Relational Metadata Integration

Cathy Wyss

April 22, 2005

Talk Overview

1. Motivating Scenario

2. Foundations (TODS, June 2005)

   - Federated Data Model
   - FIRA
   - FISQL

3. Current Work

   - Data mapping as search (SWOD, WIRI, InterDB, SIGMOD demo 2005)
   - FIRA+ for ROLAP (to appear!)

4. Summary

Motivating Scenario

`Carrier1`:

Origin	Dest	Cost
Antwerp	Brussels	25
Antwerp	Bruges	35
Antwerp	Ghent	45
...	...	...

vs.

`Carrier2`:

Dest	Antw.	Brus.	Bruges	Ghent	...
Antw.	$\perp$	27	33	47	...
Brus.	20	$\perp$	22	41	...
Bruges	22	27	$\perp$	53	...
Ghent	20	25	45	$\perp$	...
...	...	...	...	...	...

Desiderata

I Data-metadata transformations

II Dynamic schemas

III Easy incorporation of missing values

IV Relationality

Preliminaries

- Atomic elements: **dom**
  - Examples: `123`, `abc`, `SomeAtom`
- $\perp \notin$ **dom**
- $\varepsilon \in$ **dom**
- Metadata: $\mathfrak{M}_0 \subseteq$ **dom**
- Meta-metadata: $\mathfrak{M}_1$
  - $\mathfrak{M}_1 \cap$ **dom** $= \emptyset$
  - Examples: $\mathfrak{r}_3$, $\mathfrak{a}_{21}$

Relational Data Model

1. A *(Canonical) Tuple*, t is a mapping from a finite set $S \subset \mathfrak{M}_0$ to $\mathbf{dom} \cup \{\perp\}$. Elements of S are termed *attributes*. The square-bracket notation $t[A]$ is used to signify the element $t(A)$ for $A \in S$.

2. A *(Canonical) Relation* has a name $N \in \mathfrak{M}_0$ and a finite schema $S \subset \mathfrak{M}_0$. The relation body consists of a finite set of (canonical) tuples $t : S \rightarrow \mathbf{dom} \cup \{\perp\}$.

3. A *(Canonical) Database* consists of a finite set of (canonical) relations.

Federated Data Model

1. A *(Federated) Tuple* is a mapping from a finite set $S \subset \mathbf{dom} \cup \mathfrak{M}_1$ to $\mathbf{dom} \cup \{\perp\}$. S is known as the *Schema* of the tuple, i.e. $S = schema(t)$.

2. A *(Federated) Relation* has a name $N \in \mathbf{dom}$. The relation body consists of a finite set of (federated) tuples.

3. A *(Federated) Database* has a name $D \in \mathfrak{M}_0$. The database body consists of a finite set of (federated) relations.

4. A *Federation* consists of a finite set of (federated) databases.

Federated Data Model

- Given a federated relation $\mathcal{R}$, we define its *Schema* to be

$$schema(\mathcal{R}) = \bigcup_{t \in body(\mathcal{R})} schema(t).$$

- $\Rightarrow$ **Dynamic schemas**

  - $t[A] = \perp$ for $A \in \mathfrak{M}_0 - schema(\mathcal{R})$

Federated Interoperable RA (FIRA)

- RA:
  - The *Relational Algebra* (RA) is: ρ (Renaming), σ (Selection), π (Projection), $\times$ (Cartesian Product), $\cup$ (Set Union), and $-$ (Set Difference).
  - A query of RA maps a set of input relations (*i.e.* a database) to a single output relation.

- FIRA:
  - A query of FIRA maps a set of input databases (*i.e.* a federation) to a single output database.

Federated Interoperable RA:
Main Result

Theorem:
RA is isomorphic to a sub-algebra of FIRA.

Embedding:
$R \mapsto \{\langle \varepsilon, R \rangle\}$

Relational Core of FIRA:
Contains federated counterparts for unary (ρ, σ, π) and binary ($\times$, $\cup$, $-$) RA operators.

Basic Terms of FIRA

- *Basic terms* in FIRA are database names or database variables.

- Use variables of the form $\mathcal{D}_1, \mathcal{D}_2, ...$ to denote databases.

Federated Unary Relational Operators: ρ

(Renaming) Let $\mathcal{D}$ be a federated database. There are two cases.

1. *(General Renaming)* Let $A_i, B_i \in \mathbf{dom} \cup \mathfrak{M}_1$ for $1 \leq i \leq n$. Then

$$\widehat{\rho}_{A_1 \to B_1, \ldots, A_n \to B_n}(\mathcal{D}) = \{\langle name(\mathcal{R}), \rho_{A_1 \to B_1, \ldots, A_n \to B_n}(body(\mathcal{R}))\rangle | \mathcal{R} \in \mathcal{D}\}.$$

2. *(Relation Specific Renaming)* Let $A_i, B_i \in \mathbf{dom} \cup \mathfrak{M}_1$ for $1 \leq i \leq n$ and $N, M \in \mathbf{dom}$. Then

$$\widehat{\rho}^{N \to M}_{A_1 \to B_1, \ldots, A_n \to B_n}(\mathcal{D}) = \{\langle M, \rho_{A_1 \to B_1, \ldots, A_n \to B_n}(body(\mathcal{R}))\rangle | \mathcal{R} \in \mathcal{D}, name(\mathcal{R}) = N\} \cup \{\mathcal{R} \in \mathcal{D} | name(\mathcal{R}) \neq N\}.$$

Federated Unary Relational Operators: σ

(Selection) Let $\mathcal{D}$ be a federated database and C be a well-formed Boolean selection condition. Then

$$\widehat{\sigma}_C(\mathcal{D}) = \{\langle name(\mathcal{R}), \sigma_C(body(\mathcal{R}))\rangle | \mathcal{R} \in \mathcal{D}\}.$$

Federated Unary Relational Operators: π

(Projection) Let $\mathcal{D}$ be a federated database and $A_k \in \mathbf{dom} \cup \mathfrak{M}_1$ for $1 \leq k \leq n$. Then

$$\widehat{\pi}_{A_1,\ldots,A_n}(\mathcal{D}) = \{\langle name(\mathcal{R}), \pi_{A_1,\ldots,A_n}(body(\mathcal{R}))|\mathcal{R} \in \mathcal{D}\}.$$

Federated Binary Relational Operators: $\times$

Let $\mathcal{D}_1$ and $\mathcal{D}_2$ denote federated databases. Their *Federated Cartesian Product* is defined as

$$\mathcal{D}_1 \hat{\times} \mathcal{D}_2 = \{\langle name(\mathcal{R}_1), body(\mathcal{R}_1) \times body(\mathcal{R}_2)\rangle \mid \mathcal{R}_1 \in \mathcal{D}_1, \mathcal{R}_2 \in \mathcal{D}_2 \text{ and } name(\mathcal{R}_1) = name(\mathcal{R}_2)\}.$$

Federated Binary Relational Operators: $\cup$

Let $\mathcal{D}_1$ and $\mathcal{D}_2$ denote federated databases. Their *Federated Set Union* is defined as

$$\mathcal{D}_1 \hat{\cup} \mathcal{D}_2 =$$
$$\{\langle name(\mathcal{R}_1), body(\mathcal{R}_1) \cup body(\mathcal{R}_2)\rangle$$
$$\mid \mathcal{R}_1 \in \mathcal{D}_1, \mathcal{R}_2 \in \mathcal{D}_2$$
$$\text{and } name(\mathcal{R}_1) = name\mathcal{R}_2\}$$

$$\bigcup \{\mathcal{R}_1 \in \mathcal{D}_1 | \text{ there is no } \mathcal{R}_2 \in \mathcal{D}_2$$
$$\text{such that } name(\mathcal{R}_1) = name\mathcal{R}_2\}$$

$$\bigcup \{\mathcal{R}_2 \in \mathcal{D}_2 | \text{ there is no } \mathcal{R}_1 \in \mathcal{D}_1$$
$$\text{such that } name(\mathcal{R}_2) = name\mathcal{R}_1\}.$$

Federated Binary Relational Operators: $-$

Let $\mathcal{D}_1$ and $\mathcal{D}_2$ denote federated databases. The *Federated Set Difference* of $\mathcal{D}_1$ and $\mathcal{D}_2$ is defined as

$$\mathcal{D}_1 \widehat{-} \mathcal{D}_2 =$$
$$\{\langle name(\mathcal{R}_1), body(\mathcal{R}_1) - body(\mathcal{R}_2)\rangle$$
$$\mid \mathcal{R}_1 \in \mathcal{D}_1, \mathcal{R}_2 \in \mathcal{D}_2$$
$$\text{and } name(\mathcal{R}_1) = name\mathcal{R}_2\}$$

$$\bigcup \{\mathcal{R}_1 \in \mathcal{D}_1 | \text{ there is no } \mathcal{R}_2 \in \mathcal{D}_2$$
$$\text{such that } name(\mathcal{R}_1) = name\mathcal{R}_2\}.$$

Federated Interoperable RA:
Main Result

Theorem:
RA is isomorphic to a sub-algebra of FIRA.

Embedding:
$R \mapsto \{\langle \varepsilon, R \rangle\}$

Relational Core of FIRA:
Contains federated counterparts for unary (ρ, σ, π) and binary ($\times$, $\cup$, $-$) RA operators.

FIRA beyond RA: $\mathcal{U}$

1. (Drop Projection for Federated Relations) Let $\mathcal{R}$ be a federated relation and $A \in \mathbf{dom} \cup \mathfrak{M}_1$. Then

   $$\mathcal{U}_A(\mathcal{R}) = \{\langle name(\mathcal{R}), \pi_{schema(\mathcal{R})-A}(body(\mathcal{R})\rangle\}.$$

2. (Drop Projection for Federated Databases) Let $\mathcal{D}$ be a federated database and $A \in \mathbf{dom} \cup \mathfrak{M}_1$. Then $\mathcal{U}_A(\mathcal{D}) = \{\mathcal{U}_A(\mathcal{R}) | \mathcal{R} \in \mathcal{D}\}$.

   In addition, we use the shorthand notation $\widehat{\mathcal{U}}_{A_1,\ldots,A_n}(\mathcal{D})$ for $A_k \in \mathbf{dom} \cup \mathfrak{M}_1$ $(1 \leq k \leq n)$ to mean $\widehat{\mathcal{U}}_{A_1}(\cdots(\widehat{\mathcal{U}}_{A_n}(\mathcal{D}))\cdots)$.

FIRA beyond RA: $\downarrow$

Let $\mathcal{R}$ be a federated relation and $i \in \mathbb{N}$ be a fixed natural number.

Let $name(\mathcal{R}) = N \in$ **dom** and $schema(\mathcal{R}) \cap$ **dom** $= \{A_1, ..., A_n\}$. We define $metadata_i(\mathcal{R})$ to be the following set of federated tuples:

$\mathfrak{r}_i$	$\mathfrak{a}_i$
N	A_1
N	A_2
$\vdots$	$\vdots$
N	A_n

Example:

$\mathfrak{r}_i$	$\mathfrak{a}_i$
Carrier2	Dest
Carrier2	Antw.
Carrier2	Brus.
Carrier2	Bruges
Carrier2	Ghent
$\vdots$	$\vdots$

FIRA beyond RA: $\downarrow$

1. (Down Operators for Federated Relations) The *down of $\mathcal{R}$ with respect to i*, denoted $\downarrow_i (\mathcal{R})$ is the federated relation

$$\downarrow_i (\mathcal{R}) = \langle name(\mathcal{R}), \\ metadata_i(\mathcal{R}) \times \Cup_{\mathfrak{r}_i, \mathfrak{a}_i}(body(\mathcal{R}))\rangle.$$

2. (Down Operators for Federated Databases) Let $\mathcal{D}$ be a federated database. Then

$$\downarrow_i (\mathcal{D}) = \{\downarrow_i (\mathcal{R}) | \mathcal{R} \in \mathcal{D}\}.$$

FIRA beyond RA: Δ

1. (Attribute Dereference for Federated Relations) Let $\mathcal{R}$ be a federated relation and $A, B \in \mathbf{dom} \cup \mathfrak{M}_1$. Then

$$\Delta_A^B(\mathcal{R}) = \langle name(R), R' \rangle$$

where R' is obtained from $body(\mathcal{R})$ tuple-by-tuple as follows. For $t \in body(\mathcal{R})$, we obtain $s \in R'$ as:

$$s[X] = \begin{cases} t[t[A]] \text{ iff } X = B; \\ t[X] \text{ otherwise.} \end{cases}$$

2. (Attribute Dereference for Federated Databases) Let $\mathcal{D}$ be a database and $A, B \in \mathbf{dom} \cup \mathfrak{M}_1$. Then

$$\Delta_A^B(\mathcal{D}) = \{\Delta_A^B(\mathcal{R}) | \mathcal{R} \in \mathcal{D}\}.$$

FIRA beyond RA: Δ

Example:

R:

A	B	C
A	1	2
B	3	4
B	5	6
E	7	8

$\Delta_A^D(R)$:

A	B	C	D
A	1	2	A
B	3	4	3
B	5	6	5
E	7	8	$\perp$

FIRA beyond RA: Σ

Generalized (Outer) Union:

Let $\mathcal{D}$ be a federated database. Then

$$\Sigma(\mathcal{D}) = \{\langle \varepsilon, \bigcup_{\mathcal{R} \in \mathcal{D}} body(\mathcal{R}) \rangle\}.$$

FIRA beyond RA: $\wp$

1. (Partition for Federated Relations) Let $\mathcal{R}$ be a federated relation and $A \in \mathbf{dom} \cup \mathfrak{M}_1$. Then $\wp_A(\mathcal{R})$ is the federated database

$$\wp_A(\mathcal{R}) = \{\langle a, \sigma_{A=\text{"}a\text{"}}(body(\mathcal{R}))\rangle \mid \exists\, t \in body(\mathcal{R}) \text{ s.t. } t[A] = a\}.$$

2. (Partition for Federated Databases) Let $\mathcal{D}$ be a database and $A \in \mathbf{dom} \cup \mathfrak{M}_1$. Then

$$\wp_A(\mathcal{D}) = \widehat{\bigcup}_{\mathcal{R}\in\mathcal{D}} \wp_A(\mathcal{R}).$$

FIRA beyond RA: τ

1. (Transpose for Federated Relations) Let $\mathcal{R}$ be a federated relation and $A, B \in \mathbf{dom} \cup \mathfrak{M}_1$. Then the *transpose of A on B of $\mathcal{R}$*, denoted $\tau_A^B(\mathcal{R})$, is a relation having the same name as $\mathcal{R}$, where each tuple, s, in the body of the output relation is obtained from tuple $t \in body(\mathcal{R})$ as follows.

$$s[X] = \begin{cases} t[A] \text{ iff } X = t[B]; \\ t[X] \text{ iff } X \in schema(t), X \neq t[B]; \\ \perp \text{ otherwise.} \end{cases}$$

2. (Transpose for Federated Databases) Let $\mathcal{D}$ be a database and $A, B \in \mathbf{dom} \cup \mathfrak{M}_1$. Then $\tau_A^B(\mathcal{D}) = \{\tau_A^B(\mathcal{R}) | \mathcal{R} \in \mathcal{D}\}$.

FIRA beyond RA: τ

Example:

R:

A	B	C
A	1	2
D	3	4
E	5	6
F	7	8

$\tau_B^A(R)$:

A	B	C	1	D	E	F
1	1	2	1	$\perp$	$\perp$	$\perp$
D	3	4	$\perp$	3	$\perp$	$\perp$
E	5	6	$\perp$	$\perp$	5	$\perp$
F	7	8	$\perp$	$\perp$	$\perp$	7

Illustrative Query

`Carrier1`:

Origin	Dest	Cost
Antwerp	Brussels	25
Antwerp	Bruges	35
Antwerp	Ghent	45
...	...	...

vs.

`Carrier2`:

Dest	Antw.	Brus.	Bruges	Ghent	...
Antw.	$\perp$	27	33	47	...
Brus.	20	$\perp$	22	41	...
Bruges	22	27	$\perp$	53	...
Ghent	20	25	45	$\perp$	...
...	...	...	...	...	...

"Find all routes where `Carrier2` is less expensive than `Carrier1`."

Federated Interoperable RA

"Find all routes where Carrier2 is less expensive than Carrier1."

$$\sigma_{\substack{C1.\texttt{Dest}=C2.\texttt{Dest}\wedge \\ \texttt{newCol}<C1.\texttt{Cost}}}\Big($$

$$\Delta^{\texttt{newCol}}_{C1.\texttt{Origin}}(\rho^{C1}(\texttt{Carrier1})$$

$$\times\ \rho^{C2}(\texttt{Carrier2})\Big)$$

Federated Interoperable SQL

```
SELECT C1.Origin AS 'Origin',
       C1.Dest AS 'Dest'
  INTO 'Result'

FROM   Carrier1:A1 AS C1,
       Carrier2:A2 AS C2

WHERE  A2 = C1.Origin AND
       C2.Dest = C1.Dest AND
       C2.A2 < C1.Cost
```

Federated Interoperable SQL

⟨*query*⟩ ::= `SELECT` ⟨*col_decls*⟩ `INTO` ⟨*name_term*⟩
`FROM` ⟨*variable_decls*⟩
[`WHERE` {⟨*condition*⟩}*]
| (⟨*query*⟩) `UNION` (⟨*query*⟩)
| (⟨*query*⟩) `MINUS` (⟨*query*⟩)

⟨*col_decls*⟩ ::= ⟨*col_decl*⟩ {, ⟨*col_decl*⟩}*

⟨*col_decl*⟩ ::= ⟨*name_term*⟩ `AS` ⟨*string*⟩
| ⟨*name_term*⟩ `ON` ⟨*name_term*⟩
| * [`DROP` ⟨*name_term*⟩ {, ⟨*name_term*⟩}*]

⟨*variable_decls*⟩ ::= ⟨*var_decl*⟩ {, ⟨*var_decl*⟩}*

⟨*var_decl*⟩ ::= ⟨*db_name*⟩ ⟨*base_var_decl*⟩
| (⟨*query*⟩) ⟨*base_var_decl*⟩

⟨*base_var_decl*⟩ ::= :⟨*varname(rel)*⟩:⟨*varname(att)*⟩
`AS` ⟨*varname(tup)*⟩

⟨*condition*⟩ ::= (⟨*condition*⟩)
| (⟨*condition*⟩) `AND` (⟨*condition*⟩)
| (⟨*condition*⟩) `OR` (⟨*condition*⟩)
| `NOT` (⟨*condition*⟩)
| ⟨*name_term*⟩ ⟨*cond_operator*⟩ ⟨*name_term*⟩

⟨*cond_operator*⟩ ::= = | != | <= | < | > | >=

⟨*name_term*⟩ ::= ⟨*string*⟩
| ⟨*varname(meta)*⟩
| ⟨*varname(tup)*⟩.⟨*varname(meta)*⟩
| ⟨*varname(tup)*⟩.⟨*dom_elt*⟩
| ⟨*varname(tup)*⟩.⟨*varname(tup)*⟩.⟨*dom_elt*⟩

⟨*varname(meta)*⟩ ::= ⟨*varname(rel)*⟩|⟨*varname(att)*⟩

⟨*varname(X)*⟩ ::= ⟨*dom_elt*⟩ — *X* is *rel*, *att*, or *tup*

⟨*db_name*⟩ ::= ⟨*dom_elt*⟩

⟨*string*⟩ ::= "⟨*dom_elt*⟩"

⟨*dom_elt*⟩ ::= (a–z|A–Z|0–9){(a–z|A–Z|0–9|-)}*

Federated Interoperable SQL: Main Result

Theorem:

1. For every FISQL query Q there is an equivalent FIRA query $\hat{Q}$ such that for well-formed federation instances $\mathcal{F}$, $Q(\mathcal{F}) = \hat{Q}(\mathcal{F})$.

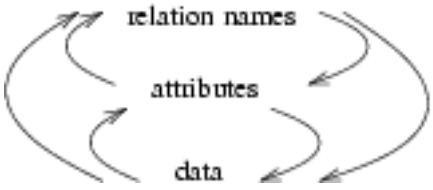
2. For every FIRA query $\hat{Q}$ there is an equivalent FISQL query Q such that for well-formed federation instances $\mathcal{F}$, $\hat{Q}(\mathcal{F}) = Q(\mathcal{F})$.

Transformational Completeness

- A query language that can express all queries of RA is said to be *Relationally Complete* [Codd 1970].

- Analogue: A query language that can express all queries of FIRA is said to be *Transformationally Complete*.

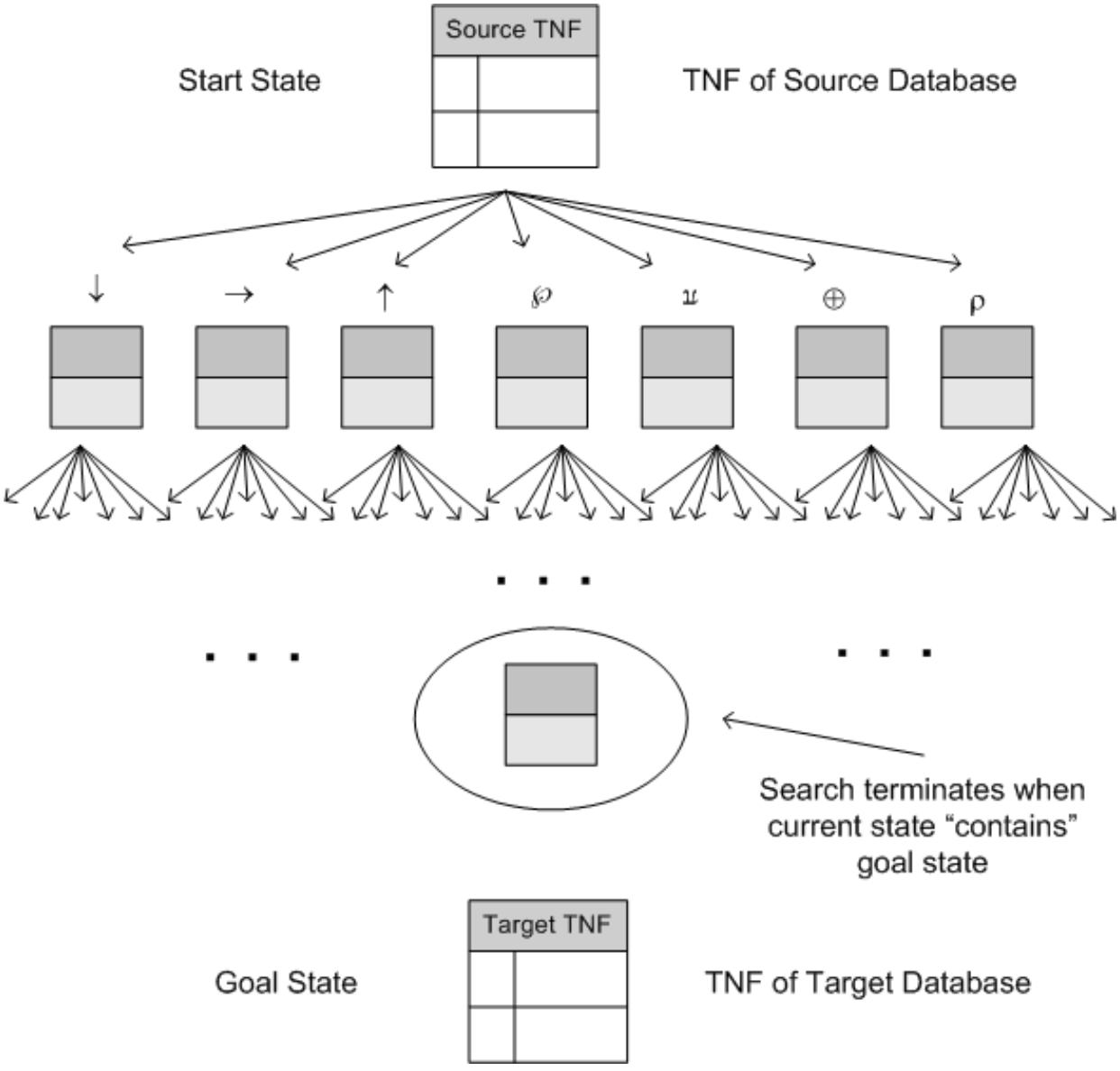
- Intuitively, a language is TC if
  - it is relationally complete,
  - it can express all data/metadata transformations.



Current Work

- George Fletcher
  - Data Mapping as Search
  - "Reverse Fagin" Framework
- Fulya Erdinc
  - FIRA extensions for OLAP

Data Mapping/Exchange as Search



Data Exchange (Fagin *et. al.*)

- A *Data Exchange* setting:
  - A source schema, $\mathcal{S}$,
  - a target schema, $\mathcal{T}$,
  - a set of source-to-target dependencies $\Sigma_{\mathcal{ST}}$, and
  - an instance I of $\mathcal{S}$.

  Given this, find a corresponding instance J of $\mathcal{T}$.

- $\Sigma_{\mathcal{ST}}$ are expressed in FOL

Turn it around?

- Given $\mathcal{S}$, $\mathcal{T}$, I, and J, find $\Sigma_{\mathcal{ST}}$
- Metadata logic more appropriate than FOL
- Our search program is "theorem proving" in an appropriate calculus
- Federated Interoperable RC
- Investigate declarative properties of this "Reverse Fagin" framework

Extending FIRA for ROLAP

- Recall the *conceptual–logical–physical* breakdown of the DBMS

- ROLAP applications are highly *conceptual* analysis programs for warehoused data.

  – Main operations: CUBE, PIVOT

- ROLAP optimizations/implementations in the literature are *physical* in nature

- Our aim: extend FIRA to provide a *logical* basis for ROLAP optimization

Extending FIRA for ROLAP: Some Early Results

- Breakdown of PIVOT into τ and *merge* operations
  - Formal characterization of error conditions
  - Generality beyond SQL PIVOT
- An RA expression for computing CUBE that linear expression size with respect to the number of input attributes.
  - Canonical RA interpretation has exponential expression size

Summary

- FIRA/FISQL provides a relational framework for dynamic metadata integration
- FIRC underpins data exchange as search
- FIRA extensions for ROLAP