StreamClean: Near real-time RFID data cleaning

Nodira Khoussainova
Magdalena Balazinska
Dan Suciu
Ting-You Wang
and the RFID Ecosystem team
Applications frequently rely on devices such as:
  - RFID antennas
  - Light/motion/temperature sensors

However, these devices are unreliable
  - Readings can be missed, duplicated, or simply erroneous.
Our Approach

- User or application specifies integrity constraints
- Constraint violations → input data errors
- StreamClean cleans input data probabilistically
Outline

- Constraint Language
- Constraint Taxonomy
- Detecting and Handling Errors
- Higher Level Events
- Probabilistic Constraints
- Future Work
Constraint Language

FORALL  INPUT1 as I1,..., INPUTn as In
WHERE  EXPR1
CHECK  EXPR2
CONF   C
Example from RFID-based tracking application
Constraint: “An object can not appear in more than one location at any time”

```
FORALL Sightings S
CHECK NOT EXISTS Sightings S1
    WHERE S.obj_id = S1.obj_id
    AND S.antenna_id <> S1.antenna_id
```
Example from RFID-based tracking application
Constraint: “If an object is sighted at antenna A, and then later at C they should have been sighted at B at some point in between.”

\[
\text{FORALL } \text{Sightings } S_1, \text{Sightings } S_3
\]
\[
\text{WHERE } \text{SEQ}(S_1, S_3) \land S_1.\text{ant-id} = 'A' \land S_3.\text{ant-id} = 'C' \land S_1.\text{obj-id} = S_3.\text{obj-id}
\]
\[
\text{CHECK EXISTS } \text{Sightings } S_2
\]
\[
\text{WHERE } S_2.\text{obj-id} = S_1.\text{obj-id} \land S_2.\text{ant-id} = 'B' \land \text{ISLATER}(S_2, S_1) \land \text{ISLATER}(S_3, S_2)
\]
## Constraint Taxonomy

### Hard constraints
- Must always hold

### Soft constraints
- Usually hold

<table>
<thead>
<tr>
<th></th>
<th>Stateless</th>
<th>Stateful</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Exclusion</strong></td>
<td>Each object can appear in at most one location</td>
<td>A person who left the building cannot appear in the building</td>
</tr>
<tr>
<td><strong>Inclusion</strong></td>
<td>Each object must appear in at least one location</td>
<td>A person returning to the office must have previously passed by the elevator</td>
</tr>
<tr>
<td><strong>Stateless</strong></td>
<td>Stateless</td>
<td>Stateful</td>
</tr>
<tr>
<td><strong>Stateful</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Detecting and Handling Errors

- **Example 1:** An object appears in three locations
  - Constraint enables error detection
  - StreamClean assigns probability 1/3 to each sighting

- **Example 2:** An object is not sighted
  - Constraint enables error detection
  - StreamClean generates all $N$ possible sightings
    - In the absence of other information, all locations are possible
    - Additional, possibly soft, constraints help reduce space (e.g. Ex 2)
  - StreamClean assigns probability $1/N$ to each sighting
Higher Level Events

- Use constraint language to define higher level events
  E.g. “Seen at elevator immediately followed by at the door.”
  
  \[
  \text{FORALL Sightings } S_1, !\text{Sightings } S, \text{ Sightings } S_2 \\
  \text{WHERE } \text{SEQ}(S_1, S, S_2) \text{ AND } S_1.\text{obj-id} = S.\text{obj-id} \text{ AND } \\
  S_1.\text{obj-id} = S_2.\text{obj-id} \text{ AND } S_1.\text{ant-id} = \text{‘elevator1’} \text{ AND } \\
  S_2.\text{ant-id} = \text{‘door3’} \\
  \text{CHECK EXIST} \text{S LeftBldg L(} \text{obj-id)}
  \]

- Write constraints on these higher level events to clean the data further.
<table>
<thead>
<tr>
<th>FORALL</th>
<th>INPUT1 as I1, ..., INPUTn as In</th>
</tr>
</thead>
<tbody>
<tr>
<td>WHERE</td>
<td>EXPR1</td>
</tr>
<tr>
<td>CHECK</td>
<td>EXPR2</td>
</tr>
<tr>
<td>CONF</td>
<td>C</td>
</tr>
</tbody>
</table>
Probabilistic Constraints

- Confidence supplied for constraints
  - Can be specified by user/application
  - Can be learned from history
    - How often is this constraint correct?
    - Parameters still need to be specified by user/application
Future Work

- Preliminary results
  - Assigned probabilities match intuition
  - Entropy maximization sufficiently fast
    - Example: 100 equations, 20 var/equ. solved in 200ms
  - But integrity constraints can get quite complex
- StreamClean Jr – on traditional database
- Extend StreamClean Jr to StreamClean
- Integrate with the RFID eco system
Thank you!

Any questions?