RAIRO-Theor. Inf. Appl. 43 (2009) 179–187
DOI: 10.1051/ita:2008023

Available online at:
www.rairo-ita.org

REPETITIONS AND PERMUTATIONS OF COLUMNS IN THE SEMIJOIN ALGEBRA

Dirk Leinders[1] and Jan Van Den Bussche[1]

Abstract. Codd defined the relational algebra [E.F. Codd, *Communications of the ACM* **13** (1970) 377–387; E.F. Codd, Relational completeness of data base sublanguages, in *Data Base Systems*, R. Rustin, Ed., Prentice-Hall (1972) 65–98] as the algebra with operations projection, join, restriction, union and difference. His projection operator can drop, permute and repeat columns of a relation. This permuting and repeating of columns does not really add expressive power to the relational algebra. Indeed, using the join operation, one can rewrite any relational algebra expression into an equivalent expression where no projection operator permutes or repeats columns. The fragment of the relational algebra known as the semijoin algebra, however, lacks a full join operation. Nevertheless, we show that any semijoin algebra expression can still be simulated in a natural way by a set of expressions where no projection operator permutes or repeats columns.

Mathematics Subject Classification. 68P15.

1. Introduction

In the 1970s Codd introduced the now standard relational data model, in which a database is a finite collection of relations, where a relation is a finite set of tuples. To express queries in the relational model, Codd introduced the relational algebra with operators selection (called restriction by Codd), projection, union, difference and join [3]. Since then the relational algebra has been extensively studied [1]. A very important result is that its expressive power is equivalent to the expressive power of first-order logic, called relational calculus in database theory [4].

The projection operator of Codd's relational algebra can permute and repeat columns. This permuting and repeating of columns, however, does not add expressive power to the relational algebra. Indeed, the two existing perspectives on

Keywords and phrases. Database, relational algebra, semijoin algebra, projection.

[1] Hasselt University and Transnational University of Limburg Agoralaan, gebouw D, 3590 Diepenbeek Belgium; dirk.leinders@uhasselt.be

Article published by EDP Sciences

© EDP Sciences 2008

the relational model, namely the *named perspective*, in which tuples are viewed as functions from the set of attributes to the domain, and the *unnamed perspective*, in which tuples are viewed as ordered lists of domain values, are equivalent [1], whereas permuting and repeating of columns can not be done in the named perspective. For completeness, we will explicitly show in this paper that any relational algebra expression can be rewritten into an equivalent relational algebra expression where no projection operator permutes or repeats columns.

The semijoin algebra is a natural fragment of the relational algebra, obtained by replacing the join operator by the semijoin operator [11]. The semijoin of two relations is equivalent to a join projected on the attributes or columns of the first relation. While in the full relational algebra permuting and repeating of columns does not add expressive power, this is not clear for the semijoin algebra. Indeed, the rewrite rule to replace a permuting or repeating projection in a relational algebra expression with a non-permuting and non-repeating one uses the join operator, that the semijoin algebra lacks. Nevertheless, in this paper we show that any semijoin algebra expression can still be simulated by semijoin algebra expressions where no projection operator permutes or repeats columns. The notion of "simulation", however, becomes more complicated. The idea is that given an arbitrary expression E, one can produce a set of permutation- and repetition-free expressions that return the relevant values of the output tuples of E, up to certain repetitions and permutations which are produced as a by-product of the translation. In particular, for boolean expressions, there is always a single equivalent boolean expression that is permutation- and repetition-free.

Our interest to study the semijoin algebra and its relationship to the full relational algebra is motivated by previous work that clearly indicates the importance and relevance of the semijoin algebra. For instance, whereas the relational algebra is equivalent in expressive power to first-order logic, the semijoin algebra is equivalent in expressive power to the guarded fragment of first-order logic [10]. This fragment was introduced by Andréka, van Benthem and Németi [2]. It has been studied extensively since its introduction [5–8] and a lot of nice properties were obtained, such as the finite model property, that are thus inherited by the semijoin algebra. Apart from this connection to the guarded fragment, the semijoin algebra has the interesting property that it can express every relational algebra expression that always produces intermediate results of size linear in the size of the database [9].

2. Preliminaries

In this section, we give the definitions necessary to formally state and prove our result. We first define a number of variants of the relational algebra and the semijoin algebra obtained by restricting the projection operator in different levels.

From the outset, we assume a universe $\mathbb{U}$ of basic data values, over which a number of predicates are defined. The names of these predicates and their arities are collected in the vocabulary Ω. The equality predicate $(=)$ is always

in Ω. Quantifier-free logic formulas over Ω will be used in the relational algebra as selection or join conditions.

A database schema is a finite set $\mathbf{S}$ of relation names, each associated with its arity. $\mathbf{S}$ is disjoint from Ω. A database D over $\mathbf{S}$ is an assignment of a finite relation $D(R) \subseteq \mathbb{U}^n$ to each $R \in \mathbf{S}$, where n is the arity of R.

Definition 2.1. Let $\mathbf{S}$ be a database schema. Syntax and semantics of the relational algebra (RA) are inductively defined as follows:

(1) Each relation name $R \in \mathbf{S}$ belongs to RA.
(2) If $E_1, E_2 \in$ RA have arity n, then $E_1 \cup E_2$, $E_1 - E_2$ belong to RA and are of arity n.
(3) If $E_1 \in$ RA has arity n and $i_1, \ldots, i_k$ are elements of $\{1, \ldots, n\}$, then $\pi_{i_1,\ldots,i_k}(E_1)$ belongs to RA and is of arity k.
(4) If $E_1 \in$ RA has arity n and $\theta(x_1, \ldots, x_n)$ is a quantifier-free formula over Ω, then $\sigma_\theta(E_1)$ belongs to RA and is of arity n.
(5) If $E_1, E_2 \in$ RA have arities n and m, respectively, and $\theta(x_1, \ldots, x_n, y_1, \ldots, y_m)$ is a quantifier-free formula over Ω, then $E_1 \bowtie_\theta E_2$ belongs to RA and is of arity $n + m$.

Let E be an RA expression over $\mathbf{S}$ and let D be a database over $\mathbf{S}$. Then the result of E on D, denoted $E(D)$, is defined inductively as follows:

(1) $R(D) := D(R)$.
(2) $E_1 \cup E_2(D) := E_1(D) \cup E_2(D)$, $E_1 - E_2(D) := E_1(D) - E_2(D)$.
(3) $\pi_{i_1,\ldots,i_k} E_1(D) := \{(a_{i_1}, \ldots, a_{i_k}) \mid (a_1, \ldots, a_n) \in E_1(D)\}$.
(4) $\sigma_{\theta_1} E_1(D) := \{\bar{a} \in E_1(D) \mid \theta_1(\bar{a}) \text{ holds}\}$.
(5) $E_1 \bowtie_{\theta_2} E_2(D) := \{(\bar{a}, \bar{b}) \mid \bar{a} \in E_1(D), \bar{b} \in E_2(D), \text{ and } \theta_2(\bar{a}, \bar{b}) \text{ holds}\}$.

Remark 2.2. The Cartesian product operator $\times$ is expressible in our setting as a join where the condition θ is the formula true. The intersection operator $\cap$ is expressible in our setting using the projection and the join operator: $R \cap S = \pi_{1,\ldots,\text{arity}(R)}(R \bowtie_\theta S)$, where θ is the formula $\bigwedge_{i=1}^{\text{arity}(R)} x_i = y_i$.

Example 2.3. Let $\mathbf{S}$ be the schema containing a single binary relation Knows. Then the expression $\text{Knows} \cap \pi_{2,1}(\text{Knows})$ defines all pairs of persons who know each other.

Definition 2.4. The semijoin algebra (SA) is the variant of RA obtained by replacing the join operator $\bowtie_\theta$ by the semijoin operator $\ltimes_\theta$. The semantics of the semijoin operator is as follows:

$$E_1 \ltimes_\theta E_2(D) := \{\bar{a} \in E_1(D) \mid \exists \bar{b} \in E_2(D) \text{ such that } \theta(\bar{a}, \bar{b}) \text{ holds}\}.$$

Example 2.5. Consider again the expression $\text{Knows} \cap \pi_{2,1}(\text{Knows})$ of Example 2.3. This expression is also an SA expression. Indeed, according to Remark 2.2 the intersection operator can be expressed as a projection of the join onto one of the relations, which has exactly the same semantics as a semijoin. The expression can thus be written in SA as $\text{Knows} \underset{\substack{x_1=y_1 \\ x_2=y_2}}{\ltimes} \pi_{2,1}\text{Knows}$.

The restrictions RA^{-r} and SA^{-r} of RA and SA, respectively, are obtained by restricting the projection operator $\pi_{i_1,\ldots,i_k}$ by requiring that the numbers $i_1,\ldots,i_k$ are all different. So, repeating columns in the projection list is not allowed. The restrictions RA^{-rp} and SA^{-rp} are obtained by requiring the projection list $i_1,\ldots,i_k$ to be strictly increasing, i.e., $i_1 < \cdots < i_k$. So, permutations are not allowed. Note that this also excludes repeating columns.

Example 2.6. The expression in Example 2.3 is an example of an SA^{-r} (and RA^{-r}) expression. It is not SA^{-rp} (nor RA^{-rp}), but it can be equivalently written in SA^{-rp} (and RA^{-rp}) as $\mathrm{Knows} \underset{\substack{x_1=y_2\\x_2=y_1}}{\ltimes} \mathrm{Knows}$ (*cf.* Ex. 2.5).

Finally, a notation: Let f be a function from $\{1,\ldots,m\}$ to $\{1,\ldots,n\}$ and let $\bar{a}$ be an n-tuple. Then $f(\bar{a})$ is the m-tuple $(a_{f(1)},\ldots,a_{f(m)})$.

3. Main result

In this section, we show our main result: allowing permuting and repeating of columns in projections does not add expressive power to the semijoin algebra. Note that this property is clear for the full relational algebra. Indeed, if R is a relation of arity n and $i_1,\ldots,i_k$ are values between 1 and n, then $\pi_{i_1,\ldots,i_k}R$ is equivalent to the RA^{-rp} expression

$$\pi_{f(1),\ldots,f(k)}\Big(\underbrace{((R \underset{\theta}{\bowtie} R) \underset{\theta}{\bowtie} \ldots) \underset{\theta}{\bowtie} R}_{k \text{ times } R}\Big)$$

where $f(j)$ is $(j-1)n + i_j$ and θ is $x_1 = y_1 \wedge \ldots \wedge x_n = y_n$.

Example 3.1. The RA^{-r} expression $\pi_{2,1}(\mathrm{Knows})$ can be expressed in RA^{-rp} as $\pi_{2,3}(\mathrm{Knows} \underset{\substack{x_1=y_1\\x_2=y_2}}{\bowtie} \mathrm{Knows})$.

This trick for RA does not work for SA, where we do not have the full join. Indeed, a projection with repetitions like $\pi_{1,1}R$ cannot be equivalently expressed in SA^{-rp}. The same holds for a nonincreasing projection like $\pi_{2,1}R$. Nevertheless, for any SA expression E that can use projections with arbitrary repetitions and permutations, we can still obtain the tuples returned by E, by means of SA^{-rp} expressions, as follows:

Theorem 3.2. *Let E be an SA expression of arity n. Then there exists a set P of pairs of the form (F, f), where F is an SA^{-rp} expression and f is a function from $\{1,\ldots,n\}$ to $\{1,\ldots,\ell\}$ with ℓ the arity of F, such that for each database D:*

$$E(D) = \bigcup_{(F,f)\in P} \{f(\bar{a}) \mid \bar{a} \in F(D)\}.$$

Before we prove the theorem, we give an example and make a remark.

Example 3.3. If E is the expression $\pi_{2,1,1}R$, then a set P according to Theorem 3.2 is the singleton $\{(R, f)\}$, where f is the function from $\{1,2,3\}$ to $\{1,2\}$ with $f(1) = 2$ and $f(2) = f(3) = 1$.

If E is the expression $(\pi_{2,1,2}R) \ltimes_{2=1} (\pi_{3,1}S)$, then a set P according to Theorem 3.2 is the singleton $\{(R \ltimes_{1=3} S, f)\}$, where $f(1) = f(3) = 2$ and $f(2) = 1$.

Remark 3.4. To see why in general P contains more than one element, consider the schema $\{R, S\}$, where R and S are binary relations. Let E be $R \cup \pi_{2,1}(S)$. Consider database D:

$$R(D) := \{(1,2)\}$$
$$S(D) := \{(3,4)\}.$$

Let P be the singleton $\{(F, f)\}$, where F is an SA^{-rp} expression and f is a function from $\{1,2\}$ to the set X, which can be either $\{1\}$ or $\{1,2\}$. If X is $\{1\}$, then $f(1) = f(2) = 1$. It is clear that in this case each tuple (x, y) in the set $\{f(\bar{a}) \mid \bar{a} \in F(D)\}$ has $x = y$. If X is $\{1,2\}$, then f can be the identical function or the permutation switching 1 and 2, *i.e.*, $f(1) = 2$ and $f(2) = 1$. An easy inductive argument shows that each tuple (x, y) in the result of a binary SA^{-rp} expression F on database D will have $x < y$. Therefore, either each tuple (x, y) in the set $\{f(\bar{a}) \mid \bar{a} \in F(D)\}$ will have $x < y$ (if f is the identical function), or each tuple (x, y) in that set will have $x > y$ (if f permutes 1 and 2). In the set $E(D) = \{(1,2),(4,3)\}$, however, none of these three properties hold.

Proof. The construction of the set P and the correctness proof are by structural induction. We will write P_E to denote that the set P corresponds to expression E.

(1) If $E = R$, then $P_E := \{(R, \mathrm{Id})\}$, where Id denotes the identity function.
(2) If $E = \sigma_\theta E_1$, then $P_E := \{(\sigma_{\theta_f} F, f) \mid (F, f) \in P_{E_1}\}$, where $\theta_f := \theta[x_i/x_{f(i)}]$. In proof:

$$\begin{aligned}
\bar{a} \in \sigma_\theta E_1(D) \quad &\Leftrightarrow \quad \bar{a} \in E_1(D) \text{ and } \bar{a} \text{ satisfies } \theta \\
&\Leftrightarrow \quad \bar{a} \in \bigcup_{(F,f)\in P_{E_1}} \{f(\bar{b}) \mid \bar{b} \in F(D)\} \quad \text{and} \quad \bar{a} \text{ satisfies } \theta \\
&\Leftrightarrow \quad \bar{a} \in \bigcup_{(F,f)\in P_{E_1}} \{f(\bar{b}) \mid \bar{b} \in \sigma_{\theta_f} F(D)\}.
\end{aligned}$$

(3) If $E = \pi_{i_1,\ldots,i_n} E_1$, then

$$P_E := \{(F, f'_{i_1,\ldots,i_n}) \mid (F, f) \in P_{E_1}\}$$

where $f'_{i_1,\ldots,i_n}$ is the function mapping j to $f(i_j)$ for all $1 \leq j \leq n$. In proof:

$$\begin{aligned}
&\bar{a} \in \pi_{i_1,\ldots,i_n} E_1(D) \\
&\quad \Leftrightarrow \quad \exists\, \bar{b} \in E_1(D) \text{ such that } (a_1,\ldots,a_n) = (b_{i_1},\ldots,b_{i_n}) \\
&\quad \Leftrightarrow \quad \exists\, \bar{b} \in \bigcup_{(F,f)\in P_{E_1}} \{f(\bar{c}) \mid \bar{c} \in F(D)\} \\
&\qquad\qquad \text{such that } (a_1,\ldots,a_n) = (b_{i_1},\ldots,b_{i_n}) \\
&\quad \Leftrightarrow \quad \bar{a} \in \bigcup_{(F,f)\in P_{E_1}} \{f'_{i_1,\ldots,i_n}(\bar{c}) \mid \bar{c} \in F(D)\} \\
&\qquad\qquad \text{where } f'_{i_1,\ldots,i_n} \text{ is defined as above.}
\end{aligned}$$

(4) If $E = E_1 \cup E_2$, then $P_E := P_{E_1} \cup P_{E_2}$.
(5) If $E = E_1 - E_2$, then

$$P_E := \{(F_1 - \bigcup_{(F_2,f_2)\in P_{E_2}} F_1 \ltimes_{\theta_{f_1\cap f_2}} F_2, f_1) \mid (F_1,f_1) \in P_{E_1}\}$$

where $\theta_{f_1\cap f_2} := \bigwedge_{i=1}^{n} x_{f_1(i)} = y_{f_2(i)}$. In proof:

$$\begin{aligned}
&\bar{a} \in E_1 - E_2(D) \\
&\quad \Leftrightarrow \quad \bar{a} \in \bigcup_{(F,f)\in P_{E_1}} \{f(\bar{b}) \mid \bar{b} \in F(D)\} - \bigcup_{(F,f)\in P_{E_2}} \{f(\bar{c}) \mid \bar{c} \in F(D)\} \\
&\quad \Leftrightarrow \quad \exists (F_1,f_1) \in P_{E_1},\, \exists \bar{b} \in F_1(D) \;:\; \bar{a} = f_1(\bar{b}) \\
&\qquad\quad \text{and } \forall (F_2,f_2) \in P_{E_2},\, \forall \bar{c} \in F_2(D) \;:\; \bar{a} \neq f_2(\bar{c}) \\
&\quad \Leftrightarrow \quad \exists (F_1,f_1) \in P_{E_1},\, \exists \bar{b} \in F_1(D) \;:\; \bar{a} = f_1(\bar{b}) \\
&\qquad\quad \text{and } \forall (F_2,f_2) \in P_{E_2},\, \forall \bar{c} \in F_1 \ltimes_{\theta_{f_1\cap f_2}} F_2(D) \;:\; \bar{a} \neq f_1(\bar{c}) \\
&\quad \Leftrightarrow \quad \exists (F_1,f_1) \in P_{E_1},\, \exists \bar{b} \in F_1 - \bigcup_{(F_2,f_2)\in P_{E_2}} F_1 \ltimes_{\theta_{f_1\cap f_2}} F_2(D) \;:\; \bar{a} = f_1(\bar{b}) \\
&\quad \Leftrightarrow \quad \bar{a} \in \bigcup_{(F_1,f_1)\in P_{E_1}} \big\{ f_1(\bar{b}) \mid \bar{b} \in F_1 - \bigcup_{(F_2,f_2)\in P_{E_2}} F_1 \ltimes_{\theta_{f_1\cap f_2}} F_2(D) \big\}.
\end{aligned}$$

(6) If $E = E_1 \ltimes_\theta E_2$, then

$$P_E := \{(F_1 \ltimes_{\theta_{f_1 f_2}} F_2, f_1) \mid (F_1,f_1) \in P_{E_1}, (F_2,f_2) \in P_{E_2}\}$$

where $\theta_{f_1 f_2} := \theta[x_i/x_{f_1(i)}, y_j/y_{f_2(j)}]$. In proof:

$$
\begin{aligned}
&\bar{a} \in E_1 \ltimes_\theta E_2(D)\\
&\quad\Leftrightarrow\quad \bar{a} \in E_1(D) \text{ and } \exists \bar{b} \in E_2(D) \text{ such that } \theta(\bar{a},\bar{b}) \text{ holds}\\
&\quad\Leftrightarrow\quad \bar{a} \in \bigcup_{(F_1,f_1)\in P_{E_1}} \{f_1(\bar{c}) \mid \bar{c} \in F_1(D)\}\\
&\qquad\qquad \text{and } \exists \bar{b} \in \bigcup_{(F_2,f_2)\in P_{E_2}} \{f_2(\bar{d}) \mid \bar{d} \in F_2(D)\} \text{ such that } \theta(\bar{a},\bar{b}) \text{ holds}\\
&\quad\Leftrightarrow\quad \exists (F_1,f_1) \in P_{E_1},\ \exists \bar{c} \in F_1(D)\ :\ \bar{a} = f_1(\bar{c})\\
&\qquad\qquad \text{and } \exists (F_2,f_2) \in P_{E_2},\ \exists \bar{d} \in F_2(D),\ \exists \bar{b}\ :\ \bar{b} = f_2(\bar{d})\\
&\qquad\qquad \text{such that } \theta(f_1(\bar{c}), f_2(\bar{d})) \text{ holds}\\
&\quad\Leftrightarrow\quad \bar{a} \in \bigcup_{\substack{(F_1,f_1)\in P_{E_1}\\(F_2,f_2)\in P_{E_2}}} \{f_1(\bar{c}) \mid \bar{c} \in F_1 \ltimes_{\theta_{f_1 f_2}} F_2(D)\}.
\end{aligned}
$$

This concludes our proof. □

If one is only interested in "boolean" queries, *i.e.*, yes/no properties of databases, which is often the case in practice, *e.g.*, integrity constraints or decision queries, then we can strengthen our simulation result into a full equivalence result:

Corollary 3.5. *Let E be an SA expression of arity n. Then there exists an SA^{-rp} expression E' such that for each database D:*

$$E(D) \neq \emptyset \quad\Leftrightarrow\quad E'(D) \neq \emptyset.$$

Proof. From Theorem 3.2, it follows that $E(D) \neq \emptyset$ if and only if for some pair (F, f) in the set P_E, we have: $F(D) \neq \emptyset$. Note that each F is an SA^{-rp} expression. Expression E' is now defined as

$$\bigcup_{(F,f)\in P_E} \pi_{()} F,$$

where $()$ is the empty projection list. □

4. Complexity issues

The algorithm in the proof of Theorem 3.2 has an exponential worst-case complexity. In order to make this statement precise, define $\mathrm{size}(E)$, for an SA expression E, as the number of operators in E. Furthermore, for a set P of pairs as in Theorem 3.2, define $\mathrm{size}(P)$ as the sum of the sizes of the SA^{-rp} expressions F in P, *i.e.*, $\mathrm{size}(P) = \sum_{(F,f)\in P} \mathrm{size}(F)$. We then have:

Proposition 4.1. *Let E be an SA expression and let P_E be the set constructed by the algorithm in the proof of Theorem 3.2. Then,* $\text{size}(P_E) \leq 2^{3\cdot\text{size}(E)}$.

Proof. We simultaneously show by structural induction that $\text{size}(P_E) \leq 2^{3\cdot\text{size}(E)}$ and $|P_E| \leq 2^{\text{size}(E)}$. Here, we only present the case where $E = E_1 - E_2$. The other cases are similar but easier. For $E = E_1 - E_2$, we have

$$|P_E| = |P_{E_1}| \leq 2^{\text{size}(E_1)} \leq 2^{\text{size}(E_1)+\text{size}(E_2)+1} = 2^{\text{size}(E)},$$

and

$$\begin{aligned}
\text{size}(P_E) &= \text{size}(P_{E_1}) + |P_{E_1}| + \text{size}(P_{E_1}) \cdot |P_{E_2}| \\
&\quad + \text{size}(P_{E_2}) \cdot |P_{E_1}| + 2 \cdot |P_{E_1}| \cdot |P_{E_2}| \\
&\leq 2^{3\cdot\text{size}(E_1)} + 2^{\text{size}(E_1)} + 2^{3\cdot\text{size}(E_1)+\text{size}(E_2)} \\
&\quad + 2^{3\cdot\text{size}(E_2)+\text{size}(E_1)} + 2^{\text{size}(E_1)+\text{size}(E_2)+1} \\
&\leq 2^2 \cdot 2^{3\cdot\text{size}(E_1)+3\cdot\text{size}(E_2)} + 2^{\text{size}(E_1)+\text{size}(E_2)+1} \\
&\leq 2^{3\cdot\text{size}(E_1)+3\cdot\text{size}(E_2)+3} \\
&= 2^{3\cdot\text{size}(E)}.
\end{aligned}$$

□

This upper bound is sharp. Indeed, let E be the expression

$$\Big(\big((\pi_{2,1}R - S_1) - S_2\big) - \cdots\Big) - S_n,$$

where R and S_i are binary relations for all i. Then the set P_E constructed by the algorithm in the proof of Theorem 3.2 is the singleton $\{(E_n, f)\}$, where $f(1) = 2$ and $f(2) = 1$, and where E_n is inductively defined as follows:

$$\begin{aligned}
E_1 &:= R - R \underset{\substack{2=1\\1=2}}{\ltimes} S_1 \\
E_{i+1} &:= E_i - E_i \underset{\substack{2=1\\1=2}}{\ltimes} S_{i+1} \quad (\text{for } 1 < i \leq n).
\end{aligned}$$

Clearly, the size of E_n is exponential in the size of E.

For this particular expression E, however, there is a set P'_E of pairs satisfying the conditions in Theorem 3.2 of size polynomial in the size of E. Indeed, note that expression E is equivalent to the expression $\pi_{2,1}F$, where F is

$$R - R \underset{\substack{2=1\\1=2}}{\ltimes} (S_1 \cup \cdots \cup S_n).$$

Therefore, a set P'_E polynomial in the size of E would be the singleton $\{(F, f)\}$.

The question whether, in Theorem 3.2 in general, such a polynomial-size set P always exists, remains open.

5. Conclusion

We have shown that SA expressions that employ repetitions and permutations in projection lists can be simulated using SA expressions that do not employ these features. The same holds for the full relational algebra RA, but there this is trivial to prove, while here the proof is not trivial. There are probably no practical applications of the theorem, and indeed, neither are we aware of practical applications of the corresponding theorem for RA. Nevertheless, as already mentioned in the Introduction, the distinction between the "named" and the "unnamed" perspective in the relational model has received sufficient attention in a renowned textbook [1], and SA is a sufficiently important fragment of RA, so that it seems warranted, if only for the didactical purpose of thorough theoretical understanding of SA, to investigate the named–unnamed distinction for SA as well as for RA, as we have done in the present paper.

References

[1] S. Abiteboul, R. Hull and V. Vianu, *Foundations of Databases*. Addison-Wesley (1995).
[2] H. Andréka, I. Németi and J. van Benthem, Modal languages and bounded fragments of predicate logic. *J. Philosophical Logic* **27** (1998) 217–274.
[3] E.F. Codd, A relational model of data for large shared data banks. *Communications of the ACM* **13** (1970) 377–387.
[4] E.F. Codd, Relational completeness of data base sublanguages, in *Data Base Systems*, R. Rustin, Ed. Prentice-Hall (1972) pp. 65–98.
[5] E. Grädel, On the restraining power of guards. *J. Symbolic Logic* **64** (1999) 1719–1742.
[6] E. Grädel, Guarded fixed point logics and the monadic theory of countable trees. *Theor. Comput. Sci.* **288** (2002) 129–152.
[7] E. Grädel, C. Hirsch and M. Otto, Back and forth between guarded and modal logics. *ACM Transactions on Computational Logic* **3** (2002) 418–463.
[8] E. Grädel and I. Walukiewicz, Guarded fixed point logic, in *Proceedings of the 14th IEEE Symposium on Logic in Computer Science LICS '99* (1999) pp. 45–54.
[9] D. Leinders and J. Van den Bussche, On the complexity of division and set joins in the relational algebra. *J. Comput. Syst. Sci.* **73** (2007) 538–549. Special issue with selected papers on database theory.
[10] D. Leinders, M. Marx, J. Tyszkiewicz and J. Van den Bussche, The semijoin algebra and the guarded fragment. *J. Logic, Language and Information* **14** (2005) 331–343.
[11] D. Leinders, J. Tyszkiewicz and J. Van den Bussche, On the expressive power of semijoin queries. *Inform. Process. Lett.* **91** (2004) 93–98.

Communicated by C. Choffrut.
Received August 21, 2006. Accepted April 3, 2008.