Trio: A System for Data, Uncertainty, and Lineage

Jennifer Widom
Stanford University
Two faculty independently proclaim uncertainty as the next major theme in Computer Science

- One old-timer, one youngster
- Proclamations not motivated by their own (or our) research
Uncertainty in Databases

- Not a new idea — proposed 20+ years ago
- Most initial (18+ years) work largely theoretical; not much systems-building until recently
- But applications weren’t ready anyway
  ➔ Are they now?
The Trio database system, developed by Professor Jennifer Widom and her research team, can account for the uncertainty of data and its sourcing.
The “Trio” in Trio

1. Data
   Student #123 is majoring in Econ: \((123, \text{Econ}) \in \text{Major}\)

2. Uncertainty
   Student #123 is majoring in Econ or CS:
   \((123, \text{Econ} \parallel \text{CS}) \in \text{Major}\)
   With confidence 60% student #456 is a CS major:
   \((456, \text{CS 0.6}) \in \text{Major}\)

3. Lineage
   456 \in \text{HardWorker} derived from:
   \((456, \text{CS}) \in \text{Major}\)
   “CS is hard” \in \text{some web page}
The Trio database system, developed by Professor Jennifer Widom and her research team, can account for the uncertainty of data and its sourcing.
Original Motivation for the Project

New Applications

• Many involve data that is uncertain (approximate, probabilistic, inexact, incomplete, imprecise, fuzzy, inaccurate, ...)

• Many of the same ones need to track the lineage of their data

Coincidence or Fate?
Claim

The connection between uncertainty and lineage goes deeper than just a shared need by several applications

Coincidence or Fate?
Substantiation of Connection

[technical] **Lineage:**

1. Enables simple and consistent representation of uncertain data
2. Correlates uncertainty in query results with uncertainty in the input data
3. Can make computation over uncertain data more efficient

[fluffy] **Applications use lineage to reduce or resolve uncertainty**
Sample Applications

Information extraction
• Find & label entities in unstructured text
• Often probabilistic

Information integration
• Combine data from multiple sources
• Inconsistencies

Scientific experiments
• Inexact/incomplete data
• Many levels of “derived data products”
Sample Applications

Sensor data management
  • Approximate readings
  • Missing readings
  • Levels of data aggregation

Deduplication ("data cleaning")
  • Object linkage, entity resolution
  • Often heuristic/probabilistic

Approximate query processing
  • Fast but inexact answers
Why should every slide be making you squirm in your seat?
Our Goal

Develop a new kind of database management system (DBMS) in which:

1. Data
2. Uncertainty
3. Lineage

are all first-class interrelated concepts

⇒ With all the “usual” DBMS features
The “Usual” DBMS Features

(From first lecture of my Intro. to Databases class)

1. Efficient,
2. Convenient,
3. Safe,
4. Multi-User storage of and access to
5. Massive amounts of
6. Persistent data
In Standard Relational DBMS

Persistent; Convenient
- All data stored in simple tables ("relations")
- Queries and updates via simple but powerful declarative language (SQL)

Multi-User; Safe (from failures)
- Transactions

Massive; Efficient
- Storage and indexing structures
- Query optimization
**Example Database: Crime-Solving**

**Standard relational database**

<table>
<thead>
<tr>
<th>Saw (witness, color, car)</th>
<th>Drives (person, color, car)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amy</td>
<td>red</td>
</tr>
<tr>
<td>Betty</td>
<td>blue</td>
</tr>
<tr>
<td>Carol</td>
<td>green</td>
</tr>
<tr>
<td>Jimmy</td>
<td>red</td>
</tr>
<tr>
<td>Billy</td>
<td>blue</td>
</tr>
<tr>
<td>Frank</td>
<td>red</td>
</tr>
<tr>
<td>Frank</td>
<td>green</td>
</tr>
</tbody>
</table>

Create Table Suspects as
Select person
From Saw, Drives
Where Saw.color = Drives.color
And Saw.car = Drives.car

Suspects
Frank
Billy
1. **Data Model**
   Simplest extension to relational model that’s sufficiently expressive

2. **Query Language**
   Simple extension to SQL with well-defined semantics and intuitive behavior

3. **System**
   A complete open-source DBMS that people want to use
Another “Trio” in Trio

1. Data Model
   \textit{Uncertainty-Lineage Databases (ULDB’s)}

2. Query Language
   \textit{TriQL}

3. System
   \textit{Trio-One}: built on top of standard DBMS
Remainder of Talk

1. **Data Model**
   In some detail

2. **Query Language**
   In somewhat less detail

3. **System**
   Overview + Demo
Disclaimer

We are not about machine learning or probabilistic reasoning!

We are about efficient and convenient storage, manipulation, and retrieval of large data sets (with uncertainty and lineage in them)
An uncertain database represents a set of possible instances

- Amy saw either a Honda or a Toyota
- Jimmy drives a Toyota, a Mazda, or both
- Betty saw an Acura with confidence 0.5 or a Toyota with confidence 0.3
- Hank is a suspect with confidence 0.7
Our Model for Uncertainty

1. Alternatives
2. ‘?’ (Maybe) Annotations
3. Confidences
Our Model for Uncertainty

1. **Alternatives:** uncertainty about value

2. ‘?’ (Maybe) Annotations

3. Confidences

<table>
<thead>
<tr>
<th>Saw (witness, color, car)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amy</td>
</tr>
</tbody>
</table>

Three possible instances
Our Model for Uncertainty

1. Alternatives
2. ‘?’ (Maybe): uncertainty about presence
3. Confidences

<table>
<thead>
<tr>
<th>Saw (witness, color, car)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amy</td>
</tr>
<tr>
<td>Betty</td>
</tr>
</tbody>
</table>

Six possible instances
1. Alternatives

2. ‘?’ (Maybe): uncertainty about presence

3. Confidences

---

<table>
<thead>
<tr>
<th>Saw (witness, color, car)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Amy</strong></td>
</tr>
<tr>
<td>*red, Honda</td>
</tr>
<tr>
<td><strong>Betty</strong></td>
</tr>
<tr>
<td><em>blue, Acura</em></td>
</tr>
<tr>
<td><strong>Betty</strong></td>
</tr>
<tr>
<td>*blue, Acura</td>
</tr>
</tbody>
</table>
Our Model for Uncertainty

1. Alternatives

2. ‘?’ (Maybe) Annotations

3. Confidences: weighted uncertainty

<table>
<thead>
<tr>
<th>Saw (witness, color, car)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Amy</strong></td>
</tr>
<tr>
<td>*red, Honda 0.5</td>
</tr>
<tr>
<td><strong>Betty</strong></td>
</tr>
<tr>
<td><em>blue, Acura 0.6</em></td>
</tr>
</tbody>
</table>

Six possible instances, each with a probability
Models for Uncertainty

Our model (so far) is not especially new

We spent some time exploring the space of models for uncertainty

Tension between understandability and expressiveness

– Our model is understandable
– But it is not complete, or even closed under common operations
Closure and Completeness

Completeness
Can represent all sets of possible instances

Closure
Can represent results of operations

Note: Completeness $\Rightarrow$ Closure
Our Model is Not Closed

<table>
<thead>
<tr>
<th>Saw (witness, car)</th>
<th>Drives (person, car)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cathy</td>
<td>Jimmy, Toyota</td>
</tr>
<tr>
<td></td>
<td>Billy, Honda</td>
</tr>
<tr>
<td></td>
<td>Hank, Honda</td>
</tr>
</tbody>
</table>

Suspects = \( \pi_{\text{person}}(\text{Saw Drives}) \)

CANNOT correctly capture possible instances in the result
Lineage to the Rescue

Lineage (provenance): “where data came from”
- Internal lineage
- External lineage

In Trio: A function $\lambda$ from data elements to other data elements (or external sources)
**Example with Lineage**

<table>
<thead>
<tr>
<th>ID</th>
<th>Saw (witness, car)</th>
<th>Drives (person, car)</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>Cathy Honda</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td></td>
<td>Billy, Honda</td>
</tr>
<tr>
<td>23</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[ \text{Suspects} = \pi_{\text{person}}(\text{Saw Drives}) \]

<table>
<thead>
<tr>
<th>ID</th>
<th>Suspects</th>
<th>( \lambda )</th>
</tr>
</thead>
<tbody>
<tr>
<td>31</td>
<td>Jimmy</td>
<td>( \lambda(31) = (11,2),(21,2) )</td>
</tr>
<tr>
<td>32</td>
<td>Billy</td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>Hank</td>
<td>( \lambda(33) = (11,1),23 )</td>
</tr>
</tbody>
</table>

Correctly captures possible instances in the result
Example with Lineage

<table>
<thead>
<tr>
<th>ID</th>
<th>Saw (witness, car)</th>
<th>Drives (person, car)</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>Cathy, Honda</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>23</td>
</tr>
</tbody>
</table>

Suspects = \( \prod_{\text{person}} \) \( \text{Saw} \) \( \text{Drives} \)
Trio Data Model

1. Alternatives
2. ‘?’ (Maybe) Annotations
3. Confidences
4. Lineage

**Uncertainty-Lineage Databases (ULDBs)**

ULDBs are closed and complete
Minimal representation of possible-instances

- Data-minimal
- Lineage-minimal

Lineage expressiveness

- Arbitrary sequence of queries \( \rightarrow \) arbitrary boolean formula

Membership questions (complexity)

- Is tuple \( T \) in any (every) possible instance?
- Is relation \( R \) a possible-instance?
Querying ULDBs

- Simple extension to SQL
- Formal semantics, intuitive meaning
- Query uncertainty, confidences, and lineage
## Simple TriQL Example

<table>
<thead>
<tr>
<th>ID</th>
<th>Saw (witness, car)</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>Cathy Honda</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ID</th>
<th>Drives (person, car)</th>
</tr>
</thead>
<tbody>
<tr>
<td>21</td>
<td>Jimmy, Toyota</td>
</tr>
<tr>
<td>22</td>
<td>Billy, Honda</td>
</tr>
<tr>
<td>23</td>
<td>Hank, Honda</td>
</tr>
</tbody>
</table>

Create Table Suspects as
Select personFrom Saw, DrivesWhere Saw.car = Drives.car

<table>
<thead>
<tr>
<th>ID</th>
<th>Suspects</th>
</tr>
</thead>
<tbody>
<tr>
<td>31</td>
<td>Jimmy</td>
</tr>
<tr>
<td>32</td>
<td>Billy</td>
</tr>
<tr>
<td>33</td>
<td>Hank</td>
</tr>
</tbody>
</table>

\[ \lambda(31) = (11,2), (21,2) \]
\[ \lambda(32,1) = (11,1), (22,1); \lambda(32,2) = (11,1), (22,2) \]
\[ \lambda(33) = (11,1), 23 \]
Formal Semantics

Query $Q$ on ULDB $D$

- $D_1, D_2, ..., D_n$ possible instances
- $Q(D_1), Q(D_2), ..., Q(D_n)$ $Q$ on each instance
- $D + Result$ representation of instances
- implementation of $Q$ operational semantics
TriQL

SQL over uncertain data with lineage

+ New constructs for querying alternatives, ?’s, confidences, and lineage

• Conf(), Maybe(), Lineage()
• “Horizontal” subqueries
• Query-defined result confidences
• Aggregations: low, high, expected
Confidences supplied with base data

Trio computes confidences on query results

- Default probabilistic interpretation
- Can choose to plug in different arithmetic

### Saw (witness, car)

<table>
<thead>
<tr>
<th>Witness</th>
<th>Car</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cathy</td>
<td>Honda 0.6</td>
</tr>
</tbody>
</table>

### Drives (person, car)

<table>
<thead>
<tr>
<th>Person</th>
<th>Car</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jimmy</td>
<td>0.3</td>
</tr>
<tr>
<td>Hank</td>
<td>1.0</td>
</tr>
<tr>
<td></td>
<td>Honda</td>
</tr>
</tbody>
</table>

### Suspects

<table>
<thead>
<tr>
<th>Person</th>
<th>Confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jimmy</td>
<td>0.3</td>
</tr>
<tr>
<td>Hank</td>
<td>0.6</td>
</tr>
</tbody>
</table>
One More Example Query

PrimeSuspect (crime#, suspect, accuser)

|   | Jimmy, Amy || Billy, Betty || Hank, Cathy |
|---|-------------|
| 1 |             |
| 2 | Frank, Cathy || Freddy, Betty |

Credibility (person, score)

<table>
<thead>
<tr>
<th></th>
<th>Amy</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Betty</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>Cathy</td>
<td>5</td>
</tr>
</tbody>
</table>

List suspects with conf values based on accuser credibility

<table>
<thead>
<tr>
<th>Suspects</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Jimmy 0.33</td>
<td>Billy 0.5</td>
<td></td>
</tr>
<tr>
<td>Frank 0.25</td>
<td>Freddy 0.75</td>
<td></td>
</tr>
</tbody>
</table>
One More Example Query

PrimeSuspect (crime#, suspect, accuser)

|   | Jimmy, Amy || Billy, Betty || Hank, Cathy |
|---|-------------|
| 1 |             |
| 2 | Frank, Cathy || Freddy, Betty |

Credibility (person, score)

<table>
<thead>
<tr>
<th></th>
<th>Amy</th>
<th>Betty</th>
<th>Cathy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>10</td>
<td>15</td>
<td>5</td>
</tr>
</tbody>
</table>

SELECT suspect, score/[sum(score)] as conf
FROM (SELECT suspect,
  (SELECT score FROM Credibility C
   WHERE C.person = P.accuser)
  FROM PrimeSuspect P)

Suspects

|    | Jimmy 0.33 || Billy 0.5 || Hank 0.166 |
|----|-------------|
|    |             |
|    | Frank 0.25 || Freddy 0.75 |
Trio System: Version 1

- **Trio API and translator (Python)**
- **TrioExplorer (GUI client)**
- **Command-line client**

**Standard relational DBMS**

- **Standard SQL**
- **Trio Stored Procedures**
- **Encoded Data Tables**
- **Lineage Tables**
- **Trio Metadata**

**Features**

- Partition and “verticalize”
- Shared IDs for alternatives
- Columns for confidence, “?”
- One per result table
- Uses unique IDs
- Encodes formulas

**Functionalities**

- DDL commands
- TriQL queries
- Schema browsing
- Table browsing
- Explore lineage
- On-demand confidence computation

- Table types
- Schema-level lineage structure
- conf()
- lineage()
System Issue: Confidence Computation

Previous approach (probabilistic databases)

- Each operator computes confidences during query execution
- *Restricts allowable query execution strategies*

In Trio

Confidence of data element $d$ is function of confidences in $\lambda^*(d)$
Confidence Computation (cont’d)

Our approach

• Use any query execution strategy
• Compute confidences on-demand based on lineage
• Some optimizations
  – “Independent lineage subtrees”
  – Memoization
  – Batch computation
Current Topics (sample)

More forms of uncertainty
- Continuous uncertainty (intervals, Gaussians)
- Correlated uncertainty
- Incomplete relations

More forms of lineage
- External lineage
- Update lineage

Confidence-based queries
- Threshold; “Top-K”
Design theory
  • Dependencies, normal forms, decomposition
  • New definitions, twists, and challenges

System
  • Version 2: Go native?
    – Storage and indexing structures
    – Statistics
    – Query optimization
Students...

Don’t be afraid to build systems

– Time-consuming
– Frustrating
– Where’s the paper?
+ Rewarding
+ Visible
+ Long-term impact
+ Here’s the demo! (and the papers)
Search “stanford trio”

Trio contributors, past and present

Parag Agrawal, Omar Benjelloun, Ashok Chandra, Anish Das Sarma, Alon Halevy, Chris Hayworth, Ander de Keijzer, Raghotham Murthy, Michi Mutsuzaki, Shubha Nabar, Tomoe Sugihara, Martin Theobald