Complements of Database Views: Uniqueness and Optimality Issues

Stephen J. Hegner
Umeå University
Department of Computing Science
SE-901 87 Umeå, Sweden
hegner@cs.umu.se
http://www.cs.umu.se/~hegner

Schemata, Databases, and Schema Morphisms

Context: State-based database schemata: A *database schema* $\mathbf{D}$ is characterized by a set $\mathrm{LDB}(\mathbf{D})$ of *legal databases*.

Schemata, Databases, and Schema Morphisms

Context: State-based database schemata: A *database schema* $\mathbf{D}$ is characterized by a set $\mathsf{LDB}(\mathbf{D})$ of *legal databases*.

- At each point in time, there is exactly one legal database.

Context: State-based database schemata: A *database schema* $\mathbf{D}$ is characterized by a set $\mathrm{LDB}(\mathbf{D})$ of *legal databases*.

- At each point in time, there is exactly one legal database.

Prototypical example: Relational schemata $\mathbf{D} = (\mathrm{Rels}(\mathbf{D}), \mathrm{Constr}(\mathbf{D}))$.

Schemata, Databases, and Schema Morphisms

Context: State-based database schemata: A *database schema* $\mathbf{D}$ is characterized by a set $\mathrm{LDB}(\mathbf{D})$ of *legal databases*.

- At each point in time, there is exactly one legal database.

Prototypical example: Relational schemata $\mathbf{D} = (\mathrm{Rels}(\mathbf{D}), \mathrm{Constr}(\mathbf{D}))$.

- $\mathrm{LDB}(\mathbf{D})$ = set of databases of $\mathbf{D}$ which satisfy the integrity constraints $\mathrm{Constr}(\mathbf{D})$.

Schemata, Databases, and Schema Morphisms

Context: State-based database schemata: A *database schema* $\mathbf{D}$ is characterized by a set $\mathrm{LDB}(\mathbf{D})$ of *legal databases*.

- At each point in time, there is exactly one legal database.

Prototypical example: Relational schemata $\mathbf{D} = (\mathrm{Rels}(\mathbf{D}), \mathrm{Constr}(\mathbf{D}))$.

- $\mathrm{LDB}(\mathbf{D})$ = set of databases of $\mathbf{D}$ which satisfy the integrity constraints $\mathrm{Constr}(\mathbf{D})$.

Database morphism: A morphism $f : \mathbf{D}_1 \to \mathbf{D}_2$ is characterized by a function $f : \mathrm{LDB}(\mathbf{D}_1) \to \mathrm{LDB}(\mathbf{D}_2)$.

Schemata, Databases, and Schema Morphisms

Context: State-based database schemata: A *database schema* $\mathbf{D}$ is characterized by a set $\mathrm{LDB}(\mathbf{D})$ of *legal databases*.

- At each point in time, there is exactly one legal database.

Prototypical example: Relational schemata $\mathbf{D} = (\mathrm{Rels}(\mathbf{D}), \mathrm{Constr}(\mathbf{D}))$.

  - $\mathrm{LDB}(\mathbf{D})$ = set of databases of $\mathbf{D}$ which satisfy the integrity constraints $\mathrm{Constr}(\mathbf{D})$.

Database morphism: A morphism $f : \mathbf{D}_1 \to \mathbf{D}_2$ is characterized by a function $f : \mathrm{LDB}(\mathbf{D}_1) \to \mathrm{LDB}(\mathbf{D}_2)$.

- In the relational model, such functions are typically defined in one of two ways: (Example for $R, S \in \mathrm{Rels}(\mathbf{D}_1)$, $T \in \mathrm{Rels}(\mathbf{D}_2)$)

Schemata, Databases, and Schema Morphisms

Context: State-based database schemata: A *database schema* $\mathbf{D}$ is characterized by a set $\mathrm{LDB}(\mathbf{D})$ of *legal databases*.

- At each point in time, there is exactly one legal database.

Prototypical example: Relational schemata $\mathbf{D} = (\mathrm{Rels}(\mathbf{D}), \mathrm{Constr}(\mathbf{D}))$.

  - $\mathrm{LDB}(\mathbf{D})$ = set of databases of $\mathbf{D}$ which satisfy the integrity constraints $\mathrm{Constr}(\mathbf{D})$.

Database morphism: A morphism $f : \mathbf{D}_1 \to \mathbf{D}_2$ is characterized by a function $f : \mathrm{LDB}(\mathbf{D}_1) \to \mathrm{LDB}(\mathbf{D}_2)$.

- In the relational model, such functions are typically defined in one of two ways: (Example for $R, S \in \mathrm{Rels}(\mathbf{D}_1)$, $T \in \mathrm{Rels}(\mathbf{D}_2)$)
  Relational algebra: $T[AC] = \pi_{AC}(R[AB] * S[BC])$

Schemata, Databases, and Schema Morphisms

Context: State-based database schemata: A *database schema* $\mathbf{D}$ is characterized by a set $\mathrm{LDB}(\mathbf{D})$ of *legal databases*.

- At each point in time, there is exactly one legal database.

Prototypical example: Relational schemata $\mathbf{D} = (\mathrm{Rels}(\mathbf{D}), \mathrm{Constr}(\mathbf{D}))$.

  - $\mathrm{LDB}(\mathbf{D})$ = set of databases of $\mathbf{D}$ which satisfy the integrity constraints $\mathrm{Constr}(\mathbf{D})$.

Database morphism: A morphism $f : \mathbf{D}_1 \to \mathbf{D}_2$ is characterized by a function $f : \mathrm{LDB}(\mathbf{D}_1) \to \mathrm{LDB}(\mathbf{D}_2)$.

- In the relational model, such functions are typically defined in one of two ways: (Example for $R, S \in \mathrm{Rels}(\mathbf{D}_1)$, $T \in \mathrm{Rels}(\mathbf{D}_2)$)

  Relational algebra: $T[AC] = \pi_{AC}(R[AB] * S[BC])$

  Relational calculus: $T(x,z) \Leftrightarrow (\exists y)((R(x,y) \wedge S(y,z))$.

Schemata, Databases, and Schema Morphisms

Context: State-based database schemata: A *database schema* $\mathbf{D}$ is characterized by a set $\mathrm{LDB}(\mathbf{D})$ of *legal databases*.

- At each point in time, there is exactly one legal database.

Prototypical example: Relational schemata $\mathbf{D} = (\mathrm{Rels}(\mathbf{D}), \mathrm{Constr}(\mathbf{D}))$.

- $\mathrm{LDB}(\mathbf{D})$ = set of databases of $\mathbf{D}$ which satisfy the integrity constraints $\mathrm{Constr}(\mathbf{D})$.

Database morphism: A morphism $f : \mathbf{D}_1 \to \mathbf{D}_2$ is characterized by a function $f : \mathrm{LDB}(\mathbf{D}_1) \to \mathrm{LDB}(\mathbf{D}_2)$.

- In the relational model, such functions are typically defined in one of two ways: (Example for $R, S \in \mathrm{Rels}(\mathbf{D}_1)$, $T \in \mathrm{Rels}(\mathbf{D}_2)$)
  - Relational algebra: $T[AC] = \pi_{AC}(R[AB] * S[BC])$
  - Relational calculus: $T(x,z) \Leftrightarrow (\exists y)((R(x,y) \wedge S(y,z))$.
- However, the results are not limited to the relational model in any way.

Database Views

- A *view* $\Gamma = (\mathbf{V}, \gamma)$ of the schema $\mathbf{D}$ is given by:

Database Views

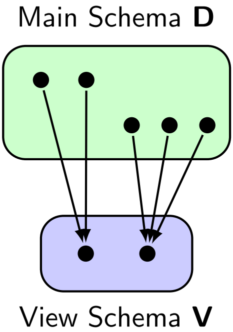
- A *view* $\Gamma = (\mathbf{V}, \gamma)$ of the schema $\mathbf{D}$ is given by:
  - A schema $\mathbf{V}$;

Database Views

- A *view* $\Gamma = (\mathbf{V}, \gamma)$ of the schema $\mathbf{D}$ is given by:
  - A schema $\mathbf{V}$;
  - A morphism $\gamma : \mathbf{D} \to \mathbf{V}$ for which $\gamma : \mathrm{LDB}(\mathbf{D}) \to \mathrm{LDB}(\mathbf{V})$ is surjective.

Database Views

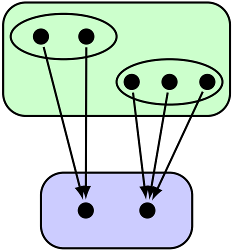
- A *view* $\Gamma = (\mathbf{V}, \gamma)$ of the schema $\mathbf{D}$ is given by:
  - A schema $\mathbf{V}$;
  - A morphism $\gamma : \mathbf{D} \to \mathbf{V}$ for which $\gamma : \mathrm{LDB}(\mathbf{D}) \to \mathrm{LDB}(\mathbf{V})$ is surjective.
- Surjectivity implies that the state of $\mathbf{V}$ is always determined completely by the state of $\mathbf{D}$.



- A *view* $\Gamma = (\mathbf{V}, \gamma)$ of the schema $\mathbf{D}$ is given by:
  - A schema $\mathbf{V}$;
  - A morphism $\gamma : \mathbf{D} \to \mathbf{V}$ for which $\gamma : \mathrm{LDB}(\mathbf{D}) \to \mathrm{LDB}(\mathbf{V})$ is surjective.
- Surjectivity implies that the state of $\mathbf{V}$ is always determined completely by the state of $\mathbf{D}$.

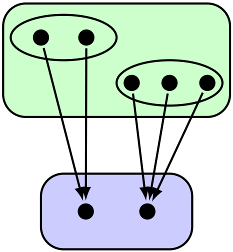
Congruence: The *congruence* $\mathrm{Congr}(\Gamma)$ is given by $\{(M_1, M_2) \in \mathrm{LDB}(\mathbf{D}) \mid \gamma(M_1) = \gamma(M_2)\}$.

Main Schema **D**



View Schema **V**

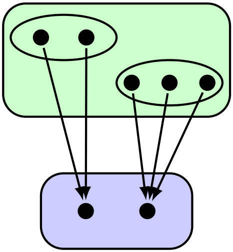
- A *view* $\Gamma = (\mathbf{V}, \gamma)$ of the schema $\mathbf{D}$ is given by:
  - A schema $\mathbf{V}$;
  - A morphism $\gamma : \mathbf{D} \to \mathbf{V}$ for which $\gamma : \mathrm{LDB}(\mathbf{D}) \to \mathrm{LDB}(\mathbf{V})$ is surjective.
- Surjectivity implies that the state of $\mathbf{V}$ is always determined completely by the state of $\mathbf{D}$.



Congruence: The *congruence* $\mathrm{Congr}(\Gamma)$ is given by $\{(M_1, M_2) \in \mathrm{LDB}(\mathbf{D}) \mid \gamma(M_1) = \gamma(M_2)\}$.

- There is a natural bijective correspondence between the states of $\mathbf{V}$ and the blocks of $\mathrm{Congr}(\Gamma)$.

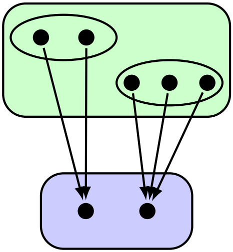
- A *view* $\Gamma = (\mathbf{V}, \gamma)$ of the schema $\mathbf{D}$ is given by:
  - A schema $\mathbf{V}$;
  - A morphism $\gamma : \mathbf{D} \to \mathbf{V}$ for which $\gamma : \mathrm{LDB}(\mathbf{D}) \to \mathrm{LDB}(\mathbf{V})$ is surjective.
- Surjectivity implies that the state of $\mathbf{V}$ is always determined completely by the state of $\mathbf{D}$.



Congruence: The *congruence* $\mathrm{Congr}(\Gamma)$ is given by $\{(M_1, M_2) \in \mathrm{LDB}(\mathbf{D}) \mid \gamma(M_1) = \gamma(M_2)\}$.

- There is a natural bijective correspondence between the states of $\mathbf{V}$ and the blocks of $\mathrm{Congr}(\Gamma)$.
- Thus, view construction is fundamentally a *quotient* operation, and not a *subset* operation.

- A *view* $\Gamma = (\mathbf{V}, \gamma)$ of the schema $\mathbf{D}$ is given by:
  - A schema $\mathbf{V}$;
  - A morphism $\gamma : \mathbf{D} \to \mathbf{V}$ for which $\gamma : \mathrm{LDB}(\mathbf{D}) \to \mathrm{LDB}(\mathbf{V})$ is surjective.
- Surjectivity implies that the state of $\mathbf{V}$ is always determined completely by the state of $\mathbf{D}$.



Congruence: The *congruence* $\mathrm{Congr}(\Gamma)$ is given by $\{(M_1, M_2) \in \mathrm{LDB}(\mathbf{D}) \mid \gamma(M_1) = \gamma(M_2)\}$.

- There is a natural bijective correspondence between the states of $\mathbf{V}$ and the blocks of $\mathrm{Congr}(\Gamma)$.
- Thus, view construction is fundamentally a *quotient* operation, and not a *subset* operation.
- For the purposes of this work, views with identical congruences are considered to be *isomorphic*.

Implied Constraints on the View

- The constraints Constr(**V**) of **V** are completely determined by the constraints of Constr(**D**).

Implied Constraints on the View

- The constraints Constr(**V**) of **V** are completely determined by the constraints of Constr(**D**).
- In the relational model, simple constraints on **D** can nevertheless result in complex constraints on **V**.

Implied Constraints on the View

- The constraints Constr($\mathbf{V}$) of $\mathbf{V}$ are completely determined by the constraints of Constr($\mathbf{D}$).
- In the relational model, simple constraints on $\mathbf{D}$ can nevertheless result in complex constraints on $\mathbf{V}$.

Example: $\mathbf{D} = (R[ABCD], \{A \to D, B \to D, CD \to A\})$

$\Gamma = \Pi^{\mathbf{D}}_{ABC}$ = projection of $R[ABCD]$ onto $R[ABC]$

admits no finite basis of first-order constraints.

Implied Constraints on the View

- The constraints Constr($\mathbf{V}$) of $\mathbf{V}$ are completely determined by the constraints of Constr($\mathbf{D}$).
- In the relational model, simple constraints on $\mathbf{D}$ can nevertheless result in complex constraints on $\mathbf{V}$.

Example: $\mathbf{D} = (R[ABCD], \{A \to D, B \to D, CD \to A\})$
$\Gamma = \Pi^{\mathbf{D}}_{ABC}$ = projection of $R[ABCD]$ onto $R[ABC]$
admits no finite basis of first-order constraints.

- Constr($\mathbf{V}$) is not finitely axiomatizable.

Implied Constraints on the View

- The constraints Constr($\mathbf{V}$) of $\mathbf{V}$ are completely determined by the constraints of Constr($\mathbf{D}$).
- In the relational model, simple constraints on $\mathbf{D}$ can nevertheless result in complex constraints on $\mathbf{V}$.

Example: $\mathbf{D} = (R[ABCD], \{A \to D, B \to D, CD \to A\})$
$\Gamma = \Pi^{\mathbf{D}}_{ABC}$ = projection of $R[ABCD]$ onto $R[ABC]$
admits no finite basis of first-order constraints.

- Constr($\mathbf{V}$) is not finitely axiomatizable.

Example:

$$\begin{array}{lcl} \mathbf{D} & = & (R[AB], \{A \to B\}) \\ \Gamma = (\mathbf{V}, \gamma) & = & \Pi^{\mathbf{D}}_{A+B} \text{ with } \gamma : r \mapsto (\pi_A(r), \pi_B(r)) \end{array}$$

admits no first-order axiomatization for infinite models.

Implied Constraints on the View

- The constraints Constr($\mathbf{V}$) of $\mathbf{V}$ are completely determined by the constraints of Constr($\mathbf{D}$).
- In the relational model, simple constraints on $\mathbf{D}$ can nevertheless result in complex constraints on $\mathbf{V}$.

Example: $\mathbf{D} = (R[ABCD], \{A \to D, B \to D, CD \to A\})$
$\Gamma = \Pi^{\mathbf{D}}_{ABC}$ = projection of $R[ABCD]$ onto $R[ABC]$
admits no finite basis of first-order constraints.

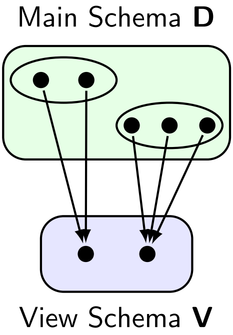
- Constr($\mathbf{V}$) is not finitely axiomatizable.

Example:
$$\begin{aligned} \mathbf{D} &= (R[AB], \{A \to B\}) \\ \Gamma = (\mathbf{V}, \gamma) &= \Pi^{\mathbf{D}}_{A+B} \text{ with } \gamma : r \mapsto (\pi_A(r), \pi_B(r)) \end{aligned}$$
admits no first-order axiomatization for infinite models.

- $\text{Constr}(\mathbf{V}) = \{\text{Card}(R[B]) \leq \text{Card}(R[A])\}$.

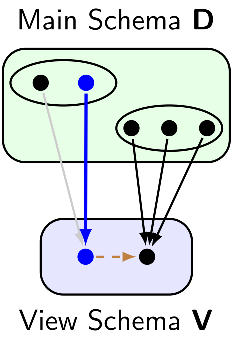
The View-Update Problem

Context: A view $\Gamma = (\mathbf{V}, \gamma)$ of the schema $\mathbf{D}$.



Context: A view $\Gamma = (\mathbf{V}, \gamma)$ of the schema $\mathbf{D}$.

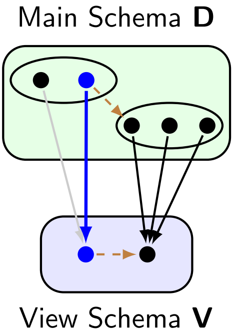
- Given the state of the main schema and a view update ...



The View-Update Problem

Context: A view $\Gamma = (\mathbf{V}, \gamma)$ of the schema $\mathbf{D}$.

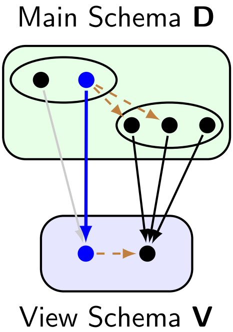
- Given the state of the main schema and a view update ...
- there are in general many possible *reflections* of that view update to the main schema.



The View-Update Problem

Context: A view $\Gamma = (\mathbf{V}, \gamma)$ of the schema $\mathbf{D}$.

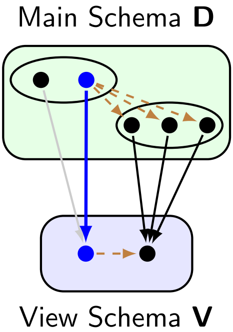
- Given the state of the main schema and a view update ...
- there are in general many possible *reflections* of that view update to the main schema.



The View-Update Problem

Context: A view $\Gamma = (\mathbf{V}, \gamma)$ of the schema $\mathbf{D}$.

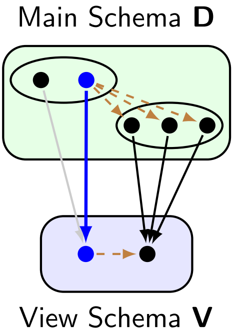
- Given the state of the main schema and a view update ...
- there are in general many possible *reflections* of that view update to the main schema.



The View-Update Problem

Context: A view $\Gamma = (\mathbf{V}, \gamma)$ of the schema $\mathbf{D}$.

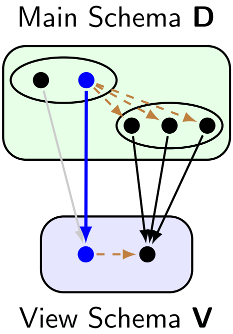
- Given the state of the main schema and a view update ...
- there are in general many possible *reflections* of that view update to the main schema.
- Note that there is always at least one.



The View-Update Problem

Context: A view $\Gamma = (\mathbf{V}, \gamma)$ of the schema $\mathbf{D}$.

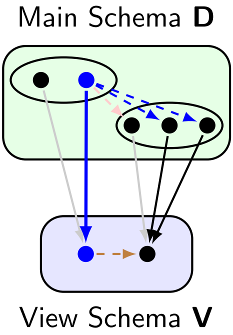
- Given the state of the main schema and a view update ...
- there are in general many possible *reflections* of that view update to the main schema.
- Note that there is always at least one.
- The *view-update problem* is to determine:



The View-Update Problem

Context: A view $\Gamma = (\mathbf{V}, \gamma)$ of the schema $\mathbf{D}$.

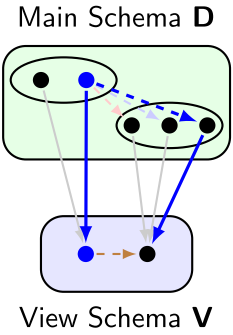
- Given the state of the main schema and a view update ...
- there are in general many possible *reflections* of that view update to the main schema.
- Note that there is always at least one.
- The *view-update problem* is to determine:
  - which reflections, if any, are suitable; and



The View-Update Problem

Context: A view $\Gamma = (\mathbf{V}, \gamma)$ of the schema $\mathbf{D}$.

- Given the state of the main schema and a view update ...
- there are in general many possible *reflections* of that view update to the main schema.
- Note that there is always at least one.
- The *view-update problem* is to determine:
  - which reflections, if any, are suitable; and
  - if there is more than one suitable choice, which is best.



Approaches to the View-Update Problem

- Three main classifications of most work on this problem:

Approaches to the View-Update Problem

- Three main classifications of most work on this problem:

Direct modelling:

Minimal/least change:

Constant complement:

Approaches to the View-Update Problem

- Three main classifications of most work on this problem:

Direct modelling:

- Look for direct solutions, usually using the relational algebra and null values.

Minimal/least change:

Constant complement:

Approaches to the View-Update Problem

- Three main classifications of most work on this problem:

Direct modelling:

- Look for direct solutions, usually using the relational algebra and null values.
- "Bag-of-tricks" approaches rather than comprehensive theories.

Minimal/least change:

Constant complement:

Approaches to the View-Update Problem

- Three main classifications of most work on this problem:

Direct modelling:

- Look for direct solutions, usually using the relational algebra and null values.
- "Bag-of-tricks" approaches rather than comprehensive theories.

Minimal/least change:

- A measure of distance between database states is identified.

Constant complement:

Approaches to the View-Update Problem

- Three main classifications of most work on this problem:

Direct modelling:

- Look for direct solutions, usually using the relational algebra and null values.
- "Bag-of-tricks" approaches rather than comprehensive theories.

Minimal/least change:

- A measure of distance between database states is identified.
- For reflected updates, smaller is better (intuitively, fewer changes).

Constant complement:

Approaches to the View-Update Problem

- Three main classifications of most work on this problem:

Direct modelling:

- Look for direct solutions, usually using the relational algebra and null values.
- "Bag-of-tricks" approaches rather than comprehensive theories.

Minimal/least change:

- A measure of distance between database states is identified.
- For reflected updates, smaller is better (intuitively, fewer changes).
- This approach is a favorite in the deductive-database community.

Constant complement:

Approaches to the View-Update Problem

- Three main classifications of most work on this problem:

Direct modelling:

- Look for direct solutions, usually using the relational algebra and null values.
- "Bag-of-tricks" approaches rather than comprehensive theories.

Minimal/least change:

- A measure of distance between database states is identified.
- For reflected updates, smaller is better (intuitively, fewer changes).
- This approach is a favorite in the deductive-database community.
- But it has also been applied in the state-based context.

Constant complement:

Approaches to the View-Update Problem

- Three main classifications of most work on this problem:

Direct modelling:

- Look for direct solutions, usually using the relational algebra and null values.
- "Bag-of-tricks" approaches rather than comprehensive theories.

Minimal/least change:

- A measure of distance between database states is identified.
- For reflected updates, smaller is better (intuitively, fewer changes).
- This approach is a favorite in the deductive-database community.
- But it has also been applied in the state-based context.

Constant complement:

- In updating view Γ, identify a second view Γ' which recaptures the "rest" of the main schema $\mathbf{D}$.

Approaches to the View-Update Problem

- Three main classifications of most work on this problem:

Direct modelling:

- Look for direct solutions, usually using the relational algebra and null values.
- "Bag-of-tricks" approaches rather than comprehensive theories.

Minimal/least change:

- A measure of distance between database states is identified.
- For reflected updates, smaller is better (intuitively, fewer changes).
- This approach is a favorite in the deductive-database community.
- But it has also been applied in the state-based context.

Constant complement:

- In updating view Γ, identify a second view Γ' which recaptures the "rest" of the main schema $\mathbf{D}$.
- Updates to Γ must keep Γ' constant.

Approaches to the View-Update Problem

- Three main classifications of most work on this problem:

Direct modelling:

- Look for direct solutions, usually using the relational algebra and null values.
- "Bag-of-tricks" approaches rather than comprehensive theories.

Minimal/least change:

- A measure of distance between database states is identified.
- For reflected updates, smaller is better (intuitively, fewer changes).
- This approach is a favorite in the deductive-database community.
- But it has also been applied in the state-based context.

Constant complement:

- In updating view Γ, identify a second view Γ' which recaptures the "rest" of the main schema $\mathbf{D}$.
- Updates to Γ must keep Γ' constant.
- Support for this approach is the main focus of this presentation.

A Concise Formulation of View Update

- $\text{Updates}\langle \mathbf{D} \rangle = \text{LDB}(\mathbf{D}) \times \text{LDB}(\mathbf{D})$ for any schema $\mathbf{D}$.

A Concise Formulation of View Update

- $\text{Updates}\langle \mathbf{D} \rangle = \text{LDB}(\mathbf{D}) \times \text{LDB}(\mathbf{D})$ for any schema $\mathbf{D}$.

Context: Main schema $\mathbf{D}$, view $\Gamma = (\mathbf{V}, \gamma)$.

A Concise Formulation of View Update

- Updates$\langle \mathbf{D} \rangle = \mathrm{LDB}(\mathbf{D}) \times \mathrm{LDB}(\mathbf{D})$ for any schema $\mathbf{D}$.

Context: Main schema $\mathbf{D}$, view $\Gamma = (\mathbf{V}, \gamma)$.

- A *translation* (*reflection*) of $(N_1, N_2) \in$ Updates$\langle \mathbf{V} \rangle$ with respect to $M_1 \in \mathrm{LDB}(\mathbf{D})$ with $\gamma(M_1) = N_1$ is an $M_2 \in \mathrm{LDB}(\mathbf{D})$ with $\gamma(M_2) = N_2$.

$$\begin{array}{ccc} M_1 & \longmapsto & M_2 \\ \downarrow \gamma & & \downarrow \gamma \\ N_1 & \longmapsto & N_2 \end{array}$$

A Concise Formulation of View Update

- Updates$\langle \mathbf{D} \rangle = \mathrm{LDB}(\mathbf{D}) \times \mathrm{LDB}(\mathbf{D})$ for any schema $\mathbf{D}$.

Context: Main schema $\mathbf{D}$, view $\Gamma = (\mathbf{V}, \gamma)$.

- A *translation* (*reflection*) of $(N_1, N_2) \in$ Updates$\langle \mathbf{V} \rangle$ with respect to $M_1 \in \mathrm{LDB}(\mathbf{D})$ with $\gamma(M_1) = N_1$ is an $M_2 \in \mathrm{LDB}(\mathbf{D})$ with $\gamma(M_2) = N_2$.

$$\begin{array}{ccc} M_1 & \longmapsto & M_2 \\ \downarrow \gamma & & \downarrow \gamma \\ N_1 & \longmapsto & N_2 \end{array}$$

- Everything is specified by:

A Concise Formulation of View Update

- $\text{Updates}\langle \mathbf{D} \rangle = \text{LDB}(\mathbf{D}) \times \text{LDB}(\mathbf{D})$ for any schema $\mathbf{D}$.

Context: Main schema $\mathbf{D}$, view $\Gamma = (\mathbf{V}, \gamma)$.

- A *translation* (*reflection*) of $(N_1, N_2) \in \text{Updates}\langle \mathbf{V} \rangle$ with respect to $M_1 \in \text{LDB}(\mathbf{D})$ with $\gamma(M_1) = N_1$ is an $M_2 \in \text{LDB}(\mathbf{D})$ with $\gamma(M_2) = N_2$.

$$\begin{array}{ccc} M_1 & \longmapsto & M_2 \\ \downarrow \gamma & & \downarrow \gamma \\ N_1 & \longmapsto & N_2 \end{array}$$

- Everything is specified by:
  - the current state M_1 of the main schema; and

A Concise Formulation of View Update

- Updates$\langle \mathbf{D} \rangle = \text{LDB}(\mathbf{D}) \times \text{LDB}(\mathbf{D})$ for any schema $\mathbf{D}$.

Context: Main schema $\mathbf{D}$, view $\Gamma = (\mathbf{V}, \gamma)$.

- A *translation* (*reflection*) of $(N_1, N_2) \in$ Updates$\langle \mathbf{V} \rangle$ with respect to $M_1 \in \text{LDB}(\mathbf{D})$ with $\gamma(M_1) = N_1$ is an $M_2 \in \text{LDB}(\mathbf{D})$ with $\gamma(M_2) = N_2$.

$$\begin{array}{ccc} M_1 & \longmapsto & M_2 \\ \downarrow \gamma & & \downarrow \gamma \\ N_1 & \longmapsto & N_2 \end{array}$$

- Everything is specified by:
  - the current state M_1 of the main schema; and
  - the desired new state N_2 of the view schema.

A Concise Formulation of View Update

- $\text{Updates}\langle \mathbf{D} \rangle = \text{LDB}(\mathbf{D}) \times \text{LDB}(\mathbf{D})$ for any schema $\mathbf{D}$.

Context: Main schema $\mathbf{D}$, view $\Gamma = (\mathbf{V}, \gamma)$.

- A *translation* (*reflection*) of $(N_1, N_2) \in \text{Updates}\langle \mathbf{V} \rangle$ with respect to $M_1 \in \text{LDB}(\mathbf{D})$ with $\gamma(M_1) = N_1$ is an $M_2 \in \text{LDB}(\mathbf{D})$ with $\gamma(M_2) = N_2$.

$$\begin{array}{ccc} M_1 & \longmapsto & M_2 \\ \downarrow \gamma & & \downarrow \gamma \\ N_1 & \longmapsto & N_2 \end{array}$$

- Everything is specified by:
  - the current state M_1 of the main schema; and
  - the desired new state N_2 of the view schema.
- N_1 is recaptured as $\gamma(M_1)$.

A Concise Formulation of View Update

- $\text{Updates}\langle \mathbf{D} \rangle = \text{LDB}(\mathbf{D}) \times \text{LDB}(\mathbf{D})$ for any schema $\mathbf{D}$.

Context: Main schema $\mathbf{D}$, view $\Gamma = (\mathbf{V}, \gamma)$.

- A *translation* (*reflection*) of $(N_1, N_2) \in \text{Updates}\langle \mathbf{V} \rangle$ with respect to $M_1 \in \text{LDB}(\mathbf{D})$ with $\gamma(M_1) = N_1$ is an $M_2 \in \text{LDB}(\mathbf{D})$ with $\gamma(M_2) = N_2$.

$$\begin{array}{ccc} M_1 & \longmapsto & M_2 \\ \downarrow \gamma & & \downarrow \gamma \\ N_1 & \longmapsto & N_2 \end{array}$$

- Everything is specified by:
  - the current state M_1 of the main schema; and
  - the desired new state N_2 of the view schema.
- N_1 is recaptured as $\gamma(M_1)$.

Update request: Formally, an *update request* from Γ to $\mathbf{D}$ is a pair $(M_1, N_2) \in \text{LDB}(\mathbf{D}) \times \text{LDB}(\mathbf{V})$.

A Concise Formulation of View Update

- Updates$\langle \mathbf{D} \rangle = \mathrm{LDB}(\mathbf{D}) \times \mathrm{LDB}(\mathbf{D})$ for any schema $\mathbf{D}$.

Context: Main schema $\mathbf{D}$, view $\Gamma = (\mathbf{V}, \gamma)$.

- A *translation* (*reflection*) of $(N_1, N_2) \in$ Updates$\langle \mathbf{V} \rangle$ with respect to $M_1 \in \mathrm{LDB}(\mathbf{D})$ with $\gamma(M_1) = N_1$ is an $M_2 \in \mathrm{LDB}(\mathbf{D})$ with $\gamma(M_2) = N_2$.

$$\begin{array}{ccc} M_1 & \longmapsto & M_2 \\ \downarrow\gamma & & \downarrow\gamma \\ N_1 & \longmapsto & N_2 \end{array}$$

- Everything is specified by:
  - the current state M_1 of the main schema; and
  - the desired new state N_2 of the view schema.
- N_1 is recaptured as $\gamma(M_1)$.

Update request: Formally, an *update request* from Γ to $\mathbf{D}$ is a pair $(M_1, N_2) \in \mathrm{LDB}(\mathbf{D}) \times \mathrm{LDB}(\mathbf{V})$.

Realization: A *realization* of (M_1, N_2) along Γ is a translation of $(\gamma(M_1), N_2)$ with respect to M_1.

View Complements and the Constant-Complement Approach

- The view $\Gamma' = (\mathbf{V}', \gamma')$ is a complement of $\Gamma = (\mathbf{V}, \gamma)$ if the *decomposition morphism*

$$\begin{array}{rcl} \gamma \times \gamma' \;:\; \mathsf{LDB}(\mathbf{D}) & \rightarrow & \mathsf{LDB}(\mathbf{V}) \times \mathsf{LDB}(\mathbf{V}') \\ M & \mapsto & \langle \gamma(M), \gamma'(M) \rangle \end{array}$$

  is injective.

View Complements and the Constant-Complement Approach

- The view $\Gamma' = (\mathbf{V}', \gamma')$ is a complement of $\Gamma = (\mathbf{V}, \gamma)$ if the *decomposition morphism*

$$\gamma \times \gamma' : \begin{array}{rcl} \mathrm{LDB}(\mathbf{D}) & \to & \mathrm{LDB}(\mathbf{V}) \times \mathrm{LDB}(\mathbf{V}') \\ M & \mapsto & \langle \gamma(M), \gamma'(M) \rangle \end{array}$$

  is injective.

Observation: [Bancilhon & Spyratos 1981] If $\Gamma' = (\mathbf{V}', \gamma')$ is a complement of $\Gamma = (\mathbf{V}, \gamma)$, then for any update request (M_1, N_2) from Γ to $\mathbf{D}$, there is at most one realization which keeps the state of Γ' constant.

View Complements and the Constant-Complement Approach

- The view $\Gamma' = (\mathbf{V}', \gamma')$ is a complement of $\Gamma = (\mathbf{V}, \gamma)$ if the *decomposition morphism*
$$\begin{array}{rrcl} \gamma \times \gamma' : & \mathsf{LDB}(\mathbf{D}) & \to & \mathsf{LDB}(\mathbf{V}) \times \mathsf{LDB}(\mathbf{V}') \\ & M & \mapsto & \langle \gamma(M), \gamma'(M) \rangle \end{array}$$
is injective.

Observation: [Bancilhon & Spyratos 1981] If $\Gamma' = (\mathbf{V}', \gamma')$ is a complement of $\Gamma = (\mathbf{V}, \gamma)$, then for any update request (M_1, N_2) from Γ to $\mathbf{D}$, there is at most one realization which keeps the state of Γ' constant.

Proof: This realization must be $(M_1, (\gamma \times \gamma')^{-1}((N_1, \gamma_2(M_1))))$. $\square$

- The view $\Gamma' = (\mathbf{V}', \gamma')$ is a complement of $\Gamma = (\mathbf{V}, \gamma)$ if the *decomposition morphism*

$$\begin{array}{rrcl} \gamma \times \gamma' : & \mathrm{LDB}(\mathbf{D}) & \to & \mathrm{LDB}(\mathbf{V}) \times \mathrm{LDB}(\mathbf{V}') \\ & M & \mapsto & \langle \gamma(M), \gamma'(M) \rangle \end{array}$$

  is injective.

Observation: [Bancilhon & Spyratos 1981] If $\Gamma' = (\mathbf{V}', \gamma')$ is a complement of $\Gamma = (\mathbf{V}, \gamma)$, then for any update request (M_1, N_2) from Γ to $\mathbf{D}$, there is at most one realization which keeps the state of Γ' constant.

Proof: This realization must be $(M_1, (\gamma \times \gamma')^{-1}((N_1, \gamma_2(M_1))))$. □

Familiar example: $\mathbf{E}_0 = (R[ABC], \{B \to C\})$.

$$\begin{array}{c} \{B \to C\} \\ R[ABC] \end{array}$$

View Complements and the Constant-Complement Approach

- The view $\Gamma' = (\mathbf{V}', \gamma')$ is a complement of $\Gamma = (\mathbf{V}, \gamma)$ if the *decomposition morphism* $\begin{array}{rrcl} \gamma \times \gamma' : & \mathrm{LDB}(\mathbf{D}) & \to & \mathrm{LDB}(\mathbf{V}) \times \mathrm{LDB}(\mathbf{V}') \\ & M & \mapsto & \langle \gamma(M), \gamma'(M) \rangle \end{array}$

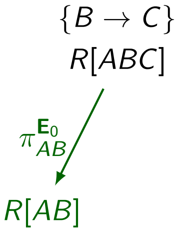
  is injective.

Observation: [Bancilhon & Spyratos 1981] If $\Gamma' = (\mathbf{V}', \gamma')$ is a complement of $\Gamma = (\mathbf{V}, \gamma)$, then for any update request (M_1, N_2) from Γ to $\mathbf{D}$, there is at most one realization which keeps the state of Γ' constant.

Proof: This realization must be $(M_1, (\gamma \times \gamma')^{-1}((N_1, \gamma_2(M_1))))$. $\square$

Familiar example: $\mathbf{E}_0 = (R[ABC], \{B \to C\})$.

View to be updated: $\Pi_{AB}^{\mathbf{E}_0} = (\mathbf{E}_0^{AB}, \pi_{AB}^{\mathbf{E}_0})$.



View Complements and the Constant-Complement Approach

- The view $\Gamma' = (\mathbf{V}', \gamma')$ is a complement of $\Gamma = (\mathbf{V}, \gamma)$ if the *decomposition morphism*

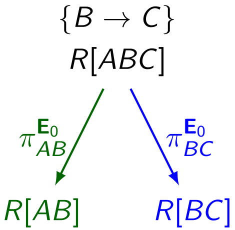
$$\begin{array}{rccl} \gamma \times \gamma' : & \mathsf{LDB}(\mathbf{D}) & \to & \mathsf{LDB}(\mathbf{V}) \times \mathsf{LDB}(\mathbf{V}') \\ & M & \mapsto & \langle \gamma(M), \gamma'(M) \rangle \end{array}$$

  is injective.

Observation: [Bancilhon & Spyratos 1981] If $\Gamma' = (\mathbf{V}', \gamma')$ is a complement of $\Gamma = (\mathbf{V}, \gamma)$, then for any update request (M_1, N_2) from Γ to $\mathbf{D}$, there is at most one realization which keeps the state of Γ' constant.

Proof: This realization must be $(M_1, (\gamma \times \gamma')^{-1}((N_1, \gamma_2(M_1))))$. $\square$

Familiar example: $\mathbf{E}_0 = (R[ABC], \{B \to C\})$.
View to be updated: $\Pi_{AB}^{\mathbf{E}_0} = (\mathbf{E}_0^{AB}, \pi_{AB}^{\mathbf{E}_0})$.
Natural complement: $\Pi_{BC}^{\mathbf{E}_0} = (\mathbf{E}_0^{BC}, \pi_{BC}^{\mathbf{E}_0})$.



View Complements and the Constant-Complement Approach

- The view $\Gamma' = (\mathbf{V}', \gamma')$ is a complement of $\Gamma = (\mathbf{V}, \gamma)$ if the *decomposition morphism*

$$\begin{array}{rrcl} \gamma \times \gamma' : & \mathrm{LDB}(\mathbf{D}) & \to & \mathrm{LDB}(\mathbf{V}) \times \mathrm{LDB}(\mathbf{V}') \\ & M & \mapsto & \langle \gamma(M), \gamma'(M) \rangle \end{array}$$

  is injective.

Observation: [Bancilhon & Spyratos 1981] If $\Gamma' = (\mathbf{V}', \gamma')$ is a complement of $\Gamma = (\mathbf{V}, \gamma)$, then for any update request (M_1, N_2) from Γ to $\mathbf{D}$, there is at most one realization which keeps the state of Γ' constant.

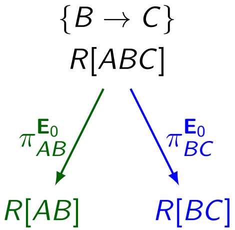
Proof: This realization must be $(M_1, (\gamma \times \gamma')^{-1}((N_1, \gamma_2(M_1))))$. $\square$

Familiar example: $\mathbf{E}_0 = (R[ABC], \{B \to C\})$.

View to be updated: $\Pi^{\mathbf{E}_0}_{AB} = (\mathbf{E}_0^{AB}, \pi^{\mathbf{E}_0}_{AB})$.

Natural complement: $\Pi^{\mathbf{E}_0}_{BC} = (\mathbf{E}_0^{BC}, \pi^{\mathbf{E}_0}_{BC})$.

- The updates to $\Pi^{\mathbf{E}_0}_{AB}$ with constant complement $\Pi^{\mathbf{E}_0}_{BC}$ are precisely those which keep $\Pi^{\mathbf{E}_0}_{B}$ fixed.



Three Uniqueness Issues for Constant Complement

Locality: The constant-complement strategy is intuitively appealing because it formalizes the notion of *locality* — the part of the main schema not included in the view to be updated (the complement) is held constant.

Three Uniqueness Issues for Constant Complement

Locality: The constant-complement strategy is intuitively appealing because it formalizes the notion of *locality* — the part of the main schema not included in the view to be updated (the complement) is held constant.

- However, there are at least three invariance issues surrounding the constant-complement approach to the reflection of view update.

Three Uniqueness Issues for Constant Complement

Locality: The constant-complement strategy is intuitively appealing because it formalizes the notion of *locality* — the part of the main schema not included in the view to be updated (the complement) is held constant.

- However, there are at least three invariance issues surrounding the constant-complement approach to the reflection of view update.

Context: Main schema $\mathbf{D}$, view $\Gamma = (\mathbf{V}, \gamma)$, complement $\Gamma' = (\mathbf{V}', \gamma')$, with:

Three Uniqueness Issues for Constant Complement

Locality: The constant-complement strategy is intuitively appealing because it formalizes the notion of *locality* — the part of the main schema not included in the view to be updated (the complement) is held constant.

- However, there are at least three invariance issues surrounding the constant-complement approach to the reflection of view update.

Context: Main schema $\mathbf{D}$, view $\Gamma = (\mathbf{V}, \gamma)$, complement $\Gamma' = (\mathbf{V}', \gamma')$, with:

- update request $u = (M_1, N_2)$ from Γ to $\mathbf{D}$.

Three Uniqueness Issues for Constant Complement

Locality: The constant-complement strategy is intuitively appealing because it formalizes the notion of *locality* — the part of the main schema not included in the view to be updated (the complement) is held constant.

- However, there are at least three invariance issues surrounding the constant-complement approach to the reflection of view update.

Context: Main schema $\mathbf{D}$, view $\Gamma = (\mathbf{V}, \gamma)$, complement $\Gamma' = (\mathbf{V}', \gamma')$, with:

- update request $u = (M_1, N_2)$ from Γ to $\mathbf{D}$.

State invariance: If u is realizable with constant complement Γ', then every $u' = (M_1', N_2)$ with $\gamma(M_1') = \gamma(M_1)$ is also so realizable.

Three Uniqueness Issues for Constant Complement

Locality: The constant-complement strategy is intuitively appealing because it formalizes the notion of *locality* — the part of the main schema not included in the view to be updated (the complement) is held constant.

- However, there are at least three invariance issues surrounding the constant-complement approach to the reflection of view update.

Context: Main schema $\mathbf{D}$, view $\Gamma = (\mathbf{V}, \gamma)$, complement $\Gamma' = (\mathbf{V}', \gamma')$, with:

- update request $u = (M_1, N_2)$ from Γ to $\mathbf{D}$.

State invariance: If u is realizable with constant complement Γ', then every $u' = (M_1', N_2)$ with $\gamma(M_1') = \gamma(M_1)$ is also so realizable.

Reflection invariance: If u is also realizable with respect to constant complement $\Gamma'' = (\mathbf{V}'', \gamma'')$, then the two realizations the same.

Three Uniqueness Issues for Constant Complement

Locality: The constant-complement strategy is intuitively appealing because it formalizes the notion of *locality* — the part of the main schema not included in the view to be updated (the complement) is held constant.

- However, there are at least three invariance issues surrounding the constant-complement approach to the reflection of view update.

Context: Main schema $\mathbf{D}$, view $\Gamma = (\mathbf{V}, \gamma)$, complement $\Gamma' = (\mathbf{V}', \gamma')$, with:

- update request $u = (M_1, N_2)$ from Γ to $\mathbf{D}$.

State invariance: If u is realizable with constant complement Γ', then every $u' = (M_1', N_2)$ with $\gamma(M_1') = \gamma(M_1)$ is also so realizable.

Reflection invariance: If u is also realizable with respect to constant complement $\Gamma'' = (\mathbf{V}'', \gamma'')$, then the two realizations the same.

Update-set invariance: If $\Gamma'' = (\mathbf{V}'', \gamma'')$ is a second complement, then Γ' and Γ'' support the same constant-complement updates.

Three Uniqueness Issues for Constant Complement

Locality: The constant-complement strategy is intuitively appealing because it formalizes the notion of *locality* — the part of the main schema not included in the view to be updated (the complement) is held constant.

- However, there are at least three invariance issues surrounding the constant-complement approach to the reflection of view update.

Context: Main schema $\mathbf{D}$, view $\Gamma = (\mathbf{V}, \gamma)$, complement $\Gamma' = (\mathbf{V}', \gamma')$, with:

  - update request $u = (M_1, N_2)$ from Γ to $\mathbf{D}$.

State invariance: If u is realizable with constant complement Γ', then every $u' = (M_1', N_2)$ with $\gamma(M_1') = \gamma(M_1)$ is also so realizable.

Reflection invariance: If u is also realizable with respect to constant complement $\Gamma'' = (\mathbf{V}'', \gamma'')$, then the two realizations the same.

Update-set invariance: If $\Gamma'' = (\mathbf{V}'', \gamma'')$ is a second complement, then Γ' and Γ'' support the same constant-complement updates.

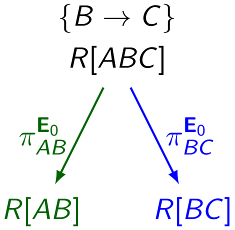
Observation: These three conditions are in general independent of one another.

Recall: $\mathbf{E}_0 = (R[ABC], \{B \to C\})$.

View to be updated: $\Pi_{AB}^{\mathbf{E}_0} = (\mathbf{E}_0^{AB}, \pi_{AB}^{\mathbf{E}_0})$.

Natural complement: $\Pi_{BC}^{\mathbf{E}_0} = (\mathbf{E}_0^{BC}, \pi_{BC}^{\mathbf{E}_0})$.

- The updates to $\Pi_{AB}^{\mathbf{E}_0}$ with constant complement $\Pi_{BC}^{\mathbf{E}_0}$ are precisely those which keep $\Pi_{B}^{\mathbf{E}_0}$ fixed.



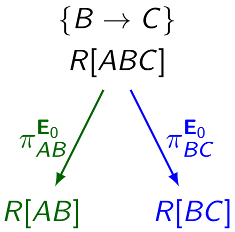
Dependency Preservation $\Rightarrow$ State Invariance

Recall: $\mathbf{E}_0 = (R[ABC], \{B \rightarrow C\})$.

View to be updated: $\Pi^{\mathbf{E}_0}_{AB} = (\mathbf{E}^{AB}_0, \pi^{\mathbf{E}_0}_{AB})$.

Natural complement: $\Pi^{\mathbf{E}_0}_{BC} = (\mathbf{E}^{BC}_0, \pi^{\mathbf{E}_0}_{BC})$.

- The updates to $\Pi^{\mathbf{E}_0}_{AB}$ with constant complement $\Pi^{\mathbf{E}_0}_{BC}$ are precisely those which keep $\Pi^{\mathbf{E}_0}_{B}$ fixed.
- This situation exhibits *state invariance*.



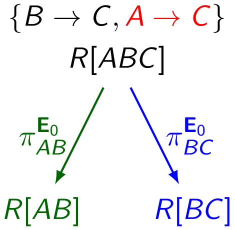
Dependency Preservation $\Rightarrow$ State Invariance

Recall: $\mathbf{E}_0 = (R[ABC], \{B \to C\})$.

View to be updated: $\Pi_{AB}^{\mathbf{E}_0} = (\mathbf{E}_0^{AB}, \pi_{AB}^{\mathbf{E}_0})$.

Natural complement: $\Pi_{BC}^{\mathbf{E}_0} = (\mathbf{E}_0^{BC}, \pi_{BC}^{\mathbf{E}_0})$.

- The updates to $\Pi_{AB}^{\mathbf{E}_0}$ with constant complement $\Pi_{BC}^{\mathbf{E}_0}$ are precisely those which keep $\Pi_{B}^{\mathbf{E}_0}$ fixed.



- This situation exhibits *state invariance*.
- If the FD $A \to C$ is added, this property no longer holds.

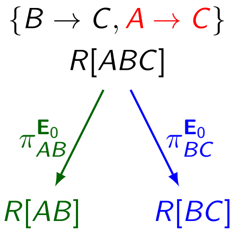
Dependency Preservation $\Rightarrow$ State Invariance

Recall: $\mathbf{E}_0 = (R[ABC], \{B \to C\})$.

View to be updated: $\Pi^{\mathbf{E}_0}_{AB} = (\mathbf{E}^{AB}_0, \pi^{\mathbf{E}_0}_{AB})$.

Natural complement: $\Pi^{\mathbf{E}_0}_{BC} = (\mathbf{E}^{BC}_0, \pi^{\mathbf{E}_0}_{BC})$.

- The updates to $\Pi^{\mathbf{E}_0}_{AB}$ with constant complement $\Pi^{\mathbf{E}_0}_{BC}$ are precisely those which keep $\Pi^{\mathbf{E}_0}_{B}$ fixed.



- This situation exhibits *state invariance*.
- If the FD $A \to C$ is added, this property no longer holds.

Example:

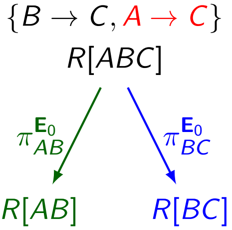
$$M_1 = \{R(\mathrm{a}_1, \mathrm{b}_1, \mathrm{c}_1), R(\mathrm{a}_2, \mathrm{b}_2, \mathrm{c}_1)\}$$
$$M_1' = \{R(\mathrm{a}_1, \mathrm{b}_1, \mathrm{c}_1), R(\mathrm{a}_2, \mathrm{b}_2, \mathrm{c}_2)\}$$

Recall: $\mathbf{E}_0 = (R[ABC], \{B \to C\})$.

View to be updated: $\Pi^{\mathbf{E}_0}_{AB} = (\mathbf{E}^{AB}_0, \pi^{\mathbf{E}_0}_{AB})$.

Natural complement: $\Pi^{\mathbf{E}_0}_{BC} = (\mathbf{E}^{BC}_0, \pi^{\mathbf{E}_0}_{BC})$.

- The updates to $\Pi^{\mathbf{E}_0}_{AB}$ with constant complement $\Pi^{\mathbf{E}_0}_{BC}$ are precisely those which keep $\Pi^{\mathbf{E}_0}_{B}$ fixed.

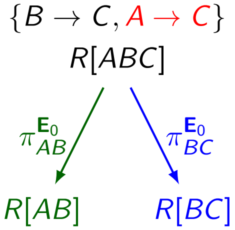


- This situation exhibits *state invariance*.
- If the FD $A \to C$ is added, this property no longer holds.

Example:
$$M_1 = \{R(\mathrm{a}_1, \mathrm{b}_1, \mathrm{c}_1), R(\mathrm{a}_2, \mathrm{b}_2, \mathrm{c}_1)\}$$
$$M_1' = \{R(\mathrm{a}_1, \mathrm{b}_1, \mathrm{c}_1), R(\mathrm{a}_2, \mathrm{b}_2, \mathrm{c}_2)\}$$

- The view update $(\{R(\mathrm{a}_1, \mathrm{b}_1), R(\mathrm{a}_2, \mathrm{b}_2)\}, \{R(\mathrm{a}_1, \mathrm{b}_1), R(\mathrm{a}_1, \mathrm{b}_2)\})$ is realizable if the state of $\mathbf{E}_0$ is M_1 but not if it is M_1'.

Dependency Preservation $\Rightarrow$ State Invariance



Recall: $\mathbf{E}_0 = (R[ABC], \{B \rightarrow C\})$.
View to be updated: $\Pi^{\mathbf{E}_0}_{AB} = (\mathbf{E}^{AB}_0, \pi^{\mathbf{E}_0}_{AB})$.
Natural complement: $\Pi^{\mathbf{E}_0}_{BC} = (\mathbf{E}^{BC}_0, \pi^{\mathbf{E}_0}_{BC})$.

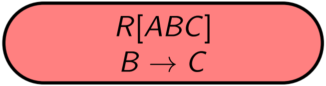
- The updates to $\Pi^{\mathbf{E}_0}_{AB}$ with constant complement $\Pi^{\mathbf{E}_0}_{BC}$ are precisely those which keep $\Pi^{\mathbf{E}_0}_{B}$ fixed.
- This situation exhibits *state invariance*.
- If the FD $A \rightarrow C$ is added, this property no longer holds.

Example:
$$M_1 = \{R(\mathrm{a}_1, \mathrm{b}_1, \mathrm{c}_1), R(\mathrm{a}_2, \mathrm{b}_2, \mathrm{c}_1)\}$$
$$M'_1 = \{R(\mathrm{a}_1, \mathrm{b}_1, \mathrm{c}_1), R(\mathrm{a}_2, \mathrm{b}_2, \mathrm{c}_2)\}$$

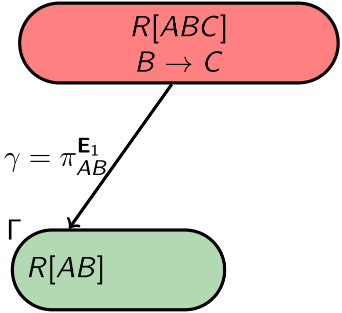
- The view update $(\{R(\mathrm{a}_1, \mathrm{b}_1), R(\mathrm{a}_2, \mathrm{b}_2)\}, \{R(\mathrm{a}_1, \mathrm{b}_1), R(\mathrm{a}_1, \mathrm{b}_2)\})$ is realizable if the state of $\mathbf{E}_0$ is M_1 but not if it is M'_1.
- With the addition of $A \rightarrow C$, a cover of the dependencies no longer embeds in the views, so these dependencies cannot be checked on a view-by-view basis.

Meets — the Generalization of Dependency Preservation

- A visualization of the previous example:

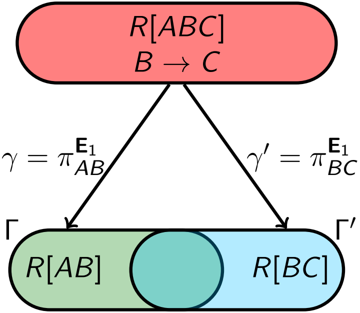


- A visualization of the previous example:
- The goal is to update the view $\Gamma = \Pi^{\mathbf{E}_1}_{AB}$...



Meets — the Generalization of Dependency Preservation

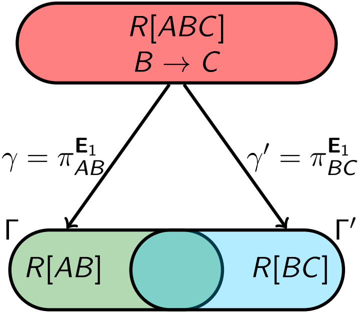
- A visualization of the previous example:
- The goal is to update the view $\Gamma = \Pi^{\mathbf{E}_1}_{AB}$...
- while holding the view $\Gamma' = \Pi^{\mathbf{E}_1}_{BC}$ constant.



Meets — the Generalization of Dependency Preservation

- A visualization of the previous example:
- The goal is to update the view $\Gamma = \Pi^{\mathbf{E}_1}_{AB}$...
- while holding the view $\Gamma' = \Pi^{\mathbf{E}_1}_{BC}$ constant.

Question: When does the visualization describe reality?

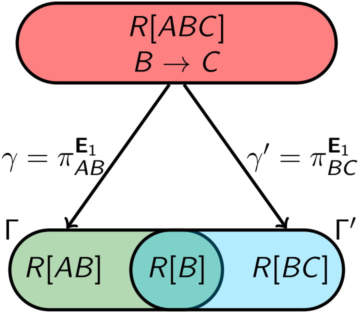


Meets — the Generalization of Dependency Preservation

- A visualization of the previous example:
- The goal is to update the view $\Gamma = \Pi^{\mathbf{E}_1}_{AB}$...
- while holding the view $\Gamma' = \Pi^{\mathbf{E}_1}_{BC}$ constant.

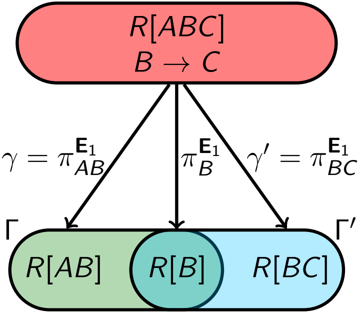
Question: When does the visualization describe reality?

  - When does it suffice to keep the overlap area $R[B]$ constant?



Meets — the Generalization of Dependency Preservation

- A visualization of the previous example:
- The goal is to update the view $\Gamma = \Pi^{\mathbf{E}_1}_{AB}$...
- while holding the view $\Gamma' = \Pi^{\mathbf{E}_1}_{BC}$ constant.



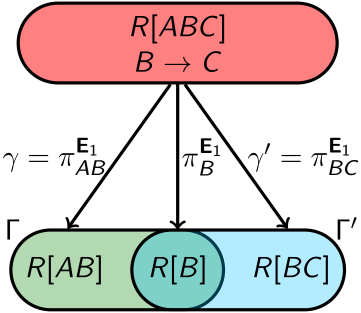
Question: When does the visualization describe reality?

- When does it suffice to keep the overlap area $R[B]$ constant?

Answer: The overlap area must define a view Γ'', (the *meet* of Γ and Γ') with: $\text{Congr}(\Gamma) \subseteq \text{Congr}(\Gamma'')$ and $\text{Congr}(\Gamma') \subseteq \text{Congr}(\Gamma'')$

Meets — the Generalization of Dependency Preservation

- A visualization of the previous example:
- The goal is to update the view $\Gamma = \Pi^{\mathbf{E}_1}_{AB}$...
- while holding the view $\Gamma' = \Pi^{\mathbf{E}_1}_{BC}$ constant.



Question: When does the visualization describe reality?

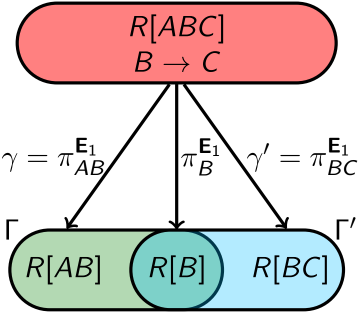
- When does it suffice to keep the overlap area $R[B]$ constant?

Answer: The overlap area must define a view Γ'', (the *meet* of Γ and Γ') with: $\mathrm{Congr}(\Gamma) \subseteq \mathrm{Congr}(\Gamma'')$ and $\mathrm{Congr}(\Gamma') \subseteq \mathrm{Congr}(\Gamma'')$

Solution: This happens precisely when the congruences commute: $\mathrm{Congr}(\Gamma) \circ \mathrm{Congr}(\Gamma') = \mathrm{Congr}(\Gamma') \circ \mathrm{Congr}(\Gamma)$

Meets — the Generalization of Dependency Preservation

- A visualization of the previous example:
- The goal is to update the view $\Gamma = \Pi^{\mathbf{E}_1}_{AB}$...
- while holding the view $\Gamma' = \Pi^{\mathbf{E}_1}_{BC}$ constant.



Question: When does the visualization describe reality?

- When does it suffice to keep the overlap area $R[B]$ constant?

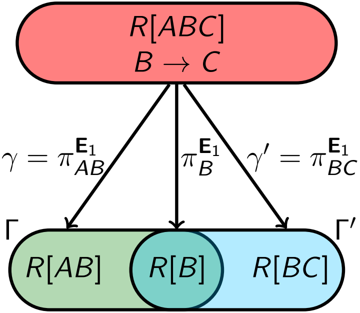
Answer: The overlap area must define a view Γ'', (the *meet* of Γ and Γ') with: $\mathrm{Congr}(\Gamma) \subseteq \mathrm{Congr}(\Gamma'')$ and $\mathrm{Congr}(\Gamma') \subseteq \mathrm{Congr}(\Gamma'')$

Solution: This happens precisely when the congruences commute: $\mathrm{Congr}(\Gamma) \circ \mathrm{Congr}(\Gamma') = \mathrm{Congr}(\Gamma') \circ \mathrm{Congr}(\Gamma)$

Theorem: Commuting congruences identifies precisely the conditions under which state invariance holds. □

Meets — the Generalization of Dependency Preservation

- A visualization of the previous example:
- The goal is to update the view $\Gamma = \Pi^{\mathbf{E}_1}_{AB}$...
- while holding the view $\Gamma' = \Pi^{\mathbf{E}_1}_{BC}$ constant.



Question: When does the visualization describe reality?

- When does it suffice to keep the overlap area $R[B]$ constant?

Answer: The overlap area must define a view Γ'', (the *meet* of Γ and Γ') with: $\mathrm{Congr}(\Gamma) \subseteq \mathrm{Congr}(\Gamma'')$ and $\mathrm{Congr}(\Gamma') \subseteq \mathrm{Congr}(\Gamma'')$

Solution: This happens precisely when the congruences commute: $\mathrm{Congr}(\Gamma) \circ \mathrm{Congr}(\Gamma') = \mathrm{Congr}(\Gamma') \circ \mathrm{Congr}(\Gamma)$

Theorem: Commuting congruences identifies precisely the conditions under which state invariance holds. $\square$

- A complement with commuting congruences is called a *meet complement*.

Constraints and Meet Complements

- In general, the constraints on a view may be very complex, if if the constraints on the main schema are simple (*e.g.*, FDs) and the view is simple (*e.g.*, a projection).

Constraints and Meet Complements

- In general, the constraints on a view may be very complex, if if the constraints on the main schema are simple (*e.g.*, FDs) and the view is simple (*e.g.*, a projection).
- In the case of update via meet complement, the constraint which actually need be checked are those which embed from the main schema.

Constraints and Meet Complements

- In general, the constraints on a view may be very complex, if if the constraints on the main schema are simple (*e.g.*, FDs) and the view is simple (*e.g.*, a projection).
- In the case of update via meet complement, the constraint which actually need be checked are those which embed from the main schema.

Example: $\mathbf{E}_1 = (R[ABCDE], \{A \to D, B \to D, CD \to A, A \to E\})$.

Constraints and Meet Complements

- In general, the constraints on a view may be very complex, if if the constraints on the main schema are simple (*e.g.*, FDs) and the view is simple (*e.g.*, a projection).
- In the case of update via meet complement, the constraint which actually need be checked are those which embed from the main schema.

Example: $\mathbf{E}_1 = (R[ABCDE], \{A \to D, B \to D, CD \to A, A \to E\})$.

View to be updated: $\Pi^{\mathbf{E}_1}_{ABCE} = (\mathbf{E}_1^{ABCE}, \pi^{\mathbf{E}_1}_{ABCE})$.

Constraints and Meet Complements

- In general, the constraints on a view may be very complex, if if the constraints on the main schema are simple (*e.g.*, FDs) and the view is simple (*e.g.*, a projection).
- In the case of update via meet complement, the constraint which actually need be checked are those which embed from the main schema.

Example: $\mathbf{E}_1 = (R[ABCDE], \{A \to D, B \to D, CD \to A, A \to E\})$.

View to be updated: $\Pi^{\mathbf{E}_1}_{ABCE} = (\mathbf{E}_1^{ABCE}, \pi^{\mathbf{E}_1}_{ABCE})$.

  - Embedded constraints: $\{A \to E\}$.

- In general, the constraints on a view may be very complex, if if the constraints on the main schema are simple (*e.g.*, FDs) and the view is simple (*e.g.*, a projection).
- In the case of update via meet complement, the constraint which actually need be checked are those which embed from the main schema.

Example: $\mathbf{E}_1 = (R[ABCDE], \{A \to D, B \to D, CD \to A, A \to E\})$.

View to be updated: $\Pi^{\mathbf{E}_1}_{ABCE} = (\mathbf{E}_1^{ABCE}, \pi^{\mathbf{E}_1}_{ABCE})$.

- Embedded constraints: $\{A \to E\}$.
- The view itself does not admit a finite basis of constraints.

Constraints and Meet Complements

- In general, the constraints on a view may be very complex, if if the constraints on the main schema are simple (*e.g.*, FDs) and the view is simple (*e.g.*, a projection).
- In the case of update via meet complement, the constraint which actually need be checked are those which embed from the main schema.

Example: $\mathbf{E}_1 = (R[ABCDE], \{A \to D, B \to D, CD \to A, A \to E\})$.

View to be updated: $\Pi^{\mathbf{E}_1}_{ABCE} = (\mathbf{E}_1^{ABCE}, \pi^{\mathbf{E}_1}_{ABCE})$.

- Embedded constraints: $\{A \to E\}$.
- The view itself does not admit a finite basis of constraints.

Meet complement: $\Pi^{\mathbf{E}_1}_{ABCD} = (\mathbf{E}_1^{ABCD}, \pi^{\mathbf{E}_1}_{ABCD})$ with meet $\Pi^{\mathbf{E}_1}_{ABC}$.

Constraints and Meet Complements

- In general, the constraints on a view may be very complex, if if the constraints on the main schema are simple (*e.g.*, FDs) and the view is simple (*e.g.*, a projection).
- In the case of update via meet complement, the constraint which actually need be checked are those which embed from the main schema.

Example: $\mathbf{E}_1 = (R[ABCDE], \{A \to D, B \to D, CD \to A, A \to E\})$.

View to be updated: $\Pi^{\mathbf{E}_1}_{ABCE} = (\mathbf{E}^{ABCE}_1, \pi^{\mathbf{E}_1}_{ABCE})$.

- Embedded constraints: $\{A \to E\}$.
- The view itself does not admit a finite basis of constraints.

Meet complement: $\Pi^{\mathbf{E}_1}_{ABCD} = (\mathbf{E}^{ABCD}_1, \pi^{\mathbf{E}_1}_{ABCD})$ with meet $\Pi^{\mathbf{E}_1}_{ABC}$.

- Embedded constraints: $\{A \to D, B \to D, CD \to A\}$.

Constraints and Meet Complements

- In general, the constraints on a view may be very complex, if if the constraints on the main schema are simple (*e.g.*, FDs) and the view is simple (*e.g.*, a projection).
- In the case of update via meet complement, the constraint which actually need be checked are those which embed from the main schema.

Example: $\mathbf{E}_1 = (R[ABCDE], \{A \rightarrow D, B \rightarrow D, CD \rightarrow A, A \rightarrow E\})$.

View to be updated: $\Pi^{\mathbf{E}_1}_{ABCE} = (\mathbf{E}_1^{ABCE}, \pi^{\mathbf{E}_1}_{ABCE})$.

- Embedded constraints: $\{A \rightarrow E\}$.
- The view itself does not admit a finite basis of constraints.

Meet complement: $\Pi^{\mathbf{E}_1}_{ABCD} = (\mathbf{E}_1^{ABCD}, \pi^{\mathbf{E}_1}_{ABCD})$ with meet $\Pi^{\mathbf{E}_1}_{ABC}$.

- Embedded constraints: $\{A \rightarrow D, B \rightarrow D, CD \rightarrow A\}$.

- The updates on $\Pi^{\mathbf{E}_1}_{ABCE}$ with $\Pi^{\mathbf{E}_1}_{ABCD}$ constant = updates with $\Pi^{\mathbf{E}_1}_{ABC}$ constant which satisfy $A \rightarrow E$.

Constraints and Meet Complements

- In general, the constraints on a view may be very complex, if if the constraints on the main schema are simple (*e.g.*, FDs) and the view is simple (*e.g.*, a projection).
- In the case of update via meet complement, the constraint which actually need be checked are those which embed from the main schema.

Example: $\mathbf{E}_1 = (R[ABCDE], \{A \to D, B \to D, CD \to A, A \to E\})$.

View to be updated: $\Pi^{\mathbf{E}_1}_{ABCE} = (\mathbf{E}_1^{ABCE}, \pi^{\mathbf{E}_1}_{ABCE})$.

- Embedded constraints: $\{A \to E\}$.
- The view itself does not admit a finite basis of constraints.

Meet complement: $\Pi^{\mathbf{E}_1}_{ABCD} = (\mathbf{E}_1^{ABCD}, \pi^{\mathbf{E}_1}_{ABCD})$ with meet $\Pi^{\mathbf{E}_1}_{ABC}$.

- Embedded constraints: $\{A \to D, B \to D, CD \to A\}$.

- The updates on $\Pi^{\mathbf{E}_1}_{ABCE}$ with $\Pi^{\mathbf{E}_1}_{ABCD}$ constant = updates with $\Pi^{\mathbf{E}_1}_{ABC}$ constant which satisfy $A \to E$.
  - $\{A \to D, B \to D, CD \to A\}$ are satisfied by virtue of $\Pi^{\mathbf{E}_1}_{ABCD}$ being held constant.

A Simple Example of the Nonuniqueness of Complements

- Recall that *reflection invariance* requires that a constant-complement update be independent of the choice of complement.

A Simple Example of the Nonuniqueness of Complements

- Recall that *reflection invariance* requires that a constant-complement update be independent of the choice of complement.
- It is easy to show how such invariance may fail.

A Simple Example of the Nonuniqueness of Complements

- Recall that *reflection invariance* requires that a constant-complement update be independent of the choice of complement.
- It is easy to show how such invariance may fail.

Example: $\mathbf{E}_2$ has two relation symbols $R[A]$ and $S[A]$.

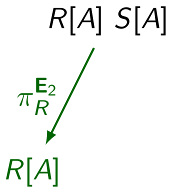
$R[A]$ $S[A]$

A Simple Example of the Nonuniqueness of Complements

- Recall that *reflection invariance* requires that a constant-complement update be independent of the choice of complement.
- It is easy to show how such invariance may fail.

Example: $\mathbf{E}_2$ has two relation symbols $R[A]$ and $S[A]$.

- The view to be updated is $\Pi_R^{\mathbf{E}_2}$.

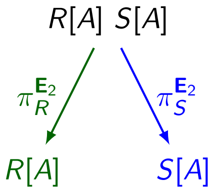


A Simple Example of the Nonuniqueness of Complements

- Recall that *reflection invariance* requires that a constant-complement update be independent of the choice of complement.
- It is easy to show how such invariance may fail.

Example: $\mathbf{E}_2$ has two relation symbols $R[A]$ and $S[A]$.

- The view to be updated is $\Pi_R^{\mathbf{E}_2}$.
- The obvious and natural complement is $\Pi_S^{\mathbf{E}_2}$.

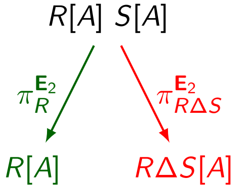


A Simple Example of the Nonuniqueness of Complements

- Recall that *reflection invariance* requires that a constant-complement update be independent of the choice of complement.
- It is easy to show how such invariance may fail.

Example: $\mathbf{E}_2$ has two relation symbols $R[A]$ and $S[A]$.

- The view to be updated is $\Pi^{\mathbf{E}_2}_{R}$.
- The obvious and natural complement is $\Pi^{\mathbf{E}_2}_{S}$.
- Another complement: $\Pi^{\mathbf{E}_2}_{R\Delta S} = (T[A], \pi^{\mathbf{E}_2}_{R\Delta S})$.

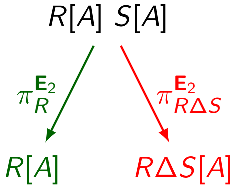


A Simple Example of the Nonuniqueness of Complements

- Recall that *reflection invariance* requires that a constant-complement update be independent of the choice of complement.
- It is easy to show how such invariance may fail.

Example: $\mathbf{E}_2$ has two relation symbols $R[A]$ and $S[A]$.

- The view to be updated is $\Pi_R^{\mathbf{E}_2}$.
- The obvious and natural complement is $\Pi_S^{\mathbf{E}_2}$.
- Another complement: $\Pi_{R\Delta S}^{\mathbf{E}_2} = (T[A], \pi_{R\Delta S}^{\mathbf{E}_2})$.
  - $T[x] \Leftrightarrow (R(x) \wedge (\neg S(x)) \vee ((\neg R(x)) \wedge S(x)))$.



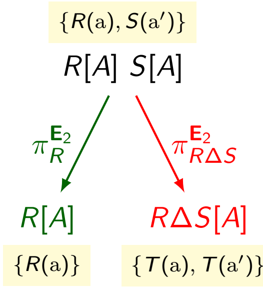
A Simple Example of the Nonuniqueness of Complements

- Recall that *reflection invariance* requires that a constant-complement update be independent of the choice of complement.
- It is easy to show how such invariance may fail.

Example: $\mathbf{E}_2$ has two relation symbols $R[A]$ and $S[A]$.

- The view to be updated is $\Pi_R^{\mathbf{E}_2}$.
- The obvious and natural complement is $\Pi_S^{\mathbf{E}_2}$.
- Another complement: $\Pi_{R\Delta S}^{\mathbf{E}_2} = (T[A], \pi_{R\Delta S}^{\mathbf{E}_2})$.
  - $T[x] \Leftrightarrow (R(x) \wedge (\neg S(x)) \vee ((\neg R(x)) \wedge S(x)))$.

Current state of main schema $\mathbf{E}_2$: $M_1 = \{R(\mathrm{a}), S(\mathrm{a}')\}$.

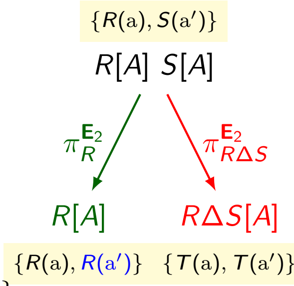


A Simple Example of the Nonuniqueness of Complements

- Recall that *reflection invariance* requires that a constant-complement update be independent of the choice of complement.
- It is easy to show how such invariance may fail.

Example: $\mathbf{E}_2$ has two relation symbols $R[A]$ and $S[A]$.

- The view to be updated is $\Pi_R^{\mathbf{E}_2}$.
- The obvious and natural complement is $\Pi_S^{\mathbf{E}_2}$.
- Another complement: $\Pi_{R\Delta S}^{\mathbf{E}_2} = (T[A], \pi_{R\Delta S}^{\mathbf{E}_2})$.
  - $T[x] \Leftrightarrow (R(x) \wedge (\neg S(x)) \vee ((\neg R(x)) \wedge S(x)))$.

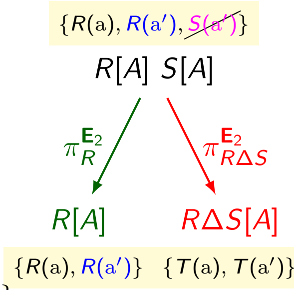


Current state of main schema $\mathbf{E}_2$: $M_1 = \{R(\mathrm{a}), S(\mathrm{a}')\}$.

View update: $u = (\{R(\mathrm{a})\}, \{R(\mathrm{a}), R(\mathrm{a}')\})$ on $\Pi_R^{\mathbf{E}_2}$. (Insert $R(\mathrm{a}')$).

A Simple Example of the Nonuniqueness of Complements

- Recall that *reflection invariance* requires that a constant-complement update be independent of the choice of complement.
- It is easy to show how such invariance may fail.



Example: $\mathbf{E}_2$ has two relation symbols $R[A]$ and $S[A]$.

- The view to be updated is $\Pi^{\mathbf{E}_2}_{R}$.
- The obvious and natural complement is $\Pi^{\mathbf{E}_2}_{S}$.
- Another complement: $\Pi^{\mathbf{E}_2}_{R\Delta S} = (T[A], \pi^{\mathbf{E}_2}_{R\Delta S})$.
  - $T[x] \Leftrightarrow (R(x) \wedge (\neg S(x)) \vee ((\neg R(x)) \wedge S(x)))$.

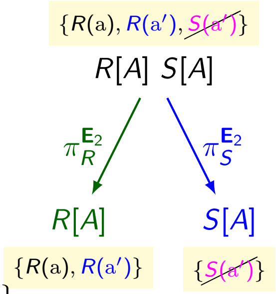
Current state of main schema $\mathbf{E}_2$: $M_1 = \{R(\mathrm{a}), S(\mathrm{a}')\}$.

View update: $u = (\{R(\mathrm{a})\}, \{R(\mathrm{a}), R(\mathrm{a}')\})$ on $\Pi^{\mathbf{E}_2}_{R}$. (Insert $R(\mathrm{a}')$).

New state of $\mathbf{E}_2$: $M_2 = \{R(\mathrm{a}), R(\mathrm{a}')\}$ with constant complement $\Pi^{\mathbf{E}_2}_{R\Delta S}$.

A Simple Example of the Nonuniqueness of Complements

- Recall that *reflection invariance* requires that a constant-complement update be independent of the choice of complement.
- It is easy to show how such invariance may fail.



Example: $\mathbf{E}_2$ has two relation symbols $R[A]$ and $S[A]$.

- The view to be updated is $\Pi_R^{\mathbf{E}_2}$.
- The obvious and natural complement is $\Pi_S^{\mathbf{E}_2}$.
- Another complement: $\Pi_{R\Delta S}^{\mathbf{E}_2} = (T[A], \pi_{R\Delta S}^{\mathbf{E}_2})$.
  - $T[x] \Leftrightarrow (R(x) \wedge (\neg S(x)) \vee ((\neg R(x)) \wedge S(x)))$.

Current state of main schema $\mathbf{E}_2$: $M_1 = \{R(\mathrm{a}), S(\mathrm{a}')\}$.

View update: $u = (\{R(\mathrm{a})\}, \{R(\mathrm{a}), R(\mathrm{a}')\})$ on $\Pi_R^{\mathbf{E}_2}$. (Insert $R(\mathrm{a}')$).

New state of $\mathbf{E}_2$: $M_2 = \{R(\mathrm{a}), R(\mathrm{a}')\}$ with constant complement $\Pi_{R\Delta S}^{\mathbf{E}_2}$.

- But note that *update-set invariance* is satisfied — both complements support all view updates.

Order-Based Views and Updates

Problem: Characterize “good” complements formally.

Problem: Characterize "good" complements formally.

Order: The states of database schemata often admit a natural order.

Order-Based Views and Updates

Problem: Characterize "good" complements formally.

Order: The states of database schemata often admit a natural order.

Example: In the relational model, relation-by-relation inclusion.

Order-Based Views and Updates

Problem: Characterize "good" complements formally.

Order: The states of database schemata often admit a natural order.

Example: In the relational model, relation-by-relation inclusion.

Notation: $\sqsubseteq_{\mathbf{D}}$ for this order on $\mathrm{LDB}(\mathbf{D})$.

Order-Based Views and Updates

Problem: Characterize "good" complements formally.

Order: The states of database schemata often admit a natural order.

Example: In the relational model, relation-by-relation inclusion.

Notation: $\sqsubseteq_{\mathbf{D}}$ for this order on $\mathrm{LDB}(\mathbf{D})$.

- The following have natural and obvious definitions:

Order-Based Views and Updates

Problem: Characterize "good" complements formally.

Order: The states of database schemata often admit a natural order.

Example: In the relational model, relation-by-relation inclusion.

Notation: $\sqsubseteq_{\mathbf{D}}$ for this order on $\mathsf{LDB}(\mathbf{D})$.

- The following have natural and obvious definitions:
  - *order-based schema*

Order-Based Views and Updates

Problem: Characterize "good" complements formally.

Order: The states of database schemata often admit a natural order.

Example: In the relational model, relation-by-relation inclusion.

Notation: $\sqsubseteq_{\mathbf{D}}$ for this order on $\mathrm{LDB}(\mathbf{D})$.

- The following have natural and obvious definitions:
  - *order-based schema*
    - In the relational model, morphisms which are defined without using negation (explicitly or implicitly) are order morphisms.

Order-Based Views and Updates

Problem: Characterize "good" complements formally.

Order: The states of database schemata often admit a natural order.

Example: In the relational model, relation-by-relation inclusion.

Notation: $\sqsubseteq_{\mathbf{D}}$ for this order on $\mathrm{LDB}(\mathbf{D})$.

- The following have natural and obvious definitions:
  - *order-based schema*
    - In the relational model, morphisms which are defined without using negation (explicitly or implicitly) are order morphisms.
  - *order-preserving database morphism* (or *order morphism*)

Order-Based Views and Updates

Problem: Characterize "good" complements formally.

Order: The states of database schemata often admit a natural order.

Example: In the relational model, relation-by-relation inclusion.

Notation: $\sqsubseteq_{\mathbf{D}}$ for this order on $\mathrm{LDB}(\mathbf{D})$.

- The following have natural and obvious definitions:
  - *order-based schema*
    - In the relational model, morphisms which are defined without using negation (explicitly or implicitly) are order morphisms.
  - *order-preserving database morphism* (or *order morphism*)
  - *order view*

Order-Based Views and Updates

Problem: Characterize "good" complements formally.

Order: The states of database schemata often admit a natural order.

Example: In the relational model, relation-by-relation inclusion.

Notation: $\sqsubseteq_{\mathbf{D}}$ for this order on $\mathsf{LDB}(\mathbf{D})$.

- The following have natural and obvious definitions:
  - *order-based schema*
    - In the relational model, morphisms which are defined without using negation (explicitly or implicitly) are order morphisms.
  - *order-preserving database morphism* (or *order morphism*)
  - *order view*

Insertion: (M_1, M_2) with $M_1 \sqsubseteq_{\mathbf{D}} M_2$.

Order-Based Views and Updates

Problem: Characterize "good" complements formally.

Order: The states of database schemata often admit a natural order.

Example: In the relational model, relation-by-relation inclusion.

Notation: $\sqsubseteq_{\mathbf{D}}$ for this order on $\mathsf{LDB}(\mathbf{D})$.

- The following have natural and obvious definitions:
  - *order-based schema*
    - In the relational model, morphisms which are defined without using negation (explicitly or implicitly) are order morphisms.
  - *order-preserving database morphism* (or *order morphism*)
  - *order view*

Insertion: (M_1, M_2) with $M_1 \sqsubseteq_{\mathbf{D}} M_2$.

Deletion: (M_1, M_2) with $M_2 \sqsubseteq_{\mathbf{D}} M_1$.

Order-Based Views and Updates

Problem: Characterize "good" complements formally.

Order: The states of database schemata often admit a natural order.

Example: In the relational model, relation-by-relation inclusion.

Notation: $\sqsubseteq_{\mathbf{D}}$ for this order on $\mathsf{LDB}(\mathbf{D})$.

- The following have natural and obvious definitions:
  - *order-based schema*
    - In the relational model, morphisms which are defined without using negation (explicitly or implicitly) are order morphisms.
  - *order-preserving database morphism* (or *order morphism*)
  - *order view*

Insertion: (M_1, M_2) with $M_1 \sqsubseteq_{\mathbf{D}} M_2$.

Deletion: (M_1, M_2) with $M_2 \sqsubseteq_{\mathbf{D}} M_1$.

Order-based update: An update which is representable as a composition of insertions and deletions.

The Uniqueness Theorem for Order-Based Updates

Order complement: $\Gamma' = (\mathbf{V}', \gamma')$ is an *order complement* of $\Gamma = (\mathbf{V}, \gamma)$ if

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

is an order isomorphism onto its image.

The Uniqueness Theorem for Order-Based Updates

Order complement: $\Gamma' = (\mathbf{V}', \gamma')$ is an *order complement* of $\Gamma = (\mathbf{V}, \gamma)$ if

$$\gamma \times \gamma' : \mathrm{LDB}(\mathbf{D}) \rightarrow \mathrm{LDB}(\mathbf{V}) \times \mathrm{LDB}(\mathbf{V}')$$

is an order isomorphism onto its image.

$R[A]$ $S[A]$

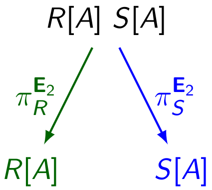
Example: In the context $\mathbf{E}_2 = (\{R[A], S[A]\}, \emptyset)$:

The Uniqueness Theorem for Order-Based Updates

Order complement: $\Gamma' = (\mathbf{V}', \gamma')$ is an *order complement* of $\Gamma = (\mathbf{V}, \gamma)$ if

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

is an order isomorphism onto its image.



Example: In the context $\mathbf{E}_2 = (\{R[A], S[A]\}, \emptyset)$:

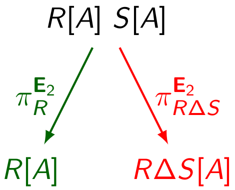
- $\Pi_S^{\mathbf{E}_2}$ is an order complement of $\Pi_R^{\mathbf{E}_2}$.

The Uniqueness Theorem for Order-Based Updates

Order complement: $\Gamma' = (\mathbf{V}', \gamma')$ is an *order complement* of $\Gamma = (\mathbf{V}, \gamma)$ if

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

is an order isomorphism onto its image.



Example: In the context $\mathbf{E}_2 = (\{R[A], S[A]\}, \emptyset)$:

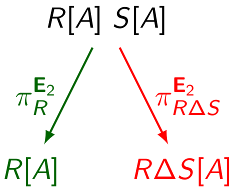
- $\Pi^{\mathbf{E}_2}_{S}$ is an order complement of $\Pi^{\mathbf{E}_2}_{R}$.
- $\Pi^{\mathbf{E}_2}_{R\Delta S}$ is not an order complement of $\Pi^{\mathbf{E}_2}_{R}$.

The Uniqueness Theorem for Order-Based Updates

Order complement: $\Gamma' = (\mathbf{V}', \gamma')$ is an *order complement* of $\Gamma = (\mathbf{V}, \gamma)$ if

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

is an order isomorphism onto its image.



Example: In the context $\mathbf{E}_2 = (\{R[A], S[A]\}, \emptyset)$:

- $\Pi_S^{\mathbf{E}_2}$ is an order complement of $\Pi_R^{\mathbf{E}_2}$.
- $\Pi_{R\Delta S}^{\mathbf{E}_2}$ is not an order complement of $\Pi_R^{\mathbf{E}_2}$.

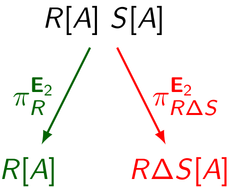
Theorem: Reflection invariance holds for order-based updates in an order-based context: the realization of an order-based view update is independent of the choice of order complement. □

The Uniqueness Theorem for Order-Based Updates

Order complement: $\Gamma' = (\mathbf{V}', \gamma')$ is an *order complement* of $\Gamma = (\mathbf{V}, \gamma)$ if

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

is an order isomorphism onto its image.



Example: In the context $\mathbf{E}_2 = (\{R[A], S[A]\}, \emptyset)$:

- $\Pi_S^{\mathbf{E}_2}$ is an order complement of $\Pi_R^{\mathbf{E}_2}$.
- $\Pi_{R\Delta S}^{\mathbf{E}_2}$ is not an order complement of $\Pi_R^{\mathbf{E}_2}$.

Theorem: Reflection invariance holds for order-based updates in an order-based context: the realization of an order-based view update is independent of the choice of order complement. $\square$

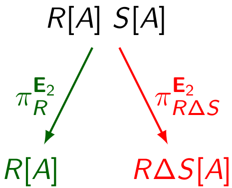
Tricks in the relational context to make additional updates order based:

The Uniqueness Theorem for Order-Based Updates

Order complement: $\Gamma' = (\mathbf{V}', \gamma')$ is an *order complement* of $\Gamma = (\mathbf{V}, \gamma)$ if

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

is an order isomorphism onto its image.



Example: In the context $\mathbf{E}_2 = (\{R[A], S[A]\}, \emptyset)$:

- $\Pi^{\mathbf{E}_2}_{S}$ is an order complement of $\Pi^{\mathbf{E}_2}_{R}$.
- $\Pi^{\mathbf{E}_2}_{R\Delta S}$ is not an order complement of $\Pi^{\mathbf{E}_2}_{R}$.

Theorem: Reflection invariance holds for order-based updates in an order-based context: the realization of an order-based view update is independent of the choice of order complement. □

Tricks in the relational context to make additional updates order based:

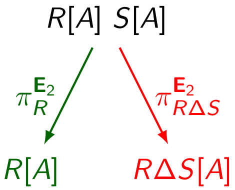
- Forget all constraints except the decomposition dependency.

The Uniqueness Theorem for Order-Based Updates

Order complement: $\Gamma' = (\mathbf{V}', \gamma')$ is an *order complement* of $\Gamma = (\mathbf{V}, \gamma)$ if

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

is an order isomorphism onto its image.



Example: In the context $\mathbf{E}_2 = (\{R[A], S[A]\}, \emptyset)$:

- $\Pi_S^{\mathbf{E}_2}$ is an order complement of $\Pi_R^{\mathbf{E}_2}$.
- $\Pi_{R\Delta S}^{\mathbf{E}_2}$ is not an order complement of $\Pi_R^{\mathbf{E}_2}$.

Theorem: Reflection invariance holds for order-based updates in an order-based context: the realization of an order-based view update is independent of the choice of order complement. □

Tricks in the relational context to make additional updates order based:

- Forget all constraints except the decomposition dependency.
- Extend the schemata using null values.

An Example of the Nonuniqueness of Order Complements

- The order-based context exhibits *reflection invariance*.

An Example of the Nonuniqueness of Order Complements

- The order-based context exhibits *reflection invariance*.
- A simple example shows that it need not exhibit *update-set invariance*.

An Example of the Nonuniqueness of Order Complements

- The order-based context exhibits *reflection invariance*.
- A simple example shows that it need not exhibit *update-set invariance*.
- Let $\mathbf{E}_3 = (R[ABCD], \{B \to D, C \to D\})$.

$$\{B \to D, C \to D\}$$
$$R[ABCD]$$

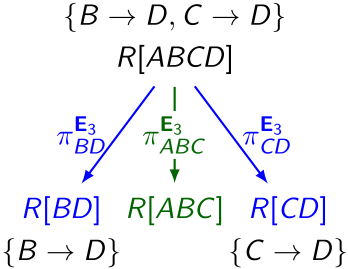
An Example of the Nonuniqueness of Order Complements

- The order-based context exhibits *reflection invariance*.
- A simple example shows that it need not exhibit *update-set invariance*.
- Let $\mathbf{E}_3 = (R[ABCD], \{B \to D, C \to D\})$.
- The view to be updated is $\Pi^{\mathbf{E}_3}_{ABC}$.

$$\{B \to D, C \to D\}$$
$$R[ABCD]$$
$$\downarrow \pi^{\mathbf{E}_3}_{ABC}$$
$$R[ABC]$$

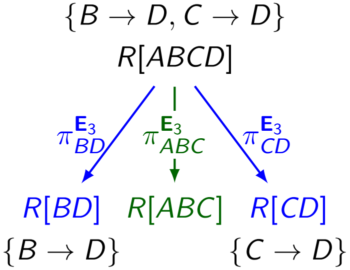
An Example of the Nonuniqueness of Order Complements

- The order-based context exhibits *reflection invariance*.
- A simple example shows that it need not exhibit *update-set invariance*.
- Let $\mathbf{E}_3 = (R[ABCD], \{B \to D, C \to D\})$.
- The view to be updated is $\Pi^{\mathbf{E}_3}_{ABC}$.
- Both $\Pi^{\mathbf{E}_3}_{BD}$ and $\Pi^{\mathbf{E}_3}_{CD}$ are complements.



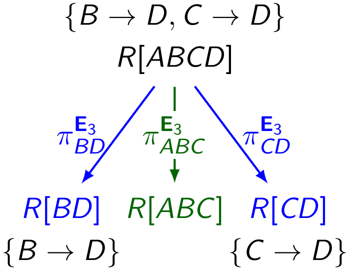
An Example of the Nonuniqueness of Order Complements

- The order-based context exhibits *reflection invariance*.
- A simple example shows that it need not exhibit *update-set invariance*.
- Let $\mathbf{E}_3 = (R[ABCD], \{B \to D, C \to D\})$.
- The view to be updated is $\Pi^{\mathbf{E}_3}_{ABC}$.
- Both $\Pi^{\mathbf{E}_3}_{BD}$ and $\Pi^{\mathbf{E}_3}_{CD}$ are complements.
- The schema $\mathbf{E}_3$ is completely symmetric in B and C, so (mathematically) there is no way to prefer one complement to the other.



An Example of the Nonuniqueness of Order Complements

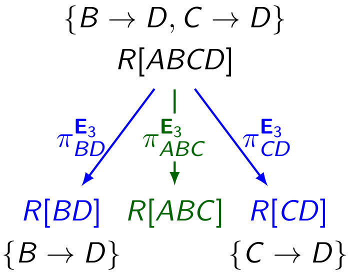
- The order-based context exhibits *reflection invariance*.
- A simple example shows that it need not exhibit *update-set invariance*.
- Let $\mathbf{E}_3 = (R[ABCD], \{B \to D, C \to D\})$.
- The view to be updated is $\Pi^{\mathbf{E}_3}_{ABC}$.
- Both $\Pi^{\mathbf{E}_3}_{BD}$ and $\Pi^{\mathbf{E}_3}_{CD}$ are complements.
- The schema $\mathbf{E}_3$ is completely symmetric in B and C, so (mathematically) there is no way to prefer one complement to the other.



- There is no smaller projection which is a complement.

An Example of the Nonuniqueness of Order Complements

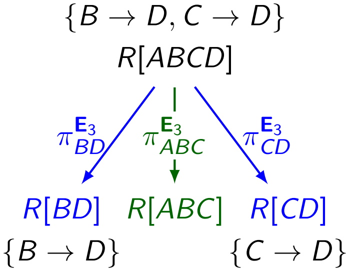
- The order-based context exhibits *reflection invariance*.
- A simple example shows that it need not exhibit *update-set invariance*.
- Let $\mathbf{E}_3 = (R[ABCD], \{B \to D, C \to D\})$.
- The view to be updated is $\Pi^{\mathbf{E}_3}_{ABC}$.
- Both $\Pi^{\mathbf{E}_3}_{BD}$ and $\Pi^{\mathbf{E}_3}_{CD}$ are complements.
- The schema $\mathbf{E}_3$ is completely symmetric in B and C, so (mathematically) there is no way to prefer one complement to the other.



- There is no smaller projection which is a complement.
- $\Pi^{\mathbf{E}_3}_{BD}$ constant $\Rightarrow$ $R[AC]$ may change, $R[B]$ may not change.

An Example of the Nonuniqueness of Order Complements

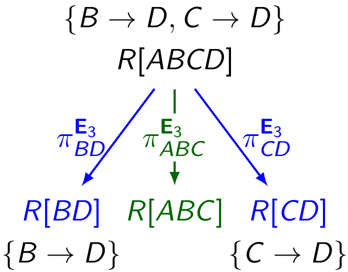
- The order-based context exhibits *reflection invariance*.
- A simple example shows that it need not exhibit *update-set invariance*.
- Let $\mathbf{E}_3 = (R[ABCD], \{B \to D, C \to D\})$.
- The view to be updated is $\Pi^{\mathbf{E}_3}_{ABC}$.
- Both $\Pi^{\mathbf{E}_3}_{BD}$ and $\Pi^{\mathbf{E}_3}_{CD}$ are complements.
- The schema $\mathbf{E}_3$ is completely symmetric in B and C, so (mathematically) there is no way to prefer one complement to the other.



- There is no smaller projection which is a complement.
- $\Pi^{\mathbf{E}_3}_{BD}$ constant $\Rightarrow$ $R[AC]$ may change, $R[B]$ may not change.
- $\Pi^{\mathbf{E}_3}_{CD}$ constant $\Rightarrow$ $R[AB]$ may change, $R[C]$ may not change.

An Example of the Nonuniqueness of Order Complements

- The order-based context exhibits *reflection invariance*.
- A simple example shows that it need not exhibit *update-set invariance*.
- Let $\mathbf{E}_3 = (R[ABCD], \{B \to D, C \to D\})$.
- The view to be updated is $\Pi^{\mathbf{E}_3}_{ABC}$.
- Both $\Pi^{\mathbf{E}_3}_{BD}$ and $\Pi^{\mathbf{E}_3}_{CD}$ are complements.
- The schema $\mathbf{E}_3$ is completely symmetric in B and C, so (mathematically) there is no way to prefer one complement to the other.



- There is no smaller projection which is a complement.
- $\Pi^{\mathbf{E}_3}_{BD}$ constant $\Rightarrow$ $R[AC]$ may change, $R[B]$ may not change.
- $\Pi^{\mathbf{E}_3}_{CD}$ constant $\Rightarrow$ $R[AB]$ may change, $R[C]$ may not change.

Reflection invariance: Updates which are possible with both complements must keep both constant $R[A]$ only may change, with the same reflections in each case.

Minimal and Optimal Complements

- The examples so far have worked implicitly with *minimal* complements.

Minimal and Optimal Complements

- The examples so far have worked implicitly with *minimal* complements.

Formal context: Schema $\mathbf{D}$; set $\mathcal{V}$ of views of $\mathbf{D}$; $\Gamma_1, \Gamma_2 \in \mathcal{V}$.

Minimal and Optimal Complements

- The examples so far have worked implicitly with *minimal* complements.

Formal context: Schema $\mathbf{D}$; set $\mathcal{V}$ of views of $\mathbf{D}$; $\Gamma_1, \Gamma_2 \in \mathcal{V}$.

- $\Gamma_1 \preceq_{\mathbf{D}} \Gamma_2$ iff $\mathrm{Congr}(\Gamma_2) \subseteq \mathrm{Congr}(\Gamma_1)$.

Minimal and Optimal Complements

- The examples so far have worked implicitly with *minimal* complements.

Formal context: Schema $\mathbf{D}$; set $\mathcal{V}$ of views of $\mathbf{D}$; $\Gamma_1, \Gamma_2 \in \mathcal{V}$.

- $\Gamma_1 \preceq_{\mathbf{D}} \Gamma_2$ iff $\text{Congr}(\Gamma_2) \subseteq \text{Congr}(\Gamma_1)$.
- $\Gamma_1 \prec_{\mathbf{D}} \Gamma_2$ iff $\text{Congr}(\Gamma_2) \subsetneq \text{Congr}(\Gamma_1)$ iff $\Gamma_1 \preceq_{\mathbf{D}} \Gamma_2$ and $\Gamma_2 \not\preceq_{\mathbf{D}} \Gamma_1$.

Minimal and Optimal Complements

- The examples so far have worked implicitly with *minimal* complements.

Formal context: Schema $\mathbf{D}$; set $\mathcal{V}$ of views of $\mathbf{D}$; $\Gamma_1, \Gamma_2 \in \mathcal{V}$.

- $\Gamma_1 \preceq_{\mathbf{D}} \Gamma_2$ iff $\mathrm{Congr}(\Gamma_2) \subseteq \mathrm{Congr}(\Gamma_1)$.
- $\Gamma_1 \prec_{\mathbf{D}} \Gamma_2$ iff $\mathrm{Congr}(\Gamma_2) \subsetneq \mathrm{Congr}(\Gamma_1)$ iff $\Gamma_1 \preceq_{\mathbf{D}} \Gamma_2$ and $\Gamma_2 \not\preceq_{\mathbf{D}} \Gamma_1$.
- $\Gamma_2 \in \mathcal{V}$ is a *minimal [meet] complement* of Γ_1 relative to $\mathcal{V}$ if for no other *[meet]* complement $\Gamma_3 \in \mathcal{V}$ it is the case that $\Gamma_3 \prec_{\mathbf{D}} \Gamma_2$.

Minimal and Optimal Complements

- The examples so far have worked implicitly with *minimal* complements.

Formal context: Schema $\mathbf{D}$; set $\mathcal{V}$ of views of $\mathbf{D}$; $\Gamma_1, \Gamma_2 \in \mathcal{V}$.

- $\Gamma_1 \preceq_{\mathbf{D}} \Gamma_2$ iff $\mathrm{Congr}(\Gamma_2) \subseteq \mathrm{Congr}(\Gamma_1)$.
- $\Gamma_1 \prec_{\mathbf{D}} \Gamma_2$ iff $\mathrm{Congr}(\Gamma_2) \subsetneq \mathrm{Congr}(\Gamma_1)$ iff $\Gamma_1 \preceq_{\mathbf{D}} \Gamma_2$ and $\Gamma_2 \not\preceq_{\mathbf{D}} \Gamma_1$.
- $\Gamma_2 \in \mathcal{V}$ is a *minimal [meet] complement* of Γ_1 relative to $\mathcal{V}$ if for no other *[meet]* complement $\Gamma_3 \in \mathcal{V}$ it is the case that $\Gamma_3 \prec_{\mathbf{D}} \Gamma_2$.

  Motivation: The smaller the complement, the greater the number of view updates supported.

Minimal and Optimal Complements

- The examples so far have worked implicitly with *minimal* complements.

Formal context: Schema $\mathbf{D}$; set $\mathcal{V}$ of views of $\mathbf{D}$; $\Gamma_1, \Gamma_2 \in \mathcal{V}$.

- $\Gamma_1 \preceq_{\mathbf{D}} \Gamma_2$ iff $\mathrm{Congr}(\Gamma_2) \subseteq \mathrm{Congr}(\Gamma_1)$.
- $\Gamma_1 \prec_{\mathbf{D}} \Gamma_2$ iff $\mathrm{Congr}(\Gamma_2) \subsetneq \mathrm{Congr}(\Gamma_1)$ iff $\Gamma_1 \preceq_{\mathbf{D}} \Gamma_2$ and $\Gamma_2 \not\preceq_{\mathbf{D}} \Gamma_1$.
- $\Gamma_2 \in \mathcal{V}$ is a *minimal [meet] complement* of Γ_1 relative to $\mathcal{V}$ if for no other *[meet]* complement $\Gamma_3 \in \mathcal{V}$ it is the case that $\Gamma_3 \prec_{\mathbf{D}} \Gamma_2$.

  Motivation: The smaller the complement, the greater the number of view updates supported.

  - Clearly, minimal is always desirable.

Minimal and Optimal Complements

- The examples so far have worked implicitly with *minimal* complements.

Formal context: Schema $\mathbf{D}$; set $\mathcal{V}$ of views of $\mathbf{D}$; $\Gamma_1, \Gamma_2 \in \mathcal{V}$.

- $\Gamma_1 \preceq_{\mathbf{D}} \Gamma_2$ iff $\text{Congr}(\Gamma_2) \subseteq \text{Congr}(\Gamma_1)$.
- $\Gamma_1 \prec_{\mathbf{D}} \Gamma_2$ iff $\text{Congr}(\Gamma_2) \subsetneq \text{Congr}(\Gamma_1)$ iff $\Gamma_1 \preceq_{\mathbf{D}} \Gamma_2$ and $\Gamma_2 \not\preceq_{\mathbf{D}} \Gamma_1$.
- $\Gamma_2 \in \mathcal{V}$ is a *minimal [meet] complement* of Γ_1 relative to $\mathcal{V}$ if for no other *[meet]* complement $\Gamma_3 \in \mathcal{V}$ it is the case that $\Gamma_3 \prec_{\mathbf{D}} \Gamma_2$.

  Motivation: The smaller the complement, the greater the number of view updates supported.

  - Clearly, minimal is always desirable.
  - However, minimal cannot guarantee *update-set invariance*, since distinct minimal complements give rise to distinct update sets.

Minimal and Optimal Complements

- The examples so far have worked implicitly with *minimal* complements.

Formal context: Schema $\mathbf{D}$; set $\mathcal{V}$ of views of $\mathbf{D}$; $\Gamma_1, \Gamma_2 \in \mathcal{V}$.

- $\Gamma_1 \preceq_{\mathbf{D}} \Gamma_2$ iff $\mathrm{Congr}(\Gamma_2) \subseteq \mathrm{Congr}(\Gamma_1)$.
- $\Gamma_1 \prec_{\mathbf{D}} \Gamma_2$ iff $\mathrm{Congr}(\Gamma_2) \subsetneq \mathrm{Congr}(\Gamma_1)$ iff $\Gamma_1 \preceq_{\mathbf{D}} \Gamma_2$ and $\Gamma_2 \not\preceq_{\mathbf{D}} \Gamma_1$.
- $\Gamma_2 \in \mathcal{V}$ is a *minimal [meet] complement* of Γ_1 relative to $\mathcal{V}$ if for no other *[meet]* complement $\Gamma_3 \in \mathcal{V}$ it is the case that $\Gamma_3 \prec_{\mathbf{D}} \Gamma_2$.

  Motivation: The smaller the complement, the greater the number of view updates supported.

  - Clearly, minimal is always desirable.
  - However, minimal cannot guarantee *update-set invariance*, since distinct minimal complements give rise to distinct update sets.

- $\Gamma_2 \in \mathcal{V}$ is an *optimal [meet] complement* of Γ_1 relative to $\mathcal{V}$ if for every other *[meet]* complement $\Gamma_3 \in \mathcal{V}$, it is the case that $\Gamma_2 \preceq_{\mathbf{D}} \Gamma_3$.

Minimal and Optimal Complements

- The examples so far have worked implicitly with *minimal* complements.

Formal context: Schema $\mathbf{D}$; set $\mathcal{V}$ of views of $\mathbf{D}$; $\Gamma_1, \Gamma_2 \in \mathcal{V}$.

- $\Gamma_1 \preceq_{\mathbf{D}} \Gamma_2$ iff $\mathrm{Congr}(\Gamma_2) \subseteq \mathrm{Congr}(\Gamma_1)$.
- $\Gamma_1 \prec_{\mathbf{D}} \Gamma_2$ iff $\mathrm{Congr}(\Gamma_2) \subsetneq \mathrm{Congr}(\Gamma_1)$ iff $\Gamma_1 \preceq_{\mathbf{D}} \Gamma_2$ and $\Gamma_2 \not\preceq_{\mathbf{D}} \Gamma_1$.
- $\Gamma_2 \in \mathcal{V}$ is a *minimal [meet] complement* of Γ_1 relative to $\mathcal{V}$ if for no other *[meet]* complement $\Gamma_3 \in \mathcal{V}$ it is the case that $\Gamma_3 \prec_{\mathbf{D}} \Gamma_2$.

  Motivation: The smaller the complement, the greater the number of view updates supported.

  - Clearly, minimal is always desirable.
  - However, minimal cannot guarantee *update-set invariance*, since distinct minimal complements give rise to distinct update sets.

- $\Gamma_2 \in \mathcal{V}$ is an *optimal [meet] complement* of Γ_1 relative to $\mathcal{V}$ if for every other *[meet]* complement $\Gamma_3 \in \mathcal{V}$, it is the case that $\Gamma_2 \preceq_{\mathbf{D}} \Gamma_3$.

  Movivation: It is precisely an optimal complement which guarantees *update-set independence*.

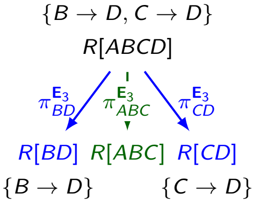
Examples of Minimal and Optimal Complements

Context: Consider again the running example $\mathbf{E}_3$.

Examples of Minimal and Optimal Complements

Context: Consider again the running example $\mathbf{E}_3$.

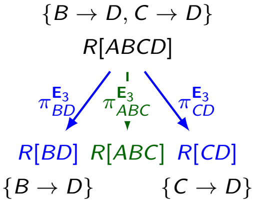
- Both $\Pi_D^{\mathbf{E}_3}$ and $\Pi_{CD}^{\mathbf{E}_3}$ are minimal complements of $\Pi_{ABC}^{\mathbf{E}_3}$ relative to the projections Π-Views$\langle \mathbf{E}_3 \rangle$.



Examples of Minimal and Optimal Complements

Context: Consider again the running example $\mathbf{E}_3$.

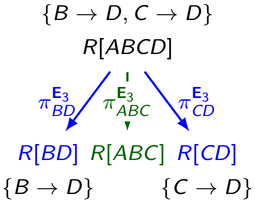
- Both $\Pi_{D}^{\mathbf{E}_3}$ and $\Pi_{CD}^{\mathbf{E}_3}$ are minimal complements of $\Pi_{ABC}^{\mathbf{E}_3}$ relative to the projections Π-Views$\langle \mathbf{E}_3 \rangle$.
- Thus, neither can be optimal.

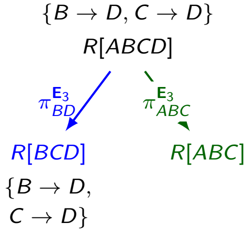


Examples of Minimal and Optimal Complements

Context: Consider again the running example $\mathbf{E}_3$.

- Both $\Pi_D^{\mathbf{E}_3}$ and $\Pi_{CD}^{\mathbf{E}_3}$ are minimal complements of $\Pi_{ABC}^{\mathbf{E}_3}$ relative to the projections Π-Views$\langle \mathbf{E}_3 \rangle$.
- Thus, neither can be optimal.
- $\Pi_{BCD}^{\mathbf{E}_3}$ is a complement which is not minimal relative to Π-Views$\langle \mathbf{E}_3 \rangle$.

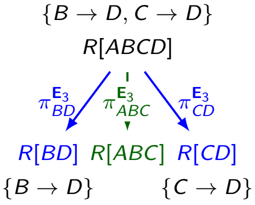


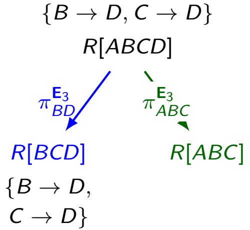


Examples of Minimal and Optimal Complements

Context: Consider again the running example $\mathbf{E}_3$.

- Both $\Pi_D^{\mathbf{E}_3}$ and $\Pi_{CD}^{\mathbf{E}_3}$ are minimal complements of $\Pi_{ABC}^{\mathbf{E}_3}$ relative to the projections Π-Views$\langle \mathbf{E}_3 \rangle$.
- Thus, neither can be optimal.
- $\Pi_{BCD}^{\mathbf{E}_3}$ is a complement which is not minimal relative to Π-Views$\langle \mathbf{E}_3 \rangle$.
- But $\Pi_{BCD}^{\mathbf{E}_3}$ is an optimal *meet* complement amongst projections.

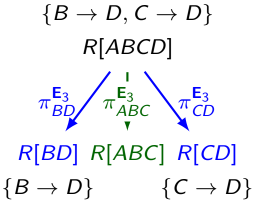


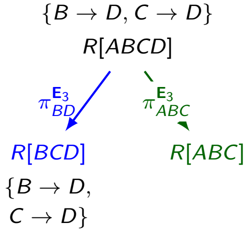


Examples of Minimal and Optimal Complements

Context: Consider again the running example $\mathbf{E}_3$.

- Both $\Pi_D^{\mathbf{E}_3}$ and $\Pi_{CD}^{\mathbf{E}_3}$ are minimal complements of $\Pi_{ABC}^{\mathbf{E}_3}$ relative to the projections $\Pi\text{-Views}\langle\mathbf{E}_3\rangle$.
- Thus, neither can be optimal.
- $\Pi_{BCD}^{\mathbf{E}_3}$ is a complement which is not minimal relative to $\Pi\text{-Views}\langle\mathbf{E}_3\rangle$.
- But $\Pi_{BCD}^{\mathbf{E}_3}$ is an optimal *meet* complement amongst projections.
- If *state invariance* is desired, $\Pi_{BCD}^{\mathbf{E}_3}$ is the best which can be achieved.

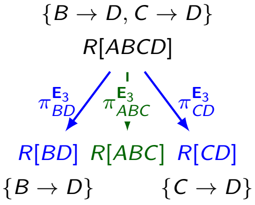


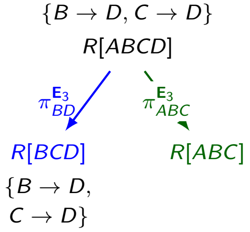


Examples of Minimal and Optimal Complements

Context: Consider again the running example $\mathbf{E}_3$.

- Both $\Pi_{D}^{\mathbf{E}_3}$ and $\Pi_{CD}^{\mathbf{E}_3}$ are minimal complements of $\Pi_{ABC}^{\mathbf{E}_3}$ relative to the projections $\Pi\text{-Views}\langle\mathbf{E}_3\rangle$.
- Thus, neither can be optimal.
- $\Pi_{BCD}^{\mathbf{E}_3}$ is a complement which is not minimal relative to $\Pi\text{-Views}\langle\mathbf{E}_3\rangle$.
- But $\Pi_{BCD}^{\mathbf{E}_3}$ is an optimal *meet* complement amongst projections.
- If *state invariance* is desired, $\Pi_{BCD}^{\mathbf{E}_3}$ is the best which can be achieved.
  - *Update-set independence comes as a bonus*, but for a smaller set of updates than supported by $\Pi_{BD}^{\mathbf{E}_3}$ or $\Pi_{CD}^{\mathbf{E}_3}$ as complements.

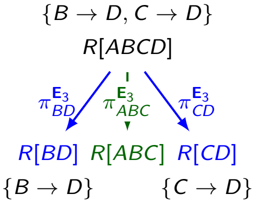


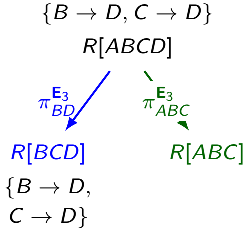


Examples of Minimal and Optimal Complements

Context: Consider again the running example $\mathbf{E}_3$.

- Both $\Pi_D^{\mathbf{E}_3}$ and $\Pi_{CD}^{\mathbf{E}_3}$ are minimal complements of $\Pi_{ABC}^{\mathbf{E}_3}$ relative to the projections Π-Views$\langle \mathbf{E}_3 \rangle$.
- Thus, neither can be optimal.
- $\Pi_{BCD}^{\mathbf{E}_3}$ is a complement which is not minimal relative to Π-Views$\langle \mathbf{E}_3 \rangle$.
- But $\Pi_{BCD}^{\mathbf{E}_3}$ is an optimal *meet* complement amongst projections.
- If *state invariance* is desired, $\Pi_{BCD}^{\mathbf{E}_3}$ is the best which can be achieved.
  - *Update-set independence comes as a bonus*, but for a smaller set of updates than supported by $\Pi_{BD}^{\mathbf{E}_3}$ or $\Pi_{CD}^{\mathbf{E}_3}$ as complements.
- Clearly, there are tradeoffs.





The Context of $\bigvee\Pi$-Views

- Consider again the running example.

$$\{B \to D, C \to D\}$$

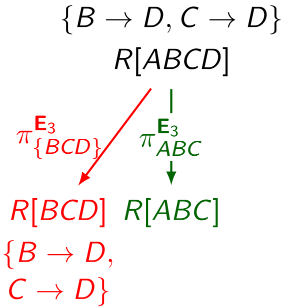
$$R[ABCD]$$

$$\downarrow \pi^{\mathbf{E}_3}_{ABC}$$

$$R[ABC]$$

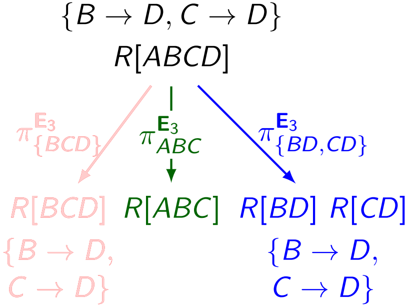
The Context of $\bigvee\Pi$-Views

- Consider again the running example.
- The view $\Pi^{\mathbf{E}_3}_{BCD}$ is the optimal meet complement of $\Pi^{\mathbf{E}_3}_{ABC}$ amongst all projections.



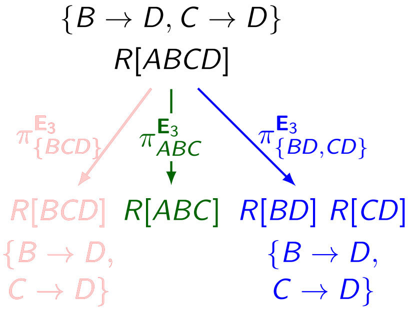
The Context of $\bigvee\Pi$-Views

- Consider again the running example.
- The view $\Pi^{\mathbf{E}_3}_{BCD}$ is the optimal meet complement of $\Pi^{\mathbf{E}_3}_{ABC}$ amongst all projections.
- However, consider the view $\Pi^{\mathbf{E}_3}_{\{BD,CD\}}$ which consists of two projections BD and CD.



The Context of $\bigvee\Pi$-Views

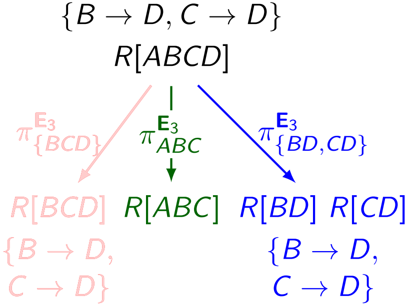
- Consider again the running example.
- The view $\Pi^{\mathbf{E}_3}_{BCD}$ is the optimal meet complement of $\Pi^{\mathbf{E}_3}_{ABC}$ amongst all projections.
- However, consider the view $\Pi^{\mathbf{E}_3}_{\{BD,CD\}}$ which consists of two projections BD and CD.



- It is a smaller meet complement: $\Pi^{\mathbf{E}_3}_{\{BD,CD\}} \prec_{\mathbf{E}_3} \Pi^{\mathbf{E}_3}_{BCD}$.

The Context of $\bigvee\Pi$-Views

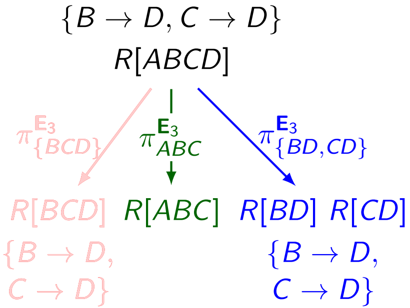
- Consider again the running example.
- The view $\Pi^{\mathbf{E}_3}_{BCD}$ is the optimal meet complement of $\Pi^{\mathbf{E}_3}_{ABC}$ amongst all projections.
- However, consider the view $\Pi^{\mathbf{E}_3}_{\{BD,CD\}}$ which consists of two projections BD and CD.



- It is a smaller meet complement: $\Pi^{\mathbf{E}_3}_{\{BD,CD\}} \prec_{\mathbf{E}_3} \Pi^{\mathbf{E}_3}_{BCD}$.
- The association of B-values and C-values is not preserved by this view.

The Context of $\bigvee\Pi$-Views

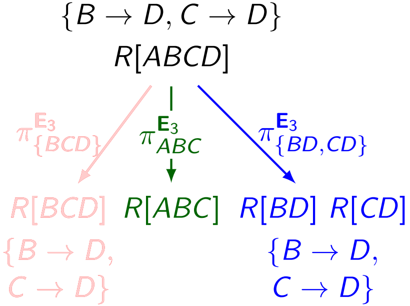
- Consider again the running example.
- The view $\Pi^{\mathbf{E}_3}_{BCD}$ is the optimal meet complement of $\Pi^{\mathbf{E}_3}_{ABC}$ amongst all projections.
- However, consider the view $\Pi^{\mathbf{E}_3}_{\{BD,CD\}}$ which consists of two projections BD and CD.



- It is a smaller meet complement: $\Pi^{\mathbf{E}_3}_{\{BD,CD\}} \prec_{\mathbf{E}_3} \Pi^{\mathbf{E}_3}_{BCD}$.
- The association of B-values and C-values is not preserved by this view.
- Such a view consisting of multiple projections is called a $\bigvee\Pi$-view.

The Context of $\bigvee\Pi$-Views

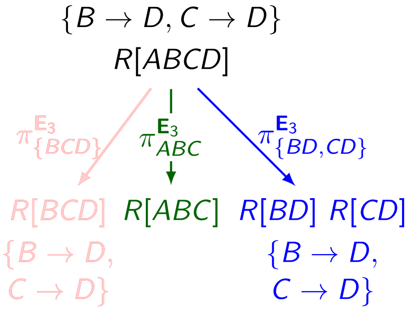
- Consider again the running example.
- The view $\Pi^{\mathbf{E}_3}_{BCD}$ is the optimal meet complement of $\Pi^{\mathbf{E}_3}_{ABC}$ amongst all projections.
- However, consider the view $\Pi^{\mathbf{E}_3}_{\{BD,CD\}}$ which consists of two projections BD and CD.



- It is a smaller meet complement: $\Pi^{\mathbf{E}_3}_{\{BD,CD\}} \prec_{\mathbf{E}_3} \Pi^{\mathbf{E}_3}_{BCD}$.
- The association of B-values and C-values is not preserved by this view.
- Such a view consisting of multiple projections is called a $\bigvee\Pi$-view.
- They can be used instead of single projections with little or no extra work.

The Context of $\bigvee\Pi$-Views

- Consider again the running example.
- The view $\Pi^{\mathbf{E}_3}_{BCD}$ is the optimal meet complement of $\Pi^{\mathbf{E}_3}_{ABC}$ amongst all projections.
- However, consider the view $\Pi^{\mathbf{E}_3}_{\{BD,CD\}}$ which consists of two projections BD and CD.



- It is a smaller meet complement: $\Pi^{\mathbf{E}_3}_{\{BD,CD\}} \prec_{\mathbf{E}_3} \Pi^{\mathbf{E}_3}_{BCD}$.
- The association of B-values and C-values is not preserved by this view.
- Such a view consisting of multiple projections is called a $\bigvee\Pi$-view.
- They can be used instead of single projections with little or no extra work.

Notation: $\bigvee\Pi$-Views$\langle \mathbf{D} \rangle$ denotes the set of all $\bigvee\Pi$ views of $\mathbf{D}$.

Nonuniqueness of Meet Complements in the FD context

- Context: A Universal-relational schema constrained by FDs.

Nonuniqueness of Meet Complements in the FD context

- Context: A Universal-relational schema constrained by FDs.
- A simple example of the nonexistence of optimal projective complements has been given:

Nonuniqueness of Meet Complements in the FD context

- Context: A Universal-relational schema constrained by FDs.
- A simple example of the nonexistence of optimal projective complements has been given:
  - $\mathbf{E}_3 = (R[ABCD], \{B \to D, C \to D\})$.

Nonuniqueness of Meet Complements in the FD context

- Context: A Universal-relational schema constrained by FDs.
- A simple example of the nonexistence of optimal projective complements has been given:
  - $\mathbf{E}_3 = (R[ABCD], \{B \to D, C \to D\})$.
  - $\Pi^{\mathbf{E}_2}_{ABC}$ has distinct minimal $\bigvee\Pi$-complements $\Pi^{\mathbf{E}_3}_{BD}$ and $\Pi^{\mathbf{E}_3}_{CD}$.

Nonuniqueness of Meet Complements in the FD context

- Context: A Universal-relational schema constrained by FDs.
- A simple example of the nonexistence of optimal projective complements has been given:
  - $\mathbf{E}_3 = (R[ABCD], \{B \to D, C \to D\})$.
  - $\Pi^{\mathbf{E}_2}_{ABC}$ has distinct minimal $\bigvee\Pi$-complements $\Pi^{\mathbf{E}_3}_{BD}$ and $\Pi^{\mathbf{E}_3}_{CD}$.
- However, it does have an optimal meet $\bigvee\Pi$-complement: $\Pi^{\mathbf{E}_3}_{\{BC,CD\}}$.

Nonuniqueness of Meet Complements in the FD context

- Context: A Universal-relational schema constrained by FDs.
- A simple example of the nonexistence of optimal projective complements has been given:
  - $\mathbf{E}_3 = (R[ABCD], \{B \to D, C \to D\})$.
  - $\Pi^{\mathbf{E}_2}_{ABC}$ has distinct minimal $\bigvee\Pi$-complements $\Pi^{\mathbf{E}_3}_{BD}$ and $\Pi^{\mathbf{E}_3}_{CD}$.
- However, it does have an optimal meet $\bigvee\Pi$-complement: $\Pi^{\mathbf{E}_3}_{\{BC,CD\}}$.

Question: Are there examples without optimal meet complements?

Nonuniqueness of Meet Complements in the FD context

- Context: A Universal-relational schema constrained by FDs.
- A simple example of the nonexistence of optimal projective complements has been given:
  - $\mathbf{E}_3 = (R[ABCD], \{B \to D, C \to D\})$.
  - $\Pi^{\mathbf{E}_2}_{ABC}$ has distinct minimal $\bigvee\Pi$-complements $\Pi^{\mathbf{E}_3}_{BD}$ and $\Pi^{\mathbf{E}_3}_{CD}$.
- However, it does have an optimal meet $\bigvee\Pi$-complement: $\Pi^{\mathbf{E}_3}_{\{BC,CD\}}$.

Question: Are there examples without optimal meet complements?

Yes: $\mathbf{E}_4 = (R[ABC], \{A \to BC, B \to AC\})$.

$$\{A \to BC, B \to AC\}$$
$$R[ABC]$$

Nonuniqueness of Meet Complements in the FD context

- Context: A Universal-relational schema constrained by FDs.
- A simple example of the nonexistence of optimal projective complements has been given:
  - $\mathbf{E}_3 = (R[ABCD], \{B \to D, C \to D\})$.
  - $\Pi^{\mathbf{E}_2}_{ABC}$ has distinct minimal $\bigvee\Pi$-complements $\Pi^{\mathbf{E}_3}_{BD}$ and $\Pi^{\mathbf{E}_3}_{CD}$.
- However, it does have an optimal meet $\bigvee\Pi$-complement: $\Pi^{\mathbf{E}_3}_{\{BC,CD\}}$.

Question: Are there examples without optimal meet complements?

Yes: $\mathbf{E}_4 = (R[ABC], \{A \to BC, B \to AC\})$.

- The two minimal complements $\Pi^{\mathbf{E}_4}_{AB}$ and $\Pi^{\mathbf{E}_4}_{BC}$ are related by an attribute equivalence $A \leftrightarrow B$ of keys.

$\{A \to BC, B \to AC\}$

$R[ABC]$

$\downarrow \pi^{\mathbf{E}_4}_{AB}$

$R[AB]$

$\{A \leftrightarrow B\}$

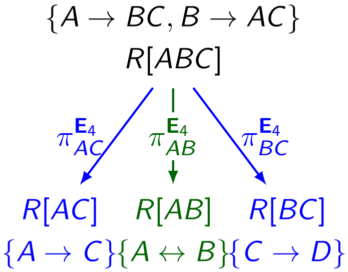
Nonuniqueness of Meet Complements in the FD context

- Context: A Universal-relational schema constrained by FDs.
- A simple example of the nonexistence of optimal projective complements has been given:
  - $\mathbf{E}_3 = (R[ABCD], \{B \to D, C \to D\})$.
  - $\Pi^{\mathbf{E}_2}_{ABC}$ has distinct minimal $\bigvee\Pi$-complements $\Pi^{\mathbf{E}_3}_{BD}$ and $\Pi^{\mathbf{E}_3}_{CD}$.
- However, it does have an optimal meet $\bigvee\Pi$-complement: $\Pi^{\mathbf{E}_3}_{\{BC,CD\}}$.

Question: Are there examples without optimal meet complements?

Yes: $\mathbf{E}_4 = (R[ABC], \{A \to BC, B \to AC\})$.

- The two minimal complements $\Pi^{\mathbf{E}_4}_{AB}$ and $\Pi^{\mathbf{E}_4}_{BC}$ are related by an attribute equivalence $A \leftrightarrow B$ of keys.
- This is the only way that such non-isomorphic minimal complements can occur.



Equivalence of Meet Complements in the $\bigvee\sqcap$-FD framework

Context:
- Universal relational schema $\mathbf{D} = (R[\mathbf{U}], \mathcal{F})$; $\mathcal{F}$ = FDs.
- $\sqcap^{\mathbf{D}}_{\{\mathbf{W}_2,\mathbf{W}_2,\ldots\mathbf{W}_m\}}$ a $\bigvee\sqcap$-view.

Equivalence of Meet Complements in the $\bigvee\Pi$-FD framework

Context:
- Universal relational schema $\mathbf{D} = (R[\mathbf{U}], \mathcal{F})$; $\mathcal{F}$ = FDs.
- $\Pi^{\mathbf{D}}_{\{\mathbf{W}_2, \mathbf{W}_2, \ldots \mathbf{W}_m\}}$ a $\bigvee\Pi$-view.

Reduced: An FD $\mathbf{Y} \rightarrow A \in \mathcal{F}^+$ is *reduced* if

Equivalence of Meet Complements in the $\bigvee\Pi$-FD framework

Context:
- Universal relational schema $\mathbf{D} = (R[\mathbf{U}], \mathcal{F})$; $\mathcal{F}$ = FDs.
- $\Pi^{\mathbf{D}}_{\{\mathbf{W}_2, \mathbf{W}_2, \ldots \mathbf{W}_m\}}$ a $\bigvee\Pi$-view.

Reduced: An FD $\mathbf{Y} \rightarrow A \in \mathcal{F}^+$ is *reduced* if

- $A \in \mathbf{U}$ (single attribute on RHS)

Context:
- Universal relational schema $\mathbf{D} = (R[\mathbf{U}], \mathcal{F})$; $\mathcal{F}$ = FDs.
- $\Pi^{\mathbf{D}}_{\{\mathbf{W}_2, \mathbf{W}_2, \ldots \mathbf{W}_m\}}$ a $\bigvee\Pi$-view.

Reduced: An FD $\mathbf{Y} \rightarrow A \in \mathcal{F}^+$ is *reduced* if

- $A \in \mathbf{U}$ (single attribute on RHS)
- For any proper subset $\mathbf{Y}' \subsetneq \mathbf{Y}$, $\mathbf{Y}' \rightarrow A \notin \mathcal{F}^+$.

Equivalence of Meet Complements in the $\bigvee\Pi$-FD framework

Context:
- Universal relational schema $\mathbf{D} = (R[\mathbf{U}], \mathcal{F})$; $\mathcal{F}$ = FDs.
- $\Pi^{\mathbf{D}}_{\{\mathbf{W}_2, \mathbf{W}_2, \ldots \mathbf{W}_m\}}$ a $\bigvee\Pi$-view.

Reduced: An FD $\mathbf{Y} \to A \in \mathcal{F}^+$ is *reduced* if

- $A \in \mathbf{U}$ (single attribute on RHS)
- For any proper subset $\mathbf{Y}' \subsetneq \mathbf{Y}$, $\mathbf{Y}' \to A \notin \mathcal{F}^+$.

FD-equivalence: $\mathbf{Y}$ and $\mathbf{Z}$ are *FD-equivalent (for $\mathcal{F}$)*, written $\mathbf{Y} \leftrightarrow \mathbf{Z}$, if both $\mathbf{Y} \to \mathbf{Z}$ and $\mathbf{Z} \to \mathbf{Y}$ hold.

Context:
- Universal relational schema $\mathbf{D} = (R[\mathbf{U}], \mathcal{F})$; $\mathcal{F}$ = FDs.
- $\sqcap^{\mathbf{D}}_{\{\mathbf{W}_2, \mathbf{W}_2, \ldots \mathbf{W}_m\}}$ a $\bigvee\sqcap$-view.

Reduced: An FD $\mathbf{Y} \to A \in \mathcal{F}^+$ is *reduced* if
- $A \in \mathbf{U}$ (single attribute on RHS)
- For any proper subset $\mathbf{Y}' \subsetneq \mathbf{Y}$, $\mathbf{Y}' \to A \notin \mathcal{F}^+$.

FD-equivalence: $\mathbf{Y}$ and $\mathbf{Z}$ are *FD-equivalent (for $\mathcal{F}$)*, written $\mathbf{Y} \leftrightarrow \mathbf{Z}$, if both $\mathbf{Y} \to \mathbf{Z}$ and $\mathbf{Z} \to \mathbf{Y}$ hold.

Definition: $\sqcap^{\mathbf{D}}_{\{\mathbf{W}'_1, \mathbf{W}'_2, \ldots \mathbf{W}'_{m'}\}}$, and $\sqcap^{\mathbf{D}}_{\{\mathbf{W}''_1, \mathbf{W}''_2, \ldots \mathbf{W}''_{m''}\}}$ are *FD-equivalent* if for every $i \in \{1, 2, \ldots, m\}$ and every $\mathbf{Y} \subseteq \mathbf{W}_i$ which is reduced for $\mathcal{F}$, there is a $j \in \{1, 2, \ldots, m''\}$ and a $\mathbf{Z} \subseteq \mathbf{W}''_j$ with $\mathbf{Y} \leftrightarrow \mathbf{Z}$; and conversely.

Equivalence of Meet Complements in the $\bigvee\sqcap$-FD framework

Context:
- Universal relational schema $\mathbf{D} = (R[\mathbf{U}], \mathcal{F})$; $\mathcal{F}$ = FDs.
- $\sqcap^{\mathbf{D}}_{\{\mathbf{W}_2,\mathbf{W}_2,\ldots\mathbf{W}_m\}}$ a $\bigvee\sqcap$-view.

Reduced: An FD $\mathbf{Y} \rightarrow A \in \mathcal{F}^+$ is *reduced* if

- $A \in \mathbf{U}$ (single attribute on RHS)
- For any proper subset $\mathbf{Y}' \subsetneq \mathbf{Y}$, $\mathbf{Y}' \rightarrow A \notin \mathcal{F}^+$.

FD-equivalence: $\mathbf{Y}$ and $\mathbf{Z}$ are *FD-equivalent (for $\mathcal{F}$)*, written $\mathbf{Y} \leftrightarrow \mathbf{Z}$, if both $\mathbf{Y} \rightarrow \mathbf{Z}$ and $\mathbf{Z} \rightarrow \mathbf{Y}$ hold.

Definition: $\sqcap^{\mathbf{D}}_{\{\mathbf{W}'_1,\mathbf{W}'_2,\ldots\mathbf{W}'_{m'}\}}$, and $\sqcap^{\mathbf{D}}_{\{\mathbf{W}''_1,\mathbf{W}''_2,\ldots\mathbf{W}''_{m''}\}}$ are *FD-equivalent* if for every $i \in \{1, 2, \ldots, m\}$ and every $\mathbf{Y} \subseteq \mathbf{W}_i$ which is reduced for $\mathcal{F}$, there is a $j \in \{1, 2, \ldots, m''\}$ and a $\mathbf{Z} \subseteq \mathbf{W}''_j$ with $\mathbf{Y} \leftrightarrow \mathbf{Z}$; and conversely.

Theorem: Any two meet complements are FD-equivalent. $\square$

Equivalence of Meet Complements in the $\bigvee\!\sqcap$-FD framework

Context:
- Universal relational schema $\mathbf{D} = (R[\mathbf{U}], \mathcal{F})$; $\mathcal{F}$ = FDs.
- $\sqcap^{\mathbf{D}}_{\{\mathbf{W}_2,\mathbf{W}_2,\ldots\mathbf{W}_m\}}$ a $\bigvee\!\sqcap$-view.

Reduced: An FD $\mathbf{Y} \rightarrow A \in \mathcal{F}^+$ is *reduced* if
- $A \in \mathbf{U}$ (single attribute on RHS)
- For any proper subset $\mathbf{Y}' \subsetneq \mathbf{Y}$, $\mathbf{Y}' \rightarrow A \notin \mathcal{F}^+$.

FD-equivalence: $\mathbf{Y}$ and $\mathbf{Z}$ are *FD-equivalent (for $\mathcal{F}$)*, written $\mathbf{Y} \leftrightarrow \mathbf{Z}$, if both $\mathbf{Y} \rightarrow \mathbf{Z}$ and $\mathbf{Z} \rightarrow \mathbf{Y}$ hold.

Definition: $\sqcap^{\mathbf{D}}_{\{\mathbf{W}'_1,\mathbf{W}'_2,\ldots\mathbf{W}'_{m'}\}}$, and $\sqcap^{\mathbf{D}}_{\{\mathbf{W}''_1,\mathbf{W}''_2,\ldots\mathbf{W}''_{m''}\}}$ are *FD-equivalent* if for every $i \in \{1, 2, \ldots, m\}$ and every $\mathbf{Y} \subseteq \mathbf{W}_i$ which is reduced for $\mathcal{F}$, there is a $j \in \{1, 2, \ldots, m''\}$ and a $\mathbf{Z} \subseteq \mathbf{W}''_j$ with $\mathbf{Y} \leftrightarrow \mathbf{Z}$; and conversely.

Theorem: Any two meet complements are FD-equivalent. $\square$

Corollary If $\mathcal{F}$ does not contain any nontrivial FD-equivalences ($\mathbf{Y} \neq \mathbf{Z}$), then $\sqcap^{\mathbf{D}}_{\{\mathbf{W}_1,\mathbf{W}_2,\ldots\mathbf{W}_m\}}$ has a unique optimal meet $\bigvee\!\sqcap$-complement. $\square$

Examples of Equivalent Meet Complements

Context:
- $\mathbf{E}_5 = (R[ABCDE], \mathcal{F}_3)$
- $\mathcal{F}_5 = \{B \to C, C \to D, D \to E\}$
- $\Pi^{\mathbf{E}_3}_{\{AB,\ CD\}}$

Examples of Equivalent Meet Complements

Context:
- $\mathbf{E}_5 = (R[ABCDE], \mathcal{F}_3)$
- $\mathcal{F}_5 = \{B \to C, C \to D, D \to E\}$
- $\sqcap^{\mathbf{E}_3}_{\{AB,\, CD\}}$

- $\mathcal{F}_5$ implies no nontrivial FD-equivalences.

Examples of Equivalent Meet Complements

Context:
- $\mathbf{E}_5 = (R[ABCDE], \mathcal{F}_3)$
- $\mathcal{F}_5 = \{B \to C, C \to D, D \to E\}$
- $\Pi^{\mathbf{E}_3}_{\{AB,\, CD\}}$

- $\mathcal{F}_5$ implies no nontrivial FD-equivalences.
- The view $\Pi^{\mathbf{E}_5}_{\{AB,\, CD\}}$ has a unique meet $\bigvee\Pi$-complement: $\Pi^{\mathbf{E}_5}_{\{BC,\, DE\}}$

Examples of Equivalent Meet Complements

Context:
- $\mathbf{E}_5 = (R[ABCDE], \mathcal{F}_3)$
- $\mathcal{F}_5 = \{B \to C, C \to D, D \to E\}$
- $\Pi^{\mathbf{E}_3}_{\{AB,\ CD\}}$

- $\mathcal{F}_5$ implies no nontrivial FD-equivalences.
- The view $\Pi^{\mathbf{E}_5}_{\{AB,\ CD\}}$ has a unique meet $\bigvee\Pi$-complement: $\Pi^{\mathbf{E}_5}_{\{BC,\ DE\}}$

Context:
- $\mathbf{E}_6 = (R[AB_{11}B_{12}B_2CD_1D_2E], \mathcal{F}_6)$
- $\mathcal{F}_6 = \{B_{11}B_{12} \leftrightarrow B_2, D_1 \leftrightarrow D_2, B_1 \to C, C \to D_1, D_1 \to E\}$
- $\Pi^{\mathbf{E}_6}_{\{AB_{11}B_{12}B_2,\ CD_1D_2\}}$

Examples of Equivalent Meet Complements

Context:
- $\mathbf{E}_5 = (R[ABCDE], \mathcal{F}_3)$
- $\mathcal{F}_5 = \{B \to C, C \to D, D \to E\}$
- $\Pi^{\mathbf{E}_3}_{\{AB,\ CD\}}$

- $\mathcal{F}_5$ implies no nontrivial FD-equivalences.

- The view $\Pi^{\mathbf{E}_5}_{\{AB,\ CD\}}$ has a unique meet $\bigvee\!\Pi$-complement: $\Pi^{\mathbf{E}_5}_{\{BC,\ DE\}}$

Context:
- $\mathbf{E}_6 = (R[AB_{11}B_{12}B_2CD_1D_2E], \mathcal{F}_6)$
- $\mathcal{F}_6 = \{B_{11}B_{12} \leftrightarrow B_2, D_1 \leftrightarrow D_2, B_1 \to C, C \to D_1, D_1 \to E\}$
- $\Pi^{\mathbf{E}_6}_{\{AB_{11}B_{12}B_2,\ CD_1D_2\}}$

- $\mathcal{F}_6$ implies two nontrivial FD-equivalences: $B_{11}B_{12} \leftrightarrow B_2$ and $D_1 \leftrightarrow D_2$.

Examples of Equivalent Meet Complements

Context:
- $\mathbf{E}_5 = (R[ABCDE], \mathcal{F}_3)$
- $\mathcal{F}_5 = \{B \to C, C \to D, D \to E\}$
- $\Pi^{\mathbf{E}_3}_{\{AB,\ CD\}}$

- $\mathcal{F}_5$ implies no nontrivial FD-equivalences.
- The view $\Pi^{\mathbf{E}_5}_{\{AB,\ CD\}}$ has a unique meet $\bigvee\Pi$-complement: $\Pi^{\mathbf{E}_5}_{\{BC,\ DE\}}$

Context:
- $\mathbf{E}_6 = (R[AB_{11}B_{12}B_2CD_1D_2E], \mathcal{F}_6)$
- $\mathcal{F}_6 = \{B_{11}B_{12} \leftrightarrow B_2, D_1 \leftrightarrow D_2, B_1 \to C, C \to D_1, D_1 \to E\}$
- $\Pi^{\mathbf{E}_6}_{\{AB_{11}B_{12}B_2,\ CD_1D_2\}}$

- $\mathcal{F}_6$ implies two nontrivial FD-equivalences: $B_{11}B_{12} \leftrightarrow B_2$ and $D_1 \leftrightarrow D_2$.
- The view $\Pi^{\mathbf{E}_6}_{\{AB_{11}B_{12}B_2,\ CD_1D_2\}}$ has four distinct meet complements:

$\Pi^{\mathbf{E}_6}_{\{B_{11}B_{12}C,\ D_1E\}}$ $\Pi^{\mathbf{E}_6}_{\{B_{11}B_{12}C,\ D_2E\}}$ $\Pi^{\mathbf{E}_6}_{\{B_2C,\ D_1E\}}$ $\Pi^{\mathbf{E}_6}_{\{B_2C,\ D_2E\}}$

Extension to "Real-World" Situations

- *"Real world"* schemata have:

Extension to "Real-World" Situations

- *"Real world"* schemata have:
  - Multiple relations

Extension to "Real-World" Situations

- *"Real world"* schemata have:
  - Multiple relations
  - Referential integrity constraints (foreign-key dependencies):

Extension to "Real-World" Situations

- *"Real world"* schemata have:
  - Multiple relations
  - Referential integrity constraints (foreign-key dependencies):
- The extension to multirelational schemata with FDs is trivial.

Extension to “Real-World” Situations

- *“Real world”* schemata have:
  - Multiple relations
  - Referential integrity constraints (foreign-key dependencies):
- The extension to multirelational schemata with FDs is trivial.
  - Apply previous results on a relation-by-relation basis.

Extension to “Real-World” Situations

- *“Real world”* schemata have:
  - Multiple relations
  - Referential integrity constraints (foreign-key dependencies):
- The extension to multirelational schemata with FDs is trivial.
  - Apply previous results on a relation-by-relation basis.
- The theory also extends to *fanout-free* unary inclusion dependencies:

Extension to "Real-World" Situations

- *"Real world"* schemata have:
  - Multiple relations
  - Referential integrity constraints (foreign-key dependencies):
- The extension to multirelational schemata with FDs is trivial.
  - Apply previous results on a relation-by-relation basis.
- The theory also extends to *fanout-free* unary inclusion dependencies:
  - $(R[A] \subseteq S[B] \wedge R[A] \subseteq T[C]) \Rightarrow (S[B] \subseteq T[C] \vee T[C] \subseteq S[B])$.

Extension to "Real-World" Situations

- *"Real world"* schemata have:
  - Multiple relations
  - Referential integrity constraints (foreign-key dependencies):
- The extension to multirelational schemata with FDs is trivial.
  - Apply previous results on a relation-by-relation basis.
- The theory also extends to *fanout-free* unary inclusion dependencies:
  - $(R[A] \subseteq S[B] \wedge R[A] \subseteq T[C]) \Rightarrow (S[B] \subseteq T[C] \vee T[C] \subseteq S[B])$.
  - Foreign-key dependencies are always fanout free.

Extension to “Real-World” Situations

- *“Real world”* schemata have:
  - Multiple relations
  - Referential integrity constraints (foreign-key dependencies):
- The extension to multirelational schemata with FDs is trivial.
  - Apply previous results on a relation-by-relation basis.
- The theory also extends to *fanout-free* unary inclusion dependencies:
  - $(R[A] \subseteq S[B] \wedge R[A] \subseteq T[C]) \Rightarrow (S[B] \subseteq T[C] \vee T[C] \subseteq S[B])$.
  - Foreign-key dependencies are always fanout free.
- Each *one-way UID* must always be embedded into one of the two views.

Extension to "Real-World" Situations

- *"Real world"* schemata have:
  - Multiple relations
  - Referential integrity constraints (foreign-key dependencies):
- The extension to multirelational schemata with FDs is trivial.
  - Apply previous results on a relation-by-relation basis.
- The theory also extends to *fanout-free* unary inclusion dependencies:
  - $(R[A] \subseteq S[B] \wedge R[A] \subseteq T[C]) \Rightarrow (S[B] \subseteq T[C] \vee T[C] \subseteq S[B])$.
  - Foreign-key dependencies are always fanout free.
- Each *one-way UID* must always be embedded into one of the two views.
  One-way UID: $R[A] \subseteq S[B]$ holds; $S[B] \subseteq R[A]$ does not.

Extension to "Real-World" Situations

- *"Real world"* schemata have:
  - Multiple relations
  - Referential integrity constraints (foreign-key dependencies):
- The extension to multirelational schemata with FDs is trivial.
  - Apply previous results on a relation-by-relation basis.
- The theory also extends to *fanout-free* unary inclusion dependencies:
  - $(R[A] \subseteq S[B] \wedge R[A] \subseteq T[C]) \Rightarrow (S[B] \subseteq T[C] \vee T[C] \subseteq S[B])$.
  - Foreign-key dependencies are always fanout free.
- Each *one-way UID* must always be embedded into one of the two views.
  One-way UID: $R[A] \subseteq S[B]$ holds; $S[B] \subseteq R[A]$ does not.
- *Two-way* UIDS ($R[A] = S[B]$) define true isomorphism, and must satisfy a condition similar to FD-equivalence.

Extension to "Real-World" Situations

- *"Real world"* schemata have:
  - Multiple relations
  - Referential integrity constraints (foreign-key dependencies):
- The extension to multirelational schemata with FDs is trivial.
  - Apply previous results on a relation-by-relation basis.
- The theory also extends to *fanout-free* unary inclusion dependencies:
  - $(R[A] \subseteq S[B] \wedge R[A] \subseteq T[C]) \Rightarrow (S[B] \subseteq T[C] \vee T[C] \subseteq S[B])$.
  - Foreign-key dependencies are always fanout free.
- Each *one-way UID* must always be embedded into one of the two views.

  One-way UID: $R[A] \subseteq S[B]$ holds; $S[B] \subseteq R[A]$ does not.
- *Two-way* UIDS ($R[A] = S[B]$) define true isomorphism, and must satisfy a condition similar to FD-equivalence.

Bottom line: The extension to multirelational settings constrained by both FDs and fanout-free UIDs is complete.

Extension to "Real-World" Situations

- *"Real world"* schemata have:
  - Multiple relations
  - Referential integrity constraints (foreign-key dependencies):
- The extension to multirelational schemata with FDs is trivial.
  - Apply previous results on a relation-by-relation basis.
- The theory also extends to *fanout-free* unary inclusion dependencies:
  - $(R[A] \subseteq S[B] \wedge R[A] \subseteq T[C]) \Rightarrow (S[B] \subseteq T[C] \vee T[C] \subseteq S[B])$.
  - Foreign-key dependencies are always fanout free.
- Each *one-way UID* must always be embedded into one of the two views.
  One-way UID: $R[A] \subseteq S[B]$ holds; $S[B] \subseteq R[A]$ does not.
- *Two-way* UIDS ($R[A] = S[B]$) define true isomorphism, and must satisfy a condition similar to FD-equivalence.

Bottom line: The extension to multirelational settings constrained by both FDs and fanout-free UIDs is complete.

- Certain useful cases of non-unary IDs can also be handled.

Conclusions and Further Directions

Conclusions:

Further Directions:

Conclusions and Further Directions

Conclusions:

- Three distinct forms of invariance have been considered for constant-complement update:

Further Directions:

Conclusions and Further Directions

Conclusions:

- Three distinct forms of invariance have been considered for constant-complement update:

  State invariance: The existence of a reflection does not depend upon the state of the complement.

Further Directions:

Conclusions and Further Directions

Conclusions:

- Three distinct forms of invariance have been considered for constant-complement update:

  State invariance: The existence of a reflection does not depend upon the state of the complement.

  Reflection invariance: The reflection of a view update is identical for all complements which support it.

Further Directions:

Conclusions and Further Directions

Conclusions:

- Three distinct forms of invariance have been considered for constant-complement update:

  State invariance: The existence of a reflection does not depend upon the state of the complement.

  Reflection invariance: The reflection of a view update is identical for all complements which support it.

  Update-set invariance: There is a single complement which supports all constant-complement updates.

Further Directions:

Conclusions and Further Directions

Conclusions:

- Three distinct forms of invariance have been considered for constant-complement update:

  State invariance: The existence of a reflection does not depend upon the state of the complement.

  Reflection invariance: The reflection of a view update is identical for all complements which support it.

  Update-set invariance: There is a single complement which supports all constant-complement updates.

- Reasonably broad theories characterizing the first two forms of invariance have been developed.

Further Directions:

Conclusions and Further Directions

Conclusions:

- Three distinct forms of invariance have been considered for constant-complement update:
  - State invariance: The existence of a reflection does not depend upon the state of the complement.
  - Reflection invariance: The reflection of a view update is identical for all complements which support it.
  - Update-set invariance: There is a single complement which supports all constant-complement updates.
- Reasonably broad theories characterizing the first two forms of invariance have been developed.

Further Directions:

- Pursue a more general theory of optimal meet complements which is not dependent upon specific constraints and the relational model.