-linear datasets to a CL sentence.*

Note that there are FO[**R**]-queries that are topological over semi-linear datasets but not over all semi-algebraic datasets. Indeed one can write an FO[**R**]-sentence (even without multiplication) that is true exactly for those datasets that are "line-like"—definable with addition (possibly with real parameters). Such a sentence is a tautology over semi-linear datasets but is not topological over semi-algebraic ones.

The theorem also lifts up to any family of sets that includes the semi-linear sets and in which every set is isotopic to a semi-linear one; for example, this is known to hold for the collection of sets definable in an *o*-minimal expansion of the real ordered group.

Likewise we use little about FO[**R**]*queries.* Our argument goes through for constraint query languages over expansions of the real field which have the properties that: a) definable sets are isotopic to semi-linear sets and b) the generic collapse theorem holds. Both of these are known to hold in every o-minimal expansion of the reals [23, 2]. Hence, for example, if we add exponentiation to our query language, we get:

Corollary 3. *Every* $\mathrm{FO}[+, *, <, e^x, S]$ *query that is topological over closed semi-linear (resp. semi-algebraic,* $\mathrm{FO}[+, *, <, e^x]$ *definable) sets in the plane is equivalent over semi-linear (resp. semi-algebraic,* $\mathrm{FO}[+, *, <, e^x]$ *definable) sets to a CL sentence.*

Acknowledgment

We are grateful to Bart Kuijpers and Luc Segoufin for helpful discussions.

9. REFERENCES

[1] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases.* Addison-Wesley, 1995.

[2] M. Benedikt, G. Dong, L. Libkin, and L. Wong. Relational expressive power of constraint query languages. *Journal of the ACM*, 45(1):1–34, 1998.

[3] J. Bochnak, M. Coste, and M.-F. Roy. *Real Algebraic Geometry.* Springer-Verlag, 1998.

[4] H.-D. Ebbinghaus and J. Flum. *Finite Model Theory.* Springer, 1995.

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

[6] M. Egenhofer and R. Franzosa. On the equivalence of topological relations. *Int. J. Geographical Information Systems*, 9(2):133–152, 1995.

[7] M. Egenhofer and D. Mark. Modeling conceptual neighborhoods of topological line-region relations. *Int. J. Geographical Information Systems*, 9(5):555–565, 1995.

[8] M. Grohe and L. Segoufin. On first-order topological queries. *ACM Transactions on Computational Logic*, 3(3):336–358, 2002.

[9] S. Grumbach and J. Su. Queries with arithmetical constraints. *Theoretical Computer Science*, 173(1):151–181, 1997.

[10] 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.

[11] B. Kuijpers and J. Van den Bussche. On capturing first-order topological properties of planar spatial databases. In C. Beeri and P. Buneman, editors, *Database Theory, ICDT'99*, volume 1540 of *Lecture Notes in Computer Science*, pages 187–198, 1999.

[12] G. Kuper, L. Libkin, and J. Paredaens, editors. *Constraint Databases.* Springer, 2000.

[13] R. Laurini and D. Thompson. *Fundamentals of Spatial Information Systems.* Number 37 in APIC Series. Academic Press, 1992.

[14] R. McNaughton and S. A. Papert. *Counter-Free Automata.* MIT Press, Cambridge, MA, 1971.

[15] E.E. Moise. *Geometric topology in dimensions 2 and 3*, volume 47 of *Graduate Texts in Mathematics.* Springer, 1977.

[16] M. Otto and J. Van den Bussche. First-order queries on databases embedded in an infinite structure. *Information Processing Letters*, 60:37–41, 1996.

[17] C.H. Papadimitriou, D. Suciu, and V. Vianu. Topological queries in spatial databases. *Journal of Computer and System Sciences*, 58(1):29–53, 1999.

[18] J. Paredaens and B. Kuijpers. Data models and query languages for spatial databases. *Data & Knowledge Engineering*, 25:29–53, 1998.

[19] J. Paredaens, B. Kuijpers, and J. Van den Bussche. On topological elementary equivalence of closed semi-algebraic sets in the plane. *Journal of Symbolic Logic*, 65(4):1530–1555, 2000.

[20] 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.

[21] L. Segoufin and V. Vianu. Querying spatial databases via topological invariants. *Journal of Computer and System Sciences*, 61(2):270–301, 2000.

[22] W. Thomas. Languages, automata, and logic. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Language Theory*, volume III. Springer, 1997.

[23] L. van den Dries. *Tame Topology and O-Minimal Structures.* Cambridge University Press, 1998.