Provenance, Relevance-based Data Management, and the Value of Data

ProvenanceWeek '23

Boris Glavic[1]

DBGroup

Illinois Institute of Technology

2023-04-30

Have to follow the laws of keynotes

Keynote Checklist

1. Overly grandiose vision
2. Repackage existing work under fancy new name
3. Shamelessly plug your own work
4. Rant about the state of the field

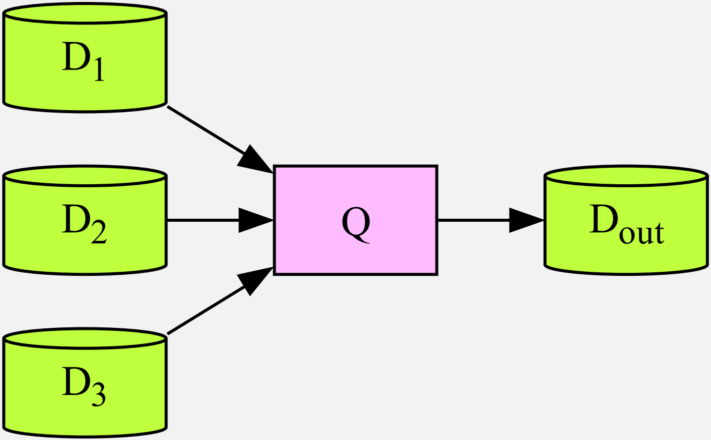
This Keynote's Provenance

Have to follow the laws of keynotes

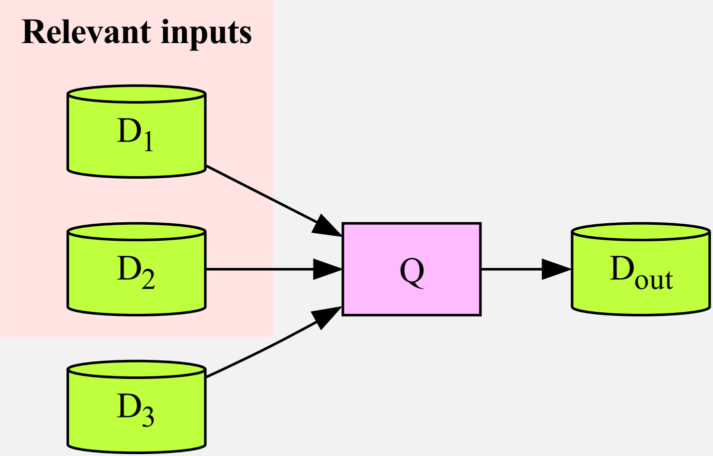
Keynote Checklist

1. **Overly grandiose vision**
2. Repackage existing work under fancy new name
3. Shamelessly plug your own work
4. Rant about the state of the field

What is relevance?



What is relevance?



Revenue of states with > 10000 sales from category toys or food

state	revenue
CA	11,100

```
SELECT state, sum(sales) AS revenue
FROM prod-sales WHERE prod-category IN (‘toys’,’food’)
GROUP BY state HAVING sum(sales) > 10000
```

state	city	sales	prod-category
IL	Chicago	4000	toys
IL	Springfield	5000	toys
CA	San Fransico	6000	toys
CA	Santa Cruz	2500	food
CA	Sacramento	2600	food
CA	San Fransico	1400	alcohol

Running Example Relevance

What data is relevant for this query?

- only rows where `prod_category IN (toys, food)`
- only states where `sum(sales) > 10000` for these categories

```
SELECT state, sum(sales) AS revenue
FROM prod-sales WHERE prod-category IN ('toys','food')
GROUP BY state HAVING sum(sales) > 10000
```

state	city	sales	prod-category
IL	Chicago	4000	toys
IL	Springfield	5000	toys
CA	San Fransico	6000	toys
CA	Santa Cruz	2500	food
CA	Sacramento	2600	food
CA	San Fransico	1400	alcohol

Speed-up computations

- Restrict computations to relevant data
- How to reuse relevance information across similar computations?

Allocating wrangling resources

- Data collection and preparation is expensive
- Why spend effort on data that is not relevant?

Data life-cycle management

- Relevant data should be kept in fast storage (cache)
- Irrelevant data can be compressed / deleted / moved to slow storage

Objective metric of data value

- Relevance is an **objective metric of data value**
- **Dark data**: data that is not useful or has not yet been discovered to be useful

Defining Data Relevance

Definition (Relevance)

- **Input**: dataset D and computation Q
- **Output**: $\mathcal{R}(Q, D) = D' \subseteq D$ fulfilling
  - **sufficiency**: $Q(D') = Q(D)$
  - **minimality**: $\forall D'' \subset D' : Q(D'') \neq Q(D)$

Relevance vs. Data Access

- D was accessed $\not\Rightarrow$ D is relevant

Aggregated Relevance

- Workload $\mathcal{W} = Q_1, \ldots, Q_n$
- Relevance of $d_i \in D$ for $\mathcal{W}$ is:

$$\frac{\sum_i \mathbb{1}[d_i \in \mathcal{R}(Q_i, D)]}{|\mathcal{W}|}$$

Remarks

Problems with minimality

- **not unique** (e.g., disjunctive operations)!
  - for some applications we don't care
  - otherwise,

$$\mathcal{R}(Q, D) = \{d \mid \exists D' \subset D : d \in D' \wedge D' \textbf{ is sufficient and minimal}\}$$

- **computational complexity**
  - **NP-hard** for `sum` $\Rightarrow$ subset-sum problem
  - drop minimality requirement (and/or sufficiency)

Remarks

Degrees of relevance

- relevance is all-or-nothing
- some applications (e.g., approximation) need degree of relevance

Data- + computation-centric

- relevance is data-centric
  - specific to D
- relevance is computation-centric
  - specific to Q or $\mathcal{W}$

This Keynote's Provenance

ILLINOIS INSTITUTE OF TECHNOLOGY

Keynote Checklist

1. Overly grandiose vision
2. **Repackage existing work under fancy new name**
3. Shamelessly plug your own work
4. Rant about the state of the field

Relevance vs. Provenance

Provenance = Relevance?

- Consider a tuple level provenance model $P(Q, D, t)$
  - most provenance models are sufficient (e.g., [GKT07, Gla21])
  - some fulfill our fixed minimality notion
- For such models:

$$\mathcal{R}(Q, D) = \bigcup_{t \in Q(D)} P(Q, D, t)$$

Degrees of relevance = attribution / responsibility?

- If approximate answers are acceptable, then we want more information
  - how much does a data item contribute to a result
  - e.g., bias sampling in APQ towards more relevant data
- closely related to:
  - causal responsibility [MGMS10]
  - attribution techniques (e.g., Shapley [LBKS20, GZ19, dBLSS21])

Isn't that just . . . ?

ILLINOIS INSTITUTE OF TECHNOLOGY

Provenance

- **yes, but** . . .
  - only care about sufficiency not completeness for many relevance use cases
  - novel challenges
    - reusing provenance across computations
    - maintaining provenance under updates
    - (over-)approximations

Heatmaps

- **yes, but** . . .
  - heatmaps track data access, not relevance

Materialized views

- **no, but** . . .
  - store relevant inputs instead of outputs

This Keynote's Provenance

ILLINOIS INSTITUTE OF TECHNOLOGY

Keynote Checklist

1. Overly grandiose vision
2. Repackage existing work under fancy new name
3. Shamelessly plug your own work
4. **Rant about the state of the field**

The obligatory keynote rant

- **Most provenance work has focused on end user consumption**
  - debugging,
  - auditing
  - understanding computations
  - ...

ILLINOIS INSTITUTE OF TECHNOLOGY

Provenance as a supporting technology

- partial result refresh [ISW11]
- reuse [PW18]
- provisioning [AKLT15]
- probabilistic databases [VdBS17]
- reproducibility [MSM+22, KGFB22]
- ...

Why is System-centric Provenance Different?

We can now ignore

- visualization of provenance
- can a human comprehend the provenance we create?

Facing objective truths

- optimizing against objective metrics
  - speed
  - storage consumptions
  - accuracy
  - ...

This Keynote's Provenance

Keynote Checklist

1. Overly grandiose vision
2. Repackage existing work under fancy new name
3. **Shamelessly plug your own work**
4. Rant about the state of the field

Outline

1. Relevance-based Data Management
2. Provenance-based Data Skipping
3. Provenance Sketches
4. Sketch Capture and Use
5. Reusing Sketches and Sketch Safety
6. Experiments
7. Relevance and The Value of Data
8. Conclusions

Fundamental principle in query processing

- determine upfront (**statically**) what data is needed (**relevant**) to answer a query
- **exclude** irrelevant data as early and as efficiently as possible

- Decades of research & developement - efficient storage organization for databases
  - **Index structures** [LLS13, AS13, Gra06, Com79, BS77, Moe98]
    - B-trees, Hash-index, Bitmap-index, zone maps, . . .
    - select rows based on attribute values
  - **Partitioned tables** [DGTM17, RJ17, SFWW16, AEHS+14, TMS+14, PCZ12, ANY04, CY90]
    - split tables into horizontal/vertial fragments
  - . . .
- Decades of research & development - query optimization and execution
  - **Selection move-around** [LM97]
    - filter data based on selections implied by the query
  - **Semi-join reducers** [Mul90, BC81, BG81]
    - filter input tables before joining
  - . . .

Revenue of states with > 10000 sales from category toys or food

state	revenue
CA	11,100

```
SELECT state, sum(sales) AS revenue
FROM prod-sales WHERE prod-category IN ('toys','food')
GROUP BY state HAVING sum(sales) > 10000
```

state	city	sales	prod-category
IL	Chicago	4000	toys
IL	Springfield	5000	toys
CA	San Fransico	6000	toys
CA	Santa Cruz	2500	food
CA	Sacramento	2600	food
CA	San Fransico	1400	alcohol

What data is relevant for this query?

- only rows where `prod_category IN (toys, food)`
- only states where `sum(sales) > 10000` for these categories

What data can be skipped by the DBMS?

- Utilize zone map (index, partitioning, ...) on `prod_category`
- however, we cannot filter states because we will only know at runtime which states qualify!

```
SELECT state, sum(sales) AS revenue
FROM prod-sales WHERE prod-category IN ('toys','food')
GROUP BY state HAVING sum(sales) > 10000
```

Runtime Relevance Analysis to the Rescue

ILLINOIS INSTITUTE OF TECHNOLOGY

Runtime Relevance Analysis

- Determine **dynamically** at runtime what data is **relevant** for a query
  - pay overhead once to determine what data is relevant
- Use this information to **benefit future queries**
  - **amortize** cost over time

Synergy with existing technologies

- To use relevance information to efficiently filter data we need to exploit ...
  - zone maps
  - indexes
  - partitioning
  - in-memory caches
  - ...

ILLINOIS INSTITUTE OF TECHNOLOGY

Filtering relevant data

- only CA qualifies, add condition `state IN ('CA')`

```
SELECT state, sum(sales) AS revenue
FROM prod-sales
WHERE prod-category IN ('toys','food')
      AND state IN ('CA')
GROUP BY state HAVING sum(sales) > 10000
```

state	city	sales	prod-category
IL	Chicago	4000	toys
IL	Springfield	5000	toys
CA	San Fransico	6000	toys
CA	Santa Cruz	2500	food
CA	Sacramento	2600	food
CA	San Fransico	1400	alcohol

Challenges

- (1) Capture
  - **How to compactly over-approximate what data is relevant?**
    - just listing relevant rows is inefficient
  - **How to determine efficiently at runtime what data is relevant?**
    - needs to be applicable to relatively complex queries
- (2) Use
  - **How to filter irrelevant data effectively?**
    - need to encode relevant data for efficient filtering
    - want **synergy** with existing physical design and self-tuning technologies
- (3) Safety
  - **How to determine whether an over-approximation of relevant data yields the same result?**

Challenges (cont.)

- (4) Reuse
  - **How to utilize relevance information for multiple queries?**
    - need to determine statically whether the relevant data for query Q_1 subsumes relevant data of Q_2
- (5) Maintenance
  - **How to maintain relevance information under updates?**
    - when data is updated, relevance information becomes stale

When is this beneficial?

Fundamental requirement

- We have `|relevant data|` ≪ `|data that DBMS can filter|`

Counterexample

- If in our running example `CA` is the only state that sells *food* and *toys*, then the data returned by the DBMS based on the query's `WHERE` clause is exactly what is relevant for the query.
- ⇒ there is no way for us to benefit

Common query types with these characteristics

- `HAVING` queries
- Top-k queries

Provenance-based Data Skipping (PBDS)

ILLINOIS INSTITUTE OF TECHNOLOGY

1. **Compact over-approximation of relevant data**
   - Use (*virtual*) range-partitioning to represent subsets of the data
   - **Provenance sketches**: stores which fragments of a partition contain relevant data
2. **Determine at runtime what data is relevant for a query**
   - Our method utilizes **data provenance** techniques
   - Instrument queries to **capture** provenance sketches (**query rewrites**)
3. **Instrument queries to filter out irrelevant data early-on**
   - Use provenance sketches to speed-up execution of a query
   - **Filter data** based on sketch by adding `WHERE` conditions
4. **Re-use of provenance sketches**
   - Use sketch captured for query Q_1 to answer Q_2
   - Check statically if Q_1's sketch subsumes relevant data for Q_2

Revisiting the Running Example

Range partitioning

- We partition on `sales`

$$\{[0,600],[601,1200],[1201,2700],[2701,3500],[3501,5500],[5501,10000]\}$$

Provenance Sketch

- Fragments in the sketch (that belong to provenance)

$$\{[1201,2700],[5501,10000]\}$$

Captured data

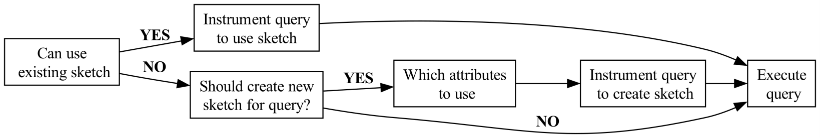
state	city	sales	prod-category
CA	San Francisco	6000	toys
CA	Santa Cruz	2500	food
CA	Sacramento	2600	food
CA	San Francisco	1400	alcohol

PDBS Life-cycle

Data structures

- Maintain index of sketches that maps parameterized queries + bind parameters to sketches
- Bookkeep costs and benefits

Workflow for a query



1. Relevance-based Data Management
2. Provenance-based Data Skipping
3. Provenance Sketches
4. Sketch Capture and Use
5. Reusing Sketches and Sketch Safety
6. Experiments
7. Relevance and The Value of Data
8. Conclusions

What do we want?

1. compact over-approximation of provenance
2. fast to compute
3. can be exploited by the DBMS to skip data

Horizontal Partitioning

Horizontal partitioning of a table R = set of fragments $f_1, \ldots, f_n$ such that:

1. $\forall i : f_i \subseteq R$
2. $\bigcup_{i=1}^{n} f_i = R$
3. $\forall i, j : f_i \cap f_j = \emptyset$

ILLINOIS INSTITUTE OF TECHNOLOGY

Definition (Provenance Sketch)

A **provenance sketch** $\mathcal{P}$ for a query Q and database D contains for each table R in D accessed by Q a pair:

- a *horizontal partitioning* $\mathcal{F}_R$ for R
- a set of fragments: $\mathcal{P}(R) \subseteq \mathcal{F}_R$

Data contained in a sketch:

$$D_{\mathcal{P}} = \bigcup_{f \in \mathcal{P}} f$$

- **over-approximates provenance:**

$$Prov(Q, D) \subseteq D_{\mathcal{P}}$$

Accurate Sketches

Definition (Accurate Sketches)

A provenance $\mathcal{P}$ for a query Q and database D is **accurate** iff:

$$\forall f \in \mathcal{P} : Prov(Q, D) \cap f \neq \emptyset$$

state	rev
CA	10000

```
SELECT state, sum(sales) AS rev
FROM sal
GROUP BY state
HAVING sum(sales) > 5000
```

state	city	sales	page#
IL	Chicago	3000	P_1
IL	Schaumburg	500	
CA	**Sacramento**	**2000**	P_2
IL	Springfield	10	
CA	**San Francisco**	**8000**	P_3
CA	**Santa Cruz**	**1000**	

Provenance Sketches

- **Page-based**

$\{P_2, P_3\}$

- **Hash-based** (on sales)

$\{1, 2\}$

Value	Hash
10	0
20	1
500	2
1000	0
3000	1
8000	0

- **Range-based** (on sales)

$\{[801, 5000], [5001, 100000]\}$

for ranges:

$\{[0, 500], [501, 800], [801, 5000], [5001, 20000]\}$

Provenance Sketches (Observations)

1. **Any partitioning can be utilized to create a provenance sketch**
   - Sketch contains all fragments that contains provenance
   - e.g., **range** - divide the input table into fragments based on a partitioning of an attribute domain (can exploit histograms [CGHJ12, Ioa03])
2. **Physical vs. Virtual**
   - **Physical** = partitioning aligns with physical design
   - **Virtual** = partitioning does not align with physical design

Outline

Capturing Provenance Sketches

Query Instrumentation

- Given query Q
- Construct query Q_{sketch} that generates a provenance sketch

Relational Algebra Rewrite Rules

- One rewrite rule per algebra operator
  - $\Rightarrow$ composable
- Queries are rewritten by recursive application of these rules

Extension of our previous work on query instrumentation [AFG^{+}18]

- Represent sets of fragments as bitvectors
- Combine used user-defined aggregation functions

Query Instrumentation

ILLINOIS INSTITUTE OF TECHNOLOGY

Input Query

```sql
SELECT dept, avg(salary) AS avgsal
FROM emp
GROUP BY dept
```

Instrumented Query

```sql
SELECT dept, avgsal, provName, provSalary, provDept
FROM (SELECT dept, avg(salary) AS avgsal
      FROM emp
      GROUP BY dept) o,
      (SELECT name as provName,
              salary AS provSalary,
              dept AS provDept
      FROM emp) p
WHERE o.dept = p.provDept
```

Relational Encoding of Provenance

Result with Provenance

- Using the instrumented query from the previous slide

dept	avgsal	provName	provSalary	provDept
HR	15	Peter	10	HR
HR	15	Bob	20	HR
IT	5	Alice	5	IT

name	salary	dept
Peter	10	HR
Bob	20	HR
Alice	5	IT

Input Query

```
SELECT dept, avg(salary) AS avgsal
FROM emp
GROUP BY dept
```

Instrumented Query

- for ranges: $\{[0, 500], [501, 800], [801, 5000], [5001, 20000]\}$

```
SELECT bit_or_agg(provsketch)
FROM (SELECT salary, dept,
  CASE WHEN salary BETWEEN 0 AND 500 THEN 1 << 0
       WHEN salary BETWEEN 0 AND 500 THEN 1 << 1
            ...
  END AS provsketch
    FROM empl)
GROUP BY dept
```

Sketch Use

- To benefit from using a provenance sketch we have to instruct the database what data to access
- Most sketch types: compile sketch into selection condition
- Database systems are good at filtering data based on conditions!

We compile the compact representation of what is relevant provided by the provenance sketch into a format understood by the DBMS!

Provenance Sketch Use (Example)


```sql
SELECT state, sum(sales) AS rev
FROM sal
WHERE salary BETWEEN 801 AND 5000
   OR salary BETWEEN 5001 AND 100000
GROUP BY state
HAVING sum(sales) > 5000
```

state	city	sales	page#
IL	**Chicago**	**3000**	P_1
IL	Schaumburg	500	
CA	**Sacramento**	**2000**	P_2
IL	Springfield	10	
CA	**San Francisco**	**8000**	P_3
CA	**Santa Cruz**	**1000**	

Provenance Sketches

- **Range-based** (on sales)

$\{[801, 5000], [5001, 100000]\}$

for ranges:

$\{[0, 500], [501, 800], [801, 5000], [5001, 20000]\}$

Running Example (using a sketch)

ILLINOIS INSTITUTE OF TECHNOLOGY

Instrumented query

```
SELECT state, sum(sales) AS revenue
FROM prod-sales
WHERE prod-category IN ('toys','food')
   AND (salary BETWEEN 1201 AND 2700
      OR salary BETWEEN 5500 AND 10000)
GROUP BY state HAVING sum(sales) > 10000
```

Sketch

- Partition sales

  $\{[0, 600], [601, 1200], [1201, 2700], [2701, 3500], [3501, 5500], [5501, 10000]\}$

- Sketch

  $\{[1201, 2700], [5501, 10000]\}$

state	city	sales	prod-category	fragment
IL	Chicago	4000	toys	[3501,5500]
IL	Springfield	5000	toys	[3501,5500]
CA	San Fransico	6000	toys	**[5501,10000]**
CA	Santa Cruz	2500	food	**[1201,2700]**
CA	Sacramento	2600	food	**[1201,2700]**
CA	San Fransico	1400	alcohol	**[1201,2700]**

Integrating everything

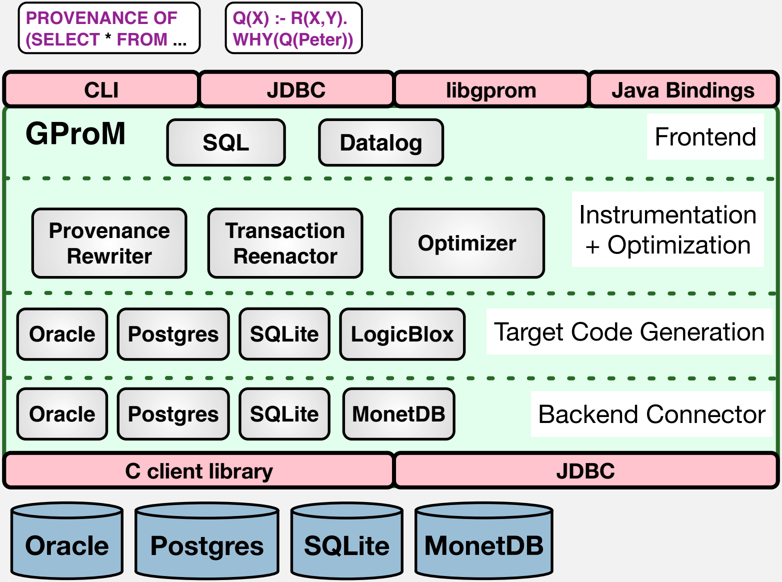
Problem Definition

- A database D
- A workload of queries $Q_1, \ldots, Q_n$ which is unveiled one query at a time
- Decide for each query ...
  - what (if any) relevance information to capture
  - whether to reuse previously captured relevance information to speed-up the query

Bookkeeping

- track what queries we have seen so far
- store provenance sketch(es) for queries
- fetch sketch if needed

- **Project page**
  - `http://www.cs.iit.edu/~dbgroup/projects/gprom.html`
- Available on **github**
  - `https://github.com/IITDBGroup/gprom`
- **GProM** [?] one sentence:
  - *A SQL+X to SQL optimizing compiler*
- Implemented in C
  - **Clients**: CLI, shared library, Java bindings
  - **Frontend languages**: SQL, Datalog
  - **Backends** (use native C libraries or ODBC): Oracle, Postgres, SQLite, MonetDB, MSSQL



PROVENANCE OF (SELECT * FROM ...
Q(X) :- R(X,Y). WHY(Q(Peter))
CLI
JDBC
libgprom
Java Bindings
GProM
SQL
Datalog
Frontend
Provenance Rewriter
Transaction Reenactor
Optimizer
Instrumentation + Optimization
Oracle
Postgres
SQLite
LogicBlox
Target Code Generation
Oracle
Postgres
SQLite
MonetDB
Backend Connector
C client library
JDBC
Oracle
Postgres
SQLite
MonetDB

Outline

Reusing Sketches

Sketch Reuse

- Given query Q_1 and its sketch $\mathcal{P}_1$
  - can it be reused to answer Q_2
- Prove that sketch for Q_1 contains all relevant data for Q_2
  - *Provenance containment* [Gre11]

Results

- Limited to parameterized queries
- Sound static method utilizing database statistics
  - inspired by work for proving query equivalence ([ZAN+19])

Parameterized Queries

- Query whose selection conditions may contain placeholders
- **Instance** of a parameterized query:
  - Query we get for a given parameter setting

Example

```
SELECT sum(popden) AS totalPop, country
FROM city WHERE  popden < $1
GROUP BY country
```

- $1 = 100000

```
SELECT sum(popden) AS totalPop, country
FROM city WHERE  popden < 100000
GROUP BY country
```

Theorem (Provenance Containment)

Given a sketch $\mathcal{P}$ for an instance Q_1 a parameterized query $\mathcal{T}$ and a database D, $\mathcal{P}$ is sufficient for any instance Q_2 of $\mathcal{T}$ if[a]*:*

$$Prov(Q_1, D) \supseteq Prov(Q_2, D)$$

[a] Under the assumption that (i) provenance is sufficient and (ii) sufficiency is preserved under $\subseteq$

Sound Containment Condition

Our sound condition

- Universally quantified formula based on per operator rules
- Encodes constraints that have to hold for (intermediate) result query results
  - for each t of Q_2 we exists corresponding t' in Q_1:
    - $Prov(Q_2, t) \subseteq Prov(Q_1, t')$
- Formula is valid $\Rightarrow$ provenance containment for every database D [a]

[a] Can add statistics of D as additional constraints.

Testing the condition

- We use an SMT solver (Z3 [dMB08])

Provenance Containment Example

totalPop	state
3500	TX
6000	NY
4200	AK

$\alpha_{sum(popden);state}$

den	city	state
1500	Austin	TX
2000	Houston	TX
2200	Buffalo	NY
3800	New York	NY
4200	Anchorage	AK

$\sigma_{popden<5000}$

den	city	state
1500	Austin	TX
2000	Houston	TX
2200	Buffalo	NY
3800	New York	NY
4200	Anchorage	AK

cities

totalPop	state
3500	TX
6000	NY
4200	AK

$\alpha_{sum(popden');state}$

den	city	state
1500	Austin	TX
2000	Houston	TX
2200	Buffalo	NY

$\sigma_{popden'<3000}$

den	city	state
1500	Austin	TX
2000	Houston	TX
2200	Buffalo	NY
3800	New York	NY
4200	Anchorage	AK

cities

$$popden = popden' \wedge popden' < 3000 \rightarrow popden < 5000$$

Sketch Safety

Problem

- Provenance sketches encode super sets of a query's provenance
- $\Rightarrow$ this may lead to incorrect results some queries
- How to determine which sketches are **safe** to use?

Definition (Sketch Safety)

We call a sketch $\mathcal{P}$ safe for a query Q and database D if

$$Q(D) = Q(D_{\mathcal{P}})$$

Example (Unsafe Sketch)

state	rev
IL	3510

```sql
SELECT state, sum(sales) AS rev
FROM sal
GROUP BY state
HAVING sum(sales) < 5000
```

state	city	sales
IL	**Chicago**	**3000**
IL	**Schaumburg**	**500**
CA	**Sacramento**	**2000**
IL	**Springfield**	**10**
CA	San Francisco	8000
CA	**Santa Cruz**	**1000**

Provenance Sketches

- **Range-based** (on sales)

$\{[0, 500], [501, 4000]\}$

for ranges:

$\{[0, 500], [501, 4000], [4001, 20000]\}$

Example (Unsafe Sketch)

ILLINOIS INSTITUTE OF TECHNOLOGY

state	rev
IL	3510
CA	3000

```sql
SELECT state, sum(sales) AS rev
FROM sal
GROUP BY state
HAVING sum(sales) < 5000
```

state	city	sales
IL	Chicago	3000
IL	Schaumburg	500
CA	Sacramento	2000
IL	Springfield	10
CA	Santa Cruz	1000

Provenance Sketches

- **Range-based** (on sales)

$\{[0, 500], [501, 4000]\}$

for ranges:

$\{[0, 500], [501, 4000], [4001, 20000]\}$

Safety is Data Dependent!

state	rev
IL	3510

```
SELECT state, sum(sales) AS rev
FROM sal
GROUP BY state
HAVING sum(sales) < 5000
```

state	city	sales
IL	Chicago	3000
IL	Schaumburg	500
CA	Sacramento	2000
IL	Springfield	10
CA	San Fansico	4000
CA	Santa Cruz	1000

Provenance Sketches

- **Range-based** (on sales)

$\{[0, 500], [501, 4000]\}$

for ranges:

$\{[0, 500], [501, 4000], [4001, 20000]\}$

Requirements for Safety Checks

- **Avoid generating incorrect sketches**
  - $\Rightarrow$ determine sketch safety at query compile time
  - $\Rightarrow$ don't have full access to the data or intermediate results
- **Gather information about the data as long as this is cheap**
  - access stats the DBMS keeps anyways
- **Ensure Soundness**
  - have to give up **completeness**
  - $\Rightarrow$ we may fail to find safe sketches but never erroneously declare a sketch as safe

Safety of Monotone Queries

ILLINOIS INSTITUTE OF TECHNOLOGY

Definition (Monotone Queries)

A query Q is **monotone** if for any pair of databases $D \subseteq D'$:

$$Q(D) \subseteq Q(D')$$

Theorem (Any Sketch is Safe for Monotone Queries)

- *Let Q be a monotone query and D a database*
- *$\Rightarrow$ any provenance sketch for Q and D is safe*

Proof.

- $Prov(Q, D) \subseteq D_{\mathcal{P}} \subseteq D$ (**provenance sketch definition**)
- $Q(Prov(Q, D)) = Q(D)$ (**sufficiency of provenance**)
- $\Rightarrow Q(Prov(Q, D)) \subseteq Q(D_{\mathcal{P}}) \subseteq Q(D)$ (**monotonicity**)
- $\Rightarrow Q(D_{\mathcal{P}}) = Q(D)$

$\square$

Sound Safety Condition

Definition (Column Safety)

A set of columns X is **safe** for a query Q if for all databases D, a sketch created on X for any horizontal partitioning of tables on X is safe

Overview

- Define generalized containment to reason about relationships of tuples from $Q(D)$ and $Q(D_{\mathcal{P}})$
- Construct universally quantified condition for a query and set of attributes X that implies safety of X

Experimental Setup

ILLINOIS INSTITUTE OF TECHNOLOGY

Workload and setup

- **Machine**:
  - 2 x AMD Opteron CPUs (12 cores)
  - 128 GB RAM
  - 4 x 1TB 7.2K HDs (RAID 5)
- **DBMS**
  - Postgres, MonetDB, System X[a]
- **Workloads**:
  - TPC-H [Tra09]
  - Real-world datasets
    - movie ratings (movie lens)
    - crime (City of Chicago Data portal)
    - Stack overflow

[a] Name redacted because of licensing restrictions

- **Microbenchmarks**
  - Overhead of capturing provenance sketches
  - Benefits of using provenance sketches
- **End-to-end experiments**
  - Execute a complete workload with and w/o PBDS

Microbenchmarks

ILLINOIS INSTITUTE OF TECHNOLOGY

Use Speed-up

	TPCH-1GB	TPCH-10GB	Crimes	Movies	StackOverflow
Min	**1.2**	**1.2**	1.4	1.5	32.4
Avg	5.5	13.3	5.1	2.6	46.0
Max	21.3	33.1	8.7	3.6	**66.3**

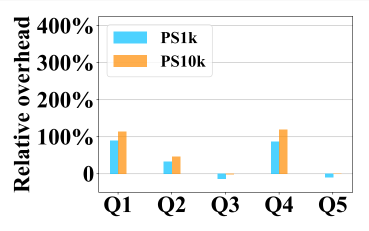
Capture Overhead

	TPCH-1GB	TPCH-10GB	Crimes	Movies	StackOverflow
Min	0.01	0.08	2.27	1.91	**-0.02**
Avg	0.35	0.38	2.68	0.90	0.56
Max	0.90	1.06	**3.08**	2.66	1.20

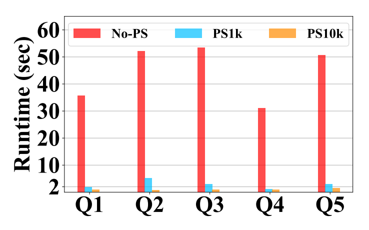
Setup

- **Database:** Postgres
- **Machine:** 2 x AMD Opteron CPUs (12 cores), 128 GB RAM, 4 x 1TB 7.2K HDs (RAID 5)
- **Dataset:** Stackoverflow data (real data) (186 GB) (Brin indexes / Postgres Zonemaps)

Capture



Use



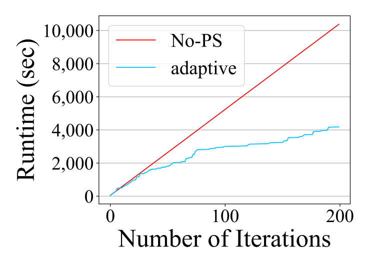
End-to-end experiments

ILLINOIS INSTITUTE OF TECHNOLOGY

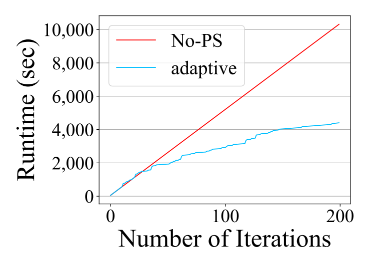
Setup

- Query workload consisting of a single template (200 instances)
- Compare DBMS (`No-PS`) against approach that creates and uses provenance sketches (`adaptive`)

0.7% selectivity



2% selectivity



Outline

1. Relevance-based Data Management
2. Provenance-based Data Skipping
3. Provenance Sketches
4. Sketch Capture and Use
5. Reusing Sketches and Sketch Safety
6. Experiments
7. **Relevance and The Value of Data**
8. Conclusions

An Objective Metric for Data Value

What is the value of data

- Organizations access their data through computations (queries)
  - only what is returned by these computations matters
- Data that does not contribute to any result could be deleted without affecting the result of any computation
  - $\Rightarrow$ irrelevant data has no value!
- **data relevance = data value**

What is needed?

- Aggregating relevance information across workloads
- Computing relevance for all queries is prohibitive
- Under-approximations could be ok

The Value of a Query

Query value

- Not all computations are equally important!

How to assess the value of a query?

- **User-provided metrics**
  - *queries from management are more important than queries from HR*
- **Feedback mechanism**
  - *being accessed by a relevant query makes data relevant*
  - *accessing relevant data makes a query relevant*
- **Economic models**

Dealing with Dark Data

Dark data

- data that has no or very little relevance
- is this data simply not useful or is its value unknown as of now?

Potential data value

- Some data has **potential value**
  - It could help the organization, but this has not been realized yet
- How to improve data management for dark data?
  - Explicit detection and controlled exploration of dark dark

Outline

The Vision

ILLINOIS INSTITUTE OF TECHNOLOGY

Relevance for Data Management

- **Make informed data management decisions based on relevance information**
- A fundamentally data-centric approach [?]

Relevance and Data Value

- Relevance = objective metric for the value of data

Current Work (PBDS)

Maintaining Sketches under Updates

- Maintain sketches incrementally / approximately when data is updated

Integration with Query Optimization & Self-tuning

- Cost-based decisions for when to create / use what sketches

Reuse beyond parameterized queries

- Normalize query structure
- Generalize our sound condition

Measure data value using sketches

- Sketches approximate what data is actually needed to answer queries
- Combine with metric for value of queries towards an objective metric for data value

Open Questions (RBDM)

Value of Queries

- How to assess the value of queries for an organization

Sketch reuse beyond parameterized queries

- Generalize sound condition

Sketches for semi-structured and graph data

- Larger space of sketching methods

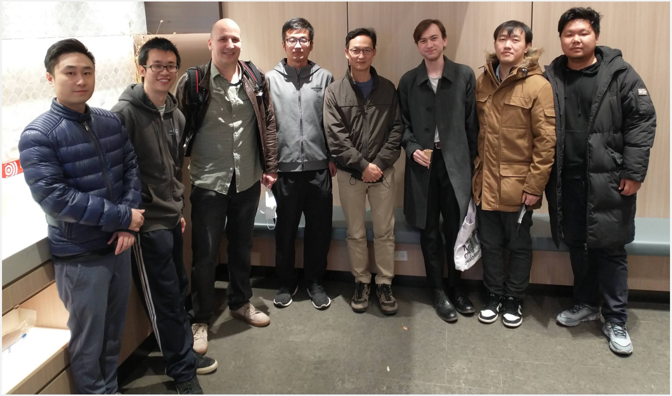
Sketches for approximate queries

- Allow both under- and over-approximations

Questions?

IITDBGroup

- **www**: `http://www.cs.iit.edu/~dbgroup/`



Outline

[AEHS+14] Lyublena Antova, Amr El-Helw, Mohamed A Soliman, Zhongxian Gu, Michalis Petropoulos, and Florian Waas.
Optimizing queries over partitioned tables in mpp systems.
In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pages 373–384. ACM, 2014.

[AFG+18] Bahareh Arab, Su Feng, Boris Glavic, Seokki Lee, Xing Niu, and Qitian Zeng.
GProM - A swiss army knife for your provenance needs.
IEEE Data Engineering Bulletin, 41(1):51–62, 2018.

[AKLT15] Sepehr Assadi, Sanjeev Khanna, Yang Li, and Val Tannen.
Algorithms for provisioning queries and analytics.
arXiv preprint arXiv:1512.06143, 2015.

[ANY04] S. Agrawal, V. Narasayya, and B. Yang.
Integrating vertical and horizontal partitioning into automated physical database design.
In *Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, pages 359–370. ACM, 2004.

[AS13] Daniar Achakeev and Bernhard Seeger.
Efficient bulk updates on multiversion b-trees.
Proceedings of the VLDB Endowment, 6(14):1834–1845, 2013.

[BC81] P.A. Bernstein and D. Chiu.
Using semijoins to solve relational queries.
JACM, 28(1), 1981.

[BG81] Philip A Bernstein and Nathan Goodman.
Power of natural semijoins.
SIAM Journal on Computing, 10(4):751–771, 1981.

[BS77] R. Bayer and M. Schkolnick.
Concurrency of operations on b-trees.
Acta informatica, 9(1):1–21, 1977.

[CGHJ12] Graham Cormode, Minos Garofalakis, Peter J Haas, and Chris Jermaine.
Synopses for massive data: Samples, histograms, wavelets, sketches.

Foundations and Trends in Databases, 4(1–3):1–294, 2012.

[Com79] D. Comer.
Ubiquitous b-tree.
ACM Computing Surveys (CSUR), 11(2):121–137, 1979.

[CY90] D.W. Cornell and P.S. Yu.
An effective approach to vertical partitioning for physical design of relational databases.
IEEE Transactions on Software engineering, 16(2):248–258, 1990.

[dBLSS21] Guy Van den Broeck, Anton Lykov, Maximilian Schleich, and Dan Suciu.
On the tractability of SHAP explanations.
In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 6505–6513. AAAI Press, 2021.

[DGTM17] Jiang Du, Boris Glavic, Wei Tan, and Renée J. Miller.
DeepSea: Adaptive Workload-Aware Partitioning of Materialized Views in Scalable Data Analytics.
In *Proceedings of the 20th International Conference on Extending Database Technology*, pages 198–209, 2017.

[dMB08] Leonardo Mendonça de Moura and Nikolaj Bjørner.
Z3: an efficient SMT solver.
In C. R. Ramakrishnan and Jakob Rehof, editors, *Tools and Algorithms for the Construction and Analysis of Systems, 14th International Conference, TACAS 2008, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2008, Budapest, Hungary, March 29-April 6, 2008. Proceedings*, volume 4963 of *Lecture Notes in Computer Science*, pages 337–340. Springer, 2008.

[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.

[Gla21] Boris Glavic.
Data provenance - origins, applications, algorithms, and models.
Foundations and Trends® in Databases, 9(3-4):209–441, 2021.

[Gra06] Goetz Graefe.
B-tree indexes for high update rates.
ACM Sigmod Record, 35(1):39–44, 2006.

[Gre11] T.J. Green.
Containment of conjunctive queries on annotated relations.
Theory of Computing Systems, 49(2):429–459, 2011.

[GZ19] Amirata Ghorbani and James Y. Zou.
Data shapley: Equitable valuation of data for machine learning.
In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pages 2242–2251, 2019.

[Ioa03] Yannis Ioannidis.
The history of histograms (abridged).
In *Proceedings of the 29th international conference on Very large data bases-Volume 29*, pages 19–30. VLDB Endowment, 2003.

[ISW11] Robert Ikeda, Semih Salihoglu, and Jennifer Widom.
Provenance-based refresh in data-oriented workflows.
In *Proceedings of the 20th ACM international conference on Information and knowledge management*, CIKM '11, pages 1659–1668, New York, NY, USA, 2011. ACM.

[KGFB22] Oliver Kennedy, Boris Glavic, Juliana Freire, and Mike Brachmann.
The right tool for the job: Data-centric workflows in vizier.
IEEE Data Eng. Bull., 45(3):129–144, 2022.

[LBKS20] Ester Livshits, Leopoldo E. Bertossi, Benny Kimelfeld, and Moshe Sebag.
The shapley value of tuples in query answering.
In *23rd International Conference on Database Theory, ICDT 2020, March 30-April 2, 2020, Copenhagen, Denmark*, pages 20:1–20:19, 2020.

[LLS13] Justin J Levandoski, David B Lomet, and Sudipta Sengupta.
The bw-tree: A b-tree for new hardware platforms.
In *Data Engineering (ICDE), 2013 IEEE 29th International Conference on*, pages 302–313. IEEE, 2013.

[LM97] A.Y. Levy and I.S. Mumick.

Query optimization by predicate move-around, August 19 1997.
US Patent 5,659,725.

[MGMS10] A. Meliou, W. Gatterbauer, K.F. Moore, and D. Suciu.
The Complexity of Causality and Responsibility for Query Answers and non-Answers.
Proceedings of the VLDB Endowment, 4(1):34–45, 2010.

[Moe98] Guido Moerkotte.
Small Materialized Aggregates: A light weight index structure for data warehousing.
In *VLDB*, pages 476–487, 1998.

[MSM+22] Naga Nithin Manne, Shilvi Satpati, Tanu Malik, Amitabha Bagchi, Ashish Gehani, and Amitabh Chaudhary.
Chex: Multiversion replay with ordered checkpoints.
arXiv preprint arXiv:2202.08429, 2022.

[Mul90] James K. Mullin.
Optimal semijoins for distributed database systems.
IEEE Transactions on Software Engineering, 16(5):558–560, 1990.

[NLL+21] Xing Niu, Ziyu Liu, Pengyuan Li, Boris Glavic, Dieter Gawlick, Vasudha Krishnaswamy, Zhen Hua Liu, and Danica Porobic.
Provenance-based data skipping.
Proceedings of the VLDB Endowment, 15(3):451 – 464, 2021.

[PCZ12] Andrew Pavlo, Carlo Curino, and Stanley Zdonik.
Skew-aware automatic database partitioning in shared-nothing, parallel oltp systems.
In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, pages 61–72. ACM, 2012.

[PW18] Fotis Psallidas and Eugene Wu.
Smoke: Fine-grained lineage at interactive speed.
Proc. VLDB Endow., 11(6):719–732, 2018.

[RJ17] Tilmann Rabl and Hans-Arno Jacobsen.
Query centric partitioning and allocation for partially replicated database systems.

In *Proceedings of the 2017 ACM International Conference on Management of Data*, pages 315–330. ACM, 2017.

[SFWW16] Liwen Sun, Michael J Franklin, Jiannan Wang, and Eugene Wu.
Skipping-oriented partitioning for columnar layouts.
Proceedings of the VLDB Endowment, 10(4):421–432, 2016.

[TMS+14] Rebecca Taft, Essam Mansour, Marco Serafini, Jennie Duggan, Aaron J ElmoreA, Ashraf Aboulnaga, Andrew Pavlo, and Michael Stonebraker.
E-store: Fine-grained elastic partitioning for distributed transaction processing systems.
Proceedings of the VLDB Endowment, 8(3), 2014.

[Tra09] Transaction Processing Council.
TPC-H Benchmark Specification, 2009.

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

[ZAN+19] Qi Zhou, Joy Arulraj, Shamkant B. Navathe, William Harris, and Dong Xu.
Automated verification of query equivalence using satisfiability modulo theories.
PVLDB, 12(11):1276–1288, 2019.

Outline

1. Relevance-based Data Management
2. Provenance-based Data Skipping
3. Provenance Sketches
4. Sketch Capture and Use
5. Reusing Sketches and Sketch Safety
6. Experiments
7. Relevance and The Value of Data
8. Conclusions

Definition (Generalized Containment)

$R \preceq_{\Psi} R'$ for two relations $R(a_1, \ldots, a_n)$ and $R'(b_1, \ldots, b_n)$ if there exists a mapping $\mathcal{M} \subseteq R \times R'$ such that:

- $\forall t \in R : \exists t' \in R' : \mathcal{M}(t, t')$
- $\forall t, t_1, t_2 : \mathcal{M}(t, t_1) \wedge \mathcal{M}(t, t_2) \rightarrow t_1 = t_2$
- $\forall (t, t') \in \mathcal{M} : (t, t') \models \Psi$