Characterization and Uniqueness of Closed Update Strategies for Database Views

Stephen J. Hegner
Department of Computing Science
Umeå University
Sweden

Statement of the Problem

- Two fundamental services to be provided by a database system:

  Support for updates: The data which are stored in the system may be altered.

  - The term *update* is used here in a general sense, to denote *insertions* and *deletions*, as well as alterations of existing values.

  Support via views: Access to parts of the database, via windows which are known as *views*, is allowed.

- Three reasons for supporting views:

  Security: It may not be appropriate for every user of the system to have full access to every part of the database.

  Simplicity: It is easier for a user to perform a given task if only the necessary information is presented.

  Summary: It may be appropriate to provide summary information which is not explicit in the main database.

Problem: Updates and views mix about as well as oil and water.

A View Example in the Relational Context

Example base schema and instance:

$\mathcal{F}_P = \{\mathsf{Name} \longrightarrow \mathsf{Dept}\}$

Rel P:

Name	Dept	Proj
Smith	1	A
Jones	2	A
Jones	2	B
Wilson	3	C

$\mathcal{F}_Q = \{\mathsf{Proj} \longrightarrow \mathsf{Budget}\}$

Rel Q:

Proj	Budget
A	100
C	300
D	300

Example view $\Gamma = \pi_{\mathsf{Name,Proj,Budget}}(P \bowtie Q)$

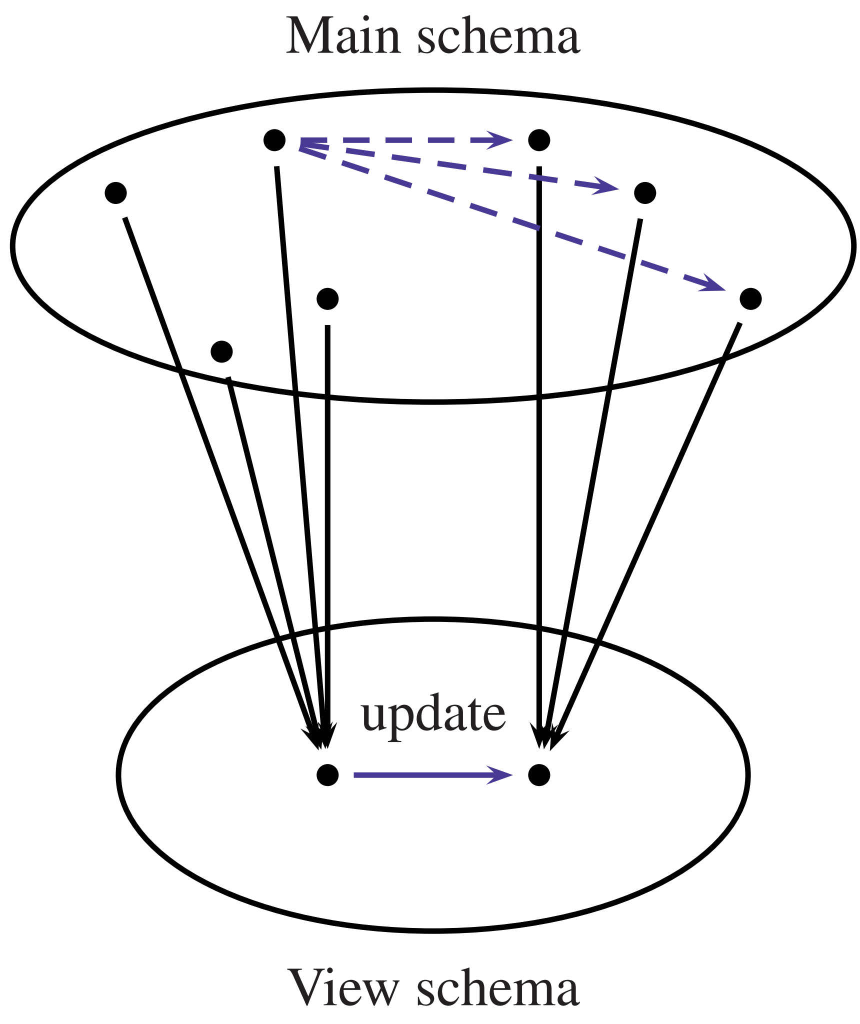
$\mathcal{F}_\Gamma = \{\mathsf{Proj} \longrightarrow \mathsf{Budget}\}$

Name	Proj	Budget
Smith	A	100
Jones	A	100
Wilson	C	300

- Note that the underlying mapping for this view is not injective (one-to-one).

Why the Update Problem for Views is Difficult

- On the underlying states, the view mapping is generally *surjective* (onto) but not *injective* (one-to-one).
- Thus, a single view state corresponds, in general, to many states of the main schema.
- The problem of identifying the correct main schema update corresponding to a view update is known as the *update translation problem*.



Extreme Cases of the View Update Problem

Example base schema and instance:

$\mathcal{F}_P = \{\mathsf{Name} \longrightarrow \mathsf{Dept}\}$

Rel P:

Name	Dept	Proj
Smith	1	A
Jones	2	A
Jones	2	B

$\mathcal{F}_Q = \{\mathsf{Proj} \longrightarrow \mathsf{Budget}\}$

Rel Q:

Proj	Budget
A	100
C	300
D	300

Example views:

View Γ_a: All of Q

$\mathcal{F}_{\Gamma_a} = \{\mathsf{Proj} \longrightarrow \mathsf{Budget}\}$

Proj	Budget
A	100
C	300
D	300

- Any update which respects the FD is allowed.
- Natural translation of view updates keeps relation P constant in all reflections of view updates.

View Γ_b: $\pi_{\mathsf{Budget}}(\mathsf{Q})$

Budget
100
300

- No view updates possible under any reasonable translation strategy.

Criteria for Determining Admissibility of Update Translations

In less extreme cases, there are two types of criteria which may be applied to assess translatability of view updates.

- *Uniqueness criteria:*
  - A view update is supported if it has only one “reasonable” reflection to an update of the base schema.
  - No *ad hoc* changes to the base schema are permitted in the translation.
  - Most work on the support of view updates has focused upon this type of criterion.

- *Interface criteria:*
  - These criteria focus upon how the view appears to its users. Important examples include the following:
    - The translation of a view update to an update of the base schema must be completely “understandable” within the context of the view itself.
    - Changes to the base schema which are not visible within the view schema are discouraged.
  - How the allowable updates interact as a group is also a point of focus.
  - Relatively little work on the support of view updates has focused upon this type of criterion.

Example 1 — Uniqueness vs. Interface Criteria

Base schema and instance:

Rel P:

Name	Dept	Proj
Smith	1	A
Jones	2	A
Jones	2	B

Constraints:

Name $\longrightarrow$ Dept

No nulls allowed.

View and instance:

$\mathsf{R} = \Pi_{(\mathsf{Name},\mathsf{Proj})}(\mathsf{P})$:

Name	Proj
Smith	A
Jones	A
Jones	B

Constraints:

No FD's

No nulls allowed.

Proposed view update: Delete (Smith, A) from R

- This update would be allowed under most uniqueness criteria.
  - The unique "reasonable" base update is:

    Delete $(\text{Smith}, 1, A)$ from P
- This view update might be disallowed under certain interface criteria.
  - The update involves a *hidden trigger*. The fact that Dept $= 1$ for Smith is removed from the base schema, but this deletion is not visible within the view.
  - The update is *irreversible* without knowledge of the state history of the base schema. Re-insertion of (Smith, A) into the view cannot magically re-create the fact that Smith was in Department 1.

Example 2 — Uniqueness vs. Interface Criteria

Base schema and instance:

Rel P:

Name	Dept	Proj
Smith	1	A
Jones	2	A
Jones	2	B
Wilson	1	Null

Constraints:
Name $\longrightarrow$ Dept
Nulls allowed for Proj.

View and instance:

$\mathsf{R} = \Pi_{(\mathsf{Name},\widetilde{\mathsf{Proj}})}(\mathsf{P})$:
$\widetilde{\mathsf{Proj}} = \mathsf{Proj}$
with nulls
disallowed

Name	Proj
Smith	A
Jones	A
Jones	B

Constraints:
No FD's
No nulls allowed.

Proposed view update: Delete (Smith, A) from R

- This view update is realizable by the base update:
  Modify $(\text{Smith}, 1, A) \mapsto (\text{Smith}, 1, \mathsf{Null})$
- The hidden trigger and irreversibility problems are not present in this modified view.
- Unfortunately, this view poses other update problems with respect to interface criteria: a *hidden dynamic constraint*.

Example 2a — Uniqueness vs. Interface Criteria

Base schema and instance:

Rel P:

Name	Dept	Proj
Smith	1	Null
Jones	2	A
Jones	2	B
Wilson	1	Null

Constraints:

Name $\longrightarrow$ Dept

Nulls allowed for Proj.

View and instance:

$\mathsf{R} = \Pi_{(\mathsf{Name},\widetilde{\mathsf{Proj}})}(\mathsf{P})$:

$\widetilde{\mathsf{Proj}} = \mathsf{Proj}$ with nulls disallowed

Name	Proj
Jones	A
Jones	B

Constraints:

No FD's

No nulls allowed.

Proposed view updates: Insert (Smith, A) into R
Insert (Young, A) into R

- The first is realizable by the following base update:

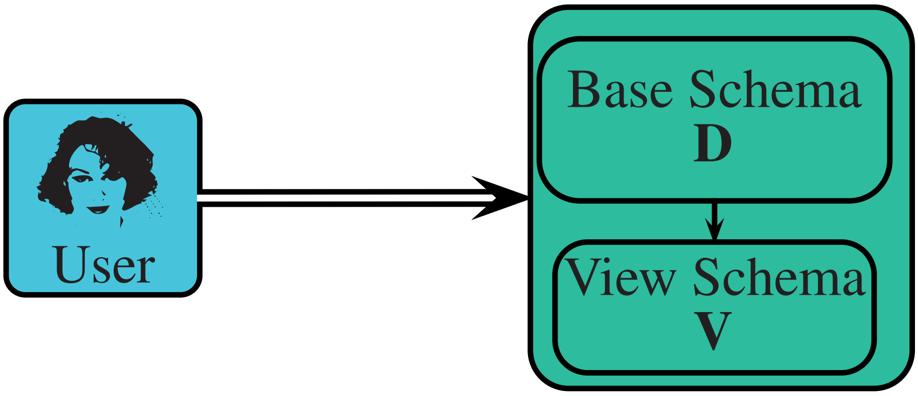
  Modify (Smith, 1, Null) $\mapsto$ (Smith, 1, A)

- The second is not realizable, even under uniqueness conditions, because no department information is available for Young.

- Note that it is not possible to determine, from the view state alone, whether or not a proposed update is admissible. Further information from the base schema state must be known. This view contains a *hidden dynamic constraint*.

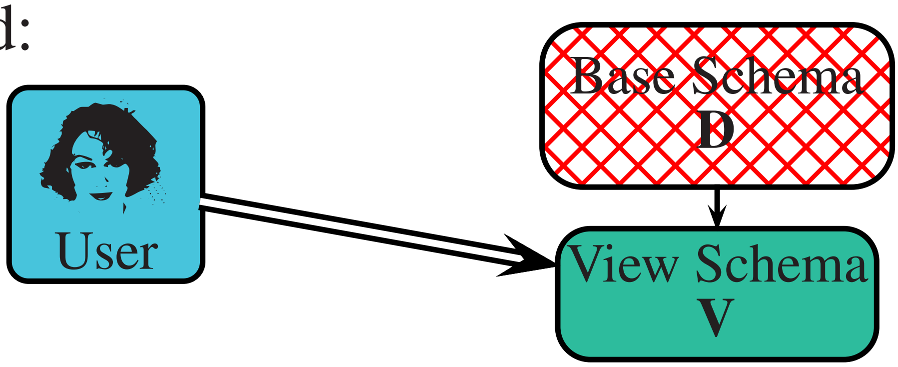
Open vs. Closed Views

Open:



- The user has access to both the view and the base schema.
- The view is provided as a convenience.
- Uniqueness criteria suffice for update translation.

Closed:



- The user has access only to the view.
- The user has no direct knowledge of the base schema.
- The view must be self contained in terms of knowledge needed to effect updates.
- The view should preferably look “just” like a complete base schema.
- Interface criteria for update translations are extremely important.

Major Goal of this Work

- The overall goal is to develop a systematic theory of update support for *closed* database views.
  - This implies in particular that careful attention be paid to interface criteria.
- The strategy is to build upon the seminal *constant-complement* approach of Bancilhon and Spyratos.
- Major enhancements developed here:
  - Uniqueness of translations
  - Meet-based characterization of admissible updates
  - Compatibility of distinct update strategies
- The context:

  Generality: A general theory is sought which is applicable to wide range of data models.

  Applicability: The theory must have something nontrivial and useful to say about the most common of all data models, the relational model.

Initial Abstraction of the Concept of a View

- First introduced by Bancilhon and Spyratos (1979-1981) in the study of independence and updates.

Schema: A *database schema* $\mathbf{D}$ is represented entirely by its set $\mathsf{LDB}(\mathbf{D})$ of *legal states*. All other information is "forgotten."

Morphism: A *database mapping* or *database morphism*

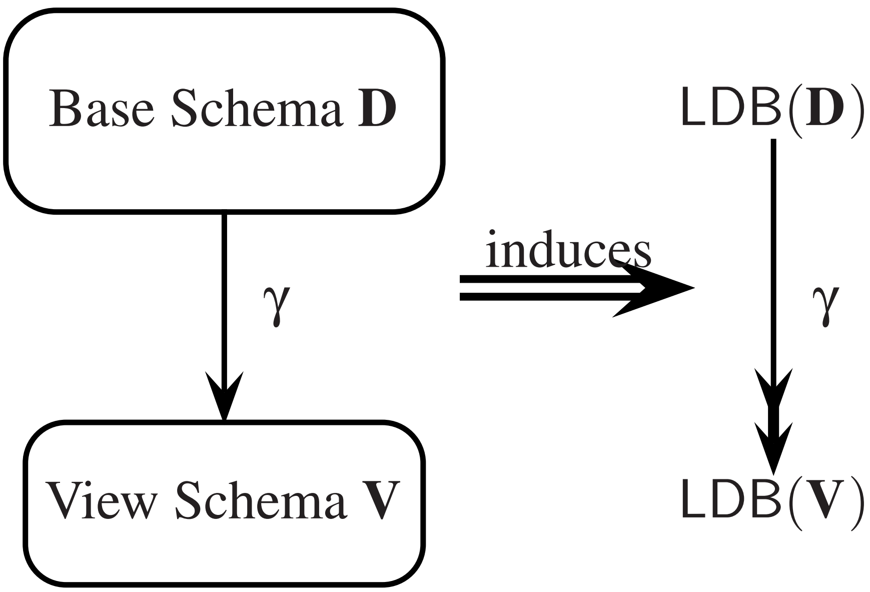
$$f : \mathbf{D}_1 \to \mathbf{D}_2$$

is just a function

$$f : \mathsf{LDB}(\mathbf{D}_1) \to \mathsf{LDB}(\mathbf{D}_2)$$

View: A *view* of the schema $\mathbf{D}$ is a pair $\Gamma = (\mathbf{V}, \gamma)$ in which:

- $\mathbf{V}$ is a schema;
- $\gamma : \mathbf{D} \to \mathbf{V}$ is a *surjective* morphism.



Closed Update Families

- A *closed update family* for $\mathbf{V}$ is an equivalence relation U on $\mathsf{LDB}(\mathbf{V})$.
- $(M_1, M_2) \in U$ means that the update $M_1 \longrightarrow M_2$ is admissible on $\mathbf{V}$.
- Interpretation of equivalence relation properties:
  - ➪ *Reflexivity* implies that the identity update is always allowed.
  - ➪ *Symmetry* implies that every update is reversible.
  - ➪ *Transitivity* implies that updates may be composed.

Fundamental modelling assumption: The set of admissible updates on a view forms a closed update family.

Translators for View Update

- An *update strategy* is a partial function:

  $\rho : \mathsf{LDB}(\mathbf{D}) \times \mathsf{LDB}(\mathbf{V}) \to \mathsf{LDB}(\mathbf{D})$

  $\rho :$ Base States $\times$ View States $\to$ Base States

  (Current Base State, New View State) $\mapsto$ New Base State.

$$\begin{tikzcd}
M_1 \arrow[r, mapsto, dashed, "\text{Translated update}", "\text{on base schema}"'] \arrow[d, mapsto, "\gamma"'] & \rho(M_1, N_2) \arrow[d, mapsto, dashed, "\gamma"'] \\
\gamma(M_1) = N_1 \arrow[r, mapsto, "\text{View update}"] & N_2 = \gamma(\rho(M_1, N_2))
\end{tikzcd}$$

- Not all view updates are allowed; thus ρ is a partial function, in general.

Closed Update Strategies

- Let T = a closed update family for the view schema $\mathbf{V}$.
  U = a closed update family for the base schema $\mathbf{D}$.
- In a closed setting, the following conditions are imposed to yield a *closed update strategy* for T with respect to U.

(upt:1) [The allowable view updates are exactly the pairs in T.]
$\rho(M,N)\downarrow$ iff $(\gamma(M),N) \in T$.

(upt:2) [Only base schema updates from U are embodied in ρ.]
If $\rho(M,N)\downarrow$, then $(M,\rho(M,N)) \in U$ and $\gamma(\rho(M,N)) = N$.

(upt:3) [Identity updates are reflected as identities.]
For every $M \in \mathsf{LDB}(\mathbf{D})$, $\rho(M,\gamma(M)) = M$.

(upt:4) [Every view update is globally reversible.]
If $\rho(M,N)\downarrow$, then $\rho(\rho(M,N),\gamma(M)) = M$.

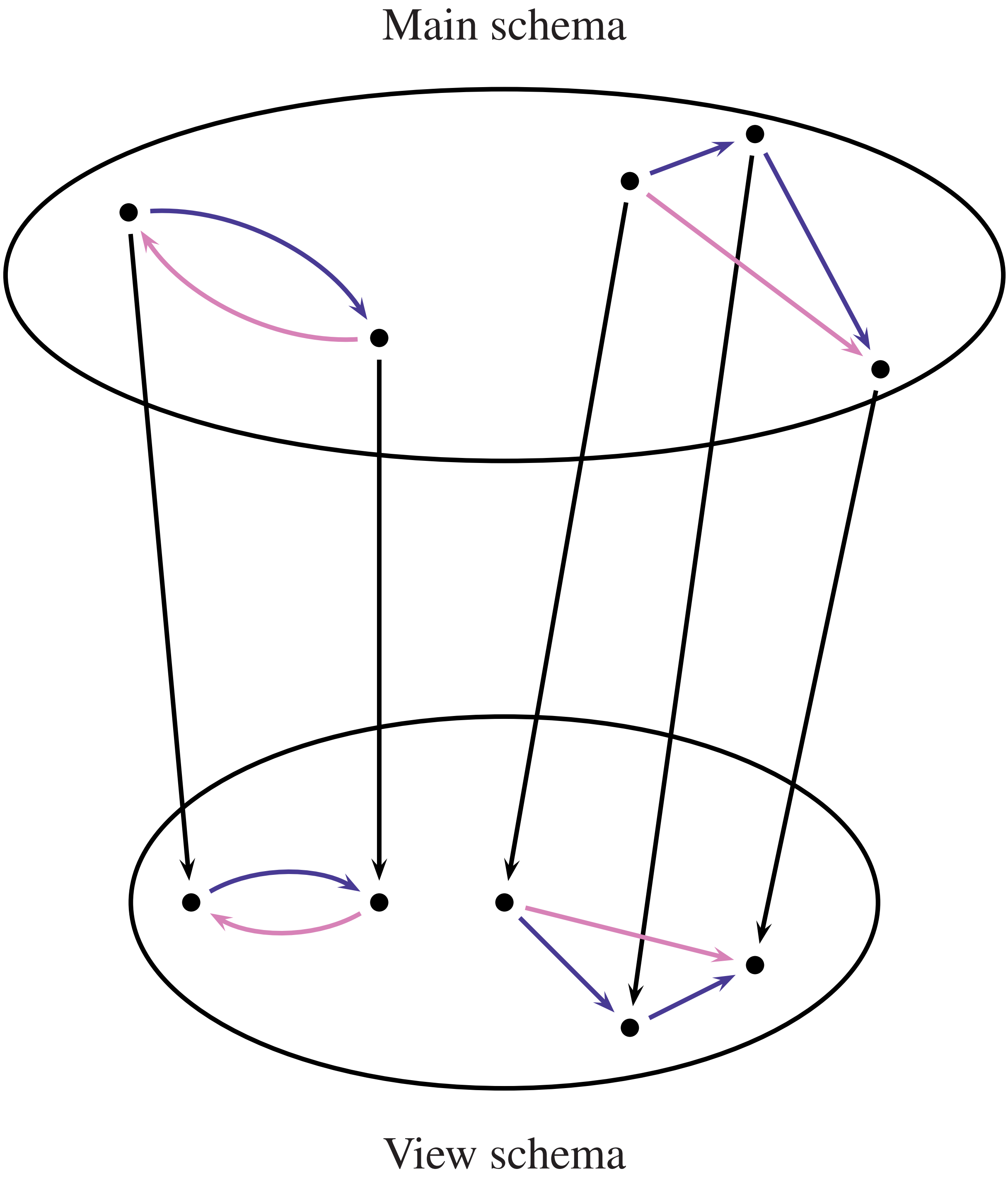
(upt:5) [View update reflection is transitive.]
If $\rho(M,N_1)\downarrow$ and $\rho(\rho(M,N_1),N_2)\downarrow$, then $\rho(M,N_2) = \rho(\rho(M,N_1),N_2)$.

Modelling convention: Typically

$$U = \mathsf{LDB}(\mathbf{D}) \times \mathsf{LDB}(\mathbf{D})$$

so that all possible updates are allowed on the base schema.

Illustration of Reversibility and Transitivity



Complement-Based Update Strategies

- Context:

$$\begin{array}{rl} \mathbf{D}: & \text{base schema} \\ \Gamma_1 = (\mathbf{V}_1,\ \gamma_1 : \mathbf{D} \to \mathbf{V}_1): & \text{a view of } \mathbf{D} \\ \Gamma_2 = (\mathbf{V}_2,\ \gamma_2 : \mathbf{D} \to \mathbf{V}_2): & \text{a view of } \mathbf{D} \end{array}$$

- The view Γ_2 is a *(subdirect) complement* of Γ_1 if the state of **D** may be recovered from the combined states of $\mathbf{V}_1$ and $\mathbf{V}_2$. Formally, the *decomposition mapping*

$$\begin{aligned} \gamma_1 \times \gamma_2 \ : \ \mathsf{LDB}(\mathbf{D}) &\to \mathsf{LDB}(\mathbf{V}_1) \times \mathsf{LDB}(\mathbf{V}_2) \\ M &\mapsto (\gamma_1(M), \gamma_2(M)) \end{aligned}$$

  must be injective.

- The terminology that $\{\Gamma_1, \Gamma_2\}$ forms a *lossless decomposition* of **D** is also in common use.

Observation [Bancilhon & Spyratos 81]: Every subdirect complement Γ_2 of Γ_1 defines an update strategy on Γ_1 as follows:

$$\mathsf{UpdStr}\langle\Gamma_1,\Gamma_2\rangle(M_1,N_2) =$$

$$\begin{cases} (\gamma_1 \times \gamma_2)^{-1}(N_2, \gamma_2(M_1)) & \text{if } (N_2, \gamma_2(M_1)) \in (\gamma_1 \times \gamma_2)(\mathsf{LDB}(\mathbf{D})) \\ \text{undefined} & \text{otherwise} \end{cases}$$

- This is called the update strategy on Γ_1 defined by *constant complement* Γ_2.

Constant Complement in the Relational Context

Example base schema and instance:

$(E[\mathsf{ABC}], \{\mathsf{B} \rightarrow \mathsf{C}\})$:

A	B	C
a_0	b_0	c_0
a_1	b_1	c_1
a_2	b_1	c_1

View to be updated:

View $\Pi_{\mathsf{AB}} = (E[\mathsf{AB}], \emptyset)$

A	B
a_0	b_0
a_1	b_1
a_2	b_1

Complementary view:

View $\Pi_{\mathsf{BC}} = (E[\mathsf{BC}], \{\mathsf{B} \rightarrow \mathsf{C}\})$

B	C
b_0	c_0
b_1	c_1

- This is a familiar lossless join-based decomposition, based upon the functional dependency $\mathsf{B} \rightarrow \mathsf{C}$.

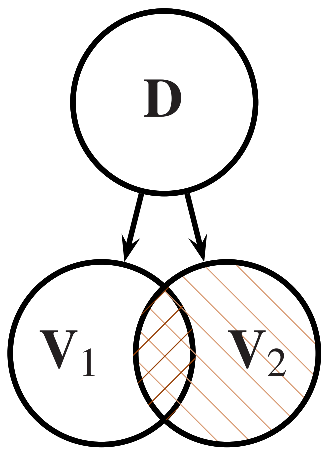
- The updates which are allowed to $E[\mathsf{AB}]$ under constant complement Π_{BC} are precisely those which hold the values which occur in column B fixed.

Equivalence of Strategies

Theorem [Bancilhon & Spyratos 81]: Every closed update strategy is defined by a constant complement update strategy. $\square$

Fact: There exist subdirect complements which do not define closed update families. $\square$

- To understand the problem, consider the following visualization, in which $\Gamma_1 = (\mathbf{V}_1, \gamma_1)$ is to be updated with constant complement $\Gamma_2 = (\mathbf{V}_2, \gamma_2)$.



- Intuitively, there is an "overlap" of the two views, induced by the mappings γ_1 and γ_2.
- When updating Γ_1, the part of the state of $\mathbf{V}_1$ which overlaps $\mathbf{V}_2$ must be held constant, while the rest may be modified at will.
- Formalization of this notion is the key to identifying just those complements which define closed update families.

Failure in the Relational Context

Example base schema and instance:

$(E[\mathsf{ABC}], \{\mathsf{A} \to \mathsf{C}, \mathsf{B} \to \mathsf{C}\})$:

A	B	C
a_0	b_0	c_0
a_1	b_1	c_1
a_2	b_1	c_1

View to be updated:

View $\Pi_{\mathsf{AB}} = (E[\mathsf{AB}], \emptyset)$

A	B
a_0	b_0
a_1	b_1
a_2	b_1

Complementary view:

View $\Pi_{\mathsf{BC}} = (E[\mathsf{BC}], \{\mathsf{B} \to \mathsf{C}\})$

B	C
b_0	c_0
b_1	c_1

- This is still a familiar lossless join-based decomposition, based upon the functional dependency $\mathsf{B} \to \mathsf{C}$.
- The decomposition is not *dependency preserving*, since $\mathsf{A} \to \mathsf{C}$ is lost.
- There is no nontrivial constant-complement update strategy for Π_{AB} which holds Π_{BC} fixed.
- It does not suffice to hold column B fixed.

Failure in the Relational Context

Example base schema and instance:

$(E[\mathsf{ABC}], \{\mathsf{A} \rightarrow \mathsf{C}, \mathsf{B} \rightarrow \mathsf{C}\})$:

A	B	C
a_0	b_0	c_0
a_1	b_1	c_1
a_2	b_1	c_1

View to be updated:

View $\Pi_{\mathsf{AB}} = (E[\mathsf{AB}], \emptyset)$

A	B
a_0	b_0
a_0 $\cancel{a_1}$	b_1
a_2	b_1

Complementary view:

View $\Pi_{\mathsf{BC}} = (E[\mathsf{BC}], \{\mathsf{B} \rightarrow \mathsf{C}\})$

B	C
b_0	c_0
b_1	c_1

- This is still a familiar lossless join-based decomposition, based upon the functional dependency $\mathsf{B} \rightarrow \mathsf{C}$.
- The decomposition is not *dependency preserving*, since $\mathsf{A} \rightarrow \mathsf{C}$ is lost.
- There is no nontrivial constant-complement update strategy for Π_{AB} which holds Π_{BC} fixed.
- It does not suffice to hold column B fixed.

Failure in the Relational Context

Example base schema and instance:

$(E[\mathsf{ABC}], \{\mathsf{A} \rightarrow \mathsf{C}, \mathsf{B} \rightarrow \mathsf{C}\})$:

A	B	C
a_0	b_0	c_0
a_0 $\cancel{a_1}$	b_1	$\cancel{c_1}$ c_0
a_2	b_1	$\cancel{c_1}$ c_0

View to be updated:

View $\Pi_{\mathsf{AB}} = (E[\mathsf{AB}], \emptyset)$

A	B
a_0	b_0
a_0 $\cancel{a_1}$	b_1
a_2	b_1

Complementary view:

View $\Pi_{\mathsf{BC}} = (E[\mathsf{BC}], \{\mathsf{B} \rightarrow \mathsf{C}\})$

B	C
b_0	c_0
b_1	$\cancel{c_1}$ c_0

- This is still a familiar lossless join-based decomposition, based upon the functional dependency $\mathsf{B} \rightarrow \mathsf{C}$.
- The decomposition is not *dependency preserving*, since $\mathsf{A} \rightarrow \mathsf{C}$ is lost.
- There is no nontrivial constant-complement update strategy for Π_{AB} which holds Π_{BC} fixed.
- It does not suffice to hold column B fixed.

Failure in the Relational Context

Example base schema and instance:

$(E[\mathsf{ABC}], \{\mathsf{A} \rightarrow \mathsf{C}, \mathsf{B} \rightarrow \mathsf{C}\})$:

A	B	C
a_0	b_0	c_0
a_1	b_1	c_0
a_2	b_1	c_0

View to be updated:

View $\Pi_{\mathsf{AB}} = (E[\mathsf{AB}], \emptyset)$

A	B
a_0	b_0
a_1	b_1
a_2	b_1

Complementary view:

View $\Pi_{\mathsf{BC}} = (E[\mathsf{BC}], \{\mathsf{B} \rightarrow \mathsf{C}\})$

B	C
b_0	c_0
b_1	c_0

- This is still a familiar lossless join-based decomposition, based upon the functional dependency $\mathsf{B} \rightarrow \mathsf{C}$.
- The decomposition is not *dependency preserving*, since $\mathsf{A} \rightarrow \mathsf{C}$ is lost.
- There is no nontrivial constant-complement update strategy for Π_{AB} which holds Π_{BC} fixed.
- It does not suffice to hold column B fixed.

Failure in the Relational Context

Example base schema and instance:

$(E[\mathsf{ABC}], \{\mathsf{A} \rightarrow \mathsf{C}, \mathsf{B} \rightarrow \mathsf{C}\})$:

A	B	C
a_0	b_0	c_0
a_0 $\cancel{a_1}$	b_1	c_0
a_2	b_1	c_0

View to be updated:

View $\Pi_{\mathsf{AB}} = (E[\mathsf{AB}], \emptyset)$

A	B
a_0	b_0
a_0 $\cancel{a_1}$	b_1
a_2	b_1

Complementary view:

View $\Pi_{\mathsf{BC}} =$
$(E[\mathsf{BC}], \{\mathsf{B} \rightarrow \mathsf{C}\})$

B	C
b_0	c_0
b_1	c_0

- This is still a familiar lossless join-based decomposition, based upon the functional dependency $\mathsf{B} \rightarrow \mathsf{C}$.
- The decomposition is not *dependency preserving*, since $\mathsf{A} \rightarrow \mathsf{C}$ is lost.
- There is no nontrivial constant-complement update strategy for Π_{AB} which holds Π_{BC} fixed.
- It does not suffice to hold column B fixed.

The Congruence of a View

- The *congruence* of a view identifies those pairs of states of the base schema which cannot be distinguished via the view.

- Formally, the *congruence* $\mathsf{Congr}(\Gamma)$ of a view $\Gamma = (\mathbf{V}, \gamma)$ of $\mathbf{D}$ is the equivalence relation defined by

$$(M_1, M_2) \in \mathsf{Congr}(\Gamma) \text{ iff } \gamma(M_1) = \gamma(M_2)$$

- In a set-based context without additional structure, every view is defined, up to isomorphism, by its congruence.

- Thus, there is a natural correspondence:

Equivalence relations on states of the base schema $\longleftrightarrow$ Isomorphism classes of views of the base schema

- Note that a closed update family U on $\mathbf{D}$ defines a view of $\mathbf{D}$, since it is an equivalence relation.

- Given a closed update strategy ρ for $\Gamma = (\mathbf{V}, \gamma)$,

$$\{(M_1, M_2) \in \mathsf{LDB}(\mathbf{D}) \times \mathsf{LDB}(\mathbf{D}) \mid (\exists N \in \mathsf{LDB}(\mathbf{V}))(\rho(M_1, N) = M_2)\}$$

is the congruence of a view complementary to Γ which yields ρ with constant-complement update.

Commuting Congruences and Meet Complements

- A pair $\{\Gamma_1, \Gamma_2\}$ of views of $\mathbf{D}$ is called a *fully commuting pair* if $\mathsf{Congr}(\Gamma_1) \circ \mathsf{Congr}(\Gamma_2) = \mathsf{Congr}(\Gamma_2) \circ \mathsf{Congr}(\Gamma_1)$.

- In this case, $\mathsf{Congr}(\Gamma_1) \circ \mathsf{Congr}(\Gamma_2)$ is an equivalence relation.

- If $\{\Gamma_1, \Gamma_2\}$ is a fully commuting pair, the view whose congruence is $\Gamma_1 \circ \Gamma_2$ is called the *meet* of $\Gamma_1 \circ \Gamma_2$, and is denoted $\Gamma_1 \wedge \Gamma_2 = (\mathbf{V}_1 {}_{\gamma_1}\!\wedge_{\gamma_2} \mathbf{V}_2, \gamma_1 \wedge \gamma_2)$.

- $\Gamma_1 \wedge \Gamma_2$ may also be regarded as a view of $\mathbf{V}_1$ and $\mathbf{V}_2$, as well as of $\mathbf{D}$.

$$
\begin{tikzcd}
 & \mathbf{D} \arrow[dl, "\gamma_1"'] \arrow[dr, "\gamma_2"] \arrow[dd, "\gamma_1 \wedge \gamma_2"'] & \\
\mathbf{V}_1 \arrow[dr, "\lambda\langle\Gamma_1, \Gamma_2\rangle"'] & & \mathbf{V}_2 \arrow[dl, "\lambda\langle\Gamma_2, \Gamma_1\rangle"] \\
 & \mathbf{V}_1 {}_{\gamma_1}\!\wedge_{\gamma_2} \mathbf{V}_2 &
\end{tikzcd}
$$

- $\Gamma_1 \wedge \Gamma_2$ is effectively a greatest lower bound of Γ_1 and Γ_2.

View Meet in the Relational Context

Example base schema:

$(E[\mathsf{ABC}], \{\mathsf{B} \to \mathsf{C}\})$:

A	B	C

View to be updated:

View $\Pi_{\mathsf{AB}} = (E[\mathsf{AB}], \emptyset)$

A	B

Complementary view:

View $\Pi_{\mathsf{BC}} =$
$(E[\mathsf{BC}], \{\mathsf{B} \to \mathsf{C}\})$

B	C

$$
\begin{tikzcd}
 & (E[\mathsf{ABC}], \{\mathsf{B} \to \mathsf{C}\}) \arrow[dl, "\pi_{\mathsf{AB}}"'] \arrow[dr, "\pi_{\mathsf{BC}}"] \arrow[dd, "\pi_{\mathsf{AB}} \wedge \pi_{\mathsf{BC}} = \pi_{\mathsf{B}}"] & \\
(E[\mathsf{AB}], \emptyset) \arrow[dr, "\pi_{\mathsf{B}}"] & & (E[\mathsf{BC}], \{\mathsf{B} \to \mathsf{C}\}) \arrow[dl, "\pi_{\mathsf{B}}"'] \\
 & (E[\mathsf{B}], \emptyset) &
\end{tikzcd}
$$

- This property fails if the functional dependency $\mathsf{A} \to \mathsf{C}$ is added to the main schema.
- In the relational context, a decomposition is meet complementary iff it is *dependency preserving*.

Meet Complements and Closed Update Strategies

Theorem [Hegner ICDT90, FoIKS02]: The update strategy $\mathsf{UpdStr}\langle \Gamma_1, \Gamma_2 \rangle$ is closed iff $\{\Gamma_1, \Gamma_2\}$ forms a fully commuting pair. $\Box$

- The view update family on Γ_1 defined by the meet complementary pair $\{\Gamma_1, \Gamma_2\}$ is just $\mathsf{Congr}(\Gamma_1 \wedge \Gamma_2)$ (with $\Gamma_1 \wedge \Gamma_2$ regarded as a view of $\mathbf{V}_1$).

  - In other words, the admissible updates on Γ_1 under constant complement Γ_2 are precisely those which keep the meet $\Gamma_1 \wedge \Gamma_2$ constant.

Bottom Line: For a given view Γ, there is a natural bijective correspondence:

$$\begin{array}{rcl} \text{Closed update strategies} & \longleftrightarrow & \text{Meet complements} \\ \rho & \longmapsto & \tilde{\Gamma}^{\rho} \\ \mathsf{UpdStr}\langle \Gamma, \Gamma' \rangle & \longleftarrow\!\!\!\shortmid & \Gamma' \end{array}$$

The Uniqueness Question

- Generally speaking, distinct complements give rise to distinct update strategies under constant-complement translation.
- In the general sets-and-mappings framework, complements are never unique, except in degenerate cases.
- This is true even for complements with the same meet.

Question: How does one choose the “right” complement to support update translation on a view?

Answer: Usually, this is done on æsthetic grounds, by selecting a most natural complement.

Observation: In many cases, there is an obvious ”natural complement” which appears to be the only one which makes sense.

Goal: Develop a formal *theory* which identifies this natural complement as the only reasonable one.

Alternate Update in the Relational Context

Base schema:

$(R[\mathsf{A}], \emptyset)$:

A
a_0
a_1
a_2

$(S[\mathsf{A}], \emptyset)$:

A
a_1
a_3
a_4

View to be updated:

View $\Pi_{R[\mathsf{A}]}$:

A
a_0
a_1
a_2

Natural Complement:

View $\Pi_{S[\mathsf{A}]}$:

A
a_1
a_3
a_4

Alternative Complement:

View $\Pi_{R\Delta S[\mathsf{A}]}$:

A
a_0
a_2
a_3
a_4

- $R\Delta S$ is the *symmetric difference* of R and S.

$$R\Delta S[\mathsf{A}] = (R[\mathsf{A}] \cup S[\mathsf{A}]) \setminus (R[\mathsf{A}] \cap S[\mathsf{A}])$$

Base schema after $\mathsf{Insert}(a_4, R[\mathsf{A}])$
with constant complement $\Pi_{R\Delta S[\mathsf{A}]}$:

$(R[\mathsf{A}], \emptyset)$:

A
a_0
a_1
a_2
a_4

$(S[\mathsf{A}], \emptyset)$:

A
a_1
a_3

Question: How may $\Pi_{R[\mathsf{A}]}$ be favored over $\Pi_{R\Delta S[\mathsf{A}]}$ on formal grounds?

Order — the Key to Update Uniqueness

- Database states in common data models often admit a natural order structure.

  Example: Relation-by-relation inclusion in the relational model.

- Database morphisms in common data models often preserve this order structure.

  Example: Of the six base operations in the relational algebra (select, project, join, union, intersection, difference), only difference fails to be monotonic under the natural ordering.

- These properties have been used to establish uniqueness of *direct* complements [Hegner94 JCSS].
  - The view $\Pi_{R\Delta S[\mathsf{A}]}$ may be ruled out using this theory.

Question: Is is possible to extend these results to subdirect complements?

Short Answer:

- It is not generally true that subdirect complements are unique, even in the presence of order constraints.

- Despite this, it can be shown that the reflections of updates which are order based (*i.e.*, insertions or deletions) are unique within the context of closed update strategies.

Key Features of the Order-Based Framework

- The legal databases of a schema $\mathbf{D}$ form a partially ordered set $(\mathsf{LDB}(\mathbf{D}), \leq_{\mathbf{D}})$.

- The mapping $\gamma : \mathsf{LDB}(\mathbf{D}) \to \mathsf{LDB}(\mathbf{V})$ of a view $\Gamma = (\mathbf{V}, \gamma)$ is an open poset morphism; *i.e.*, both of the following conditions hold.

  ➪ $M_1 \leq_{\mathbf{D}} M_2$ implies $\gamma(M_1) \leq_{\mathbf{V}} \gamma(M_2)$

  ➪ $\gamma(M_1) \leq_{\mathbf{V}} \gamma(M_2)$ implies
  $(\exists M_1' \in \gamma^{-1}(\gamma(M_1)))(\exists M_2' \in \gamma^{-1}(\gamma(M_2)))(M_1' \leq_{\mathbf{D}} M_2')$

- For $\{\Gamma_1 = (\mathbf{V}_1, \gamma_1), \Gamma_2 = (\mathbf{V}_2, \gamma_2)\}$ to be a pair of subdirect complements, the decomposition mapping

$$\gamma_1 \times \gamma_2 : \mathsf{LDB}(\mathbf{D}) \to \mathsf{LDB}(\mathbf{V}_1) \times \mathsf{LDB}(\mathbf{V}_2)$$

  must be a section (isomorphism into).

- A closed update family for $\mathbf{D}$ is an *order-compatible* equivalence relation on $\mathsf{LDB}(\mathbf{D})$.

Closed Update Strategies in the Order-Based Framework

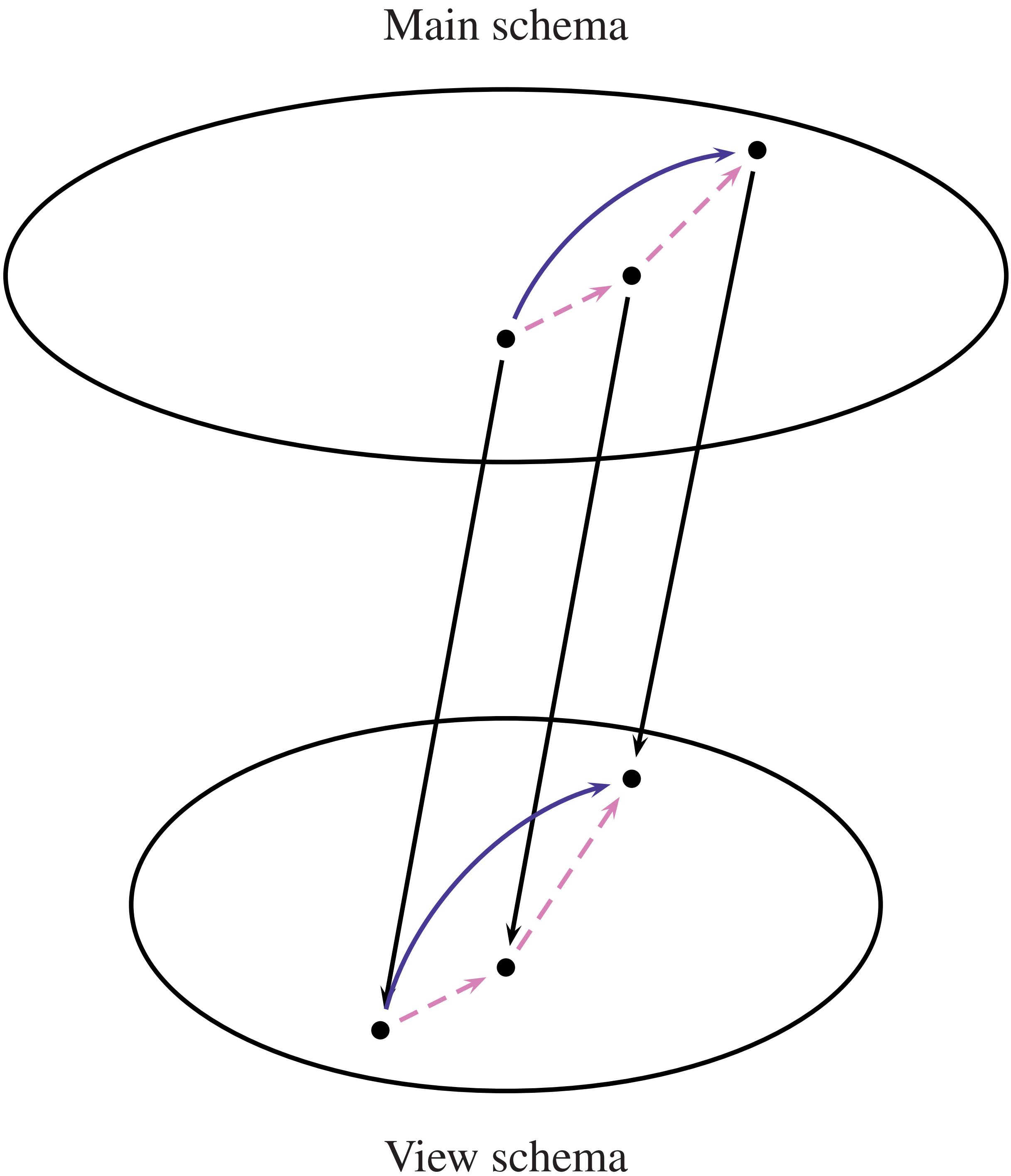
- Let T = an order-compatible closed update family for the view schema $\mathbf{V}$.
  U = an order-compatible closed update family for the base schema $\mathbf{D}$.
- In addition to conditions (upt:1) - (upt:5) on slide 14, the following conditions are imposed to yield an *order-based closed update strategy* for T with respect to U.

(upt:7) [This condition is called *order completeness*.]
If $\rho(M_1, N_1)\downarrow$ with $M_1 \leq_{\mathbf{D}} \rho(M_1, N_1)$, then for all $M_2 \in \mathsf{LDB}(\mathbf{D})$ with $M_1 \leq_{\mathbf{D}} M_2 \leq_{\mathbf{D}} \rho(M_1, N_1)$, there is an $N_2 \in \mathsf{LDB}(\mathbf{V})$ with $\rho(M_1, N_2) = M_2$ and $\rho(M_2, \gamma(\rho(M_1, N_1))) = \rho(M_1, N_1)$.

(upt:8) [This condition is called *order reflection*.]
If $M_1, M_2 \in \mathsf{LDB}(\mathbf{D})$, $N_1, N_2 \in \mathsf{LDB}(\mathbf{V})$ with $M_1 \leq_{\mathbf{D}} M_2$ and $N_1 \leq_{\mathbf{V}} N_2$, then $\rho(M_1, N_1) \leq_{\mathbf{D}} \rho(M_2, N_2)$ whenever both are defined.

Theorem: The order-based closed update strategies are defined precisely by the order-based complements. $\square$

Illustration of Order Reflection and Completeness



The Main Result

- Let $\mathbf{D}$ be a database schema, and let U be an order-compatible closed update family for $\mathbf{D}$. A pair $(M_1.M_2) \in U$ is called:
  (i) a *formal insertion* with respect to U if $M_1 \leq_{\mathbf{D}} M_2$;
  (ii) a *formal deletion* with respect to U if $M_2 \leq_{\mathbf{D}} M_1$;
  (iii) an *order-based update* with respect to U if it is realizable as a sequence of formal insertions and deletions.
- The update family U is called *order realizable* if every pair in U is an order-based update.

Main Theorem: (Set in the order-based framework.)

- Let:
  - $\mathbf{D}$ = database schema.
  - $\Gamma = (\mathbf{V}, \gamma)$ = view of $\mathbf{D}$.
  - U = closed update family for $\mathbf{D}$.
  - T = closed update family for $\mathbf{V}$.
  - ρ_1, ρ_2 = closed update strategies for for T with respect to U.
- Then:
  - For any $M \in \mathsf{LDB}(\mathbf{D})$ and $N \in \mathsf{LDB}(\mathbf{V})$ with $(\gamma(M), N) \in T$ an order-based update of $\mathbf{V}$, $\rho_1(M,N) = \rho_2(M,N)$.
  - In particular, if T is order realizable, then $\rho_1 = \rho_2$. $\square$

Alternate Update in the Relational Context

Base schema:

$(E[\mathsf{ABC}], \{\mathsf{B} \to \mathsf{C}\})$

A	B	C
a_0	b_0	c_0
a_1	b_1	c_1
a_2	b_1	c_1

View to be updated:

View $\Pi_{\mathsf{AB}} = (E[\mathsf{AB}], \emptyset)$

A	B
a_0	b_0
a_1	b_1
a_2	b_1

- For any $b \in \mathsf{Dom}(\mathsf{B})$, $\#_{\mathsf{A}}(b)$ = number of distinct values of for attribute A associated with b in $\pi_{\mathsf{AB}}(M)$.
- Let $\mathsf{Dom}(C) = \{c_0, c_1, c_2\}$
- $\alpha : \mathsf{Dom}(C) \to \mathsf{Dom}(C); c_i \mapsto c_{(i+1 \mod 3)}$.
- Define $\pi'_{BC}(M) =$ $\{(b,c) \mid (b,c) \in \pi_{BC}(M) \text{ and } \#_A(b) \text{ is odd}\} \cup \{(b,\alpha(c)) \mid (b,c) \in \pi_{BC}(M) \text{ and } \#_A(b) \text{ is even}\}$.
- The updates which are allowed to $E[\mathsf{AB}]$ under constant complement Π'_{BC} are exactly the same as those allowed under constant complement Π_{BC}.
- Consider the update Insert (a_1, b_0) into Π_{AB}.

Constant complement view:

$\Pi'_{\mathsf{BC}} = (E[\mathsf{BC}], \{\mathsf{B} \to \mathsf{C}\})$

B	C
b_0	c_0
b_1	c_2

Base schema after update:

$(E[\mathsf{ABC}], \{\mathsf{B} \to \mathsf{C}\})$

A	B	C
a_0	b_0	c_2
a_1	b_0	c_2
a_1	b_1	c_1
a_2	b_1	c_1

Update Uniqueness in a Relational Example

Base schema:

$(E[\mathsf{ABC}], \{\mathsf{B} \to \mathsf{C}\})$

A	B	C
a_0	b_0	c_0
a_1	b_1	c_1
a_2	b_1	c_1

View to be updated:

View $\Pi_{\mathsf{AB}} = (E[\mathsf{AB}], \emptyset)$

A	B
a_0	b_0
a_1	b_1
a_2	b_1

- Recall that the "natural" update strategy uses the complement Π_{BC}.
- This implies that the allowable updates T are those which hold the meet view Π_{B} constant.
- Observe that every admissible update to Π_{AB} is order based.
  - An update such as

    Replace (a_0, b_0) with (a_3, b_0)

    may be realized as the sequence:

    Insert (a_3, b_0)
    Delete (a_0, b_0).
- Thus, within the order-based framework, there is only one closed update strategy which supports T.
- The bizarre complement Π'_{BC} fails to be order based, and thus the update strategy defined by holding it constant is not within the order-based framework.

Update Uniqueness in a Relational Example II

Base schema:

$(E[\mathsf{ABC}], \{\mathsf{B} \to \mathsf{CA}\})$

A	B	C
a_0	b_0	c_0
a_1	b_1	c_1

View to be updated:

$\Pi_{\mathsf{AB}} = (E[\mathsf{AB}], \{\mathsf{B} \to \mathsf{A}\})$

A	B
a_0	b_0
a_1	b_1

- The additional functional dependency $\mathsf{B} \to \mathsf{A}$ blocks the ability to realize all updates as insertions followed by deletions.
- Using the theory just presented, the natural relational ordering cannot be used to guarantee the uniqueness of update reflections in a closed strategy.
- The solution is to make use of additional order structure which is available in the relational and many other data models.

Key Idea:

- Temporarily "forget" the FD $\mathsf{B} \to \mathsf{A}$.
- The update is then realizable as an insertion followed by a deletion; (*i.e.*, it is order based), and so has a unique reflection.
- If the update on the view Π_{AB} without $\mathsf{B} \to \mathsf{A}$ has a unique reflection to $E[\mathsf{ABC}]$, then this reflection is also unique when $\mathsf{B} \to \mathsf{A}$ is enforced.

Question: How is this key idea formalized within a general order-based framework?

PF-Schemata

- In the simple order model, each $M \in \mathsf{LDB}(\mathbf{D})$ is a single element; it has no further structure.

- It is therefore not clear how to distinguish databases which satisfy constraints from those which do not.

- *PF-schemata* address this limitation.

Definition: A *PF-schema* is a pair

$$\mathbf{D} = (\mathsf{SynFnd}(\mathbf{D}), \mathsf{LDB}(\mathbf{D}))$$

in which

- $\mathsf{SynFnd}(\mathbf{D})$ is a set.
  - ➪ Think of $\mathsf{SynFnd}(\mathbf{D})$ as a set of "tuples" or "entities" or "objects".
- $\mathsf{LDB}(\mathbf{D}) \subseteq \mathscr{P}_f(\mathsf{SynFnd}(\mathbf{D}))$ = all *finite* subsets of $\mathsf{SynFnd}(\mathbf{D})$.
  - ➪ Think of $\mathsf{LDB}(\mathbf{D})$ as a set of finite relations over $\mathsf{SynFnd}(\mathbf{D})$ (the *legal* or *constrained databases*).
  - ➪ $\mathsf{LDB}(\mathbf{D})$ has the natural ordering induced by $\subseteq$.

Definition: Define the set of *unconstrained databases* over $\mathbf{D}$:

$$\mathsf{DB}(\mathbf{D}) = \mathscr{P}_f(\mathsf{SynFnd}(\mathbf{D}))$$

Observation:

$$\mathsf{LDB}(\mathbf{D}) \subseteq \mathsf{DB}(\mathbf{D})$$

PF-Morphisms and PF-views

$$
\begin{array}{ccccc}
\mathbf{D}_1 & = & (\mathsf{SynFnd}(\mathbf{D}_1), & \mathsf{LDB}(\mathbf{D}_1)) & \mathsf{DB}(\mathbf{D}_1) \\
 & & \Big\downarrow f^{\sharp} \quad \overset{\text{induces}}{\Longrightarrow} & \Big\downarrow f^{\flat} & \Big\downarrow \bar{f}^{\flat} \\
\mathbf{D}_2 & = & (\mathsf{SynFnd}(\mathbf{D}_2) \cup \{\eta_{\mathbf{D}_2}\}, & \mathsf{LDB}(\mathbf{D}_2)) & \mathsf{DB}(\mathbf{D}_2)
\end{array}
$$

- $f^{\sharp}(t) = \eta_{\mathbf{D}_2}$ means that the "tuple" t is discarded by the mapping. $f^{\flat}$ and/or $\bar{f}^{\flat}$.

- This model recaptures operations such as *projection* and *selection* (even on multi-relational schemata), but not *join*.

Definition: A *PF-view* $\Gamma = (\mathbf{V}, \gamma)$ of the PF-schema $\mathbf{D}$ has the property that the "LDB" morphism is surjective.

$$\gamma^{\flat} : \mathsf{LDB}(\mathbf{D}) \to \mathsf{LDB}(\mathbf{V})$$

- Since PF-views may be regarded as order views in a natural way, the theory for order views applies to them.

- In particular, all order-based updates for a closed update strategy have a unique reflection to the main schema.

Example of a PF-schema

Main schema: Let $\mathbf{R}$ have two relations: $R_1[\mathsf{ABC}]$, $R_2[\mathsf{CDE}]$, with $\mathsf{A} \to \mathsf{BC}$ (on $R_1[\mathsf{ABC}]$).

- The (possibly infinite) domains of A, B, C, D, and E are $\mathsf{Dom}(\mathsf{A})$, $\mathsf{Dom}(\mathsf{B})$, $\mathsf{Dom}(\mathsf{C})$, $\mathsf{Dom}(\mathsf{E})$, and $\mathsf{Dom}(\mathsf{F})$.
- $$\begin{array}{ll} & \mathsf{SynFnd}(\mathbf{R}) = \\ & \{{}_1(a,b,c) \mid a \in \mathsf{Dom}(\mathsf{A}) \wedge b \in \mathsf{Dom}(\mathsf{B}) \wedge c \in \mathsf{Dom}(\mathsf{C}) \ \} \\ \cup & \{{}_2(c,d,e) \mid c \in \mathsf{Dom}(\mathsf{C}) \wedge d \in \mathsf{Dom}(\mathsf{D}) \wedge e \in \mathsf{Dom}(\mathsf{E}) \ \} \end{array}$$

View schema: $\mathbf{S}$ has just one relation $S[\mathsf{AB}]$, $\mathsf{A} \to \mathsf{B}$.

- $\mathsf{SynFnd}(\mathbf{S}) = \{(a,b) \mid a \in \mathsf{Dom}(\mathsf{A}) \wedge b \in \mathsf{Dom}(\mathsf{B})\}$

Null tuples: $$\begin{array}{ll} \mathsf{ExtSynFnd}(\mathbf{R}) & = \mathsf{SynFnd}(\mathbf{R}) \cup \{\eta_{\mathbf{R}}\} \\ \mathsf{ExtSynFnd}(\mathbf{S}) & = \mathsf{SynFnd}(\mathbf{S}) \cup \{\eta_{\mathbf{S}}\} \end{array}$$

View morphism on elements:

$$\begin{array}{rcccc} \omega^\sharp & : & \mathsf{ExtSynFnd}(\mathbf{R}) & \to & \mathsf{ExtSynFnd}(\mathbf{S}) \\ & & {}_1(a,b,c) & \mapsto & (a,b) \\ & & {}_2(c,d,e) & \mapsto & \eta_{\mathbf{S}} \\ & & \eta_{\mathbf{R}} & \mapsto & \eta_{\mathbf{S}} \end{array}$$

View morphism on (finite) databases:

$$\omega^\flat : \mathsf{LDB}(\mathbf{R}) \to \mathsf{LDB}(\mathbf{S})$$

is just the projection of $R_1[\mathsf{ABC}]$ onto $S[\mathsf{AB}]$.

➪ $R_2[\mathsf{CDE}]$ is ignored.

Summary of the Key Results – PF-Decompositions

- If a pair $\{\Gamma_1, \Gamma_2\}$ of PF-views forms a meet-complementary pair, then so too do their syntactic counterparts $\{\widehat{\Gamma}_1, \widehat{\Gamma}_2\}$ in which all constraints are ignored (*i.e.*, $\mathsf{DB}(\mathbf{V}_i)$ replaces $\mathsf{LDB}(\mathbf{V}_i)$.) [Only the join dependency governing the decomposition must be retained.]

- In the syntactic context of $\widehat{\Gamma}_1$, all updates with constant complement $\widehat{\Gamma}_2$ are order-based, and so have a unique reflection to the main schema.

- This implies that all such updates have unique representations in the constrained environment Γ_1 with constant complement Γ_2 as well.

Example: On slide 37, the update strategy on Π_{AB} obtained by holding Π_{BC} constant while ignoring the FD $\mathsf{B} \rightarrow \mathsf{A}$ on intermediate states is unique among closed strategies. In other words, there is no other “natural” complement which defines a different closed strategy.

Non-Mergeability of Update Strategies

- It has been established that under "natural" circumstances of closed strategies, the reflection of a view update to the main schema is unique.

- However, this does not imply that all such strategies are mergeable; *i.e.*, that there is a maximal update strategy.

- Example base schema: $(E[\mathsf{ABC}], \{\mathsf{A} \to \mathsf{B}, \mathsf{B} \to \mathsf{C}, \mathsf{C} \to \mathsf{A}\})$

A	B	C

- View to be updated: $\Pi_{\mathsf{AB}} = (E[\mathsf{AB}], \{\mathsf{A} \to \mathsf{B}, \mathsf{B} \to \mathsf{A}\})$

A	B

- Two choices for a meet-complementary view:

  - $\Pi_{\mathsf{AC}} = (E[\mathsf{AC}], \{\mathsf{A} \to \mathsf{C}, \mathsf{C} \to \mathsf{A}\})$

A	C

    meet = Π_{A}; updates change only B.

  - $\Pi_{\mathsf{BC}} = (E[\mathsf{BC}], \{\mathsf{B} \to \mathsf{C}, \mathsf{C} \to \mathsf{B}\})$

B	C

    meet = Π_{B}; updates change only A.

- There is no way to merge these update strategies.

- There is no reasonable way to prefer one complement over the other.

Conclusions and Further Directions

Conclusion:

- Order-based techniques may be used to establish strong uniqueness properties for closed update strategies on views.

Further directions:

- Unification of distinct closed update strategies.
- Relaxation of the properties to allow for limited classes of irreversible updates.
  - ⊳ Build upon early work of (Gottlob, Paolini, Ziacari 1988).
- *View-centered* schema design.

For further information:

- PS and/or PDF versions of all of my publications since 1985 may be found on my web site.

`http://www.cs.umu.se/~hegner`