Reexamining some Holy Grails of Provenance

Boris Glavic Renée J. Miller

University of Toronto

UofT:DB Group

TaPP 2011, June 21, 2011

Insensitivity

Insensitivity to Query Rewrite

- Equivalent queries have the same provenance
- $Q \equiv Q' \Rightarrow \mathcal{P}(Q, I, t) = \mathcal{P}(Q', I, t)$

Insensitivity

Insensitivity to Query Rewrite

- Equivalent queries have the same provenance
- $Q \equiv Q' \Rightarrow \mathcal{P}(Q, I, t) = \mathcal{P}(Q', I, t)$
- Caveat: Which queries are equivalent?
  - Set vs. Bag semantics
  - Query language / Operators

Stability

Stability with Respect to Query Language Extension

- Extend query language with new operators ⇒ no change to provenance of queries that do not use new operators

Where

Where [Buneman et al., 2003]

Captures which attribute values in the result of a query have been copied from which attribute values in the instance.

Representation: $\mathbb{P}(Attr(I))$

- **Where**: Operator-level syntax-based annotation propagation
- **IWhere**: Insensitive variant: Union of *Where* for all Q' with $Q' \equiv Q$

Where

Where

- Sensitive, traditionally attributed to being based on query syntax
- Depends on the internal data-flow inside the query
  - How values are routed through the query

IWhere

- Insensitive by combining *Where* for all equivalent queries
- Counterintuitive effect that if (R, t, A) is in the provenance then all (R, t', A) with $t.A = t'.A$ are in the provenance too.
  - Reason: Can construct equivalent query adding self-join on A

Where

Example

$$Q_a = R$$

$$Q_b = \pi_{A,B}(R \bowtie_{A=C} \pi_{A \rightarrow C, B \rightarrow D}(R))$$

R

	A	B
r_1	1	2
r_2	1	3
r_3	2	3
r_4	2	5

Q_a & Q_b

	A	B
a_1	1	2
a_2	1	3
a_3	2	3
a_4	2	5

Provenance

$$Where(Q_a, a_1, A) = \{(r_1, A)\}$$

$$Where(Q_b, a_1, A) = \{(r_1, A), (r_2, A)\}$$

$$IWhere(Q_a, a_1, A) = IWhere(Q_b, a_1, A) = \{(r_1, A), (r_2, A)\}$$

Arguments for Insensitivity

1. Traditionally observed as advantageous in database research
   ⇒ Tradition not a solid argument
2. External implementation of *sensitive* semantics. Computing provenance for a query different from the one that will be executed by DBMS
   ⇒ No way to solve this, but provenance based on user query seems to be reasonable
3. Implementation of *sensitive* semantics in DB-engine limits optimizer search space
   ⇒ Insensitive semantics may be harder to compute
   ⇒ Lack of practical experience
   ⇒ "Realistic" sensitivity example?

Instability of IWhere

- *IWhere* is union of *Where* for all equivalent queries

Instability of IWhere

- *IWhere* is union of *Where* for all equivalent queries
- e.g., SPJ and USPJ equivalences are different

Instability of IWhere

- *IWhere* is union of *Where* for all equivalent queries
- e.g., SPJ and USPJ equivalences are different
- e.g., union Q with a join of Q with some other relation

Instability of IWhere

- *IWhere* is union of *Where* for all equivalent queries
- e.g., SPJ and USPJ equivalences are different
- e.g., union Q with a join of Q with some other relation
- Let *UWhere* be *IWhere* for USPJ queries

Instability of IWhere

- *IWhere* is union of *Where* for all equivalent queries
- e.g., SPJ and USPJ equivalences are different
- e.g., union Q with a join of Q with some other relation
- Let *UWhere* be *IWhere* for USPJ queries
- $\Rightarrow$ *UWhere* an attribute value is annotated with all annotations from attribute positions that have the same value

Instability of IWhere

Example

$Q_a = R$

R

	A	B
r_1	1	2
r_2	1	3
r_3	2	3
r_4	2	5

S

	C
s_1	2
s_2	3

Q_a & Q_b

	A	B
a_1	1	2
a_2	1	3
a_3	2	3
a_4	2	5

Provenance

$$Where(Q_a, a_3, A) = \{(r_3, A)\}$$

$$IWhere(Q_a, a_3, A) = \{(r_3, A), (r_4, A)\}$$

$$UWhere(Q_a, a_3, A) = \{(r_3, A), (r_4, A), (r_1, B), (s_1, C)\}$$

Conclusions

Take Away Messages

- Be careful how to achieve a property
- Insensitivity less applicable to semantics that address internal data-flow
  - Queries with the same external but possibly different internal behaviour have the same provenance

Some Things I'd Like to See

- "Declarative" Semantics ⇒ derive operator-level construction
- Semantics model processing, but have a insensitive "core"
- The never-ending quest: Deal with Negation
- Other data-models (order)

Questions

Semantics		Sound	Complete	Responsible	Insensitive (set)	Insensitive (bag)	Stable
Why	Wit	-	X	-	X	X	X
	Why	-	X	-	-	?	X
	IWhy	-	X	X	X	X	X
Where	Where	-	-	-	-	?	X
	IWhere	-	-	-	X	X	-
How		-	X	-	-	X	X
Lineage-based	Lineage	X	X	-	-	-	X
	PI-CS	X	X	-	-	-	X
	C-CS	X	-	-	-	-	X
Causality		-	X	X	X	X	X

Semantics Summary

Representation	Used by
$\mathbb{P}(Attr(I))$	Where, IWhere
$\mathbb{P}(\mathbb{P}(Tuple(I)))$	Wit, Why, IWhy
$\mathbb{N}[Tuple(I)]$	How
$\{< R_1^*, \ldots, R_n^* >\mid R_i^* \subseteq R_i(Q)\}$	Lineage
$\mathbb{P}(\{< t_1, \ldots, t_n >\mid t_i \in R_i(Q) \vee t_i = \perp\})$	PI-CS, C-CS
$\mathbb{P}(Tuple(I))$	Causality

Sound, Complete, Responsible

Sound, Complete, Responsible

- **Sound:** Provenance of t produces nothing different from t.
  - $t' \neq t \Rightarrow t' \notin Q(\mathcal{P}(Q, I, t))$

Sound, Complete, Responsible

Sound, Complete, Responsible

- **Sound:** Provenance of t produces nothing different from t.
  - $t' \neq t \Rightarrow t' \notin Q(\mathcal{P}(Q, I, t))$
- Caveat: Semantics of evaluating query over provenance

Sound, Complete, Responsible

Sound, Complete, Responsible

- **Sound:** Provenance of t produces nothing different from t.
  - $t' \neq t \Rightarrow t' \notin Q(\mathcal{P}(Q, I, t))$
- **Complete:** Provenance of t produces at least t
  - $t \in Q(\mathcal{P}(Q, I, t))$

Sound, Complete, Responsible

Sound, Complete, Responsible

- **Sound:** Provenance of t produces nothing different from t.
  - $t' \neq t \Rightarrow t' \notin Q(\mathcal{P}(Q, I, t))$
- **Complete:** Provenance of t produces at least t
  - $t \in Q(\mathcal{P}(Q, I, t))$
- Caveat: Semantics of evaluating query over provenance

Sound, Complete, Responsible

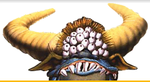
Sound, Complete, Responsible

- **Sound:** Provenance of t produces nothing different from t.
  - $t' \neq t \Rightarrow t' \notin Q(\mathcal{P}(Q, I, t))$
- **Complete:** Provenance of t produces at least t
  - $t \in Q(\mathcal{P}(Q, I, t))$
- **Responsible:** Every tuple in the provenance of t is necessary to derive t

Sound, Complete, Responsible

Sound, Complete, Responsible

- **Sound:** Provenance of t produces nothing different from t.
  - $t' \neq t \Rightarrow t' \notin Q(\mathcal{P}(Q, I, t))$
- **Complete:** Provenance of t produces at least t
  - $t \in Q(\mathcal{P}(Q, I, t))$
- **Responsible:** Every tuple in the provenance of t is necessary to derive t
- Caveat: . . . from every alternative derivation in the provenance . . .
  - $\Rightarrow$ factor provenance into alterative derivations
- Caveat: Different ways to model that.
  - E.g., $\forall t' \in \mathcal{P}(Q, I, t) : t \notin Q(\mathcal{P}(Q, I, t) - \{t'\})$



Sound, Complete, Responsible

Sound, Complete, Responsible

Example

$$Q_b = \pi_{A,B}(R \bowtie_{A=C} \pi_{A \to C, B \to D}(R))$$

R

	A	B
r_1	1	2
r_2	1	3
r_3	2	3
r_4	2	5

Q_b

	A	B
a_1	1	2
a_2	1	3
a_3	2	3
a_4	2	5

Provenance

$$\mathcal{P}(Q_b, I, a_1) = \{r_1, r_2\}$$

Lineage-based

Lineage-based [Cui et al., 2000]

Operator-level declarative semantics similar to *Why*. Provenance is modeled as a list of subsets of the relations accessed by the query (leafs of the algebra tree of Q)

Representation: $\{< R_1^*, \ldots, R_n^* > \mid R_i^* \subseteq R_i(Q)\}$

- **Lineage**: List of subsets of the algebra-tree nodes

Lineage-based

Lineage-based [Glavic et al., 2009]

Provenance is modeled as a set of **witness lists**. A witness list is a list of tuples - one from each relation accessed by the query.

Representation: $\mathbb{P}(\{< t_1, \ldots, t_n >| \, t_i \in R_i(Q) \vee t_i = \perp\})$

- **PI-CS**: *Lineage* with different representation and broader query language coverage
- **C-CS**: Similar to *Where* but with tuple granularity

Lineage-based

Example

$$Q_c = \pi_A(R) \cup \pi_B(R)$$
$$Q_d = \pi_A(R \bowtie_{B=C} S)$$

R

	A	B
r_1	1	2
r_2	1	3
r_3	2	3
r_4	2	5

S

	C
s_1	2
s_2	3

Q_c

	A
c_1	1
c_2	2
c_3	3
c_4	5

Q_d & Q_e

	A
d_1	1
d_2	2

Provenance

$$PI-CS(Q_c, c_2) = \{<\perp, r_1>, <r_3, \perp>, <r_4, \perp>\}$$
$$PI-CS(Q_d, d_1) = \{<r_1, s_1>, <r_2, s_2>\}$$
$$C-CS(Q_d, d_1) = \{<r_1, \perp>, <r_2, \perp>\}$$
$$Lineage(Q_d, d_1) = <\{r_1, r_2\}, \{s_1, s_2\}>$$

Why

Why [Buneman et al., 2003]

Why-provenance models provenance as a set of **witnesses**. A witness w for a tuple t is a subset of the instance I where $t \in Q(w)$.

Representation: $\mathbb{P}(\mathbb{P}(Tuple(I)))$

- **Wit**: Set of all witnesses
- **Why**: Query-syntax based "proof-witnesses"
- **IWhy**: Minimal elements from *Wit* resp. *Why*

Why

Example

$$Q_a = R$$

R

	A	B
r_1	1	2
r_2	1	3
r_3	2	3
r_4	2	5

Q_a

	A	B
a_1	1	2
a_2	1	3
a_3	2	3
a_4	2	5

Provenance

$$Wit(Q_a, a_1) = \{J \mid J \subset R \wedge r_1 \in J\}$$

$$Why(Q_a, a_1) = \{\{r_1\}\}$$

$$IWhy(Q_a, a_1) = \{\{r_1\}\}$$

How

Provenance Semirings [Green et al., 2007]

Tuples of relations annotated with elements from a semiring. Annotation propagation defined for positive relational algebra as operations of the semiring (set difference and aggregation later).
Representation: $\mathbb{N}(\mathit{Tuple}(I))$

- **How**: Most general form of annotations: polynomials over variable representing the instance tuples. Addition indicates alternative use of tuples; multiplication conjunctive use.

How

Example

$$Q_b = \pi_{A,B}(R \bowtie_{A=C} \pi_{A \to C, B \to D}(R))$$

R

	A	B
r_1	1	2
r_2	1	3
r_3	2	3
r_4	2	5

Q_b

	A	B
a_1	1	2
a_2	1	3
a_3	2	3
a_4	2	5

Provenance

$$How(Q_b, a_1) = r_1^2 + r_1 \times r_2$$

Causality

Causality [Meliou et al., 2010]

Provenance is modeled as a set of causes. A **cause** $c \in I$ for a tuple t is defined as follows:

1. $t \notin Q(I - \{c\})$
2. there exists a set $C \subset I$ called contingency so that $t \in Q(I - C)$ and $t \notin Q(I - C - \{c\})$

Representation: $\mathbb{P}(Tuple(I))$

- **Causality**: Set of all causes

Causality

Example

$Q_a = R$

R

	A	B
r_1	1	2
r_2	1	3
r_3	2	3
r_4	2	5

$\mathbf{Q_a}$

	A	B
a_1	1	2
a_2	1	3
a_3	2	3
a_4	2	5

Provenance

$$Causality(Q_a, a_1) = \{r_1\}$$

Insensitivity to Query Rewrite

Insensitivity to Query Rewrite

- Equivalent queries have the same provenance
- $Q \equiv Q' \Rightarrow \mathcal{P}(Q, I, t) = \mathcal{P}(Q', I, t)$
- Set resp. Bag semantics

Overview

- **Insensitive (Set, Bag)** *Wit*, *IWhy*, *IWhere*, *Causality*
- **Insensitive (Bag)** *How*
- **Sensitive:** *Lineage*, *PI-CS*, *C-CS*, *Where*

Why

Wit

- Defined over black-box behaviour of query ⇒ trivially insensitive

Why

- Sensitive, traditionally attributed to being based on query syntax
- *Why* may contain tuples that do not contribute to t
- ⇒ Equivalent queries that apply redundant computations may contain larger provenance
- Caveat: But why does this argument not apply to *Wit*?
- Positive queries: super-set of a witness is also a witness ⇒ tuples used by redundant computations are in *Wit*

IWhy

Why

Example

$Q_a = R$

$Q_b = \pi_{A,B}(R \bowtie_{A=C} \pi_{A\rightarrow C, B\rightarrow D}(R))$

R

	A	B
r_1	1	2
r_2	1	3
r_3	2	3
r_4	2	5

Q_a & Q_b

	A	B
a_1	1	2
a_2	1	3
a_3	2	3
a_4	2	5

Provenance

$$Wit(Q_a, a_1) = Wit(Q_b, a_1) = \{J \mid J \subset R \wedge r_1 \in J\}$$

$$Why(Q_a, a_1) = \{\{r_1\}\}$$

$$Why(Q_b, a_1) = \{\{r_1\}, \{r_1, r_2\}\}$$

$$IWhy(Q_a, a_1) = IWhy(Q_b, a_1) = \{\{r_1\}\}$$

Lineage-based

- Provenance representation based on query syntax
- Trivial examples for sensitivity based on reordering of the arguments of commutative operators

Lineage-based

Example

$$Q_d = \pi_A(R \bowtie_{B=C} S)$$
$$Q_e = \pi_A(S \bowtie_{B=C} R)$$

R

	A	B
r_1	1	2
r_2	1	3
r_3	2	3
r_4	2	5

S

	C
s_1	2
s_2	3

Q_d & Q_e

	A
d_1	1
d_2	2

Provenance

$$Lineage(Q_d, d_1) = < \{r_1, r_2\}, \{s_1, s_2\} >$$
$$Lineage(Q_e, d_1) = < \{s_1, s_2\}, \{r_1, r_2\} >$$
$$PI-CS(Q_d, d_1) = \{< r_1, s_1 >, < r_2, s_2 >\}$$
$$PI-CS(Q_e, d_1) = \{< s_1, r_1 >, < s_2, r_2 >\}$$

How

- Sensitive (Set):
- Insensitive (Bag): Operator semantics defined to take bag semantics into account

How

Example

$Q_a = R$

$Q_b = \pi_{A,B}(R \bowtie_{A=C} \pi_{A \to C, B \to D}(R))$

R

	A	B
r_1	1	2
r_2	1	3
r_3	2	3
r_4	2	5

Q_a & Q_b

	A	B
a_1	1	2
a_2	1	3
a_3	2	3
a_4	2	5

Provenance

$$How(Q_a, a_1) = r_1$$

$$How(Q_b, a_1) = r_1^2 + r_1 \times r_2$$

Causality

- Trivially insensitive: Defined over the black-box behaviour of a query

Causality

Example

$Q_a = R$

$Q_b = \pi_{A,B}(R \bowtie_{A=C} \pi_{A \rightarrow C, B \rightarrow D}(R))$

R

	A	B
r_1	1	2
r_2	1	3
r_3	2	3
r_4	2	5

$\mathbf{Q_a}$ & $\mathbf{Q_b}$

	A	B
a_1	1	2
a_2	1	3
a_3	2	3
a_4	2	5

Provenance

$$Causality(Q_a, a_1) = Causality(Q_b, a_1) = \{r_1\}$$