Approximating Certain and Possible Answers beyond Sets

Logic and Algebra for Query Evaluation, Logic and Algorithms in Database Theory and AI

Boris Glavic[1]

DBGroup

Illinois Institute of Technology

2023-11-14

Outline

ILLINOIS INSTITUTE OF TECHNOLOGY

Incomplete & Probabilistic Databases

- Principled techniques for managing uncertain data
- High computational complexity for query answering
  - Many tractable cases based on query / data structure are known [DS12, DS07, FO16, FHO12, RS09, ABS15]
  - **intractable** even for relatively simple queries

Definition (Incomplete Database)

An incomplete database is a set $\mathcal{D} = \{D_1, \ldots D_n\}$ where each D_i, called a *possible world*, is a deterministic database.

Example

D_1

name	country
Boris	USA
Peter	USA
Alexandra	USA

D_2

name	country
Boris	Switzerl.
Alexandra	USA

D_3

name	country
Boris	Germany
Peter	Netherl.
Alexandra	USA

Definition (Possible World Semantics)

- **Incomplete Databases**

$$Q(\mathcal{D}) = \{Q(D) \mid D \in \mathcal{D}\}$$

Possible & Certain Tuples

Definition (Possible Tuples)

$$Possible(\mathcal{D}) = \{t \mid \exists D \in \mathcal{D} : t \in D\}$$

Definition (Certain Tuples)

$$Certain(\mathcal{D}) = \{t \mid \forall D \in \mathcal{D} : t \in D\}$$

Definition (Possible & Certain Answers)

$$Certain(Q, \mathcal{D}) = Certain(Q(\mathcal{D}))$$

$$Possible(Q, \mathcal{D}) = Possible(Q(\mathcal{D}))$$

- **A practical data model for uncertain data management**
  1. Supports **bags** (in addition to sets)
  2. **PTIME data complexity** (through **approximation**)
  3. Support for **complex queries** (full relational algebra with aggregation, sort-based operations, recursion)
  4. **Modular**: closed under queries
  5. **Compact**: be able to trade size for approximation accuracy
  6. **Compatible**: approximate existing uncertain data models and data cleaning paradigms (consistent query answering)

Outline

ILLINOIS INSTITUTE OF TECHNOLOGY

- **Incomplete Databases beyond sets**
  - Incomplete bag databases
  - Incomplete provenance
  - Incomplete temporal databases
  - Incomplete access control
  - ...

K-relations

ILLINOIS INSTITUTE OF TECHNOLOGY

Semirings

- **semiring** $(K, +_K, \cdot_K, 0_K, 1_K)$
- a set K
- binary operations $+_K$ and $\cdot_K$ which are associative and commutative
- 0_K (1_K) are the neutral elements of $+_K$ and $\cdot_K$
- $k \cdot_K 0_K = 0_K$ for all $k \in K$
- $k_1 \cdot_K (k_2 +_K k_3) = (k_1 \cdot_K k_2) + (k_1 \cdot_K k_3)$

Semiring-annotated relations

- An n-ary K-relation R is a function $\mathbb{U}^n \rightarrow K$ with finite support: $\{t \mid R(t) \neq 0_K\}$ ([GKT07, GT17])
- Tuples that are annotated with 0_K do not exist (omit them)

ILLINOIS INSTITUTE OF TECHNOLOGY

Natural order

- $k_1 \leq_K k_2$ if exists k_3 such that $k_1 + k_3 = k_2$
- Semirings where the **natural order** has a lattice structure are called **l-semirings** [KB12]
  - Each pair of elements k_1 and k_2 has ...
    - a greatest lower bound $\sqcap_K(k_1, k_2)$
    - a least upper bound $\sqcup_K(k_1, k_2)$

Natural order of $\mathbb{N}$

- $k_1 \leq_{\mathbb{N}} k_2$ is the standard order of natural numbers (e.g., $2 < 3$)
- $\sqcap_{\mathbb{N}}(k_1, k_2) = min(k_1, k_2)$
- $\sqcup_{\mathbb{N}}(k_1, k_2) = max(k_1, k_2)$

Definition (Incomplete K-database)

An **incomplete** K**-database** is a set $\mathcal{D} = \{D_1, \ldots, D_n\}$ such that each D_i is a K-database.

Possible world semantics

- Possible world semantics uses K-relational query semantics

Certain and possible answers

- Consider only **l-semirings**
- The **certain annotation** of a tuple
  - $Cert_K(\mathcal{D}, t) = \sqcap_K \{D_i(t) \mid D_i \in \mathcal{D}\}$
- The **possible annotation** of a tuple
  - $Poss_K(\mathcal{D}, t) = \sqcup_K \{D_i(t) \mid D_i \in \mathcal{D}\}$
- Coincides with certain / possible tuples for sets / bags

Bag Semantics Example ($\mathbb{N}$ annotations)

- Annotations encode the multiplicity of tuples under bag semantics
- Possible annotation is `max`, certain annotation is `min`
- The certain (possible) annotation of $(Peter, USA)$ is 2 (3)
- The certain (possible) annotation of $(Boris, USA)$ is 0 (2)

D_1

name	country	$\mathbb{N}$
Boris	USA	2
Peter	USA	3

D_2

name	country	$\mathbb{N}$
Peter	USA	2

Outline

ILLINOIS INSTITUTE OF TECHNOLOGY

Definition (Containment)

For two K-database D_1 and D_2 over the same schema, D_1 is contained in D_2 ($D_1 \sqsubseteq_K D_2$) iff

$$\forall t : D_1(t) \leq_K D_2(t)$$

Definition (Under-approximating certain annotations)

A K-database D is an **under-approximation** of an incomplete K-database $\mathcal{D}$ iff:

$$D \sqsubseteq_K Certain(\mathcal{D})$$

Definition (Over-approximating possible annotations)

A K-database D is an **over-approximation** of an incomplete K-database $\mathcal{D}$ iff:

$$D \sqsupseteq_K Possible(\mathcal{D})$$

Example TIDB

Definition (Tuple-independent database (TIDB))

A tuple-independent database $\boldsymbol{D} = D_{certain} \cup D_{possible}$ encodes an incomplete $\mathbb{B}$-database with the possible worlds $Mod(\boldsymbol{D})$:

$$Mod(\boldsymbol{D}) = \{D' \mid D_{certain} \sqsubseteq_{\mathbb{B}} D' \wedge D' \sqsubseteq_{\mathbb{B}} \boldsymbol{D}\}$$

Approximating TIDBs

- **Under-approximation**: $D = D_{certain}$
- **Over-approximation**: $D = \boldsymbol{D}$

Key repair

Consider a $\mathbb{B}$-relation $R(A_1, \ldots, A_n)$ with a key $K \subseteq \{A_1, \ldots, A_n\}$ and the incomplete database of all S-repairs:

$$\mathcal{R} = \{R' \mid R' \subseteq R \wedge \forall k \in \pi_K(R) : \exists t \in R' : t.K = k \\ \wedge \nexists t_1 \neq t_2 \in R' : t_1.K = t_2.K\}$$

Approximating key repairs

- **under-approximation**:

$$R' = \{t \mid R(t) = \top \wedge \nexists t' \neq t : t.K = t'.K \wedge R(t') = \top\}$$

- **over-approximation**:

$$R' = R$$

Definition (Uncertainty-annotated databases)

- A UA-DB $\boldsymbol{D}$ is a K_{AU}-relation where K_{AU} is the product semiring $K \times K$ restricted to $\{(k_l, k_u) \mid k_l \leq_K k_2\}$
- Addition and multiplication in K^2 are defined point-wise
- Define $(c,p)^{\downarrow} = c$ and $(c,p)^{\uparrow} = p$ and lift this operation pointwise to databases
  - $\boldsymbol{D}^{\downarrow}(t) = \boldsymbol{D}(t)^{\downarrow}$ and $\boldsymbol{D}^{\uparrow}(t) = \boldsymbol{D}(t)^{\uparrow}$

Proposition (Structure K_{AU} is a semiring)

For any naturally-ordered semiring K, K_{AU} is a semiring

Bounding incomplete K-databases

An K-UA-DB $\boldsymbol{D}$ **bounds** an incomplete K-database $\mathcal{D}$ ($\boldsymbol{D} \leqslant_K \mathcal{D}$) iff:

$$\boldsymbol{D}^{\downarrow} \sqsubseteq_K Certain(\mathcal{D})$$
$$\boldsymbol{D}^{\uparrow} \sqsupseteq_K Possible(\mathcal{D})$$

Bound-preserving query semantics

A query semantics for K_{AU} for is called **bound-preserving** iff for any query Q:

$$\boldsymbol{D} \leqslant_K \mathcal{D} \Rightarrow Q(\boldsymbol{D}) \leqslant_K Q(\mathcal{D})$$

Bound-preserving Queries

Theorem (Positive relational algebra is bound-preserving)

For any positive relational algebra query Q, standard K-relational query semantics is bound preserving.

Proof Sketch.

Addition and multiplication are monotone wrt. the natural order. If $k_1 \leq_K k_3$ and $k_2 \leq_K k_4$ then:

- $k_1 +_K k_2 \leq_K k_3 +_K k_4$
- $k_1 \cdot_K k_2 \leq_K k_3 \cdot_K k_4$

$\square$

Theorem (Full relational algebra is bound-preserving)

Full relational algebra is bound-preserving assuming the monus-semiring definition from [GP10] and by defining:

$$(\boldsymbol{R} - \boldsymbol{S})(t) = (\boldsymbol{R}(t)^{\downarrow} -_K \boldsymbol{S}(t)^{\uparrow}, \boldsymbol{R}(t)^{\uparrow} -_K \boldsymbol{S}(t)^{\downarrow})$$

Theorem (Bound preservation)

Recursive Datalog queries over UA-DBs are bound preserving.

Theorem (Convergence)

As computations in $\boldsymbol{D}^{\downarrow}$ and $\boldsymbol{D}^{\uparrow}$ are standard K-relational query evaluation, existing results for Datalog over K-relations are applicable. Specifically, we get the same convergence guarantees as proven for K-relations in [KNP+22]

Consistent query answering & certain answers

- The following approach is bound preserving:
  - Answer a supported subquery of a query Q using an existing technique for certain / possible answers
  - Interpret the result as a UA-DB
  - Evaluate the remainder of the query using UA-DBs

Outline

Motivation

- **Certain answers are often empty**
  - *e.g., operations like aggregation that return different aggregation function results in each world*
- **Some types of uncertainty cannot be efficiently over-approximated with *"concrete"* tuples**
  - This person certainly exists, but their salary is in `[100000,200000]`
- **Operations like `sum` aggregation may produce exponentially many possible results**

Related work

- **Representation systems that use (labelled) nulls**
  - C-tables [ILJ84], m-tables [SKL+17], Codd-tables, V-tables, ...
- ***"Factorized"* representations**
  - OR-tables [IVV95], x-tables (block-independent databases) [BSHW06, VdBS17]
- **Returning the lowest and highest possible aggregation function result**
  - [ABC+03, Fux07, ACK+10, LSV02, AK08]
- **Abstract interpretation**
  - [CC77, CC04]

Interval domains

- Domain $(\mathbb{D}, \leq_{\mathbb{D}})$ where $\leq_{\mathbb{D}}$ is a partial order
- Interval domain $\mathbb{D}^{\leq} = \{[l, u] \mid l, u \in \mathbb{D} \land l \leq_{\mathbb{D}} u\}$
- Representation system $Mod([l, u]) = \{c \mid c \in [l, u]\}$

Over-approximating operations

- $[l_1, u_1] + [l_2, u_2] := [l_1 + l_2, u_1 + u_2]$
- $[l_1, u_1] = [l_2, u_2] := [l_1 = u_1 = l_2 = u_2, l_1 \leq u_2 \land l_2 \leq u_1]$

The AU-DB Model

AU-DB relations

- Special type of K-relations
  - Value domains are intervals
  - Annotation domain is a UA-semiring K_{AU}

The possible worlds encoded by a UA-DB

- A UA-DB $\boldsymbol{D}$ encodes as possible worlds $Mod(\boldsymbol{D})$ every K-database D such that we can "*redistribute*" the annotations of tuples from D to bounding tuples from $\boldsymbol{D}$ such that
  - for any tuple $\boldsymbol{t}$ from $\boldsymbol{D}$, the total annotation mass distributed to $\boldsymbol{t}$ is withing its annotation bounds

UA-DB Example

UA-DB

name	salary	$\mathbb{N}^3$
Boris	[120k,120k]	[0,2]
Peter	[140k,400k]	[2,3]

Two possible worlds

name	salary	$\mathbb{N}$
Boris	120k	1
Peter	150k	1
Peter	380k	2

name	salary	$\mathbb{N}$
Peter	180k	2

Bounding incomplete databases

- Given an incomplete $\mathbb{N}$ database $\mathcal{D}$ and selected world $D_{sg} \in \mathcal{D}$
- Create UA-DB $\boldsymbol{D}$ with selected-guess D_{sg} such that $Mod(\boldsymbol{D}) \supseteq Mod(\mathcal{D})$

Incomplete bag database

D_1

name	salary	$\mathbb{N}$
Boris	120k	2
Peter	400k	3

D_2

name	salary	$\mathbb{N}$
Peter	140k	2

A Bounding UA-DB for SG D_1

name	salary	$\mathbb{N}^3$
Boris	[120k,120k]	[0,2]
Peter	[140k,400k]	[2,3]

Bounding

- Given a UA-DB $\boldsymbol{D}$ that bounds an incomplete K-database $\mathcal{D}$

Under-approximation of certain tuples with attribute uncertainty

- The lower bound tuple annotations of $\boldsymbol{D}$
  - Under-approximation of certain tuples
- Generalizes certain answers by allowing attribute-level uncertainty

Over-approximation of possible tuples with attribute uncertainty

- The upper bound tuple annotations of $\boldsymbol{D}$
  - Over-approximation of possible tuples
- Generalizes possible answers by allowing attribute-level uncertainty

Queries over *K*-relations

- Defined using the semiring addition / multiplication operations
- Disjunctive use of tuples (union, projection) is addition
- Conjunctive use of tuples (join) is multiplication

$\mathcal{RA}^+$ Semantics

$$\begin{aligned}
\textbf{Union:}\quad & (R_1 \cup R_2)(t) = R_1(t) +_{\mathcal{K}} R_2(t) \\
\textbf{Join:}\quad & (R_1 \bowtie R_2)(t) = R_1(t[\mathrm{Sch}(R)_1]) \cdot_{\mathcal{K}} R_2(t[\mathrm{Sch}(R)_2]) \\
\textbf{Projection:}\quad & (\pi_U(R))(t) = \sum_{t=t'[U]} R(t') \\
\textbf{Selection:}\quad & (\sigma_\theta(R))(t) = R(t) \cdot_{\mathcal{K}} \theta(t)
\end{aligned}$$

Conditional expressions over range-annotated values

- Expressions over range-annotated values evaluate to a range of Boolean values
  - e.g., $[1,3] = [3,3]$ evaluates to $[F, T]$

Evaluating selections

- in K-relations selection on θ
  - *false* maps to 0_K and *true* to 1_K
  - tuples get their input annotation if they do fulfill θ
  - tuples get a 0_K annotation in the result if they do not fulfill θ
- in K_{AU} relations: apply this to every dimension
  - e.g., $[F, T]$ maps to $[0, 1]$

Semi-module Aggregation?

Semi-modules

- Polynomial representation system for aggregation results [ADT11]

Proposition (SUM_I, MAX_I, and MIN_I are monoids)

- *using our definitions of addition, min, max over the interval domain*

Proposition (Semi-modules cannot be bound preserving)

- *Use* $\mathbb{N}_{AU}$ *and* SUM_I*, set* $k = (1, 2)$ *and* $m = [0, 0]$ *and* $m_1 = [-1, -1]$ *and* $m_2 = [1, 1]$
- *Observe that* $m = m_1 + m_2$
- *Based on semi-module laws we have to have* $k \otimes m = k \otimes (m_1 + m_2) = k \otimes m_1 + k \otimes m_2$
- *However,* $k \otimes m_2 = (l_2, u_2)$ *with* $l_2 \leq 1 < 2 \leq u_2$

Aggregation over AU-DBs

ILLINOIS INSTITUTE OF TECHNOLOGY

Escape into uncertainty

- Use bound-preserving multiplication of intervals instead of a semimodule
- $k \circledast m = k \cdot m$

Approach

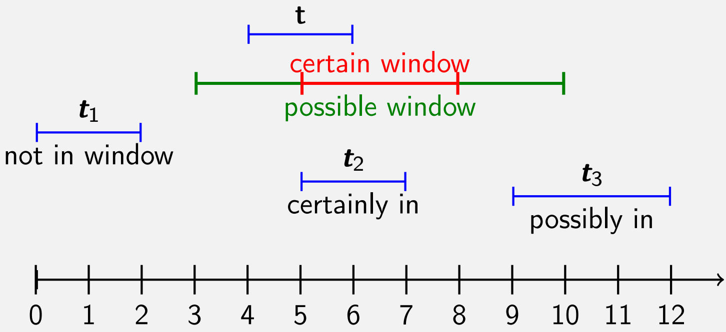
- Group-by attribute bounds in the result determined based on assigning tuples to groups based on their selected-guess values
- Aggregation function result bounds by reasoning about
    - Which tuples could / have to belong to a group
    - Taking the lowest / highest possible multiplicities of tuples into account
    - Using lowest / highest possible values of the attribute we are aggregating over
    - Utilize expression semantics for range-annotated values

Sort Order as Data

- there is only one physical sort order
- encode sort position as UA-DB range-values!

Windowed Aggregation

- exploit fixed window sizes



Outline

ILLINOIS INSTITUTE OF TECHNOLOGY

1. Incomplete Databases, Certain & Possible Answers
2. Incomplete K-relations
3. Approximating Incomplete K-relations
4. Attribute-level Uncertainty
5. **Conclusions & Future Work**
6. Appendix and References

UA-DB Data Model

- **Lightweight, but powerful uncertain data model**
- Under-approximation of **certain answers**
- Over-approximation of **possible answers**

Query semantics

- **Bound preservation**: the bounding of certain and possible answers is preserved under queries
- Ranges (attribute-level uncertainty) as first-class citizens: generalizes
  - certain answers with nulls [LJ84]
  - "bounding range semantics" for aggregation over incomplete data

Future Work

Even more queries

- Recursive and iterative computations *(e.g., training ML models)*

Even better performance

- New specialized physical operators
- Optimizing representations and queries

Better bounds

- Using more expressive set representations instead of boxes

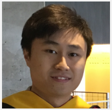
Beyond ranges

- Partial orders / Ontologies instead of ranges

Questions?

Implementations

- **GProM** `https://github.com/IITDBGroup/gprom`
- **Vizier** `https://vizierdb.info/`



Su Feng

Oliver Kenndy

Aaron Huber

Outline

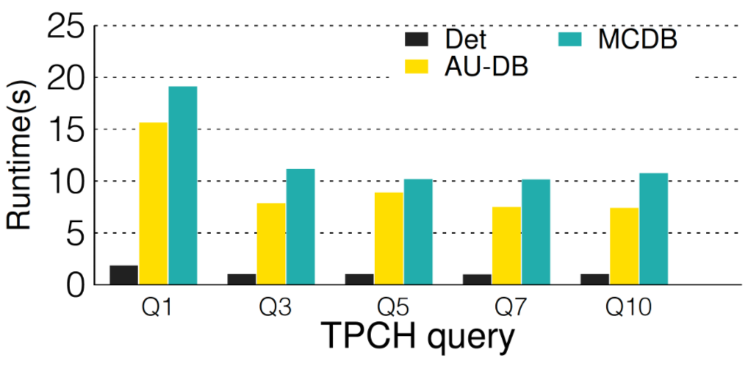


Figure: TPC-H (SF=1)

Competitors

- UA-DB
  - `UA-DB`: no over-approximation of possible and no attribute-uncertainty
  - `AU-DB`: the model discussed so far
- Under-approximation of certain for DBs with nulls (Libkin, [Lib16])
- Sampling-based (*MCDB* w/o probabilities, [JXW+08])
- All possible answers
  - *MayBMS* w/o probabilities[a], [AKO07, OHK09])
  - *Trio* w/o probabilities[b], [BSHW06]
  - *Semi-modules* w/o probabilities [FHO12]

[a] Does not support aggregation

[b] Does not support uncertain group-by and querying aggregation results

Datasets

- *PD-Bench* (TPC-H data with errors), [AJKO08]
- Real-world open datasets with missing values

Machines

- 2×6 core AMD Opteron 4238 CPUs
- 128GB RAM
- 4×1TB 7.2K HDDs (RAID 5)
- Postgres 11

Experimental Results - SPJ

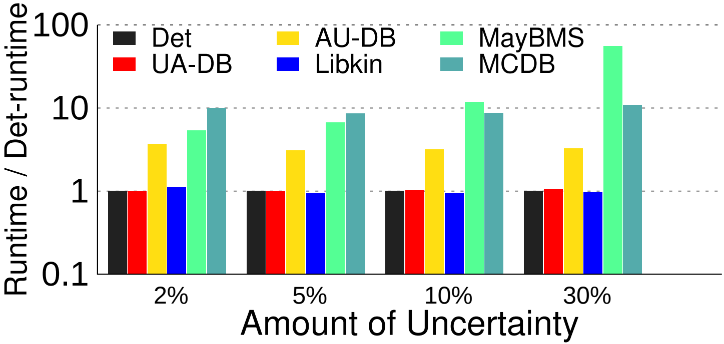


Figure: PD-Bench Queries (1GB) - Varying input uncertainty

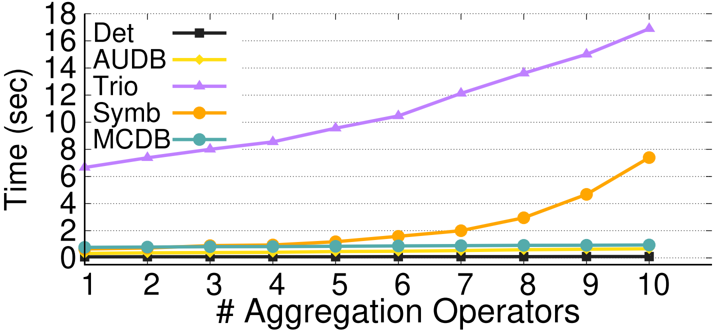


Figure: Simple aggregation queries over TPC-H (1GB) - Varying number of aggregation operations

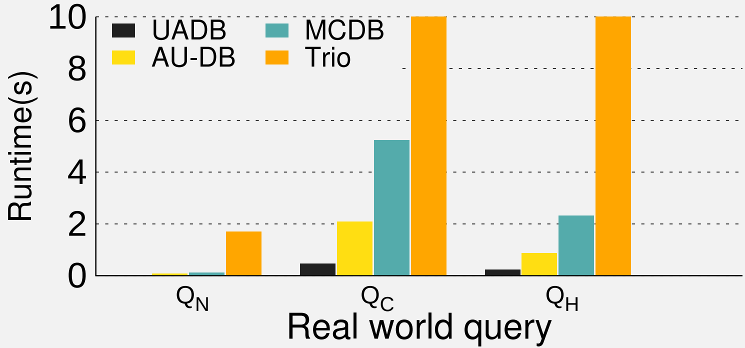


Figure: Experiments with Real Datasets (Netflix, Crimes, Healthcare)

Experimental Results - Real World Data

ILLINOIS INSTITUTE OF TECHNOLOGY

Datasets & Queries			Time (sec)	cert. tup.	attr. bounds min	attr. bounds max	pos.tup. by id	pos.tup. by val
Netflix (1.9%, 2.1)	$Q_{n,1}$ SPJ	AU-DB	**0.011**	100%	1	1	100%	100%
		Trio	0.900	100%	1	1	100%	100%
		MCDB	0.049	N/A	1	1	99.6%	98.5%
		UA-DB	0.006	100%	N/A	N/A	99.1%	97.3%
	$Q_{n,2}$ GB	AU-DB	**0.082**	100%	1	4	100%	100%
		Trio	1.700	100%	1	1	98.8%	98.0%
		MCDB	0.118	N/A	1	1	99.9%	97.9%
		UA-DB	0.009	0%	N/A	N/A	99.3%	95.7%
Crimes (0.1%, 3.2)	$Q_{c,1}$ SPJ	AU-DB	**1.58**	100%	1	1	100%	100%
		Trio	59.0	100%	1	1	100%	100%
		MCDB	6.91	N/A	0.6	1	99.9%	92.1%
		UA-DB	0.63	100%	N/A	N/A	99.9%	87.5%
	$Q_{c,2}$ GB	AU-DB	**2.09**	100%	1	1.01	100%	100%
		Trio	103.1	100%	1	1	100%	100%
		MCDB	5.24	N/A	0.99	0	100%	$\sim 0\%$
		UA-DB	0.47	0%	N/A	N/A	100%	$\sim 0\%$
Healthcare (1.0%, 2.7)	$Q_{h,1}$ SPJ	AU-DB	**0.179**	99.5%	1	1	100%	100%
		Trio	20.6	100%	1	1	100%	100%
		MCDB	0.501	N/A	0.4	1	99.9%	87.6%
		UA-DB	0.042	98.2%	N/A	N/A	99.3%	65.4%
	$Q_{h,2}$ GB	AU-DB	**0.859**	100%	1	45	100%	100%
		Trio	29.2	100%	1	1	100%	100%
		MCDB	2.31	N/A	0.78	1	100%	$\sim 0\%$
		UA-DB	0.235	0%	N/A	N/A	100%	$\sim 0\%$

Lenses

- Uncertainty-aware Data Cleaning Operations [YMF+15]

UA-DBs

- The basic UA-DB model [FHGK19]
- Attribute-level uncertainty, range values, and aggregation over UA-DBs (the AU-DB model) [FHGK21]
- Sorting, top-k, and windowed aggregation [FGK23]

Vizer

- `http://www.vizierdb.info`
- Publications: `http://www.cs.iit.edu/~dbgroup/publications.html` (click on project Vizier to see relevant publications)

Encode UA-DBs as relational databases

- attribute-level ranges
  - for each attribute `A` add `A_lb` and `A_ub`
- annotation ranges
  - add attributes `row_lb`, `row_sg`, `row_ub`

UA-DB

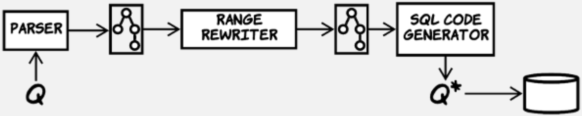
name	salary	$\mathbb{N}^3$
Boris	[120k,120k,120k]	[0,2,2]
Peter	[140k,400k,400k]	[2,3,3]

Relational encoding

name	$salary_{lb}$	salary	$salary_{ub}$	row_{lb}	row_{sg}	row_{ub}
Boris	120k	120k	120k	0	2	2
Peter	140k	400k	400k	2	3	3

ILLINOIS INSTITUTE OF TECHNOLOGY

Rewrite-based approach



Specialized Operator Implementations

- one pass algorithms for sorting and windowed aggregation

Existing implementations

- **GProM** (`https://github.com/IITDBGroup/gprom`)
- **Vizier** (`https://vizierdb.info/`)

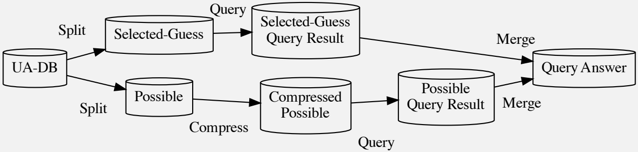
ILLINOIS INSTITUTE OF TECHNOLOGY

Performance of Joins and Aggregation

- Aggregation and joins require interval overlap joins
- $O(n^2)$ in many DBMS

Optimizations

- Split possible from selected-guess and process them separately
  - Selected-guess does not require interval overlap joins
  - Compress possible, interval overlap joins over smaller inputs
    - Trade performance for accuracy of the over-approximation
- This works, because approximate compression is bound preserving



Sorting and Windowed Aggregation

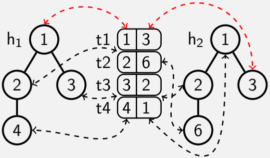
Rewrite-based approach

- can be expensive: range-joins and multiple sorting steps

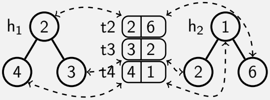
Native Algorithms

- one pass algorithms
- implemented inside Postgres
- requires reasoning about certain and possible windows and maintain rows in multiple sort orders

Connected Heaps



Result of `h1.pop()`



COVID infection rate data

- COVID infection rate per location from two sources

Query - maximum infection rate per location size

```
SELECT size, max(rate) as mrate
FROM R GROUP BY size
```

Source 1

city	size	rate
Los Angeles	Metro	2%
Washington	Metro	6%
Chicago	City	4%
Springfield	Town	2%

Source 2

city	size	rate
Los Angeles	Metro	7%
Washington	Village	8%
Chicago	City	3%

Conflicting Information

- Washington may belong to group *Metro* or *Village*
- Conflicting rates for Los Angeles
- Does Springfield exist?

Source 1

city	size	rate
Los Angeles	Metro	2%
Washington	Metro	6%
Chicago	City	4%
Springfield	Town	2%

Source 2

city	size	rate
Los Angeles	Metro	7%
Washington	Village	8%
Chicago	City	4%

Ignoring uncertainty in data leads to hard to trace errors with severe real world consequences.

Query - maximum infection rate per location size

```
SELECT size, max(rate) as mrate
FROM R GROUP BY size
```

"Cleaned" Sources

city	size	rate
Los Angeles	Metro	2%
Washington	Village	8%
Chicago	City	4%

Query Result

city	rate
Village	8%
Metro	2%
City	4%

Why Not Clean Your Data Upfront?

- **Heuristic algorithms**: pick one repair heuristically
- **Highest probability repair**: pick the most likely repair

Cleaned Data = Ground Truth?

- The ground truth is typically hard / impossible to determine

Take-away

Effectively, most data cleaning & integration techniques select one possible repair

- All information about uncertainty is lost!
- How can we trust any analysis result based on the cleaned data?

- **(R1) Efficiency**
  - Efficiently generate compact representations of uncertainty
  - Running queries over uncertain data should be efficient
  - Otherwise, users will just ignore uncertainty

Requirements for Managing Uncertainty

- **(R1) Efficiency**
  - Efficiently generate compact representations of uncertainty
  - Running queries over uncertain data should be efficient
  - Otherwise, users will just ignore uncertainty
- **(R2) Expressiveness**
  - Model complex types of uncertainty
  - Complex SQL queries / computations
  - Unlikely to be adopted otherwise

Requirements for Managing Uncertainty

- **(R1) Efficiency**
  - Efficiently generate compact representations of uncertainty
  - Running queries over uncertain data should be efficient
  - Otherwise, users will just ignore uncertainty
- **(R2) Expressiveness**
  - Model complex types of uncertainty
  - Complex SQL queries / computations
  - Unlikely to be adopted otherwise
- **(R3) Guarantees**
  - Provide as much information about uncertainty as possible
  - Queries should take uncertainty into account

Requirements for Managing Uncertainty

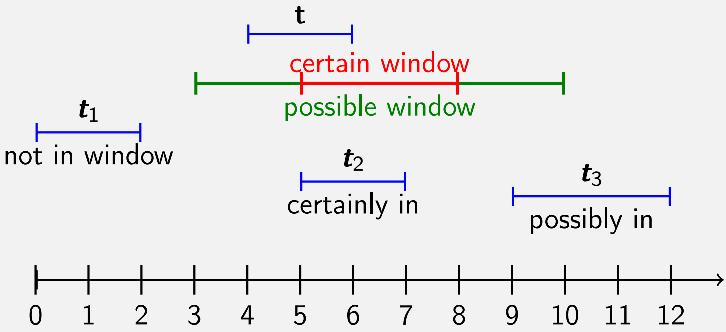
- **(R1) Efficiency**
  - Efficiently generate compact representations of uncertainty
  - Running queries over uncertain data should be efficient
  - Otherwise, users will just ignore uncertainty
- **(R2) Expressiveness**
  - Model complex types of uncertainty
  - Complex SQL queries / computations
  - Unlikely to be adopted otherwise
- **(R3) Guarantees**
  - Provide as much information about uncertainty as possible
  - Queries should take uncertainty into account
- **(R4) Usability**
  - Interpretable by mere mortals
  - Backwards-compatible with existing cleaning & integration solutions

Sort Order as Data

- there is only one physical sort order
- encode sort position as UA-DB range-values!

Windowed Aggregation

- exploit fixed window sizes



Experimental Results - TPC-H Queries

ILLINOIS INSTITUTE OF TECHNOLOGY

Queries		2%/SF0.1	2%/SF1	5%/SF1	10%/SF1	30%/SF1
Q1	AU-DB	1.607	15.636	15.746	15.811	16.021
	Det	0.560	1.833	1.884	1.882	1.883
	MCDB	5.152	19.107	18.938	19.063	19.279
Q3	AU-DB	0.713	7.830	8.170	8.530	7.972
	Det	0.394	1.017	1.058	1.092	1.175
	MCDB	4.112	11.138	11.222	10.936	11.454
Q5	AU-DB	0.846	8.877	8.803	8.839	8.925
	Det	0.247	0.999	1.012	1.123	1.117
	MCDB	2.599	10.152	10.981	11.527	11.909
Q7	AU-DB	0.791	7.484	7.537	7.303	7.259
	Det	0.145	0.977	0.985	0.989	1.044
	MCDB	1.472	10.123	10.277	10.749	10.900
Q10	AU-DB	0.745	7.377	7.283	7.715	8.012
	Det	0.263	1.024	0.993	1.004	1.015
	MCDB	2.691	10.743	10.937	11.826	11.697

Experimental Results - Ranking + Windows

- **MCDB** [JXW+08] - sample-based approach (10 or 20 samples)
- *PT-k* [HPZL08] - Full certain / possible answers for top-k queries
- *Symb* - Computing certain (possible) answers with constraint solver

Datasets & Queries		*Imp* (time)	*Det* (time)	*MCDB20* (time)	*Rewr* (time)	*Symb* (time)	*PT-k* (time)
Iceberg [ice07] (1.1%, 167K)	Rank	0.816msms	0.123ms	2.337ms	1.269ms	278ms	1s
	Window	2.964ms	0.363ms	7.582ms	1.046ms	589ms	N.A.
Crimes [crigo] (0.1%, 1.45M)	Rank	1043.505ms	94.306ms	2001.12ms	14787.723ms	>10min	>10min
	Window	3.050ms	0.416ms	8.337ms	2.226ms	>10min	N.A.
Healthcare [heare] (1.0%, 171K)	Rank	287.515ms	72.289ms	1451.232ms	4226.260ms	15s	8s
	Window	130.496ms	15.212ms	323.911ms	13713.218ms	>10min	N.A.

Datasets & Measures		*Imp*/*Rewr*	*MCDB20*	*PT-k*/*Symb*
Iceberg [ice07]	bound accuracy	0.891	1	1
	bound recall	1	0.765	1
Crimes [crigo]	bound accuracy	0.996	1	1
	bound recall	1	0.919	1
Healthcare [heare]	bound accuracy	0.990	1	1
	bound recall	1	0.767	1

- **Open-source**: `https://vizierdb.info/`
- **A Data-centric Workflow Engine Instead of a REPL**
  - no hidden state!
  - incremental workflow execution
- **A Hybrid Notebook + Spreadsheet UI**
  - build workflows incrementally through a notebook interface
  - edit datasets as spreadsheets
- **Caveats: Data Concerns that Propagate**
  - document data errors / concerns / uncertainty
  - caveats propagate alongside the data
  - supports quantification of data uncertainty using UA-DBs
  - summaries as overview of problems with a workflow
- **Workflow Provenance and Versioning**
  - Versioning (workflow evolution provenance)

Data-centric Workflows

Datasets

- Currently relational tables
- Spreadsheet interpretation (co-ordinate system)

Workflow Steps (cells)

- Cell executions are isolated from each other
- Dataflow through Vizier's dataset API
    - Cells consume and produce datasets

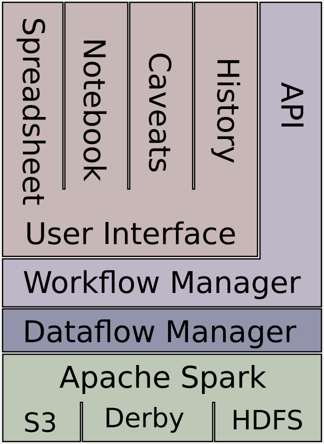


Figure: Vizier's architecture

The Notebook UI

Projects consists of a notebook & datasets

A notebook is a sequence of cells

- Cells are steps in a workflow
- User-facing interface like Jupyter or similar
- Cells run in isolation and communicate through datasets

Datasets

- Datasets are created / read / updated by cells
- Dataset creation
  - Loading (special cell type)
  - Result of a cell (e.g., SQL cell)
- Dataset update
  - Spreadsheet operations (e.g., update value)
  - Curation cells (e.g., impute missing values)

Spreadsheet UI

- Updates are translated into workflow steps

Spreadsheets as Datasets

- Relation + coordinate system (cell locations)

Vizual language

- Scripting language for common spreadsheet operations

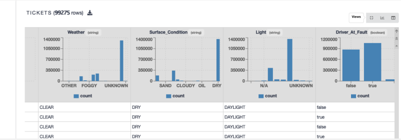


Figure: Column data distributions shown in the spreadsheet view

Workflow versioning

- Vizier captures provenance for the evolution of a user's workflow
- Like Git, but automates versioning
  - Every edit creates a new version
  - Branching is supported
- Versions of the workflow and of datasets have unique URLs

Incremental Execution model

Immutable objects

- Cells & their outputs are immutable objects (at least conceptually)

Dependency tracking

- Cell ordering is sequential (position in the notebook)
- Dependencies are tracked automatically
    - Monitoring Vizier dataset API calls

Automatic refresh

- On cell update, we determine automatically which cells are dependent and need to be re-executed

Caveats

- Caveats are annotations on data (attribute value)
- Record concerns / document assumptions about data
- May list possible alternative values or may mark value as unknown (uncertainty)
  - UA-DBs without boolean attribute markers instead of ranges

Caveat Generation

- Users can manually create caveats
- Data curation operations can annotated their outputs with caveats

Repairing a violated PK constraint

- **Primary key**: `Name`
- **Key repair**: `avg(Age)`
- Repaired value is caveated (highlighted in red)
  - Equivalent to annotating with a range that covers the whole domain

Name	Age
Peter	30
Peter	35
Bob	25

Name	Age
Peter	32.5
Bob	25

UI & API Support for Caveats

ILLINOIS INSTITUTE OF TECHNOLOGY

Dataset error summary

- Per dataset
- Shows counts of caveats grouped by type

Spreadsheet view

- Caveated values are shown in red
- Users can inspect caveat

Plots

- Bars / points based on caveated values are show with a red dot

Dataset API / SQL / Lenses

- Functions for accessing / creating / manipulating caveats
- SQL functions to create / check for caveats

Cell types - curation & cleaning

Data curation (lenses)

- Missing value imputation
- Pivot
- Geocoding
- Type detection
- Schema Matching
- ...

Uncertainty-aware through UA-DBs

- Results stored as UA-DBs

ILLINOIS INSTITUTE OF TECHNOLOGY

[ABC+03] Marcelo Arenas, Leopoldo Bertossi, Jan Chomicki, Xin He, Vijay Raghavan, and Jeremy Spinrad.
Scalar aggregation in inconsistent databases.
Theoretical Computer Science, 296(3):405–434, 2003.

[ABS15] Antoine Amarilli, Pierre Bourhis, and Pierre Senellart.
Probabilities and provenance via tree decompositions.
Preprint: http://a3nm. net/publications/amarilli2015probabilities. pdf. Submitted to PODS, 2015.

[ACK+10] Serge Abiteboul, T.-H. Hubert Chan, Evgeny Kharlamov, Werner Nutt, and Pierre Senellart.
Aggregate queries for discrete and continuous probabilistic XML.
In Luc Segoufin, editor, *Database Theory - ICDT 2010, 13th International Conference, Lausanne, Switzerland, March 23-25, 2010, Proceedings*, ACM International Conference Proceeding Series, pages 50–61. ACM, 2010.

[ADT11] Yael Amsterdamer, Daniel Deutch, and Val Tannen.
Provenance for aggregate queries.
In *Proceedings of the thirtieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 153–164. ACM, 2011.

[AJKO08] Lyublena Antova, Thomas Jansen, Christoph Koch, and Dan Olteanu.
Fast and simple relational processing of uncertain data.
In *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*, pages 983–992. IEEE, 2008.

[AK08] Foto N. Afrati and Phokion G. Kolaitis.
Answering aggregate queries in data exchange.
In *Proceedings of the Twenty-Seventh ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2008, June 9-11, 2008, Vancouver, BC, Canada*, pages 129–138, 2008.

ILLINOIS INSTITUTE OF TECHNOLOGY

[AKO07] L. Antova, C. Koch, and D. Olteanu.
MayBMS: Managing incomplete information with probabilistic world-set decompositions.
In *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*, pages 1479–1480. IEEE, 2007.

[BSHW06] Omar Benjelloun, Anish Das Sarma, Alon Y. Halevy, and Jennifer Widom.
ULDBs: Databases with Uncertainty and Lineage.
In *Proceedings of the 32th International Conference on Very Large Data Bases (VLDB)*, pages 953–964, 2006.

[CC77] Patrick Cousot and Radhia Cousot.
Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints.
In *Proceedings of the 4th ACM SIGACT-SIGPLAN symposium on Principles of programming languages*, pages 238–252, 1977.

[CC04] P. COUSOT and R. COUSOT.
Abstract Interpretation Frameworks.
Journal of Logic and Computation, 2(4):511–547, 2004.

[crigo] Chicago crimes dataset.
`https://www.kaggle.com/currie32/crimes-in-chicago`.

[DS07] Nilesh Dalvi and Dan Suciu.
The dichotomy of conjunctive queries on probabilistic structures.
In *Proceedings of the twenty-sixth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 293–302. ACM, 2007.

[DS12] Nilesh Dalvi and Dan Suciu.
The dichotomy of probabilistic inference for unions of conjunctive queries.
Journal of the ACM (JACM), 59(6):30, 2012.

[FGK23] Su Feng, Boris Glavic, and Oliver Kennedy.
Efficient approximation of certain and possible answers for ranking and window queries over uncertain data.
Proceedings of the VLDB Endowment, 16(6):1346 – 1358, 2023.

[FHGK19] Su Feng, Aaron Huber, Boris Glavic, and Oliver Kennedy.
Uncertainty annotated databases - a lightweight approach for approximating certain answers.
In *Proceedings of the 44th International Conference on Management of Data*, pages 1313–1330, 2019.

[FHGK21] Su Feng, Aaron Huber, Boris Glavic, and Oliver Kennedy.
Efficient uncertainty tracking for complex queries with attribute-level bounds.
In *Proceedings of the 46th International Conference on Management of Data*, page 528 – 540, 2021.

[FHO12] Robert Fink, Larisa Han, and Dan Olteanu.
Aggregation in probabilistic databases via knowledge compilation.
Proceedings of the VLDB Endowment, 5(5):490–501, 2012.

[FO16] Robert Fink and Dan Olteanu.
Dichotomies for queries with negation in probabilistic databases.
ACM Trans. Database Syst., 41(1):4:1–4:47, 2016.

[Fux07] A.D. Fuxman.
Efficient query processing over inconsistent databases.
PhD thesis, University of Toronto, 2007.

[GKT07] Todd J. Green, Gregory Karvounarakis, and Val Tannen.
Provenance Semirings.
In *PODS '07: Proceedings of the 26th Symposium on Principles of Database Systems*, pages 31–40, 2007.

[GP10] F. Geerts and A. Poggi.
On database query languages for K-relations.
Journal of Applied Logic, 8(2):173–185, 2010.

[GT17] Todd J Green and Val Tannen.
The semiring framework for database provenance.
In *Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, pages 93–99. ACM, 2017.

[heare] Medicare hospital dataset.
`https://data.medicare.gov/data/hospital-compare`.

[HPZL08] Ming Hua, Jian Pei, Wenjie Zhang, and Xuemin Lin.
Ranking queries on uncertain data: a probabilistic threshold approach.
In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008*, pages 673–686, 2008.

[ice07] Iceberg dataset.
`https://nsidc.org/data/g00807`.

[ILJ84] Tomasz Imieliński and Witold Lipski Jr.
Incomplete Information in Relational Databases.
Journal of the ACM (JACM), 31(4):761–791, 1984.

[IVV95] Tomasz Imielinski, Ron Vandermeyden, and Kumar V Vadaparty.
Complexity tailored design: A new design methodology for databases with incomplete information.
Journal of Computer and System Sciences, 51(3):405–432, 1995.

[JXW+08] R. Jampani, F. Xu, M. Wu, L.L. Perez, C. Jermaine, and P.J. Haas.
MCDB: a monte carlo approach to managing uncertain data.
In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 687–700. ACM, 2008.

[KB12] Egor V Kostylev and Peter Buneman.
Combining dependent annotations for relational algebra.
In *Proceedings of the 15th International Conference on Database Theory*, pages 196–207. ACM, 2012.

[KNP+22] Mahmoud Abo Khamis, Hung Q. Ngo, Reinhard Pichler, Dan Suciu, and Yisu Remy Wang.
Convergence of datalog over (pre-) semirings.
In Leonid Libkin and Pablo Barceló, editors, *PODS '22: International Conference on Management of Data, Philadelphia, PA, USA, June 12 - 17, 2022*, pages 105–117. ACM, 2022.

[Lib16] Leonid Libkin.
Sql's three-valued logic and certain answers.
ACM Transactions on Database Systems (TODS), 41(1):1, 2016.

[LJ84] Witold Lipski Jr.
On relational algebra with marked nulls.
In *Proceedings of the 3rd ACM SIGACT-SIGMOD symposium on Principles of database systems*, pages 201–203. ACM, 1984.

[LSV02] Jens Lechtenbörger, Hua Shu, and Gottfried Vossen.
Aggregate Queries over Conditional Tables.
Journal of Intelligent Information Systems, 19(3):343–362, 2002.

[OHK09] D. Olteanu, J. Huang, and C. Koch.
Sprout: Lazy vs. eager query plans for tuple-independent probabilistic databases.
In *Data Engineering, 2009. ICDE'09. IEEE 25th International Conference on*, pages 640–651. IEEE, 2009.

[RS09] Christopher Ré and Dan Suciu.
The trichotomy of HAVING queries on a probabilistic database.
VLDB J., 18(5):1091–1116, 2009.

[SKL+17] Bruhathi Sundarmurthy, Paraschos Koutris, Willis Lang, Jeffrey Naughton, and Val Tannen.
m-tables: Representing missing data.
In *LIPIcs-Leibniz International Proceedings in Informatics*, volume 68. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

[VdBS17] Guy Van den Broeck and Dan Suciu.
Query processing on probabilistic data: A survey.
2017.

[YMF+15] Ying Yang, Niccolo Meneghetti, Ronny Fehling, Zhen Hua Liu, and Oliver Kennedy.
Lenses: an on-demand approach to etl.
Proceedings of the VLDB Endowment, 8(12):1578–1589, 2015.