Probabilistic/Uncertain Data Management -- III

1. *Dalvi, Suciu. "Efficient query evaluation on probabilistic databases", VLDB'2004.*
2. *Das Sarma et al. "Working models for uncertain data", ICDE'2006.*

Slides based on the Suciu/Dalvi SIGMOD'05 tutorial

What is a Probabilistic Database ?

- “An item belongs to the database” is a probabilistic event
  - Tuple-existence uncertainty
  - Attribute-value uncertainty
- “A tuple is an answer to the query” is a probabilistic event
- Can be extended to all data models; we discuss only probabilistic *relational* data

Possible Worlds Semantics

The set of all possible database instances:

$$\boxed{\text{INST} = \{I_1, I_2, I_3, \ldots, I_N\}}$$

Definition A *probabilistic database* I^p is a probability distribution on INST

$$\boxed{\Pr : \text{INST} \rightarrow [0,1]} \quad \text{s.t. } \sum_{i=1,N} \Pr(I_i) = 1$$

Definition A *possible world* is I s.t. Pr(I) > 0

Query Semantics

Given a query Q and a probabilistic database I^p, what is the meaning of $Q(I^p)$?

Query Semantics

Semantics 1: Possible Answers

A probability distribution on *sets of tuples*

$$\forall\ A.\ \Pr(Q = A) = \sum_{I\ \in\ INST.\ Q(I) = A} \Pr(I)$$

Semantics 2: Possible Tuples

A probability function on *tuples*

$$\forall\ t.\ \Pr(t \in Q) = \sum_{I\ \in\ INST.\ t \in Q(I)} \Pr(I)$$

Possible Worlds Query Semantics

Possible answers semantics

- Precise
- Can be used to compose queries
- Difficult user interface

Possible tuples semantics

- Less precise, but simple; sufficient for most apps
- Cannot be used to compose queries
- Simple user interface

Possible Worlds Semantics: Summary

Complete model; Clean formal semantics for SQL queries

Not very useful as a representation or implementation tool

- HUGE number of possible worlds!

Need more effective representation formalisms

- Something that users can understand/explore
- Allow more efficient query execution
  - Avoid "possible worlds explosion"
- *Perhaps giving up completeness*

Representation Formalisms

Problem

Need a good representation formalism

- Will be interpreted as possible worlds
- Several formalisms exists, but no winner

Main open problem in probabilistic db

Evaluation of Formalisms

Completeness?

- What possible worlds can it represent?
- What probability distributions on worlds?

Closure?

- Is it closed under evaluation of query operators?

A Complete Formalism: Intensional Databases

Atomic event ids $e_1, e_2, e_3, \ldots$

Probabilities: $p_1, p_2, p_3, \ldots \in [0,1]$

Event expressions: $\wedge, \vee, \neg$ $e_3 \wedge (e_5 \vee \neg e_2)$

Intensional probabilistic database J:
each tuple t has an event attribute t.E

Intensional DB $\Rightarrow$ Possible Worlds

J =

Name	**Address**	**E**
John	Seattle	$e_1 \wedge (e_2 \vee e_3)$
Sue	Denver	$(e_1 \wedge e_2) \vee (e_2 \wedge e_3)$

$e_1e_2e_3=$ 000 001 010 011 100 101 110 111

I^p

$\emptyset$

$(1-p_1)(1-p_2)(1-p_3)$
$+(1-p_1)(1-p_2)p_3$
$+(1-p_1)p_2(1-p_3)$
$+p_1(1-p_2)(1-p_3$

John	Seattle

$p_1(1-p_2)\ p_3$

Sue	Denver

$(1-p_1)p_2\ p_3$

John	Seattle
Sue	Denver

$p_1p_2(1-p_3)$
$+p_1p_2p_3$

Possible Worlds $\Rightarrow$ Intensional DB

Name	Address
John	Seattle
John	Boston
Sue	Seattle

p_1

Name	Address
John	Seattle
Sue	Seattle

p_2

Name	Address
Sue	Seattle

p_3

Name	Address
John	Boston

p_4

$E_1 = e_1$ $\quad$ $Pr(e_1) = p_1$

$E_2 = \neg e_1 \wedge e_2$ $\quad$ $Pr(e_2) = p_2/(1-p_1)$

$E_3 = \neg e_1 \wedge \neg e_2 \wedge e_3$ $\quad$ $Pr(e_3) = p_3/(1-p_1-p_2)$

$E_4 = \neg e_1 \wedge \neg e_2 \wedge \neg e_3 \wedge e_4$ $\quad$ $Pr(e_4) = p_4/(1-p_1-p_2-p_3)$

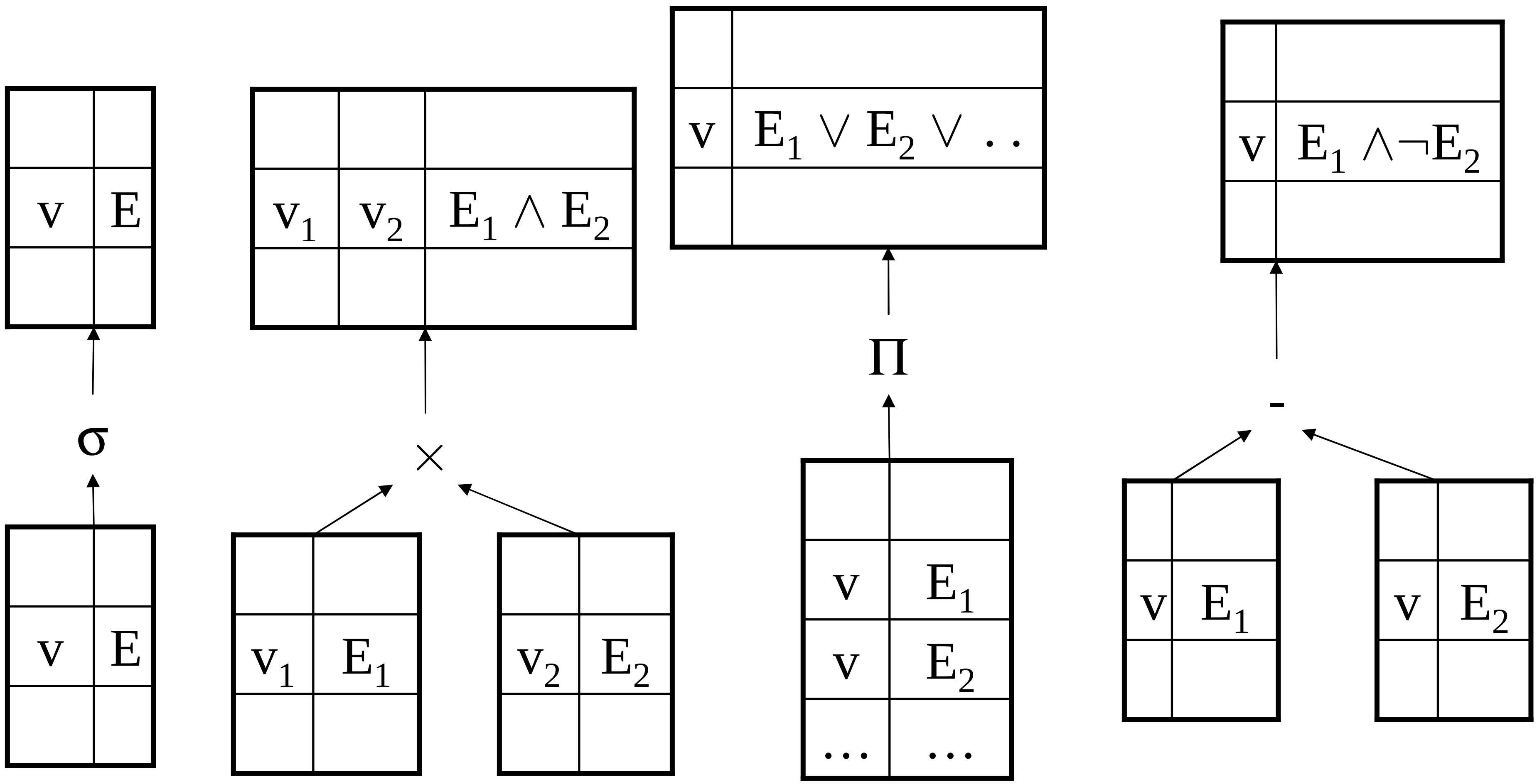
"Prefix code"

$=I^p$ $\Rightarrow$ $J =$

Name	Address	E
John	Seattle	$E_1 \vee E_2$
John	Boston	$E_1 \vee E_4$
Sue	Seattle	$E_1 \vee E_2 \vee E_3$

Intensional DBs are complete

Closure Under Operators



One still needs to compute probability of event expression

Summary on Intensional Databases

Event expression for each tuple

- Possible worlds: any subset
- Probability distribution: any

Complete… but impractical

- Evaluate the probability of *long* event expressions

Important abstraction: consider restrictions

Related to *c-tables* [Imilelinski&Lipski:1984]

A Restricted Formalism: Explicit Independent Tuples

Tuple independent probabilistic database

INST = $\mathcal{P}$(TUP)
N = 2^M

$\boxed{\text{TUP} = \{t_1, t_2, \ldots, t_M\}}$ = all tuples

$\boxed{pr : \text{TUP} \rightarrow [0,1]}$ No restrictions

$$\boxed{\Pr(I) = \prod_{t \in I} pr(t) \times \prod_{t \notin I} (1-pr(t))}$$

Tuple Prob. ⇒ Possible Worlds

J =

Name	**City**	**pr**
John	Seattle	$p_1 = 0.8$
Sue	Boston	$p_2 = 0.6$
Fred	Boston	$p_3 = 0.9$

E[size(I^p)] = 2.3 tuples

I^p =

I_1: $\emptyset$

$(1-p_1)(1-p_2)(1-p_3)$

I_2:

Name	**City**
John	Seattl

$p_1(1-p_2)(1-p_3)$

I_3:

Name	**City**
Sue	Bosto

$(1-p_1)p_2(1-p_3)$

I_4:

Name	**City**
Fred	Bosto

$(1-p_1)(1-p_2)p_3$

I_5:

Name	**City**
John	Seattl
Sue	Bosto

$p_1p_2(1-p_3)$

I_6:

Name	**City**
John	Seattl
Fred	Bosto

$p_1(1-p_2)p_3$

I_7:

Name	**City**
Sue	Bosto
Fred	Bosto

$(1-p_1)p_2p_3$

I_8:

Name	**City**
John	Seattl
Sue	Bosto
Fred	Bosto

$p_1p_2p_3$

$\sum = 1$

Tuple-Independent DBs are Incomplete

Name	Address
John	Seattle

p_1

Name	Address
John	Seattle
Sue	Seattle

$p_1 p_2$

$\emptyset$ $1-p_1 - p_1 p_2$

$= I^p$

~~Name~~	~~Address~~	~~pr~~
~~John~~	~~Seattle~~	~~p_1~~
~~Sue~~	~~Seattle~~	~~p_2~~

Very limited – cannot capture correlations across tuples

Not Closed

- Query operators can introduce complex correlations!

Tuple Prob. $\Rightarrow$ Query Evaluation

Name	**City**	**pr**
John	Seattle	p_1
Sue	Boston	p_2
Fred	Boston	p_3

Customer	**Product**	**Date**	**pr**
John	Gizmo	. . .	q_1
John	Gadget	. . .	q_2
John	Gadget	. . .	q_3
Sue	Camera	. . .	q_4
Sue	Gadget	. . .	q_5
Sue	Gadget	. . .	q_6
Fred	Gadget	. . .	q_7

```
SELECT DISTINCT x.city
FROM Person x, Purchase y
WHERE x.Name = y.Customer
      and y.Product = ‘Gadget’
```

Tuple	**Probability**
Seattle	$p_1(1-(1-q_2)(1-q_3))$
Boston	$1-(1-p_2(1-(1-q_5)(1-q_6)))$ $\times(1-p_3\, q_7)$

Application: Similarity Predicates

Name	City	Profession
John	Seattle	statistician
Sue	Boston	musician
Fred	Boston	physicist

Step 1:
evaluate ~ predicates

Cust	Product	Category
John	Gizmo	dishware
John	Gadget	instrument
John	Gadget	instrument
Sue	Camera	musicware
Sue	Gadget	microphone
Sue	Gadget	instrument
Fred	Gadget	microphone

SELECT DISTINCT x.city
FROM Person x, Purchase y
WHERE x.Name = y.Cust
    and y.Product = ‘Gadget’
    and **x.profession ~ ‘scientist’**
    and **y.category ~ ‘music’**

Application: Similarity Predicates

Name	City	Profession	pr
John	Seattle	statistician	p_1=0.8
Sue	Boston	musician	p_2=0.2
Fred	Boston	physicist	p_3=0.9

Step 1:
evaluate ~ predicates

Cust	Product	Category	pr
John	Gizmo	dishware	q_1=0.2
John	Gadget	instrument	q_2=0.6
John	Gadget	instrument	q_3=0.6
Sue	Camera	musicware	q_4=0.9
Sue	Gadget	microphone	q_5=0.7
Sue	Gadget	instrument	q_6=0.6
Fred	Gadget	microphone	q_7=0.7

```
SELECT DISTINCT x.city
FROM Person^p x, Purchase^p y
WHERE x.Name = y.Cust
      and y.Product = 'Gadget'
      and x.profession ~ 'scientis
      and y.category ~ 'music'
```

Step 2:
evaluate rest
of query

Tuple	Probability
Seattle	$p_1(1-(1-q_2)(1-q_3))$
Boston	$1-(1-p_2(1-(1-q_5)(1-q_6)))$ $\times(1-p_3q_7)$

Summary on Explicit Independent Tuples

Independent tuples

- Possible worlds: subsets
- Probability distribution: restricted
- Closure: no

Query Evaluation on Probabilistic DBs

- Focus on possible tuple semantics
  - Compute likelihood of individual answer tuples
- Probability of Boolean expressions
- Complexity of query evaluation

Needed for query processing

Probability of Boolean Expressions

$$E = X_1X_3 \vee X_1X_4 \vee X_2X_5 \vee X_2X_6$$

Randomly make each variable **true** with the following probabilities

$\Pr(X_1) = p_1$, $\Pr(X_2) = p_2$, ... , $\Pr(X_6) = p_6$

What is Pr(E) ???

Answer: re-group cleverly

$$E = X_1 (X_3 \vee X_4) \vee X_2 (X_5 \vee X_6)$$

$$\Pr(E)=1 - (1-p_1(1-(1-p_3)(1-p_4))) (1-p_2(1-(1-p_5)(1-p_6)))$$

Now let's try this:

$$E \;=\; X_1X_2 \vee X_1X_3 \vee X_2X_3$$

No clever grouping seems possible.
Brute force:

X_1	X_2	X_3	E	Pr
0	0	0	0	
0	0	1	0	
0	1	0	0	
0	1	1	1	$(1-p_1)p_2p_3$
1	0	0	0	
1	0	1	1	$p_1(1-p_2)p_3$
1	1	0	1	$p_1p_2(1-p_3)$
1	1	1	1	$p_1p_2p_3$

$$\begin{aligned}\Pr(E)=(1-p_1)p_2p_3 &+ \\ p_1(1-p_2)p_3 &+ \\ p_1p_2(1-p_3) &+ \\ p_1p_2p_3 &\end{aligned}$$

Seems inefficient in general…

[Valiant:1979]

Complexity of Boolean Expression Probability

Theorem [Valiant:1979]
For a boolean expression E, computing Pr(E) is #P-complete

NP = class of problems of the form "is there a witness ?" SAT
#P = class of problems of the form "how many witnesses ?" #SAT

The decision problem for 2CNF is in PTIME
The counting problem for 2CNF is #P-complete

Summary on Boolean Expression Probability

- #P-complete
- It's hard even in simple cases: 2DNF
- Can approximate through Monte Carlo (MC) simulation

Query Complexity

Data complexity of a query Q:

- Compute $Q(I^p)$, for probabilistic database I^p

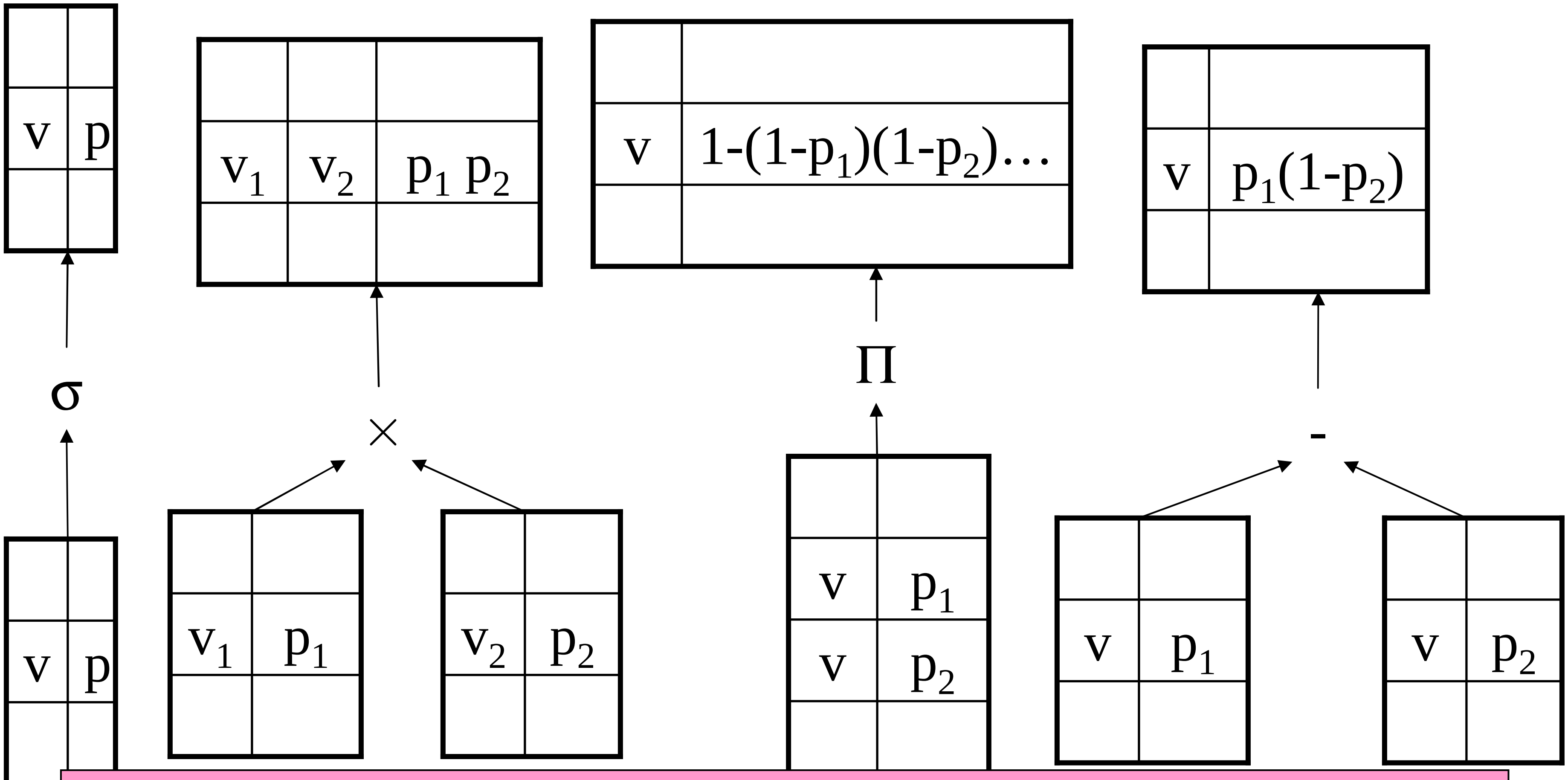
Simplest scenario only:

- Possible tuples semantics for Q
- Independent tuples for I^p

[Fuhr&Roellke:1997,Dalvi&Suciu:2004]

Extensional Query Evaluation

Relational ops compute probabilities



Unlike intensional evaluation, data complexity: PTIME

[Dalvi&Suciu:2004]

SELECT DISTINCT x.City
FROM Person^p x, Purchase^p y
WHERE x.Name = y.Cust
      and y.Product = 'Gadget'

Wrong !

Sea	$1-(1-p_1q_1)(1- p_1q_2)(1- p_1q_3)$

Π

Jon	Sea	p_1q_1
Jon	Sea	p_1q_2
Jon	Sea	p_1q_3

$\times$

Jon	Sea	p_1

Jon	q_1
Jon	q_2
Jon	q_3

Correct

Jon	Sea	$p_1(1-(1-q_1)(1-q_2)(1-q_3))$

$\times$

Jon	Sea	p_1

Jon	$1-(1-q_1)(1-q_2)(1-q_3)$

Π

Jon	q_1
Jon	q_2
Jon	q_3

Depends on plan !!!

[Dalvi&Suciu:2004]

Query Complexity

Sometimes $\nexists$ correct ("safe") extensional plan

Q_{bad} :- R(x), S(x,y), T(y)

Data complexity is #P complete

Theorem The following are equivalent

- Q has PTIME data complexity
- Q admits an extensional plan (and one finds it in PTIME)
- Q does not have Q_{bad} as a subquery

Computing a Safe SPJ Extensional Plan

Problem is due to projection operations

- An "unsafe" extensional projection combines tuples that are correlated assuming independence

Projection over a join that projects away at least one of the join attrs ➔ Unsafe projection!

- *Intuitive:* Joins create correlated output tuples

Computing a Safe SPJ Extensional Plan

Algorithm for Safe Extensional SPJ Evaluation

- Apply safe projections as late as possible in the plan
- If no more safe projections exist, look for joins where all attributes are included in the output
  - Recurse on the LHS, RHS of the join

Sound and complete safe SPJ evaluation algorithm

- *If a safe plan exists*, the algo finds it!

Summary on Query Complexity

Extensional query evaluation:

- Very popular
- Guarantees polynomial complexity
- However, <u>result depends on query plan and correctness not always possible!</u>

General query complexity

- #P complete (not surprising, given #SAT)
- Already #P hard for very simple query (Q_{bad})

Probabilistic databases have high query complexity