DIMACS Technical Report 95-56
December 1995

Relational expressive power of local generic queries

by

Oleg V. Belegradek
Department of Mathematics
Kemerovo State University
Kemerovo, Russia 650000
`beleg@kaskad.kemerovo.su`

Alexei P. Stolboushkin[1]
Department of Mathematics
UCLA
Los Angeles, CA 90024-1555
`aps@math.ucla.edu`

Michael A. Taitslin[2]
Department of Computer Science
Tver State University
Tver, Russia 170000
`mat@mat.tvegu.tver.su`

[1] This work has been partially supported by NSF Grant CCR 9403809.
[2] A part of this research was carried out while the author was visiting UCLA and supported in part by NSF Grant CCR 9403809 and while the author was visiting DIMACS and Princeton and supported in part by Grant of Princeton University.

ABSTRACT

Consider a scheme of databases Q and two signatures: $L_0 = \{<\}$, $L = \{<\} \cup \Omega$, where Ω is a finite relational signature. For a database scheme $SC = \{R_1, \dots, R_n\}$, denote $L_0^+ = L_0 \cup SC$ and $L^+ = L \cup SC$. FO is the first-order language. FO in L_0^+ is called restricted. FO in L^+ is called extended. Consider a countable universe U in L. Assume CH. It is possible to reconstruct the proofs of corollaries into ones not using CH. Let V be a saturated elementary extension of U in power $\aleph_1$. There is the only such V up to elementary isomorphisms over U.

Th.1. An extended query Φ is equal for all the finite database states over U to a restricted query iff Φ is generic for all the pseudo-finite database states over V.

Th.2. If U is o-minimal, every locally generic for all the finite database states over U extended query is generic for all the pseudo-finite database states over V.

Divisible ordered Abelian groups are o-minimal structures. So, for example, groups of rational and real numbers are o-minimal structures. But the additive group of the integer numbers and the additive semigroup of the natural numbers are not.

Th.3. If U is the additive group of the integer numbers or is the additive semigroup of the natural numbers, every locally generic for all the finite database states over U extended query is generic for all the pseudo-finite database states over V.

Corollary 1. ([1]) If U is o-minimal, every locally generic for all the finite database states over U extended query is equal for all the finite database states over U to a restricted query.

Corollary 2. If U is the additive group of the integer numbers or the additive semigroup of the natural numbers, every locally generic for all the finite database states over U extended query is equal for all the finite database states over U to a restricted query.

We propose a notion of local genericity for extended queries over finitely representable order database states. We show that every finitely representable order database state can be represented by a finite state (of another scheme) such that these two states are uniformly FO-translatable to one another. Using this result, we show that, for any divisible ordered Abelian group, the additive group of the integer numbers, the additive semigroup of the natural numbers, and any o-minimal structure, every locally generic extended query over finitely representable order database states can be translated into a restricted query. Over all rational database states, however, these two query languages differ.

Introduction

Although relational databases were invented for finite collections of data, it is often convenient to assume that there is an infinite *domain*—for example, integer or rational numbers or strings—such that the data elements are chosen from this domain. Functions and relations defined over the entire domain, like $<$ and $+$, may also be used in querying, for example, if the language of first-order logic FO is used as the query language, its formulas may use database relations as well as the domain relations, while variables range over the entire domain.

Consider two signatures $L_0 = \{<\}$, and $L = \{<\} \cup \Omega$. For a database scheme $SC = \{R_1, \ldots, R_n\}$, denote $L_0^+ = L_0 \cup SC$ and $L^+ = L \cup SC$. FO in L_0^+ is called restricted. FO in L^+ is called extended.

The simplest way is to use only restricted queries. But the expressive power of restricted queries in many domains is, however, too restricted, as it fails to express many practically important queries (cf. [8, 6]). There are several ways to increase the expressive power of queries. One of them is to add additional domain functions or relations.

For example, in [5, 4] extended queries over the domain $\mathbf{Q}$ of rational numbers together with the relations $=$ of equality and $<$ of the rational linear order and the usual operation $+$ were considered. Clearly, $x+x=x$ defines 0, which is not definable without $+$, on the other hand, queries like $x+x=x$ are not about order at all, as they are not preservable under order automorphism. To level the playing field, we consider *locally generic* extended queries only, that is the queries preservable under partial $<$-isomorphisms of the universe. To put it another way, while computations can be carried out inside the query, the input-output relation defined by this query may only rely on the order properties of its input. Attempts to distinguish restricted queries and extended ones for the domain using specific combinatorial or spatial queries not in AC^0—like parity, Eulerian traversal, or region connectivity—has been unsuccessful, and, finally, Grumbach et al. [5, 4] proved the AC^0 uniform data complexity for extended queries over finitely representable inputs *with integer order constraints.* However, the question of whether or not extended queries are more expressive than restricted ones, has remained open, and as Grumbach and Su [3] pointed out, "…there is a serious lack of proof techniques…" in this area.

At the same time, the special case of finite (rather than finitely representable) states and Boolean queries has been the focus of a series of recent papers, including [10, 9, 1].

Our paper is devoted to studying of translation of extended locally generic queries into restricted ones. Of course, there are examples, when an extended locally generic query is not translated over all the finite database states into any restricted one. The natural number with $<, +, *$ gives such an example. So we are going to find conditions which guarantee the translation.

First we study possibility of the translation for finite states only. We propose a general condition and give examples of using our techniques. Some examples are not new. Similar results were recently proved by Paradaens et al. [10] (Theorem 5.1) by a different method for the additive group of the real numbers and by Benedikt et al. [1] for o-minimal domains.

Our approach uses the idea from [1]. But our presentation is more elementary and more simple. Also we admit non-Boolean queries.

Then we show that finitely representable order database states can indeed be represented by finite states with two-way FO translations (Theorem 2.13). The technique in this part works for an arbitrary linear order. This result, in our view, is interesting independently on the other results of this paper. Together with Theorem 2.5 and Theorem 2.8, it implies our main result. Our reduction can also be used to expand applicability of numerous results for finite database states to the case of finitely representable states. Space limits imposed on this abstract, however, prevent us from discussing such applications any further.

1 Basic Notions

We suppose the reader has a small background in Model theory and Algebra. Information about ultra-products and saturation is in [2].

A signature is a list of names. Each name has the arity or the number of argument places. The arity is finite for each name. Consider a fixed finite relational signature Ω. Let $L_0 = \{<\}$, $L = \{<\} \cup \Omega$. A structure of the signature L is a non-empty set with a mapping which assigns a relation of the same arity over the set to every relation name from L. Fix such a countable structure U. The structure is called the universe. But sometimes U will also denote the set of the elements of the universe. A set of one of the forms:

$$\{x : x < a\}, \{x : x \leq a\}, \{x : a < x\}, \{x : a \leq x\}$$

or

$$\{x : b < x < a\}, \{x : b < x \leq a\}, \{x : b \leq x < a\}, \{x : b \leq x \leq a\}$$

where a and b are constants is called an interval. U is called o-minimal if every subset of U that is definable with parameters in U is a finite union of intervals.

A *database scheme* is a finite collection of relation names of fixed arities. A *database state* (over U) is an assignment to these relation symbols of concrete relations of corresponding arities over U. These relations are called database relations. A database state is *finite* iff all the relations are finite. We fix a database scheme SC and denote $L_0^+ = L_0 \cup SC$ and $L^+ = L \cup SC$.

Fix a non-principal ultra-filter D over the set ω of all the natural numbers. Let U^D be the ultra-power of U by D. We identify U with its image in U^D under the diagonal embedding; so $U \subset U^D$. We call an n-ary relation on U^D *pseudo-finite* if it is a D-ultra-product of a family of n-ary finite relations on U. Let V be U^D. V is a countable-saturated structure of signature L. If elements $a_1, \ldots, a_m$ are ultra-products of

$$a_1(i)\ (i \in \omega), \ldots, a_m(i)\ (i \in \omega),$$

and relations $P_1, \ldots, P_s$ are ultra-products of

$$P_1(i)\ (i \in \omega), \ldots, P_s(i)\ (i \in \omega)$$

then

$$(V, a_1, \ldots, a_m, P_1, \ldots, P_s) \models \Phi$$

if and only if

$$\{i \in \omega : (U, a_1(i), \ldots, a_m(i), P_1(i), \ldots, P_s(i)) \models \Phi\} \in D.$$

This is the Łoś Theorem (see [2]).

A *database query* can formally be defined as a mapping that takes in a database state (of a fixed d.b. scheme), and produces a new relation, of a fixed arity, over U. Thus, every query has an arity. Specifically, queries of arity 0 are called *Boolean.* A Boolean query defines a mapping from d.b. states to $\{0,1\}$, or, in other words, subsets of all possible d.b. states of a given d.b. scheme.

Queries are formulated using *query languages,* the simplest of which is the language of first-order logic FO. Formulas (=queries) of this language use $=$ and $<$, as well as the relation names of the signature and of the database scheme. Thus, a d.b. state essentially defines a structure of a larger signature with U as the domain; then a formula with n free variables defines an n-ary relation over U.

Generally, a FO query may yield an infinite answer even in a finite d.b. state. [7] introduced the notion of *finitely representable* database state as a database state where every relation corresponding to a relation name from SC is defined (independently on the others) by a quantifier-free formula using the equality symbol, the name of the order relation, and names for elements of U.

We will consider two languages for querying. Queries of the first one will be FO formulas of signature L_0^+. We will called them restricted. Queries of the second language will be FO formulas of the signature L^+. We will called them extended.

It is easy to see that the restricted queries are *generic* w.r.t. order-preserving bijections of U. In other words, if $\phi : U \longleftrightarrow U$ is an automorphism of $\langle U, <, = \rangle$, and a restricted query Q transforms a database state s into a relation R, then Q transforms $\phi(s)$ into $\phi(R)$. The problem with extended queries is, they may not be generic.

We also need consider a more strong notion of *locally generic* queries.

For a given database state, its *active domain* AD is the set of all the elements of the domain U that occur in some tuple in some of the database relations. AD can be expressed by a query that only uses the database scheme relation names (and does not use either $<$ or names from Ω). The active domain of a finite d.b. state is always finite.

Let A be a subset of U. A mapping ϕ from A in U is monotone if $\phi(x) < \phi(y)$ for any x and y such that $x, y \in A$ and $x < y$. A query Q of the arity k is locally generic over finite states iff for any partial monotone mapping ϕ from the union of AD and $\{a_1, \ldots, a_k\}$ in U, if query Q transforms a database state s into a relation R and $R(a_1, \ldots, a_k)$ is true then Q transforms $\phi(s)$ into such a relation S that $S(\phi(a_1), \ldots, \phi(a_k))$ is true. The problem is that a restricted query may not be locally generic.

An expression of one of the forms $x < c$, $x > c$, or $x = c$ where x is a variable and c is a name of an element of U is called simple. A relation R is said to be *simple* if it can be finitely represented as a conjunction of disjunctions of simple formulas. R is said to be *prime* if it is representable in the form

$$\bigwedge_{i<j} x_i < x_j \wedge \gamma,$$

where γ represents a simple relation. A finitely-representable database state is said to be simple (prime) if every database relation in this state is simple (prime).

Given a prime relation P of arity k, we say that two elements r_1, r_2 of U are *P-separable* iff there exists a position j in the relation P such that:

$$(\exists x_1 x_2 \ldots x_{j-1} x_{j+1} \ldots x_k)(\bigwedge_{i=1}^{j-2} x_i < x_{i+1} \wedge \bigwedge_{i=j+1}^{k-1} x_i < x_{i+1} \wedge$$

$$x_{j-1} < r_1 < x_{j+1} \wedge x_{j-1} < r_2 < x_{j+1} \wedge$$

$$(P(x_1 x_2 \ldots x_{j-1} r_1 x_{j+1} \ldots x_k) \not\equiv P(x_1 x_2 \ldots x_{j-1} r_2 x_{j+1} \ldots x_k))).$$

Hence, this relation of separability is expressible as a restricted query.

An element x of U is said to be a *left boundary point* of a prime relation P iff for each $y \in U,\ y > x$ there exist P-separable $u, v \in U$ such that $x \leq u \leq y,\ x \leq v \leq y$. An element x of U is said to be a *right boundary point* of a prime relation P iff for each $y \in U,\ y < x$ there exist P-separable $u, v \in U$ such that $x \geq u \geq y,\ x \geq v \geq y$. An element is a *boundary point* of P iff it is a right or left boundary point of P. It can be observed that any boundary point of P is a constant in definition of P or has such a constant as his adjacent. So the set of the boundary points for each prime relation P is finite and can be expressed as an restricted query. Denote this query by $\Gamma_P(x)$. If a constant is not a boundary point of P, it is possible to define P not using the constant. So we will suppose that any constant in a definition of P is a boundary point of P.

Every quantifier-free formula representing a relation P of arity k can always be represented as a disjunction of the following form:

$$\bigvee_{\rho \in R_k} (\rho \wedge \delta_\rho),$$

where R_k is a collection of all the maximal conjunctions of equalities and comparisons between $x_1, x_2, \ldots, x_k$, and δ_ρ is a quantifier-free formula in which inter-variable comparisons or equalities are not used. Then for every $\rho \in R_k$, $\rho \wedge P$ can be easily encoded by a prime relation (of possibly a smaller arity). The union of the sets of the boundary points of the prime relations $\rho \wedge P$ for all $\rho \in R_k$ is the set of the boundary points of P. For a finite representable state, the set of the boundary points of the state is the union of the sets of the boundary points of all the database relations of the state.

A query Q is locally generic over finitely representable states iff for any partial monotone mapping ϕ from the union of the set of the boundary points of a finitely representable state s and $\{a_1, \ldots, a_k\}$ in U, if query Q transforms a database state s into a relation R and $R(a_1, \ldots, a_k)$ is true then Q transforms $\phi(s)$ into such a relation S that $S(\phi(a_1), \ldots, \phi(a_k))$

is true. A definition of the state $\phi(s)$ is obtained from a definition of s by replacing every constant c in the definition with $\phi(c)$. The problem is that a restricted query may not be locally generic for finitely representable states.

2 RESTRICTED vs. EXTENDED

Let $\mathbf{Q}$ be the set of the rational numbers. The goal of this section is to compare restricted and generic extended queries from the viewpoint of their expressive power over $\langle \mathbf{Q}, <, = \rangle$. We start from showing that, generally speaking, $+$ does add to the expressive power of FO query language, even for Boolean queries:

Theorem 2.1 *The finiteness of the database state is expressible by an extended query but is not expressible by any restricted query.*

PROOF: The fact that the query is generic is obvious. To express it, first recall that the active domain AD of the database state (that is, the elements present in the database state) is expressible by an FO query. Let $AD(x)$ be this query. The d.b. state is finite iff AD is finite. Observe now that a set of rational numbers is finite iff the following two conditions hold:

1. the set fully belongs to an interval, and
2. there is a rational quantity $r > 0$ such that any two distinct elements in the set are at least r apart from one another.

Therefore, we can express this query as:

$$(\exists ulr > 0)((\forall x)(AD(x) \longrightarrow (l < x \wedge x < u)) \wedge (\forall xy)((AD(x) \wedge AD(y) \wedge x < y) \longrightarrow x + r < y)).$$

Here, the "$x + r < y$" is an obvious abbreviation.

Of course the boundedness of AD is expressed in the first part of the formula without $+$; we claim, however that the constant-separation is not expressible without $+$.

Formally, we claim that if a Boolean restricted query holds in every finite database state, it also holds in an infinite state that belongs to an interval and consists of isolated points. A formal proof of this can be carried out for a database scheme consisting of a single unary relation by applying a Fraïssé-Taimanov-Ehrenfeucht game. This part is omitted from the current abstract. *Q.E.D.*

Note that the technique in this theorem is $\mathbf{Q}$-specific, and cannot be carried out for, say, real numbers.

Corollary 2.2 *Extended querying is more expressible than restricted one w.r.t. Boolean generic queries.*

2.1 Locally generic extended queries over finite d.b. states

Theorem 2.3 *An extended query Φ of arity k is equal for all the finite database states over U to a restricted query iff Φ is generic for all the pseudo-finite database states over V.*

PROOF: If Φ is equal for all the finite database states over U to a restricted query Ψ, Φ is equal to Ψ for all the pseudo-finite states over V. But Ψ and all the other restricted queries are stable under order automorphisms of V.

Let Φ is not equal over finite states to any restricted query. Let ω is the set of all the natural numbers. Let $(\phi_n : n < \omega)$ be an enumeration of all L_0^+-formulas, that is, all $<$-queries. For every $n < \omega$ there are tuples a_n and b_n of elements of U and finite states s_n and r_n over U which agree on $\phi_0, \dots, \phi_n$ for a_n and b_n (that means that for $j = 0, \dots, n$

$$(U, s_n, a_n) \models \phi_j \quad iff \quad (U, r_n, b_n) \models \phi_j)$$

but

$$(U, s_n, a_n) \models \Phi \ \ and \ \ (U, r_n, b_n) \models \neg\Phi.$$

Let s be the ultra-product of s_n $(n < \omega)$ and r be the ultra-product of r_n $(n < \omega)$. Let a be the ultra-product of a_n $(n < \omega)$. Let b be the ultra-product of b_n $(n < \omega)$. It is obvious that (V, s, a) is elementary equivalent to (V, r, b) as a structure of the signature $L_0^+, c_1, \dots, c_k$. Since V is $\aleph_1$-saturated there is a $<$-automorphism of V transforming a in b and s in r. But Φ is true in (V, s, a) and is false in (V, r, b). So Φ is not generic for pseudo-finite states over V.

Q.E.D.

Let $AD(s)$ be the active domain of a state s. If a is a tuple of elements V and s is a state, let $\{AD(s), a\}$ be the union of the elements of $AD(s)$ and the elements of a.

Lemma 2.4 *Let an extended query Φ of arity k is locally generic for all the finite states over U and is not generic for all the pseudo-finite states over V. There are pseudo-finite states s and r over V, k-tuples a and b of elements V, and L-indiscernible sequence Σ of elements of V such that*

1. *$\{AD(s), a\}$ and $\{AD(r), b\}$ are subsets of Σ and there is L_0^+-isomorphism h from $\{AD(s), a\}$ onto $\{AD(r), b\}$*
2. *$(V, s, a) \models \Phi$ and $(V, r, b) \models \neg\Phi$*

PROOF: Let s be the ultra-product of s_n $(n < \omega)$ and a be the ultra-product of a_n $(n < \omega)$. Let $(\theta_n : n < \omega)$ be an enumeration of all L-formulas. For each $n \in \omega$, there is an infinite $(\theta_1, \dots, \theta_n)$-indiscernible sequence of elements U. Using locally genericity of Φ we can suppose that for all $n \in \omega$, $\{AD(s_n), a_n\}$ is a subset of the sequence. *Q.E.D.*

Theorem 2.5 *If U is o-minimal, every locally generic for all the finite database states over U extended query is generic for all the pseudo-finite database states over V. So (see [1]) if U is o-minimal, every locally generic for all the finite database states over U extended query is equal for all the finite database states over U to a restricted query.*

PROOF: Consider the map h from Lemma 2.4. As Σ is an L-indiscernible sequence, the map h is, in fact, an L-elementary partial isomorphism from $\{AD(s), a\}$ onto $\{AD(r), b\}$.

We will show below that h is even L^+-elementary, that is, h is an elementary partial isomorphism from $\{AD(s), a\}$ onto $\{AD(r), b\}$. This will contradict to (2) of Lemma 2.4.

Since h is a partial isomorphism from $\{AD(s), a\}$ onto $\{AD(r), b\}$ then, to prove that h is elementary as a map from $\{AD(s), a\}$ onto $\{AD(r), b\}$, we show how, beginning with h, to win the Fraïssé-Taimanov-Ehrenfeucht game of the length ω. It suffices to prove the following lemma.

Lemma 2.6 *Let A and B be pseudo-finite sets in U^D, and $g : A \to B$ be an L-elementary bijection. Then, for any $a \in U^D$, there is $b \in U^D$ such that $s \cup \{(a, b)\}$ is an L-elementary bijection.*

PROOF: Lemma follows from the $\aleph_1$-saturation of V and the following fact.

Lemma 2.7 *Let A be a pseudo-finite set in V. There is a countable $A_0 \subseteq A$ such that* $\mathrm{tp}(a/A_0)$ *implies* $\mathrm{tp}(a/A)$.

PROOF: The proof of the latter lemma uses the o-minimality of U. We begin with two observations.

(1) It is easy to see that, for any set P definable in an o-minimal structure with a set of parameters C, every boundary point of P is C-definable. The reason is that the set of boundary points of P is finite and C-definable.

(2) Let F be a finite set in U, $d \in U$, and $\chi(x, \bar{y})$ an L-formula. Then among the boundary points of the sets of the form $\{u \in U : U \models \chi(u, \bar{c}) \wedge u \leq d\}$, where $\bar{c}$ runs over F, there is the greatest one. Analogously, among the boundary points of the sets of the form $\{u \in U : U \models \chi(u, \bar{c}) \wedge d \leq u\}$ where $\bar{c}$ runs over F, there is the least one. Therefore, the same is true for pseudo-finite sets F in V and $d \in V$. Denote the two elements in V by $m(F, d, \chi)$ and $M(F, d, \chi)$.

Q.E.D.

Now a proof of Lemma 2.6. If a is A-definable the result is obvious. Suppose it is not the case. Then, for every $\chi(x, \bar{c})$ in $\mathrm{tp}(a/A)$, a is an internal point of $\chi(\mathcal{U}, \bar{c})$. Hence the interval $m(A, a, \chi) < u < M(A, a, \chi)$ is contained in $\chi(\mathcal{U}, \bar{c})$. As the language is countable, there is a countable $A_0 \subseteq A$ such that all $m(A, a, \chi)$, $M(A, a, \chi)$ are definable over A_0, and we are done.

Q.E.D.

Q.E.D.

Theorem 2.8 *If U is the additive group of the integer numbers or is the additive semigroup of the natural numbers, every locally generic for all the finite database states over U extended query is generic for all the pseudo-finite database states over V. So if U is the additive group of the integer numbers or the additive semigroup of the natural numbers, every locally generic for all the finite database states over U extended query is equal for all the finite database states over U to a restricted query.*

PROOF: The proof is similar because the same Lemma 2.7 can be used. *Q.E.D.*

2.2 Locally generic extended queries over finitely representable d.b. states

Let R be a set of elements of U. A relation P is said to be *R-invariant* iff:

$$(\forall \vec{x}\vec{y}) \left((\forall z \in R) \left(\bigwedge_i (x_i > z \equiv y_i > z \wedge x_i < z \equiv y_i < z) \right) \longrightarrow (P(\vec{x}) \equiv P(\vec{y})) \right).$$

The following lemma is then obvious.

Lemma 2.9 *A relation P is simple iff there exists a finite set R of elements of U such that P is R-invariant.*

An interval $[x, y]$ is a set $\{z : x \leq z \leq y\}$.

Lemma 2.10 *Let P be a prime relation. If r, q are P-separable and $r < q$, then the interval $[r, q]$ contains a boundary point of P.*

Let γ be a formula, P be a relation symbol, and let δ be a formula of the same arity as P. By $\gamma \left[\frac{P}{\delta}\right]$ we denote the result of substitution of δ for all occurrences of P in γ. As usual, this may require renaming of variables.

Lemma 2.11 *For any database scheme σ there exist another database scheme κ, $\sigma \bigcap \kappa = \emptyset$, and* FO *formulas $\{\delta_P\}_{P\in\sigma}$ of the scheme κ, and $\{\delta_P\}_{P\in\kappa}$ of the scheme σ, δ_P of the arity of P, such that for any finitely representable σ-state for any $P \in \sigma$:*

1. *δ_P yields a simple answer*
2. *for any $P \in \sigma$, $(\forall \vec{x}) \left(\delta_P \left[\frac{Q}{\delta_Q}\right]_{Q\in\kappa} (\vec{x}) \equiv P(\vec{x}) \right)$ holds in every finitely representable state.*

PROOF: It suffices to show (2) for prime database states only. Indeed, every quantifier-free formula representing a relation P of arity k can always be represented as a disjunction of the following form:

$$\bigvee_{\rho \in R_k} (\rho \wedge \delta_\rho),$$

where R_k is a collection of all the maximal conjunctions of equalities and comparisons between $x_1, x_2, \ldots, x_k$, and δ_ρ is a quantifier-free formula in which inter-variable comparisons or equalities are not used. Then for every $\rho \in R_k$, $\rho \wedge P$ can be easily encoded by a prime relation (of possibly a smaller arity).

For every $P \in \sigma$, include the relation symbol P_* of the same arity k into κ. Define $\delta_{P_*}(x_1, x_2, \ldots, x_k)$ as follows:

$$(\exists y_1 y_2 \ldots y_k)\left(\left((\forall z)\left(\Gamma_P(z) \longrightarrow \bigwedge_{i=1}^{k}((x_i > z \equiv y_i > z) \wedge (x_i < z \equiv y_i < z))\right) \wedge P(y_1, y_2, \ldots, y_k)\right)\right.$$

Essentially, $\delta_{P_*}(x_1, x_2, \ldots, x_k)$ extends $P(x_1, x_2, \ldots, x_k)$ to a relation invariant w.r.t. the set of boundary points of P. Denote $x_1 < x_2 < \ldots < x_k$ by ρ_P. Obviously, answers to $\delta_{P_*}(x_1, x_2, \ldots, x_k)$ will not necessarily satisfy ρ_P.

Since the set of boundary points of a prime relation is finite (Lemma 2.10), and $\delta_{P_*}(x_1, x_2, \ldots, x_k)$ yields an invariant relation w.r.t. the set of boundary points of P, this answer is simple (Lemma 2.9).

Notice that $\rho_P \wedge \delta_{P_*}(x_1, x_2, \ldots, x_k)$ is equivalent to $P(x_1, x_2, \ldots, x_k)$. Clearly, P implies $\rho_P \wedge \delta_{P_*}$. Let $\rho_P \wedge \delta_{P_*}(a_1, a_2, \ldots, a_k)$ and $\neg P(a_1, a_2, \ldots, a_k)$. Then there exist $b_1, b_2, \ldots, b_k$, and $i \leq k$ such that:

1. $a_1, a_2, \ldots, a_k$ and $b_1, b_2, \ldots, b_k$ are positioned the same way w.r.t. the set of boundary points of P
2. for all $j = 1, 2, \ldots, k-1$, $a_j < a_{j+1}$ and $b_j < b_{j+1}$
3. $P(b_1, b_2, \ldots, b_k)$, but $\neg P(a_1, a_2, \ldots, a_k)$
4. $a_1 = b_1, \ldots, a_{i-1} = b_{i-1}$, and $a_i < b_i$ (the case $a_i > b_i$ is similar)

If a_i, b_i were P-separable then by Lemma 2.10 there would be a boundary point between a_i and b_i—a contradiction. Otherwise $a_1, a_2, \ldots, a_k$ and $a_1, a_2, \ldots, a_i, b_{i+1}, \ldots, b_k$ satisfy the same four conditions as above for $i + 1$.

Hence, we can define δ_P as $\rho \wedge P_*$. This proves the lemma. *Q.E.D.*

For a simple relation P of an arity k, define the relation $x \not\simeq_P y$ of P-separability as follows:

$$(\exists i < k)(\exists x_1 x_2 \ldots x_{i-1} x_{i+1} \ldots x_k)$$
$$(P(x_1, x_2, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_k) \not\equiv P(x_1, x_2, \ldots, x_{i-1}, y, x_{i+1}, \ldots, x_k)).$$

Lemma 2.12 *For any database scheme κ there exist another database scheme η, $\kappa \bigcap \eta = \emptyset$, and* FO *formulas $\{\delta_P\}_{P \in \kappa}$ of the scheme η, and $\{\delta_P\}_{P \in \eta}$ of the scheme κ, δ_P of the arity of P, such that for any simple κ-state for any $P \in \kappa$:*

1. *δ_P yields a finite answer*

2. for any $P \in \kappa$, $(\forall \vec{x}) \left(\delta_P \left[\frac{Q}{\delta_Q}\right]_{Q\in\eta} (\vec{x}) \equiv P(\vec{x}) \right)$ *holds for every simple state*

Proof: Take any relation symbol $P \in \kappa$. By assumption, a finite representation for P is a quantifier-free first order formula whose atomic formulas are either equalities or comparisons between variables on one side and constants on the other. Consider the (finite!) set B of boundary points of P. The formula $\Theta(x)$ that would express this set is easy to write using $\not\simeq_P$. Clearly, P is B-invariant, and therefore, the behavior of P can be explained in terms of B. Formally, we will introduce 3^k new relation symbols $\{P_g\}$ of the same arity as P indexed with sequences of $\{<, >, =\}$ of length k. These new relations are going to be defined as subset of B^k, and will therefore be finite.

$<$ in ith position in the index indicates that the boundary point in this position the entire interval of inseparable elements immediately preceding the point. Similarly, $>$ indicates the interval following this point, and $=$ indicates the point itself. For example, for a ternary P, $\delta_{P_{=<=}}(x, y, z)$ can be written as follows:

$$\Theta(x) \wedge \Theta(y) \wedge \Theta(z) \wedge$$

$$(\forall u)\left((u < y \wedge (\forall v)(\Theta(v) \wedge v < y \longrightarrow v < u)) \longrightarrow P(x, u, z)\right).$$

In the opposite direction, P can be reconstructed using all $\{P_g\}$ and their active domain. Technical details are left to the reader. *Q.E.D.*

The following main theorem follows from Lemmas 2.11, 2.12:

Theorem 2.13 *For any database scheme* σ *there exist another database scheme* η, $\sigma \bigcap \eta = \emptyset$, *and* FO *formulas* $\{\delta_P\}_{P\in\sigma}$ *of the scheme* η, *and* $\{\delta_P\}_{P\in\eta}$ *of the scheme* σ, δ_P *of the arity of* P, *such that for any finitely representable* σ*-state for any* $P \in \eta$*:*

1. δ_P *yields a finite answer*

2. for any $P \in \sigma$, $(\forall \vec{x}) \left(\delta_P \left[\frac{Q}{\delta_Q}\right]_{Q\in\eta} (\vec{x}) \equiv P(\vec{x}) \right)$ *holds for every finitely representable state*

Theorem 2.14 *Let* U *be an o-minimal structure, the additive group of the integer numbers, or the additive semigroup of the natural numbers. For example,* U *can be a divisible ordered Abelian group. So, for example,* U *can be a group of rational or real numbers. Let* $\phi(\vec{x})$ *be a locally generic over finitely representable states extended query. There exists an restricted query* $\psi(\vec{x})$ *that is equivalent to* $\phi(\vec{x})$ *over finitely representable database states.*

Proof: Using Theorem 2.11, rewrite the query $\phi(\vec{x})$ to a query $\phi'(\vec{x})$ over a new signature η, such that:

- there are formulas $\{\delta_P\}_{P\in\eta}$ over the initial signature σ with the property $\phi(\vec{x}) \iff \phi' \left[\frac{P}{\delta_P}\right]_{P\in\eta} (\vec{x})$, and
- for any $P \in \eta$, δ_P yields a finite answer

By Theorem 2.5 and Theorem 2.8, ϕ' is equivalent to a restricted formula ψ' over all finite d.b. states. Then, $\psi'\left[\frac{P}{\delta_P}\right]_{P\in\eta}(\vec{x})$ is a restricted formula equivalent to $\phi(\vec{x})$. Denote this formula by ψ.

Q.E.D.

References

[1] M. Benedikt, G. Dong, L. Libkin, and L. Wong. Relational expressive power of constraint query languages. Manuscript, 1995.

[2] C. C. Chang and H. J. Keisler. Model theory. North Holland, 1990.

[3] S. Grumbach and J. Su. Finitely representable databases. In *Proc. 13th ACM Symp. on Principles of Database Systems*, 1994.

[4] S. Grumbach and J. Su. Dense-order constraint databases. In *Proc. 14th ACM Symp. on Principles of Database Systems*, pages 66–77, 1995.

[5] S. Grumbach, J. Su, and C. Tollu. Linear constraint databases. In *Proc. Logic and Computational Complexity (LCC'94), Springer-Verlag Lecture Notes in Computer Science*, 1995. To appear.

[6] P.C. Kanellakis and D.Q. Goldin. Constraint programming and database query languages. In *Proc. International Symposium on Theoretical Aspects of Computer Software (TACS'94)*, pages 96–120, 1994.

[7] P.C. Kanellakis, G.M. Kuper, and P.Z. Revesz. Constraint query languages. In *Proc. 9th ACM Symp. on Principles of Database Systems*, pages 299–313, 1990.

[8] 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.

[9] M. Otto and J. Van den Bussche. First-order queries on databases embedded in an infinite structure. Manuscript, 1995.

[10] J. Paradaens, J. Van den Bussche, and D. Van Gucht. First-order queries on finite structures over reals. In *Proc. 10th IEEE Symp. on Logic in Computer Science*, pages 79–87. IEEE Computer Society Press, 1995.