3.1 Intuitively Defining Update Rules
- Transition Rules -
Working with Two Database States

- **Dilemma:**
  - Effectiveness tests for insertions to be performed in the "old" state of the DB (before executing the resp. updates).
  - Effectiveness tests for deletions to be performed in the "new" state of the DB (after updating the base facts).
  - If insertions and deletions are combined (as, e.g., in U₃), both states are required simultaneously.

  **How to work simultaneously with both states?**

- **Way out:** Fortunately, the "missing" state can always be "simulated" via rules and derivation!

- Two variants can be imagined:
  - "Pessimistic" variant (expecting integrity violations as very likely): Keep the old state and simulate the new state!
  - "Optimistic" variant (accepts the – low – risk of a rollback): Perform changes ahead of integrity checking and simulate the old state!
In set-theoretic terms, the new state can be defined by the equation

\[ F_{\text{new}} = (F_{\text{old}} - F_{-}) \cup F_{+} \]

Here \( F_{+} \) contains all insertions from the update \( U \), \( F_{-} \) all deletions due to \( U \).

This equation is valid only if in addition the following assumptions about keeping changes „under control“ are always satisfied:

\[
\begin{align*}
F_{+} \cap F_{\text{old}} &= \emptyset \\
F_{-} \cap F_{\text{old}} &= F_{-} \\
F_{+} \cap F_{-} &= \emptyset \\
\end{align*}
\]

{ All updates are true. Net effects are already computed. }
Working with Two Database States (3)

- The set equation on the previous slide forms the basis for a simulation of the new state from the old state and the "differentials" \((F^+, F^-)\), i.e., for the pessimistic approach:

  \[
  F^{\text{new}} = (F^{\text{old}} - F^-) \cup F^+
  \]

- If preferring to work **optimistically**, one has to carry \(F^{\text{old}}\) to the other side of the equation:

  \[
  F^{\text{old}} = (F^{\text{new}} - F^+) \cup F^-
  \]

- In the following, we will follow the **pessimistic approach** \((F^{\text{new}}\) simulated) and thus will omit the index \(^{\text{old}}\).

- "Simulating" means that first for each **base relation** the new state is determined via a derived auxiliary relation, the definition of which can be obtained from the above equation, e.g.:

  \[
  e^{\text{new}}(X,Y) \leftarrow e(X,Y), \text{ not } e^-(X,Y).
  \]

  \[
  e^{\text{new}}(X,Y) \leftarrow e^+(X,Y).
  \]

  **Transition Rules**
In the motivating example, the effectiveness test just needs the new state of the derived relation \( p \), not that of \( e \). If trying to define \( p^{\text{new}} \) via transition rules defined as for \( e \), the following internal set of rules will be constructed:

\[
\begin{align*}
\text{R} & \quad e_{\text{cyclic}} \leftarrow s(X). \\
& \quad s(X) \leftarrow p(X,X). \\
& \quad p(X,Y) \leftarrow e(X,Y). \\
& \quad p(X,Y) \leftarrow e(X,Z), p(Z,Y).
\end{align*}
\]

\[
\begin{align*}
\text{R}^\Delta & \quad p^{-}(X,Y) \leftarrow e^{-}(X,Y), \text{ not } p^{\text{new}}(X,Y). \\
& \quad p^{-}(X,Y) \leftarrow e^{-}(X,Z), p(Z,Y), \text{ not } p^{\text{new}}(X,Y). \\
& \quad p^{-}(X,Y) \leftarrow e^{-}(X,Z), p^{-}(Z,Y), \text{ not } p^{\text{new}}(X,Y).
\end{align*}
\]

\[
\begin{align*}
\text{R}^\tau & \quad p^{\text{new}}(X,Y) \leftarrow p(X,Y), \text{ not } p^{-}(X,Y). \\
& \quad p^{\text{new}}(X,Y) \leftarrow p^{+}(X,Y).
\end{align*}
\]

It is immediately visible that the dependency graph of the combined rule set is no longer stratifiable and, moreover, that paradox derivations are inevitable:
The combined rule set is not only **no longer stratifiable** but it does not represent the true transition portrait in any case due to **paradox derivations** :

For example:

- $F: e(1,2). e(2,1).$
- $F^*: p(1,2). p(1,1). p(2,1). p(2,2).$
- $U_3: \neg e(2,1)$

All these consequences would be undefined using this erroneous rule set!
Fortunately, there is a different way of defining transition rules for derived relations: The new state of the derived relation (in the rule head) results from the new state of the relations referenced in the rule body!

We call this – quite obvious – way of defining the new state "naive" transition rules.

Only for base relations we will have to stick with the previous variant, which will be called incremental transition rules (as there is no alternative):

\[
\begin{align*}
p(X,Y) & \leftarrow e(X,Y). \\
p(X,Y) & \leftarrow e(X,Z), p(Z,Y). \\
p^{\text{new}}(X,Y) & \leftarrow e^{\text{new}}(X,Y). \\
p^{\text{new}}(X,Y) & \leftarrow e^{\text{new}}(X,Z), p^{\text{new}}(Z,Y). \\
e^{\text{new}}(X,Y) & \leftarrow e(X,Y), \not e^{-}(X,Y). \\
e^{\text{new}}(X,Y) & \leftarrow e^{+}(X,Y).
\end{align*}
\]
Naive Rules Avoid Stratification Problems

Avoiding incremental transition rules and using naive ones instead **solves the problem**:

- **R**
  
  \[
  \begin{align*}
  e_{\text{cyclic}} & \leftarrow s(X). \\
  s(X) & \leftarrow p(X,X). \\
  p(X,Y) & \leftarrow e(X,Y). \\
  p(X,Y) & \leftarrow e(X,Z), p(Z,Y).
  \end{align*}
  \]

- **R^\Delta**
  
  \[
  \begin{align*}
  p^-(X,Y) & \leftarrow e^-(X,Y), \textbf{not } p^{\text{new}}(X,Y). \\
  p^-(X,Y) & \leftarrow e^-(X,Z), p(Z,Y), \textbf{not } p^{\text{new}}(X,Y). \\
  p^-(X,Y) & \leftarrow e(X,Z), p^-(Z,Y), \textbf{not } p^{\text{new}}(X,Y).
  \end{align*}
  \]

- **R^\tau**
  
  \[
  \begin{align*}
  p^{\text{new}}(X,Y) & \leftarrow e^{\text{new}}(X,Y). \\
  p^{\text{new}}(X,Y) & \leftarrow e^{\text{new}}(X,Z), p^{\text{new}}(Z,Y).
  \end{align*}
  \]
If combining positive and negative propagation rules, . . .

. . . another problem pops up: Both rule sets don’t reference each other at all!

However, we did observe such side effects in our example!

For example, consider the rule set $U_3$:

$U_3: \{ + e(4,1); - e(2,3) \}$

$U_3$ was constructed on purpose in such a way that deletions compensate the consequences of insertions: Instead of 10 positive deltas (in case of $U_1$) there were just 4 positive deltas remaining if applying $U_3$!
False Residual Evaluations

- The "error in reasoning" causing this defect was to perform residual evaluation in the old state, thus failing to consider the consequences of deletions (which can be observed in the new state only):

\[
p^+(4,3) \leftarrow e^+(4,1), p(1,3), \neg p(4,3).
\]

\[U_3:\{ + e(4,1) ; - e(2,3) \} ! \rightarrow p^-(2,3) \rightarrow p^-(1,3)\]

- Instead, residual evaluation has to be performed over those facts only which "survive" in the new state:

\[
p^+(X,Y) \leftarrow e^+(X,Z), p(Z,Y), \neg p^-(Z,Y), \neg p(X,Y).
\]

- Combining rules for simplicity’s sake:

\[
p^+(X,Y) \leftarrow e^+(X,Z), p^+(Z,Y), \neg p(X,Y).
\]
Correct Residual Evaluation

\[ p(X,Y) \leftarrow e(X,Y). \]
\[ p(X,Y) \leftarrow e(X,Z), p(Z,Y). \]

\[ p(X,Y) \leftarrow e^+(X,Y), p(X,Y). \]
\[ p(X,Y) \leftarrow e^+(X,Z), p^\text{new}(Z,Y), p(Z,Y). \]
\[ p(X,Y) \leftarrow e^\text{new}(X,Z), p^+(Z,Y), p^\text{new}(Z,Y). \]

\[ p^-(X,Y) \leftarrow e^-(X,Y), p^\text{new}(X,Y). \]
\[ p^-(X,Y) \leftarrow e^-(X,Z), p^-(Z,Y), p^\text{new}(X,Y). \]
\[ p^-(X,Y) \leftarrow e(X,Z), p^-(Z,Y), p^\text{new}(X,Y). \]

\[ e^\text{cyclic}^+ \leftarrow s^+(X). \]
\[ s^+(X) \leftarrow p^+(X,X). \]
\[ p^+(X,Y) \leftarrow e^+(X,Y), \text{not } p(X,Y). \]
\[ p^+(X,Y) \leftarrow e^+(X,Z), p^\text{new}(Z,Y), \text{not } p(X,Y). \]
\[ p^+(X,Y) \leftarrow e^\text{new}(X,Z), p^+(Z,Y), \text{not } p(X,Y). \]

U_3: \{ + e(4,1); - e(2,3) \}!
Correct Old and New States: Summary

- **Residual evaluation** in a propagation rule is performed
  - in the **new** state, if defining induced insertions,
  - in the **old** state, if defining induced deletions.

- **Effectiveness** tests on the other hand is performed
  - in the **old** state, if defining induced insertions,
  - in the **new** state, if defining induced deletions.

- Residual evaluation is **unnecessary**, if the respective original rules are pure projection or union rules (which means that bodies consists of one literal only).
  
  \[
  \begin{align*}
  p(X) & \leftarrow q(X). \\
  p(X) & \leftarrow r(X).
  \end{align*}
  \]

- Effectiveness tests are **unnecessary**, if the respective relations are defined by just one rule **without** local variables, i.e., intersection, join, product, or difference:
  
  \[
  \begin{align*}
  p(X) & \leftarrow r(X), q(X). \\
  p(X,Y) & \leftarrow r(X), q(Y). \\
  p(X,Y) & \leftarrow q(X,Y), \text{not} r(X,Y). \\
  p(X,Y,Z) & \leftarrow q(X,Y), r(Y,Z).
  \end{align*}
  \]
Rule Transformation in the Example

In our delta and transition rules we only need the p-rules from the original rule set anymore, not s- and e_cyclic-rules: \( R' \)

\[
\begin{align*}
p(X,Y) & \leftarrow e(X,Y). \\
p(X,Y) & \leftarrow e(X,Z), p(Z,Y).
\end{align*}
\]

Thus:

\[
R_{\text{update}} = R^\Delta \cup R^\tau \cup R'
\]
Dependency Graph of the Internal Rule Set

Possible seed relations

e_{cyclic}^+

s^+

new

p

p^+

p^-

not

not

not

e^-

e^+

e

"Dependency Graph of the Internal Rule Set"
Update propagation in this example requires at least two strata:
Update Propagation and Negation

- Update propagation "through" negative literals is possible in a very similar way as for positive literals.

- Only (important) deviation: In the 3rd step (propagation) the inverted sign is transferred to the induced update:

```plaintext
+ p(2).
− r(3)

Fold:

p(X) ← q(X,Y), not r(Y).

Propagation rule (for deletions) in this example:

p^+(X) ← q^{new}(X,Y), r^-(Y), not p(X).
```
Test literals (with or without arithmetic operators) may not serve as "entry points" for propagating base updates:

\[
p(X) \leftarrow q(X,Y), X > Y + 1.
\]

\[
- q(1,3)
\]

\[
1 > 3 + 1
\]

When evaluating such test literals it is only required to take care that a suitable Sideways Information Strategy (SIPS) is used during residual evaluation: Remember that test literals (like negative literals) may be evaluated in fully instantiated form only:

\[
p(X) \leftarrow q(X,Y), X > Z + 1, r(X,Z).
\]

\[
- q(1,3)
\]
3.2 A Systematic Approach
Develop Propagation Rules Systematically?

- The update propagation rules have been developed so far quite intuitively, motivated by erroneous derivations and stepwise refinements.

- For developing propagation rules systematically, we start with a basic axiom about induced insertions and deletions:

\[
A^+ \leftarrow \text{new } A, \text{old } \neg A.
\]

\[
A^- \leftarrow \text{old } A, \text{new } \neg A.
\]

Meta-predicates (mappings) referring to the new respectively old database state.

Superscripts ‘+’ and ‘−’ are mappings:

- positive literal \( A \equiv r(t_1, \ldots, t_n) \Rightarrow A^+ \equiv r^+(t_1, \ldots, t_n)\) and \( A^- \equiv r^-(t_1, \ldots, t_n)\)

- negative literal \( \neg A \Rightarrow L^+ \equiv (\neg A)^+ \equiv A^- \) and \( L^- \equiv (\neg A)^- \equiv A^+ \)
The Meaning of the Meta-Predicates

- Meta-predicates `old` and `new` refer to the old respectively new database state and it is assumed that evaluations on both states are correctly performed.

- Transition rules together with the original rules could be used for the respective evaluation:

\[
\begin{align*}
p^+(X,Y) & \leftarrow \text{new } p(X,Y), \text{old } \text{not } p(X,Y). \\
p^+(X,Y) & \leftarrow p^\text{new}(X,Y), \text{not } p^\text{old}(X,Y). \\
p^\text{new}(X,Y) & \leftarrow e^\text{new}(X,Y) \quad p^\text{old}(X,Y) \leftarrow e^\text{old}(X,Y) \\
p^\text{new}(X,Y) & \leftarrow e^\text{new}(X,Z), p^\text{new}(Z,Y) \quad p^\text{old}(X,Y) \leftarrow e^\text{old}(X,Z), p^\text{old}(Z,Y) \\
e^\text{new}(X,Y) & \leftarrow e^+(X,Y) \\
e^\text{new}(X,Y) & \leftarrow e^\text{old}(X,Y), \text{not } e^-(Z,Y)
\end{align*}
\]
Derivability and Effectiveness Tests

- The derivability test \( \{\text{new} \mid \text{old}\} \ A \) is performed in order to determine whether \( A \) is derivable in the new respectively old state.

  \[ A^+ \leftarrow \text{new} \ A, \quad \text{old} \not A. \]
  \[ A^- \leftarrow \text{old} \ A, \quad \text{new} \not A. \]

  \[ \Rightarrow \text{responsible for calculating potential updates} \]

- The effectiveness test \( \{\text{old} \mid \text{new}\} \not A \) checks whether the fact obtained by the derivability is not derivable in the opposite state.

  \[ \Rightarrow \text{checking whether a potential update is a true update} \]
Propagation Rules for True Updates

- All propagation rules are derived from a stratifiable rule set $R$.

- For each rule $A \leftarrow L_1, \ldots, L_n \in R$ and each body literal $L_i$ ($i=1,\ldots,n$) two propagation rules of the form

$$A^+ \leftarrow L^+_{i, \text{new}} (L_1, \ldots, L_{i-1}, L_{i+1}, \ldots, L_n), \quad \text{old not } A.$$

$$A^- \leftarrow L^-_{i, \text{old}} (L_1, \ldots, L_{i-1}, L_{i+1}, \ldots, L_n), \quad \text{new not } A.$$

are in $R^A$.

- The literals new $L_j$ and old $L_j$ ($j=1,\ldots,i-1,i+1,\ldots,n$) are called side literals.

Effectiveness tests
### Example for True Updates

**R**

<table>
<thead>
<tr>
<th>Rule</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p(X,Y) \leftarrow c(X,Y), \text{not } b(X,Y)$.</td>
<td></td>
</tr>
<tr>
<td>$p(X,Y) \leftarrow p(X,Z), p(Z,Y)$.</td>
<td></td>
</tr>
</tbody>
</table>

**$R^\Delta$**

<table>
<thead>
<tr>
<th>Rule</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p^+(X,Y) \leftarrow c^+(X,Y)$, new, not $b(X,Y)$, old, not $p(X,Y)$.</td>
<td></td>
</tr>
<tr>
<td>$p^+(X,Y) \leftarrow b^-(X,Y)$, new, $c(X,Y)$, old, not $p(X,Y)$.</td>
<td></td>
</tr>
<tr>
<td>$p^-(X,Y) \leftarrow c^-(X,Y)$, old, not $b(X,Y)$, new, not $p(X,Y)$.</td>
<td></td>
</tr>
<tr>
<td>$p^-(X,Y) \leftarrow b^+(X,Y)$, old, $c(X,Y)$, new, not $p(X,Y)$.</td>
<td></td>
</tr>
<tr>
<td>$p^+(X,Y) \leftarrow p^+(X,Z)$, new, $p(Z,Y)$, old, not $p(X,Y)$.</td>
<td></td>
</tr>
<tr>
<td>$p^+(X,Y) \leftarrow p^+(Z,Y)$, new, $p(X,Z)$, old, not $p(X,Y)$.</td>
<td></td>
</tr>
<tr>
<td>$p^-(X,Y) \leftarrow p^-(X,Z)$, old, $p(Z,Y)$, new, not $p(X,Y)$.</td>
<td></td>
</tr>
<tr>
<td>$p^-(X,Y) \leftarrow p^-(Z,Y)$, old, $p(X,Z)$, new, not $p(X,Y)$.</td>
<td></td>
</tr>
</tbody>
</table>
True Updates – Example (cont’d)

Resolving the meta-predicates:

**R^A**

\[
\begin{align*}
p^+(X,Y) & \leftarrow c^+(X,Y), \not b^{\text{new}}(X,Y), \not p^{\text{old}}(X,Y). \\
p^+(X,Y) & \leftarrow b^-(X,Y), c^{\text{new}}(X,Y), \not p^{\text{old}}(X,Y). \\
p^-(X,Y) & \leftarrow c^-(X,Y), \not b^{\text{old}}(X,Y), \not p^{\text{new}}(X,Y). \\
p^-(X,Y) & \leftarrow b^+(X,Y), c^{\text{old}}(X,Y), \not p^{\text{new}}(X,Y). \\
p^+(X,Y) & \leftarrow p^+(X,Z), p^{\text{new}}(Z,Y), \not p^{\text{old}}(X,Y). \\
p^+(X,Y) & \leftarrow p^+(Z,Y), p^{\text{new}}(X,Z), \not p^{\text{old}}(X,Y). \\
p^-(X,Y) & \leftarrow p^-(X,Z), p^{\text{old}}(Z,Y), \not p^{\text{new}}(X,Y). \\
p^-(X,Y) & \leftarrow p^-(Z,Y), p^{\text{old}}(X,Z), \not p^{\text{new}}(X,Y). 
\end{align*}
\]

**R^C**

\[
\begin{align*}
p^{\text{new}}(X,Y) & \leftarrow c^{\text{new}}(X,Y), \not b^{\text{new}}(X,Y). \\
p^{\text{new}}(X,Y) & \leftarrow p^{\text{new}}(X,Z), p^{\text{new}}(Z,Y). \\
c^{\text{new}}(X,Y) & \leftarrow c^+(X,Y). \\
c^{\text{new}}(X,Y) & \leftarrow c^{\text{old}}(X,Y), \not c^-(X,Y). \\
b^{\text{new}}(X,Y) & \leftarrow b^+(X,Y). \\
b^{\text{new}}(X,Y) & \leftarrow b^{\text{old}}(X,Y), \not b^-(X,Y). \\
p^{\text{old}}(X,Y) & \leftarrow c^{\text{old}}(X,Y), \not b^{\text{old}}(X,Y). \\
p^{\text{old}}(X,Y) & \leftarrow p^{\text{old}}(X,Z), p^{\text{old}}(Z,Y).
\end{align*}
\]
Properties of this Approach to UP

- Each propagation rule includes a **delta literal** for restricting the evaluation to the changes induced for the respective body literal.

- A bottom-up materialization will nevertheless **completely** determine both, the **new** as well as **old** state of path.

- The situation would be different, if propagation were based on a top-down evolution techniques as e.g. in SQL.

- The proposed propagation rules still allow for **further** refinements, e.g.
  - Unnecessary effectiveness tests
  - Unnecessary rules for integrity checking
    (but check for empty delta relations avoids redundant rule applications)

- Propagation rules can be also further simplified if different **update granularities** (instead of true updates) are considered.