On the Expressive Power of the Relational Algebra with Partially Ordered Domains

Wilfred Ng * Mark Levene † Trevor I. Fenner ‡

Abstract

Assuming data domains are partially ordered, we apply Paredaens' and Bancilhon's Theorem to examine the expressiveness of the extended relational algebra (the PORA), which allows the ordering predicate $\sqsubseteq$ to be used in formulae of the selection operator (σ). The PORA expresses exactly the set of all possible relations which are invariant under order-preserving automorphism of databases. Our main result shows that there is a one-to-one correspondence between three hierarchies of: (1) computable queries, (2) query languages and (3) partially ordered domains.

Key Words: ordered domains, expressive power, relational algebra, queries.

C.R. Categories: F.4.3, H.2.1

1 Introduction

Many naturally arising data types have an associated structure, of which domain ordering is a very important one [5, 7]. With the assumption of partially ordered domains, we examine the extra expressiveness of the relational algebra that we can gain with respect to an ordered database instance, and the relationship between ordered domains and classes of queries. Our first result, which is a generalisation of Paredaens' and Bancilhon's Theorem [8, 2], shows that the PORA expresses exactly the set of ordered relations which are invariant under order-preserving automorphisms. Our second result, which involves the notion of a meaningful computable query with respect to a given ordered domain, shows that there exist hierarchies of (1) meaningful computable queries, (2) partially ordered relational algebras and (3) partially ordered domains, and that there is a one-to-one correspondence between them.

Throughout this paper we follow the usual set notation [6]. We denote the singleton $\{A\}$ simply by A when no ambiguity arises and let id be the identity mapping on any set.

*Department of Computing, Hong Kong Polytechnic University, email: csshng@comp.polyu.edu.hk

†Department of Computer Science, University College London, University of London, email: m.levene@cs.ucl.ac.uk

‡Department of Computer Science, Birkbeck College, University of London, email: trevor@dcs.bbk.ac.uk

2 Ordered Databases and the PORA

A *partial ordering* $\sqsubseteq$ on the set S is a binary relation on S satisfying the conditions of *reflexivity*, *anti-symmetry* and *transitivity* [6]. We denote the special cases of *linear ordering* by $\leq$. At the other extreme, when S is completely unordered, $\sqsubseteq$ is just the equality predicate $=$. A *partially ordered set* (or simply an ordered set) is a structure $\langle S, \sqsubseteq \rangle$.

We assume that the readers are familiar with the basic teminology for the relational databases. [4, 9, 1]. In the ordered databases we let U be a countably infinite set of attribute names. Without loss of generality, we assume that all attributes $A \in U$ share the same domain $\langle D, \sqsubseteq \rangle$.

A relation schema (or simply a schema) R is a subset of U. A *database schema* is a finite set $\mathbf{R} = \{R_1, \ldots, R_n\}$ of relation schemas, for some $n \geq 1$.

An *ordered relation* (or simply a relation) r defined over a schema R is a finite set of tuples over R. An *ordered database* (or simply a database) over $\mathbf{R} = \{R_1, \ldots, R_n\}$ is a finite set $d = \{r_1, \ldots, r_n\}$. We call r and d an *unordered relation* and an *unordered database* if the underlying domain is unordered. Similiar comments also apply to r and d when the domain is linearly ordered.

We restrict our discussion to the *active domain*, denoted by $adom(d)$, containing only the values that appear in the database instance d; so $adom(d)$ is ordered by $\sqsubseteq$. The *possible information* of d is the countably infinite set of all relations that can be derived from $adom(d)$, denoted by $Poss(d)$, is defined by $Poss(d) = \bigcup_{i=0}^{\infty} \mathcal{P}(adom(d)^i)$.

We call f an *ordering automorphism* of $\langle S, \sqsubseteq_S \rangle$ whenever f is a permutation on S such that the ordering $\sqsubseteq_S$ is preserved. If the set $\{a \in S \mid f(a) \neq a\}$ is finite, then we call f a *finite* ordering automorphism. We denote the set of all finite ordering automorphisms of an ordered set $\langle S, \sqsubseteq_S \rangle$ by $Aut(S, \sqsubseteq_S)$, or simply $Aut(S)$ when $\sqsubseteq_S$ is clear from the context.

Definition 2.1 (Order-preserving Database Automorphism) Let h be an ordering automorphism of $\langle adom(d), \sqsubseteq \rangle$. We call h an order-preserving database automorphism if its extension to d satisfies $h(d) = d$; by this we mean that $h(r_i) = r_i$ for $1 \leq i \leq n$. We denote the set of all order-preserving database automorphisms of database d by $Aut(\sqsubseteq, d)$, or simply $Aut(d)$ when $\sqsubseteq$ is clear from the context.

It follows from Definition 2.1 that, for all partial orderings $\sqsubseteq$ and all linear orderings $\leq$, $Aut(\leq, d) = \{id\} \subseteq Aut(\sqsubseteq, d) \subseteq Aut(=, d)$. It also follows that $Aut(\sqsubseteq, d) = Aut(=, d) \cap Aut(adom(d), \sqsubseteq)$. The following example should help to clarify the meaning of $Aut(d)$.

Example 1 Let a database d contain just a single relation having 4 tuples, $r = \{xz, yz, xw, yw\}$, and let $\langle adom(d), \sqsubseteq \rangle = \langle \{w, x, y, z\}, \{x \sqsubseteq y, x \sqsubseteq z, x \sqsubseteq w\} \rangle$ We define functions: h_1 by $h_1(x) = y$, $h_1(y) = x$, $h_1(z) = z$ and $h_1(w) = w$;

h_2 by $h_2(x) = x$, $h_2(y) = z$, $h_2(z) = y$ and $h_2(w) = w$; and h_3 by $h_3(x) = x$, $h_3(y) = y$, $h_3(z) = w$ and $h_3(w) = z$. Then $h_1 \notin Aut(d)$ because, although it preserves d, it does not preserve the ordering; and $h_2 \notin Aut(d)$ because, although it preserves the ordering, it does not preserve d; however, $h_3 \in Aut(d)$ because it preserves both the ordering and d.

The partially ordered relational algebra (PORA) consists of a collection of six operators (see [1, 9]): *union*, *Cartesian product*, *difference*, *projection*, *renaming*, and lastly extended *selection* (σ_F), where the *selection formula* F is restricted to be the forms $A \sqsubseteq B$ or $A \not\sqsubseteq B$, where A, $B \in U$. A PORA expression is a well-formed expression composed of a finite number of operators in the PORA whose operands are relation schemas. We denote by E_{PORA} the set of all PORA expressions. In addition, we use E_{UORA} and E_{LORA} to represent E_{PORA} in the contexts of unordered and linearly ordered databases.

We need the following technical lemma to establish our main theorem, in which we define $Aut(r) = Aut(\{r, adom(d)\})$ for a relation r, where $adom(d)$ is regarded as a unary relation and d is understood from the context.

Lemma 2.1 Let $d = \{r_1, \ldots, r_n\}$ be a database over $\{R_1, \ldots, R_n\}$, s be the unordered relation over S given by $s = \{\langle a, b\rangle \mid a \sqsubseteq b \text{ and } a, b \in adom(d)\}$, $d' = \{r_1, \ldots, r_n, s\}$ considered as an unordered database over $\{R_1, \ldots, R_n, S\}$, and $r' = r \times s$ considered as an unordered relation over RS. Then

(a) $Aut(=, d') = Aut(\sqsubseteq, d)$,

(b) $Aut(=, r') = Aut(\sqsubseteq, r)$,

(c) $e'(d') = r'$ for some $e' \in E_{UORA}$ if and only if $e(d) = r$ for some $e \in E_{PORA}$.
□

Using our notation, we can state Paredaens' and Bancilhon's Theorem (PB Theorem) in [8] as follows, $e(d) = r$ for some $e \in E_{UORA}$ if and only if $Aut(=, d) \subseteq Aut(=, r)$, where d is an unordered database. We now show that this theorem can be generalised to ordered databases.

Theorem 2.2 Let d be an ordered database over $\mathbf{R}$ and r an ordered relation over R. Then $e(d) = r$ for some $e \in E_{PORA}$ if and only if $Aut(\sqsubseteq, d) \subseteq Aut(\sqsubseteq, r)$.

Proof.
By part (a) of Lemma 2.1 $Aut(\sqsubseteq, d) = Aut(=, d')$ and by part (b) of Lemma 2.1 $Aut(\sqsubseteq, r) = Aut(=, r')$. So $Aut(\sqsubseteq, d) \subseteq Aut(\sqsubseteq, r)$ if and only if $Aut(=, d') \subseteq Aut(=, r')$. By PB Theorem $Aut(=, d') \subseteq Aut(=, r')$ if and only if $e'(d') = r'$ for some $e' \in E_{UORA}$. The result then follows by part (c) of Lemma 2.1 that $Aut(\sqsubseteq, d) \subseteq Aut(\sqsubseteq, r)$ if and only if $e(d) = r$ for some $e \in E_{PORA}$. □

Corollary 2.3 Let d be a linearly ordered database. Then, for all $r \in Poss(d)$, $e(d) = r$ for some $e \in E_{LORA}$. □

3 A Hierarchy of Computable Queries

We use an index subscript to denote different orderings on D, i.e., $\mathcal{D}_i = \langle D, \sqsubseteq_i \rangle$ where i is a positive integer. The semantics of "more ordered" domains can be defined in terms of ordering automorphisms of domains.

Definition 3.1 (More Ordered Domain) A domain $\mathcal{D}_2$ is said to be *more ordered* than another domain $\mathcal{D}_1$, denoted by $\mathcal{D}_1 \preceq \mathcal{D}_2$, if, for all finite $S \subseteq D$, $Aut(S, \sqsubseteq_2) \subseteq Aut(S, \sqsubseteq_1)$.

Note that the above definition corresponds to the intuition of more ordered. The informal reason for allowing $S \subseteq D$ in the above definition is that we take into account the fact that the active domain of a database can be defined on any subset of D. As a consequence of the definition, $Aut(d)$ is not affected by the automorphisms induced from outside the active domain.

Now we consider the expressiveness of the relational algebra for different orderings. Let the set of relations generated from the information contained in a given database d, denoted by $Gen(\sqsubseteq_i, d)$, be defined by the following expression

$$Gen(\sqsubseteq_i, d) = \{r \mid r = e(d) \text{ for some } e \in E_{PORA_i}\}.$$

Definition 3.2 (More Powerful Relational Algebra) A relational algebra $PORA_2$ is *more powerful* than another $PORA_1$, denoted by $PORA_1 \preceq PORA_2$, if, for all databases d, $Gen(\sqsubseteq_1, d) \subseteq Gen(\sqsubseteq_2, d)$.

We still need to extend the notion of computable query to ordered databases [3]. The motivation for our definition is to include those queries which are meaningful with respect to the ordered domain concerned.

Let $DB(\mathbf{R})$ be the countably infinite set of all databases defined over a database schema $\mathbf{R}$ and let $\chi = \bigcup_{i=0}^{\infty} \mathcal{P}(D^i)$.

Definition 3.3 (Meaningful Computable Query) A *meaningful computable query* with respect to a given domain $\mathcal{D}_i$ is a partial recursive function δ from $DB(\mathbf{R})$ to χ, for some database schema $\mathbf{R}$, such that, for all $d \in DB(\mathbf{R})$,

(a) if $\delta(d)$ is defined, then $\delta(d) \in Poss(d)$, and

(b) for all $h \in Aut(\sqsubseteq_i, d)$, $h(\delta(d)) = \delta(d)$.

We denote the set of all meaningful computable queries by Q_i.

Note that our definition of a meaningful computable query is the same as the conventional one if we restrict ourselves to unordered domains.

Lemma 3.1 Let $d = \{r_1, \ldots, r_n\}$ be a database over $\{R_1, \ldots, R_n\}$, s be the unordered relation over S given by $s = \{\langle a, b\rangle \mid a \sqsubseteq b \text{ and } a, b \in adom(d)\}$, and $r = r_1 \times \cdots \times r_n \times s$, considered as an unordered relation over $R_1 \cdots R_n S$. Then $Aut(\sqsubseteq, d) = Aut(=, r)$. □

Lemma 3.2 For any database schema $\mathbf{R}$, $\mathcal{D}_1 \preceq \mathcal{D}_2$ if and only if $Aut(\sqsubseteq_2, d) \subseteq Aut(\sqsubseteq_1, d)$ for all databases d over $\mathbf{R}$. $\square$

We now present our main result stating the association between domains, queries and languages. This allows us to establish hierarchies for these entities.

Theorem 3.3

(a) $\mathcal{D}_1 \preceq \mathcal{D}_2$ if and only if $Q_1 \subseteq Q_2$,

(b) $\mathcal{D}_1 \preceq \mathcal{D}_2$ if and only if $PORA_1 \preceq PORA_2$.

Proof.
(a) *(If)* Assume $\mathcal{D}_1 \not\preceq \mathcal{D}_2$. By Lemma 3.2, this implies that there exists a database d' such that $h_2 \notin Aut(\sqsubseteq_1, d')$ for some $h_2 \in Aut(\sqsubseteq_2, d')$. Let $d' = \{r'_1, \ldots, r'_n\}$. We now construct a query that is in Q_1 but not in Q_2. Given d', consider r' where we substitute d' for d and r' for r in Lemma 3.1, with respect to $\sqsubseteq_1$. Thus, for all $h \in Aut(\sqsubseteq_1, d')$, we have $h(r') = r'$. On the other hand, $h_2(r') \neq r'$ since $h_2 \notin Aut(\sqsubseteq_1, d')$. We define a query δ as follows: $\delta(d) = r'$ when $d = d'$ and $\delta(d)$ is equal to the empty set otherwise. By part (b) of Definition 3.3, $\delta \in Q_1$ but $\delta \notin Q_2$.

(Only if) Let $\delta \in Q_1$ be a query from $DB(\mathbf{R})$ to χ and let $d \in DB(\mathbf{R})$. From Definition 3.3, $\delta(d) \in Poss(d)$ and, for all $h \in Aut(\sqsubseteq_1, d)$, $h(\delta(d)) = \delta(h(d))$. By the assumption $\mathcal{D}_1 \preceq \mathcal{D}_2$ and Lemma 3.2, $Aut(\sqsubseteq_2, d) \subseteq Aut(\sqsubseteq_1, d)$. Therefore, for all $h \in Aut(\sqsubseteq_2, d)$, $h(\delta(d)) = \delta(d)$ and thus $\delta \in Q_2$.

(b) *(If)* Assume $\mathcal{D}_1 \not\preceq \mathcal{D}_2$. By Lemma 3.2, there exists a database $d' = \{r_1, \ldots, r_n\}$ such that $Aut(\sqsubseteq_2, d') \not\subseteq Aut(\sqsubseteq_1, d')$. It suffices to exhibit a database d and a relation r such that $r \in Gen(\sqsubseteq_1, d)$ but $r \notin Gen(\sqsubseteq_2, d)$. We let $d = d'$ and $r = r_1 \times \cdots \times r_n \times s$, where $s = \{\langle a, b \rangle \mid a \sqsubseteq_1 b \text{ and } a, b \in adom(d')\}$. Clearly, s can be derived from d' by some $e \in PORA_1$ and thus $r \in Gen(\sqsubseteq_1, d')$. It remains to show $r \notin Gen(\sqsubseteq_2, d')$. Suppose $r \in Gen(\sqsubseteq_2, d')$. Then, by Theorem 2.2, $Aut(\sqsubseteq_2, d') \subseteq Aut(\sqsubseteq_2, r)$, so $Aut(\sqsubseteq_2, d') \subseteq Aut(=, r)$. By Lemma 3.1, it follows that $Aut(\sqsubseteq_2, d') \subseteq Aut(\sqsubseteq_1, d')$, which leads to a contradiction.

(Only if) Let $r \in Gen(\sqsubseteq_1, d)$. We need to show that $r \in Gen(\sqsubseteq_2, d)$. By Theorem 2.2, $Aut(\sqsubseteq_1, d) \subseteq Aut(\sqsubseteq_1, r)$. Thus

$$Aut(adom(d), \sqsubseteq_2) \cap Aut(\sqsubseteq_1, d) \subseteq Aut(adom(d), \sqsubseteq_2) \cap Aut(\sqsubseteq_1, r).$$

Moreover, we have $Aut(\sqsubseteq_1, d) = Aut(adom(d), \sqsubseteq_1) \cap Aut(=, d)$ and $Aut(\sqsubseteq_1, r) = Aut(adom(d), \sqsubseteq_1) \cap Aut(=, r)$. It follows that

$$\begin{aligned} &Aut(adom(d), \sqsubseteq_2) \cap Aut(adom(d), \sqsubseteq_1) \cap Aut(=, d) \subseteq \\ &\qquad Aut(adom(d), \sqsubseteq_2) \cap Aut(adom(d), \sqsubseteq_1) \cap Aut(=, r). \end{aligned}$$

By assumption $\mathcal{D}_1 \preceq \mathcal{D}_2$, $Aut(adom(d), \sqsubseteq_2) \subseteq Aut(adom(d), \sqsubseteq_1)$. It follows that $Aut(adom(d), \sqsubseteq_2) \cap Aut(=, d) \subseteq Aut(adom(d), \sqsubseteq_2) \cap Aut(=, r)$. Hence we have $Aut(\sqsubseteq_2, d) \subseteq Aut(\sqsubseteq_2, r)$. By Theorem 2.2 again, we have $r \in Gen(\sqsubseteq_2, d)$. $\square$

Corollary 3.4 $Q_1 \subseteq Q_2$ if and only if $PORA_1 \preceq PORA_2$. $\square$

4 Conclusions

We present the following diagram which summarises the relationship between the hierarchies of (1) meaningful computable queries, (2) partially ordered domains, and (3) partially ordered relational algebras. The implications of this result are that when the underlying data domain of an ordered database has more inherent structure, then the scope of possible queries is wider. In other words, the ordered relational model can provide more expressive query languages than those of the conventional one. There remains the problem of trying to find a more convenient characterisation of the domain ordering $\preceq$ without explicitly involving the set of ordering automorphisms. The semantics of $\preceq$ may be defined in terms of the relationship of the structural features between two ordered domains, leading to the syntactical insight of the notion of "more ordered".

Queries	$Q_= \subseteq$	$\ldots$	$\subseteq Q_i \subseteq$	$\ldots$	$\subseteq Q_\leq$
	$\updownarrow$		$\updownarrow$		$\updownarrow$
Domains	$\langle D, = \rangle \preceq$	$\ldots$	$\preceq \langle D, \sqsubseteq_i \rangle \preceq$	$\ldots$	$\preceq \langle D, \leq \rangle$
	$\updownarrow$		$\updownarrow$		$\updownarrow$
Algebras	$PORA_= \preceq$	$\ldots$	$\preceq PORA_i \preceq$	$\ldots$	$\preceq PORA_\leq$

Figure 1: Correspondence between hierarchies of queries, domains and languages

References

[1] P. Atzeni and V. De Antonellis. *Relational Database Theory*. Benjamin/Cummings Publishing Company, Inc., (1993).

[2] F. Bancilhon. On the Completeness of Query Languages for Relational Databases. In *LNCS 64: Mathematical Foundations of Computer Science*, Springer-Verlag, pp. 112-124, (1978).

[3] A.K. Chandra and D. Harel. Computable Queries for Relational Data Bases. *Journal of Computer and System Sciences* **21**, pp. 156-178, (1980).

[4] E.F. Codd. A Relational Model of Data for Large Shared Data Banks. *Communications of the ACM* **13**(6), pp. 377-387, (1970).

[5] S. Ginsburg and R. Hull. Sort Sets in the Relational Model. *Journal of the Association for Computing Machinery* **33**(3), pp. 465-488, (1986).

[6] P. Halmos. *Naive Set Theory,* Springer-Verlag, New York, (1974).

[7] D. Maier and B. Vance. A Call to Order, In *ACM symp. on Principles of Databases Systems.* pp. 1-16, (1993).

[8] J. Paredaens. On the Expressive Power of the Relational Algebra. *Information Processing Letter* **7**(2), pp. 107-111, (1978).

[9] J.D. Ullman. *Principles of Database and Knowledge-Base Systems, Vol. I*, Rockville, MD., Computer Science Press, (1988).