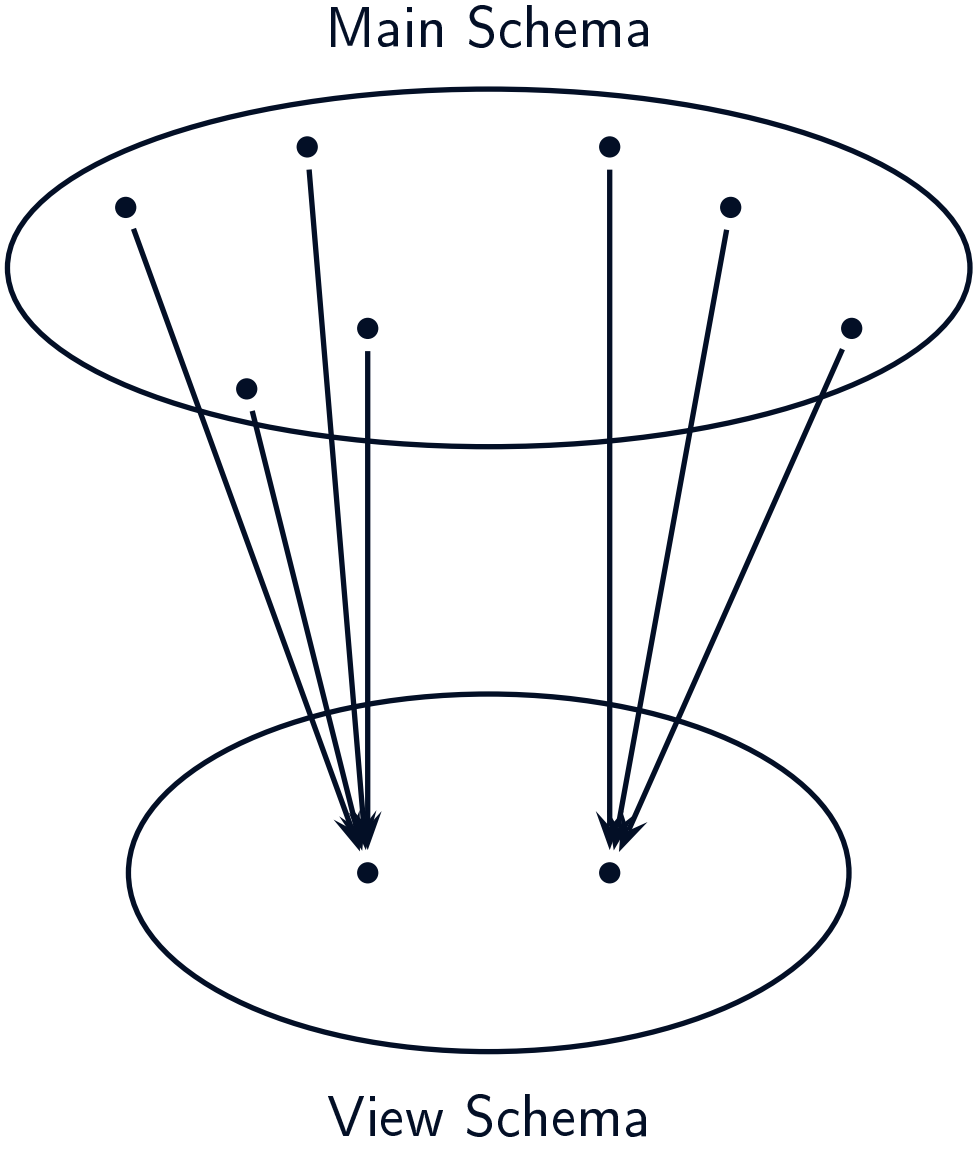
Optimal Reflection of Bidirectional View Updates using Information-Based Distance Measures

Stephen J. Hegner
Umeå University
Department of Computing Science
Sweden

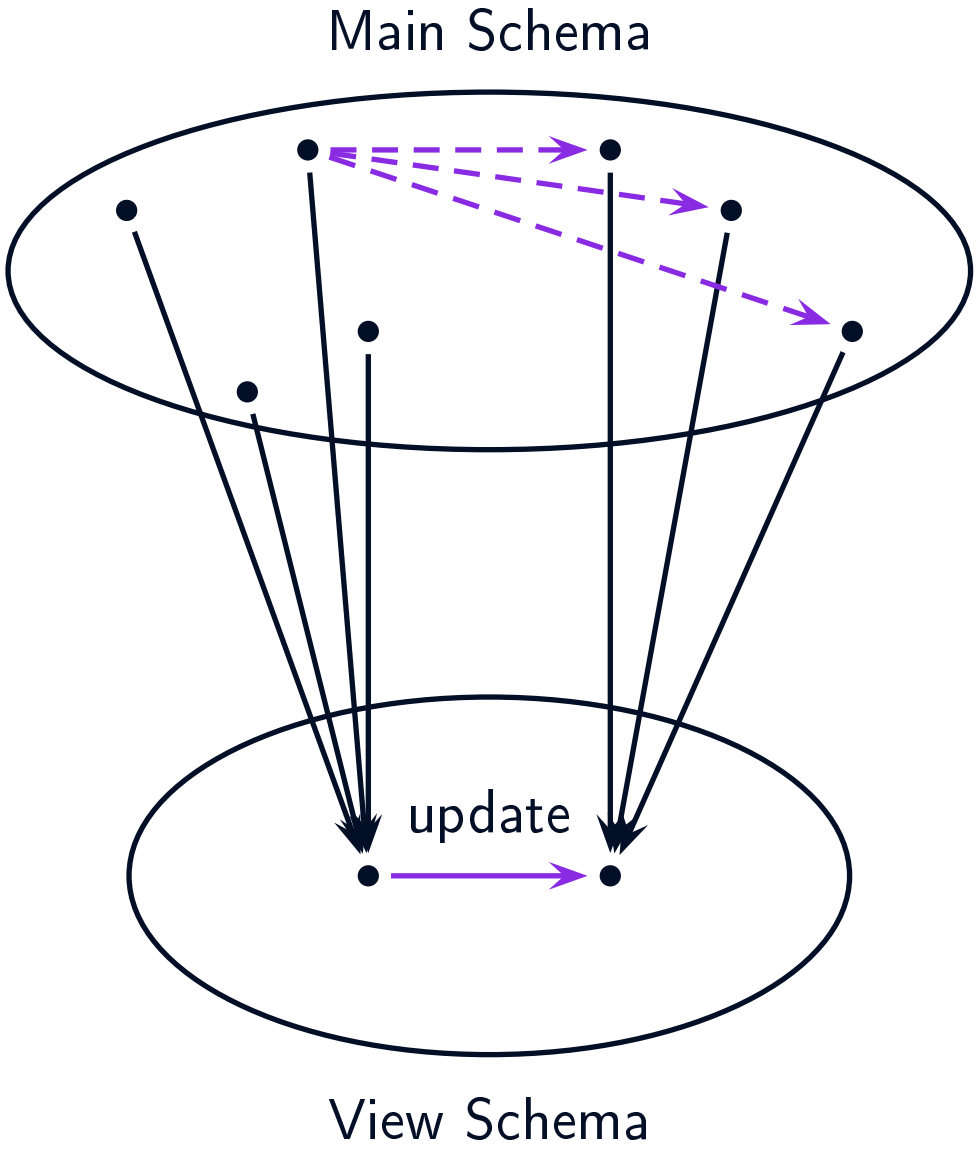
The Update Problem for Database Views

- On the underlying states, the view mapping is generally *surjective* (onto) but not *injective* (one-to-one).



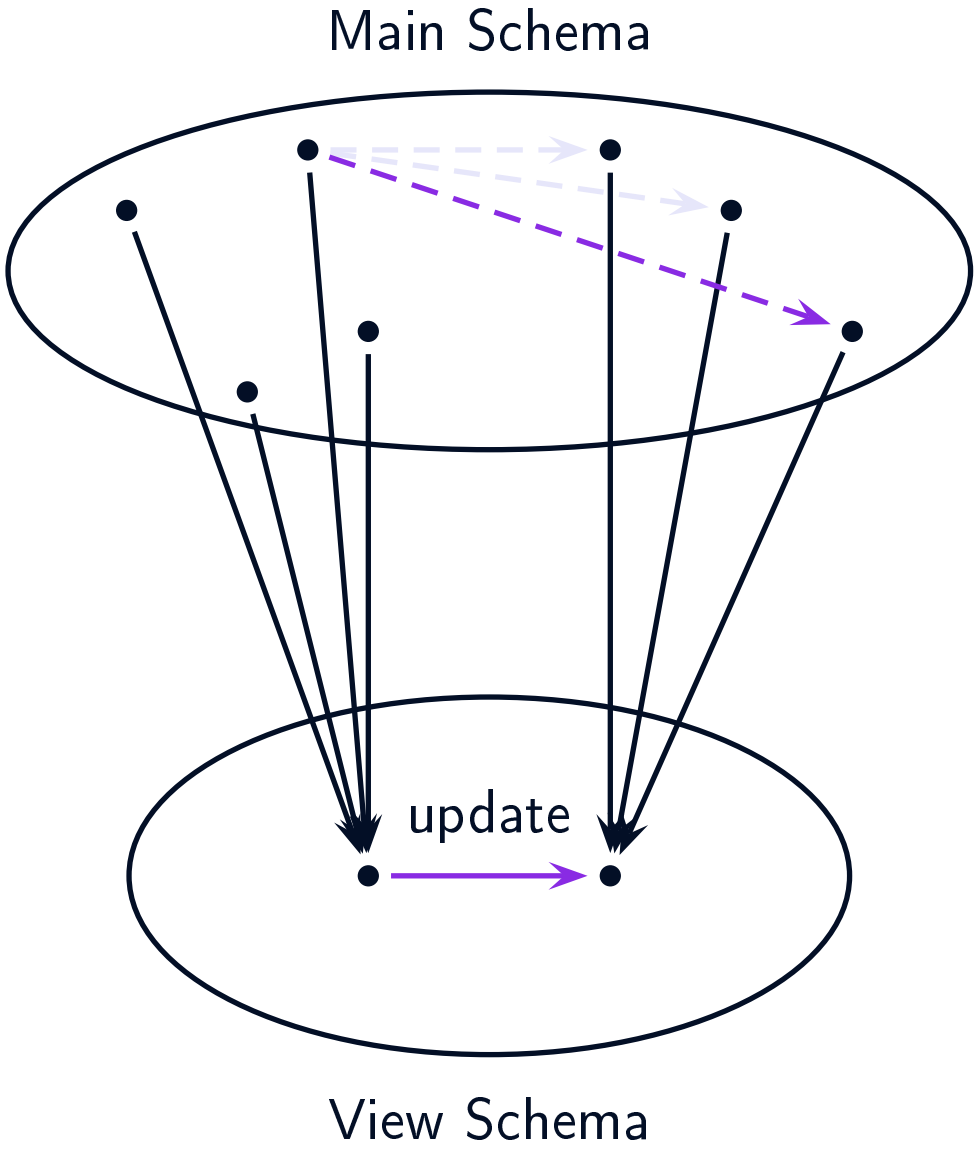
The Update Problem for Database Views

- On the underlying states, the view mapping is generally *surjective* (onto) but not *injective* (one-to-one).
- Thus, a view update has many possible *reflections* to the main schema.



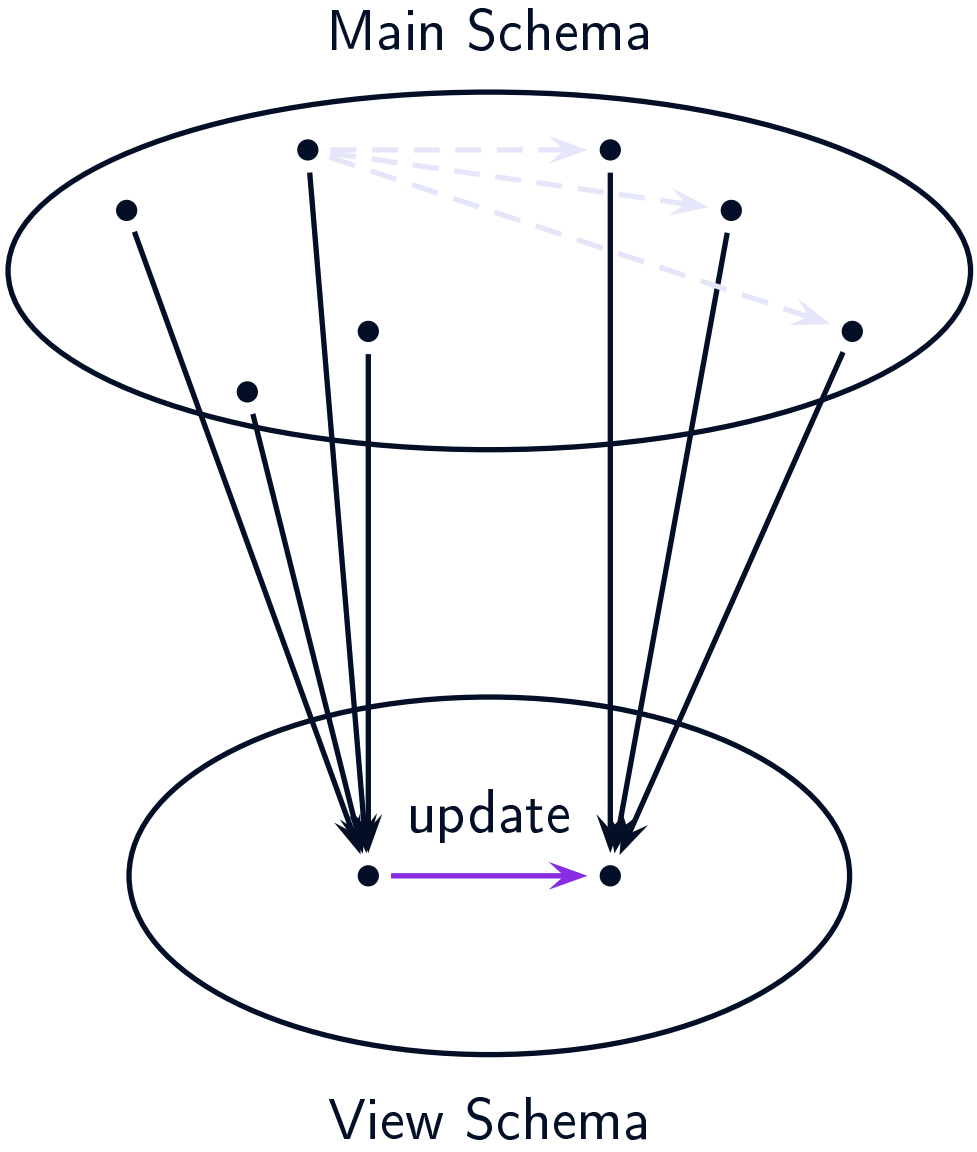
The Update Problem for Database Views

- On the underlying states, the view mapping is generally *surjective* (onto) but not *injective* (one-to-one).
- Thus, a view update has many possible *reflections* to the main schema.
- The problem of identifying a suitable reflection is known as the *update translation problem* or *update reflection problem*.



The Update Problem for Database Views

- On the underlying states, the view mapping is generally *surjective* (onto) but not *injective* (one-to-one).
- Thus, a view update has many possible *reflections* to the main schema.
- The problem of identifying a suitable reflection is known as the *update translation problem* or *update reflection problem*.
- With a reasonable definition of suitability, it may not be the case that every view update has a suitable translation.



Three Main Approaches to the View Update Problem

- *The Constant-Complement Strategy*:

- *Methods Based upon the Relational Algebra*:

- *Optimization of Distance Change*:

Three Main Approaches to the View Update Problem

- *The Constant-Complement Strategy*:
  - Formalizes notion of encapsulated schema perfectly.
  - All view updates are reversible and composable.
  - Strong requirements; relatively few updates supported.
- *Methods Based upon the Relational Algebra*:
- *Optimization of Distance Change*:

Three Main Approaches to the View Update Problem

- *The Constant-Complement Strategy*:
  - Formalizes notion of encapsulated schema perfectly.
  - All view updates are reversible and composable.
  - Strong requirements; relatively few updates supported.
- *Methods Based upon the Relational Algebra*:
  - Simple to understand and intuitive.
  - A large collection of special cases.
  - Lacks a unifying theory.
- *Optimization of Distance Change*:

Three Main Approaches to the View Update Problem

- *The Constant-Complement Strategy*:
  - Formalizes notion of encapsulated schema perfectly.
  - All view updates are reversible and composable.
  - Strong requirements; relatively few updates supported.
- *Methods Based upon the Relational Algebra*:
  - Simple to understand and intuitive.
  - A large collection of special cases.
  - Lacks a unifying theory.
- *Optimization of Distance Change*:
  - Minimize distance between old and new states of main schema.
  - Intuitively appealing.
  - Challenge is to identify a suitable notion of distance.

Three Main Approaches to the View Update Problem

- *The Constant-Complement Strategy*:
  - Formalizes notion of encapsulated schema perfectly.
  - All view updates are reversible and composable.
  - Strong requirements; relatively few updates supported.
- *Methods Based upon the Relational Algebra*:
  - Simple to understand and intuitive.
  - A large collection of special cases.
  - Lacks a unifying theory.
- *Optimization of Distance Change*:
  - Minimize distance between old and new states of main schema.
  - Intuitively appealing.
  - Challenge is to identify a suitable notion of distance.
  - Focus of this talk.

Typical Characterization of Distance

- Given is the two-relation main schema $\mathbf{E}_0$.

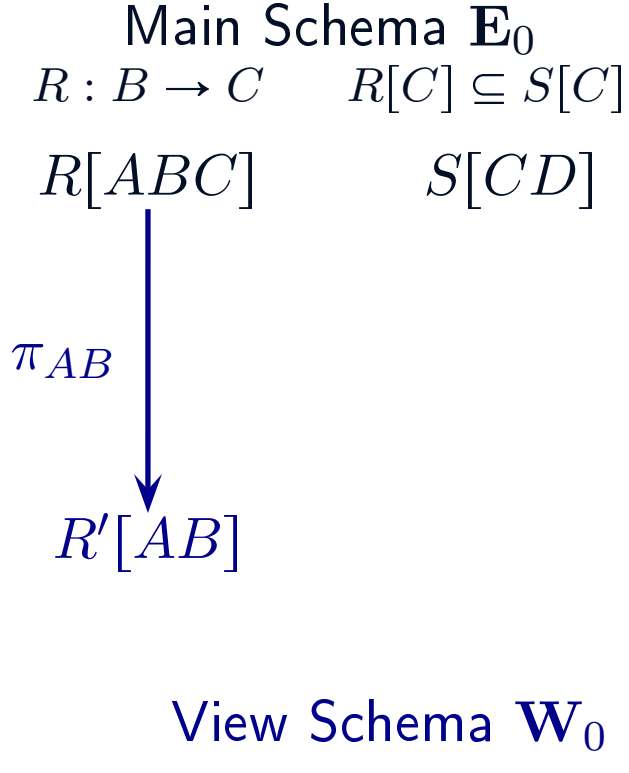
Main Schema $\mathbf{E}_0$

$R : B \rightarrow C \qquad R[C] \subseteq S[C]$

$R[ABC] \qquad S[CD]$

Typical Characterization of Distance

- Given is the two-relation main schema $\mathbf{E}_0$.
- View schema $\mathbf{W}_0$: AB projection of R.

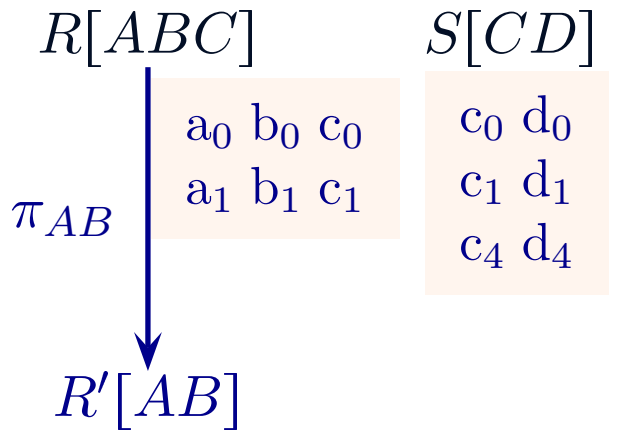


Typical Characterization of Distance

- Given is the two-relation main schema $\mathbf{E}_0$.
- View schema $\mathbf{W}_0$: AB projection of R.

Main Schema $\mathbf{E}_0$

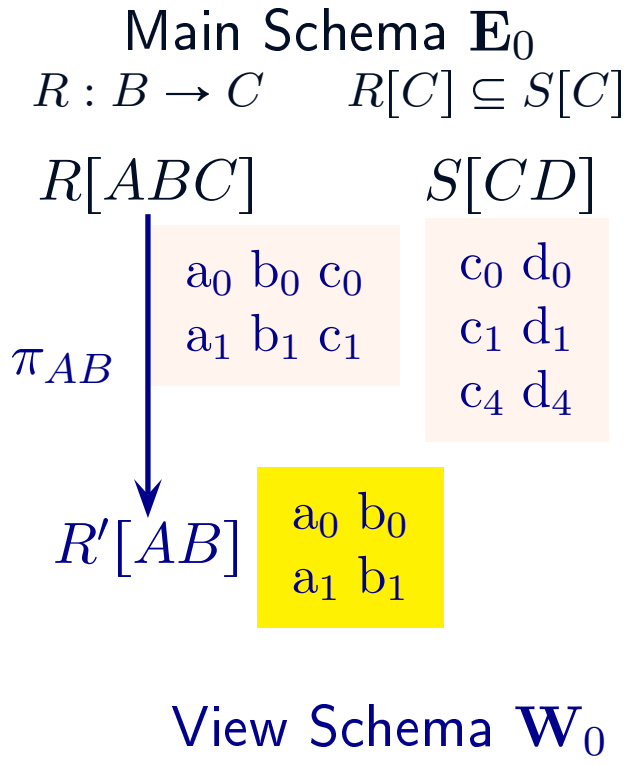
$R : B \to C$ $\quad$ $R[C] \subseteq S[C]$



View Schema $\mathbf{W}_0$

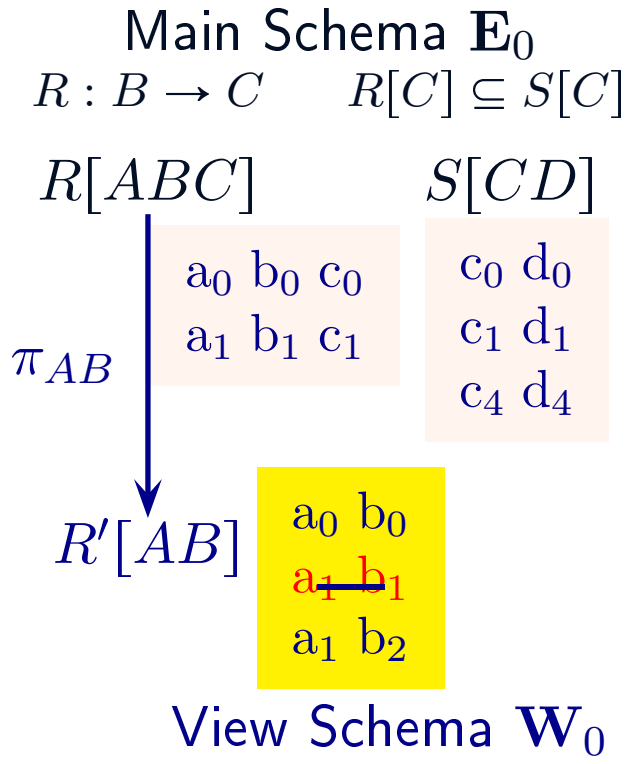
Typical Characterization of Distance

- Given is the two-relation main schema $\mathbf{E}_0$.
- View schema $\mathbf{W}_0$: AB projection of R.



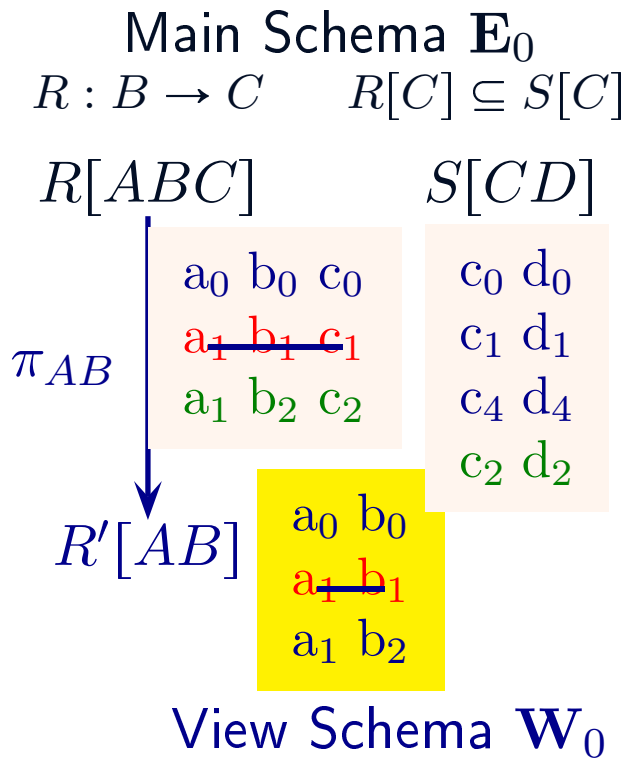
Typical Characterization of Distance

- Given is the two-relation main schema $\mathbf{E}_0$.
- View schema $\mathbf{W}_0$: AB projection of R.
- Desired view update: replace $R'(\mathrm{a}_1, \mathrm{b}_1)$ with $R'(\mathrm{a}_1, \mathrm{b}_2)$.



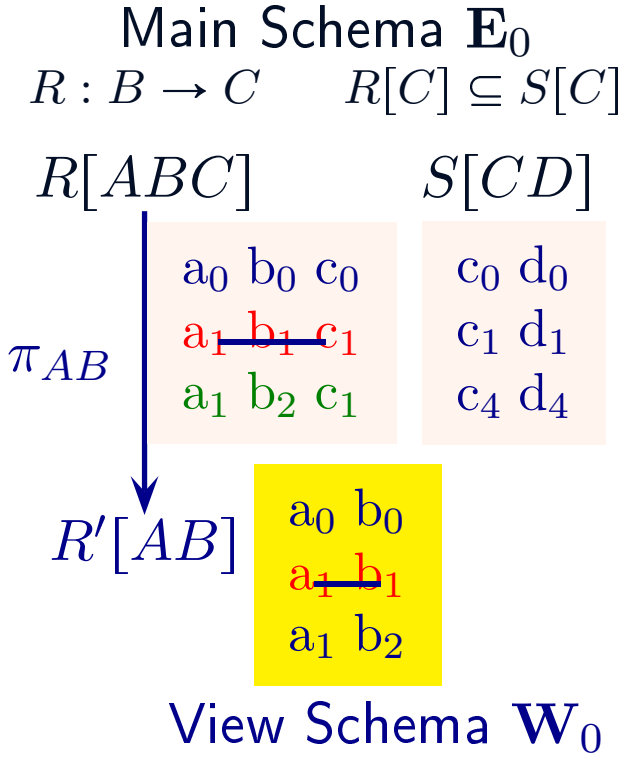
Typical Characterization of Distance

- Given is the two-relation main schema $\mathbf{E}_0$.
- View schema $\mathbf{W}_0$: AB projection of R.
- Desired view update: replace $R'(\mathrm{a}_1, \mathrm{b}_1)$ with $R'(\mathrm{a}_1, \mathrm{b}_2)$.
- The reflection shown is *tuple minimal* — no subset realizes the update. (A non-numerical distance)



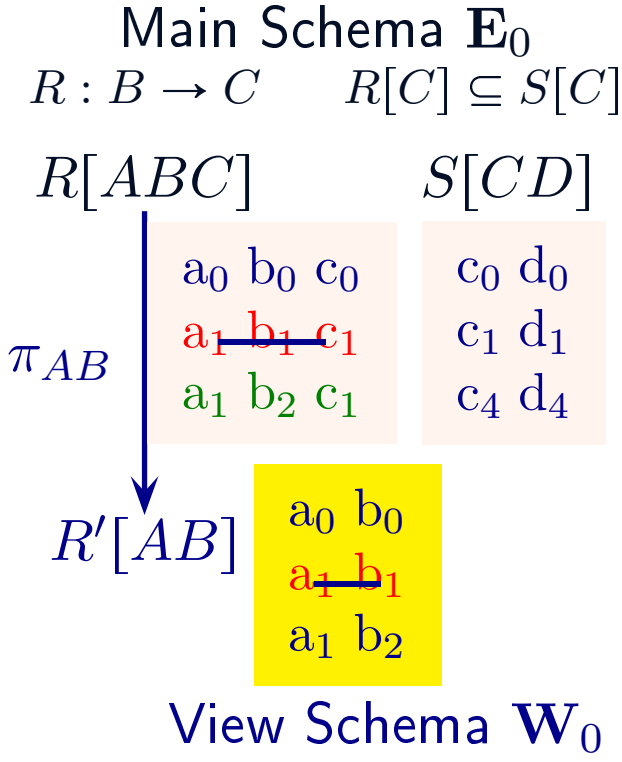
Typical Characterization of Distance

- Given is the two-relation main schema $\mathbf{E}_0$.
- View schema $\mathbf{W}_0$: AB projection of R.
- Desired view update: replace $R'(\mathrm{a}_1, \mathrm{b}_1)$ with $R'(\mathrm{a}_1, \mathrm{b}_2)$.
- The reflection shown is *tuple minimal* — no subset realizes the update. (A non-numerical distance)
- This alternate reflection is *count minimal* as well as tuple minimal — there is no reflection involving fewer tuples. (A numerical distance)



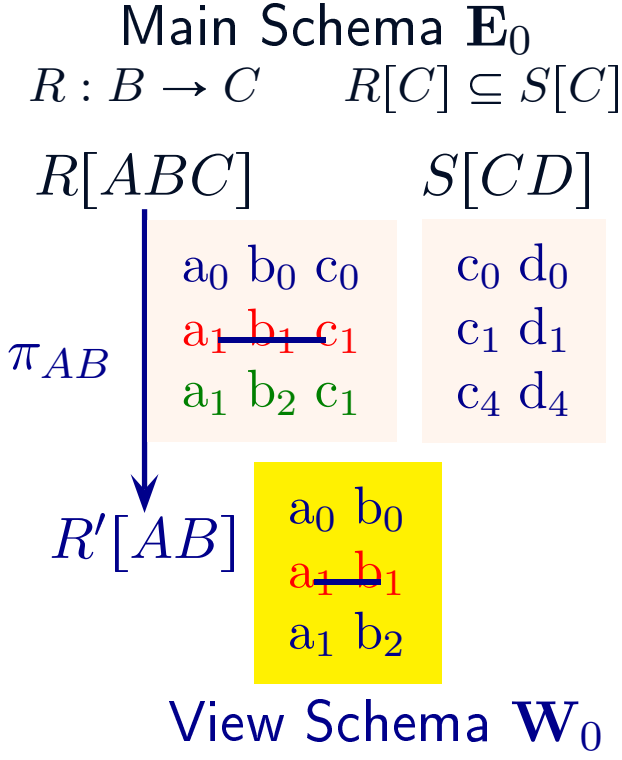
Typical Characterization of Distance

- Given is the two-relation main schema $\mathbf{E}_0$.
- View schema $\mathbf{W}_0$: AB projection of R.
- Desired view update: replace $R'(\mathrm{a}_1, \mathrm{b}_1)$ with $R'(\mathrm{a}_1, \mathrm{b}_2)$.
- The reflection shown is *tuple minimal* — no subset realizes the update. (A non-numerical distance)
- This alternate reflection is *count minimal* as well as tuple minimal — there is no reflection involving fewer tuples. (A numerical distance)
- It is even *term minimal* – the fewest number of terms in tuples have been changed. (A numerical distance)



Typical Characterization of Distance

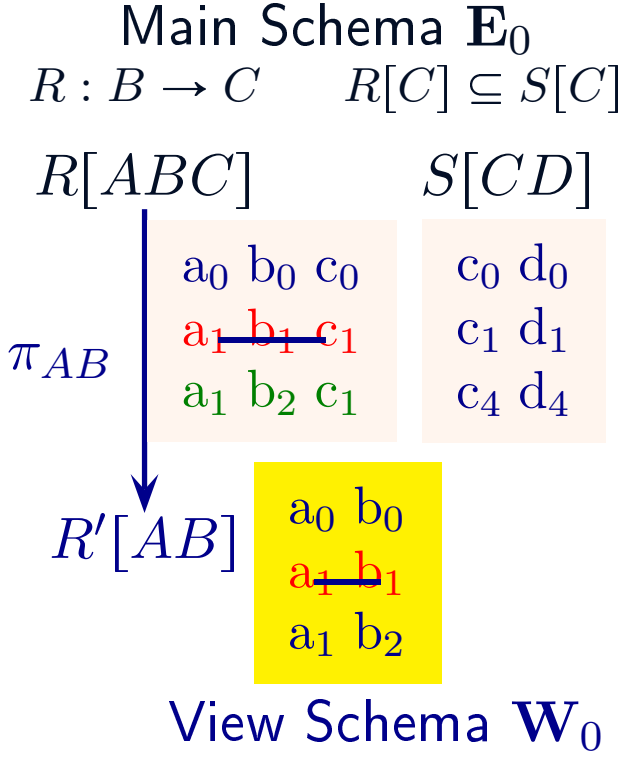
- Given is the two-relation main schema $\mathbf{E}_0$.
- View schema $\mathbf{W}_0$: AB projection of R.
- Desired view update: replace $R'(\mathrm{a}_1, \mathrm{b}_1)$ with $R'(\mathrm{a}_1, \mathrm{b}_2)$.
- The reflection shown is *tuple minimal* — no subset realizes the update. (A non-numerical distance)
- This alternate reflection is *count minimal* as well as tuple minimal — there is no reflection involving fewer tuples. (A numerical distance)
- It is even *term minimal* – the fewest number of terms in tuples have been changed. (A numerical distance)



- More sophisticated notions of distance between tuples have been proposed ([Hutchinson 1997] [Nienhuys-Cheng 1997]).

Typical Characterization of Distance

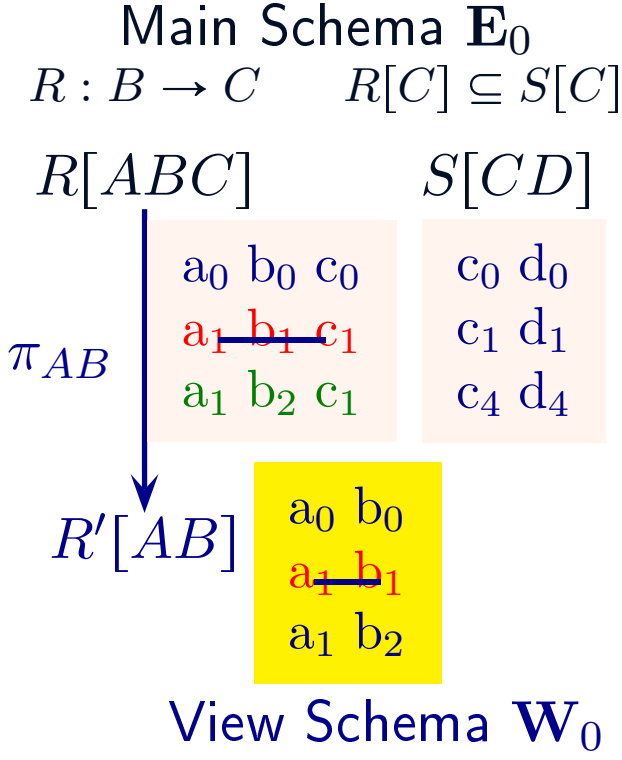
- Given is the two-relation main schema $\mathbf{E}_0$.
- View schema $\mathbf{W}_0$: AB projection of R.
- Desired view update: replace $R'(\mathrm{a}_1, \mathrm{b}_1)$ with $R'(\mathrm{a}_1, \mathrm{b}_2)$.
- The reflection shown is *tuple minimal* — no subset realizes the update. (A non-numerical distance)
- This alternate reflection is *count minimal* as well as tuple minimal — there is no reflection involving fewer tuples. (A numerical distance)
- It is even *term minimal* – the fewest number of terms in tuples have been changed. (A numerical distance)



- More sophisticated notions of distance between tuples have been proposed ([Hutchinson 1997] [Nienhuys-Cheng 1997]).
- These may in turn be used in formulas to compute a numerical distance between sets of tuples (Eiter-Mannila 1997]).

Typical Characterization of Distance

- Given is the two-relation main schema $\mathbf{E}_0$.
- View schema $\mathbf{W}_0$: AB projection of R.
- Desired view update: replace $R'(\mathrm{a}_1, \mathrm{b}_1)$ with $R'(\mathrm{a}_1, \mathrm{b}_2)$.
- The reflection shown is *tuple minimal* — no subset realizes the update. (A non-numerical distance)
- This alternate reflection is *count minimal* as well as tuple minimal — there is no reflection involving fewer tuples. (A numerical distance)
- It is even *term minimal* – the fewest number of terms in tuples have been changed. (A numerical distance)



- More sophisticated notions of distance between tuples have been proposed ([Hutchinson 1997] [Nienhuys-Cheng 1997]).
- These may in turn be used in formulas to compute a numerical distance between sets of tuples (Eiter-Mannila 1997]).
- *Question:* Are such *syntactic* notions of distance always the best choice?

Information Content

Idea: Exploit the first-order properties of the model, rather than just counting and summing syntactic differences.

Information Content

Idea: Exploit the first-order properties of the model, rather than just counting and summing syntactic differences.

- $\mathrm{WFS}(\mathbf{D}, \exists\wedge+)$ denotes the set of all *Boolean conjunctive queries* on the database schema $\mathbf{D}$: $\exists$ $\wedge$

  Examples: $R(\mathrm{a}_0, \mathrm{b}_0, \mathrm{c}_0)$, $(\exists y)(\exists z)(R(\mathrm{a}_0, y, z) \wedge S(y, \mathrm{d}_0))$

Information Content

Idea: Exploit the first-order properties of the model, rather than just counting and summing syntactic differences.

- WFS$(\mathbf{D}, \exists\wedge+)$ denotes the set of all *Boolean conjunctive queries* on the database schema $\mathbf{D}$: $\exists$ $\wedge$

  Examples: $R(\mathrm{a}_0, \mathrm{b}_0, \mathrm{c}_0)$, $(\exists y)(\exists z)(R(\mathrm{a}_0, y, z) \wedge S(y, \mathrm{d}_0))$

- WFS$(\mathbf{D}, \exists\wedge+, K) = \Upsilon^{\mathbf{D}}_K$ = subset of WFS$(\mathbf{D}, \exists\wedge+)$ involving only the constants in K.

Information Content

Idea: Exploit the first-order properties of the model, rather than just counting and summing syntactic differences.

- WFS$(\mathbf{D}, \exists\wedge+)$ denotes the set of all *Boolean conjunctive queries* on the database schema $\mathbf{D}$: $\exists$ $\wedge$ $\not\forall$ $\not\vee$ $\not\neg$

  Examples: $R(\mathrm{a}_0, \mathrm{b}_0, \mathrm{c}_0)$, $\quad (\exists y)(\exists z)(R(\mathrm{a}_0, y, z) \wedge S(y, \mathrm{d}_0))$

- WFS$(\mathbf{D}, \exists\wedge+, K) = \Upsilon^{\mathbf{D}}_K$ = subset of WFS$(\mathbf{D}, \exists\wedge+)$ involving only the constants in K.

- The *information content* of database M relative to $\Upsilon^{\mathbf{D}}_K$:

$$\mathsf{Info}\langle M, \Upsilon^{\mathbf{D}}_K\rangle = \{\varphi \in \Upsilon^{\mathbf{D}}_K \mid M \models \varphi\}$$

Information Content

Idea: Exploit the first-order properties of the model, rather than just counting and summing syntactic differences.

- WFS$(\mathbf{D}, \exists\wedge+)$ denotes the set of all *Boolean conjunctive queries* on the database schema $\mathbf{D}$: $\exists$ $\wedge$

  Examples: $R(\mathrm{a}_0, \mathrm{b}_0, \mathrm{c}_0)$, $\quad (\exists y)(\exists z)(R(\mathrm{a}_0, y, z) \wedge S(y, \mathrm{d}_0))$

- WFS$(\mathbf{D}, \exists\wedge+, K) = \Upsilon^{\mathbf{D}}_K$ = subset of WFS$(\mathbf{D}, \exists\wedge+)$ involving only the constants in K.

- The *information content* of database M relative to $\Upsilon^{\mathbf{D}}_K$:

$$\mathsf{Info}\langle M, \Upsilon^{\mathbf{D}}_K\rangle = \{\varphi \in \Upsilon^{\mathbf{D}}_K \mid M \models \varphi\}$$

Example:
$$M = \{R(\mathrm{a}_0, \mathrm{b}_0, \mathrm{c}_0), R(\mathrm{a}_1, \mathrm{b}_2, \mathrm{c}_2), S(\mathrm{c}_0, \mathrm{d}_0), S(\mathrm{c}_1, \mathrm{d}_1), S(\mathrm{c}_4, \mathrm{d}_4), S(\mathrm{c}_2, \mathrm{d}_2)\}$$
$$K = \{\mathrm{a}_0, \mathrm{a}_1, \mathrm{b}_0, \mathrm{b}_1, \mathrm{b}_2, \mathrm{c}_0, \mathrm{c}_1, \mathrm{c}_4, \mathrm{d}_0, \mathrm{d}_1, \mathrm{d}_4\}$$

$\mathsf{Info}\langle M, \Upsilon^{\mathbf{E}_0}_K\rangle =$

$$\{R(\mathrm{a}_0, \mathrm{b}_0, \mathrm{c}_0), S(\mathrm{c}_0, \mathrm{d}_0), S(\mathrm{c}_1, \mathrm{d}_1), S(\mathrm{c}_4, \mathrm{d}_4), (\exists z)(\exists w)(R(\mathrm{a}_1, \mathrm{b}_2, z) \wedge S(z, w))\}^+$$

Distance Measure Based upon Information Content

- The *update difference* for an update (M_1, M_2) on the main schema relative to K:

$$\Delta^{+}\langle (M_1, M_2), \Upsilon_K^{\mathbf{D}} \rangle = \mathsf{Info}\langle M_2, \Upsilon_K^{\mathbf{D}} \rangle \backslash \mathsf{Info}\langle M_1, \Upsilon_K^{\mathbf{D}} \rangle$$

$$\Delta^{-}\langle (M_1, M_2), \Upsilon_K^{\mathbf{D}} \rangle = \mathsf{Info}\langle M_1, \Upsilon_K^{\mathbf{D}} \rangle \backslash \mathsf{Info}\langle M_2, \Upsilon_K^{\mathbf{D}} \rangle$$

$$\Delta\langle (M_1, M_2), \Upsilon_K^{\mathbf{D}} \rangle = \Delta^{+}\langle (M_1, M_2), \Upsilon_K^{\mathbf{D}} \rangle \cup \Delta^{-}\langle (M_1, M_2), \Upsilon_K^{\mathbf{D}} \rangle$$

Distance Measure Based upon Information Content

- The *update difference* for an update (M_1, M_2) on the main schema relative to K:

$$\Delta^{+}\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_K \rangle = \mathsf{Info}\langle M_2, \Upsilon^{\mathbf{D}}_K \rangle \backslash \mathsf{Info}\langle M_1, \Upsilon^{\mathbf{D}}_K \rangle$$
$$\Delta^{-}\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_K \rangle = \mathsf{Info}\langle M_1, \Upsilon^{\mathbf{D}}_K \rangle \backslash \mathsf{Info}\langle M_2, \Upsilon^{\mathbf{D}}_K \rangle$$
$$\Delta\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_K \rangle = \Delta^{+}\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_K \rangle \cup \Delta^{-}\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_K \rangle$$

- The distance is thus not a number but rather the set of Boolean conjunctive queries whose values have changed as a result of the update.
- K consists of all constant symbols which occur in:

  $M_1 =$ Old state of the main schema

  $N_1 =$ Old state of the view schema $\qquad N_2 =$ New state of the view schema

  $\gamma =$ View-definition formula $\qquad \mathsf{Constr}(\mathbf{D}) =$ Constraints on the main schema

  but not any additional new constants occurring only in:

  $M_2 =$ New state of the main schema

Distance Measure Based upon Information Content

- The *update difference* for an update (M_1, M_2) on the main schema relative to K:

$$\Delta^{+}\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_K \rangle = \mathsf{Info}\langle M_2, \Upsilon^{\mathbf{D}}_K \rangle \backslash \mathsf{Info}\langle M_1, \Upsilon^{\mathbf{D}}_K \rangle$$
$$\Delta^{-}\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_K \rangle = \mathsf{Info}\langle M_1, \Upsilon^{\mathbf{D}}_K \rangle \backslash \mathsf{Info}\langle M_2, \Upsilon^{\mathbf{D}}_K \rangle$$
$$\Delta\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_K \rangle = \Delta^{+}\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_K \rangle \cup \Delta^{-}\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_K \rangle$$

- The distance is thus not a number but rather the set of Boolean conjunctive queries whose values have changed as a result of the update.
- K consists of all constant symbols which occur in:

  $M_1 =$ Old state of the main schema

  $N_1 =$ Old state of the view schema $\qquad N_2 =$ New state of the view schema

  $\gamma =$ View-definition formula $\qquad \mathsf{Constr}(\mathbf{D}) =$ Constraints on the main schema

  but not any additional new constants occurring only in:

  $M_2 =$ New state of the main schema

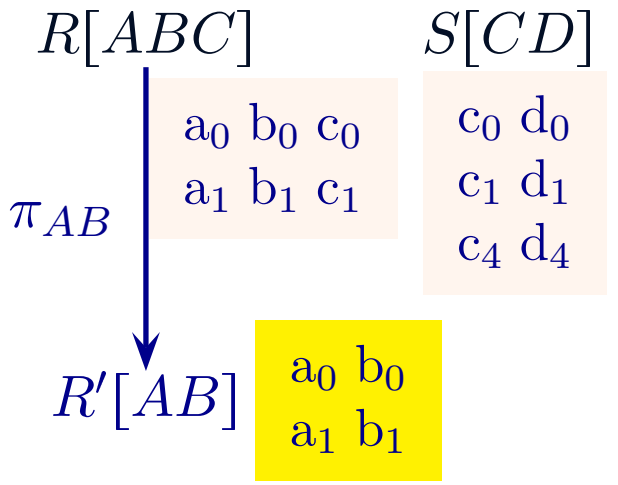
- *Admissible/optimal reflection* of a view update: a reflection to the main schema for which the update difference is minimal/least and which is tuple minimal.

Examples of Information-Based Reflection of View Update

Example:

Main Schema $\mathbf{E}_0$

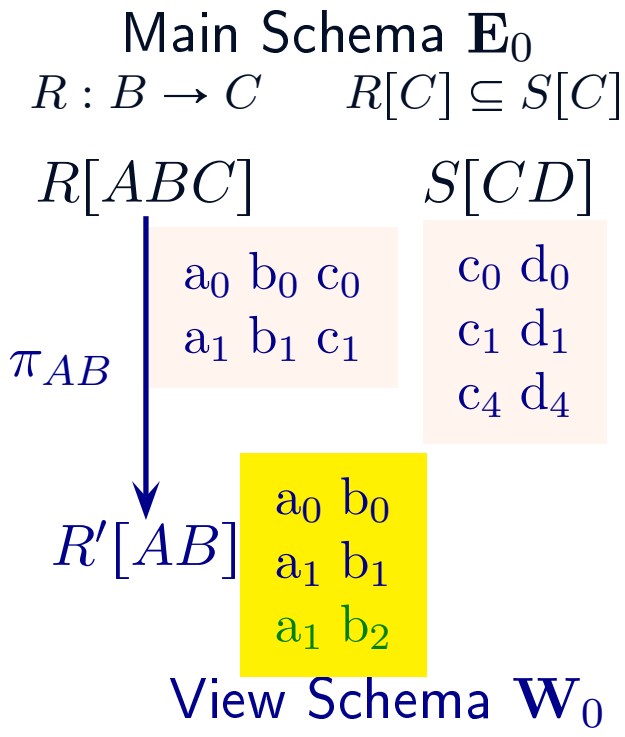
$R : B \to C \qquad R[C] \subseteq S[C]$



View Schema $\mathbf{W}_0$

Examples of Information-Based Reflection of View Update

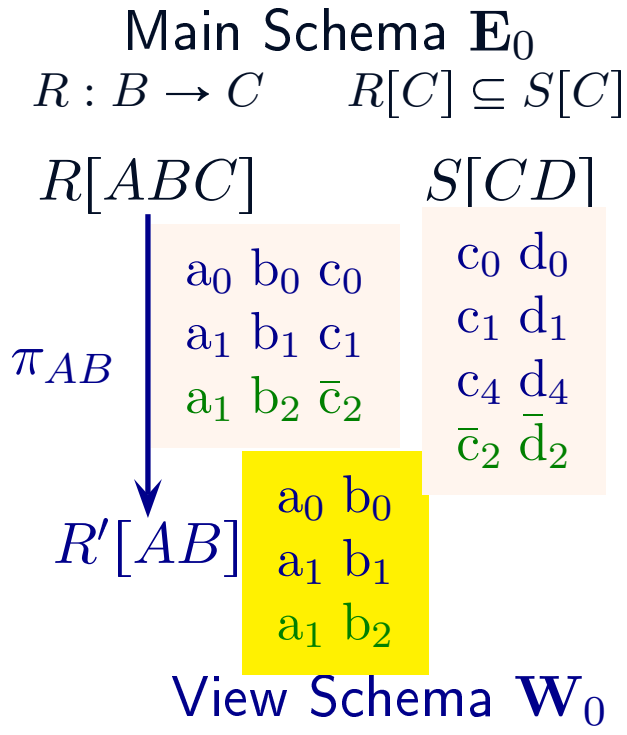
Example: The insertion of $R'(\mathrm{a}_1, \mathrm{b}_2)$ into the view.



Examples of Information-Based Reflection of View Update

Example: The insertion of $R'(\mathrm{a}_1, \mathrm{b}_2)$ into the view.

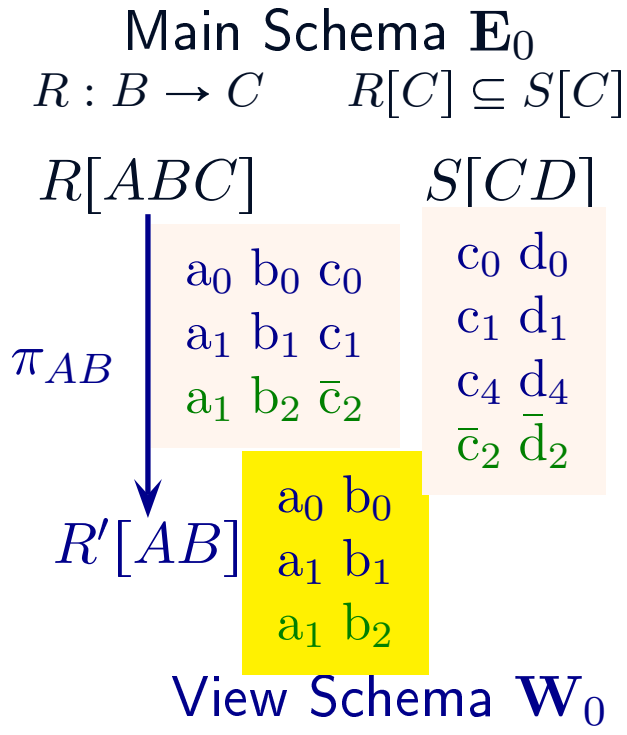
- Optimal for insertions to the main schema.



Examples of Information-Based Reflection of View Update

Example: The insertion of $R'(\mathrm{a}_1, \mathrm{b}_2)$ into the view.

- Optimal for insertions to the main schema.
- $K = \{\mathrm{a}_0, \mathrm{a}_1, \mathrm{b}_0, \mathrm{b}_1, \mathrm{b}_2, \mathrm{c}_0, \mathrm{c}_1, \mathrm{c}_4, \mathrm{d}_0, \mathrm{d}_1, \mathrm{d}_4, \bar{\mathrm{c}}_2, \bar{\mathrm{d}}_2\}$.



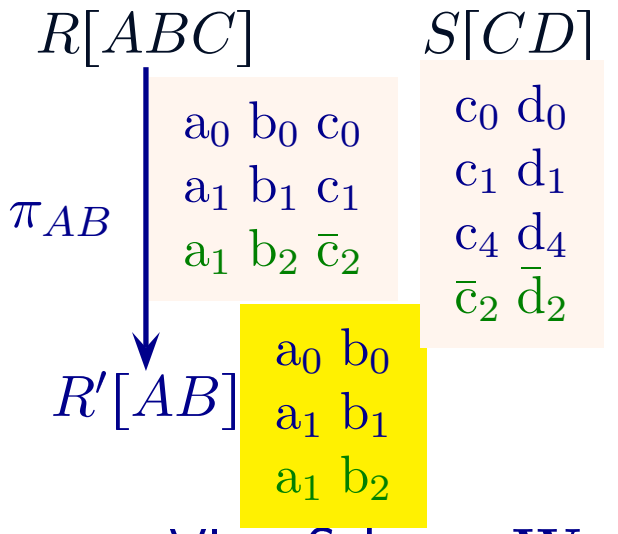
Examples of Information-Based Reflection of View Update

Example: The insertion of $R'(\mathrm{a}_1, \mathrm{b}_2)$ into the view.

- Optimal for insertions to the main schema.
- $K = \{\mathrm{a}_0, \mathrm{a}_1, \mathrm{b}_0, \mathrm{b}_1, \mathrm{b}_2, \mathrm{c}_0, \mathrm{c}_1, \mathrm{c}_4, \mathrm{d}_0, \mathrm{d}_1, \mathrm{d}_4, \bar{\mathrm{c}}_2, \bar{\mathrm{d}}_2\}$.
- $\Delta\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_K \rangle = \Delta^+\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_K \rangle =$
  $(M_1 \cup \{(\exists z)(\exists w)(R(\mathrm{a}_1, \mathrm{b}_2, z) \wedge S(z, w))\})^+$.

Main Schema $\mathbf{E}_0$

$R : B \to C$ $\quad$ $R[C] \subseteq S[C]$

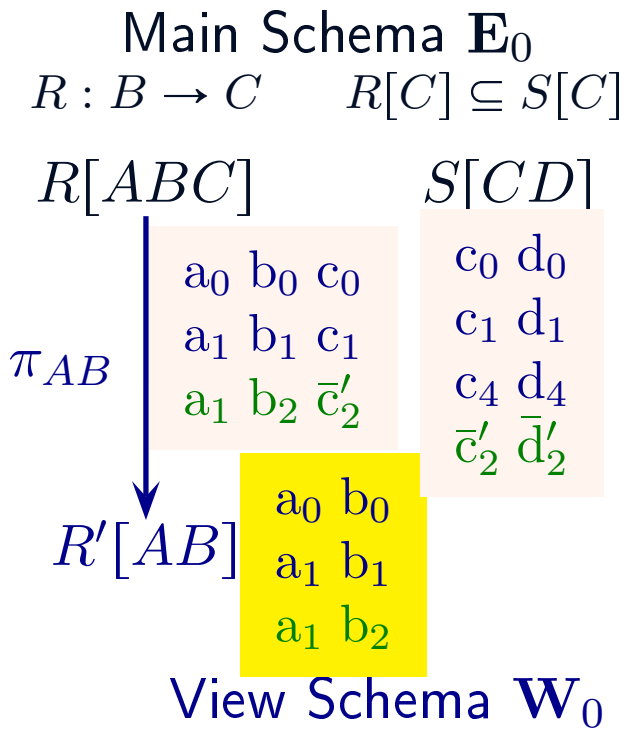


View Schema $\mathbf{W}_0$

Examples of Information-Based Reflection of View Update

Example: The insertion of $R'(\mathrm{a}_1, \mathrm{b}_2)$ into the view.

- Optimal for insertions to the main schema.
- $K = \{\mathrm{a}_0, \mathrm{a}_1, \mathrm{b}_0, \mathrm{b}_1, \mathrm{b}_2, \mathrm{c}_0, \mathrm{c}_1, \mathrm{c}_4, \mathrm{d}_0, \mathrm{d}_1, \mathrm{d}_4, \bar{\mathrm{c}}_2, \bar{\mathrm{d}}_2\}$.
- $\Delta\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle = \Delta^+\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle =$
  $(M_1 \cup \{(\exists z)(\exists w)(R(\mathrm{a}_1, \mathrm{b}_2, z) \wedge S(z, w))\})^+$.
- Changing constant names $\rightsquigarrow$ another optimal solution.

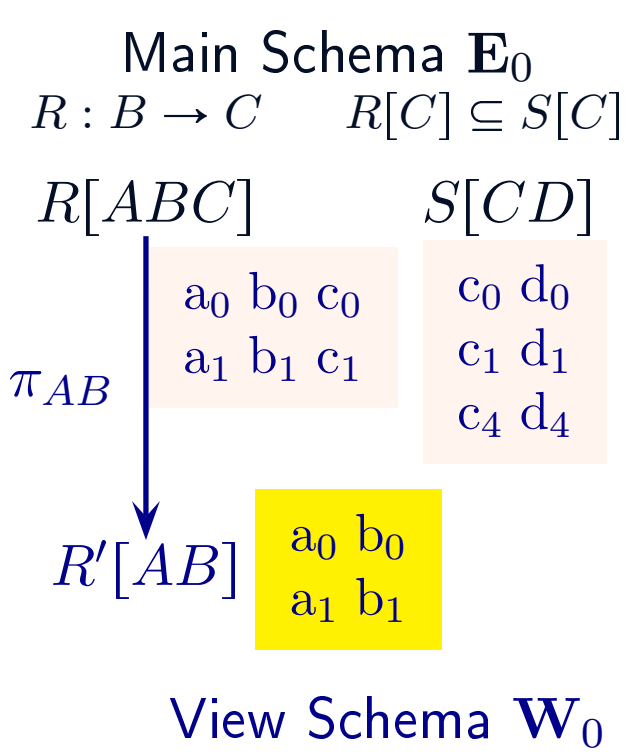


Examples of Information-Based Reflection of View Update

Example: The insertion of $R'(\mathrm{a}_1, \mathrm{b}_2)$ into the view.

- Optimal for insertions to the main schema.
- $K = \{\mathrm{a}_0, \mathrm{a}_1, \mathrm{b}_0, \mathrm{b}_1, \mathrm{b}_2, \mathrm{c}_0, \mathrm{c}_1, \mathrm{c}_4, \mathrm{d}_0, \mathrm{d}_1, \mathrm{d}_4, \bar{\mathrm{c}}_2, \bar{\mathrm{d}}_2\}$.
- $\Delta\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle = \Delta^+\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle =$
  $(M_1 \cup \{(\exists z)(\exists w)(R(\mathrm{a}_1, \mathrm{b}_2, z) \wedge S(z, w))\})^+$.
- Changing constant names $\rightsquigarrow$ another optimal solution.

Example:

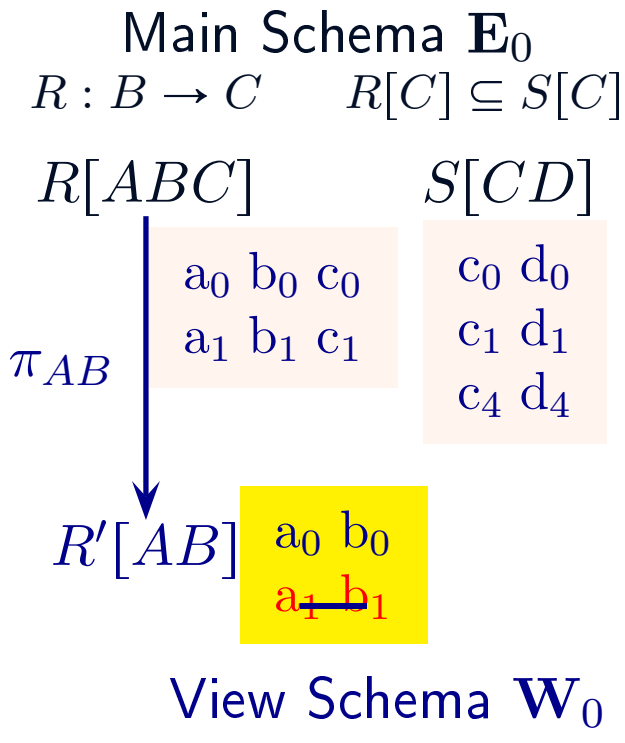


Examples of Information-Based Reflection of View Update

Example: The insertion of $R'(\mathrm{a}_1, \mathrm{b}_2)$ into the view.

- Optimal for insertions to the main schema.
- $K = \{\mathrm{a}_0, \mathrm{a}_1, \mathrm{b}_0, \mathrm{b}_1, \mathrm{b}_2, \mathrm{c}_0, \mathrm{c}_1, \mathrm{c}_4, \mathrm{d}_0, \mathrm{d}_1, \mathrm{d}_4, \bar{\mathrm{c}}_2, \bar{\mathrm{d}}_2\}$.
- $\Delta\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_K \rangle = \Delta^+\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_K \rangle = (M_1 \cup \{(\exists z)(\exists w)(R(\mathrm{a}_1, \mathrm{b}_2, z) \wedge S(z, w))\})^+$.
- Changing constant names $\rightsquigarrow$ another optimal solution.

Example: The deletion of $R'(\mathrm{a}_1, \mathrm{b}_1)$ from the view.



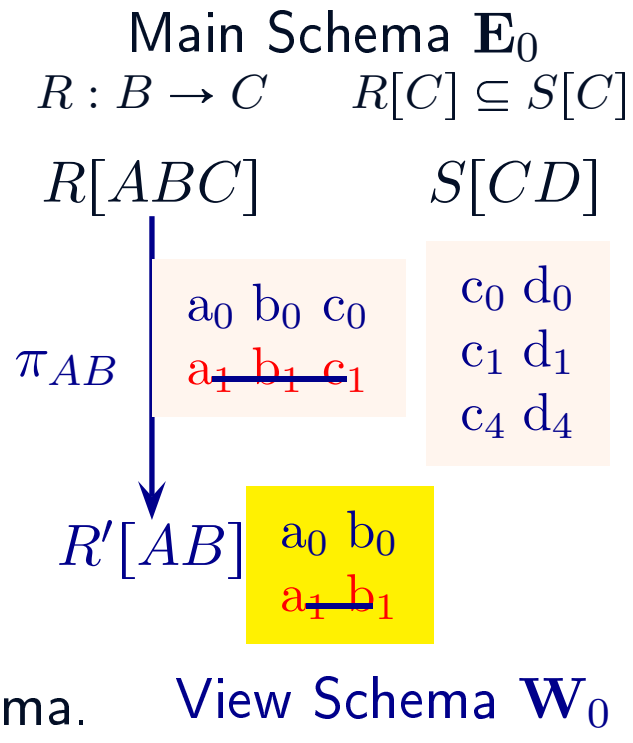
Examples of Information-Based Reflection of View Update

Example: The insertion of $R'(\mathrm{a}_1, \mathrm{b}_2)$ into the view.

- Optimal for insertions to the main schema.
- $K = \{\mathrm{a}_0, \mathrm{a}_1, \mathrm{b}_0, \mathrm{b}_1, \mathrm{b}_2, \mathrm{c}_0, \mathrm{c}_1, \mathrm{c}_4, \mathrm{d}_0, \mathrm{d}_1, \mathrm{d}_4, \bar{\mathrm{c}}_2, \bar{\mathrm{d}}_2\}$.
- $\Delta\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_K \rangle = \Delta^+\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_K \rangle = (M_1 \cup \{(\exists z)(\exists w)(R(\mathrm{a}_1, \mathrm{b}_2, z) \wedge S(z, w))\})^+$.
- Changing constant names $\rightsquigarrow$ another optimal solution.

Example: The deletion of $R'(\mathrm{a}_1, \mathrm{b}_1)$ from the view.

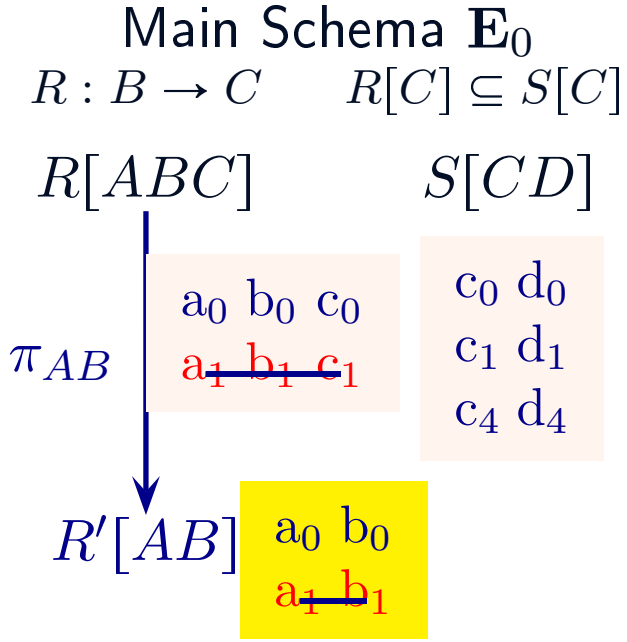
- The solution shown is optimal for deletions to the main schema.



Examples of Information-Based Reflection of View Update

Example: The insertion of $R'(\mathrm{a}_1, \mathrm{b}_2)$ into the view.

- Optimal for insertions to the main schema.
- $K = \{\mathrm{a}_0, \mathrm{a}_1, \mathrm{b}_0, \mathrm{b}_1, \mathrm{b}_2, \mathrm{c}_0, \mathrm{c}_1, \mathrm{c}_4, \mathrm{d}_0, \mathrm{d}_1, \mathrm{d}_4, \bar{\mathrm{c}}_2, \bar{\mathrm{d}}_2\}$.
- $\Delta\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_K \rangle = \Delta^+\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_K \rangle = (M_1 \cup \{(\exists z)(\exists w)(R(\mathrm{a}_1, \mathrm{b}_2, z) \wedge S(z, w))\})^+$.
- Changing constant names $\rightsquigarrow$ another optimal solution.



Example: The deletion of $R'(\mathrm{a}_1, \mathrm{b}_1)$ from the view.

- The solution shown is optimal for deletions to the main schema.
- $K = \{\mathrm{a}_0, \mathrm{a}_1, \mathrm{b}_0, \mathrm{b}_1, \mathrm{b}_2, \mathrm{c}_0, \mathrm{c}_1, \mathrm{c}_4, \mathrm{d}_0, \mathrm{d}_1, \mathrm{d}_4\}$.

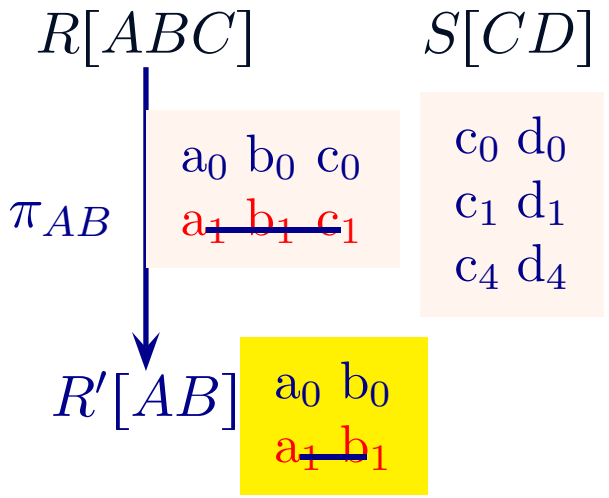
Examples of Information-Based Reflection of View Update

Example: The insertion of $R'(\mathrm{a}_1, \mathrm{b}_2)$ into the view.

- Optimal for insertions to the main schema.
- $K = \{\mathrm{a}_0, \mathrm{a}_1, \mathrm{b}_0, \mathrm{b}_1, \mathrm{b}_2, \mathrm{c}_0, \mathrm{c}_1, \mathrm{c}_4, \mathrm{d}_0, \mathrm{d}_1, \mathrm{d}_4, \bar{\mathrm{c}}_2, \bar{\mathrm{d}}_2\}$.
- $\Delta\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle = \Delta^+\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle =$
  $(M_1 \cup \{(\exists z)(\exists w)(R(\mathrm{a}_1, \mathrm{b}_2, z) \wedge S(z, w))\})^+$.
- Changing constant names $\rightsquigarrow$ another optimal solution.

Main Schema $\mathbf{E}_0$

$R : B \to C$ $\quad R[C] \subseteq S[C]$



View Schema $\mathbf{W}_0$

Example: The deletion of $R'(\mathrm{a}_1, \mathrm{b}_1)$ from the view.

- The solution shown is optimal for deletions to the main schema.
- $K = \{\mathrm{a}_0, \mathrm{a}_1, \mathrm{b}_0, \mathrm{b}_1, \mathrm{b}_2, \mathrm{c}_0, \mathrm{c}_1, \mathrm{c}_4, \mathrm{d}_0, \mathrm{d}_1, \mathrm{d}_4\}$.
- An explicit representation of $\Delta\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle = \Delta^-\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle$ is complex.

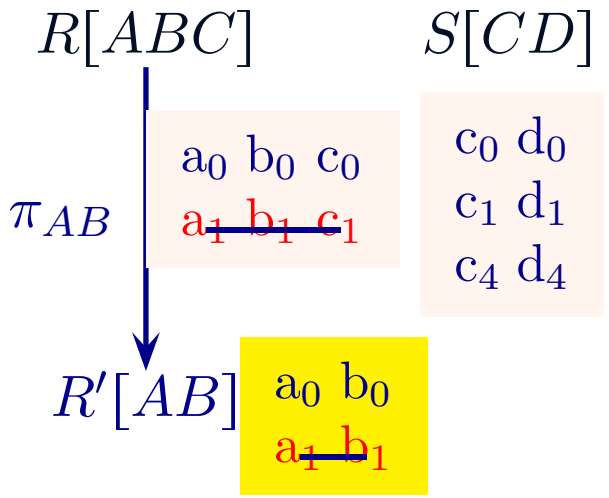
Examples of Information-Based Reflection of View Update

Example: The insertion of $R'(\mathrm{a}_1, \mathrm{b}_2)$ into the view.

- Optimal for insertions to the main schema.
- $K = \{\mathrm{a}_0, \mathrm{a}_1, \mathrm{b}_0, \mathrm{b}_1, \mathrm{b}_2, \mathrm{c}_0, \mathrm{c}_1, \mathrm{c}_4, \mathrm{d}_0, \mathrm{d}_1, \mathrm{d}_4, \bar{\mathrm{c}}_2, \bar{\mathrm{d}}_2\}$.
- $\Delta\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle = \Delta^+\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle =$
  $(M_1 \cup \{(\exists z)(\exists w)(R(\mathrm{a}_1, \mathrm{b}_2, z) \wedge S(z, w))\})^+$.
- Changing constant names $\rightsquigarrow$ another optimal solution.

Main Schema $\mathbf{E}_0$

$R : B \to C$ $\quad R[C] \subseteq S[C]$



View Schema $\mathbf{W}_0$

Example: The deletion of $R'(\mathrm{a}_1, \mathrm{b}_1)$ from the view.

- The solution shown is optimal for deletions to the main schema.
- $K = \{\mathrm{a}_0, \mathrm{a}_1, \mathrm{b}_0, \mathrm{b}_1, \mathrm{b}_2, \mathrm{c}_0, \mathrm{c}_1, \mathrm{c}_4, \mathrm{d}_0, \mathrm{d}_1, \mathrm{d}_4\}$.
- An explicit representation of $\Delta\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle = \Delta^-\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle$ is complex.
- However, it is easy to see that $\mathrm{Info}\langle M_2, \Upsilon^{\mathbf{D}}_K\rangle$ is *preserved* and nothing more.

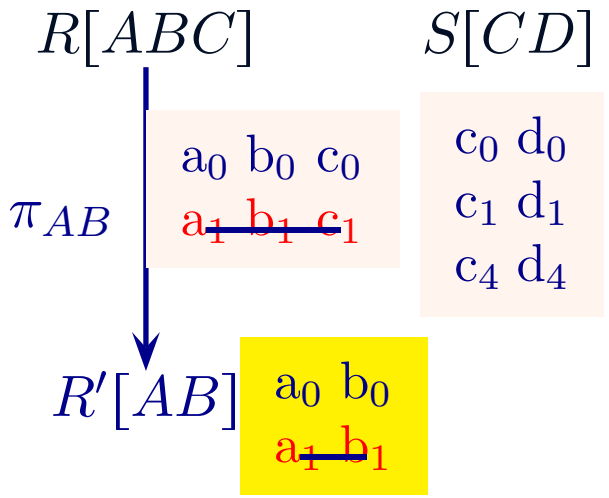
Examples of Information-Based Reflection of View Update

Example: The insertion of $R'(\mathrm{a}_1, \mathrm{b}_2)$ into the view.

- Optimal for insertions to the main schema.
- $K = \{\mathrm{a}_0, \mathrm{a}_1, \mathrm{b}_0, \mathrm{b}_1, \mathrm{b}_2, \mathrm{c}_0, \mathrm{c}_1, \mathrm{c}_4, \mathrm{d}_0, \mathrm{d}_1, \mathrm{d}_4, \bar{\mathrm{c}}_2, \bar{\mathrm{d}}_2\}$.
- $\Delta\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle = \Delta^+\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle =$
  $(M_1 \cup \{(\exists z)(\exists w)(R(\mathrm{a}_1, \mathrm{b}_2, z) \wedge S(z, w))\})^+$.
- Changing constant names $\rightsquigarrow$ another optimal solution.

Main Schema $\mathbf{E}_0$

$R : B \to C$ $\quad R[C] \subseteq S[C]$



View Schema $\mathbf{W}_0$

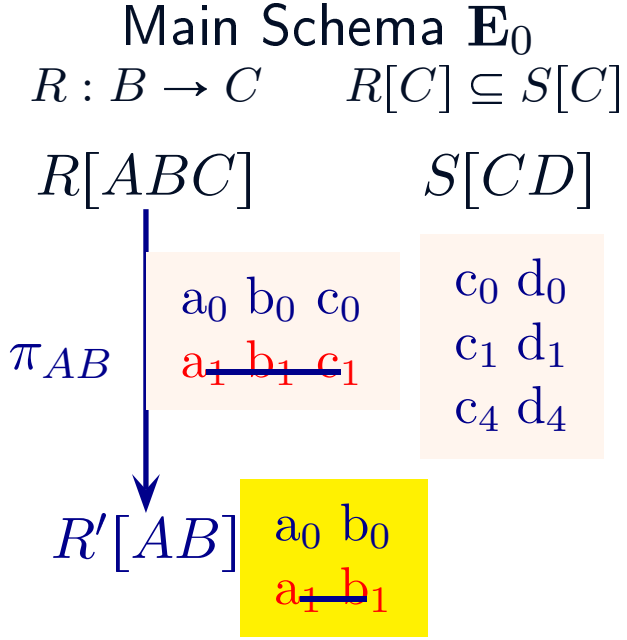
Example: The deletion of $R'(\mathrm{a}_1, \mathrm{b}_1)$ from the view.

- The solution shown is optimal for deletions to the main schema.
- $K = \{\mathrm{a}_0, \mathrm{a}_1, \mathrm{b}_0, \mathrm{b}_1, \mathrm{b}_2, \mathrm{c}_0, \mathrm{c}_1, \mathrm{c}_4, \mathrm{d}_0, \mathrm{d}_1, \mathrm{d}_4\}$.
- An explicit representation of $\Delta\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle = \Delta^-\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle$ is complex.
- However, it is easy to see that $\mathrm{Info}\langle M_2, \Upsilon^{\mathbf{D}}_K\rangle$ is *preserved* and nothing more.
- A general theory of optimality for *monotonic* reflections: [Hegner 2008-2009].

Examples of Information-Based Reflection of View Update

Example: The insertion of $R'(\mathrm{a}_1, \mathrm{b}_2)$ into the view.

- Optimal for insertions to the main schema.
- $K = \{\mathrm{a}_0, \mathrm{a}_1, \mathrm{b}_0, \mathrm{b}_1, \mathrm{b}_2, \mathrm{c}_0, \mathrm{c}_1, \mathrm{c}_4, \mathrm{d}_0, \mathrm{d}_1, \mathrm{d}_4, \bar{\mathrm{c}}_2, \bar{\mathrm{d}}_2\}$.
- $\Delta\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle = \Delta^+\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle =$
  $(M_1 \cup \{(\exists z)(\exists w)(R(\mathrm{a}_1, \mathrm{b}_2, z) \wedge S(z, w))\})^+$.
- Changing constant names $\rightsquigarrow$ another optimal solution.



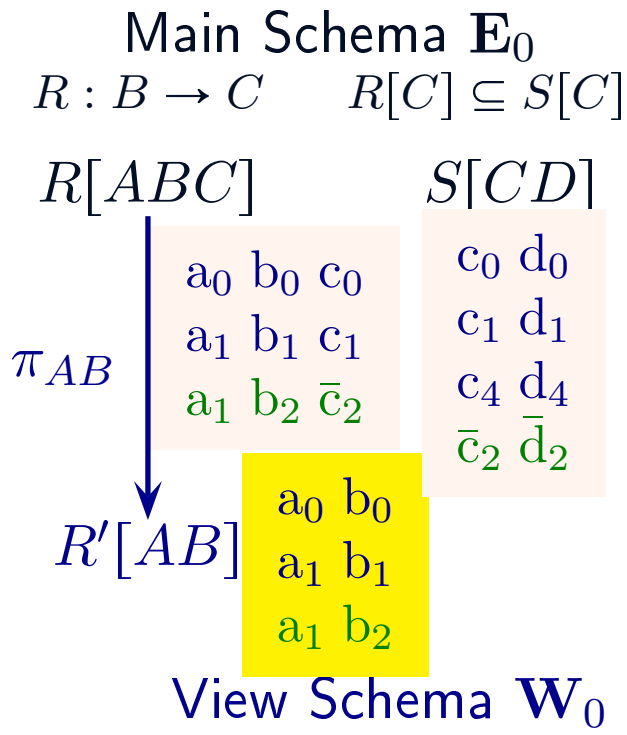
Example: The deletion of $R'(\mathrm{a}_1, \mathrm{b}_1)$ from the view.

- The solution shown is optimal for deletions to the main schema.
- $K = \{\mathrm{a}_0, \mathrm{a}_1, \mathrm{b}_0, \mathrm{b}_1, \mathrm{b}_2, \mathrm{c}_0, \mathrm{c}_1, \mathrm{c}_4, \mathrm{d}_0, \mathrm{d}_1, \mathrm{d}_4\}$.
- An explicit representation of $\Delta\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle = \Delta^-\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle$ is complex.
- However, it is easy to see that $\mathrm{Info}\langle M_2, \Upsilon^{\mathbf{D}}_K\rangle$ is *preserved* and nothing more.
- A general theory of optimality for *monotonic* reflections: [Hegner 2008-2009].

Goal of this work: Extend this theory to non-monotonic reflections (involving both insertion and deletion).

The Problem of Non-Monotonic Updates and Collateral Change

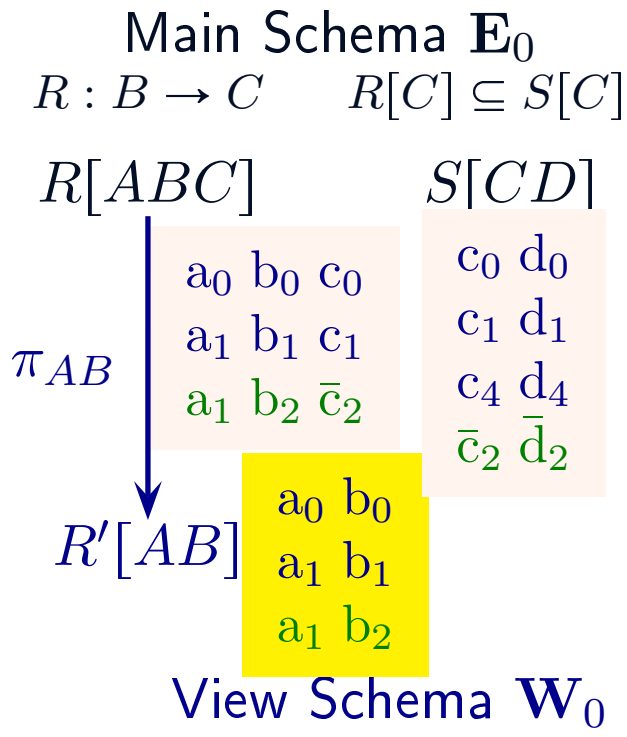
Example: Consider again the insertion of $(\mathrm{a}_1, \mathrm{b}_2)$ into the view with the insertion-optimal solution.



The Problem of Non-Monotonic Updates and Collateral Change

Example: Consider again the insertion of $(\mathrm{a}_1, \mathrm{b}_2)$ into the view with the insertion-optimal solution.

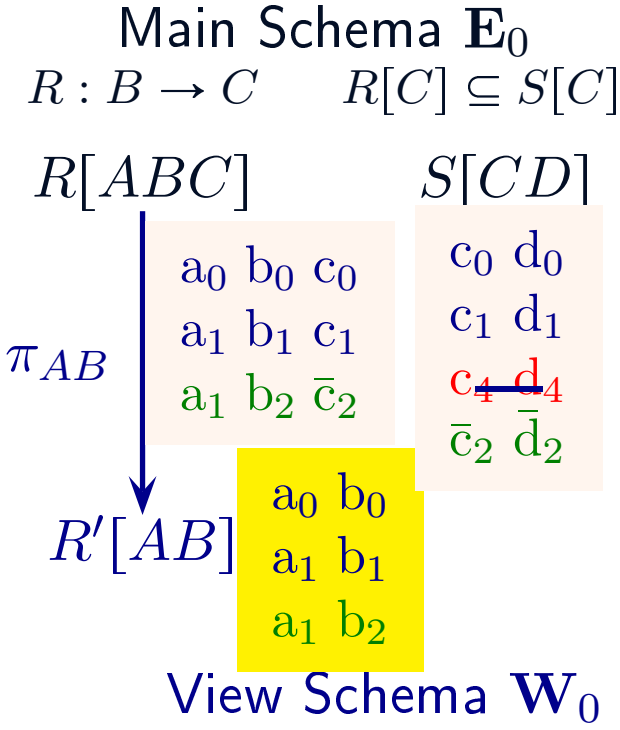
- $\varphi = (\exists z)(\exists w)(R(\mathrm{a}_1, \mathrm{b}_2, z) \wedge S(z, w)) \wedge S(\mathrm{c}_4, \mathrm{d}_4)$
$\in \Delta^{+}\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_{K} \rangle$.



The Problem of Non-Monotonic Updates and Collateral Change

Example: Consider again the insertion of $(\mathrm{a}_1, \mathrm{b}_2)$ into the view with the insertion-optimal solution.

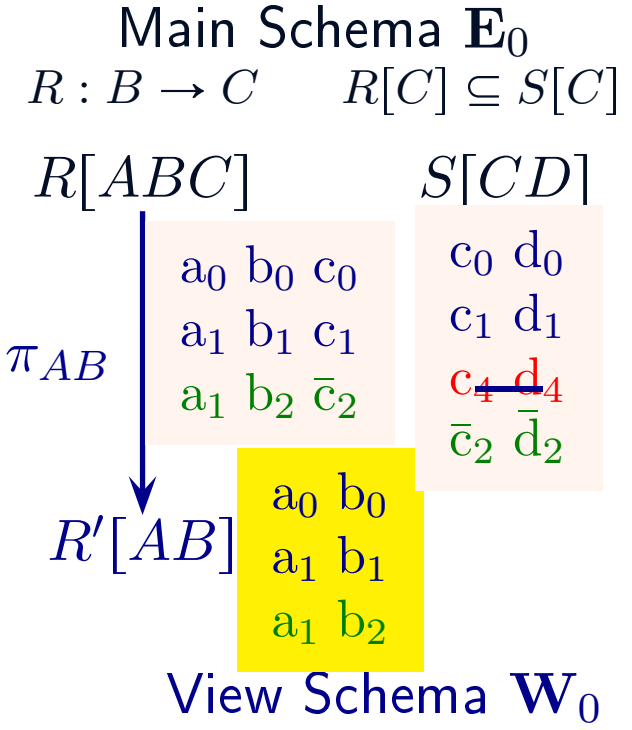
- $\varphi = (\exists z)(\exists w)(R(\mathrm{a}_1, \mathrm{b}_2, z) \wedge S(z, w)) \wedge S(\mathrm{c}_4, \mathrm{d}_4)$ $\in \Delta^{+}\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_{K} \rangle$.
- If $S(\mathrm{c}_4, \mathrm{d}_4)$ is deleted, φ will not be added.



The Problem of Non-Monotonic Updates and Collateral Change

Example: Consider again the insertion of $(\mathrm{a}_1, \mathrm{b}_2)$ into the view with the insertion-optimal solution.

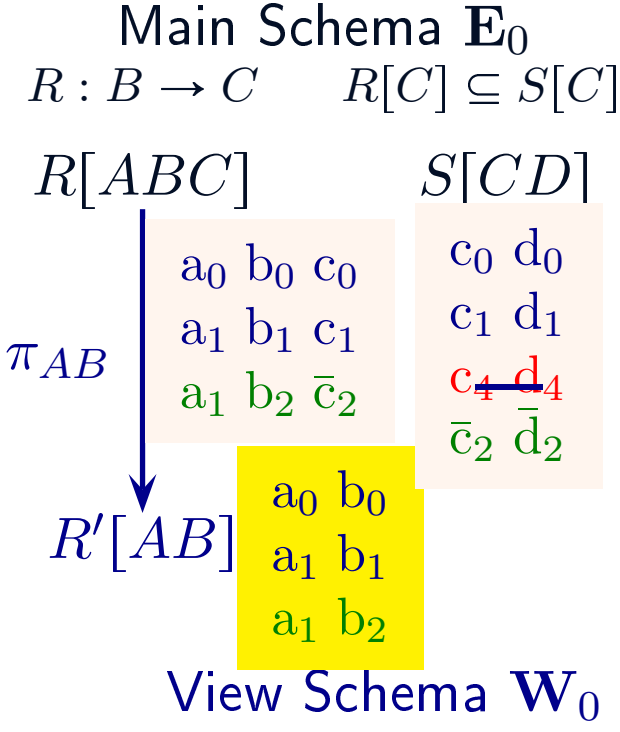
- $\varphi = (\exists z)(\exists w)(R(\mathrm{a}_1, \mathrm{b}_2, z) \wedge S(z, w)) \wedge S(\mathrm{c}_4, \mathrm{d}_4)$ $\in \Delta^+\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_K \rangle$.
- If $S(\mathrm{c}_4, \mathrm{d}_4)$ is deleted, φ will not be added.
- But deleting $S(\mathrm{c}_4, \mathrm{d}_4)$ does not make the solution better in any reasonable way.



The Problem of Non-Monotonic Updates and Collateral Change

Example: Consider again the insertion of $(\mathrm{a}_1, \mathrm{b}_2)$ into the view with the insertion-optimal solution.

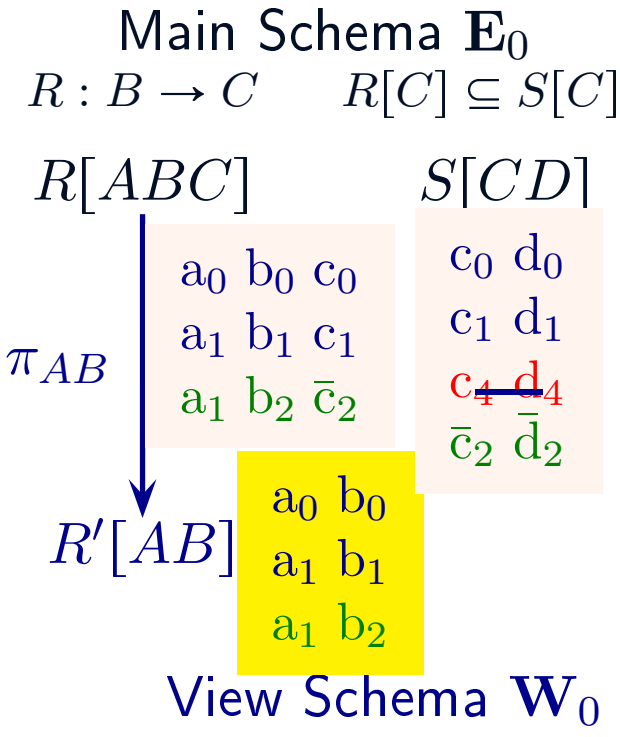
- $\varphi = (\exists z)(\exists w)(R(\mathrm{a}_1, \mathrm{b}_2, z) \wedge S(z, w)) \wedge S(\mathrm{c}_4, \mathrm{d}_4)$
  $\in \Delta^{+}\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_{K} \rangle$.
- If $S(\mathrm{c}_4, \mathrm{d}_4)$ is deleted, φ will not be added.
- But deleting $S(\mathrm{c}_4, \mathrm{d}_4)$ does not make the solution better in any reasonable way.
- φ is called a *collateral change*.



The Problem of Non-Monotonic Updates and Collateral Change

Example: Consider again the insertion of $(\mathrm{a}_1, \mathrm{b}_2)$ into the view with the insertion-optimal solution.

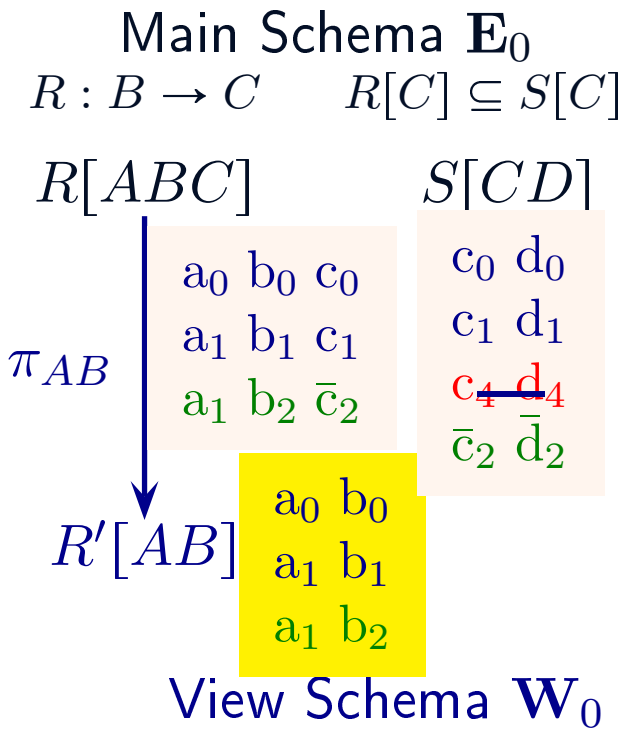
- $\varphi = (\exists z)(\exists w)(R(\mathrm{a}_1, \mathrm{b}_2, z) \wedge S(z, w)) \wedge S(\mathrm{c}_4, \mathrm{d}_4)$
  $\in \Delta^+\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_K \rangle$.
- If $S(\mathrm{c}_4, \mathrm{d}_4)$ is deleted, φ will not be added.
- But deleting $S(\mathrm{c}_4, \mathrm{d}_4)$ does not make the solution better in any reasonable way.
- φ is called a *collateral change*.



- With monotonic view updates (insertions and deletions), collateral change can be avoided by requiring that the reflection be of the same type as the view update.

The Problem of Non-Monotonic Updates and Collateral Change

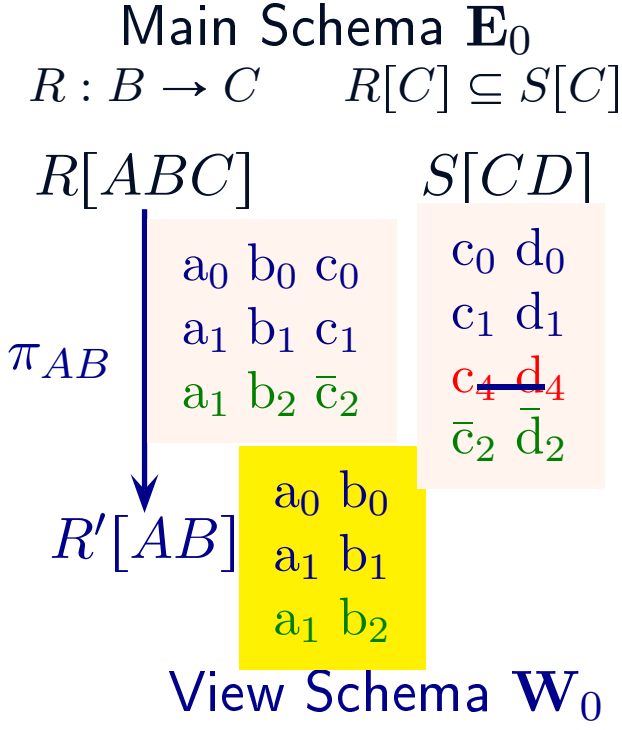
Example: Consider again the insertion of $(\mathrm{a}_1, \mathrm{b}_2)$ into the view with the insertion-optimal solution.



- $\varphi = (\exists z)(\exists w)(R(\mathrm{a}_1, \mathrm{b}_2, z) \wedge S(z, w)) \wedge S(\mathrm{c}_4, \mathrm{d}_4)$
  $\in \Delta^+\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_K \rangle$.
- If $S(\mathrm{c}_4, \mathrm{d}_4)$ is deleted, φ will not be added.
- But deleting $S(\mathrm{c}_4, \mathrm{d}_4)$ does not make the solution better in any reasonable way.
- φ is called a *collateral change*.
- With monotonic view updates (insertions and deletions), collateral change can be avoided by requiring that the reflection be of the same type as the view update.
- For view updates which are not monotonic, a more general technique for avoiding collateral change is necessary.

The Problem of Non-Monotonic Updates and Collateral Change

Example: Consider again the insertion of $(\mathrm{a}_1, \mathrm{b}_2)$ into the view with the insertion-optimal solution.



- $\varphi = (\exists z)(\exists w)(R(\mathrm{a}_1, \mathrm{b}_2, z) \wedge S(z, w)) \wedge S(\mathrm{c}_4, \mathrm{d}_4)$
  $\in \Delta^+\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_K \rangle$.
- If $S(\mathrm{c}_4, \mathrm{d}_4)$ is deleted, φ will not be added.
- But deleting $S(\mathrm{c}_4, \mathrm{d}_4)$ does not make the solution better in any reasonable way.
- φ is called a *collateral change*.
- With monotonic view updates (insertions and deletions), collateral change can be avoided by requiring that the reflection be of the same type as the view update.
- For view updates which are not monotonic, a more general technique for avoiding collateral change is necessary.
- The approach is to characterize $\Delta^+\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_K \rangle$ and $\Delta^-\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_K \rangle$ via *generators*, and then place minimal/least constraints on these generators.

Update Generators

Idea: A *generator* for the update (M_1, M_2) on the schema $\mathbf{D}$ is a pair $\langle G_+, G_{\not+} \rangle$ with the following properties:

Update Generators

Idea: A *generator* for the update (M_1, M_2) on the schema $\mathbf{D}$ is a pair $\langle G_+, G_{\not+} \rangle$ with the following properties:

1. $G_+ \subseteq \Delta^+ \langle (M_1, M_2), \Upsilon^{\mathbf{D}}_K \rangle$ generates the added information.

Update Generators

Idea: A *generator* for the update (M_1, M_2) on the schema $\mathbf{D}$ is a pair $\langle G_+, G_{\not}\rangle$ with the following properties:

1. $G_+ \subseteq \Delta^+\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle$ generates the added information.
2. $G_{\not} \subseteq \mathsf{Info}\langle M_1, \Upsilon^{\mathbf{D}}_K\rangle \backslash \Delta^-\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle$ generates the retained information.

Update Generators

Idea: A *generator* for the update (M_1, M_2) on the schema $\mathbf{D}$ is a pair $\langle G_+, G_{\neq} \rangle$ with the following properties:

1. $G_+ \subseteq \Delta^+\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_K \rangle$ generates the added information.
2. $G_{\neq} \subseteq \mathsf{Info}\langle M_1, \Upsilon^{\mathbf{D}}_K \rangle \backslash \Delta^-\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_K \rangle$ generates the retained information.
3. $G_+ \cup G_{\neq} \equiv_{\mathbf{D}} M_2$ ($G_+ \cup G_{\neq}$ defines the new state).

Update Generators

Idea: A *generator* for the update (M_1, M_2) on the schema $\mathbf{D}$ is a pair $\langle G_+, G_{\not}\rangle$ with the following properties:

1. $G_+ \subseteq \Delta^+\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle$ generates the added information.
2. $G_{\not} \subseteq \mathsf{Info}\langle M_1, \Upsilon^{\mathbf{D}}_K\rangle \backslash \Delta^-\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle$ generates the retained information.
3. $G_+ \cup G_{\not} \equiv_{\mathbf{D}} M_2$ ($G_+ \cup G_{\not}$ defines the new state).

Ordering: $\langle G^1_+, G^1_{\not}\rangle \sqsubseteq_{\mathbf{D}} \langle G^2_+, G^2_{\not}\rangle$ iff

Update Generators

Idea: A *generator* for the update (M_1, M_2) on the schema $\mathbf{D}$ is a pair $\langle G_+, G_{\neq} \rangle$ with the following properties:

1. $G_+ \subseteq \Delta^+\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_K \rangle$ generates the added information.
2. $G_{\neq} \subseteq \mathsf{Info}\langle M_1, \Upsilon^{\mathbf{D}}_K \rangle \backslash \Delta^-\langle (M_1, M_2), \Upsilon^{\mathbf{D}}_K \rangle$ generates the retained information.
3. $G_+ \cup G_{\neq} \equiv_{\mathbf{D}} M_2$ ($G_+ \cup G_{\neq}$ defines the new state).

Ordering: $\langle G^1_+, G^1_{\neq} \rangle \sqsubseteq_{\mathbf{D}} \langle G^2_+, G^2_{\neq} \rangle$ iff

1. $G^2_+ \models_{\mathbf{D}} G^1_+$ (G^1_+ is weaker than G^2_+ — want to minimize G_+).

Update Generators

Idea: A *generator* for the update (M_1, M_2) on the schema $\mathbf{D}$ is a pair $\langle G_+, G_{\not}\rangle$ with the following properties:

1. $G_+ \subseteq \Delta^+\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle$ generates the added information.
2. $G_{\not} \subseteq \mathsf{Info}\langle M_1, \Upsilon^{\mathbf{D}}_K\rangle \backslash \Delta^-\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle$ generates the retained information.
3. $G_+ \cup G_{\not} \equiv_{\mathbf{D}} M_2$ ($G_+ \cup G_{\not}$ defines the new state).

Ordering: $\langle G^1_+, G^1_{\not}\rangle \sqsubseteq_{\mathbf{D}} \langle G^2_+, G^2_{\not}\rangle$ iff

1. $G^2_+ \models_{\mathbf{D}} G^1_+$ (G^1_+ is weaker than G^2_+ — want to minimize G_+).
2. $G^1_{\not} \models_{\mathbf{D}} G^2_{\not}$ ($G^2_{\not}$ is stronger than $G^1_{\not}$ — want to maximize $G_{\not}$).

Update Generators

Idea: A *generator* for the update (M_1, M_2) on the schema $\mathbf{D}$ is a pair $\langle G_+, G_{\not}\rangle$ with the following properties:

1. $G_+ \subseteq \Delta^+\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle$ generates the added information.
2. $G_{\not} \subseteq \mathrm{Info}\langle M_1, \Upsilon^{\mathbf{D}}_K\rangle \backslash \Delta^-\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle$ generates the retained information.
3. $G_+ \cup G_{\not} \equiv_{\mathbf{D}} M_2$ ($G_+ \cup G_{\not}$ defines the new state).

Ordering: $\langle G^1_+, G^1_{\not}\rangle \sqsubseteq_{\mathbf{D}} \langle G^2_+, G^2_{\not}\rangle$ iff

1. $G^2_+ \models_{\mathbf{D}} G^1_+$ (G^1_+ is weaker than G^2_+ — want to minimize G_+).
2. $G^1_{\not} \models_{\mathbf{D}} G^2_{\not}$ ($G^2_{\not}$ is stronger than $G^1_{\not}$ — want to maximize $G_{\not}$).

- Roughly, $\models_{\mathbf{D}}$ is semantic entailment in the context of the constraints $\mathrm{Constr}(\mathbf{D})$ of $\mathbf{D}$.
- Thus $\sqsubseteq_{\mathbf{D}}$ is a preorder (and a partial order on the associated equivalence classes).

Update Generators

Idea: A *generator* for the update (M_1, M_2) on the schema $\mathbf{D}$ is a pair $\langle G_+, G_{\not}\rangle$ with the following properties:

1. $G_+ \subseteq \Delta^+\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle$ generates the added information.
2. $G_{\not} \subseteq \mathrm{Info}\langle M_1, \Upsilon^{\mathbf{D}}_K\rangle \backslash \Delta^-\langle(M_1, M_2), \Upsilon^{\mathbf{D}}_K\rangle$ generates the retained information.
3. $G_+ \cup G_{\not} \equiv_{\mathbf{D}} M_2$ ($G_+ \cup G_{\not}$ defines the new state).

Ordering: $\langle G^1_+, G^1_{\not}\rangle \sqsubseteq_{\mathbf{D}} \langle G^2_+, G^2_{\not}\rangle$ iff

1. $G^2_+ \models_{\mathbf{D}} G^1_+$ (G^1_+ is weaker than G^2_+ — want to minimize G_+).
2. $G^1_{\not} \models_{\mathbf{D}} G^2_{\not}$ ($G^2_{\not}$ is stronger than $G^1_{\not}$ — want to maximize $G_{\not}$).

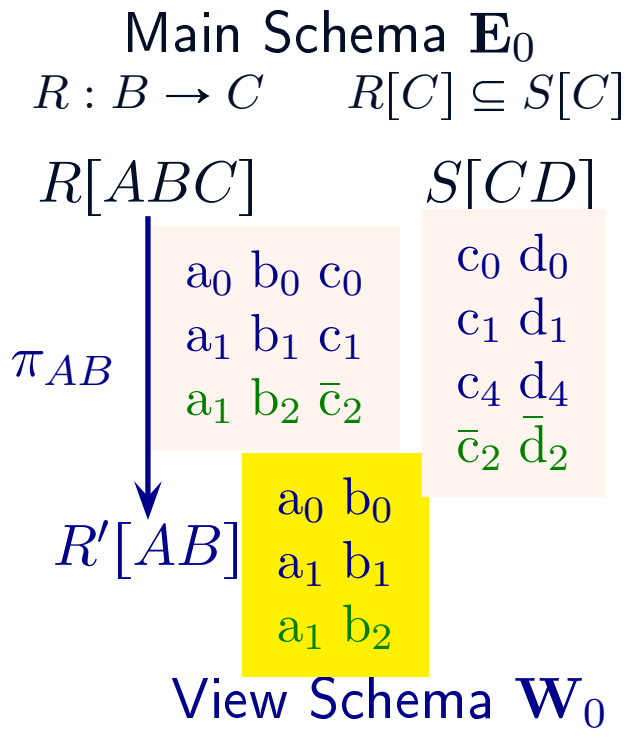
- Roughly, $\models_{\mathbf{D}}$ is semantic entailment in the context of the constraints $\mathrm{Constr}(\mathbf{D})$ of $\mathbf{D}$.
- Thus $\sqsubseteq_{\mathbf{D}}$ is a preorder (and a partial order on the associated equivalence classes).

- *Admissibility* and *optimality* are defined in terms of this ordering.
  - *Admissible* = minimal in the ordering.
  - *Optimal* = least in the ordering.

Example of Update Generator

Example: For the insertion-only reflection of the insertion of $(\mathrm{a}_1, \mathrm{b}_2)$ into the view:

$$G_+ = \{(\exists z)(\exists w)(R(\mathrm{a}_1, \mathrm{b}_2, z)\}$$
$$G_{\not\in} = M_1 \quad \text{(original state of } \mathbf{E}_0\text{)}$$



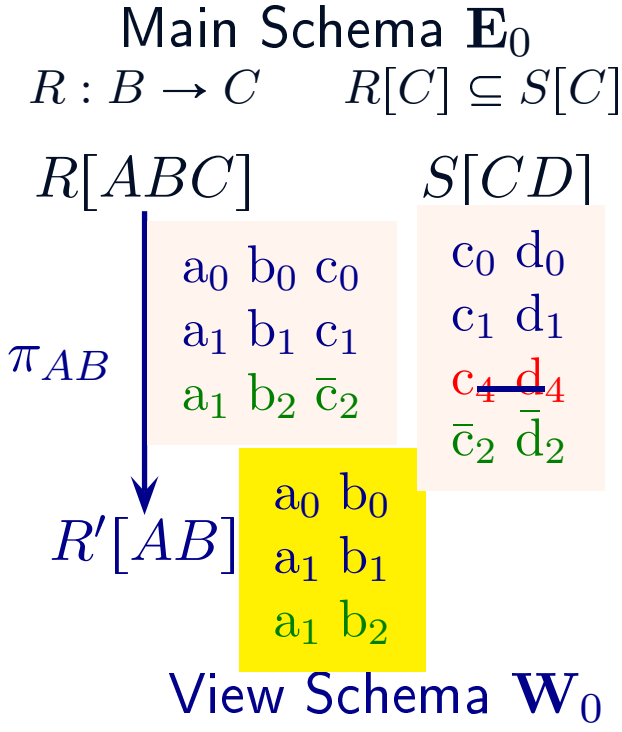
Example of Update Generator

Example: For the insertion-only reflection of the insertion of $(\mathrm{a}_1, \mathrm{b}_2)$ into the view:

$$G_+ = \{(\exists z)(\exists w)(R(\mathrm{a}_1, \mathrm{b}_2, z)\}$$
$$G_{\not}\ = M_1 \quad \text{(original state of } \mathbf{E}_0\text{)}$$

- For the solution in which $S(\mathrm{c}_4, \mathrm{d}_4)$ is also deleted:

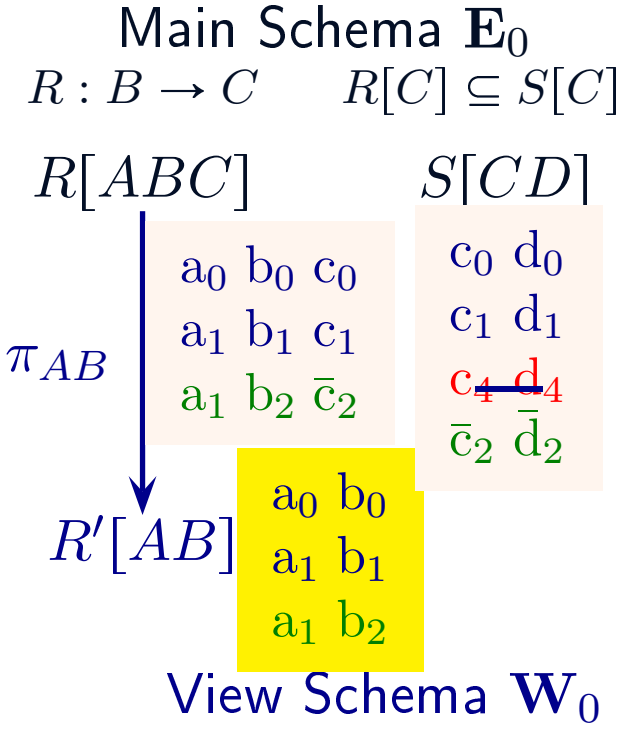
$$G'_+ = \{(\exists z)(\exists w)(R(\mathrm{a}_1, \mathrm{b}_2, z)\}$$
$$G'_{\not} = M_1 \backslash \{S(\mathrm{c}_4, \mathrm{d}_4)\}$$



Example of Update Generator

Example: For the insertion-only reflection of the insertion of $(\mathrm{a}_1, \mathrm{b}_2)$ into the view:

$$G_+ = \{(\exists z)(\exists w)(R(\mathrm{a}_1, \mathrm{b}_2, z)\}$$
$$G_{\not}= M_1 \quad \text{(original state of } \mathbf{E}_0\text{)}$$



- For the solution in which $S(\mathrm{c}_4, \mathrm{d}_4)$ is also deleted:

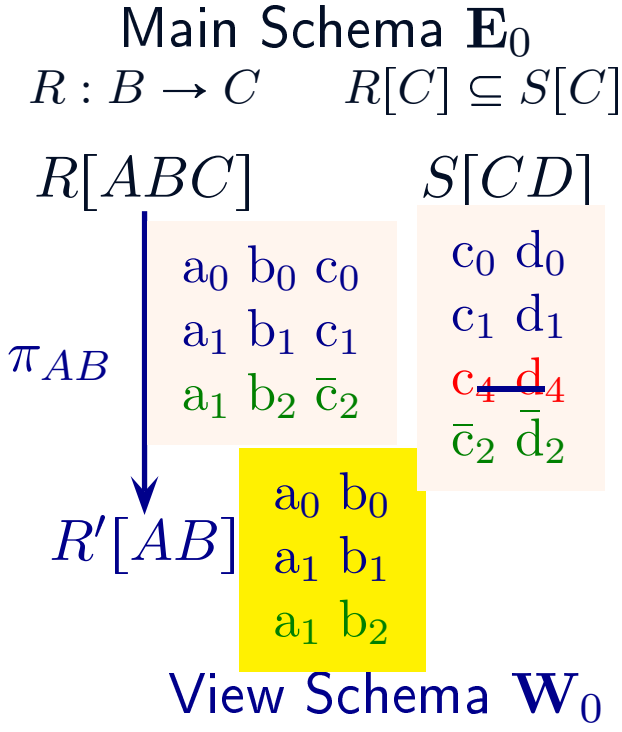
$$G'_+ = \{(\exists z)(\exists w)(R(\mathrm{a}_1, \mathrm{b}_2, z)\}$$
$$G'_{\not} = M_1 \backslash \{S(\mathrm{c}_4, \mathrm{d}_4)\}$$

- Since $\langle G_+, G_{\not}\rangle \sqsubsetneq_{\mathbf{D}} \langle G'_+, G'_{\not}\rangle$, the insertion-only solution is preferred, as desired.
- The collateral change entails in a suboptimal reflection.

Example of Update Generator

Example: For the insertion-only reflection of the insertion of $(\mathrm{a}_1, \mathrm{b}_2)$ into the view:

$$G_+ = \{(\exists z)(\exists w)(R(\mathrm{a}_1, \mathrm{b}_2, z)\}$$
$$G_{\not\perp} = M_1 \quad \text{(original state of } \mathbf{E}_0\text{)}$$



- For the solution in which $S(\mathrm{c}_4, \mathrm{d}_4)$ is also deleted:

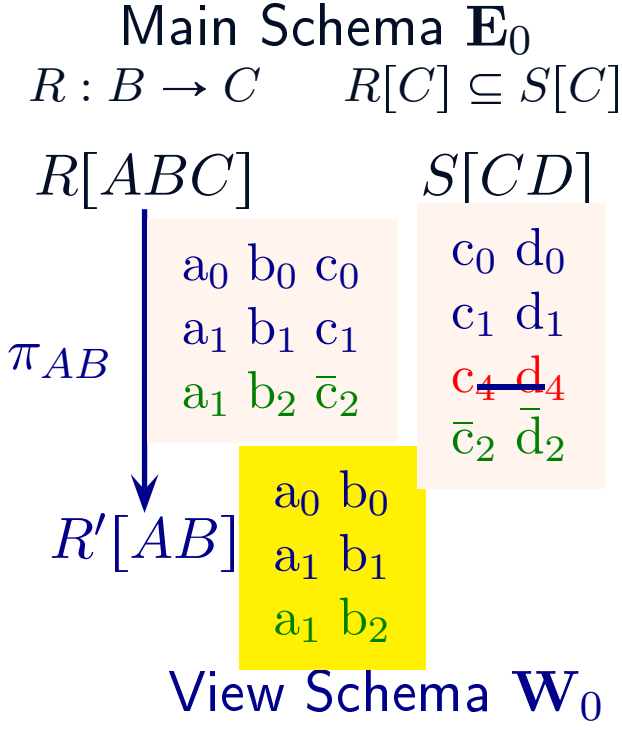
$$G'_+ = \{(\exists z)(\exists w)(R(\mathrm{a}_1, \mathrm{b}_2, z)\}$$
$$G'_{\not\perp} = M_1 \backslash \{S(\mathrm{c}_4, \mathrm{d}_4)\}$$

- Since $\langle G_+, G_{\not\perp}\rangle \sqsubsetneq_{\mathbf{D}} \langle G'_+, G'_{\not\perp}\rangle$, the insertion-only solution is preferred, as desired.
- The collateral change entails in a suboptimal reflection.
- This is a simplification; technical details have been ignored.

Example of Update Generator

Example: For the insertion-only reflection of the insertion of $(\mathrm{a}_1, \mathrm{b}_2)$ into the view:

$$G_+ = \{(\exists z)(\exists w)(R(\mathrm{a}_1, \mathrm{b}_2, z)\}$$
$$G_{\not=} = M_1 \quad \text{(original state of } \mathbf{E}_0\text{)}$$



- For the solution in which $S(\mathrm{c}_4, \mathrm{d}_4)$ is also deleted:

$$G'_+ = \{(\exists z)(\exists w)(R(\mathrm{a}_1, \mathrm{b}_2, z)\}$$
$$G'_{\not=} = M_1 \backslash \{S(\mathrm{c}_4, \mathrm{d}_4)\}$$

- Since $\langle G_+, G_{\not=} \rangle \sqsubsetneq_{\mathbf{D}} \langle G'_+, G'_{\not=} \rangle$, the insertion-only solution is preferred, as desired.
- The collateral change entails in a suboptimal reflection.
- This is a simplification; technical details have been ignored.
  - Constraints on $\mathbf{D}$: Horn with finite chase. (Implies *finite initial-model property*.)
  - G_+ and $G_{\not=}$ are *ideals*: closed under implication in the context of Constr($\mathbf{D}$).

Least-Insertion Optimality

Theorem: Under suitable conditions, every view update has a reflection which is *insertion optimal* in the sense that G_+ is least. $\square$

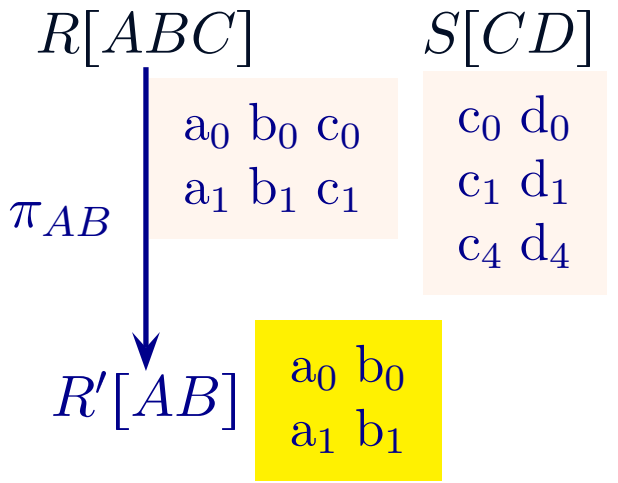
Least-Insertion Optimality

Theorem: Under suitable conditions, every view update has a reflection which is *insertion optimal* in the sense that G_+ is least. □

- Illustrate with the first update example.

Main Schema $\mathbf{E}_0$

$R : B \to C \qquad R[C] \subseteq S[C]$

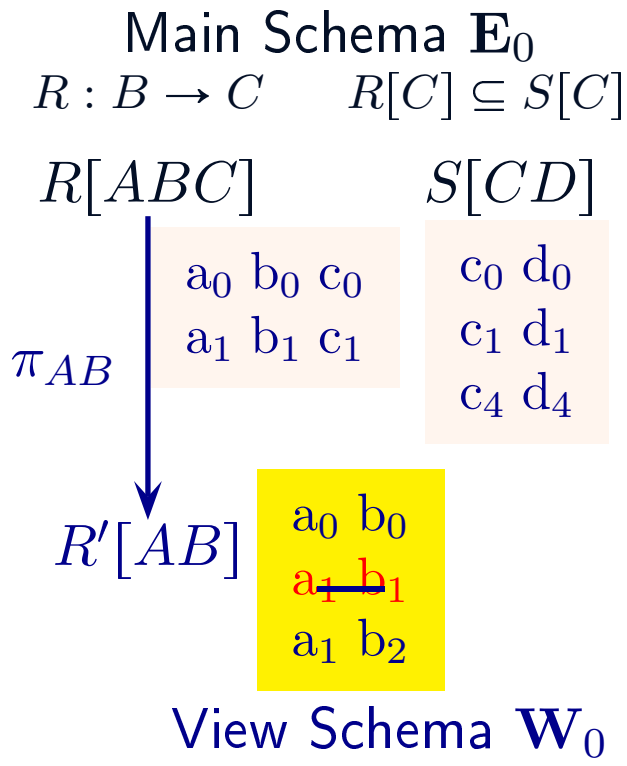


View Schema $\mathbf{W}_0$

Least-Insertion Optimality

Theorem: Under suitable conditions, every view update has a reflection which is *insertion optimal* in the sense that G_+ is least. □

- Illustrate with the first update example.
- Change $R'(\mathrm{a}_1, \mathrm{b}_1)$ to $R'(\mathrm{a}_1, \mathrm{b}_2)$.



Least-Insertion Optimality

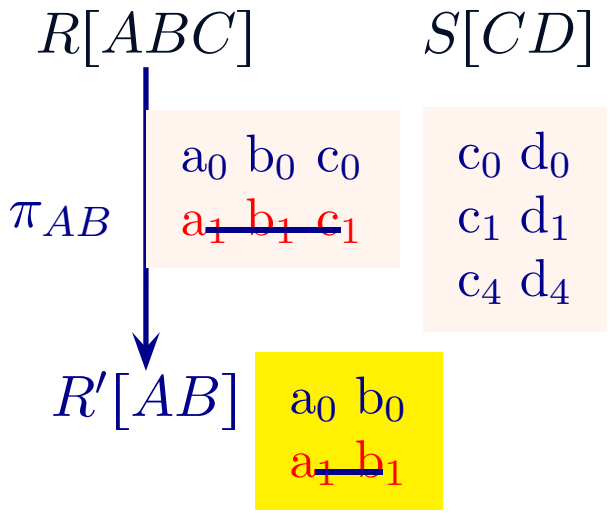
Theorem: Under suitable conditions, every view update has a reflection which is *insertion optimal* in the sense that G_+ is least. □

- Illustrate with the first update example.
- Change $R'(\mathrm{a}_1, \mathrm{b}_1)$ to $R'(\mathrm{a}_1, \mathrm{b}_2)$.

Idea: First delete $R'(\mathrm{a}_1, \mathrm{b}_1)$ with least reflection,

Main Schema $\mathbf{E}_0$

$R : B \to C$ $\quad R[C] \subseteq S[C]$



View Schema $\mathbf{W}_0$

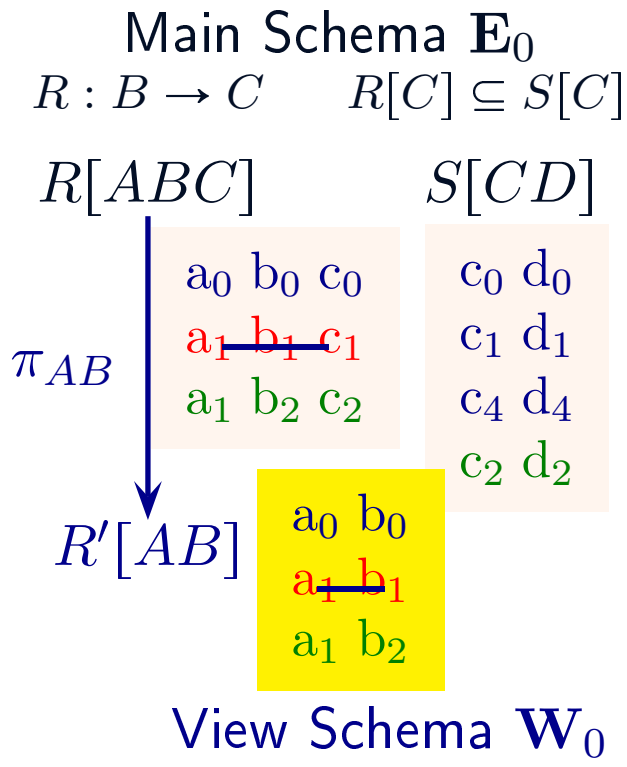
Least-Insertion Optimality

Theorem: Under suitable conditions, every view update has a reflection which is *insertion optimal* in the sense that G_+ is least. □

- Illustrate with the first update example.
- Change $R'(\mathrm{a}_1, \mathrm{b}_1)$ to $R'(\mathrm{a}_1, \mathrm{b}_2)$.

Idea: First delete $R'(\mathrm{a}_1, \mathrm{b}_1)$ with least reflection, then insert $R'(\mathrm{a}_1, \mathrm{b}_2)$ with minimal new information.

- There are details which must be solved to make this idea work, including:



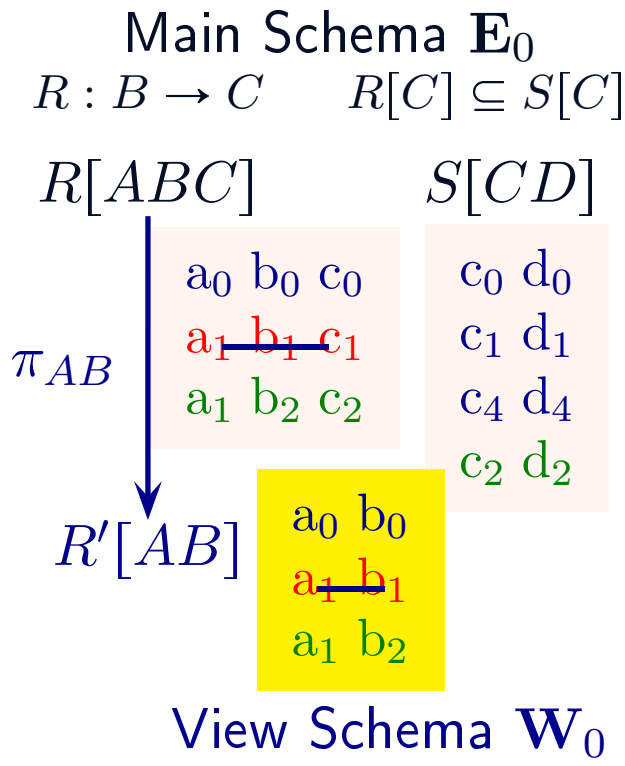
Least-Insertion Optimality

Theorem: Under suitable conditions, every view update has a reflection which is *insertion optimal* in the sense that G_+ is least. □

- Illustrate with the first update example.
- Change $R'(\mathrm{a}_1, \mathrm{b}_1)$ to $R'(\mathrm{a}_1, \mathrm{b}_2)$.

Idea: First delete $R'(\mathrm{a}_1, \mathrm{b}_1)$ with least reflection, then insert $R'(\mathrm{a}_1, \mathrm{b}_2)$ with minimal new information.

- There are details which must be solved to make this idea work, including:



Legal deletion: The state obtained by deletion in the view may not satisfy the integrity constraints, and so not correspond to a legal state in the main schema.

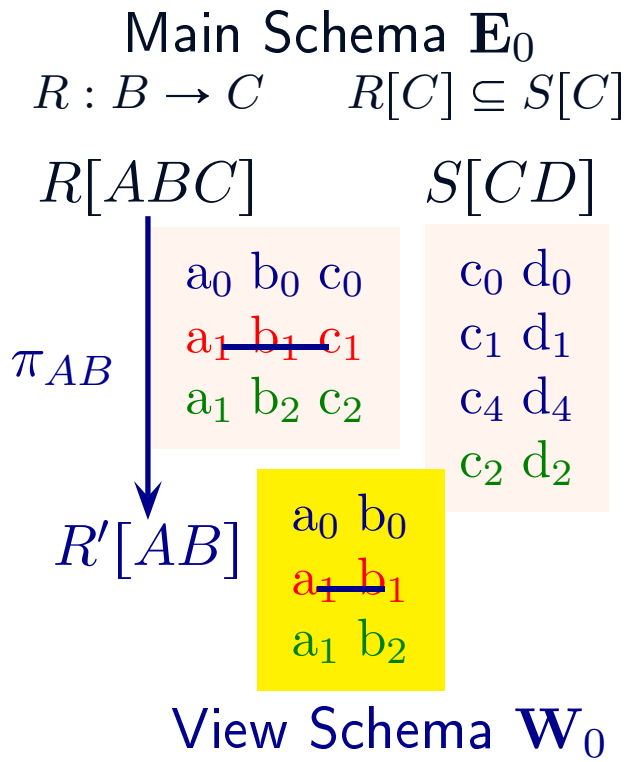
Least-Insertion Optimality

Theorem: Under suitable conditions, every view update has a reflection which is *insertion optimal* in the sense that G_+ is least. □

- Illustrate with the first update example.
- Change $R'(\mathrm{a}_1, \mathrm{b}_1)$ to $R'(\mathrm{a}_1, \mathrm{b}_2)$.

Idea: First delete $R'(\mathrm{a}_1, \mathrm{b}_1)$ with least reflection, then insert $R'(\mathrm{a}_1, \mathrm{b}_2)$ with minimal new information.

- There are details which must be solved to make this idea work, including:



Legal deletion: The state obtained by deletion in the view may not satisfy the integrity constraints, and so not correspond to a legal state in the main schema.

Solution: Finite initial model property ensures an extendible a "skeleton" in the main schema.

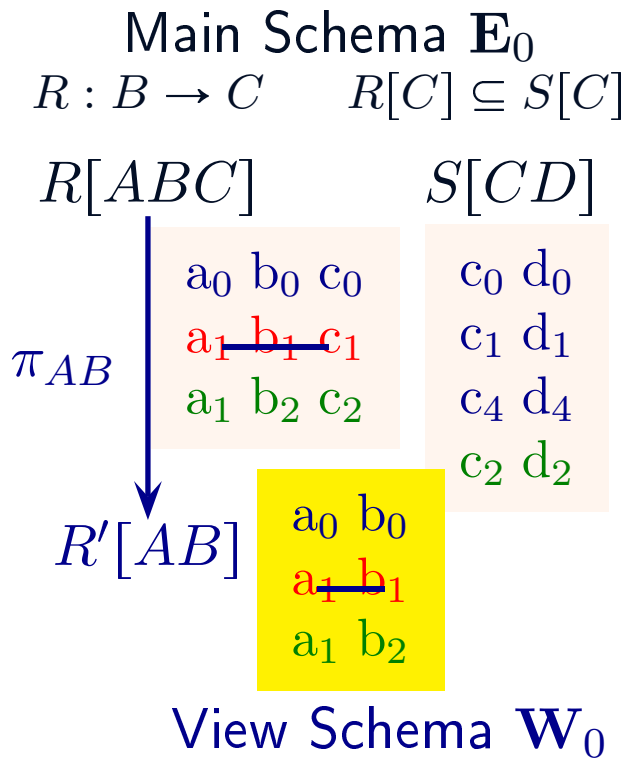
Least-Insertion Optimality

Theorem: Under suitable conditions, every view update has a reflection which is *insertion optimal* in the sense that G_+ is least. □

- Illustrate with the first update example.
- Change $R'(\mathrm{a}_1, \mathrm{b}_1)$ to $R'(\mathrm{a}_1, \mathrm{b}_2)$.

Idea: First delete $R'(\mathrm{a}_1, \mathrm{b}_1)$ with least reflection, then insert $R'(\mathrm{a}_1, \mathrm{b}_2)$ with minimal new information.

- There are details which must be solved to make this idea work, including:



Legal deletion: The state obtained by deletion in the view may not satisfy the integrity constraints, and so not correspond to a legal state in the main schema.

Solution: Finite initial model property ensures an extendible a "skeleton" in the main schema.

Optimal deletion: There may not be a least reflection of a deletion.

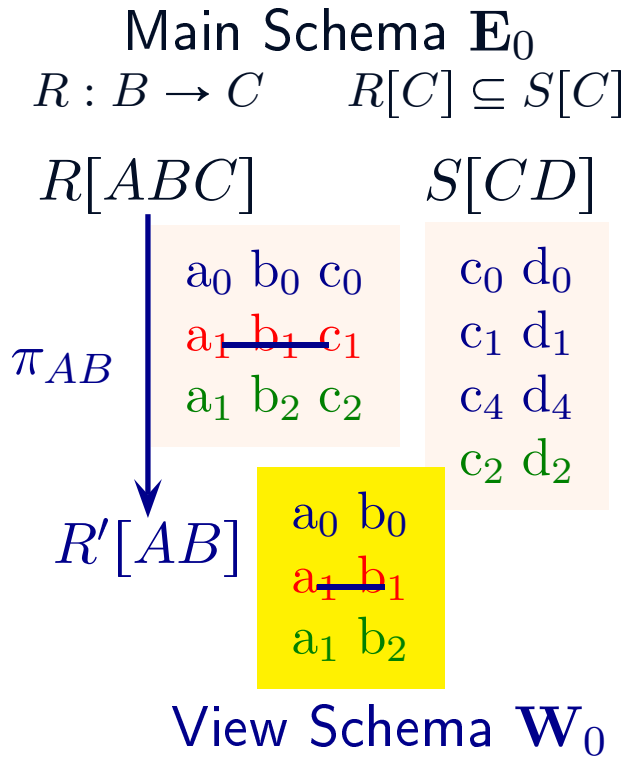
Least-Insertion Optimality

Theorem: Under suitable conditions, every view update has a reflection which is *insertion optimal* in the sense that G_+ is least. □

- Illustrate with the first update example.
- Change $R'(\mathrm{a}_1, \mathrm{b}_1)$ to $R'(\mathrm{a}_1, \mathrm{b}_2)$.

Idea: First delete $R'(\mathrm{a}_1, \mathrm{b}_1)$ with least reflection, then insert $R'(\mathrm{a}_1, \mathrm{b}_2)$ with minimal new information.

- There are details which must be solved to make this idea work, including:



Legal deletion: The state obtained by deletion in the view may not satisfy the integrity constraints, and so not correspond to a legal state in the main schema.

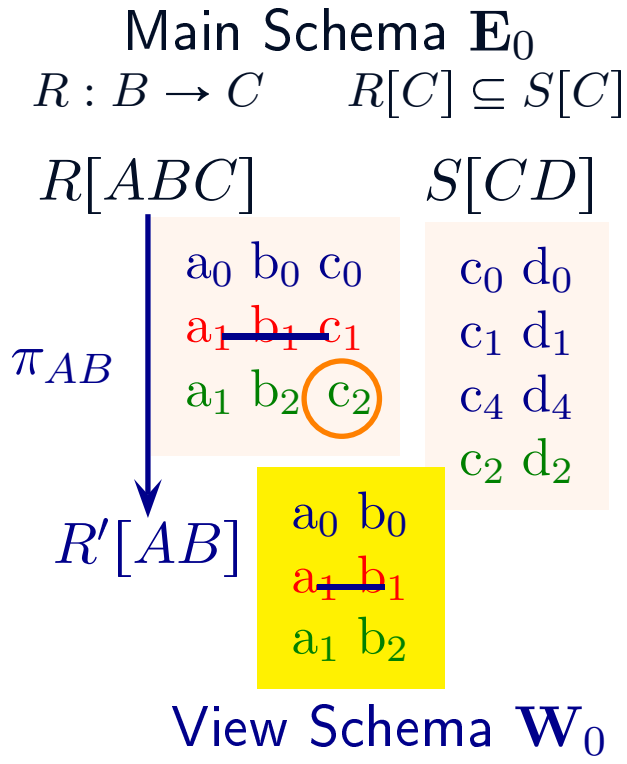
Solution: Finite initial model property ensures an extendible a "skeleton" in the main schema.

Optimal deletion: There may not be a least reflection of a deletion.

Solution: View must define a *unit-head pair*. (No rules with more than one antecedent) [Hegner 2009]

Deletion Optimality

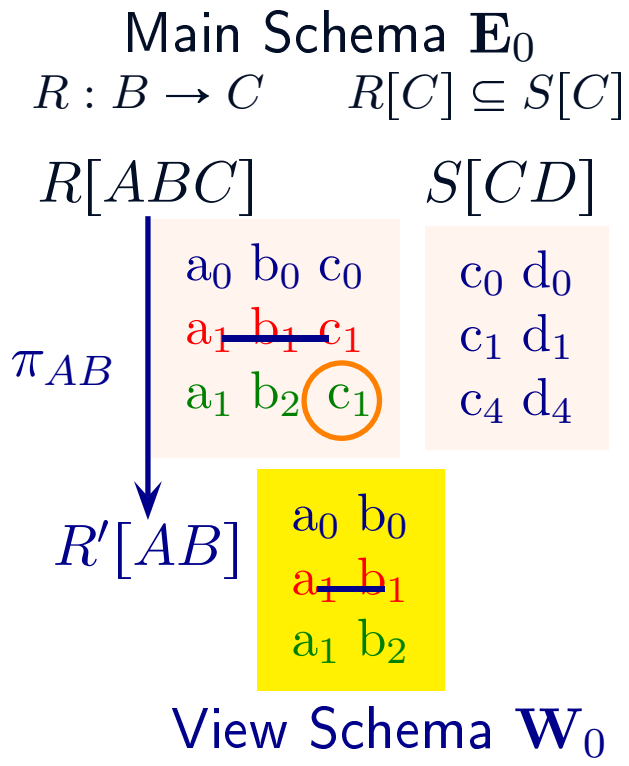
- The example under consideration is not *deletion optimal for information*.



Deletion Optimality

- The example under consideration is not *deletion optimal for information*.
- The reflection shown preserves more information (but adds more as well).

Preserves: $(\exists y)(R(\mathrm{a}_1, y, \mathrm{c}_1)$ *Adds*: $R(\mathrm{a}_1, \mathrm{b}_2, \mathrm{c}_1)$

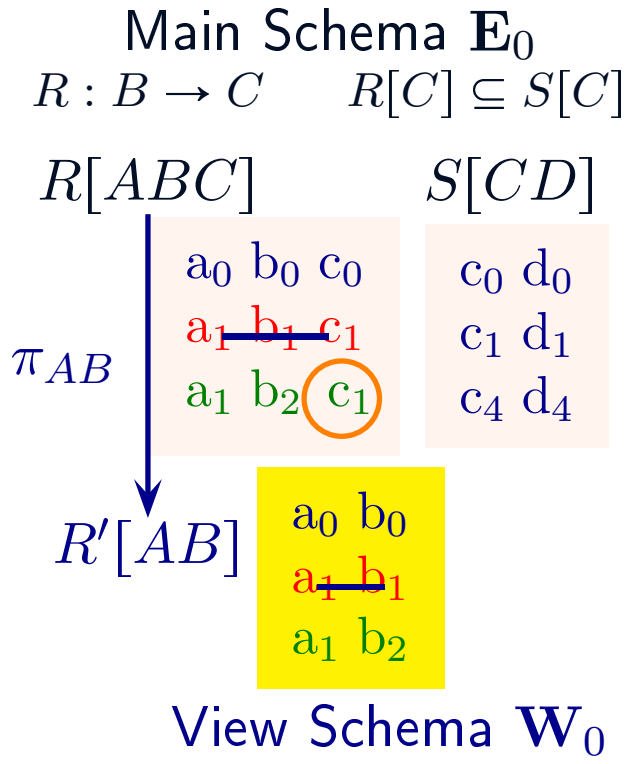


Deletion Optimality

- The example under consideration is not *deletion optimal for information*.
- The reflection shown preserves more information (but adds more as well).

Preserves: $(\exists y)(R(\mathrm{a}_1, y, \mathrm{c}_1)$ *Adds*: $R(\mathrm{a}_1, \mathrm{b}_2, \mathrm{c}_1)$

- It is in fact *deletion optimal for information* in the sense that $G_{\neq}$ is greatest.

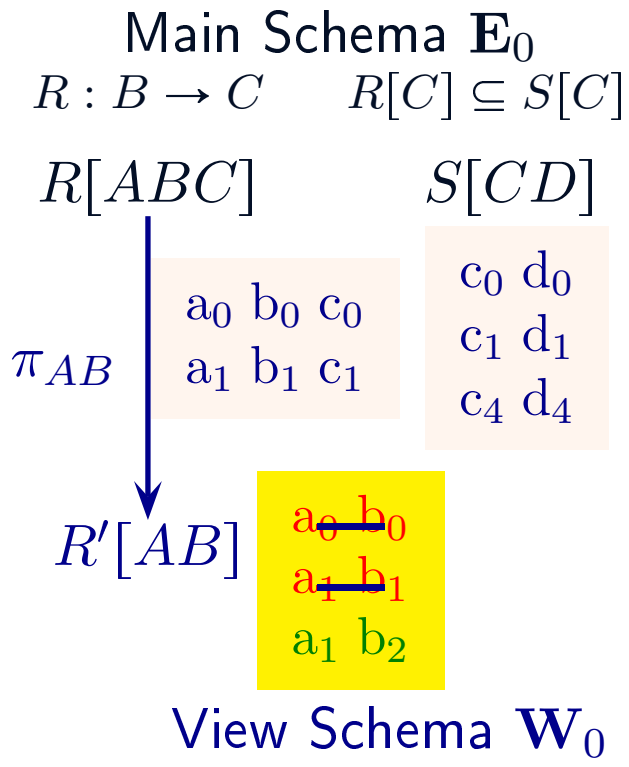


Deletion Optimality

- The example under consideration is not *deletion optimal for information*.
- The reflection shown preserves more information (but adds more as well).

Preserves: $(\exists y)(R(\mathrm{a}_1, y, \mathrm{c}_1)$ *Adds*: $R(\mathrm{a}_1, \mathrm{b}_2, \mathrm{c}_1)$

- It is in fact *deletion optimal for information* in the sense that $G_{\neq}$ is greatest.
- Such optimality is not achievable in general. Consider deleting $R'(\mathrm{a}_0, \mathrm{b}_0)$ as well.

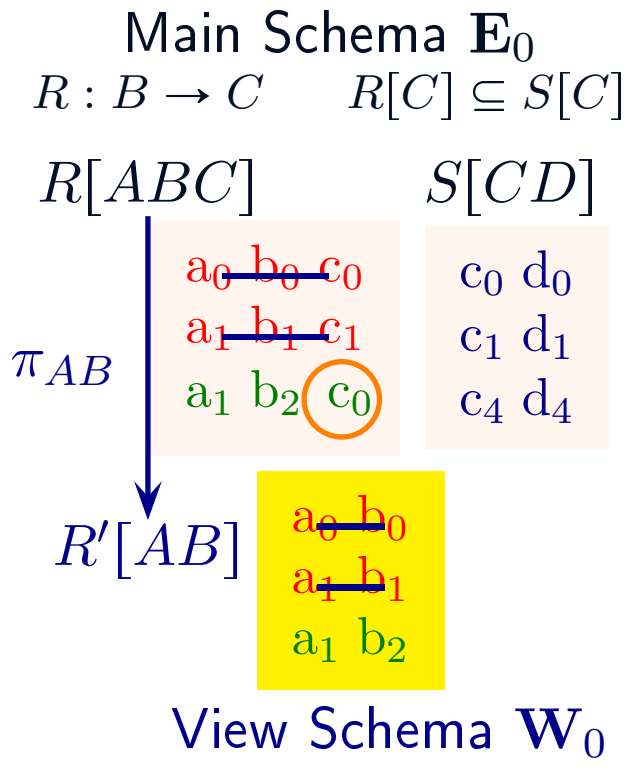


Deletion Optimality

- The example under consideration is not *deletion optimal for information*.
- The reflection shown preserves more information (but adds more as well).

Preserves: $(\exists y)(R(\mathrm{a}_1, y, \mathrm{c}_1)$ *Adds*: $R(\mathrm{a}_1, \mathrm{b}_2, \mathrm{c}_1)$

- It is in fact *deletion optimal for information* in the sense that $G_{\neq}$ is greatest.
- Such optimality is not achievable in general. Consider deleting $R'(\mathrm{a}_0, \mathrm{b}_0)$ as well.
- There are two incomparable reflections which minimize $G_{\neq}$.
  This one preserves $(\exists x)(\exists y)(R(x, y, \mathrm{c}_0)$. This one preserves $(\exists x)(\exists y)(R(x, y, \mathrm{c}_1)$.

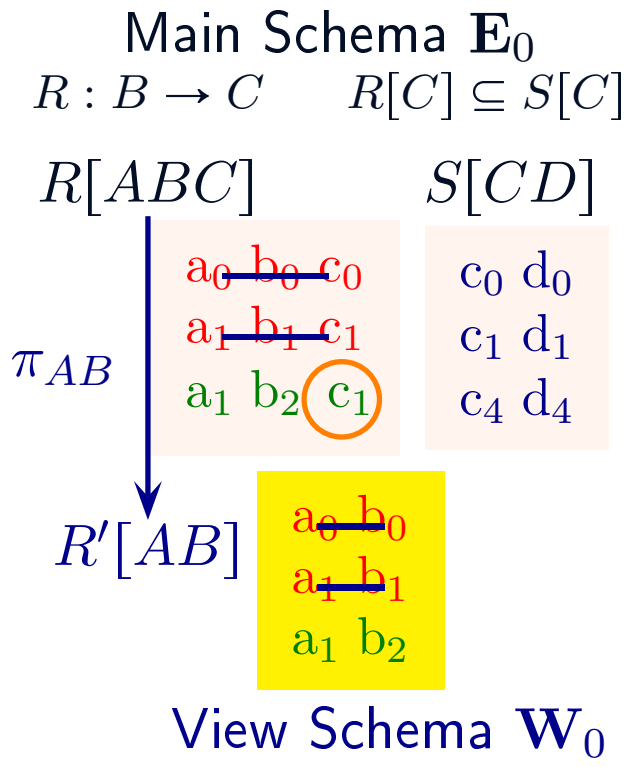


Deletion Optimality

- The example under consideration is not *deletion optimal for information*.
- The reflection shown preserves more information (but adds more as well).

Preserves: $(\exists y)(R(\mathrm{a}_1, y, \mathrm{c}_1)$ *Adds*: $R(\mathrm{a}_1, \mathrm{b}_2, \mathrm{c}_1)$

- It is in fact *deletion optimal for information* in the sense that $G_{\neq}$ is greatest.
- Such optimality is not achievable in general. Consider deleting $R'(\mathrm{a}_0, \mathrm{b}_0)$ as well.
- There are two incomparable reflections which minimize $G_{\neq}$.
  This one preserves $(\exists x)(\exists y)(R(x, y, \mathrm{c}_0)$. This one preserves $(\exists x)(\exists y)(R(x, y, \mathrm{c}_1)$.

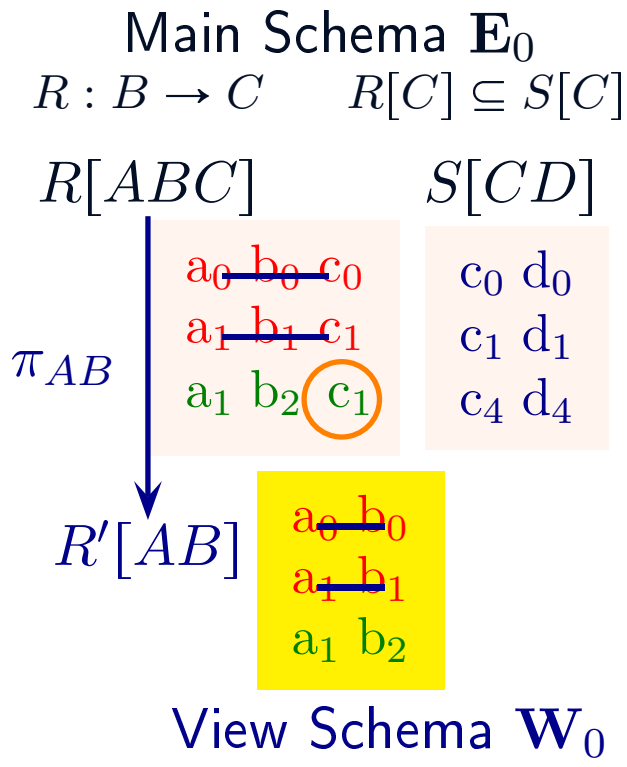


Deletion Optimality

- The example under consideration is not *deletion optimal for information*.
- The reflection shown preserves more information (but adds more as well).

Preserves: $(\exists y)(R(\mathrm{a}_1, y, \mathrm{c}_1)$ *Adds*: $R(\mathrm{a}_1, \mathrm{b}_2, \mathrm{c}_1)$

- It is in fact *deletion optimal for information* in the sense that $G_{\neq}$ is greatest.
- Such optimality is not achievable in general. Consider deleting $R'(\mathrm{a}_0, \mathrm{b}_0)$ as well.
- There are two incomparable reflections which minimize $G_{\neq}$.
  This one preserves $(\exists x)(\exists y)(R(x, y, \mathrm{c}_0)$. This one preserves $(\exists x)(\exists y)(R(x, y, \mathrm{c}_1)$.
- Deletion optimality is possible only when the view update corresponds to changes in fields of tuples.

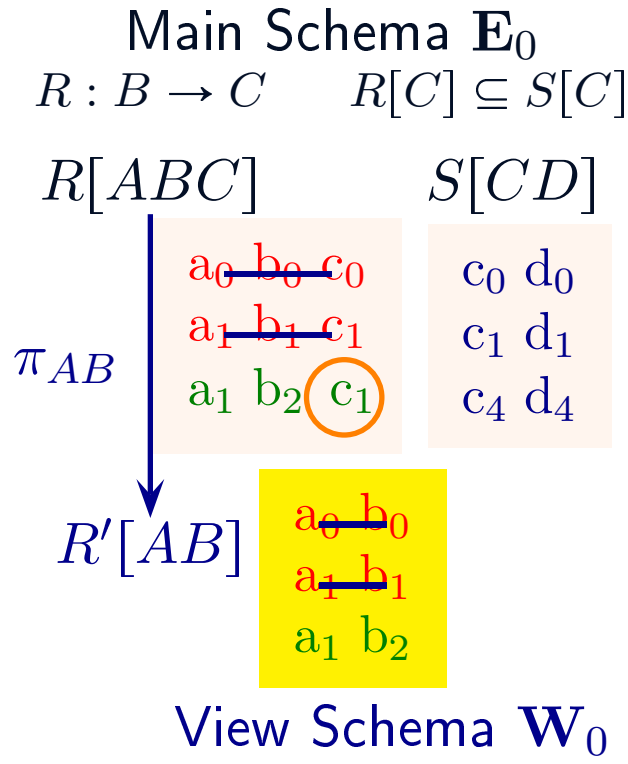


Deletion Optimality

- The example under consideration is not *deletion optimal for information*.
- The reflection shown preserves more information (but adds more as well).

Preserves: $(\exists y)(R(\mathrm{a}_1, y, \mathrm{c}_1)$ *Adds*: $R(\mathrm{a}_1, \mathrm{b}_2, \mathrm{c}_1)$

- It is in fact *deletion optimal for information* in the sense that $G_{\neq}$ is greatest.
- Such optimality is not achievable in general. Consider deleting $R'(\mathrm{a}_0, \mathrm{b}_0)$ as well.
- There are two incomparable reflections which minimize $G_{\neq}$.
  This one preserves $(\exists x)(\exists y)(R(x, y, \mathrm{c}_0)$. This one preserves $(\exists x)(\exists y)(R(x, y, \mathrm{c}_1)$.
- Deletion optimality is possible only when the view update corresponds to changes in fields of tuples.
- The insertion optimal realization of the previous slides is, however, *deletion optimal for tuples*.



Conclusions and Further Directions

Conclusions:

- The idea of a semantically motivated distance between databases based upon Boolean conjunctive queries has been applied to non-monotonic queries.

Conclusions and Further Directions

Conclusions:

- The idea of a semantically motivated distance between databases based upon Boolean conjunctive queries has been applied to non-monotonic queries.
- Under suitable conditions, reflections to view updates which are insertion-optimal as well as deletion-optimal for tuples are shown to exist.

Conclusions and Further Directions

Conclusions:

- The idea of a semantically motivated distance between databases based upon Boolean conjunctive queries has been applied to non-monotonic queries.
- Under suitable conditions, reflections to view updates which are insertion-optimal as well as deletion-optimal for tuples are shown to exist.

Further Directions:

- Classification of reflection type along *tuple update* versus *delete-insert* lines.

Conclusions and Further Directions

Conclusions:

- The idea of a semantically motivated distance between databases based upon Boolean conjunctive queries has been applied to non-monotonic queries.
- Under suitable conditions, reflections to view updates which are insertion-optimal as well as deletion-optimal for tuples are shown to exist.

Further Directions:

- Classification of reflection type along *tuple update* versus *delete-insert* lines.
- Integration of this measure with more traditional syntactic ones.