# Relational Queries Computable in Polynomial Time

Neil Immerman*
Computer Science Department
Yale University
New Haven, CT 06520

*Information and Control* 68 (1986), 86-104.

**Abstract**

We characterize the polynomial time computable queries as those expressible in relational calculus plus a least fixed point operator and a total ordering on the universe. We also show that even without the ordering one application of fixed point suffices to express any query expressible with several alternations of fixed point and negation. This proves that the fixed point query hierarchy suggested by Chandra and Harel collapses at the first fixed point level. It is also a general result showing that in finite model theory one application of fixed point suffices.

## Introduction and Summary

Query languages for relational databases have received considerable attention. In 1972 Codd showed that two natural languages for queries – one algebraic and the other a version of first order predicate calculus – have identical powers of expressibility, [Cod72]. Query languages which are as expressive as Codd's Relational Calculus are sometimes called *complete*. This term is misleading however because many interesting queries are not expressible in "complete" languages.

In 1979, Aho and Ullman, [AU79] noted that relational calculus does not suffice to express the transitive closure property. They suggested adding a least fixed point operator to relational calculus in order to create a query language which can express transitive closure. In 1980, Chandra and Harel, [CH80b], studied the expressive power of relational calculus with added primitives such

*Research supported by NSF Grant MCS81-05754 and by an NSF postdoctoral fellowship. A preliminary version of this paper appeared in *14th ACM STOC Symp.* (1982), 147-152.

as a least fixed point operator. They define a Fixed Point Hierarchy of query classes, the queries in each particular class being those expressible with a certain number of applications of the least fixed point operator, followed by a certain number of alternations of of ordinary quantification and negation. In this paper we show:

**Theorem 2.** *The Fixed Point Hierarchy collapses at the first fixed point level.*

That is, any query expressible with several applications of least fixed point can already be expressed with one. We also show:

**Theorem 1.** *Let $L$ be a query language consisting of relational calculus plus the least fixed point operator, $\mu$. Suppose that $L$ contains a relation symbol for a total ordering relation on the domain (e.g. lexicographic ordering). Then the queries expressible in $L$ are exactly the queries computable in polynomial time.*

Theorem 1 was discovered independently by M. Vardi [Va82]. It gives a syntactic categorization of those queries which can be answered in polynomial time. Of course queries requiring polynomial time in the size of the database are usually prohibitively expensive. We also consider weaker languages for expressing less complex queries.

# 1 Background and Notation

This section will briefly define and give examples of the objects under consideration. The reader is referred to [Ull80], [End72], [AHU74], and [Mo74] for excellent discussions of relational query languages, first order predicate calculus, computational complexity, and least fixed points, respectively.

First, a *relational database*, $B = \langle U, R_1^{a_1}, \ldots, R_k^{a_k}, c_0, \ldots, c_s \rangle$, is simply a first order structure with finite universe, $U$. For $i = 1 \ldots k$, $R_i^{a_i}$ is an $a_i$-ary relation on $U$, i.e. $R_i \subseteq U^{a_i}$. The superscripts, $a_i$, will be omitted where confusion does not arise.

The constants, $c_0 \ldots c_s$ are elements of $U$. As an example we can consider a database $B_0 = \langle U_0, F_0, P_0, H_0, \text{Abraham} \rangle$; where $U_0$ is a finite set of people,

$$U_0 = \{\text{Abraham, Isaac, Sarah, Leah, } \ldots\}$$

$F_0$ is a monadic relation true of the female elements of $U_0$,

$$F_0 = \{\text{Sarah, Leah, } \ldots\}$$

$P_0$ and $H_0$ are the binary relations for parent and husband, respectively, e.g.

$$P_0 = \{\langle \text{Abraham,Isaac} \rangle, \langle \text{Sarah,Isaac} \rangle, \ldots\}$$

A *similarity type*, $\tau = \langle R_1^{a_1}, \ldots, R_k^{a_k}, c_0, \ldots, c_s \rangle$, is a finite list of relation and constant symbols. $R_i^{a_i}$ is an $a_i$-ary relation symbol. For example, $B_0$ is a database of type $\tau_f = \langle F^1, P^2, H^2, c_0 \rangle$.

Define *Domain Relational Calculus* to be the query language consisting of first order predicate calculus. Thus if $\tau = \langle R_1, \ldots, R_k, c_0, \ldots, c_s \rangle$ is any similarity type, then $\mathcal{L}(\tau)$, the relational calculus for $\tau$, consists of all well formed formulas built up in the usual way from the symbols of $\tau$, together with equality: $=$, logical connectives: $\wedge, \neg$ , variables: $x, y, z, \ldots$, and quantifiers: $\forall, \exists$. For example we can express the grandparent relation as the following wff

$$\gamma(x,y) \equiv (\exists z)\big[P(x,z) \wedge P(z,y)\big]$$

The wff $\gamma \in \mathcal{L}(\tau_f)$ has free variables, $x, y$. If $\gamma$ is thought of as a query to $B_0$, then the response is the set of all pairs, $\langle a,b \rangle \in (U_0)^2$, such that $a$ is a grandparent of $b$. More formally this is the set of all pairs $\langle a,b \rangle$ such that $B_0$ satisfies $\gamma(a,b)$. In symbols:

$$\gamma^{B_0} = \{\langle a,b \rangle \in (U_0)^2 | B_0 \models \gamma(a,b)\}$$

First order logic gives a rich class of database queries, but some plausible queries are not first order expressible. For example it is impossible to express the relation "ancestor(x,y)" in $\mathcal{L}(\tau_f)$. Aho and Ullman [AU79] suggest adding a least fixed point operator to relational calculus so that transitive closures such as "ancestor" may be expressed.

For example, let $R$ be a new binary relation symbol and consider the following first order formula:

$$\varphi(R)[x,y] \equiv \Big(x = y \vee (\exists z)\big[P(x,z) \wedge R(z,y)\big]\Big)$$

For any database $B$ of type $\tau_f$, $\varphi$ induces a mapping, $\Gamma_\varphi^B$ of each binary relation, R, on the universe of $B$ to the binary relation:

$$\Gamma_\varphi^B(R) = \big\{\langle x,y \rangle \bigm| B \models \varphi(R)[x,y]\big\}$$

The operator $\Gamma_\varphi^B$ is *monotone*, i.e. if $R_1 \subseteq R_2$ then $\Gamma_\varphi^B(R_1) \subseteq \Gamma_\varphi^B(R_2)$. It follows that for any database, $B$, $\Gamma_\varphi^B$ has a least fixed point, i.e. a minimal binary relation, $R_0$ with the property that $\Gamma_\varphi^B(R_0) = R_0$. It can be shown that the least fixed point of the above $\varphi$, denoted $(\mu R)\varphi$, is just the ancestor relation – the transitive closure of $P$.

A syntactic criterion which assures that the operator, $\Gamma_\varphi^B$, is always monotone is that $\varphi$ is $R$ positive, i.e $R$ always appears within an even number of $\neg$'s in $\varphi$. As in [AU79] and [CH80b] we define the *fixed point language*, $\mathcal{L}_\mu(\tau)$, to be the closure of $\mathcal{L}(\tau)$ under the operation of taking least fixed points of $R$ positive formulas, $\varphi(R)$.

**Notation:** Given a formula $\varphi(R, \bar{x})$ where $R$ is a relation symbol of arity $a$ occuring positively in $\varphi$ and $\bar{x}$ is an a-tuple of distinct variables, we will write

$$(\bar{t}.\mu R(\bar{x}))\varphi$$

to denote the least fixed point of $\varphi$ with the a-tuple $\bar{t}$ of terms, (i.e. variables and constants), substituted in. For simplicity we will often write $(\mu R)\varphi$ for $(\bar{t}.\mu R(\bar{x}))\varphi$.[1]

**Definition:** Given a similarity type, $\tau$, let $\mathcal{L}_\mu(\tau)$ be the closure of $\mathcal{L}(\tau)$ under the usual operations of conjunction, negation, and quantification, and also under $\mu$: If $\varphi(R, \bar{x}) \in \mathcal{L}_\mu(\tau)$ is as above, then $(\mu R)\varphi$ is also in $\mathcal{L}_\mu(\tau)$.

**Constant Assumption:** We assume in the sequel that every similarity type, $\tau$, under consideration has at least one constant symbol, $c_0$. Without this assumption there is no way to write a boolean query in the form $(\bar{t}.\mu R(\bar{x}))\varphi$ simply because any such expression will have free variables. Without the constant assumption Theorems 1 and 2 must be modified to allow a single quantifier after the application of $\mu$ to get rid of the free variables.

## 2 The Complexity of Fixed Point Queries

In $\mathcal{L}_\mu(\tau)$ we have a very rich class of queries, including but not restricted to all first order queries and transitive closures. It is interesting to consider the complexity of evaluating queries in $\mathcal{L}_\mu(\tau)$. Chandra and Harel show that all fixed point queries are computable in polynomial time:

**Fact 2.1 [CH80b] .** *If $\varphi \in \mathcal{L}_\mu(\tau)$ is any fixed point query, then there is a polynomial, $p$, such that if $B$ is any database of type $\tau$ with universe $U$, then $\varphi^B$, i.e. the query $\varphi$ evaluated on $B$, may be computed in time $p(|U|)$.*

The idea behind the proof is as follows: Let $n = |U|$ be the size of the database and let $\varphi(R)$ be R positive, where the arity of R is $a$. Then $(\mu R)\varphi$ evaluated on B is equal to $\varphi^{(n^a)}(\emptyset)$, i.e. to $\varphi$ applied $n^a$ times to the empty relation. The reason that $n^a$ applications suffice is that until the least fixed point is reached each application adds at least one a-tuple, (remember that $\varphi$ is monotone!); and B has at most $n^a$ a-tuples.

Let *QPTIME* be the set of all queries computable in time polynomial in the size of the database:

$$\text{QPTIME} = \left\{ \varphi \;\middle|\; \text{graph}(\varphi) \in \text{PTIME} \right\}$$

where $\text{graph}(\varphi) = \{\langle B, \bar{c}\rangle | B \models \varphi(\bar{c})\}$[2]. Write $\mathcal{L}_\mu$ for the set of all fixed point queries with $\tau$ unspecified. Thus Fact 2.1 says that $\mathcal{L}_\mu \subseteq$ QPTIME. Chandra and Harel also show that equality does not hold:

[1] See [CH80b] for a more complete explanation of least fixed point operators. We have adopted their notation except that we write $\mu$ where they write $Y$.

[2] We are implicitly identifying a database $B$ with its encoding as a binary string listing the characteristic functions of all of its relations.

**Fact 2.2 [CH80b] .** $\mathcal{L}_\mu \neq$ *QPTIME.*

Their proof shows that queries in $\mathcal{L}_\mu$ don't necessarily have the ability to count. Thus for example the query concerning family databases, "Are there an even number of females?" is not expressible in $\mathcal{L}_\mu(\tau_f)$.

The inability of fixed point queries to count can be eliminated by adding an ordering of the universe to the language. Such an ordering, e.g. by bit representation, is always available in real databases. Let us assume that every database has a total ordering, $\leq$, on its universe. Let $\mathcal{L}_\mu(\leq)$ be the set of fixed point queries which may include the logical relation, $\leq$, which must be interpreted as a total ordering on the universe. We will show in the next section that $\mathcal{L}_\mu(\leq) =$ QPTIME.

In fact we will see that any polynomial time query may be expressed as a simple least fixed point, i.e. in the form $(\mu R)\varphi$ where $\varphi$ is first order. Let us first describe the proposed *Fixed Point Hierarchy* of Chandra and Harel. They considered a classification of queries in $\mathcal{L}_\mu$ by the number of alternate applications of quantification and of $\mu$. Let

$$\begin{array}{ll} \Sigma_0 & = \{M \in \mathcal{L}_\mu | M \text{ is quantifier and } \mu \text{ free}\} \\ \Pi_\alpha & = \{\neg\varphi | \varphi \in \Sigma_\alpha\} \\ \Sigma_{\alpha+1} & = \{(\exists x)\varphi | \varphi \in \Pi_\alpha\} \\ \Sigma_{\omega n} & = \{(\mu R)\varphi | \varphi \in \Sigma_\alpha, \alpha < \omega n\} \end{array}$$

Thus $\Sigma_m$ is the set of first order queries with $m$ alternations of quantification begining with existential; and $\Sigma_{\omega n}$ is the set of queries expressible using $n$ applications of $\mu$ with intermediate applications of quantification and negation. Since our queries must be finite we have $\mathcal{L}_\mu = \Sigma_{\omega^2}$. It is known that additional alternations of first order quantification give increased expressibility. This is proved in [CH80b] for languages without ordering. In [Si83] Sipser showed that polynomial size bounded depth circuits form a strict hierarchy for depth. A corollary of this result as noted in [Im83] is the strict first order hierarchy for alternation of quantifiers where any fixed set of logical relations including $\leq$ is used. It is also known that transitive closure is not first order expressible with or without ordering: see [AU79] or [Im81]. Thus we have

**Fact 2.3 .** *The fixed point hierarchy is strict up to and including $\Sigma_\omega$, i.e. all the following containments are strict:*

1. $\Sigma_0 \subset \Sigma_1 \subset \ldots \subset \bigcup_{i=0}^{\infty} \Sigma_i \subset \Sigma_\omega$

2. $\Sigma_0(\leq) \subset \Sigma_1(\leq) \subset \ldots \subset \bigcup_{i=0}^{\infty} \Sigma_i(\leq) \subset \Sigma_\omega(\leq)$

Chandra and Harel ask whether the hierarchy continues past $\Sigma_\omega$. We will show that it does not. In the next section we show that in the presence of ordering the hierarchy stops at $\Sigma_\omega(\leq)$ and is equal to the polynomial time computable queries. In section 4 we will show the more subtle fact that even without ordering the hierarchy stops at $\Sigma_\omega$.

## 3 In the Presence of Ordering

In this section we prove our first main result,

**Theorem 1.** $\Sigma_\omega(\leq) =$ *QPTIME* $= \mathcal{L}_\mu(\leq)$

**Proof** We have already seen that $\mathcal{L}_\mu(\leq) \subseteq$ QPTIME. We must show that QPTIME $\subseteq \Sigma_\omega(\leq)$. To make our presentation slightly simpler we will only consider boolean queries. Let S be a set of databases, B, of type $\tau$. Let M be a Turing machine which accepts S in time less than $n^k$. Here $n$ is the size of the universe of the input database, B, being tested for membership in S. We must show that there is a query $\alpha \in \Sigma_\omega(\leq)$ which expresses S, i.e.

$$S = \{B \mid B \models \alpha\} = \{B \mid M \text{ accepts } B\}$$

We will show that M's computation on input B can be described in $\Sigma_\omega(\leq)$. To do this, we will build a first order formula $\varphi_M$ whose least fixed point evaluated on any B is a coding of M's computation on input B. There are two steps to writing the formula $\varphi_M$. First we show in Lemma 3.1 that the first line of M's computation, i.e. the input database, B, can be described in a first order wff. Second we show that given one line of M's computation we can describe the next line in a first order way. Thus $\varphi_M$ will determine a monotone operator which given any partial computation, R, will add the first line of M's computation on B that has not yet been filled in. Thus the least fixed point of $\varphi_M$ will be the entire computation. Then we can read the answer of whether M accepts or rejects B from $(\mu R)\varphi_M$.

Now let's look at some of the details. Each candidate for S is a database B which has a finite universe, U, with a total ordering, $\leq$, on it. Let $n = |U|$. We can think of U as the set of integers from 0 to $n-1$ with the usual ordering. We will use k-tuples of variables to denote numbers between 0 and $n^k - 1$. Using one application of $\mu$ we will form the relation $C_M = (\mu R)\varphi_M$ which codes M's computation. That is: $B \models C_M(p_1, \ldots, p_k, t_1, \ldots, t_k, a)$ if and only if in M's computation on input B, the contents of the cell $p_1 \ldots p_k$ at time $t_1 \ldots t_k$ is 'a'. Once we have written $C_M$ we can let

$$\alpha = C_M(0, n^k - 1, q_f) \quad .$$

Here $\alpha$ says that M is in its accept state, $q_f$, after $n^k - 1$ steps. Thus as desired

$$B \models \alpha \quad \Leftrightarrow \quad B \in S$$

The first step in building $\varphi_M$ is to write the sentence $M_0(\bar{p}, a)$ meaning that at time 0, cell $\bar{p}$ is a. We will show in Lemma 3.1 that for any Turing machine, $M$, the wff $M_0$ is first order expressible.

**Lemma 3.1 .** *$M_0(\bar{p}, a)$ is first order expressible, i.e. $M_0 \in \mathcal{L}(\tau, \leq)$.*

**Proof** This is a matter of encoding and decoding B. Suppose for concreteness that $k = 3$ and that $\tau$ consists of a single binary relation symbol, E. We code B on M's input tape with a sequence of $n^2$ bits coding E, followed by a sequence of $n^3 - n^2$ blanks. Before we write $M_0$ we must know how the symbols of M's instantaneous description are coded. Assume, for example, that 0 and 1 code themselves, 2 codes 'blank', 3 codes the start state looking at a 0, and 4 codes the start state looking at a 1. Using the ordering on B's domain we may assume that we have symbols for these numbers. **Note:** In writing $\alpha$ we may assume that $n$ is larger than a given constant $k$. We can assure this by listing all the element of S whose size is at most $k$. We then say, "Either B is on this list, or B has more than $k$ elements and $\alpha$ holds."

Now $M_0$ for the above example is given as follows:

$$\begin{aligned}
M_0(p_1, p_2, p_3, a) \equiv & \\
& \Big[(p_1 = p_2 = p_3 = 0) \wedge \big((E(0,0) \wedge a = 4) \vee (\neg E(0,0) \wedge a = 3)\big)\Big] && (1)\\
\vee & \Big[p_1 = 0 \wedge (p_2 \neq 0 \vee p_3 \neq 0) \wedge (a = 0 \vee a = 1) \wedge \big(E(p_2, p_3) \leftrightarrow a = 1\big)\Big] && (2)\\
\vee & \quad \Big[p_1 \neq 0 \wedge a = 2\Big] && (3)
\end{aligned}$$

The above mess says (1): "The first tape symbol is M's start state looking at the first bit of E," (2): "The next $n^2 - 1$ tape cells are 1 or 0 according as the corresponding bit of E holds or doesn't hold," and (3): "The last $n^3 - n^2$ cells are blank." We hope the reader can generalize from this example to arbitrary $\tau$. ∎

Now that we have expressed the input tape, we can complete the description of $\varphi_M$. $\varphi_M$ will have a relation variable, R, of arity $2k+1$. If R codes a partial computation of M, then $\varphi_M(R)$ codes one additional step of this computation.

Let the notation "$\langle xyz \rangle \to w$" mean that if at a given moment the tape cells $i-1, i, i+1$ contain the letters $x, y, z$, respectively, then at the next move of M cell $i$ will contain the letter $w$. Thus "$\langle xyz \rangle \to w$" is just an abbreviation for the following disjunction:

$$\langle xyz \rangle \to w \quad \equiv \quad \bigvee_{\langle c_{-1}, c_0, c_1, c \rangle \in \delta_M} \Big(x = c_{-1} \wedge y = c_0 \wedge z = c_1 \wedge w = c\Big)$$

where $\delta_M$ is the appropriate finite set of quadruples. Thus $\varphi_M(R)$ codes the input tape and includes those tuples $\langle \bar{p}, \bar{t}, a \rangle$ whose precursors, $a_{-1}, a_0$, and $a_1$

already appear in the computation coded by R. In symbols,

$$\varphi_M(R)[\bar{p},\bar{t},a] \equiv (\bar{t}=0 \wedge M_0(\bar{p},a)) \vee \Big[\exists a_{-1}a_0a_1\big(\langle a_{-1}a_0a_1\rangle \to a \wedge R(\bar{p}-1,\bar{t}-1,a_{-1}) \wedge R(\bar{p},\bar{t}-1,a_0) \wedge R(\bar{p}+1,\bar{t}-1,a_1)\big)\Big]$$

Note that the crucial use of $\leq$ is in expressing the motion left or right on the input tape, i.e. to write $\bar{p}-1$ or $\bar{p}+1$.

Let $C_M = (\mu R)\varphi_M$ and put $\alpha \equiv C_M(0, n^{k-1}, q_f)$. Thus:

$$B \models \alpha \Leftrightarrow B \in S$$

as desired. It is in this last step that we need the constant assumption. If we have at least one constant symbol, $c_0$, then we can do our coding so that $c_0$ means the correct thing in each place, i.e. so that $C_M(0, n^{k-1}, q_f) \equiv C_M(c_0, \ldots, c_0)$[3] and we are done. Otherwise as mentioned above an additional quantification is necessary. ∎

# 4 One Fixed Point Suffices

If we do not have access to an ordering on the universe then it is not in general possible to simulate a computation, so Theorem 1 fails (cf. Fact 2.2). We can still show, however, that the hierarchy collapses at the first fixed point level:

**Theorem 2.** $\mathcal{L}_\mu = \Sigma_\omega$ .

To prove this theorem we will use some of the machinery developed in [Mos74]. Moschovakis considers inductive definitions on a fixed infinite structure and he assumes that $\mathcal{L}$ contains a constant symbol for each element of the structure. We consider uniform inductive definitions for all finite relational structures of a given similarity type and we assume only that at least one constant symbol exists. We must thus check that Moschovakis' results remain true in this new setting. We derive the facts we need in 4.1 through 4.7, below.

The outline of the proof of Theorem 2 is as follows. We want to show that several applications of $\mu$ are no more expressive than one. Following [Mos74] we show that two simultaneous inductions can be combined into a single one, (Lemma 4.1 – Simultaneous Induction Lemma). Next we show that two nested applications of $\mu$ with no occurrences of negation may be combined into a single one, (Lemma 4.2 – Transitivity Theorem). It immediately follows (Corollary 4.3) that $\Sigma_\omega$ is closed under conjunction, disjunction, and quantification. Third

[3] That is we use a slightly different ordering for position inwhich $c_0$ is the first element; for time inwhich $c_0$ is the last element; and coding symbols so that $c_0$ codes the accepting state, $q_f$. Of course there are more straight forward ways of doing the same thing by increasing the arity of the fixed point by one.

we show that $\Sigma_\omega$ is closed under negation. This surprising fact is true because in a finite structure any least fixed point will be reached after a finite number of iterations. Furthermore we can express in $\Sigma_\omega$ the relations $\bar{x} <_\varphi \bar{y}$, (and $\bar{x} \leq_\varphi \bar{y}$), meaning that in the computation of $(\mu R)\varphi$, the tuple $\bar{x}$ enters the relation and does so before $\bar{y}$, (resp. no later than $\bar{y}$). This is the Stage Comparison Theorem ( Fact 4.4). Using $<_\varphi$ and $\leq_\varphi$ we can express $\mathrm{MAX}_\varphi(\bar{x})$ meaning that $\bar{x}$ is of maximal rank with respect to $\varphi$, i.e. it comes in on the last round of the computation of $(\mu R)\varphi$. Finally once we have a tuple, $\bar{x}$, of maximum rank, anything of greater rank will never enter the fixed point, i.e. negation may be expressed as follows:

$$\neg\big(\bar{y}.\mu R\big)\varphi \equiv \exists \bar{x}\big(\mathrm{MAX}_\varphi(\bar{x}) \wedge \bar{x} <_\varphi \bar{y}\big)$$

Before we prove our first lemma we make a few convenient definitions. If $\alpha(T)$ is any formula where $T$ is an r-ary relation symbol and if $R$ is an $r+m$-ary relation symbol and $\bar{t}$ is an m-tuple of terms then the notation

$$\alpha\big(\{\bar{u}|R(\bar{t},\bar{u})\}\big)$$

will mean the result of replacing each occurrence of $T(\bar{v})$ in $\alpha(T)$ by $R(\bar{t},\bar{v})$. Let $\alpha(R,\bar{x})$ be any R positive formula where the arity of R is $|\bar{x}|$. When the structure, $\mathcal{A}$, is understood we will follow Moschovakis and use the notation $I^n_\alpha$ to denote the $n$th iterate of $\alpha$, inductively:

$$I^0_\alpha = \emptyset, \qquad I^{n+1}_\alpha = \{\bar{a}|\mathcal{A} \models \alpha(I^n_\alpha, \bar{a})\}$$

Also define the *closure ordinal* of $\varphi$ in $\mathcal{A}$, in symbols $cl(\varphi)$, to be the first ordinal $\alpha$ such that

$$\mathcal{A} \models \Big(\varphi^{(\alpha)}(\emptyset) \leftrightarrow \varphi^{(\alpha+1)}(\emptyset)\Big)$$

Note that if $\mathcal{A}$ is finite then so is $cl(\varphi)$. We also write $I_\varphi$ to denote $I^{cl(\varphi)}_\varphi$, i.e. the least fixed point of $\varphi$ in $\mathcal{A}$.

The following lemma shows that two simultaneous inductions may be combined into one:

**Lemma 4.1 (cf. Simultaneous Induction Lemma, [Mos74]).** *Suppose $\psi(\bar{y},S,T)$ and $\varphi(\bar{x},S,T)$ are first order formulas that are positive in S and T. Let $r_1 = arity(S) = |\bar{y}|$, and $r_2 = arity(T) = |\bar{x}|$. For any finite structure $\mathcal{A}$ define the relations $I^\omega_0$ and $I^\omega_1$ by simultaneous induction:*

$$\begin{array}{lcl}
I^0_0 & = & I^0_1 \;=\; \emptyset \\
\bar{a} \in I^n_0 & \Leftrightarrow & \mathcal{A} \models \psi(\bar{a}, I^{n-1}_0, I^{n-1}_1) \\
\bar{b} \in I^n_1 & \Leftrightarrow & \mathcal{A} \models \varphi(\bar{b}, I^{n-1}_0, I^{n-1}_1) \\
I^\omega_k & = & \displaystyle\bigcup_{n=1}^{\infty} I^n_k, \quad k = 0,1
\end{array}$$

*Then both $I_0^\omega$ and $I_1^\omega$ are uniformly inductive, i.e. they are expressible in $\Sigma_\omega$ and the same formula works for all structures $\mathcal{A}$.*

**Proof** Following [Mos74] first assume that we have constants $c_0$ and $c_1$ which are guaranteed to refer to distinct elements of $\mathcal{A}$.

Let $\bar{x}^*$ and $\bar{y}^*$ be any sequence of constants of lengths $r_2$ and $r_1$ respectively, for example $\bar{x}^*$ could be an $r_2$-tuple of $c_0$'s. Let $U$ be a relation symbol of arity $r_1 + r_2 + 1$. Put

$$\begin{aligned}\chi(t,\bar{y},\bar{x},U) \quad &\equiv \quad \left[t = c_0 \wedge \psi\Big(\bar{y}, \{\bar{y}'|U(c_0,\bar{y}',\bar{x}^*)\}, \{\bar{x}'|U(c_1,\bar{y}^*,\bar{x}')\}\Big)\right] \\ &\vee \quad \left[t = c_1 \wedge \varphi\Big(\bar{x}, \{\bar{y}'|U(c_0,\bar{y}',\bar{x}^*)\}, \{\bar{x}'|U(c_1,\bar{y}^*,\bar{x}')\}\Big)\right]\end{aligned}$$

Then for all $n$ and $\mathcal{A}$,

$$\bar{y} \in I_0^n \Leftrightarrow \langle c_0, \bar{y}, \bar{x}^* \rangle \in I_\chi^n \ .$$

This is a straightforward induction, see [Mos74] for details.

If we don't have the distinct constants $c_0$ and $c_1$ available then the proof can be modified as follows. We increase the arity of $U$ by one and replace occurrences of $t$ by the pair $t_1$,$t_2$. Each clause of the form $t = c_0$ is changed to $t_1 = t_2$; $t = c_1$ is changed to $t_1 \neq t_2$. Where before we substituted $c_1$, we now existentially quantify a $t_2$ not equal to $t_1$ and substitute this new pair.

The next result shows that two uses of $\mu$ are no more powerful than one assuming that there are no intervening negations.

**Lemma 4.2 (cf. Transitivity Theorem, [Mos74]).** *Suppose that $R$ and $S$ occur only positively in $\varphi(R,S)$ and $\psi(R)$. Then the nested fixed point $(\mu S)\varphi((\mu R)\psi, S)$ is expressible in $\Sigma_\omega$, i.e. there is a positive wff $\chi$ such that*

$$(\mu S)\varphi((\mu R)\psi, S) \Leftrightarrow (\mu U)\chi(U) \ .$$

**Proof** If we assume that there are two constants, $c_0$ and $c_1$, always representing distinct elements then Moschovakis' defintion of $\chi$ and his proof goes through without any change:

Let $r = \text{arity}(R)$, and $s = \text{arity}(S)$. Let $\bar{y}^*$ and $\bar{x}^*$ be an r-tuple and an s-tuple of constants. Put

$$\begin{aligned}\chi(t,\bar{y},\bar{x},U) \equiv \quad &\left[t = c_0 \wedge \psi\Big(\bar{y}, \{\bar{y}'|U(c_0,\bar{y}',\bar{x}^*)\}\Big)\right] \qquad \vee \\ &\left[t = c_1 \wedge \varphi\Big(\bar{x}, \{\bar{y}'|U(c_0,\bar{y}',\bar{x}^*)\}, \{\bar{x}'|U(c_1,\bar{y}^*,\bar{x}')\}\Big)\right]\end{aligned}$$

Let $\mathcal{A}$ be any finite structure and let m be the closure ordinal of $\varphi$ in $\mathcal{A}$. The formula $\chi$ simultaneously simulates $\psi$ and $\varphi$. Thus for all naturals $n$

$$I_\psi^n = \{\bar{y}' \mid \langle c_0, \bar{y}', \bar{x}^* \rangle \in I_\chi^n\} \tag{1}$$

After $m$ iterations the computation of $I_\psi$ is complete so the simulation of $\varphi$ can begin in earnest. We have for all $n$:

$$I_\varphi^n \subseteq \{\bar{x}' \mid \langle c_1, \bar{y}^*, \bar{x}' \rangle \in I_\chi^{n+m}\} \subseteq I_\varphi^{n+m} \tag{2}$$

Thus in particular

$$I_\varphi = \{\bar{x}' | \langle c_1, \bar{y}^*, \bar{x}' \rangle \in I_\chi\}$$

and we have

$$\Big[(\bar{x}'.\mu S)\varphi\big((\mu R)\psi, S\big)\Big] = \Big[(c_1, \bar{y}^*, \bar{x}'.\mu U)\chi\Big]$$

i.e. $(\mu S)\varphi$ is computed as a single fixed point as desired. Equations 1 and 2 are proved by inductions, see [Mos74] for further details. Note that if constants are not available then the result still holds; we modify the proof as in Lemma 4.1. ∎

Another way of stating lemma 4.2 is to say that $\Sigma_\omega$ is closed under least fixed point. Note that $\mu$ is at least as powerful as quantification, conjunction, and disjunction. For example we could write $(\forall y)R(y, \bar{x})$ as $(\bar{x}.\mu(S)(\forall y R(y, \bar{x}))$. Thus the following is immediate:

**Corollary 4.3 .** *$\Sigma_\omega$ is closed under quantification, disjunction, conjunction, and taking of least fixed points.*

Let $\varphi(x_1 \dots x_r, R)$ be an $R$ positive formula and let $\mathcal{A}$ be a finite structure. Each tuple $\langle a_1 \dots a_r \rangle \in I_\varphi$ comes in at some stage of the induction. Let $|\bar{a}|_\varphi$, the *rank* of $\bar{a}$ with respect to $\varphi$, be the step at which $\bar{a}$ enters $I_\varphi$:

$$|\bar{a}|_\varphi = \begin{cases} n & \bar{a} \in I_\varphi^n - I_\varphi^{n-1} \\ \infty & \bar{a} \notin I_\varphi \end{cases}$$

Define the relation $\bar{x} \leq_{\varphi\psi} \bar{y}$ to mean $\bar{x} \in I_\varphi$ and $|\bar{x}|_\varphi \leq |\bar{y}|_\psi$. Similarly $\bar{x} <_{\varphi\psi} \bar{y}$ means that $|\bar{x}|_\varphi < |\bar{y}|_\psi$. A powerful result is that even though we may not have an ordering on the universe we can express propositions concerning the relative times at which tuples appear in fixed points.

**Fact 4.4.** *[Mo74: Stage Comparison Theorem] Given positive formulas $\varphi(R)$ and $\psi(S)$ the relations $\leq_{\varphi\psi}$ and $<_{\varphi\psi}$ are uniformly inductive, i.e. expressible in $\Sigma_\omega$.*

Moschovakis' proof goes through without change, except for reading all ordinals as finite.

Let $\psi(x, S)$ be S positive and let $\leq_\psi$, (resp. $<_\psi$), abbreviate $\leq_{\psi\psi}$, (resp. $<_{\psi\psi}$). The following are simultaneous inductive definitions for $\leq_\psi$ and $<_\psi$.

These definitions will be used in the proof of Lemma 4.7. Note that unlike the inductions in [Mo74] they only work for finite structures because in an infinite structure $x$ may not have an immediate predecessor, $z$.

$$\begin{array}{rl} x \leq_\psi y \equiv & \psi(x,\emptyset) \vee \exists z\big(z <_\psi y \wedge \psi(x,\{x'|x' \leq_\psi z\})\big) \\ x <_\psi y \equiv & \big(\psi(x,\emptyset) \wedge \neg\psi(y,\emptyset)\big) \qquad \vee \\ & \exists z\big[z <_\psi y \wedge \psi(x,\{x'|x' \leq_\psi z\}) \wedge \neg\psi(y,\{x'|\neg z <_\psi x'\})\big] \end{array}$$

Note that the above equations fit into the form of lemma 4.1. We can rewrite them in the more familiar form:

$$\begin{array}{rl} \alpha(x,y,S,T) \equiv & \psi(x,\emptyset) \vee \exists z\big(T(z,y) \wedge \psi(x,\{x'|S(x',z)\})\big) \\ \beta(x,y,S,T) \equiv & \big(\psi(x,\emptyset) \wedge \neg\psi(y,\emptyset)\big) \qquad \vee \\ & \exists z\big[T(z,y) \wedge \psi(x,\{x'|S(x',z)\}) \wedge \neg\psi(y,\{x'|\neg T(z,x')\})\big] \end{array}$$

**Claim .** *For all k,*

$$\begin{array}{rl} I^k_\alpha & = \big\{\langle x,y\rangle \;\big|\; |x|_\psi \leq k \text{ and } |x|_\psi \leq |y|_\psi\big\} \\ I^k_\beta & = \big\{\langle x,y\rangle \;\big|\; |x|_\psi \leq k \text{ and } |x|_\psi < |y|_\psi\big\} \end{array}$$

**Proof** By induction on k. This is clear for $k = 0,1$. Let $k \geq 1$ and $|x|_\psi = k+1$. Let z be such that $|z|_\psi = k$. Then $\alpha(x,y,I^k_\alpha,I^k_\beta)$ (resp. $\alpha(x,y,I^k_\alpha,I^k_\beta)$) holds iff it holds with z as a witness iff $|y|_\psi \geq k+1$ (resp. $|y|_\psi > k+1$). ∎

In order to negate fixed points we need a slight modification of the above fact. Let $\bar{x} \ll_\varphi \bar{y}$ mean $|\bar{x}|_\varphi + 1 < |\bar{y}|_\varphi$. Then

**Lemma 4.5.** *If $\varphi(\bar{x},R)$ is R-positive then the formula $\ll_\varphi$ is first order expressible using positive occurrences of $\leq_\varphi$ and $<_\varphi$.*

**Proof**

$$\bar{x} \ll_\varphi \bar{y} \quad \Leftrightarrow \quad \bar{x} \leq_\varphi \bar{x} \;\wedge\; \neg\varphi(\bar{y},\{\bar{u}|\neg\bar{x} <_\varphi \bar{u}\})$$

∎

As we have already pointed out, if $\mathcal{A}$ is finite then $cl(\varphi)$, the closure ordinal of $\varphi$, is also finite. Thus there must be at least one tuple $\bar{m}$ of maximal rank, namely $|\bar{m}|_\varphi = cl(\varphi)$. In the next lemma we show that we can say in $\Sigma_\omega$ that $\bar{m}$ is of maximal rank. We can thus express the negation of a fixed point: the tuple $\bar{x}$ will never come into the fixed point if it's rank is greater than the maximum rank.

**Lemma 4.6.** *$\Sigma_\omega$ is closed under negation.*

**Proof** Let $\varphi(\bar{x},R)$ be R-positive. We must show that $\neg\big[(\mu R)\varphi\big] \in \Sigma_\omega$.

Let

$$\mathrm{MAX}_\varphi(\bar{y}) \equiv \Big[\forall\bar{z}(\bar{z} \leq_\varphi \bar{y} \vee \bar{y} \ll_\varphi \bar{z})\Big]$$

Note that $\mathrm{MAX}_\varphi(\bar{y})$ says that $\bar{y}$ has maximum rank. Put

$$\nu(\bar{x}) \equiv \exists \bar{y}\Big(\mathrm{MAX}_\varphi(\bar{y}) \wedge \bar{y} <_\varphi \bar{x}\Big)$$

$\nu(\bar{x})$ says that there exists a tuple $\bar{y}$ of maximum rank, and that the rank of $\bar{x}$ is greater than the rank of $\bar{y}$. Thus

$$\nu(\bar{x}) \Leftrightarrow \neg\big[(\mu R)\varphi\big]$$

It follows from 4.1 through 4.6 that $\nu \in \Sigma_\omega$. ∎

I had thought that the proof of theorem 2 was now complete. However a referee pointed out the following problem. Consider the formula $\alpha \in \Sigma_{\omega 2}$:

$$\alpha \quad \equiv \quad (\mu S)\Big[\psi\big(x, \big(\neg(\mu R)\varphi(u, \neg S, R)\big), S\big)\Big]$$

Although $S$ occurs positively in $\alpha$, it occurs positively and negatively in $\leq_\varphi$ and $<_\varphi$.[4] Thus it remains to be shown that:

**Lemma 4.7 .** *The above formula $\alpha$ is equivalent to a formula in $\Sigma_\omega$.*

**Proof** We have already seen how to define $\mathrm{MAX}_{\varphi,S}(u)$ meaning that $u$ enters the fixed point $I_{\varphi,S}$ at the final stage. It follows that we can express the condition that the fixed point is finished, and we can express its negation. Now we wish to describe a two level fixed point computation: Compute $R_0 = I_{\varphi,\emptyset}$, then $S_0 = I_{\psi,R_0}$, then $R_1 = I_{\varphi,S_0}$, and so on. To do this we define the following six relations by simultaneous induction:

$$\leq_{\varphi,z},\ <_{\varphi,z},\ \leq_{\varphi,\emptyset},\ <_{\varphi,\emptyset},\ \leq_\psi,\ <_\psi$$

Here $u \leq_{\varphi,z} v$ (resp. $u <_{\varphi,z} v$) means that $u$ enters $I_\varphi$ and does so at least as soon as (resp. sooner than) $v$ does, where

$$S = \{y' | y' \leq_\psi z\}, \neg S = \{y' | z <_\psi y'\}$$

Also $u \leq_{\varphi,\emptyset} v$ and $u <_{\varphi,\emptyset} v$ are the analagous relations when $S = \emptyset$. The positive inductive definitions of $\leq_{\varphi,z}$ and $<_{\varphi,z}$ using $\leq_\psi$ and $<_\psi$ are immediate from Fact 4.4. For example the definition of $x \leq_{\varphi,z} y$ is the conjunction of "$z \leq_\psi z$" with the old definition of "$x \leq_\varphi y$" where the above expressions have been substituted for $S$ and $\neg S$.

[4] See [GS85] for a more general result than lemma 4.7 concerning making monotone fixed points positive.

Put

$$
\begin{aligned}
\mathrm{MAX}_{\varphi,z}(u) &\equiv u \leq_{\varphi,z} u \wedge \forall w\big[w \leq_{\varphi,z} u \vee \neg\varphi(w, \{x'|\neg x' \leq_\psi z\}, \{u'|\neg u <_{\varphi,z} u'\})\big] \\
\gamma(y,z) &\equiv \exists u\big[\mathrm{MAX}_{\varphi,z}(u) \wedge \psi(y, \{u'|u <_{\varphi,z} u'\}, \{x'|x' \leq_\psi z\})\big] \\
\tilde{\gamma}(y,z) &\equiv \exists u\big[\mathrm{MAX}_{\varphi,z}(u) \wedge \neg\psi(y, \{u'|\neg u' \leq_{\varphi,z} u\}, \{x'|\neg z <_\psi x'\})\big] \\
\gamma(y,\emptyset) &\equiv \exists u\big[\mathrm{MAX}_{\varphi,\emptyset}(u) \wedge \psi(y, \{u'|u <_{\varphi,\emptyset} u'\}, \emptyset)\big] \\
\tilde{\gamma}(y,\emptyset) &\equiv \exists u\big[\mathrm{MAX}_{\varphi,\emptyset}(u) \wedge \neg\psi(y, \{u'|\neg u' \leq_{\varphi,\emptyset} u\}, \emptyset)\big]
\end{aligned}
$$

The above formulas are all positive in the relations being inductively defined. Note that $\gamma(y,z)$ (resp. $\tilde{\gamma}(y,z)$) is equivalent to $\psi(y, \neg(\mu R)\varphi(\neg S, R), S)$ (resp. $\neg\psi(y, \neg(\mu R)\varphi(\neg S, R), S)$) where $S = \{x'|x' \leq_\psi z\}$. Thus if we substitute $\gamma$ (resp. $\tilde{\gamma}$) for $\psi$, (resp. $\neg\psi$) in the equations following Fact 4.4 we arrive at inductive definitions for $\leq_\psi$ and $<_\psi$:

$$
\begin{aligned}
x \leq_\psi y &\equiv \gamma(x,\emptyset) \vee \exists z\big(z <_\psi y \wedge \gamma(x,z)\big) \\
x <_\psi y &\equiv \big(\gamma(x,\emptyset) \wedge \tilde{\gamma}(y,\emptyset)\big) \vee \exists z\Big(z <_\psi y \wedge \gamma(x,z) \wedge \tilde{\gamma}(y,z)\Big)
\end{aligned}
$$

It now follows from lemma 4.1 that $\leq_\psi$ and thus $\alpha$ are expressible in $\Sigma_\omega$. ∎

A formula with fixed points and negations nested to a depth greater than 2 can be handled by repeatedly using lemma 4.7 from the inside out. Note that the parameter $P$ from an outer fixed point $(\mu P)\theta$ may appear both positively and negatively in an inner fixed point during the construction. However when we substitue $P = \{v|v \leq_\theta z\}$ and $\neg P = \{v|z <_\theta v\}$ in these inner formulas they become positive again. This completes the proof of theorem 2. ∎

It should be noted that Theorem 2 is a general result saying that in finite model theory any property expressible with several alternations of $\mu$ and $\neg$ is already expressible with one positive application of $\mu$. This result is not true for infinite models, cf. [Mo74].

# 5 Relations to Previous Work

Another way to view $\mu$ is as an operator that iterates a given formula a polynomial number of times. More precisely, let $\varphi(x_1, \ldots, x_a, R)$ be positive in $R$ where $R$ has arity $a$. Let $\varphi^{(m)}(\emptyset)$ denote the formula $\varphi$ applied to itself $m$ times and then applied to the empty set. Inductively,

$$
\begin{aligned}
\varphi^{(0)}(\bar{x},\emptyset) &\equiv x_1 \neq x_1 \\
\varphi^{(0)}(\bar{x},R) &\equiv R(\bar{x}) \\
\varphi^{(m+1)}(\bar{x},R) &\equiv \varphi\big(\bar{x}, \{\bar{u}|\varphi^{(m)}(\bar{u},R)\}\big)
\end{aligned}
$$

We have already noted the following in our discussion of Fact 2.1:

**Proposition 5.1 .** *Let $\varphi$ be as above and let $\mathcal{A}$ be a structure of size $n$ for the language of $\varphi$. Then*

$$\mathcal{A} \models \Big( (\mu R)\varphi \leftrightarrow \varphi^{(n^a)}(\emptyset) \Big)$$

Thus as stated $\mu$ is an iteration operator. One problem with this proposition is that if $R$ occurs more than once in $\varphi$ then the formula $\varphi^{(n^a)}$ will be of size exponential in $n^a$. It is not hard to show however that $\varphi$ is always equivalent to a formula in which $R$ occurs only once.

In the next few arguments it will be convenient to use the following notation:

$$(\forall v.A)\alpha \equiv (\forall v)(A \rightarrow \alpha), \quad (\exists v.A)\alpha \equiv (\exists v)(A \wedge \alpha)$$

**Fact 5.2 [Canonical Form for Positive Formulas, Mo74] .** *Let $\varphi$ be $R$ positive. Then there is a quantifier and $R$ free formula $\theta(\bar{x}, \bar{z}, \bar{u})$ and a block of quantifiers $Q_1, \ldots, Q_k$ such that*

$$\varphi(\bar{x}, R) \equiv \Big[ (Q_1 z_1)(Q_2 z_2) \ldots (Q_k z_k)(\forall u_1) \ldots (\forall u_n.\theta(\bar{x}, \bar{z}, \bar{u})) \Big] R(\bar{u})$$

To make things neater we can requantify the variables, $x_1 \ldots x_n$. Note that

$$\varphi(\bar{x}, R) \equiv \Big[ (Q_1 z_1)(Q_2 z_2) \ldots (Q_k z_k)(\forall u_1) \ldots (\forall u_n.\theta(\bar{x}, \bar{z}, \bar{u}))(\exists x_1.x_1 = u_1) \ldots (\exists x_n.x_n = u_n) \Big] R(\bar{x})$$

Now combining 5.1 and 5.2 we see that the $\mu$ operator can be thought of as a quantifier block repeater:

**Corollary 5.3 .** *Let $\varphi$ be as above and let*

$$QBLOCK \equiv \Big[ (Q_1 z_1)(Q_2 z_2) \ldots (Q_k z_k)(\forall u_1) \ldots (\forall u_n.\theta(\bar{x}, \bar{z}, \bar{u}))(\exists x_1.x_1 = u_1) \ldots (\exists x_n.x_n = u_n) \Big]$$

*Then for any structure $\mathcal{A}$ of size $n$*

$$\mathcal{A} \models \Big( (\mu R)\varphi \leftrightarrow QBLOCK^{(n^a)}(x_1 \neq x_1) \Big)$$

The above formulation makes our results about $\mu$ fit in with some of our previous work concerning expressibility and complexity, [Im81,Im82a]. In particular an immediate corollary of Corollary 5.3 and Theorem B.5 from [Im82a] is another proof of Theorem 1.

Since $\mu$ is an iteration operator we propose a new query hierarchy based on such iterations:

**Definition:** Let $IQ[f(n)]$ be the set of queries expressible by iterating a first order query $f(n)$ times. An equivalent formulation is the set of queries whose value on a structure of size $n$ is equivalent to some quantifier block repeated $f(n)$ times:

$$IQ[f(n)] = \{ \varphi \mid (\exists \mathrm{QBLOCK})(\forall \mathcal{A})(\mathcal{A} \models (\varphi \leftrightarrow \mathrm{QBLOCK}^{(f(|\mathcal{A}|))}(x_1 \neq x_1)) \}$$

As an example let

$$\alpha(R, x_1, x_2) \equiv \Big(x_1 = x_2 \vee E(x_1, x_2) \vee (\exists z)[R(x_1, z) \wedge R(z, x_2)]\Big)$$

It is easy to see that $E^*$, the transitive closure of E, is equal to $(\mu R)\alpha$ which is in turn equal to $\alpha^{(\log n)}$ for graphs of size $n$. Thus the transitive closure query is in $IQ(\log n)$.

Let $IQ(\leq)$ be the set of iterated queries which include the logical relation $\leq$ denoting a total ordering on the universe. The following Fact summarizes some of the known facts concerning $IQ$. The proofs (though not quite these statements) may be found in [Im81,Im82a].

**Fact 5.4 .**

1. $IQ(\leq)[1] =$ *First Order Queries* $\subset\neq$ *QSPACE*$[\log n]$
2. *Transitive Closures* $\subseteq IQ[\log n]$
3. $\bigcup_{k=1}^{\infty} IQ[\log^k n] \subset\neq IQ[n]$
4. $\bigcup_{k=1}^{\infty} IQ(\leq)[n^k] = \mathcal{L}_\mu(\leq) = QPTIME$
5. $IQ(\leq) = QPSPACE$

# 6 Conclusions, and Directions for Future Work

We have shown that all queries using first order quantification and a least fixed point operator, $\mu$, may be expressed with a single occurence of $\mu$ applied to a first order expression. Furthermore, in the presence of a total ordering, $\leq$, the queries so expressible are exactly the polynomial time computable queries. Finally, a further study of the number of iterations needed to compute fixed points is desirable. The following open problems should be considered:

1. One attraction of Theorem 1 is that it shows that $\mathcal{L}_\mu(\leq)$ is a very general query language in which the complexity of a given query is clear from its syntax. The problem is that queries that take even quadratic time in the size of a database are not feasible. It is very desirable to find a fairly rich query language such that the complexity is still clear from the syntax, but the complexities involved are feasible.

2. Show that $IQ(\leq)[f(n)]$ forms a hierarchy as $f(n)$ increases. This of course will be extremely difficult as it would imply a corresponding hierarchy result for complexity classes.

3. Prove the following conjecture: **If** $f(n)$ and $g(n)$ are reasonable functions, no larger than $2^{n^k}$, and such that $\lim_{n\to\infty}(f(n)/g(n)) = 0$ **then** $IQ[f(n)]$ is strictly contained in $IQ[g(n)]$.

4. Study and compare potential hierarchies obtained by restricting the number of distinct quantified variables and the arity of fixed points, cf. [dR84, Im82a].

5. An issue raised by Chandra and Harel among others is that languages with an ordering such as $\mathcal{L}_\mu(\leq)$ treat differently numbered isomorphic databases differently. That is, the answer to some queries will depend on the ordering. It is extremely desirable to have a language without this problem and yet still rich enough to simulate computation. One possibility would be instead of ordering to add variables ranging over $\{1 \ldots n\}$ with $\leq, +, \cdot$ available over this domain. We would also add counting quantifiers, $(\exists i x's)P(x)$, meaning that there exist $i$ $x's$ such that $P$. I am anxious to know whether or not $\mathcal{L}_\mu$ plus counting quantifiers is equal to polynomial time.

**Acknowledgements:** The preliminary version of the present paper appeared as [Im82b]. Sometime later my only electronic copy of the revised paper was destroyed in a disk crash. After the referees received the first draft of the present paper they took quite a while to decide that the proofs were too sketchy. They were certainly right, but it took me a little while to get back to revising it. Over this period there were many people who have read drafts in various states and given helpful comments. I apologize for those I've forgotten, but people who come to mind include: Yuri Gurevich, Evangelos Kranakis, John Mitchell, Adi Shamir, and Venkataraman. I would especially like to thank David Harel for his continual urging of me, in his role as editor, to get this paper out. Thanks to Michel de Rougement for his observation that without something like the 'Constant Assumption' Theorems 1 and 2 are false. Thanks to Yiannis Moschovakis who once talked to me for over an hour about the results in [Im82b], and never mentioned that the proofs could be greatly simplified by referring to his book, (as I have done in the present paper.)

# References

[1] A.V. Aho, J.E. Hopcroft and J.D. Ullman, *The Design and Analysis of Computer Algorithms,* Addison- Wesley, 1974.

[2] A.V. Aho and J.D. Ullman, "Universality of Data Retrieval Languages," *Sixth Symp. on Principles of Programming Languages,* 1979, (110-117).

[3] Ashok Chandra, "Programming Primitives for Database Languages," *8th Symp. on Principles of Programming Languages,* 1981, (50-62).

[4] Ashok Chandra and David Harel, "Computable Queries for Relational Databases," *JCSS* **21**, No. 2, October, 1980, (156-178).

[5] Ashok Chandra and David Harel, "Structure and Complexity of Relational Queries," *21st Symp. on Foundations of Computer Science,* 1980, (333-347). Also appeared in *JCSS* **25**, 1982, (99-128).

[6] E.F. Codd, "Relational Completeness of Database Sublanguages," in *Database Systems,* R. Rustin, ed., Prentice-Hall, 1972, (65-98).

[7] Michel de Rougemont, "Uniform Definability on Finite Structures with Successor," *16th ACM STOC Symp.,* 1984, (409-417).

[8] H. Enderton, *A Mathematical Introduction to Logic,* Academic Press, 1972.

[9] Ron Fagin, "Generalized First-Order Spectra and Polynomial-Time Recognizable Sets," *SIAM-AMS Proc. 7,* 1974, (43-73).

[10] M. Furst, J.B. Saxe, and M. Sipser, "Parity, Circuits, and the Polynomial-Time Hierarchy," *22nd IEEE FOCS Symp.*, 1981, (260-270).

[11] Yuri Gurevich and Saharon Shelah, "Fixed-Point Extensions of First-Order Logic," to appear in *26th Symp. on Foundations of Computer Science,* 1985.

[12] Neil Immerman, "Number of Quantifiers is Better than Number of Tape Cells," *JCSS* **22**, No. 3, June 1981, (65-72).

[13] Neil Immerman, "Upper and Lower Bounds for First Order Expressibility," *JCSS* **25**, No. 1, 1982, (76-98).

[14] Neil Immerman, "Relational Queries Computable in Polynomial Time," *14th ACM STOC Symp.,* 1982, (147-152).

[15] Neil Immerman, "Languages Which Capture Complexity Classes," *15th ACM STOC Symp.,* 1983, (347-354).

[16] Yiannis N. Moschovakis, *Elementary Induction on Abstract Structures,* North Holland, 1974.

[17] Michael Sipser, "Borel Sets and Circuit Complexity," *15th Symp. on Theory of Computation,* 1983, (61-69).

[18] J.D. Ullman, *Principles of Database Systems,* Computer Science Press, 1982.

[19] M. Vardi, "Complexity of Relational Query Languages," *14th Symposium on Theory of Computation,* 1982, (137-146).