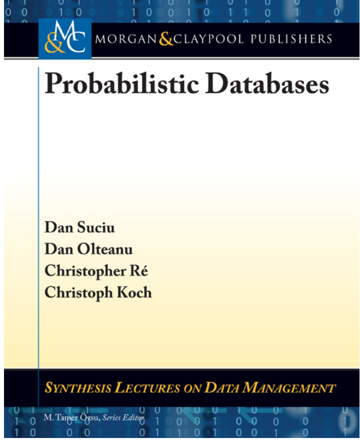
Brief Tutorial on Probabilistic Databases

Dan Suciu

University of Washington



MORGAN & CLAYPOOL PUBLISHERS

Probabilistic Databases

Dan Suciu
Dan Olteanu
Christopher Ré
Christoph Koch

SYNTHESIS LECTURES ON DATA MANAGEMENT

M. Tamer Özsu, Series Editor

About This Talk

- Probabilistic databases
  - – Tuple-independent
  - – Query evaluation
- Statistical relational models
  - – Representation, learning, inference in FO
  - – Reasoning/learning = lifted inference
- Sources:
  - – Book 2011 [S.,Olteanu,Re,Koch]
  - – Upcoming F&T survey [van Den Broek,S]

Background: Relational databases

Database D =

Smoker

X	Y
Alice	2009
Alice	2010
Bob	2009
Carol	2010

Friend

X	Z
Alice	Bob
Alice	Carol
Bob	Carol
Carol	Bob

Query: $Q(z) = \exists x\,(\text{Smoker}(x,\text{'2009'}) \wedge \text{Friend}(x,z))$

Q(D) =

Z
Bob
Carol

Constraint:

$Q = \forall x\,(\text{Smoker}(x,\text{'2010'}) \Rightarrow \text{Friend}(x,\text{'Bob'}))$

Q(D) = true

Probabilistic Database

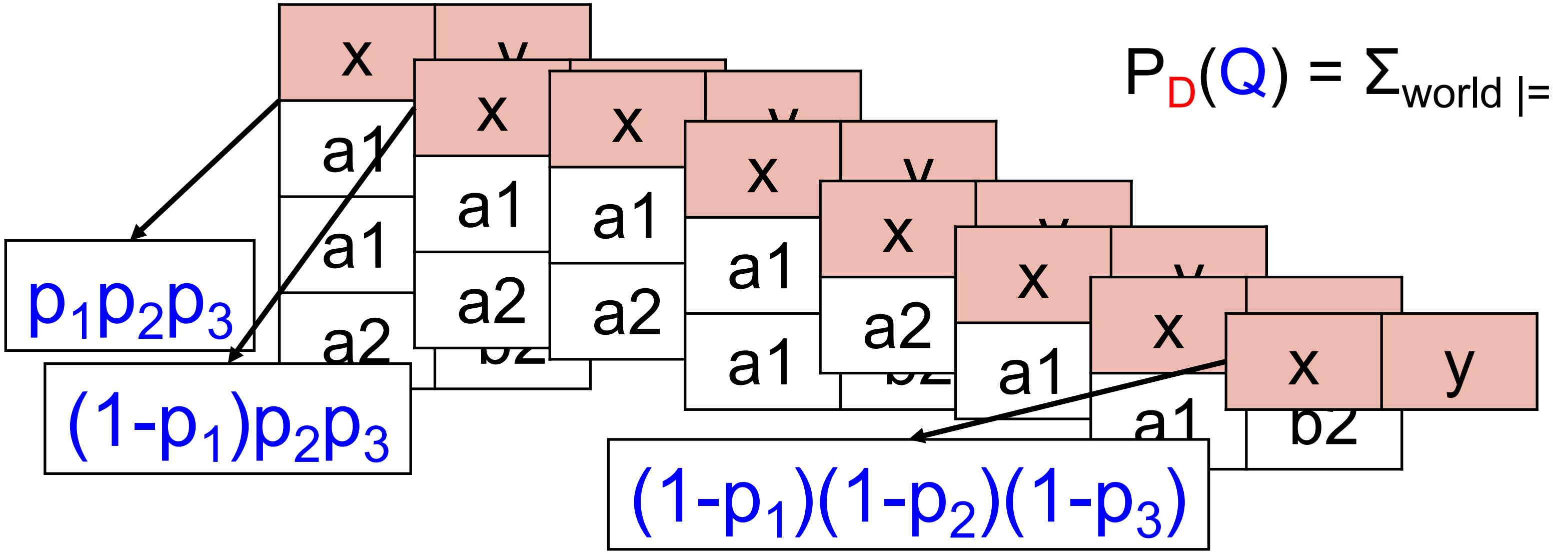
Probabilistic database D:

x	y	P
a1	b1	p_1
a1	b2	p_2
a2	b2	p_3

Possible worlds semantics:

$$\Sigma_{\text{world}}\ P_D(\text{world}) = 1$$

$$P_D(Q) = \Sigma_{\text{world} \models Q}\ P_D(\text{world})$$



$p_1p_2p_3$

$(1-p_1)p_2p_3$

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

Outline

- Model Counting
- Small Dichotomy Theorem
- Dichotomy Theorem
- Query Compilation
- Conclusions, Open Problems

Model Counting

- Given propositional Boolean formula F, compute the number of models #F

Example:

$F = (X1 \vee X2) \wedge (X2 \vee X3) \wedge (X3 \vee X1)$

$\#F = 4$

X1	X2	X3	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	**1**
1	0	0	0
1	0	1	**1**
1	1	0	**1**
1	1	1	**1**

[Valiant'79] #P-hard, even for 2CNF

Probability of a Formula

- Each variable X has a probability p(X);
- P(F) = probability that F=true, when each X is set to true independently

Example:

$F = (X1 \lor X2) \land (X2 \lor X3) \land (X3 \lor X1)$

$$P(F) = (1-p1)*p2*p3 + \\ p1*(1-p2)*p3 + \\ p1*p2*(1-p3) + \\ p1*p2*p3$$

If p(X) = ½ for all X, then P(F) = #F / 2^n

X1	X2	X3	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	**1**
1	0	0	0
1	0	1	**1**
1	1	0	**1**
1	1	1	**1**

Algorithms for Model Counting

[Gomes, Sabharwal, Selman’2009]
Based on full search DPLL:

- Shannon expansion.

  #F = #F[X=0] + #F[X=1]

- Caching.

  Store #F, look it up later

- Components. If Vars(F1) $\cap$ Vars(F2) = $\varnothing$:

  #(F1 $\wedge$ F2) = #F1 * #F2

Relational Representation (1/2)

- Fix an FO sentence Q and a domain Δ
- Ground atom → Boolean variable

Definition The lineage $F_{Q,\Delta}$ is:

$F_{Q,\Delta} = Q$ if Q = ground atom

$F_{Q1 \wedge Q2,\Delta} = F_{Q1,\Delta} \wedge F_{Q2,\Delta}$ same for $\vee$, $\rightarrow$, $\neg$

$F_{\forall x.Q,\Delta} = \bigwedge_{a \in \Delta} F_{Q[a/x],\Delta}$

$F_{\exists x.Q,\Delta} = \bigvee_{a \in \Delta} F_{Q[a/x],\Delta}$

$Q = \forall x\,(\text{Student}(x) \Rightarrow \text{Person}(x))$

$F_{Q,[n]} = (\text{Student}(1) \Rightarrow \text{Person}(1)) \wedge \ldots \wedge (\text{Student}(n) \Rightarrow \text{Person}(n))$

Relational Representation (2/2)

- For a database D, denote

  $$F_{Q,D} = F_{Q,\mathrm{domain}(D)}$$

  where all tuples not in D are set to *false*

- $F_{Q,\Delta}$ or $F_{Q,D}$ is called the *lineage* or the *provenance* or the *grounding* of Q

Weighted FO Model Counting

- Probabilities of ground atoms in D = probabilities of Boolean variables p(X)

- Fix Q. Given D, compute $P(F_{Q,D})$

x	y	P
a1	b1	p_1
a1	b2	p_2
a2	b2	p_3

- Simple fact: $P_D(Q) = P(F_{Q,D})$

This Talk

Fix a query Q:

- What is the complexity of $P_D(Q)$ in the size of D?

- What is the best runtime of a DPLL-based algorithm on $F_{Q,D}$ in the size of D?

Discussion: Correlations

[Domingos&Richardson'06] MLN = popular FO framework for Machine Learning tasks

Lise Getoor's talk today

Smoker(x)$\wedge$Friends(x,y) →Smoker(y), weight = 2.3

Theorem [Jha,S'11] One can construct effectively D s.t. $P_{MLN,\ \Delta}(Q) = P_D(Q \mid \Gamma) = P_D(Q \wedge \Gamma) / P_D(\Gamma)$

Outline

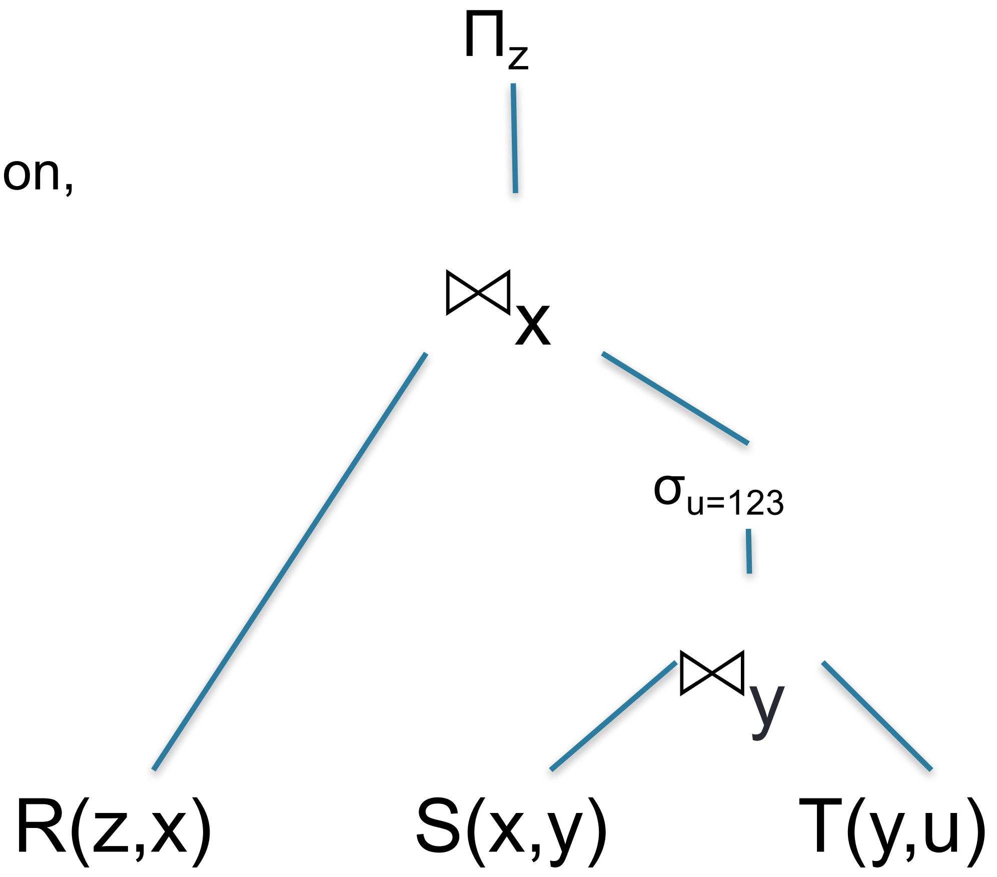
- Model Counting
- Small Dichotomy Theorem
- Dichotomy Theorem
- Query Compilation
- Conclusions, Open Problems

Background: Query Plans

Q(z) = R(z,x), S(x,y),T(y,u), u=123

Query plan = expressions
over the input relation

Operators = selection, projection,
join, union, difference



Π_z

$\bowtie_x$

$\sigma_{u=123}$

$\bowtie_y$

R(z,x)

S(x,y)

T(y,u)

Boolean query

An Example

$Q() = R(x), S(x,y)$ $= \exists x \exists y (R(x) \wedge S(x,y))$

$$P_D(Q) = 1- \{1- p1*[\ 1-(1-q1)*(1-q2)\ ]\} *$$
$$\{1-\ p2*[\ 1-(1-q3)*(1-q4)*(1-q5)]\}$$

One can compute $P_D(Q)$ in PTIME in the size of the database D

R

x	P
a1	p1
a2	p2
a3	p3

S

x	y	P
a1	b1	q1
a1	b2	q2
a2	b3	q3
a2	b4	q4
a2	b5	q5

Extensional Plans

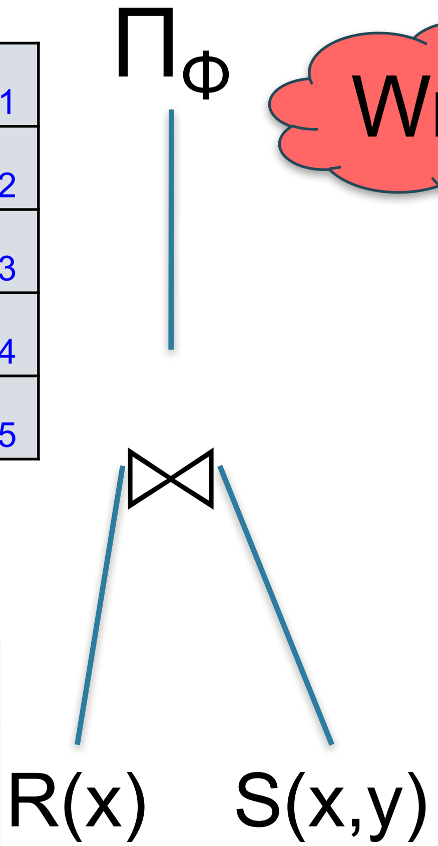
- Modify each operator to compute output probabilities, assuming independent events

$$Q() = R(x), S(x,y)$$

$$P(Q) = 1 – [1-p_1*(1-(1-q_1)*(1-q_2))]$$
$$*[1- p_2*(1-(1-q_3)*(1-q_4)*(1-q_5))]$$

$$1-(1-p_1q_1)(1-p_1q_2)(1-p_2q_3)(1-p_2q_4)(1-p_2q_5)$$

p_1 q_1
p_1 q_2
p_2 q_3
p_2 q_4
p_2 q_5

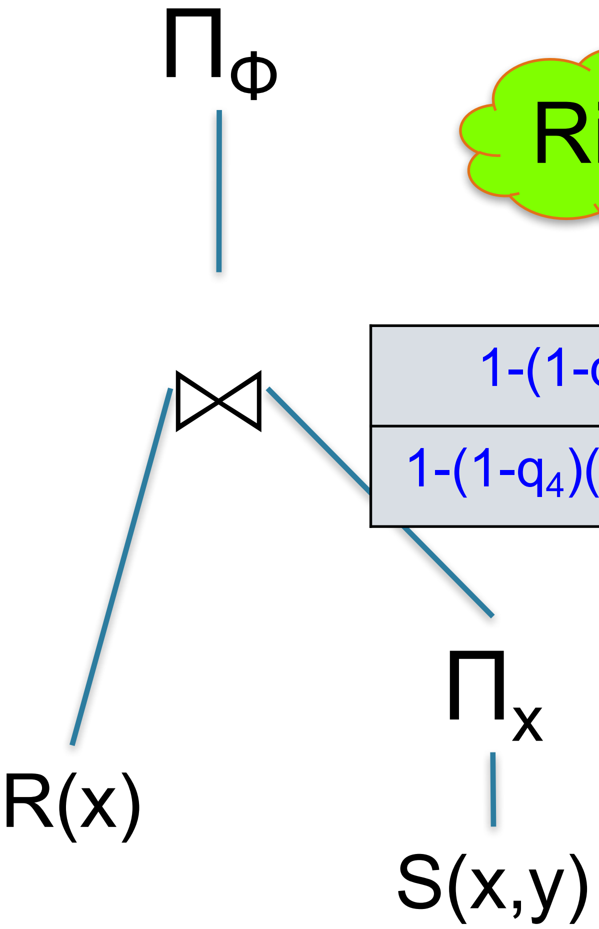


Wrong

x	P
a_1	p_1
a_2	p_2
a_3	p_3

x	y	P
a_1	b_1	q_1
a_1	b_2	q_2
a_2	b_3	q_3
a_2	b_4	q_4
a_2	b_5	q_5

$$1-\{1-p_1[1-(1-q_1)(1-q_2)]\}*$$
$$\{1-p_2[1-(1-q_4)(1-q_5) (1-q_6)]\}$$



Right

$1-(1-q_1)(1-q_2)$
$1-(1-q_4)(1-q_5) (1-q_6)$

Safe Queries

Definition A plan for Q is *safe* if it computes the probabilities correctly.

Q is *safe* if it has a safe plan.

- In AI, computing Q using a safe plan is called *lifted inference*
- *Safe query* = *Liftable query*

- If Q is safe then $P_D(Q)$ is in PTIME

Unsafe Queries

R

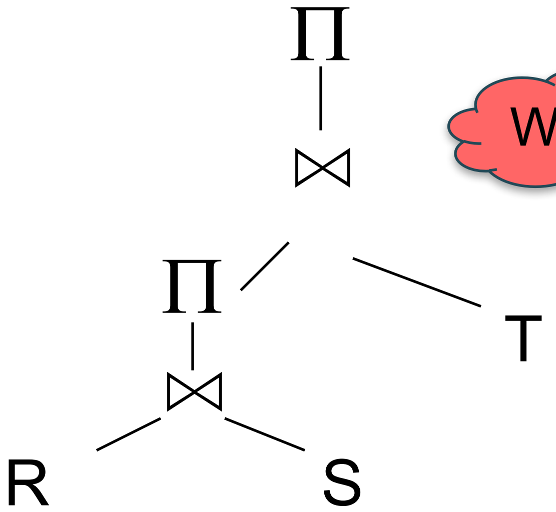
X	P
x1	p1
x2	p2

S

X	Y
x1	y1
x1	y2
x2	y2

T

Y	P
y1	q1
y2	q2



Wrong

$H_0() = R(x), S(x,y), T(y)$

Theorem. [Dalvi&S.2004] $P_D(H_0)$ is #P-hard

However:

1. This plan computes an upper bound [VLDB'15]
2. Use samples on T [VLDB'16]

Wolfgang Gatterbauer's talk today

Hierarchical Queries

Fix Q; at(x) = set of atoms (=literals) containing the variable x

Definition Q is hierarchical if forall variables x, y:

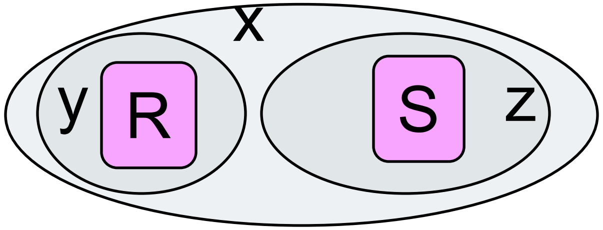
$at(x) \subseteq at(y)$ or $at(x) \supseteq at(y)$ or $at(x) \cap at(y) = \emptyset$

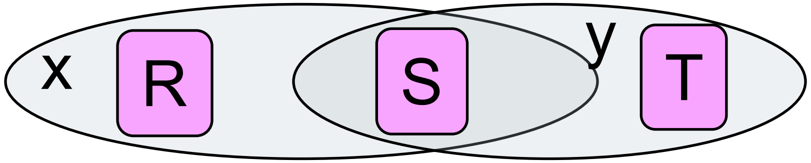
Hierarchical

Q() = R(x,y), S(x,z)

Non-hierarchical

H_0() = R(x), S(x,y), T(y)





The Small Dichotomy Theorem

Non-repeating Conjunctive Query =
= Conjunctive Query “without self-joins”
= “Simple” conjunctive query

[Dalvi&S.04]

Theorem Let Q be a non-repeating CQ

- If Q is hierarchical, then $P_D(Q)$ is in PTIME.
- If Q is not hierarchical then $P_D(Q)$ is #P-hard.

By duality, the same holds for a non-repeating clause

Summary so Far

Complexity of $P_D(Q)$	Non-repeating CQ Non-repeating clause
PTIME	Hierarchical
#P - hard	Non-hierarchical

Outline

- Model Counting
- Small Dichotomy Theorem
- Dichotomy Theorem
- Query Compilation
- Conclusions, Open Problems

The Rules for Lifted Inference

Preprocess Q (omitted from this talk; see book),
then apply these rules (some have preconditions)

$$P(\neg Q) = 1 - P(Q)$$

negation

$$P(Q1 \wedge Q2) = P(Q1)P(Q2)$$
$$P(Q1 \vee Q2) = 1 - (1 - P(Q1))(1 - P(Q2))$$

Independent join / union

$$P(\exists z\, Q) = 1 - \Pi_{a \in Domain} (1 - P(Q[a/z]))$$
$$P(\forall z\, Q) = \Pi_{a \in Domain} P(Q[a/z])$$

Independent project

$$P(Q1 \wedge Q2) = P(Q1) + P(Q2) - P(Q1 \vee Q2)$$
$$P(Q1 \vee Q2) = P(Q1) + P(Q2) - P(Q1 \wedge Q2)$$

Inclusion/ exclusion

$\mathrm{FO}^{\mathrm{un}}$ = Unate FO

An FO sentence is *unate* if:

- Negations occur only on atoms
- Every relational symbol R either occurs only positively, or only negatively

$\mathrm{FO}^{\mathrm{un}}$ = FO restricted to unate sentences

Dichotomy Theorem

[Dalvi&S'12]

Theorem For any Q in $\forall^*FO^{un}$ (or $\exists^*FO^{un}$)

- If rules succeed, then $P_D(Q)$ in PTIME in |D|
- If rules fail, then $P_D(Q)$ is #P hard in |D|

Note: Unions of Conjunctive queries (UCQ) is essentially $\exists^*FO^{un}$

Example: Liftable Query

$$\boxed{Q_J() = S(x_1,y_1), R(y_1), S(x_2,y_2), T(y_2)}$$

$$= \underbrace{[S(x_1,y_1),R(y_1)]}_{Q_1} \wedge \underbrace{[S(x_2,y_2),T(y_2)]}_{Q_2}$$

$$P(Q_J) \quad = \quad P(Q_1) + P(Q_2) - P(Q_1 \vee Q_2)$$

PTIME (have seen before)

$y = y1 = y2$

$$Q_1 \vee Q_2 = \exists y\ [S(x_1,y),R(y) \vee S(x_2,y)),T(y)]$$

$P(Q_1 \vee Q_2) =$

$$= 1 - \Pi_{b \in \text{Domain}} \ (1 - P[S(x_1,b),R(b) \vee S(x_2,b)),T(b)]])$$

$$= 1 - \Pi_{b \in \text{Domain}} \ (1 - P[S(x_1,b)] \ * \ P[R(b) \vee T(b)]]) \ = \ldots \text{etc}$$

Runtime = $O(n^2)$.

Example: Liftable Query

$$\boxed{Q_J = \forall x_1 \forall y_1 \forall x_2 \forall y_2\ (S(x_1,y_1) \vee R(y_1) \vee S(x_2,y_2) \vee T(y_2))}$$

$$= [\underbrace{\forall x_1 \forall y_1 S(x_1,y_1) \vee R(y_1)}_{Q_1}] \vee [\underbrace{\forall x_2 \forall y_2 S(x_2,y_2) \vee T(y_2)}_{Q_2}]$$

$$P(Q_J) = P(Q_1) + P(Q_2) - P(Q_1 \wedge Q_2)$$

PTIME (have seen before)

$y = y1 = y2$

$$Q_1 \wedge Q_2 = \forall y\ [(\forall x_1 S(x_1,y) \vee R(y)) \wedge (\forall x_2 S(x_2,y)) \vee T(y)]$$

$$= \forall y\ [\forall x\, S(x,y) \vee (R(y) \wedge T(y))]$$

$$P(Q_1 \wedge Q_2) = \Pi_{b \in \text{Domain}}\ P[\forall x.S(x,b) \vee (R(b) \wedge T(b))] = \ldots\text{etc}$$

Runtime = $O(n^2)$.

Unliftable Queries H_k

Will drop $\forall$ to reduce clutter

$H_0 = R(x) \vee S(x,y) \vee T(y)$

$H_1 = [R(x_0) \vee S(x_0,y_0)] \wedge [S(x_1,y_1) \vee T(y_1)]$

$H_2 = [R(x_0) \vee S_1(x_0,y_0)] \wedge [S_1(x_1,y_1) \vee S_2(x_1,y_1)] \vee [S_2(x_2,y_2) \vee T(y_2)]$

$H_3 = [R(x_0) \vee S_1(x_0,y_0)] \wedge [S_1(x_1,y_1) \vee S_2(x_1,y_1)] \wedge [S_2(x_2,y_2) \vee S_3(x_2,y_2)] \wedge [S_3(x_3,y_3) \vee T(y_3)]$

. . .

Every H_k, $k \geq 1$
is hierarchical

Theorem. [Dalvi&S'12] Every query H_k is #P-hard

A Closer Look at H_k

If we drop any one clause → in PTIME

$$H_3 = \underbrace{[R(x_0) \vee S_1(x_0,y_0)]}_{} \wedge [\cancel{S_1(x_1,y_1) \vee S_2(x_1,y_1)}] \wedge \underbrace{[S_2(x_2,y_2) \vee S_3(x_2,y_2)] \wedge [S_3(x_3,y_3) \vee T(y_3)]}_{}$$

Independent join

Summary so Far

Complexity of $P_D(Q)$	Non-repeating CQ Non-repeating clauses	$\exists^* FO^{un}$ $\forall^* FO^{un}$
PTIME	Hierarchical	Rules succeed
#P - hard	Non-hierarchical	Rules fail

Outline

- Model Counting
- Small Dichotomy Theorem
- Dichotomy Theorem
- Query Compilation
- Conclusions, Open Problems

Lifted v.s. Grounded Inference

- To compute $P_D(Q)$:
  compute the lineage $F_{Q,D}$
  use DPLL-based algorithm for $P(F_{Q,D})$

- For which queries Q can this be in PTIME?

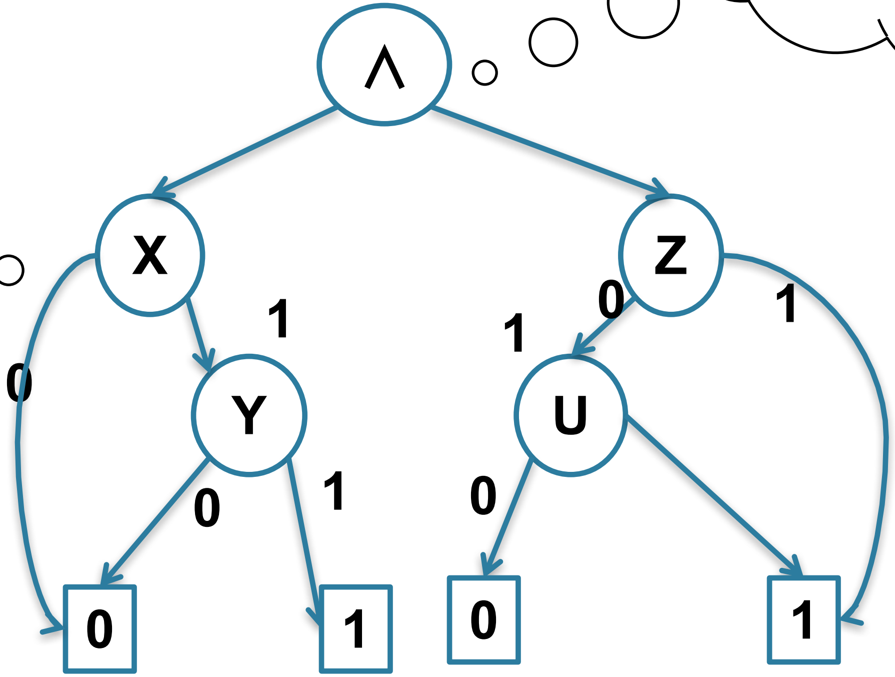
- [Huang&Darwiche'2005] The *trace* of a DPLL-based algorithm is "decision-DNNF"

Decision-DNNF

Def [Darwiche] A Decision-DNNF is a rooted DAG where:

- Internal nodes are decision or ∧
- Sink nodes are 0 or 1

Children of ∧ have disjoint sets of variables



Every root-to-sink path reads each variable at most once

Notations

$H_{k0} = \forall x \forall y\, R(x) \vee S_1(x,y)$
$H_{k1} = \forall x \forall y\, S_1(x,y) \vee S_2(x,y)$
$H_{k2} = \forall x \forall y\, S_2(x,y) \vee S_3(x,y)$
…
…
$H_{kk} = \forall x \forall y\, S_k(x,y) \vee T(y)$

$f(Z_0, Z_1, \ldots, Z_k)$ = a Boolean function

$Q = f(H_{k0}, H_{k1}, \ldots, H_{kk})$

Example: $f = Z_0 \wedge Z_1 \wedge \ldots \wedge Z_k$ then $f(H_{k0}, H_{k1}, \ldots, H_{kk}) = H_k$

Easy/Hard Queries

[Beame'14]

Theorem Let $Q = f(H_{k0}, H_{k1}, \ldots, H_{kk})$ where $f(Z_0, Z_1, \ldots, Z_k)$ is a monotone Boolean function.

- Any Decision-DNNF for $F_{Q,[n]}$ has size $2^{\Omega(\sqrt{n})}$.
- $P_D(Q)$ is in PTIME iff $\mu_Q(0, 1) = 0$

μ = Möbius function of the implicates lattice of Q

Consequence: Any DPLL-based algorithm takes time $2^{\Omega(\sqrt{n})}$, even if the query is in PTIME!

Cancellations

$$
\begin{aligned}
Q_W = \; & (H_{30} \wedge H_{32}) \vee \\
& (H_{30} \wedge H_{33}) \vee \\
& (H_{31} \wedge H_{33})
\end{aligned}
$$

$$
\begin{aligned}
H_{30} &= \forall x \forall y\, R(x) \vee S_1(x,y) \\
H_{31} &= \forall x \forall y\, S_1(x,y) \vee S_2(x,y) \\
H_{32} &= \forall x \forall y\, S_2(x,y) \vee S_3(x,y) \\
H_{33} &= \forall x \forall y\, S_3(x,y) \vee T(y)
\end{aligned}
$$

$$P(Q_W) = P(H_{30} \wedge H_{32}) + P(H_{30} \wedge H_{33}) + P(H_{31} \wedge H_{33}) +$$

$$- P(H_{30} \wedge H_{32} \wedge H_{33}) - P(H_{30} \wedge H_{31} \wedge H_{33})$$

$$\cancel{- P(H_{30} \wedge H_{31} \wedge H_{32} \wedge H_{33})}$$

$= H_3$ (hard !)

$$+ \cancel{P(H_{30} \wedge H_{31} \wedge H_{32} \wedge H_{33})}$$

Also $= H_3$

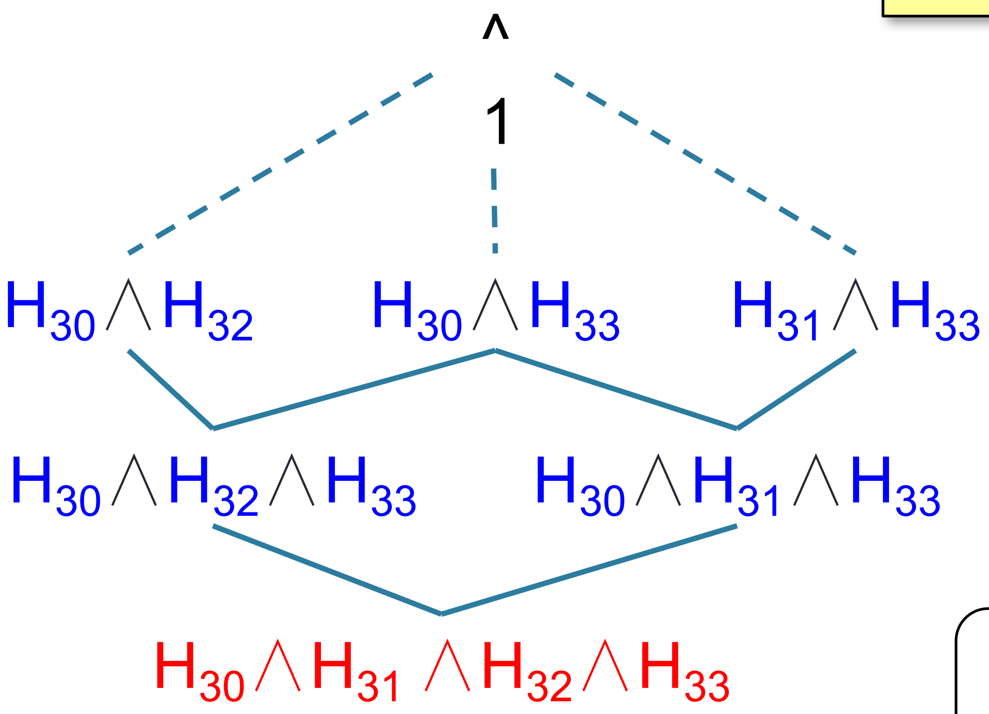
$P(Q_W)$ is in PTIME

The CNF Lattice

$$\begin{aligned} Q_W = \; & (H_{30} \wedge H_{32}) \vee \\ & (H_{30} \wedge H_{33}) \vee \\ & (H_{31} \wedge H_{33}) \end{aligned}$$

Definition. The DNF lattice of $Q = Q_1 \vee Q_2 \vee \ldots$ is:

- Elements are prime implicants
- Order is implication

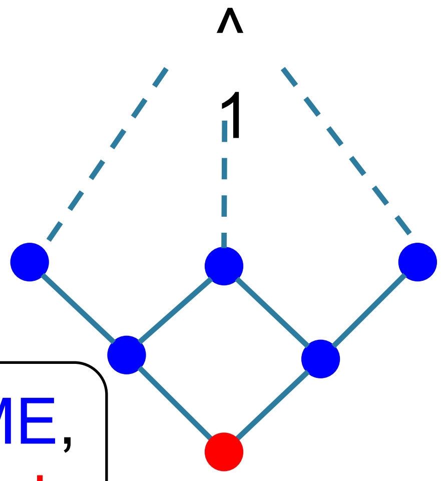


$\hat{1}$

$H_{30} \wedge H_{32}$ $H_{30} \wedge H_{33}$ $H_{31} \wedge H_{33}$

$H_{30} \wedge H_{32} \wedge H_{33}$ $H_{30} \wedge H_{31} \wedge H_{33}$

$H_{30} \wedge H_{31} \wedge H_{32} \wedge H_{33}$



Nodes ● in PTIME,
Nodes ● #P hard.

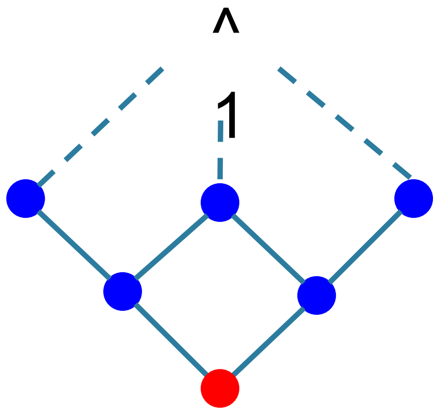
The Möbius' Function

Def. The Möbius function:

$$\mu(\hat{1},\hat{1}) = 1 \qquad \mu(u,\hat{1}) = -\Sigma_{u < v \leq \hat{1}}\ \mu(v, \hat{1})$$

Möbius' Inversion Formula:

$$P(Q) = -\Sigma_{Qi < \hat{1}}\ \mu(Qi,\hat{1})\ P(Qi)$$



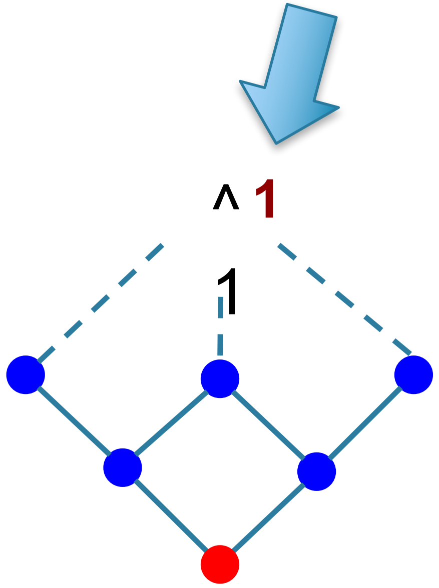
The Möbius' Function

Def. The Möbius function:

$$\mu(\hat{1},\hat{1}) = 1 \qquad \mu(u,\hat{1}) = -\sum_{u < v \leq \hat{1}} \mu(v,\hat{1})$$

Möbius' Inversion Formula:

$$P(Q) = -\sum_{Qi < \hat{1}} \mu(Qi,\hat{1})\, P(Qi)$$



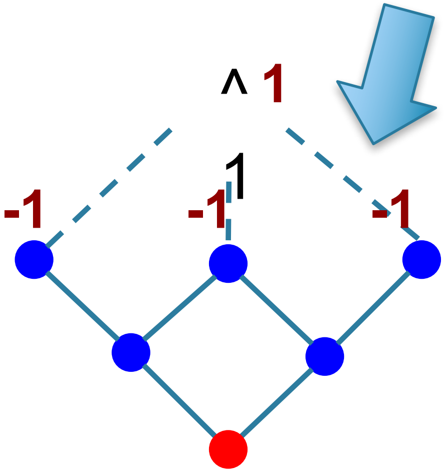
The Möbius' Function

Def. The Möbius function:

$$\mu(\hat{1},\hat{1}) = 1 \qquad \mu(u,\hat{1}) = -\sum_{u<v\leq\hat{1}} \mu(v,\hat{1})$$

Möbius' Inversion Formula:

$$P(Q) = -\sum_{Qi<\hat{1}} \mu(Qi,\hat{1})\, P(Qi)$$



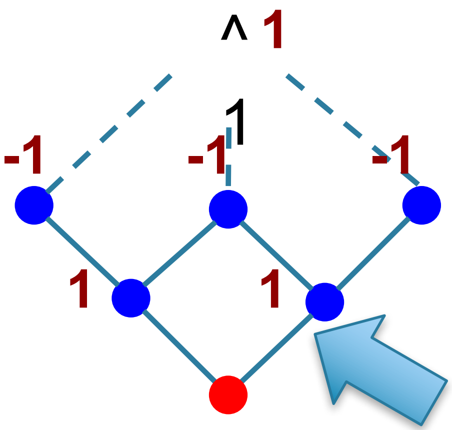
The Möbius' Function

Def. The Möbius function:

$$\mu(\hat{1},\hat{1}) = 1 \qquad \mu(u,\hat{1}) = - \Sigma_{u < v \leq \hat{1}} \, \mu(v, \hat{1})$$

Möbius' Inversion Formula:

$$P(Q) = - \Sigma_{Qi < \hat{1}} \, \mu(Qi,\hat{1}) \, P(Qi)$$



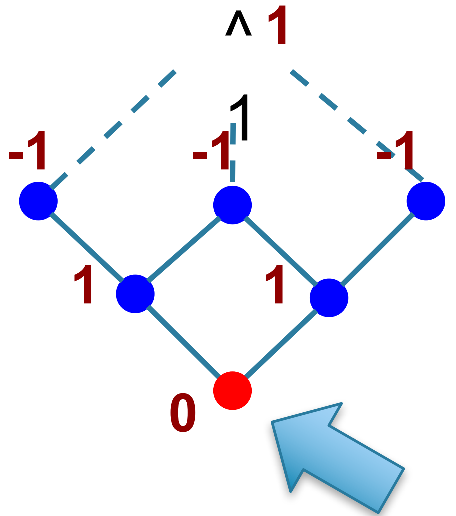
The Möbius' Function

Def. The Möbius function:

$$\mu(\hat{1},\hat{1}) = 1 \qquad \mu(u,\hat{1}) = -\Sigma_{u < v \leq \hat{1}}\ \mu(v,\hat{1})$$

Möbius' Inversion Formula:

$$P(Q) = -\Sigma_{Qi < \hat{1}}\ \mu(Qi,\hat{1})\ P(Qi)$$



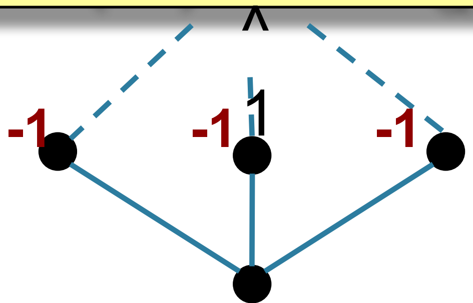
The Möbius' Function

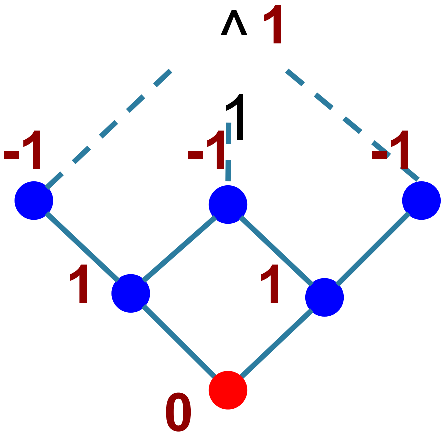
Def. The Möbius function:

$$\mu(\hat{1},\hat{1}) = 1 \qquad \mu(u,\hat{1}) = -\sum_{u < v \leq \hat{1}} \mu(v,\hat{1})$$

Möbius' Inversion Formula:

$$P(Q) = -\sum_{Q_i < \hat{1}} \mu(Q_i,\hat{1})\, P(Q_i)$$



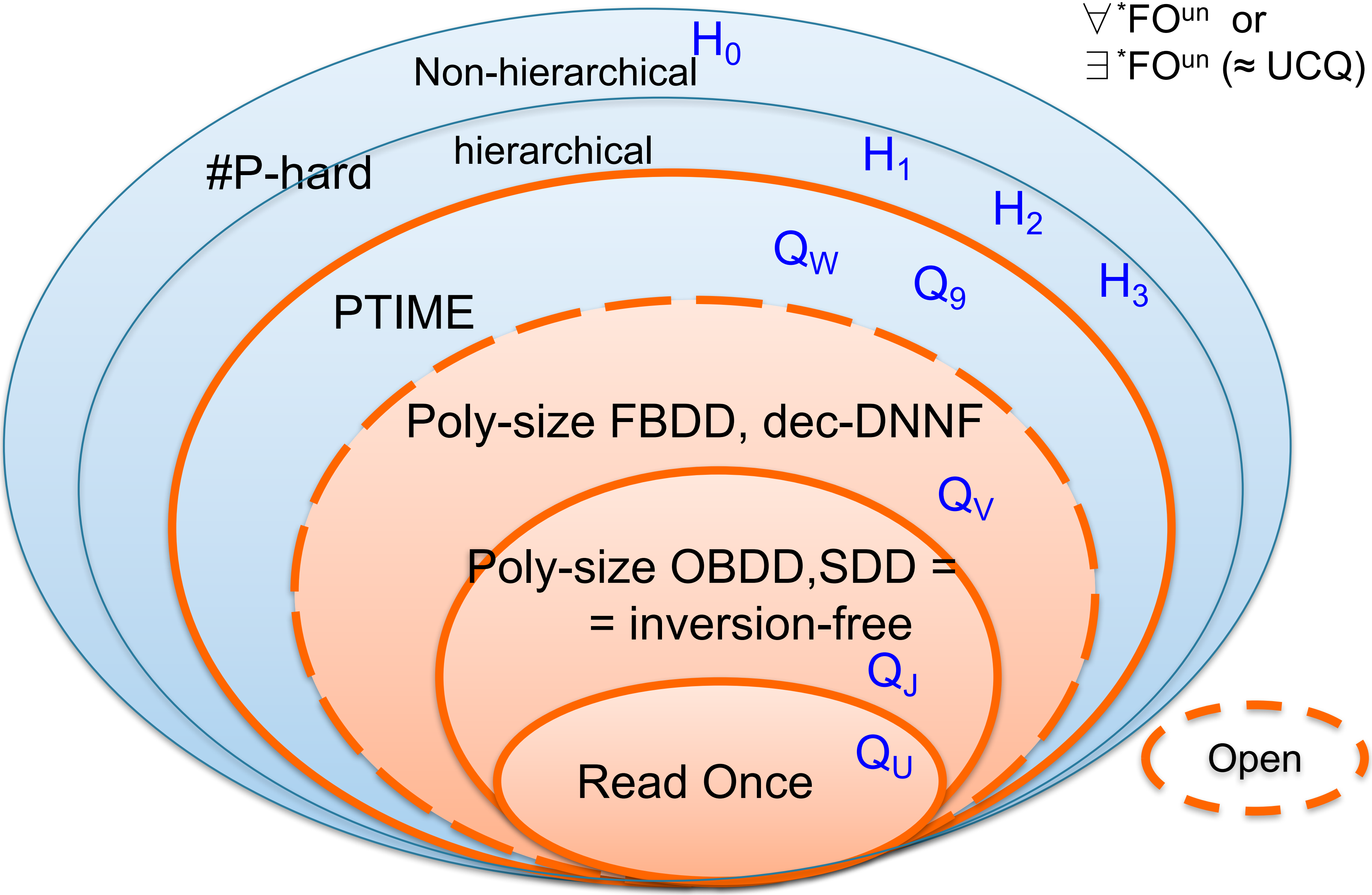


Summary

	nr CQ nr clause	$\exists^*\mathrm{FO}^{un}$ $\forall^*\mathrm{FO}^{un}$	$f(H_{k0},\ldots,H_{kk})$
PTIME	Hierarchical	Rules succeed	$\mu(0,1) = 0$
#P - hard	Non-hierarchical	Rules fail	$\mu(0,1) \neq 0$
DPLL			$2^{\Omega(\sqrt{n})}$

Möbius Über Alles

$\forall^*FO^{un}$ or
$\exists^*FO^{un}$ (≈ UCQ)



Outline

- Model Counting
- Small Dichotomy Theorem
- Dichotomy Theorem
- Query Compilation
- Conclusions, Open Problems

Summary

- Query evaluation on probabilistic databases = weighted model counting
- Each query Q defines a different WMC problem
- Dichotomy: depending on Q, WMC is in PTIME or #P-hard
- Using a DPLL-based algorithm on the grounded Q is suboptimal

Discussion: Extensions

Open problems: extend the dichotomy theorem to:

- Mixed probabilistic/deterministic relations
- Functional dependencies
- Interpreted predicates: $<$, $\neq$

Open problem: complexity of MAP

Discussion: Symmetric Relations

- A relation R is *symmetric* if all ground tuples have the same probability
- [van den Broeck'14] For every Q in FO^2, **P**(Q) is in PTIME on symmetric databases.
- [Beame'15] Hardness results.
- In general the complexity is open

Discussion: Negation

[Fink&Olteanu'14] Restrict FO to non-repeating exrpessions

Dan Olteanu's talk next

- Theorem Hierarchical expressions are in PTIME, non-hierarchical are #P-hard.

[Gribkoff,S.,v.d.Broeck'14] $\forall^*$FO or $\exists^*$FO

- Need resolution compute some queries with negation

Open problem: completeness/dichotomy?

Thank You!

BACKUP

Weighted Model Counting

- Each variable X has a weight w(X);
- Weight of a model = $\prod_{X=\text{true}} w(X)$
- WMC(F) = sum of weights of models of F

Example:

$F = (X1 \vee X2) \wedge (X2 \vee X3) \wedge (X3 \vee X1)$

WMC(F) = w2*w3 +
w1*w3 +
w1*w2 +
w1*w2*w3

X1	X2	X3	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	**1**
1	0	0	0
1	0	1	**1**
1	1	0	**1**
1	1	1	**1**

Set w(X) = 1: then WMC(F) = #F

Probability of a Formula

- Each variable X has a probability p(X);
- P(F) = probability that F=true, when each X is set to true independently

Example:
$F = (X1 \vee X2) \wedge (X2 \vee X3) \wedge (X3 \vee X1)$

$$P(F) = (1-p1)*p2*p3 + p1*(1-p2)*p3 + p1*p2*(1-p3) + p1*p2*p3$$

X1	X2	X3	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	**1**
1	0	0	0
1	0	1	**1**
1	1	0	**1**
1	1	1	**1**

Set $w(X) = p(X)/(1-p(X))$
Then $P(F) = WMC(F) / Z$, where $Z = \Pi_X (1+w(X))$

Discussion: Dichotomy for #SAT

- [Creignou&Hemann'96] consider model counting #F where the formula F is given by *generalized clauses*

- **Dichotomy Theorem**: #F is in PTIME when all clauses are affine, #P-hard otherwise

- Not helpful in our context:
  - If Q is a UCQ, then the clauses of $F_{Q,D}$ are of the form $X \lor Y \lor Z \ldots$ and are not affine;
  - But $P(F_{Q,D})$ is not always #P-hard, because Q restricts the struture of the clauses

Warm-up: Weights

Replace probabilities with weights:

R:

x	y	w
a1	b1	w_1
a2	b1	w_2
a3	b2	w_3

S:

y	w
b1	v_1
b2	v_2
b3	v_3

$$P_D(\text{world}) = \text{Weight}(\text{world})/Z$$

$$Z = \Sigma_{\text{world'}}\ \text{Weight}(\text{world'})$$

Weight of a possible world:

Weight(

R:

x	y
a1	b1
a2	b1

S:

y
b2

) = $w_1w_2v_2$

$$Z = (1+v_1)\ (1+v_2)\ (1+v_3)\ (1+w_1)\ (1+w_2)\ (1+w_3)$$

Markov Logic Networks

Replace probabilities with weights:

R:

x	y	w
a1	b1	w_1
a2	b1	w_2
a3	b2	w_3

S:

y	w
b1	v_1
b2	v_2
b3	v_3

Add soft constraints:

$R(x,y) \Rightarrow S(y)$ w_4

Markov Logic Networks

Replace probabilities with weights:

R:

x	y	w
a1	b1	w_1
a2	b1	w_2
a3	b2	w_3

S:

y	w
b1	v_1
b2	v_2
b3	v_3

Add soft constraints:

$R(x,y) \Rightarrow S(y) \quad w_4$

Weight of a possible world:

Weight(

R:

x	y
a1	b1
a3	b2

S:

y
b1

$) = w_1 w_3 v_1 w_4 w_4 w_4 w_4 w_4$

Markov Logic Networks

Replace probabilities with weights:

R:

x	y	w
a1	b1	w_1
a2	b1	w_2
a3	b2	w_3

S:

y	w
b1	v_1
b2	v_2
b3	v_3

Add soft constraints:

$R(x,y) \Rightarrow S(y) \quad w_4$

$P_{MLN}(\text{world}) = \text{Weight}(\text{world})/Z$

$Z = \Sigma_{\text{world'}} \ \text{Weight}(\text{world'})$

Weight of a possible world:

R:

x	y
a1	b1
a3	b2

S:

y
b1

Weight(R, S) = $w_1 w_3 v_1 w_4 w_4 w_4 w_4 w_4$

Markov Logic Networks

Replace probabilities with weights:

R:

x	y	w
a1	b1	w_1
a2	b1	w_2
a3	b2	w_3

S:

y	w
b1	v_1
b2	v_2
b3	v_3

Add soft constraints:

$R(x,y) \Rightarrow S(y) \quad w_4$

$$P_{MLN}(\text{world}) = \text{Weight}(\text{world})/Z$$

$$Z = \Sigma_{\text{world'}} \ \text{Weight}(\text{world'})$$

$Z = \cancel{(1+v_1)(1+v_2)(1+v_3)}$ $\cancel{(1+w_1)(1+w_2)(1+w_3)}$

Z is #P-hard to compute

Weight of a possible world:

Weight(

R:

x	y
a1	b1
a3	b2

S:

y
b1

) = $w_1w_3v_1w_4w_4w_4w_4w_4$

Discussion

Weights v.s. probabilities

Inconsistent:
S(x): p=0.5
S(x)$\wedge$R(x): p=0.9

- Soft constraints *with probabilities* may be inconsistent
- Soft constraints *with weights (≠ 0, ∞)* always consistent

Consistent:
S(x): w=5
S(x)$\wedge$R(x): w=9

Weight values have no semantics!

- Learned from training data

MLN's to Tuple-Independent PDB

Replace probabilities with weights:

R:

x	y	w
a1	b1	w_1
a2	b1	w_2
a3	b2	w_3

S:

y	w
b1	v_1
b2	v_2
b3	v_3

Soft constraint:

$R(x,y) \Rightarrow S(y)$ w_4

Replace with hard constraint:

MLN's to Tuple-Independent PDB

Replace probabilities with weights:

R:

x	y	w
a1	b1	w_1
a2	b1	w_2
a3	b2	w_3

S:

y	w
b1	v_1
b2	v_2
b3	v_3

Soft constraint:

$R(x,y) \Rightarrow S(y) \qquad w_4$

Replace with hard constraint:

$$\Gamma \equiv \forall x \forall y \; A(x,y) \Leftrightarrow (R(x,y) \Rightarrow S(y))$$

New relation A:

x	y	w
a1	b1	w_4
a1	b2	w_4
a1	b3	w_4
a1	b1	w_4
…		

MLN’s to Tuple-Independent PDB

Replace probabilities with weights:

R:

x	y	w
a1	b1	w_1
a2	b1	w_2
a3	b2	w_3

S:

y	w
b1	v_1
b2	v_2
b3	v_3

Soft constraint:

$$R(x,y) \Rightarrow S(y) \qquad w_4$$

Replace with hard constraint:

$$\Gamma \equiv \forall x \forall y \; A(x,y) \Leftrightarrow (R(x,y) \Rightarrow S(y))$$

New relation A:

x	y	w
a1	b1	w_4
a1	b2	w_4
a1	b3	w_4
a1	b1	w_4
…		

Weight(

R:

x	y
a1	b1
a3	b2

S:

y
b1

A:

x	y
a1	b1
a1	b2
a2	b1
a2	b2
a3	b1

) = $w_1 w_3 v_1 w_4 w_4 w_4 w_4 w_4$

MLN's to Tuple-Independent PDB

Replace probabilities with weights:

R:

x	y	w
a1	b1	w_1
a2	b1	w_2
a3	b2	w_3

S:

y	w
b1	v_1
b2	v_2
b3	v_3

Soft constraint:

$R(x,y) \Rightarrow S(y) \quad w_4$

Replace with hard constraint:

$$\Gamma \equiv \forall x \forall y \; A(x,y) \Leftrightarrow (R(x,y) \Rightarrow S(y))$$

New relation A:

x	y	w
a1	b1	w_4
a1	b2	w_4
a1	b3	w_4
a1	b1	w_4
…		

Weight(

R:

x	y
a1	b1
a3	b2

S:

y
b1

A:

x	y
a1	b1
a1	b2
a2	b1
a3	b1

) = $w_1w_3v_1w_4w_4w_4w_4w_4$

Theorem: $P_{MLN}(Q) = P_D(Q \mid \Gamma)$

Improved Translation

Soft constraint:

$R(x,y) \Rightarrow S(y) \quad w_4$

Replace with hard constraint:

$\Gamma \equiv \forall x \forall y$
$A(x,y) \Rightarrow (R(x,y) \Rightarrow S(x))$

Replace $\Leftrightarrow$ with $\Rightarrow$

A clause remains a clause!

New weight is w_4-1

Probability may be < 0 !!! That's OK

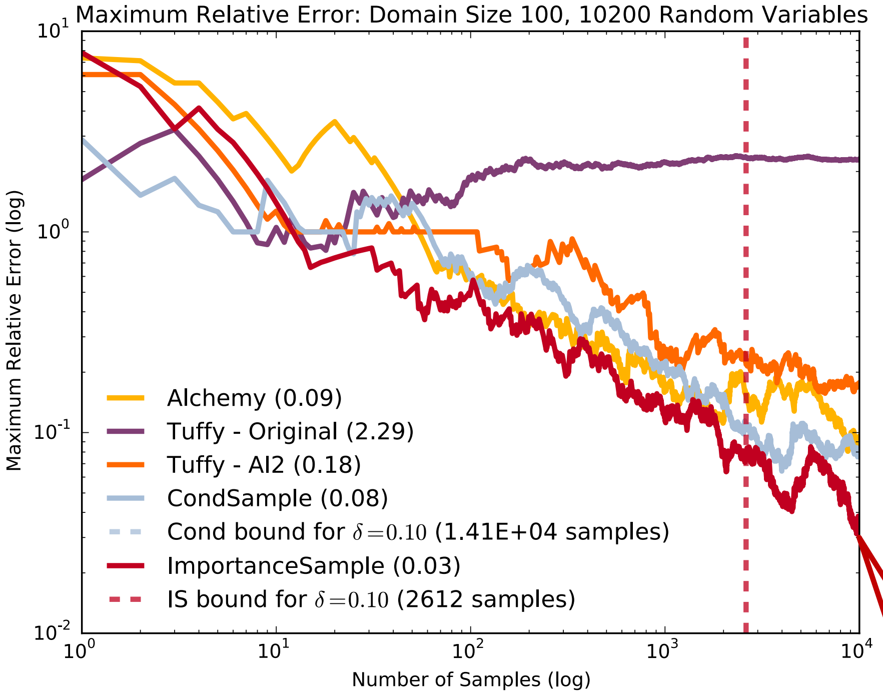
New relation A:

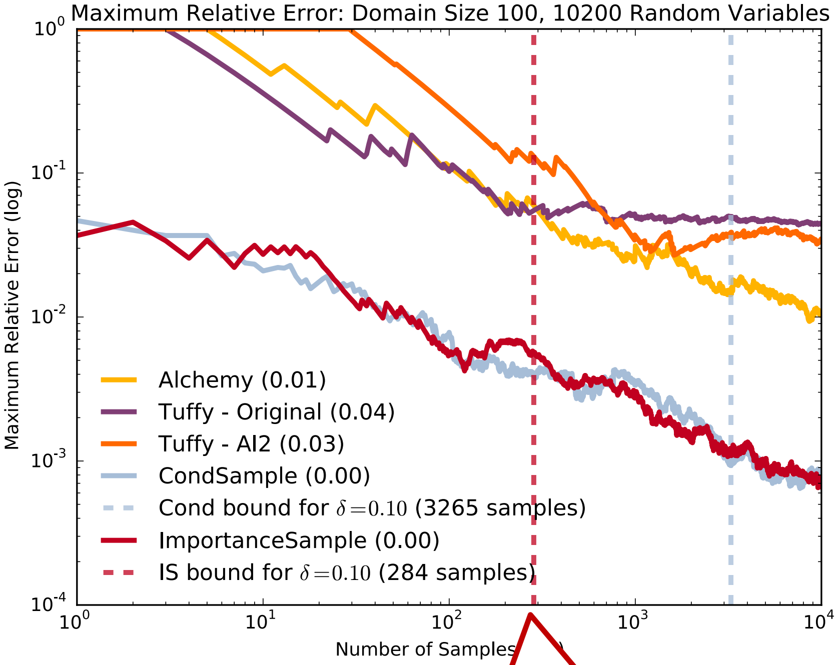
x	y	w
a1	b1	w_4-1
a1	b2	w_4-1
a1	b3	w_4-1
a1	b1	w_4-1
…		

Theorem: $P_{MLN}(Q) = P_D(Q \mid \Gamma)$

[VLDB’2016]

SlimShot = SafePlans + Sample





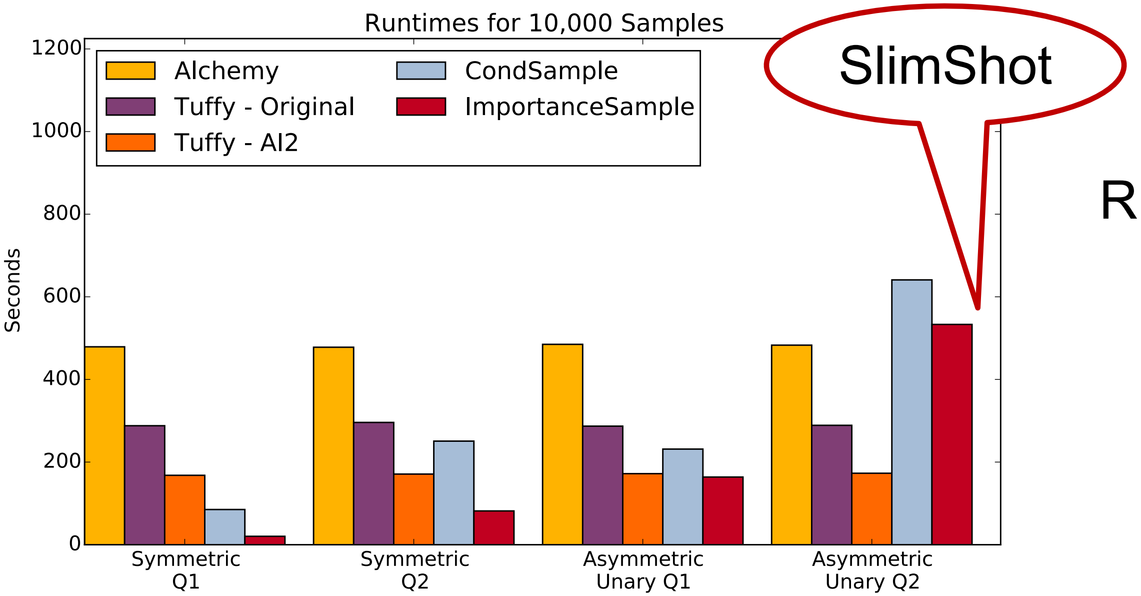
SlimShot

SlimShot
needs N≈200 for
Accuracy=10%

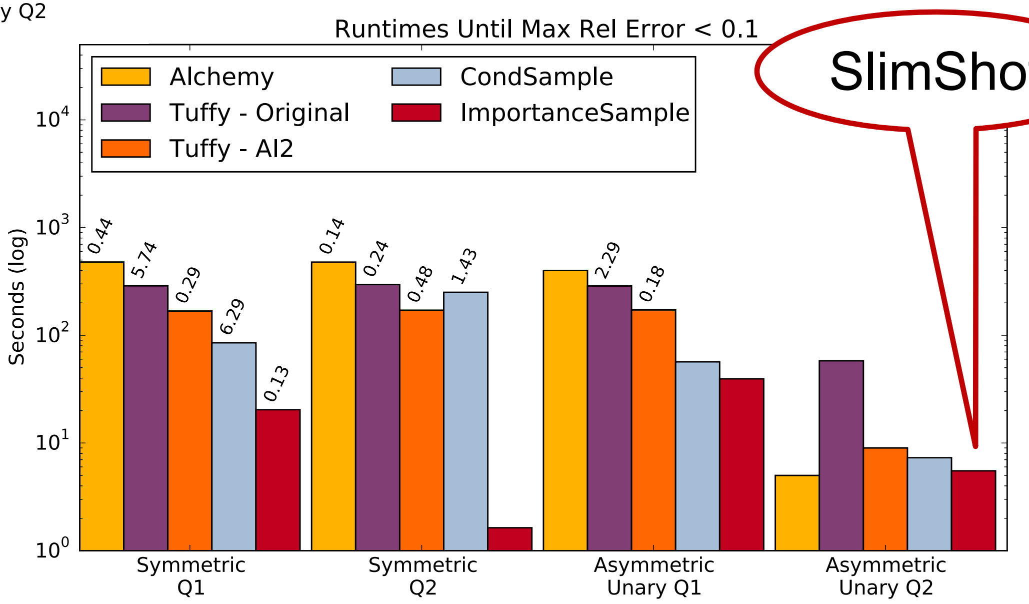
Accuracy = f(Number of Samples)
Lower is better

Dataset: Smokers

Runtime



Runtime = f(N), where N=10000

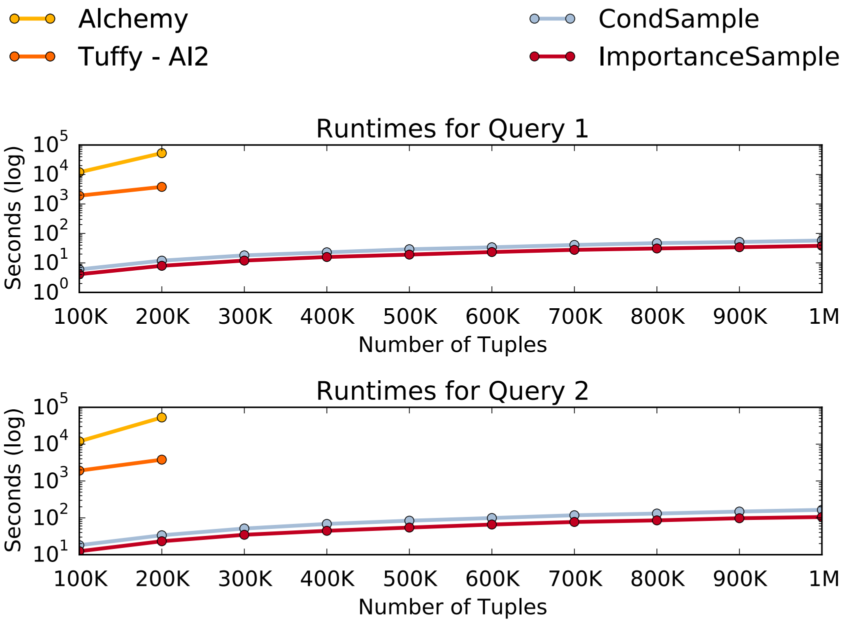


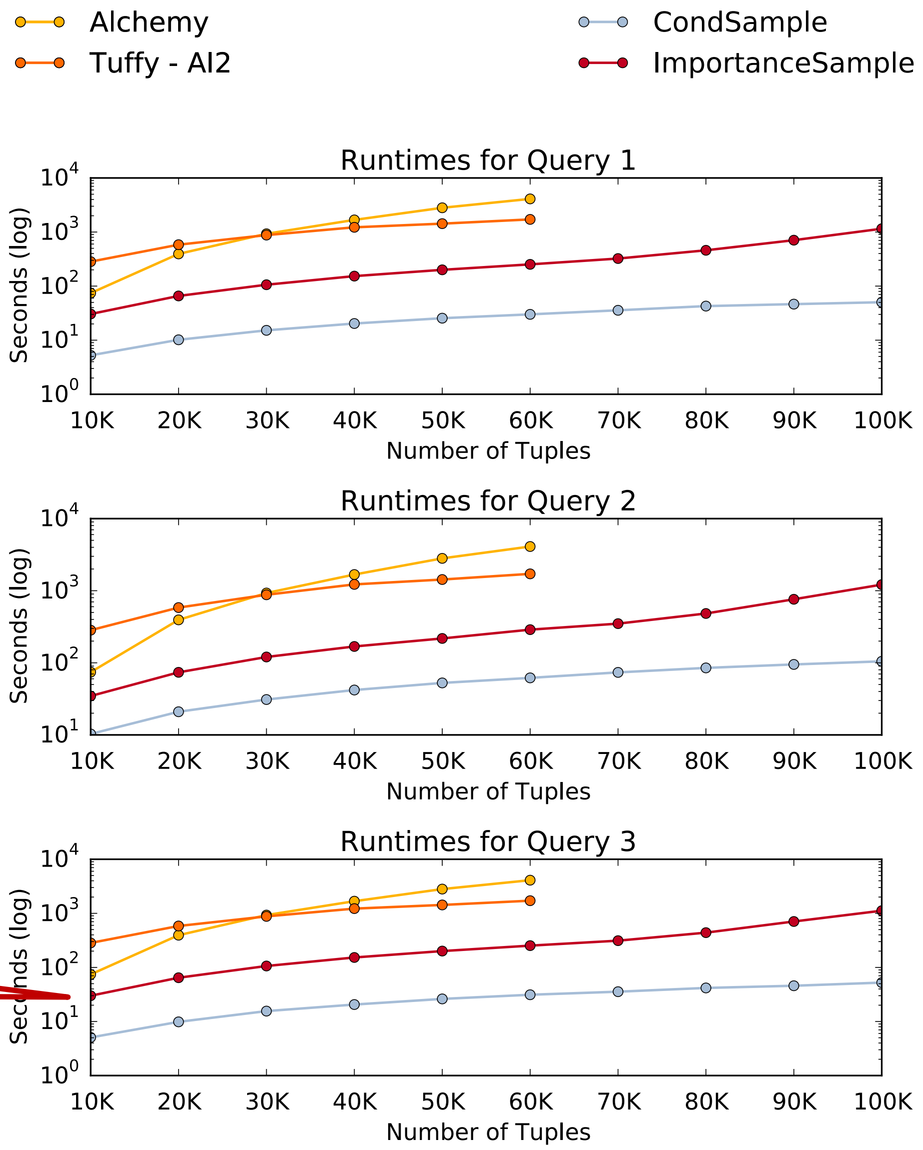
Runtime = f(precision)

Dataset: Smokers

Dataset: Drinkers

Scalability





SlimShot

Duality

- The *dual* of a query Q is the formula obtained by the following transformations:
  - $\wedge$ / $\vee$ → $\vee$ / $\wedge$ $\forall$ / $\exists$ → $\exists$ / $\forall$
- Q and its dual have the same complexity

Query:

$H_0() = R(x), S(x,y), T(y)$

$\exists x \exists y\, (R(x) \wedge S(x,y) \wedge T(y))$

Dual query:

$H_0 = \forall x \forall y\, (R(x) \vee S(x,y) \vee T(y))$

$\top \sqsubseteq \forall S.R$

Example: Liftable Clause

$Q = \forall x \forall y\, S(x,y) \Rightarrow R(y)$ $= \forall y\, (\exists x\, S(x,y) \Rightarrow R(y))$

$$P(Q) = \Pi_{b \in Domain} P(\exists x\, S(x,b) \Rightarrow R(b))$$

Indep. $\forall$

$$P(Q) = \Pi_{b \in Domain} [1 - P(\exists x\, S(x,b)) \times (1 - P(R(b)))]$$

Indep. or:
$P(X \Rightarrow Y) =$
$= P(\neg X \vee Y)$
$= P(X)\, (1\text{-}P(Y))$

$$P(Q) = \Pi_{b \in Domain} [1 - (1 - \Pi_{a \in Domain} (1\text{-}P(S(a,b)))) \times (1 - P(R(b)))]$$

Indep. $\exists$

Lookup the probabilities in D

Runtime = $O(n^2)$.