On the Expressiveness of Query Languages with Linear Constraints; Capturing Desirable Spatial Properties

Foto Afrati, Theodore Andronikos, Theodore Kavalieros

National Technical University of Athens
{afrati, thandron, kavalier}@softlab.ece.ntua.gr

Abstract. We explore the expressive power of the query language of first order logic with linear constraints. Natural queries are shown to be expressible in this language in a non-trivial fashion. We further investigate the ability of the language to capture certain properties that seem desirable in spatial databases. We show that, although these properties are captured in a certain extent, there are still cases where the language has counter-intuitive behavior. In this context, we show that FO+linear cannot distinguish between databases that carry similar spatial characteristics, e.g., two different ellipses.

1 Introduction

The traditional view of databases as finite collections of data is not adequate in many important and popular current applications. There is an increasing need for database systems that can handle data from continuous domains (e.g., the real numbers), as is the case in spatial and temporal databases. In particular, spatial data is involved in numerous important current applications such as geographic information systems, medical databases, CAD-CAM, robotics etc.

While many models and languages for spatial data have been proposed, the theoretical background in the area leaves much to be desired ([PSV96], [PBG95]). There are many ad-hoc solutions for various significant problems in this area; still a profound understanding of the formal foundations and the expressive power of the miscellaneous proposals is missing in many cases. Towards this direction, a large amount of research today focuses on the constraint database model and query languages, as first considered in [KKR90]. According to the model they propose, a constraint database consists of generalized relations, whose tuples represent a possibly infinite collection of data. Moreover, interesting query languages are proposed that combine the relational calculus with various arithmetic constraints ([GST94], [GS94], [Kup90], [BDLW96]).

In this paper, we study the expressive power of the relational calculus augmented with linear constraints, or, equivalently, of the query language *FO+linear*. *FO+linear* has been recognized by many researchers in the field as an interesting sublanguage of the relational calculus with polynomial constraints, mainly because of its applicability and the potential for efficient implementation ([HJLL90]).

Several results for *FO+linear* are already known. In [VGG95], they show that queries checking boundedness, convexity and discreteness of linear databases can be expressed in *FO+linear*. This language can also express many interesting topological properties of semi-linear sets, like topological interior, topological boundary and topological closure [VGG96]. Other interesting queries that have been studied in [ACGK94] are the *in-line* query, that asks whether L is contained in a line, the *exists-line* query that asks whether L contains a line, the *lines* query that returns the set of lines contained in L, and the *line-intersection* query that returns all points P which are intersections of (pairs of distinct) lines contained in L. These queries seem to act as benchmarks in studying the expressive power of various versions of first order query languages with constraints and, in particular *FO+linear*. We know that the *in-line* query is not expressible in *FO+linear*, whereas for the *exists-line* and the *line-intersection* queries we only know that they are expressible on a very specific subset of constraint databases [ACGK94]. About the *lines* query, we only know that it is linear (i.e., if applied on a linear database it outputs a linear database).

A major issue, in this context, is to what extent *FO+linear* can express properties of databases that are easily expressible if we consider *FO+poly*, i.e., by allowing multiplication too ([GS95]). In [AAK95] it is shown that *FO+linear* can express the query "one-line", that characterizes the class of databases conisting of exactly one line.

In view of the fact that we still do not know if the query "the database contains a line" is expressible in *FO+linear*, we investigate here whether there are queries expressible in *FO+linear* which capture properties of the database that relate to defining a direction on a straight line. Thus, we show here how to express the following queries in *FO+linear*:

- Characterize the class of databases consisting of exactly c parallel lines, where c a constant.
- Characterize the class of databases that consist of exactly two lines intersecting at a point.

In proving our results, we develop tools and a general methodology that can be useful in expressing many similar queries within the framework of *FO+linear*.

Next, we investigate the ability of *FO+linear* to capture certain properties, that seem desirable in spatial databases. We prove that we cannot distinguish between two "similar" geometric figures. Our tool in order to prove this inexpressibility result is the *Ehrenfeucht-Fraissé* games, which is one of the few tools that are useful in the presence of constraints. Finally, we investigate whether the intuition that *FO+linear* can express non-linear properties in a very limited way carries on. Thus, we show that *FO+linear* cannot distinguish between databases that carry "superficial" similar characteristics like same local maxima and minima, e.g., two different ellipses, or an ellipse and an ellipse-like figure. We conclude that there is no *FO+linear* query that characterizes exactly all the ellipses.

The rest of the paper is organized as follows: In Section 2 we present some basic concepts. Section 3 is devoted to the proof of our positive results regarding the expressive power of *FO+linear* where in Section 4 we present the negative results about "similar" figures, which demonstrate the limitations of the language. Finally, in Section 5 we conclude and summarize our results.

2 Basic Concepts

Definition 1. The first order language *FO+linear* consists of all the usual first order logical symbols, plus the following symbols:

- the equality symbol $=$
- the order symbol $<$,
- the addition symbol $+$.

The above, being logical symbols, are regarded as built-in predicates, i.e., they have a fixed a priori meaning and are not open to any other interpretation.

An immediate extension of *FO+linear* is *FO+poly* which includes the multiplication symbol $\times$. The intended structures for *FO+linear* and *FO+poly* are $< R, =, <, + >$ and $< R, =, <, +, \times >$, respectively, where R is the set of reals.

The relations defined by formulas of *FO+linear* with free variables are also known as *semi-linear* and the relations defined by formulas of *FO+poly* with free variables are also known as *semi-algebraic* in the literature ([PBG95], [VGG95], [VGG96]).

Note that $\geq, \neq$ are expressed as Boolean combinations of $>, =$. Also, when writing *FO+poly* formulas we will use abbreviations such as $x < e$ (instead of $-x + e > 0$) and $L(x+1, y)$ (instead of $\exists z\{z = x + 1 \wedge L(z, y)\}$).

FO+lines is an extension of *FO+linear* with variables ranging over points and lines in R^k. Atomic formulas $L(p), L(l)$ mean the point p (resp. the line l) is contained in the database L; $p \in l$ means the point p lies on the line l. It can be seen that *FO+lines* queries can be expressed in *FO+poly*. More generally, one can consider variables of higher dimension. It can be proved that extending *FO+poly* in this way does not increase its expressive power [KKR90].

One of the most valuable tools for inexpressibility results in first-order logic is the *Ehrenfeucht-Fraissé* games.

Definition 2. The n-round *Ehrenfeucht-Fraissé* game is played between two players on two databases $\mathcal{D}, \mathcal{D}' \subseteq R^k$. At round r player I picks a point $p_r \in R$ and associates it to either $\mathcal{D}$ or $\mathcal{D}'$; player II responds by picking $q_r \in R$ and associating it to the other database.

For each r let t_r, t'_r be the points associated with $\mathcal{D}, \mathcal{D}'$ respectively; $\{t_r, t'_r\} = \{p_r, q_r\}$. Player II *wins* the game iff

(i) $t_i = t_j$ iff $t'_i = t'_j$ and
(ii) $(t_i\ , \ldots, t_{i_k}) \in \mathcal{D}$ iff $(t'_i\ , \ldots, t'_{i_k}) \in \mathcal{D}'$.
(iii) $t_i < t_j$ iff $t'_i < t'_j$ and $t_i + t_j = t_k$ iff $t'_i + t'_j = t'_k$.

The well-known theory of Ehrenfeucht-Fraissé games gives the following result:

Theorem 3. *Let Q be a property of databases. The following are equivalent:*

(a) Q is not expressible in FO+linear.
(b) For each n there exist databases $\mathcal{D}_n, \mathcal{D}'_n$ which differ wrto Q such that player II wins the n-round Ehrenfeucht-Fraissé game on $\mathcal{D}_n, \mathcal{D}'_n$.

We say that two structures are not *distinguishable* in a query language if any sentence in the language is true on one structure iff it is true in the other. Therefore, the Ehrenfeucht-Fraissé games offer a tool for proving related results.

3 *FO+linear* Can Define Direction

In this section, we extend the first order language *FO+linear* with the nonlogical binary predicate symbol L and view it as a query language. Hereafter, we consider this extended language and, conveniently, refer to it by the same name.

Before we present how to express in *FO+linear* some interesting queries that seem to demonstrate that direction-relevant properties are expressible in this language, we define, in the following section, some auxiliary predicates that we use later on as abbreviations to avoid heavy and cumbersome notation.

3.1 Predicates defined in *FO+linear*

We define the following abbreviations and predicates which we shall use in order to prove the main results of the paper.

- The abbreviation $x - y = z \equiv y + z = x$. In the rest of the paper we shall write $L(x - y, z)$, instead of $\exists w(x - y = w \wedge L(w, z))$.
- The logical symbol $\exists!$ which asserts the existence of *exactly one* element from the universe of our structure. This symbol can be defined from the usual logical symbols in the following way:
$\exists! x\phi(x) \equiv \exists x \forall y(\phi(x) \wedge (\phi(y) \rightarrow y = x))$
- The logical symbol $\exists_2!$ which asserts the existence of *exactly two* elements from the universe of our structure. This symbol can be defined from the usual logical symbols in the following way:
$\exists_2! x\phi(x) \equiv \exists x_1 \exists x_2 \forall y(x_1 \neq x_2 \wedge \phi(x_1) \wedge \phi(x_2) \wedge (\phi(y) \rightarrow y = x_1 \vee y = x_2))$
- The logical symbol $\exists_n!$ which asserts the existence of exactly n elements from the universe of our structure. This symbol can be defined from the usual logical symbols in the following way:
$\exists_n! x\phi(x) \equiv \exists x_1 \exists x_2 \ldots \exists x_n \forall y(x_1 \neq x_2 \wedge \ldots \wedge x_1 \neq x_n \wedge x_2 \neq x_3 \wedge \ldots \wedge x_2 \neq x_n \wedge \ldots \wedge x_{n-1} \neq x_n \wedge \phi(x_1) \wedge \phi(x_2) \wedge \ldots \wedge \phi(x_n) \wedge (\phi(y) \rightarrow y = x_1 \vee \ldots \vee y = x_n))$
- Consider now the following sentence of our language:
$CONT(L) \equiv \forall x \forall y(L(x, y) \rightarrow \forall \varepsilon(\varepsilon > 0 \rightarrow \exists \delta(\delta > 0 \wedge \forall x' \forall y'(x - \delta < x' \wedge x' < x + \delta \wedge L(x', y') \rightarrow y - \varepsilon < y' \wedge y' < y + \varepsilon))))$
This sentence simply asserts that the relation L is "continuous", in the usual algebraic sense.

– We define the predicate $RECT$ as follows:
$RECT(x, y, x_A, y_A, x_B, y_B) \equiv ((x_A \leq x \leq x_B) \vee (x_B \leq x \leq x_A)) \wedge ((y_A \leq y \leq y_B) \vee (y_B \leq y \leq y_A))$
Let us consider now the following formula:
$LINE\_SEGMENT(x_A, y_A, x_B, y_B) \equiv (x_A \neq x_B \vee y_A \neq y_B) \wedge L(x_A, y_A) \wedge L(x_B, y_B) \wedge \forall x \forall y (RECT(x, y, x_A, y_A, x_B, y_B) \wedge L(x, y) \rightarrow L((x+x_A)/2, (y+y_A)/2) \wedge L((x+x_B)/2, (y+y_B)/2))$

For the predicates defined above, note:

1. Intuitively, $RECT(x, y, x_A, x_B, y_A, y_B)$ asserts that a point $P = (x, y)$ of R^2 belongs to the rectangle defined by the points $A = (x_A, y_A)$ and $B = (x_B, y_B)$. Of course, there are special cases, where the rectangle degenerates to a line segment parallel to one of the coordinate axes or even to a single point.
2. Strictly speaking, we cannot write $L(x/2, y/2)$ in our language but we can think of $L(x/2, y/2)$ as an abbreviation of the formula $\exists x' \exists y' (x' + x' = x \wedge y' + y' = y \wedge L(x', y'))$.
3. If the two points $A = (x_A, y_A)$ and $B = (x_B, y_B)$ of R^2 satisfy the formula $LINE\_SEGMENT(x_A, x_B, y_A, y_B)$ on databases consisting merely of monotone curves or a single monotone two dimensional stripe, then our database has infinitely many points within the rectangle defined by A and B which lie on the line segment AB.

3.2 Two parallel lines

Consider now the following four sentences of our language:

– $Y_2 \equiv \forall x \exists_2 ! y L(x, y)$
– $X_2 \equiv \forall y \exists_2 ! x L(x, y)$
– $PL \equiv \exists \delta \forall x \forall x' \forall y \forall y' (\delta > 0 \wedge x' - x < \delta \wedge y' - y < \delta \wedge L(x, y) \wedge L(x', y') \rightarrow LINE\_SEGMENT(x, x', y, y'))$
– $CONT(L)$

We define PAR_LINES_2 to be the conjunction: $Y_2 \wedge X_2 \wedge PL \wedge CONT(L)$ (see Fig. 1).

We are now ready to present the main result of this section:

Theorem 4. *Sentence PAR_LINES_2 characterizes the class of databases consisting of exactly two parallel lines.*

These two lines are not parallel to any of the coordinate axes. The case where the database consists of exactly two lines parallel to a coordinate axis is trivial. The combined use of sentences $CONT(L)$ and PL along with trivial modifications in sentences Y_2, X_2 allows a straightforward extension to the case of c parallel lines.

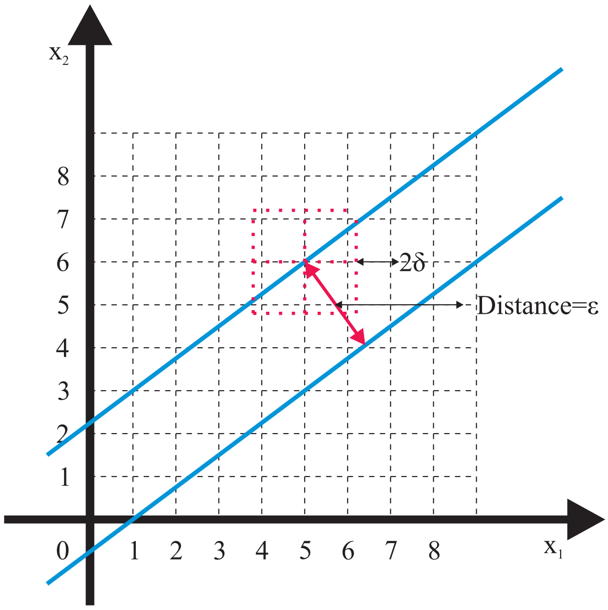

Fig. 1. Two parallel lines at distance ε.

3.3 Two intersecting lines

Consider now the following sentences of our language:

- $E_2 \equiv \exists! x_0 \exists! y_0 (L(x_0, y_0) \wedge \forall x \forall y ((x \neq x_0 \rightarrow \exists_2! y L(x, y)) \wedge (y \neq y_0 \rightarrow \exists_2! x L(x, y))))$
- $IL_2 \equiv \exists! x_0 \exists! y_0 \forall x \forall x' \forall y \forall y' (x \neq x_0 \wedge y \neq y_0 \rightarrow \exists \delta (\delta > 0 \wedge x' - x < \delta \wedge y' - y < \delta \wedge L(x, y) \wedge L(x', y') \rightarrow LINE\_SEGMENT(x, x', y, y')))$
- $CONT(L)$

Sentence E_2 expresses that, apart from the unique point of intersection (x_0, y_0), to every real $x(y)$, correspond exactly two points of the database.
We define INT_LINES_2 to be the conjunction: $E_2 \wedge IL_2 \wedge CONT(L)$ (see Fig. 2)

Theorem 5. *Sentence INT_LINES_2 characterizes the class of databases consisting of exactly two intersecting lines.*

None of these two lines is parallel to any of the coordinate axes. The case where one of the lines is parallel to a coordinate axis is trivial.

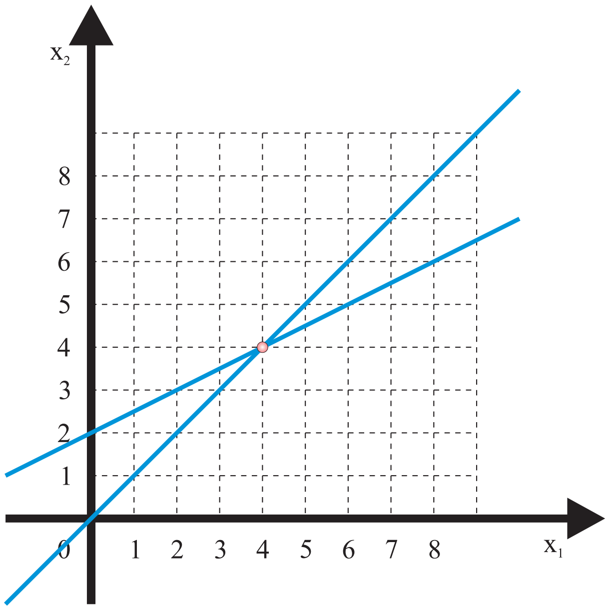

Fig. 2. Two lines intersecting at point (4,4).

4 *FO+linear* Queries on Non-linear Databases

The examples and the results in this section are expressed in terms of databases on the plane (on R^2). Most of them carry on in a straightforward way to higher dimensions.

For *FO+linear* queries, the following intuitive properties seem both desirable and expected: a) *FO+linear* queries do not distinguish between databases which differ only in that, one is a linear transformation of the other and b) *FO+linear* queries cannot capture non-linear properties of databases. In this section, we give formal proof that this intuition is true *to a certain extent*. Still there are some cases (refering to simple linear properties) where this is not true, e.g., consider the query $L(x, y) \rightarrow x = y$, which distinguishes between two lines passing through the origin.

Note the distinction between *FO+linear*, where we assume that no constants appear, and *FO+linear constraints*, where we may allow a) real constants and/or b) non-rational coefficients. The expressiveness is different. In the first, the databases $L(x, y) : x = \sqrt{2}, y = 1$ and $L(x, y) : x = 2\sqrt{2}, y = 3$ are not distinguishable (as we show below), whereas, in the latter, they are easily distinguishable.

We show:

1. Consider a database L on R^2 as a collection of points $I_L = \{(x, y) : L(x, y)\}$. Consider another database L' with $I_{L'} = \{(x, y) : L(x', y')$ and $x' = kx, y' = ky$ where k positive real$\}$. Then L and L' cannot be distinguished by *FO+linear* (see theorem 7).
2. "Most" of the point-databases (i.e., databases that consist of a single point) cannot be distinguished by *FO+linear* (see lemma 8).
3. Define an ellipse database if the points of the database are exactly the points inside the ellipse. There are ellipse databases that cannot be distinguished by *FO+linear*; moreover, *FO+linear* cannot distinguish between an ellipse database and an ellipse-like database (see lemma 9).

The result mentioned in item 3 above can be extended in a straightforward way to apply on figures with the same local minima and maxima.

Definition 6. Two figures $L_1 \subseteq R^n$ and $L_2 \subseteq R^n$ are *similar* iff there exists a bijection h: $L_1 \rightarrow L_2$ such that $h(\mathbf{x}) = k\mathbf{x}$, for some positive real k. Accordingly, we define *similar* databases (see Fig. 3).

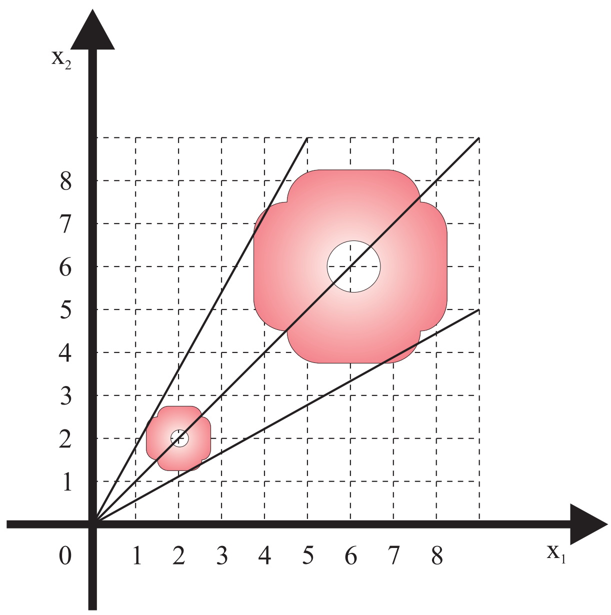

Fig. 3. Two similar figures.

FO+linear does not distinguish between similar databases. More formally, restricting our attention to the 2 dimensional case, we can prove the following:

Theorem 7. *There is no sentence in the language* FO+linear *that distinguishes between two similar databases.*

Proof. It suffices to show that the duplicator has a winning strategy if an r-round Ehrenfeucht-Fraisse game is played on two similar structures. The winning strategy is the following: If the spoiler plays point a on D_1, then the duplicator responds by playing the point $b = ka$ on D_2. If the spoiler plays point b on D_2, then the duplicator responds by playing the point $a = b/k$ on D_1.

Lemma 8. *For each point-database D_1, there is an infinite number of point-databases, such that no* FO+linear *query can distinguish between D_1 and any of them.*

Proof. Let r be a positive integer; we show that there exist two point-databases such that if the r-round Ehrenfeucht-Fraisse game is played on them, the duplicator has a winning strategy. Let the two point-databases be the following: D_1 contains the point (a_1, b_1) and D_2 contains the point (a_2, b_2) where the ratio a_i/b_i,$i = 1, 2$ is an irrational positive integer. The winning strategy of the duplicator is: in each round of the game, he considers, on D_i, the already played reals and a_i, b_i and computes all the linear combinations with coefficients integers up to r. If the real chosen by the spoiler on D_i is equal to some of these linear combinations, then the duplicator plays the corresponding linear combination on the other structure, otherwise, he simply respects the order and disallows all coresponding linear combinations on the other structure. The fact that both ratios a_i/b_i, $i = 1, 2$ are irrational guarrantees that linear combinations of critical constants (i.e., which are involved in the definition of the databases, in this case the a_i, b_i's for each database) are respected.

Lemma 9. *Let r be a positive integer. Consider an ellipse database in the plane with critical points (i.e., minima and maxima with respect to either x or y axis) the following: (a_1, b_1),(a_2, b_2),(a_3, b_3),(a_4, b_4) (see Fig. 4). There exists an integer n_0 depending only on r such that the following holds:*
Assume that all linear combinations of the a_i, b_i's with integer coefficients up to n_0 are not equal to zero. Then, there exists another ellipse or ellipse-like database such that no FO+linear *query can distinguish between them.*

Proof. **(Sketch)** Construct the other database as follows: Consider all the reals that result from the linear combinations mentioned above; consider all points on the plane that are defined by all pairs of those reals. Let δ be the minimum distance between any of these points or between one of the points and a point on the boundary of the ellipse. Choose the second database to be, a modified copy of the first where all points on the boundary are moved in the plane by $\delta' << \delta$. The latter construction, obviously can either be chosen to be an elipse or not. Now the winning strategy of the duplicator is similar to the one in lemma 8.

Corollary 10. *Properties of ellipse databases that need to distinguish between databases with different ratios of the lengths of their axes are not expressible in FO+ linear.*

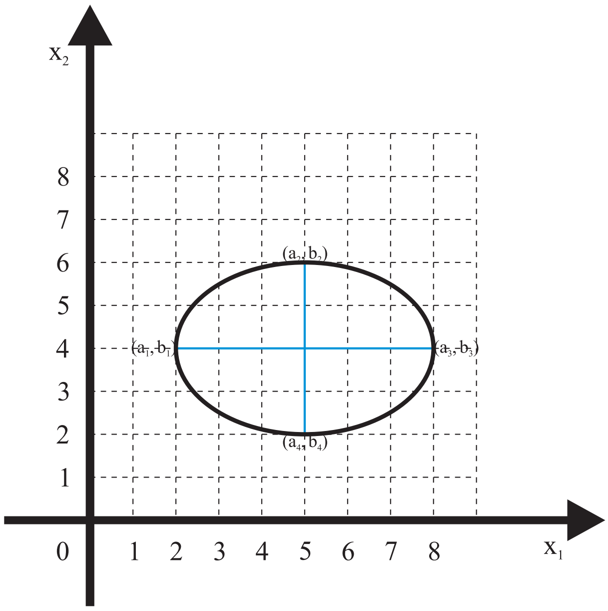

Fig. 4. The initial ellipse.

Corollary 11. *The query "Is the database an ellipse database" is not expressible in* FO+linear*.*

Note that if we restrict to rational ratios corollary 10 is not true.

5 Conclusions

We investigated the expressive power of the query language of first order logic with linear constraints. We have shown that some natural geometric queries are expressible in this language. We also demonstrated that this language "most of the times" cannot distinguish between databases with similar geometrical features. This seems to be a desirable property for a language querying spatial databases. Further research is needed to understand the necessary extensions of the language in order to capture accurately these properties.

In the same theoretical context we aim to understand the inclusion relations among sets of queries expressed by several query languages such as *FO+linear* and *FO+poly*. In that respect it is interesting to establish a hierarchy of classes of properties that corresponds to a hierarchy of query languages.

References

[AAK95] F. Afrati, T. Andronikos, T. Kavalieros. On the Expressiveness of First-Order Constraint Languages. *First CONTESSA Workshop on Constraint Databases, pp. 22-39.*

[ACGK94] F. Afrati, S. Cosmadakis, S. Grumbach, G. Kuper. Linear vs. Polynomial Constraints in Database Query Languages. *PPCP 1994.*

[BDLW96] M. Benedikt, G. Dong, L. Libkin, L. Wong Relational Expressive Power of Constraint Query Languages *PODS 1996*

[GS94] S. Grumbach and J. Su. Finitely representable databases. *PODS 1994*, pp. 289-300.

[GS95] S. Grumbach and J. Su. Dense-Order Constraint Databases. *Proc. 14th ACM PODS*, 1995

[GST94] S. Grumbach, J. Su, and C. Tollu. Linear constraint databases. *Proceedings of Logic and Computational Complexity, 1994*, pp. 426-446

[HJLL90] T. Huynh, L. Joskowicz, C. Lassez and J-L. Lassez. Reasoning About Linear Constraints Using Parametric Queries. *Foundations of Software Technology and Theoretical Computer Science.* Lecture Notes in Computer Science, Springer-Verlag vol. 472, 1990.

[KKR90] P. Kanellakis, G. Kuper and P. Revesz. Constraint Query Languages. *Proc. 9th ACM PODS*, pp. 299–313, 1990. To appear in JCSS.

[Kup90] G.M. Kuper. On the expressive power of the relational calculus with arithmetic constraints. In *Proc. Int. Conf. on Database Theory*, pp. 202–211, 1990.

[PSV96] C.H. Papadimitriou, D. Suciu, V. Vianu. Topological Queries in Spatial Databases. *PODS'96*

[PBG95] J. Paredaens, J. Van den Bussche, D. Van Gucht. First-order queries on finite structures over the reals.*LICS '95*, pp. 79-87.

[VGG95] L. Vandeurzen, M. Gyssens, D. Van Gucht. On the Desirability and Limitations of Linear Spatial Query Languages. em Proceedings of 4th Symposium on Advances in Spatial Databases, 1995.

[VGG96] L. Vandeurzen, M. Gyssens, D. Van Gucht. On Query Languages for Linear Queries Definable with Polynomial Constraints *PPCP'96*