Provenance Semirings

T.J. Green, G. Karvounarakis, V. Tannen
University of Pennsylvania

Principles of Provenance (PrOPr)

Philadelphia, PA   June 26, 2007

Provenance

- First studied in data warehousing
  - Lineage [Cui,Widom,Wiener 2000]
- Scientific applications (to assess quality of data)
  - Why-Provenance [Buneman,Khanna,Tan 2001]
- Our interest: P2P data sharing in the ORCHESTRA system (project headed by Zack Ives)
  - Trust conditions based on provenance
  - Deletion propagation

Annotated relations

- Provenance: an annotation on tuples
- Our observation: propagating provenance/lineage through views is similar to querying
  - Incomplete Databases (conditional tables)
  - Probabilistic Databases (independent tuple tables)
  - Bag Semantics Databases (tuples with multiplicities)
- Hence we look at queries on **relations with annotated tuples**

Incomplete databases: boolean *C*-tables

R

a	b	c	p
d	b	e	r
f	g	e	s

boolean variables

semantics: a set of instances

$$I(R)=\left\{\emptyset, \boxed{a\ b\ c}, \boxed{d\ b\ e}, \boxed{f\ g\ e}, \boxed{\begin{array}{c} a\ b\ c \\ d\ b\ e \end{array}}, \boxed{\begin{array}{c} a\ b\ c \\ f\ g\ e \end{array}}, \boxed{\begin{array}{c} d\ b\ e \\ f\ g\ e \end{array}}, \boxed{\begin{array}{c} a\ b\ c \\ d\ b\ e \\ f\ g\ e \end{array}}\right\}$$

Imielinski & Lipski (1984): queries on $\mathcal{C}$-tables

R

a	b	c	p
~~d~~	~~b~~	~~e~~	~~r~~
f	g	e	s

union of conjunctive queries (UCQ)

$$q(x,z) \text{ :- } R(x,\_,z)_{\boxed{r}},\ R(\_,\_,z)_{\boxed{s}}$$

$$q(x,z) \text{ :- } R(x,y,\_)_{\boxed{r}},\ R(\_,y,z)_{\boxed{r}}$$

q(R)

a c	$(p \wedge p) \vee (p \wedge p)$	=	p
a e	$p \wedge r$		$p \wedge r$
d c	$r \wedge p$		$p \wedge r$
d e	$\boxed{(r \wedge r)} \vee \boxed{(r \wedge r)} \vee \boxed{(r \wedge s)}$		r
f e	$(s \wedge s) \vee (s \wedge s) \vee (s \wedge r)$		s

p=true

r=false

s=true

a c
f e

Why-provenance/lineage

Which input tuples contribute to the presence of a tuple in the output?

same query

tuple ids

R

a	b	c	p
d	b	e	r
f	g	e	s

q(R)

a	c	{p}
a	e	{p,r}
d	c	{p,r}
d	e	{r,s}
f	e	{r,s}

[Cui,Widom,Wiener 2000]

[Buneman,Khanna,Tan 2001]

$\mathcal{C}$–tables vs. Why-provenance

a c	$(p \wedge p) \vee (p \wedge p)$
a e	$p \wedge r$
d c	$r \wedge p$
d e	$(r \wedge r) \vee (r \wedge r) \vee (r \wedge s)$
f e	$(s \wedge s) \vee (s \wedge s) \vee (s \wedge r)$

c-table calculations

Why-provenance calculations

a c	$(\{p\} \cup \{p\}) \cup (\{p\} \cup \{p\})$
a e	$\{p\} \cup \{r\}$
d c	$\{r\} \cup \{p\}$
d e	$(\{r\} \cup \{r\}) \cup (\{r\} \cup \{r\}) \cup (\{r\} \cup \{s\})$
f e	$(\{s\} \cup \{s\}) \cup (\{s\} \cup \{s\}) \cup (\{s\} \cup \{r\})$

The structure of the calculations is the same!

Another analogy, with bag semantics

tuple multiplicities

R

a	b	c	2
d	b	e	5
f	g	e	1

c-table calculations

a c	$(p \wedge p) \vee (p \wedge p)$
a e	$p \wedge r$
d c	$r \wedge p$
d e	$(r \wedge r) \vee (r \wedge r) \vee (r \wedge s)$
f e	$(s \wedge s) \vee (s \wedge s) \vee (s \wedge r)$

same query

q(R)

a c	8
a e	10
d c	10
d e	55
f e	7

multiplicity calculations

a c	$2 \cdot 2 + 2 \cdot 2$
a e	$2 \cdot 5$
d c	$5 \cdot 2$
d e	$5 \cdot 5 + 5 \cdot 5 + 5 \cdot 1$
f e	$1 \cdot 1 + 1 \cdot 1 + 1 \cdot 5$

The structure of the calculations is the same!

Abstracting the structure of these calculations

	C-tables	Bags	Why-provenance	Abstract
join	$\wedge$	$\cdot$	$\cup$	$\cdot$
union	$\vee$	$+$	$\cup$	$+$

abstract calculations

a c	$(p \cdot p) + (p \cdot p)$
a e	$p \cdot r$
d c	$r \cdot p$
d e	$(r \cdot r) + (r \cdot r) + (r \cdot s)$
f e	$(s \cdot s) + (s \cdot s) + (s \cdot r)$

These expressions capture the abstract structure of the calculations, which encodes the logical derivation of the output tuples

We shall use these expressions as provenance

Positive K-relational algebra

- We define an RA+ on K-relations:
  - The $\cdot$ corresponds to join:
  - The + corresponds to union and projection
  - 0 and 1 are used for selection predicates

  - Details in the paper (but recall how we evaluated the UCQ q earlier and we will see another example later)

RA+ identities imply semiring structure!

- Common RA+ identities
  - Union and join are associative, commutative
  - Join distributes over union
  - etc. (but not idempotence!)

These identities hold for RA+ on K-relations

iff

$(K, +, \cdot, 0, 1)$ is a commutative semiring

$(K,+,0)$ is a commutative monoid

$(K, \cdot,1)$ is a commutative monoid

$\cdot$ distributes over $+$, etc

Calculations on annotated tables are particular cases

$(\mathbb{B}, \vee, \wedge, \text{false}, \text{true})$	usual relational algebra
$(\mathbb{N}, +, \cdot, 0, 1)$	bag semantics
$(\text{PosBool}(B), \vee, \wedge, \text{false}, \text{true})$	boolean *C*-tables
$(\mathcal{P}(\Omega), \cup, \cap, \emptyset, \Omega)$	probabilistic event tables
$(\mathcal{P}(X), \cup, \cup, \emptyset, \emptyset)$	lineage/why-provenance

Provenance Semirings

- $X = \{p, r, s, \ldots\}$: indeterminates (provenance "tokens" for base tuples)
- $\mathbb{N}[X]$: multivariate polynomials with coefficients in $\mathbb{N}$ and indeterminates in X
- $(\mathbb{N}[X], +, \cdot, 0, 1)$ is the most "general" commutative semiring: its elements abstract calculations in **all** semirings
- $\mathbb{N}[X]$ -relations **are the relations with provenance!**
  - The polynomials capture the propagation of provenance through (positive) relational algebra

A provenance calculation

q(x,z) :- R(x, _,z), R(_, _,z)

q(x,z) :- R(x,y, _), R(_ ,y,z)

R

a	b	c	p
d	b	e	r
f	g	e	s

q(R)

a	c	$2p^2$
a	e	pr
d	c	pr
d	e	$2r^2 + rs$
f	e	$2s^2 + rs$

Why-provenance

a	c	{p}
a	e	{p,r}
d	c	{p,r}
d	e	{r,s}
f	e	{r,s}

same why-provenance, different polynomials

- Not just *why-* but also *how-*provenance (encodes derivations)!
- More informative than why-provenance

Further work

- Application: P2P data sharing in the ORCHESTRA system:
  - Need to express trust conditions based on provenance of tuples
  - Incremental propagation of deletions
  - Semiring provenance itself is incrementally maintainable
- Future extensions:
  - full relational algebra: For difference we need semirings with "proper subtraction"
  - richer data models: nested relations/complex values, XML