A Hierarchically Typed Relation Algebra

Patrick Roocks

Institut für Informatik, Universität Augsburg

September 17, 2012

Motivation

We have introduced a typed relational algebra to model:

- Tuples of a databases
    - There the types represent the attributes (database columns)
- Relations between database tuples

Motivation

We have introduced a typed relational algebra to model:

- Tuples of a databases
  - There the types represent the attributes (database columns)
- Relations between database tuples

Additionally we introduced:

- A *join algebra*: Differently typed elements can be "glued together"
- This is embedded in a semiring with 1 and $\top$

Motivation

We have introduced a typed relational algebra to model:

- Tuples of a databases
  - There the types represent the attributes (database columns)
- Relations between database tuples

Additionally we introduced:

- A *join algebra*: Differently typed elements can be “glued together”
- This is embedded in a semiring with 1 and $\top$

Application:

- Preferences in databases

$\longrightarrow$ See talk tomorrow

Outline

Problems:

- Arbitrary unions are not typable with our typing mechanism
- 1 and $\top$ are also not typable
- This is a lack of uniformity in our algebra

Our idea:

- Introducing an new typing concept covering arbitrary unions

Outline

Problems:

- Arbitrary unions are not typable with our typing mechanism
- 1 and $\top$ are also not typable
- This is a lack of uniformity in our algebra

Our idea:

- Introducing an new typing concept covering arbitrary unions

The talk is structured as follows:

1. Basics (Typing, Join-Algebra)
2. Sketch of the problem
3. Definition of the new typing mechanism
4. Some properties

Typed tuples

- Database tuples consist of values according to their attributes
- We introduce *types* of relations according to their *attribute names*

Typed tuples

- Database tuples consist of values according to their attributes
- We introduce *types* of relations according to their *attribute names*

We use:

- $\mathcal{A}$: set of attribute names (e.g. set of column names)
- D_A for all $A \in \mathcal{A}$: The *type domain* of the attribute, e.g. $\mathbb{R}, \mathbb{N}$, strings,... (`int, float, varchar, ...`)

Typed tuples

Definition (Typed Tuples, (1/2))

- A *type* T is a subset $T \subseteq \mathcal{A}$.
- An attribute $A \in \mathcal{A}$ also denotes the type $\{A\}$
- A T*-tuple* is a mapping

$$t : T \to \bigcup_{A \in \mathcal{A}} D_A \quad \text{where} \quad \forall A \in T : t(A) \in D_A$$

- The type domain D_T for a type T is the set of all T-tuples, i.e.

$$D_T = \prod_{A \in T} D_A$$

Typed Tuples

Definition (Typed Tuples, **(2/2)**)

- The greatest type domain is the *universe*: $\mathcal{U} =_{df} \bigcup_{T \subseteq \mathcal{A}} D_T$
- Abbreviations:

$$t :: T \;\Leftrightarrow_{df}\; t \in D_T, \qquad M :: T \;\Leftrightarrow_{df}\; M \subseteq D_T$$

Typed Tuples

Definition (Typed Tuples, (2/2))

- The greatest type domain is the *universe*: $\mathcal{U} =_{df} \bigcup_{T \subseteq \mathcal{A}} D_T$
- Abbreviations:

$$t :: T \Leftrightarrow_{df} t \in D_T, \qquad M :: T \Leftrightarrow_{df} M \subseteq D_T$$

Unions of types and differently typed tuples can be expressed by joins:

Definition (Join)

We define the join of types T_1, T_2 and sets of tuples $M_i :: T_i \ (i = 1, 2)$

$$T_1 \bowtie T_2 =_{df} T_1 \cup T_2.$$

$$M_1 \bowtie M_2 =_{df} \left\{ t :: T_1 \bowtie T_2 \mid t|_{T_i} \in M_i, i = 1, 2 \right\}$$

The join operator – an example

Example

Assume a database of cars with attributes: ID, model, power

Consider the sets

$$M_1 = \{\{\mathrm{ID} \mapsto 1,\ \mathrm{model} \mapsto \text{'BMW'}\}, \{\mathrm{ID} \mapsto 3,\ \mathrm{model} \mapsto \text{'Mercedes'}\}\}$$
$$M_2 = \{\{\mathrm{ID} \mapsto 2,\ \mathrm{power} \mapsto 230\}, \{\mathrm{ID} \mapsto 3,\ \mathrm{power} \mapsto 315\}\}.$$

The sets are typed as follows:

$$M_1 :: \mathrm{ID} \bowtie \mathrm{model}, \quad M_2 :: \mathrm{ID} \bowtie \mathrm{power}$$

The join operator – an example

Example

Assume a database of cars with attributes: ID, model, power

Consider the sets

$$M_1 = \{\{\text{ID} \mapsto 1,\ \text{model} \mapsto \text{'BMW'}\}, \{\text{ID} \mapsto 3,\ \text{model} \mapsto \text{'Mercedes'}\}\}$$
$$M_2 = \{\{\text{ID} \mapsto 2,\ \text{power} \mapsto 230\}, \{\text{ID} \mapsto 3,\ \text{power} \mapsto 315\}\}.$$

The sets are typed as follows:

$$M_1 :: \text{ID} \bowtie \text{model}, \quad M_2 :: \text{ID} \bowtie \text{power}$$

The join $M_1 \bowtie M_2 :: \text{ID} \bowtie \text{model} \bowtie \text{power}$ combines tuples with the same ID, because $(\text{ID} \bowtie \text{model}) \cap (\text{ID} \bowtie \text{power}) = \text{ID}$

$$M_1 \bowtie M_2 = \{\{\text{ID} \mapsto 3,\ \text{model} \mapsto \text{'Mercedes'},\ \text{power} \mapsto 315\}\}$$

Type assertions for elements and tests

Algebraically:

$$a :: T^2 \quad \Leftrightarrow_{df} \quad a = 1_T \cdot a \cdot 1_T$$

$$p :: T \quad \Leftrightarrow_{df} \quad p \leq 1_T$$

Type assertions for elements and tests

Algebraically:

$$a :: T^2 \quad \Leftrightarrow_{df} \quad a = 1_T \cdot a \cdot 1_T$$

$$p :: T \quad \Leftrightarrow_{df} \quad p \leq 1_T$$

In the concrete relational instances:

$$a :: T^2 \quad \Leftrightarrow \quad a \subseteq D_T \times D_T$$

$$p :: T \quad \Leftrightarrow \quad p \subseteq D_T$$

Type assertions for elements and tests

Algebraically:

$$a :: T^2 \quad \Leftrightarrow_{df} \quad a = 1_T \cdot a \cdot 1_T$$

$$p :: T \quad \Leftrightarrow_{df} \quad p \leq 1_T$$

In the concrete relational instances:

$$a :: T^2 \quad \Leftrightarrow \quad a \subseteq D_T \times D_T$$

$$p :: T \quad \Leftrightarrow \quad p \subseteq D_T$$

Note that subidentities can be represented as sets:

$$\{(x,x) \mid x \in M\} \mapsto M \quad \text{for} \quad M \subseteq D_T$$

Typed relations and their join

Definition (Typed homogeneous binary relations)

For a type T we define:

$$(t_1, t_2) :: T^2 \;\Leftrightarrow_{df}\; t_i \in D_T, \qquad R :: T^2 \;\Leftrightarrow_{df}\; R \subseteq D_T \times D_T$$

Typed relations and their join

Definition (Typed homogeneous binary relations)

For a type T we define:

$$(t_1, t_2) :: T^2 \iff_{df} t_i \in D_T, \qquad R :: T^2 \iff_{df} R \subseteq D_T \times D_T$$

Special relations:

- The full relation $\top_T =_{df} D_T \times D_T$
- The identity $1_T =_{df} \{(x, x) \mid x :: T\}$
- The empty relation $0_T =_{df} \varnothing$

Typed relations and their join

Definition (Typed homogeneous binary relations)

For a type T we define:

$$(t_1, t_2) :: T^2 \;\Leftrightarrow_{df}\; t_i \in D_T, \qquad R :: T^2 \;\Leftrightarrow_{df}\; R \subseteq D_T \times D_T$$

Special relations:

- The full relation $\top_T =_{df} D_T \times D_T$
- The identity $1_T =_{df} \{(x, x) \mid x :: T\}$
- The empty relation $0_T =_{df} \varnothing$

Definition (Join of relations / Generalised Cartesian Product)

For $R_i :: T_i^2$ $(i = 1, 2)$ we define $R_1 \bowtie R_2 :: (T_1 \bowtie T_2)^2$

$$t\,(R_1 \bowtie R_2)\,u \;\Leftrightarrow_{df}\; t|_{T_1}\, R_1\, u|_{T_1} \;\wedge\; t|_{T_2}\, R_2\, u|_{T_2}\,.$$

The problem

- Assume attributes A, B and tests $p_A :: \{A\}$ and $p_B :: \{B\}$
- Consider the union $p_A + p_B$
- $p_A + p_B :: \{A, B\}$ does **not** hold
- We have $\{A\} \cup \{B\} = \{A\} \bowtie \{B\}$, but $p_A + p_B$ is a union, not a join!

The problem

- Assume attributes A, B and tests $p_A :: \{A\}$ and $p_B :: \{B\}$
- Consider the union $p_A + p_B$
- $p_A + p_B :: \{A, B\}$ does **not** hold
- We have $\{A\} \cup \{B\} = \{A\} \bowtie \{B\}$, but $p_A + p_B$ is a union, not a join!

Consider the subsumption order

$$x \leq y \quad =_{df} \quad x + y = y$$

- Consider the following inequation:

$$p_A \;\leq\; p_A + p_B \;\leq\; 1 \;\leq\; \top$$

- This is valid due to the definition of the subsumption order
- $(p_A + p_B)$, 1 and $\top$ are all not typable with "::"

A first idea

- Consider $\mathcal{U} = \bigcup_{T \subseteq \mathcal{A}} D_T$
- T ranges over $\mathcal{P}(\mathcal{A}) - \varnothing$ ($\mathcal{P}(\ldots)$ denotes the power set)

$\Rightarrow$ "$\mathcal{P}(\mathcal{A}) - \varnothing$" could be the type of $\mathcal{U}$

A first idea

- Consider $\mathcal{U} = \bigcup_{T \subseteq \mathcal{A}} D_T$
- T ranges over $\mathcal{P}(\mathcal{A}) - \varnothing$ ($\mathcal{P}(\ldots)$ denotes the power set)

$\Rightarrow$ "$\mathcal{P}(\mathcal{A}) - \varnothing$" could be the type of $\mathcal{U}$

This would mean:

- Types are not anymore subsets of $\mathcal{A}$...
- ...but subsets of the powerset

A first idea

- Consider $\mathcal{U} = \bigcup_{T \subseteq \mathcal{A}} D_T$
- T ranges over $\mathcal{P}(\mathcal{A}) - \varnothing$ ($\mathcal{P}(\ldots)$ denotes the power set)

$\Rightarrow$ "$\mathcal{P}(\mathcal{A}) - \varnothing$" could be the type of $\mathcal{U}$

This would mean:

- Types are not anymore subsets of $\mathcal{A}$...
- ...but subsets of the powerset

$\longrightarrow$ In the following we will formalize this

Multitypes – Definition

Definition

For a (finite) attribute set $\mathcal{A}$ we define:

1. The set of fundamental types $\mathcal{F}$:
    1. *Base types*: If $A \in \mathcal{A}$, then $\{A\}$ is a base type

Multitypes – Definition

Definition

For a (finite) attribute set $\mathcal{A}$ we define:

1. The set of fundamental types $\mathcal{F}$:
   1. *Base types*: If $A \in \mathcal{A}$, then $\{A\}$ is a base type
   2. *Complex types*: Let T_1 and T_2 be fundamental
      $\Rightarrow$ Then $T_1 \cup T_2$ is a complex type.

   In summary we have:

$$\mathcal{F} = \mathcal{P}(\mathcal{A}) - \varnothing$$

Multitypes – Definition

Definition

For a (finite) attribute set $\mathcal{A}$ we define:

1. The set of fundamental types $\mathcal{F}$:
   1. *Base types*: If $A \in \mathcal{A}$, then $\{A\}$ is a base type
   2. *Complex types*: Let T_1 and T_2 be fundamental
      $\Rightarrow$ Then $T_1 \cup T_2$ is a complex type.

   In summary we have:

$$\mathcal{F} = \mathcal{P}(\mathcal{A}) - \varnothing$$

2. *Multitypes*: $\mathcal{M}$ consists of all subsets $M \subseteq \mathcal{F}$ of fundamental types:

$$\mathcal{M} = \mathcal{P}(\mathcal{F}) = \mathcal{P}(\mathcal{P}(\mathcal{A}) - \varnothing)$$

Type assertions for multitypes

In the relational case, for $M \in \mathcal{M}$:

$$R :: M^2 \quad \Leftrightarrow_{df} \quad R \subseteq \bigcup_{T \subseteq M} (D_T \times D_T)$$

$$R :: M \quad \Leftrightarrow_{df} \quad R \subseteq \bigcup_{T \subseteq M} D_T$$

Type assertions for multitypes

In the relational case, for $M \in \mathcal{M}$:

$$R :: M^2 \quad \Leftrightarrow_{df} \quad R \subseteq \bigcup_{T \subseteq M} (D_T \times D_T)$$

$$R :: M \quad \Leftrightarrow_{df} \quad R \subseteq \bigcup_{T \subseteq M} D_T$$

In the algebraic setting, for $M \in \mathcal{M}$:

$$a :: M^2 \quad \Leftrightarrow_{df} \quad a \leq \sum_{T \in M} 1_T \cdot a \cdot 1_T$$

$$p :: M \quad \Leftrightarrow_{df} \quad p \leq \sum_{T \in M} 1_T$$

Type assertions for multitypes

In the relational case, for $M \in \mathcal{M}$:

$$R :: M^2 \quad \Leftrightarrow_{df} \quad R \subseteq \bigcup_{T \subseteq M} (D_T \times D_T)$$

$$R :: M \quad \Leftrightarrow_{df} \quad R \subseteq \bigcup_{T \subseteq M} D_T$$

In the algebraic setting, for $M \in \mathcal{M}$:

$$a :: M^2 \quad \Leftrightarrow_{df} \quad a \leq \sum_{T \in M} 1_T \cdot a \cdot 1_T$$

$$p :: M \quad \Leftrightarrow_{df} \quad p \leq \sum_{T \in M} 1_T$$

Consequence: The typing of 1 and $\top$:

$$1 :: \mathcal{F}, \quad \top :: \mathcal{F}^2$$

Minimal types

- "$\subseteq$" is the natural order on $\mathcal{M} = \mathcal{P}(\mathcal{F})$
- $\mathcal{F}$ is the maximal type, i.e. $M \subseteq \mathcal{F}$ for all $M \in \mathcal{M}$

Minimal types

- "$\subseteq$" is the natural order on $\mathcal{M} = \mathcal{P}(\mathcal{F})$
- $\mathcal{F}$ is the maximal type, i.e. $M \subseteq \mathcal{F}$ for all $M \in \mathcal{M}$
- Consider an element $a :: T^2$ where $T \in \mathcal{F}$
- We also have for any $M \in \mathcal{M}$ with $T \in M$: $a :: M^2$

Minimal types

- "$\subseteq$" is the natural order on $\mathcal{M} = \mathcal{P}(\mathcal{F})$
- $\mathcal{F}$ is the maximal type, i.e. $M \subseteq \mathcal{F}$ for all $M \in \mathcal{M}$
- Consider an element $a :: T^2$ where $T \in \mathcal{F}$
- We also have for any $M \in \mathcal{M}$ with $T \in M$: $a :: M^2$

$\Rightarrow$ An element has its "real" type and all supertypes

$\Rightarrow$ We want to define a unique "minimal" type

Minimal types

- "$\subseteq$" is the natural order on $\mathcal{M} = \mathcal{P}(\mathcal{F})$
- $\mathcal{F}$ is the maximal type, i.e. $M \subseteq \mathcal{F}$ for all $M \in \mathcal{M}$
- Consider an element $a :: T^2$ where $T \in \mathcal{F}$
- We also have for any $M \in \mathcal{M}$ with $T \in M$: $a :: M^2$

$\Rightarrow$ An element has its "real" type and all supertypes

$\Rightarrow$ We want to define a unique "minimal" type

Definition (Minimal type)

The minimal type for a general element x is defined as follows:

$$x \overset{\text{min}}{::} M^2 \quad \Leftrightarrow_{df} \quad M = \bigcap \{ N \in \mathcal{M} \mid x :: N^2 \}$$

Minimal types

- "$\subseteq$" is the natural order on $\mathcal{M} = \mathcal{P}(\mathcal{F})$
- $\mathcal{F}$ is the maximal type, i.e. $M \subseteq \mathcal{F}$ for all $M \in \mathcal{M}$
- Consider an element $a :: T^2$ where $T \in \mathcal{F}$
- We also have for any $M \in \mathcal{M}$ with $T \in M$: $a :: M^2$

$\Rightarrow$ An element has its "real" type and all supertypes

$\Rightarrow$ We want to define a unique "minimal" type

Definition (Minimal type)

The minimal type for a general element x is defined as follows:

$$x \overset{\min}{::} M^2 \Leftrightarrow_{df} M = \bigcap \{N \in \mathcal{M} \mid x :: N^2\}$$

- $x \overset{\min}{::} \varnothing$ is only fulfilled by $x = 0$, hence $0_T = 0_U$

An example

Example

- Assume attributes A, B with type domains $D_A = \{A_1, A_2\}$ and $D_B = \{B_1, B_2\}$.
- The set $X = \{(A_1, A_2), (B_1, B_2)\}$ fulfils

$$X :: \{\{A\}, \{B\}\}^2$$

An example

Example

- Assume attributes A, B with type domains $D_A = \{A_1, A_2\}$ and $D_B = \{B_1, B_2\}$.
- The set $X = \{(A_1, A_2), (B_1, B_2)\}$ fulfils

$$X :: \{\{A\}, \{B\}\}^2$$

- We do not allow relations between different multitype-subsets:

$$R :: M^2 \quad \Leftrightarrow_{df} \quad R \subseteq \bigcup_{T \subseteq M} (D_T \times D_T) \neq \left(\bigcup_{T \subseteq M} D_T \right) \times \left(\bigcup_{T \subseteq M} D_T \right)$$

$\Rightarrow$ $Y = \{(A_1, B_1)\}$ does not fulfil the assertion $Y :: \{\{A\}, \{B\}\}^2$

An example

Example

- Assume attributes A, B with type domains $D_A = \{A_1, A_2\}$ and $D_B = \{B_1, B_2\}$.
- The set $X = \{(A_1, A_2), (B_1, B_2)\}$ fulfils

$$X :: \{\{A\}, \{B\}\}^2$$

- We do not allow relations between different multitype-subsets:

$$R :: M^2 \quad \Leftrightarrow_{df} \quad R \subseteq \bigcup_{T \subseteq M} (D_T \times D_T) \neq \left(\bigcup_{T \subseteq M} D_T\right) \times \left(\bigcup_{T \subseteq M} D_T\right)$$

$\Rightarrow$ $Y = \{(A_1, B_1)\}$ does not fulfil the assertion $Y :: \{\{A\}, \{B\}\}^2$

- But note that $Y :: \{A\} \bowtie \{B\}$ is true

Generalized Union

- We want to allow unions of elements in $\mathcal{F}$ and in $\mathcal{M}$
- We generalize the union between types:

$$T_1 \cup_{\mathrm{m}} T_2 =_{df} T_1' \cup T_2' \text{ where } T' := \begin{cases} \{T\} & \text{for } T \in \mathcal{F} \\ T & \text{for } T \in \mathcal{M} \end{cases}$$

- Analogously we define $\cap_{\mathrm{m}}$
- We need this to characterize the type of an addition or composition

Type of an addition

Corollary (Type of an addition)

Let $a :: M_a^2, b :: M_b^2$*. Then we have*

$$a + b :: (M_a \cup_m M_b)^2$$

Type of an addition

Corollary (Type of an addition)

Let $a :: M_a^2, b :: M_b^2$*. Then we have*

$$a + b :: (M_a \cup_{\mathrm{m}} M_b)^2$$

Proof.

- We have: $a \leq \sum_{T \in M_a} 1_T \cdot a \cdot 1_T \quad \wedge \quad b \leq \sum_{T \in M_b} 1_T \cdot b \cdot 1_T$
- We conclude:

$$\begin{aligned} a + b &\leq \sum_{T \in M_a} 1_T \cdot a \cdot 1_T + \sum_{T \in M_b} 1_T \cdot b \cdot 1_T \\ &\leq \sum_{T \in M_a} 1_T \cdot (a+b) \cdot 1_T + \sum_{T \in M_b} 1_T \cdot (a+b) \cdot 1_T \\ &= \sum_{T \in M_a \cup_{\mathrm{m}} M_b} 1_T \cdot (a+b) \cdot 1_T \end{aligned}$$

□

More typing properties

- In the relational setting we also have:

$$a \overset{\min}{::} M_a^2, b \overset{\min}{::} M_b^2 \;\Rightarrow\; a + b \overset{\min}{::} (M_a \cup_{\mathrm{m}} M_b)^2$$

More typing properties

- In the relational setting we also have:

$$a \overset{\min}{::} M_a^2, b \overset{\min}{::} M_b^2 \Rightarrow a + b \overset{\min}{::} (M_a \cup_{\mathrm{m}} M_b)^2$$

Corollary (Type of a composition)

Let $a :: M_a^2, b :: M_b^2$. Then we have

$$a \cdot b :: (M_a \cap_{\mathrm{m}} M_b)^2$$

More typing properties

- In the relational setting we also have:

$$a \overset{\min}{::} M_a^2, b \overset{\min}{::} M_b^2 \;\Rightarrow\; a + b \overset{\min}{::} (M_a \cup_m M_b)^2$$

Corollary (Type of a composition)

Let $a :: M_a^2, b :: M_b^2$. Then we have

$$a \cdot b :: (M_a \cap_m M_b)^2$$

For "$\overset{\min}{::}$" this does not hold:

- Assume $a, a' :: A$, $D_A = \{A_1, A_2\}$ and

$$a = (A_1, A_1), \;\; a' = (A_2, A_2)$$

- We have $a \cdot a' = 0$, hence $a \cdot a' \overset{\min}{::} \varnothing$
- But we have $(A \cap_m A) = \{A\}$

Conclusion

What was done in this work:

- Introduced a type hierarchy (basic/fundamental types, multitypes)
- Extended the typing to arbitrary unions of fundamental types
- Introduced a typed 1 and $\top$ in our calculus

Future work:

- Combining sub-typing ($1'_T := r \leq 1_T$) and multitypes
- Introducing projections ($(a \bowtie b)|_{T_a} = a$)...
- ...and embedding them into the multitype-setting
- Extending the multitype-setting to more complex algebras (i.e. heterogeneous relation algebras)