Lecture 11:
Provenance and Data privacy

December 8, 2010

Outline

- Database provenance
  - Slides based on Val Tannen's Keynote talk at EDBT 2010
- Data privacy
  - Slides from my UW colloquium talk in 2005

Data Provenance

provenance, n.

The fact of coming from some particular source or quarter∅ *origin, derivation* [Oxford English Dictionary]

•**Data provenance** [BunemanKhannaTan 01]**:** aims to explain how a particular result (in an experiment, simulation, query, workflow, etc.) was derived.

•Most science today is **data-intensive**. Scientists, eg., biologists, astronomers, worry about data provenance all the time.

Provenance? Lineage? Pedigree?

- Cf. Peter Buneman:
  - Pedigree is for **dogs**
  - Lineage is for **kings**
  - Provenance is for **art**
- For data, let's be artistic (artsy?)

Database transformations?

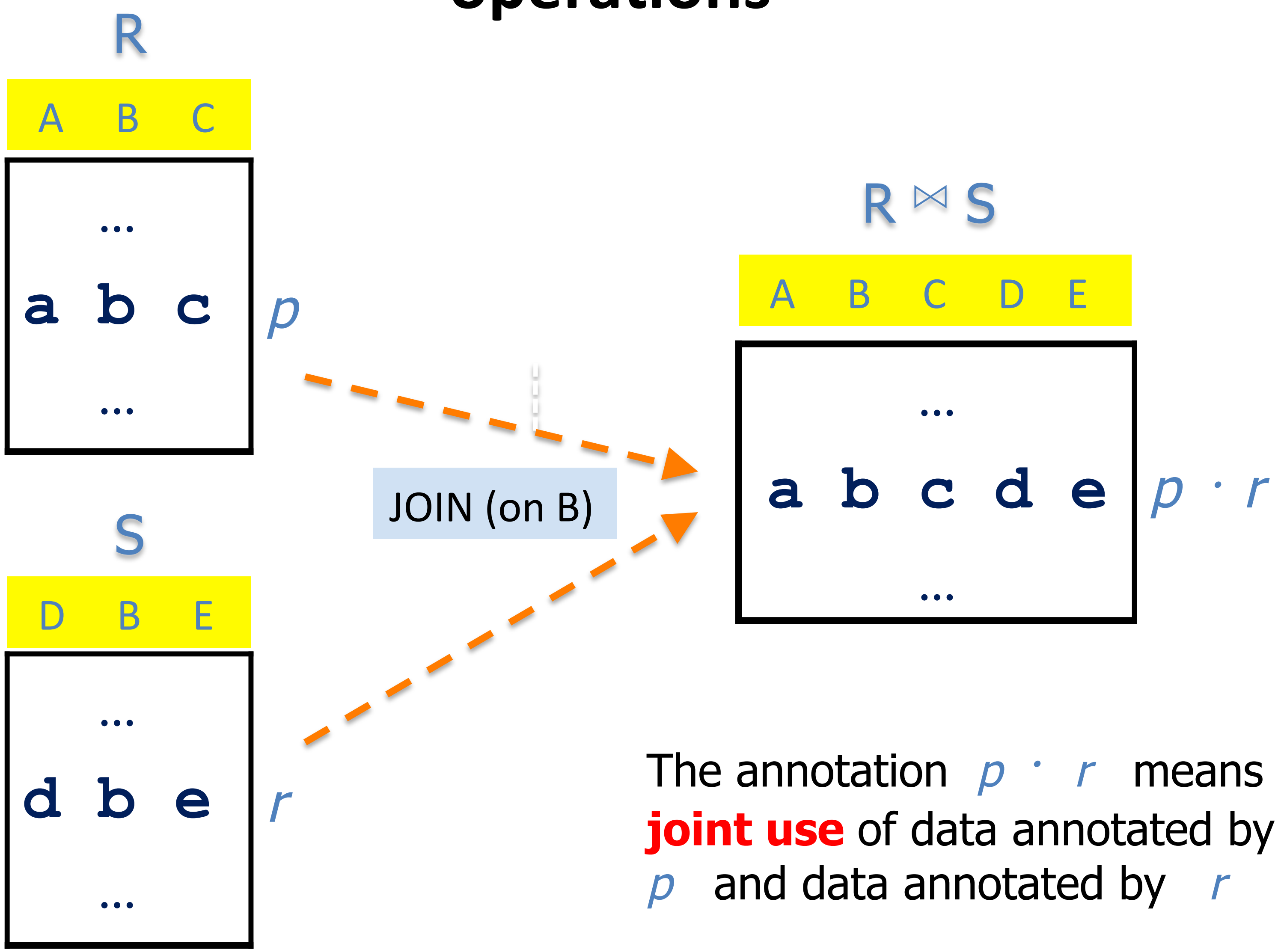
- Queries
- Views
- ETL tools
- Schema mappings (as used in data exchange)

Outline

- **What's with the semirings? Annotation propagation**
  [GK&T PODS 07, GKI&T VLDB 07]

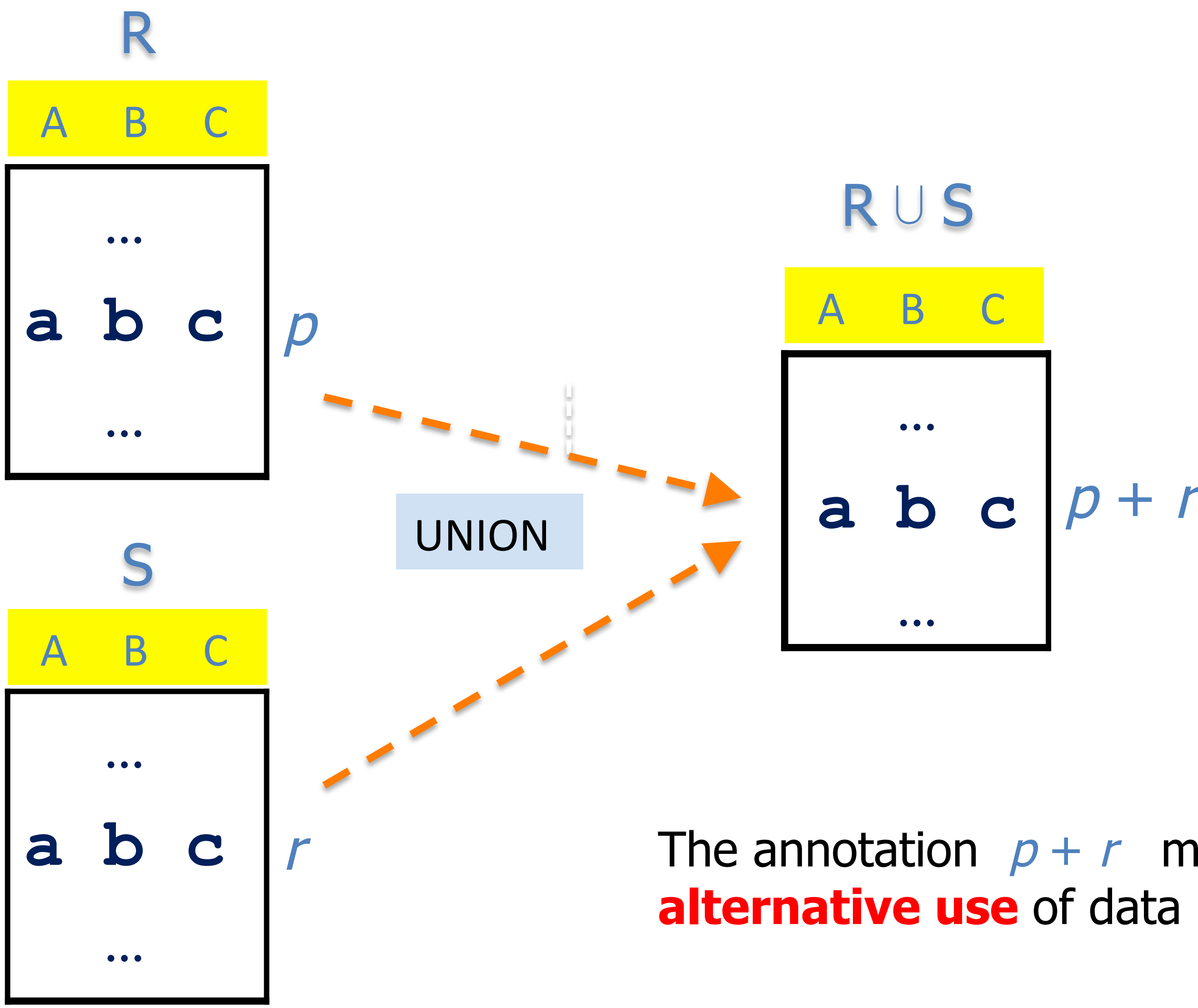
- **Housekeeping in the zoo of provenance models**

Propagating annotations through database operations



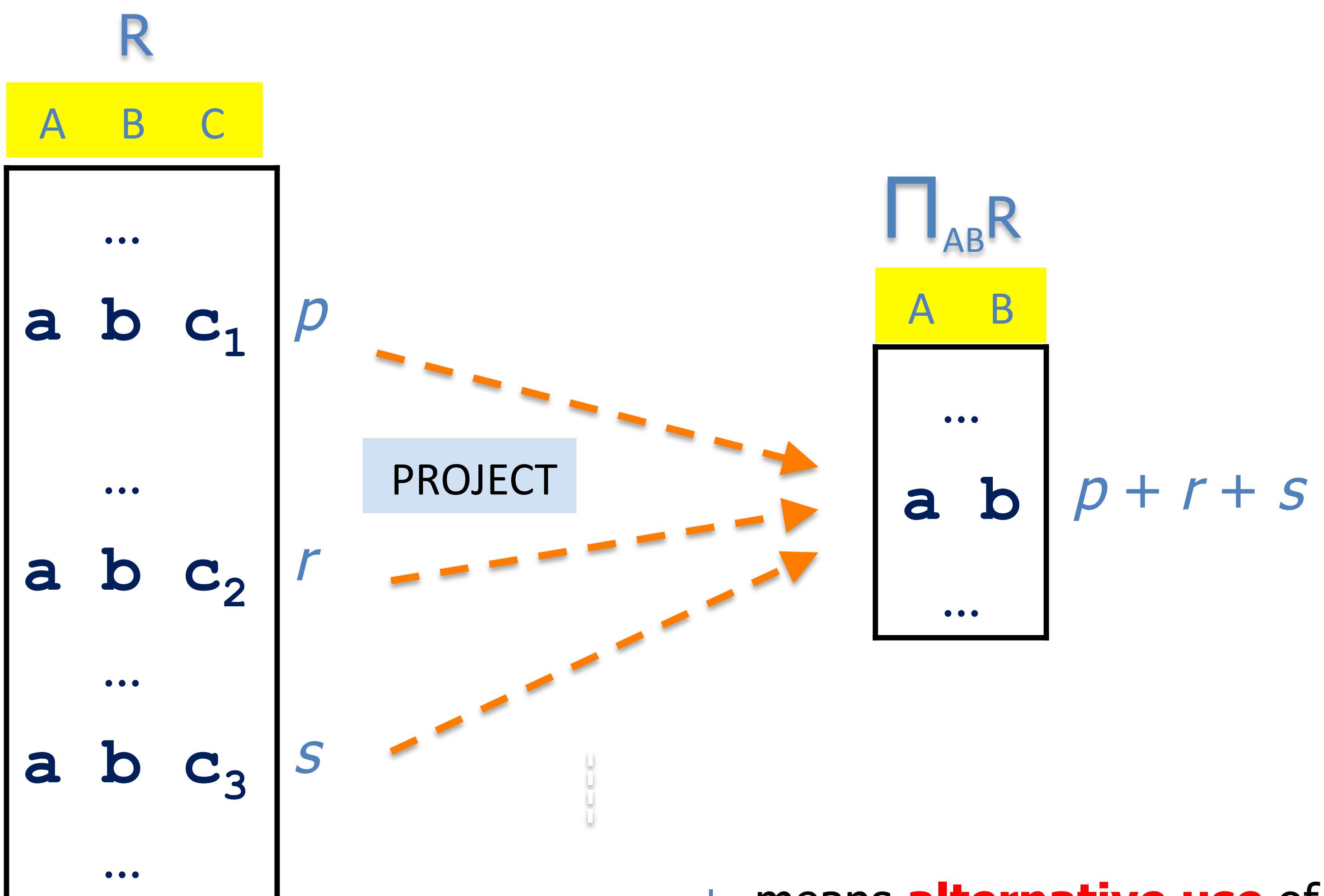
The annotation $p \cdot r$ means **joint use** of data annotated by p and data annotated by r

Another way to propagate annotations



The annotation $p + r$ means
alternative use of data

Another use of +



+ means **alternative use** of data

An example in positive relational algebra (SPJU)

$$\mathbf{Q} = \sigma_{C=e}\Pi_{AC}(\Pi_{AC}R \bowtie \Pi_{BC}R \cup \Pi_{AB}R \bowtie \Pi_{BC}R)$$

R

A	B	C	
a	b	c	p
d	b	e	r
f	g	e	s

A	C	
a	c	$(p \cdot p + p \cdot p) \cdot 0$
a	e	$p \cdot r \cdot 1$
d	c	$r \cdot p \cdot 0$
d	e	$(r \cdot r + r \cdot s + r \cdot r) \cdot 1$
f	e	$(s \cdot s + s \cdot r + s \cdot s) \cdot 1$

For selection we multiply
with two special annotations, 0 and 1

Summary so far

Summary so far

A space of annotations, K

Summary so far

A space of annotations, *K*

K-**relations**: every tuple annotated with some element from *K*.

Summary so far

A space of annotations, K

K-**relations**: every tuple annotated with some element from K.

Binary operations on K: $\cdot$ corresponds to joint use (join), and $+$ corresponds to alternative use (union and projection).

Summary so far

A space of annotations, K

K-**relations**: every tuple annotated with some element from K.

Binary operations on K: $\cdot$ corresponds to joint use (join), and $+$ corresponds to alternative use (union and projection).

We assume K contains special annotations 0 and 1.

Summary so far

A space of annotations, *K*

K-**relations**: every tuple annotated with some element from *K*.

Binary operations on *K*: $\cdot$ corresponds to joint use (join), and $+$ corresponds to alternative use (union and projection).

We assume *K* contains special annotations 0 and 1.

"Absent" tuples are annotated with 0!

Summary so far

A space of annotations, *K*

K-**relations**: every tuple annotated with some element from *K*.

Binary operations on *K*: $\cdot$ corresponds to joint use (join),
and $+$ corresponds to alternative use (union and projection).

We assume *K* contains special annotations 0 and 1.

"Absent" tuples are annotated with 0!

1 is a "neutral" annotation (no restrictions).

Summary so far

A space of annotations, K

K-**relations**: every tuple annotated with some element from K.

Binary operations on K: $\cdot$ corresponds to joint use (join), and $+$ corresponds to alternative use (union and projection).

We assume K contains special annotations 0 and 1.

"Absent" tuples are annotated with 0!

1 is a "neutral" annotation (no restrictions).

Algebra of annotations? What are the **laws** of $(K, +, \cdot, 0, 1)$?

Annotated relational algebra

- DBMS query optimizers assume certain equivalences:
  - union is associative, commutative
  - join is associative, commutative, distributes over union
  - projections and selections commute with each other and with union and join (when applicable)
  - Etc., but no $R \bowtie R = R \cup R = R$ (i.e., no idempotence, to allow for bag semantics)
- Equivalent queries should produce same annotations!

Annotated relational algebra

- DBMS query optimizers assume certain equivalences:
  - union is associative, commutative
  - join is associative, commutative, distributes over union
  - projections and selections commute with each other and with union and join (when applicable)
  - Etc., but no $R \bowtie R = R \cup R = R$ (i.e., no idempotence, to allow for bag semantics)
- Equivalent queries should produce same annotations!

Proposition. Above identities hold for queries on K-relations iff $(K, +, \cdot, 0, 1)$ is a **commutative semiring**

Annotated relational algebra

- DBMS query optimizers assume certain equivalences:
  - union is associative, commutative
  - join is associative, commutative, distributes over union
  - projections and selections commute with each other and with union and join (when applicable)
  - Etc., but no $R \bowtie R = R \cup R = R$ (i.e., no idempotence, to allow for bag semantics)
- Equivalent queries should produce same annotations!

- Hence, for each commutative semiring *K* we have a *K*-**annotated relational algebra**.

What is a commutative semiring?

An algebraic structure $(K, +, \cdot, 0, 1)$ where:

- K is the domain
- $+$ is associative, commutative, with 0 identity
- $\cdot$ is associative, with 1 identity
- $\cdot$ distributes over $+$
- $a \cdot 0 = 0 \cdot a = 0$

} **semiring**

- $\cdot$ is also **commutative**

Unlike ring, no requirement for inverses to $+$

Back to the example

R

A	B	C	
a	b	c	p
d	b	e	r
f	g	e	s

Q

A	C	
a	c	$(p \cdot p + p \cdot p) \cdot 0$
a	e	$p \cdot r \cdot 1$
d	c	$r \cdot p \cdot 0$
d	e	$(r \cdot r + r \cdot s + r \cdot r) \cdot 1$
f	e	$(s \cdot s + s \cdot r + s \cdot s) \cdot 1$

Using the laws: polynomials

R

A	B	C	
a	b	c	*p*
d	b	e	*r*
f	g	e	*s*

Q

A	C	
a	e	pr
d	e	$2r^2 + rs$
f	e	$rs + 2s^2$

Polynomials with coefficients in **N** and annotation tokens as indeterminates *p, r, s*

capture a very general form of **provenance**

Provenance reading of the polynomials

R

A	B	C	
a	b	c	p
d	b	e	r
f	g	e	s

Q

A	C	
a	e	pr
d	e	$2r^2 + rs$
f	e	$rs + 2s^2$

- three different ways to derive **d e**
- two of the ways use only r
- but they use it twice
- the third way uses r once and s once

Low-hanging fruit: deletion propagation

We used this in **Orchestra** [VLDB07]
for update propagation

R

A	B	C	
a	b	c	p
d	b	e	r
f	g	e	s

Q

A	C	
a	e	pr
d	e	$2r^2 + rs$
f	e	$rs + 2s^2$

Low-hanging fruit: deletion propagation

We used this in **Orchestra** [VLDB07]
for update propagation

R

A	B	C	
a	b	c	p
d	b	e	r
f	g	e	s

Q

A	C	
a	e	pr
d	e	$2r^2 + rs$
f	e	$rs + 2s^2$

Delete `d b e` from R ?

Low-hanging fruit: deletion propagation

We used this in **Orchestra** [VLDB07]
for update propagation

R

A	B	C	
a	b	c	p
d	b	e	r
f	g	e	s

Q

A	C	
a	e	pr
d	e	$2r^2 + rs$
f	e	$rs + 2s^2$

Delete `d b e` from R ?

Set $r = 0$!

Low-hanging fruit: deletion propagation

We used this in **Orchestra** [VLDB07]
for update propagation

R

A	B	C	
a	b	c	p
d	b	e	r
f	g	e	s

Q (crossed out)

A	C	
a	e	pr
d	e	$2r^2 + rs$
f	e	$rs + 2s^2$

Q

A	C	
a	e	0
d	e	0
f	e	$2s^2$

Delete `d b e` from R ?

Set $r = 0$!

Low-hanging fruit: deletion propagation

We used this in **Orchestra** [VLDB07]
for update propagation

R

A	B	C	
a	b	c	p
d	b	e	r
f	g	e	s

Q

A	C	
a	e	pr
d	e	$2r^2 + rs$
f	e	$rs + 2s^2$

Q

A	C	
a	e	0
d	e	0
f	e	$2s^2$

Q

A	C	
f	e	$2s^2$

Delete **d b e** from R ?

Set $r = 0$!

But are there useful commutative semirings?

$(\mathbb{B}, \wedge, \vee, \top, \bot)$	**Set semantics**
$(\mathbb{N}, +, \cdot, 0, 1)$	**Bag semantics**
$(\mathcal{P}(\Omega), \cup, \cap, \emptyset, \Omega)$	Probabilistic events [FuhrRölleke 97]
(BoolExp(*X*), $\wedge, \vee, \top, \bot$)	Conditional tables (*c*-tables) [ImielinskiLipski 84]
$(\mathbb{R}_+^\infty, \min, +, 1, 0)$	Tropical semiring (cost/distrust score/confidence need)
($\mathbb{A}$, min, max, 0, P) where $\mathbb{A}$ = P < C < S < T < 0	Access control levels [PODS8]

But are there useful commutative semirings?

$(B, \wedge, \vee, \top, \bot)$	**Set semantics**
$(\mathbb{N}, +, \cdot, 0, 1)$	**Bag semantics**
$(P(\Omega), \cup, \cap, \varnothing, \Omega)$	Probabilistic events [FuhrRölleke 97]
(BoolExp(*X*), $\wedge, \vee, \top, \bot$)	Conditional tables (*c*-tables) [ImielinskiLipski 84]
$(R_+^\infty, \min, +, 1, 0)$	Tropical semiring (cost/distrust score/confidence need)
(A, min, max, 0, P) where $A = P < C < S < T < 0$	Access control levels [PODS8]

public

But are there useful commutative semirings?

$(\mathbb{B}, \wedge, \vee, \top, \bot)$	**Set semantics**
$(\mathbb{N}, +, \cdot, 0, 1)$	**Bag semantics**
$(P(\Omega), \cup, \cap, \emptyset, \Omega)$	Probabilistic events [FuhrRölleke 97]
$(BoolExp(X), \wedge, \vee, \top, \bot)$	Conditional tables (*c*-tables) [ImielinskiLipski 84]
$(\mathbb{R}_+^\infty, \min, +, 1, 0)$	Tropical semiring (cost/distrust score/confidence need)
$(\mathbb{A}, \min, \max, 0, P)$ where $\mathbb{A} = P < C < S < T < 0$	Access control levels [PODS8]

top secret

public

Outline

- What's with the semirings? Annotation propagation

- **Housekeeping in the zoo of provenance models**
  [GK&T PODS 07, FG&T PODS 08, Green ICDT 09]

Semirings for various models of provenance (1)

R

A	B	C	
a	b	c	*p*
d	b	e	*r*
f	g	e	*s*

Q

A	C	
…		
d	e	$\{r, s\}$
…		

Lineage [CuiWidomWiener 00 etc.]

Sets of contributing tuples

Semiring: $(\mathrm{Lin}(X), \cup, \cup^*, \varnothing, \varnothing^*)$

Semirings for various models of provenance (2)

R

A	B	C	
a	b	c	*p*
d	b	e	*r*
f	g	e	*s*

Q

A	C	
…		
d	e	$\{\{r\}, \{r, s\}\}$
…		

(Witness, Proof) **why-provenance**
[BunemanKhannaTan 01] & [Buneman+ PODS08]

Sets of witnesses (w. =set of contributing tuples)

Semiring: $(Why(X), \cup, \Cup, \varnothing, \{\varnothing\})$

Semirings for various models of provenance (3)

R

A	B	C	
a	b	c	*p*
d	b	e	*r*
f	g	e	*s*

Q

A	C	
…		
d	e	{{*r*}}
…		

Minimal witness **why-provenance**
[BunemanKhannaTan 01]

Sets of minimal witnesses

Semiring: (PosBool(X), $\wedge$, $\vee$, $\top$, $\bot$)

Semirings for various models of provenance (4)

R

A	B	C	
a	b	c	*p*
d	b	e	*r*
f	g	e	*s*

Q

A	C	
...		
d	e	[{*r*}, {*r*}, {*r*, *s*}]
...		

Notation:
{ } set
[ ] bag

Trio lineage [Das Sarma+ 08]

Bags of sets of contributing tuples (of witnesses)

Semiring: (Trio(X), +, $\cdot$, 0, 1) (defined in [Green, ICDT 09])

Semirings for various models of provenance (5)

R

A	B	C	
a	b	c	p
d	b	e	r
f	g	e	s

Q

A	C	
…		
d	e	$\{[r,r], [r, s]\}$
…		

Notation:

{ } set

[ ] bag

Polynomials with boolean coefficients [Green, ICDT 09]

(B[X]-provenance)

Sets of bags of contributing tuples

Semiring: (B[X], +, ·, 0, 1)

Semirings for various models of provenance (6)

R

A	B	C	
a	b	c	*p*
d	b	e	*r*
f	g	e	*s*

Q

A	C	
…		
d	e	[[*r*,*r*], [*r*,*r*], [*r*, *s*]]
…		

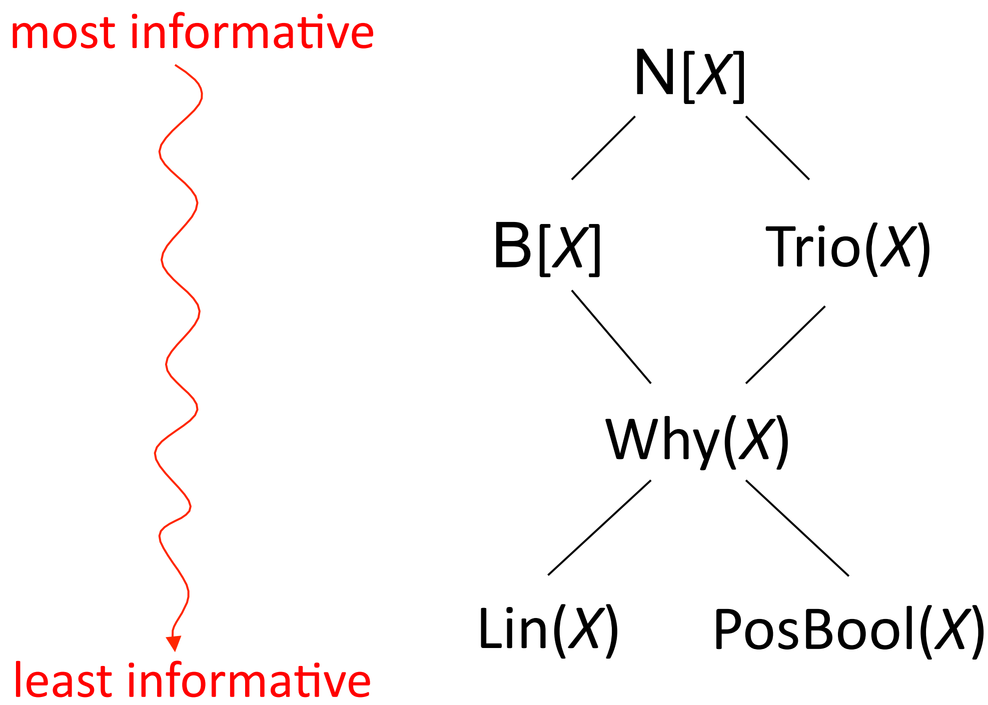
Provenance polynomials [GKT, PODS 07]

(N[*X*]-provenance)

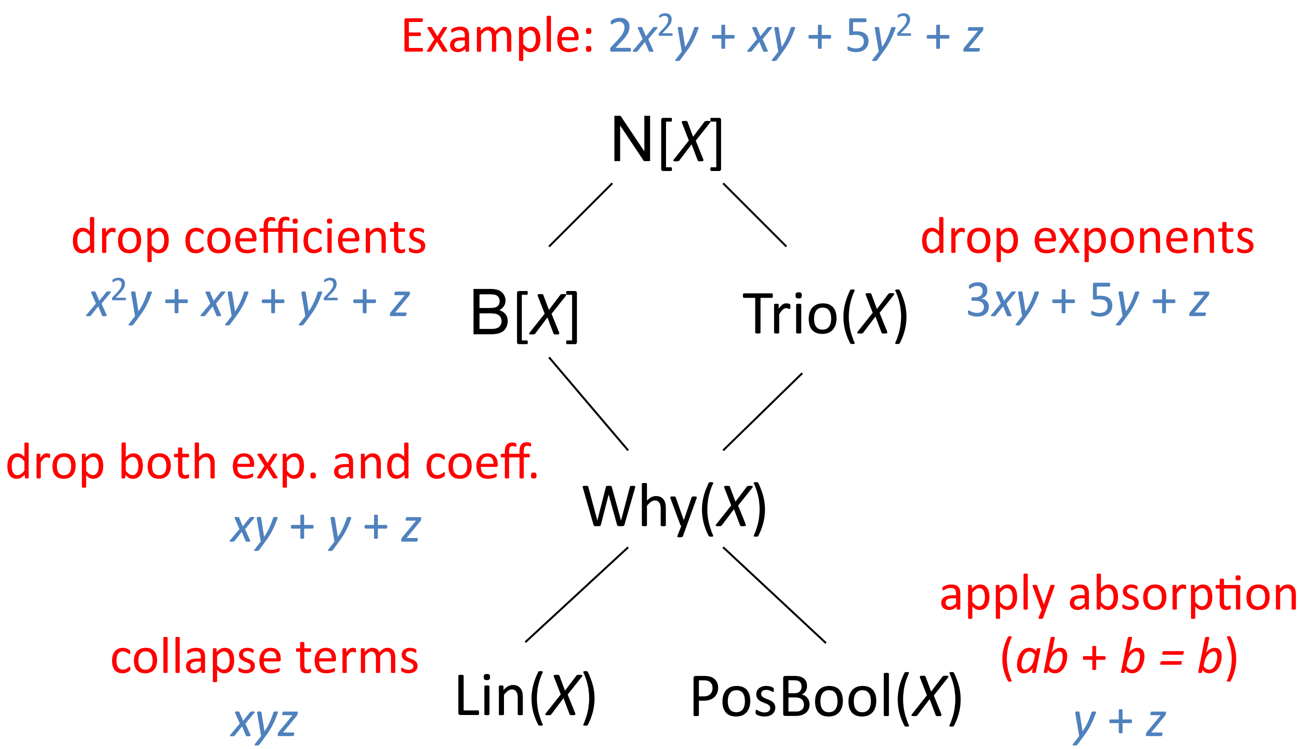
Bags of bags of contributing tuples

Semiring: (N[*X*], +, $\cdot$, 0, 1)

A provenance hierarchy

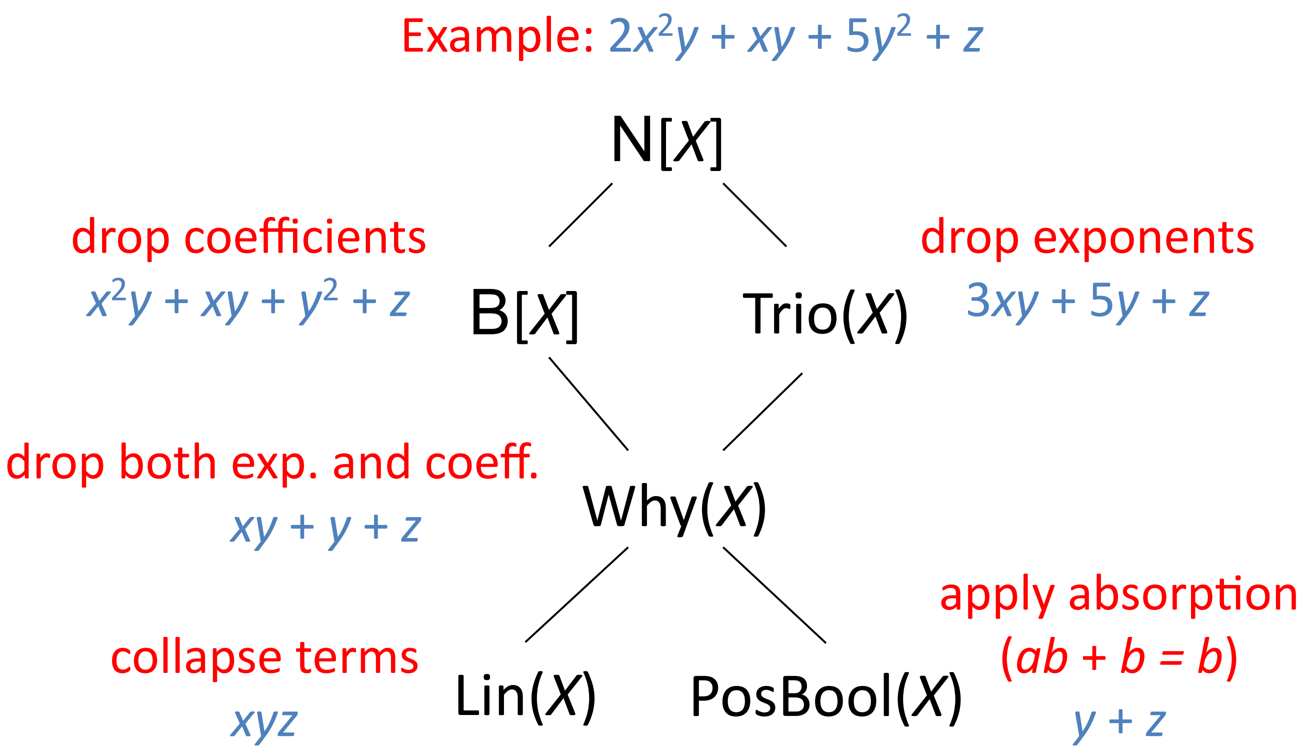


One semiring to rule them all… (apologies!)



A path downward from K_1 to K_2 indicates that there exists an **onto (surjective) semiring homomorphism** $h : K_1 \rightarrow K_2$

Using homomorphisms to relate models

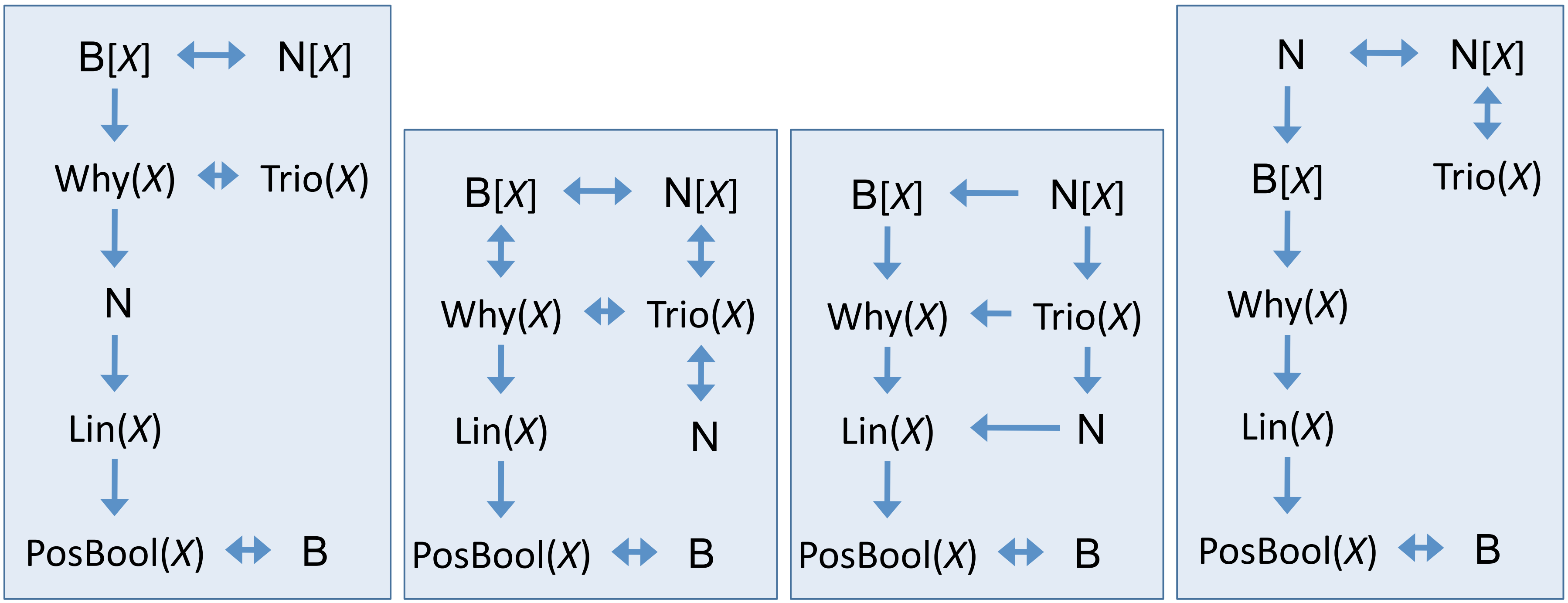


Homomorphism?

$h(x+y) = h(x)+h(y)$ $h(xy)=h(x)h(y)$ $h(0)=0$ $h(1)=1$

Moreover, for these homomorphisms $h(x)= x$

Containment and Equivalence [Green ICDT 09]



SPJ containment | SPJ equivalence | SPJU containment | SPJU equivalence

Arrow from K_1 to K_2 indicates K_1 containment (equivalence) implies K_2 cont. (equiv.)

All implications not marked ⟷ are strict

Data Security

- Based on my colloquium talk from 2005

Data Security

Dorothy Denning, 1982:

- Data Security is the science and study of methods of protecting data (...) from unauthorized disclosure and modification

- Data Security = Confidentiality + Integrity

Data Security

- Distinct from *systems* and *network* security
  - Assumes these are already secure

- Tools:
  - Cryptography, information theory, statistics, …

- Applications:
  - An *enabling* technology

Outline

- An attack
- Data security research today

Latanya Sweeney's Finding

- In Massachusetts, the Group Insurance Commission (GIC) is responsible for purchasing health insurance for state employees
- GIC has to publish the data:

GIC(**zip, dob, sex**, diagnosis, procedure, ...)

This is private ! Right ?

Latanya Sweeney's Finding

- Sweeney paid $20 and bought the voter registration list for Cambridge Massachusetts:

VOTER(name, party, ..., **zip, dob, sex**)

GIC(**zip, dob, sex**, diagnosis, procedure, ...)

This is private ! Right ?

Latanya Sweeney's Finding

zip, dob, sex

- William Weld (former governor) lives in Cambridge, hence is in VOTER
- 6 people in VOTER share his **dob**
- only 3 of them were man (same **sex**)
- Weld was the only one in that **zip**
- Sweeney learned Weld's medical records !

Latanya Sweeney's Finding

- All systems worked as specified, yet an important data has leaked

- How do we protect against that ?

Some of today's research in data security address breaches that happen even if all systems work correctly

Today’s Approaches

- K-anonymity
  - Useful, but not really private
- Differential privacy
  - Private, but not really useful

[Samarati&Sweeney'98, Meyerson&Williams'04]

k-Anonymity

Definition: each tuple is equal to at least k-1 others

Anonymizing: through suppression and generalization

First	Last	Age	Race
Harry	Stone	34	Afr-Am
John	Reyser	36	Cauc
Beatrice	Stone	47	Afr-am
John	Ramos	22	Hisp

Hard: NP-complete for supression only
Approximations exists

[Samarati&Sweeney'98, Meyerson&Williams'04]

k-Anonymity

<u>Definition</u>: each tuple is equal to at least k-1 others

Anonymizing: through suppression and generalization

First	Last	Age	Race
*	Stone	30-50	Afr-Am
John	R*	20-40	*
*	Stone	30-50	Afr-am
John	R*	20-40	*

Hard: NP-complete for supression only
Approximations exists

[Dwork'05]

Differential Privacy

- A randomized algorithm A is differentially private if by removing/inserting one tuple in the database, the output of A is "almost the same", i.e. every possible outcome for A has almost the same probability

[Dwork’05]

Differential Privacy

- How can we achieve that ? Add some random noise to the result of A
- For example:
  - Query: select count(*) from R where blah
  - Add some random noise (Laplacian distribution: $e^{-x/x0}$)

- Problem: can only ask a limited number of queries
  - Must keep track of the queries answered, then deny
  - Cannot release “the entire data”

Privacy

- All these techniques address *confidentiality*, but they are often claim *privacy*

- Privacy is more complex:
  - "Is the right of individuals to determine for themselves when, how and to what extent information about them is communicated to others"

[Agrawal'03]

Take Home Lessons

- Data management does not stop at normal forms and query optimization
- Our field (Computer Science) is becoming data-centric. Dominated by massive amounts of data.
- This affects businesses, science, society
- Watch the data management & data mining fields for excitement future innovations