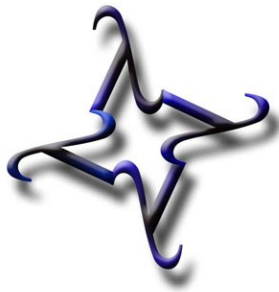


Repair guidance in a WYSIWYG data editor



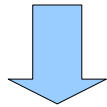
Student: Eric Kao

Advisor: Michael Genesereth

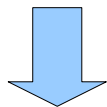
Logic Group

Traditional data editing

Make a series of changes
to the database



Commit changes



!
“Cannot commit because
of constraint violations”

- Hard to understand how to avoid constraint violation

Restricted Interface

- Ask a series of questions in a fixed order
- Disallow choices that lead to violations
- Inflexible
- Hard to “navigate” between states

DataWYZ

p	
aa	bb
bb	cc
cc	dd

q
a
b
c
e
d

r		
aa	bb	cc
bb	cc	dd

- Unrestricted editing
- Visually rendered violations
- Visual repair guidance

Formal Representation

- D : Data
 - e.g., { use(drug1), use(drug2), use(drug3),
~use(drug4) }
- C : Constraints
 - e.g., {
use(drug1) \Rightarrow use(drug2),
use(drug2) \Rightarrow ~use(drug3),
use(drug1) \Rightarrow ~use(drug3)
}

Violation Pinpointing

- Pinpointing: highlight every minimal subset D' of D such that D' violates C .
 - e.g.,
 $mv(D,C) = \{ \{ \text{use}(\text{drug1}), \text{use}(\text{drug3}) \} \}$
 - Note: $\{ \text{use}(\text{drug1}), \text{use}(\text{drug2}), \text{use}(\text{drug3}) \}$ is not minimal
- Naïve method has exponential running time
- Precompile constraints

Basic repair guidance

- Suggest which tuples to add or remove
- Suggest adding $\{\text{neg}(S) : S \in \text{mv}(D,C)\}$
- Suggest removing $\{\text{pos}(S) : S \in \text{mv}(D,C)\}$

The image displays two side-by-side screenshots of a database repair tool interface, showing logical constraints and a table of values.

Left Window (p):

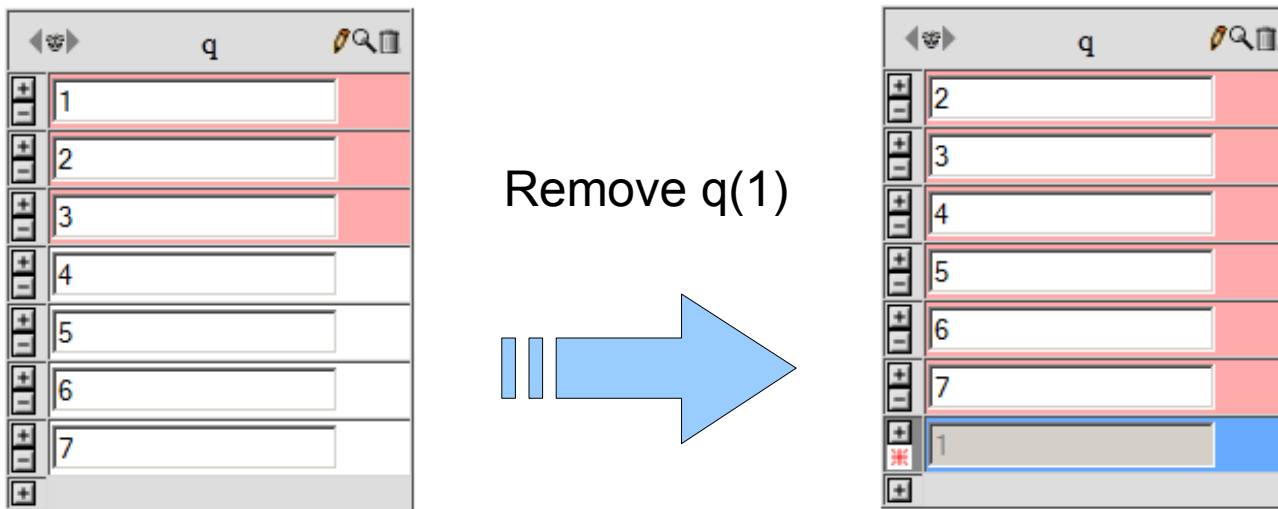
- Constraints:
 - $\text{true} \Rightarrow p(b) \mid p(c)$
 - $\text{true} \Rightarrow p(c) \mid p(d)$
 - $\text{true} \Rightarrow p(d) \mid p(e)$
 - $p(b) \ \& \ p(c) \ \& \ p(d) \Rightarrow \text{false}$
 - $p(b) \ \& \ p(c) \Rightarrow p(e)$
- Table:
 - Column 'a': empty
 - Column 'b': empty (highlighted in red)
 - Column 'c': empty (highlighted in red)
 - Column 'e': empty (highlighted in blue)
 - Column 'd': empty (highlighted in blue)

Right Window (q):

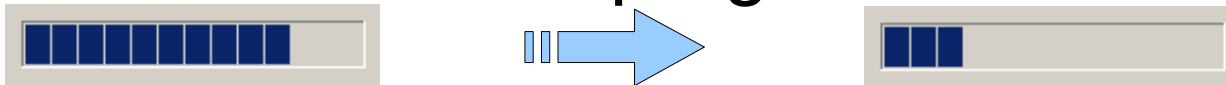
- Constraints:
 - $\text{true} \Rightarrow q(\text{drug1a}) \mid q(\text{drug1b})$
 - $\text{true} \Rightarrow q(\text{drug2a}) \mid q(\text{drug2b})$
 - $\text{true} \Rightarrow q(\text{drug3a}) \mid q(\text{drug3b})$
 - $q(\text{drug1a}) \ \& \ q(\text{drug2a}) \ \& \ q(\text{drug3a}) \Rightarrow \text{false}$
- Table:
 - Column 'drug2a': empty
 - Column 'drug3a': empty (highlighted in blue)
 - Column 'drug3b': empty (highlighted in blue)
 - Column 'drug1a': empty (highlighted in blue)
 - Column 'drug1b': empty (highlighted in blue)

Problem: non-monotone progress

- Choosing certain changes lead to “bigger” violations
 - $q(1) \ \& \ q(2) \Rightarrow \text{false}$
 - $q(2) \ \& \ q(3) \Rightarrow \text{false}$
 - $q(3) \ \& \ q(4) \ \& \ q(5) \ \& \ q(6) \ \& \ q(7) \Rightarrow q(1)$

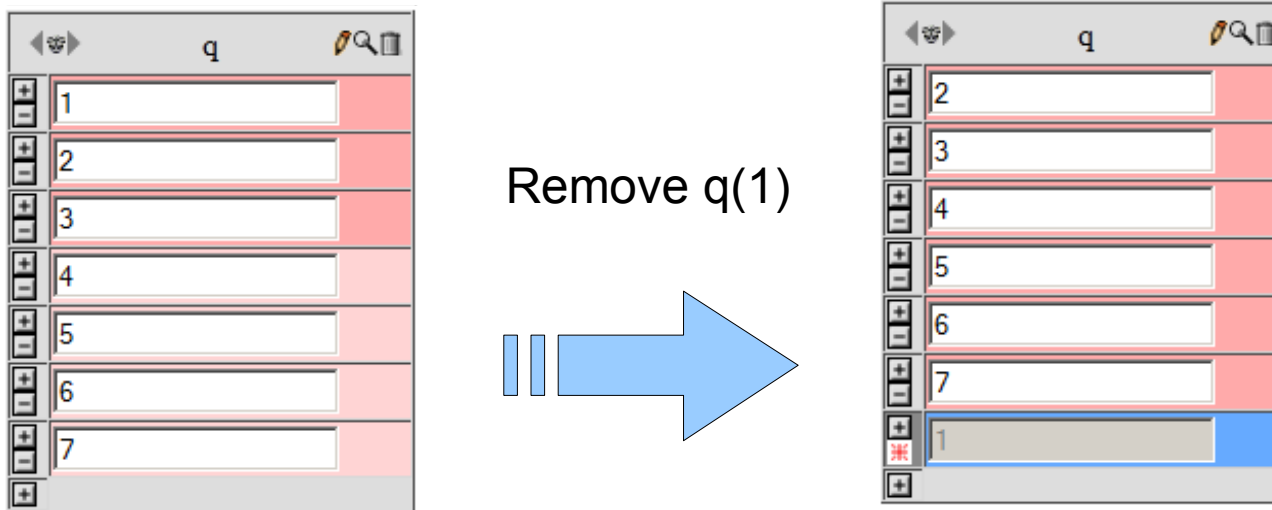


- No intuitive sense of progress.



Potentially affected

- Highlight all potentially affected tuples (PA)
- The set of highlighted tuple monotonically decrease

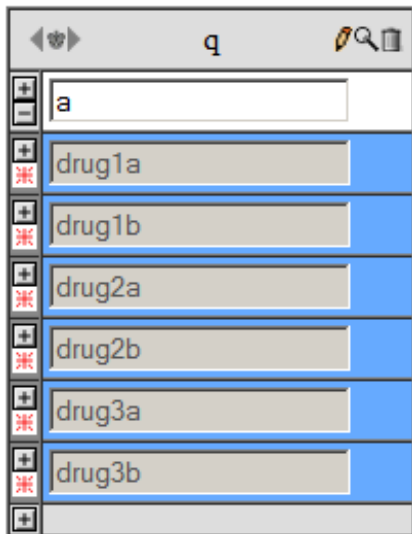


- Naïve procedure: simulate all repair choices
- Precompile constraints to compute PA quickly

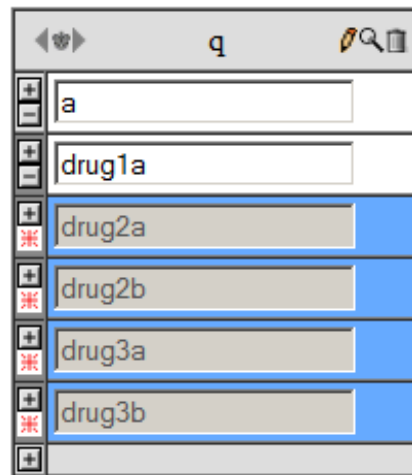
Back-tracking

- In following repair guidance, the user may make choices that

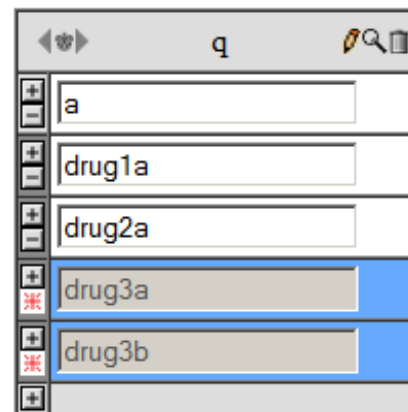
```
true => q(drug1a) | q(drug1b)
true => q(drug2a) | q(drug2b)
true => q(drug3a) | q(drug3b)
q(drug1a) & q(drug2a) & q(drug3a) => false
```



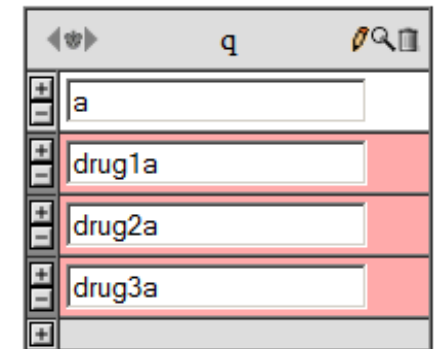
➡
Add drug1a



➡
Add drug2a



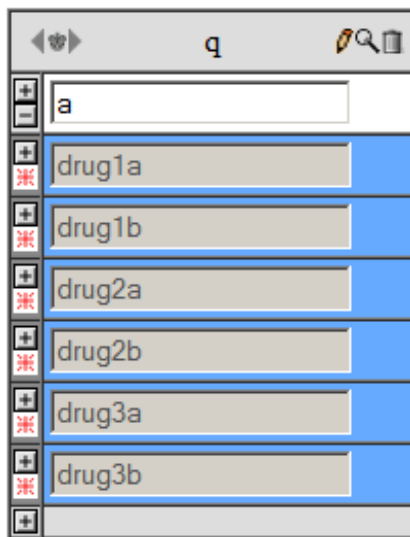
➡
Add drug3a



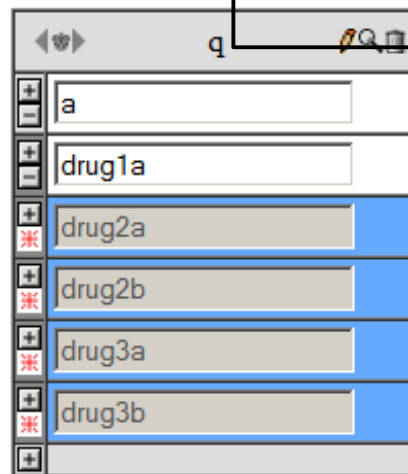
Back-track-free guidance

- When back-track free guidance is enabled, DataWYZ hides choices that would lead to back-tracking

$\text{true} \Rightarrow q(\text{drug1a}) \mid q(\text{drug1b})$
 $\text{true} \Rightarrow q(\text{drug2a}) \mid q(\text{drug2b})$
 $\text{true} \Rightarrow q(\text{drug3a}) \mid q(\text{drug3b})$
 $q(\text{drug1a}) \ \& \ q(\text{drug2a}) \ \& \ q(\text{drug3a}) \Rightarrow \text{false}$



|| →
Add drug1a



|| →
Add drug2a

