Probabilistic/Uncertain Data Management -- IV

1. *Dalvi, Suciu. "Efficient query evaluation on probabilistic databases", VLDB'2004.*
2. *Sen, Deshpande. "Representing and Querying Correlated Tuples in Probabilistic DBs", ICDE'2007.*

A Restricted Formalism: Explicit Independent Tuples

Tuple independent probabilistic database

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

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

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

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

Tuple Prob. $\Rightarrow$ 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

$p_1 p_2$

$\emptyset$ $1\text{-}p_1 - p_1 p_2$

$= I^p$

~~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!

Query Evaluation on Probabilistic DBs

- Focus on possible tuple semantics
  - Compute likelihood of individual answer tuples
- Probability of Boolean expressions
  - Key operation for *Intensional Query Evaluation*
- Complexity of query evaluation

[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

Query Complexity

Data complexity of a query Q:

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

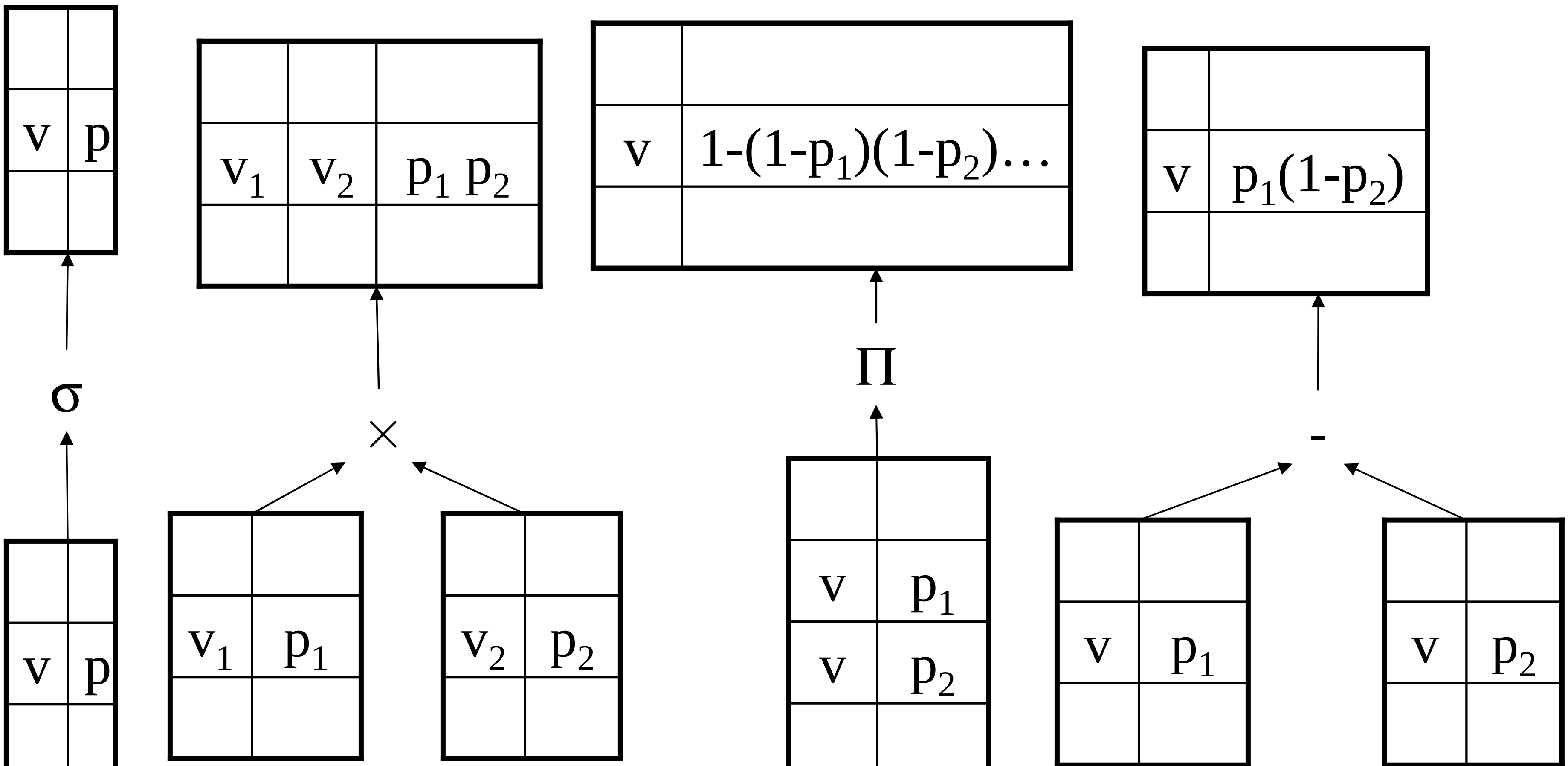
Simplest scenario only:

- 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

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

$\times$

Correct

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

Π

Jon	Sea	p_1

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

Efficient Approximate Evaluation: Monte-Carlo Simulation

Run evaluation with no projection/dup elimination till the very final step

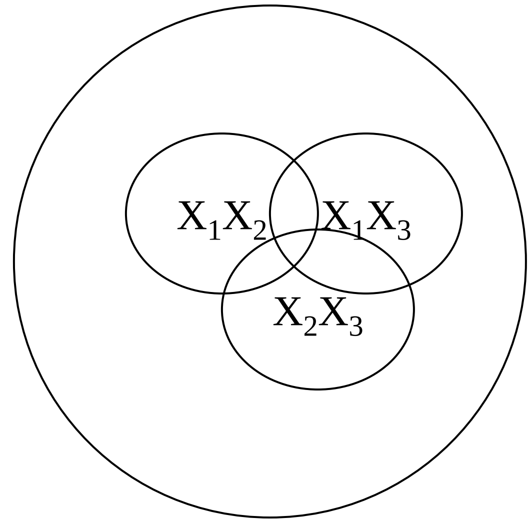
- Intermediate tuples carry all attributes
- Each result tuple = *group* $t_1, \ldots, t_n$ of tuples with the same projection attribute values
  - Prob(group) = Prob(C_1 OR C_2 OR… C_n) , where each $C_i = e_1$ AND e_2 … AND e_k
  - Evaluate the probability of a *large* DNF expression
  - Can be efficiently approximated through MC simulation (a.k.a. sampling)

[Karp,Luby&Madras:1989]

Monte Carlo Simulation

Naïve:

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



Cnt $\leftarrow$ 0
repeat N times
    randomly choose $X_1, X_2, X_3 \in \{0,1\}$
    if $E(X_1, X_2, X_3) = 1$
        then Cnt = Cnt+1
P = Cnt/N
return P /* $\simeq$ Pr(E) */

May be very big

0/1-estimator theorem

Theorem. If $N \geq (1/\Pr(E)) \times (4\ln(2/\delta)/\varepsilon^2)$ then:

$$\Pr[\,|\,P/\Pr(E) - 1\,| > \varepsilon\,] \;<\; \delta$$

Works for any E
Not in PTIME

[Karp,Luby&Madras:1989]

Monte Carlo Simulation

Improved:

$E \;=\; C_1 \vee C_2 \vee \ldots \vee C_m$

Cnt $\leftarrow$ 0;   S $\leftarrow$ Pr(C_1) + … + Pr(C_m);
repeat N times
  randomly choose i $\in$ {1,2,…, m}, with prob. Pr(C_i) / S
  randomly choose $X_1, \ldots, X_n \in \{0,1\}$ s.t. $C_i = 1$
  if C_1=0 and C_2=0 and … and C_{i-1} = 0
    then Cnt = Cnt+1
P = Cnt/N * 1/
return P /* $\simeq$ Pr(E) */

Now it's better

Theorem. If $N \geq (1/\ m) \times (4\ln(2/\delta)/\varepsilon^2)$ then:
  $\Pr[\ |\ P/\Pr(E) - 1\ | > \varepsilon\ ] \;<\; \delta$

Only for E in DNF
In PTIME

Summary on Monte Carlo

Some form of simulation is needed in probabilistic databases, to cope with the #P-hardness bottleneck

- Naïve MC: works well when Prob is big
- Improved MC: needed when Prob is small

Recent work *[Re,Dalvi,Suciu, ICDE'07]* describes optimized MC for top-k tuple evaluation

Handling Tuple Correlations

Tuple correlations/dependencies arise naturally

- *Sensor networks:* Temporal/spatial correlations
- During query evaluation (even starting with independent tuples)

Need representation formalism that can capture and evaluate queries over such correlated tuples

Capturing Tuple Correlations: Basic Ideas

Use key ideas of Probabilistic Graphical Models (PGMs)

- Bayes and Markov networks are special cases

Tuple-based random variables

- Each tuple t corresponds to a Boolean RV X_t

Factors capturing correlations across subsets of RVs

- f(**X**) is a function of a (small) subset **X** of the X_t RVs

[Sen, Deshpande:2007]

Capturing Tuple Correlations: Basic Ideas

Associate each probabilistic tuple with a binomial RV

Define PGM factors capturing correlations across subsets of tuple RVs

Probability of a possible world = product of all PGM factors

- PGM = factored, economical representation of possible worlds distribution
- *Closed & complete* representation formalism

Example: Mutual Exclusion

Want to capture mutual exclusion (XOR) between tuples s_1 and t_1

S

	A	B	*prob*
s_1	m	1	0.6
s_2	n	1	0.5

T

	C	D	*prob*
t_1	1	p	0.4

possible worlds

instance	probability
$\{s_1, s_2, t_1\}$	0
$\{s_1, s_2\}$	0.3
$\{s_1, t_1\}$	0
$\{s_1\}$	0.3
$\{s_2, t_1\}$	0.2
$\{s_2\}$	0
$\{t_1\}$	0.2
$\emptyset$	0

X_{t_1}	X_{s_1}	f_1
0	0	0
0	1	0.6
1	0	0.4
1	1	0

X_{s_2}	f_2
0	0.5
1	0.5

Example: Positive Correlation

Want to capture positive correlation between tuples s_1 and t_1

S

	A	B
s_1	m	1
s_2	n	1

T

	C	D
t_1	1	p

possible worlds

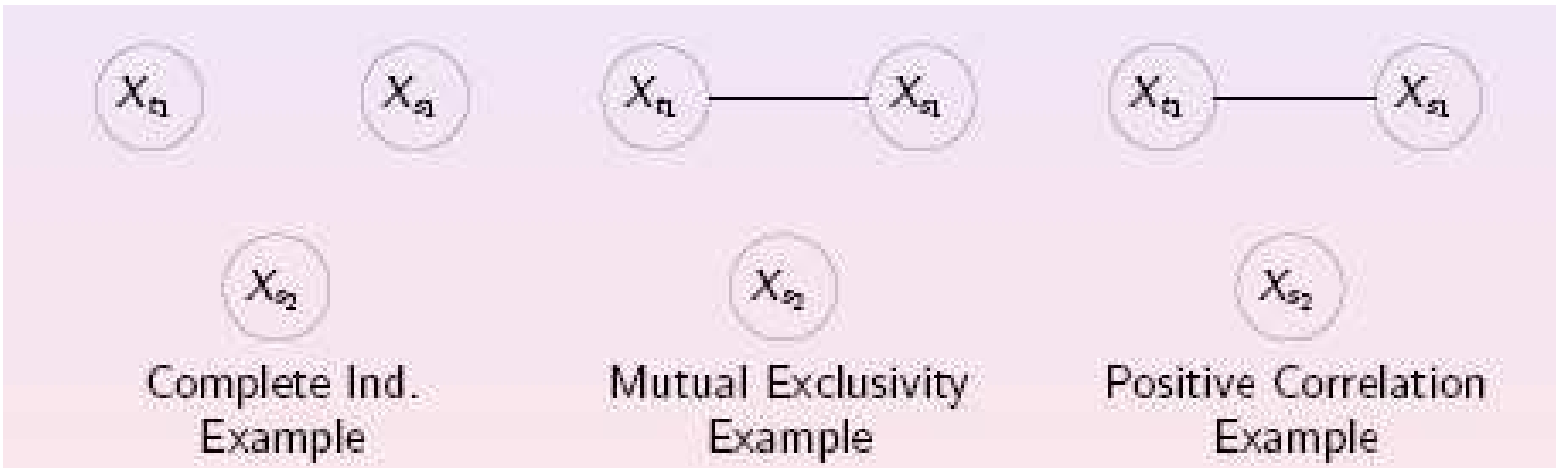
instance	probability
$\{s_1, s_2, t_1\}$	0.2
$\{s_1, s_2\}$	0.1
$\{s_1, t_1\}$	0.2
$\{s_1\}$	0.1
$\{s_2, t_1\}$	0
$\{s_2\}$	0.2
$\{t_1\}$	0
$\emptyset$	0.2

X_{t_1}	X_{s_1}	f_1
0	0	0.4
0	1	0.2
1	0	0
1	1	0.4

X_{s_2}	f_2
0	0.5
1	0.5

PGM Representation

Definition: A PGM is a graph whose nodes represent RVs and edges represent correlations



Factors correspond to the cliques of the PGM graph

– Graph structure encodes conditional independencies

– Joint pdf = Π clique factors

– Economical representation (O(2^k), k=|max clique|)

Query Evaluation: Basic Ideas

Carefully represent correlations between base, intermediate, and result tuples to generate a PGM for the query result distribution

– Each relational op generates *Boolean factors* capturing the dependencies of its input/output tuples

Final model = product of all generated factors

Cast probabilistic computations in query evaluation as a *probabilistic inference* problem over the resulting (factored) PGM

– Can import ML techniques and optimizations

Query Evaluation: Example

Compute $\prod_{D}(S \bowtie_{B=C} T)$

S:

	A	B
s_1	m	1
s_2	n	1

f_{s_1}, f_{s_2}

T:

	C	D
t_1	1	p

f_{t_1}

$\xrightarrow{S \bowtie_{B=C} T}$

$f^{AND}_{i_1, s_1, t_1}, f^{AND}_{i_2, s_2, t_1}$

	A	B	C	D
i_1	m	1	1	p
i_2	n	1	1	p

$\prod_{D}(S \bowtie_{B=C} T)$ $\downarrow$

	D
r_1	p

$f^{OR}_{r_1, i_1, i_2}$

Probabilistic DBs: Summary

- Principled framework for managing uncertainties
  - Uncertainty management: ML, AI, Stats
  - Benefits of DB world: declarative QL, optimization, scale to large data, physical access structs, …

- Prob DBs = "Marriage" of DBs and ML/AI/Stats
  - ML folks have also been moving our way: Relational extensions to ML models (PRMs, FO models, inductive logic programming, …)

Probabilistic DBs: Future

- Importing more sophisticated ML techniques and tools inside the DBMS
  - Inference as queries, FO models and optimizations, access structs for relational queries + inference, …

- More on the algorithmic front: Probabilstic DBs and possible worlds semantics brings new challenges
  - E.g., approximate query processing, probabilistic data streams (e.g., sketching), …