

Outline

- Part 1: Motivation
- Part 2: Probabilistic Databases
- Part 3: Weighted Model Counting
- Part 4: Lifted Inference for WFOMC



- Part 5: Completeness of Lifted Inference
- Part 6: Query Compilation
- Part 7: Symmetric Lifted Inference Complexity
- Part 8: Open-World Probabilistic Databases
- Part 9: Discussion & Conclusions

What Everyone Should Know about Databases

- **Database** = several relations (a.k.a. tables)
- **SQL Query** = FO Formula
- **Boolean Query** = FO Sentence

What Everyone Should Know about Databases

Database: relations (= tables)

D =

Smoker

X	Y
Alice	2009
Alice	2010
Bob	2009
Carol	2010

Friend

X	Z
Alice	Bob
Alice	Carol
Bob	Carol
Carol	Bob

What Everyone Should Know about Databases

Database: relations (= tables)

$D =$

X	Y
Alice	2009
Alice	2010
Bob	2009
Carol	2010

X	Z
Alice	Bob
Alice	Carol
Bob	Carol
Carol	Bob

Query: First Order Formula

Find friends of smokers in 2009

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

Query answer: $Q(D) =$

Z
Bob
Carol

Conjunctive Queries **CQ** = FO(\exists , \wedge)

Union of CQs **UCQ** = FO(\exists , \wedge , \vee)

What Everyone Should Know about Databases

Database: relations (= tables)

$D =$

X	Y
Alice	2009
Alice	2010
Bob	2009
Carol	2010

X	Z
Alice	Bob
Alice	Carol
Bob	Carol
Carol	Bob

Query: First Order Formula

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

Find friends of smokers in 2009

Query answer: $Q(D) =$

Z
Bob
Carol

Conjunctive Queries $CQ = FO(\exists, \wedge)$

Union of CQs $UCQ = FO(\exists, \wedge, \vee)$

Boolean Query: FO Sentence

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

Query answer: $Q(D) = \text{TRUE}$

What Everyone Should Know about Databases

Declarative Query

“what”

→

→

Query Plan

“how”

What Everyone Should Know about Databases

Declarative Query
“what”

→
→

Query Plan
“how”

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

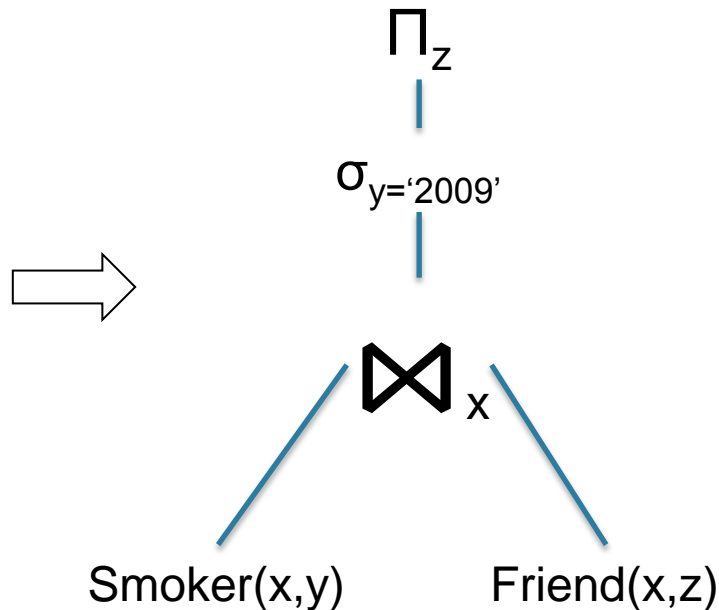
What Everyone Should Know about Databases

Declarative Query
“what”

→
→

Query Plan
“how”

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



Logical Query Plan

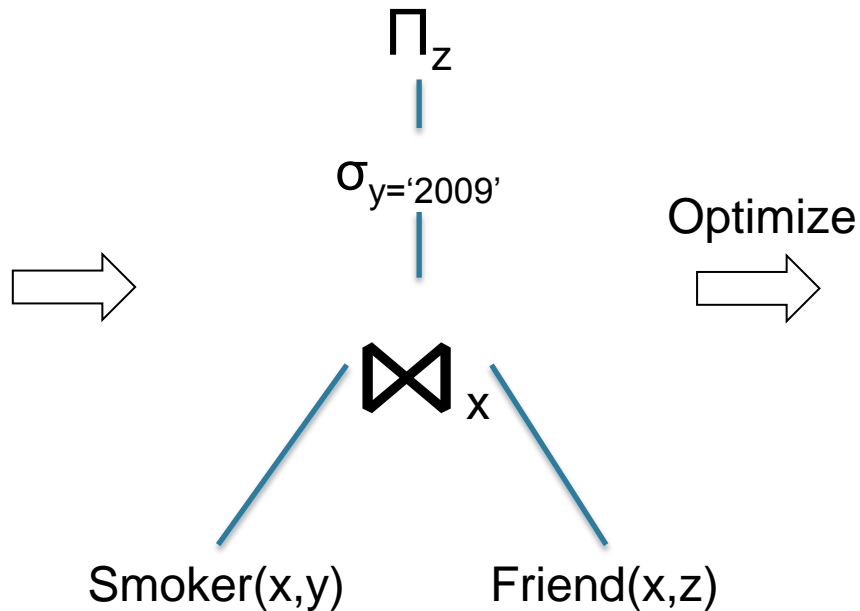
What Everyone Should Know about Databases

Declarative Query
“what”



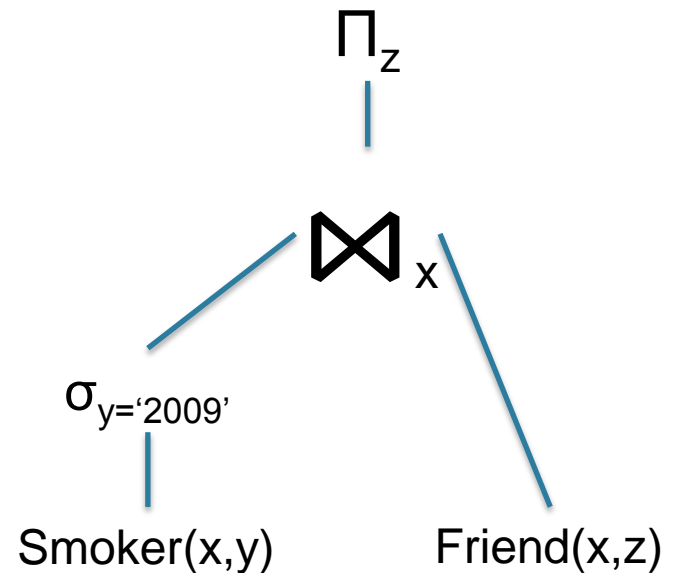
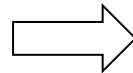
Query Plan
“how”

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



Logical Query Plan

Optimize



Logical Query Plan

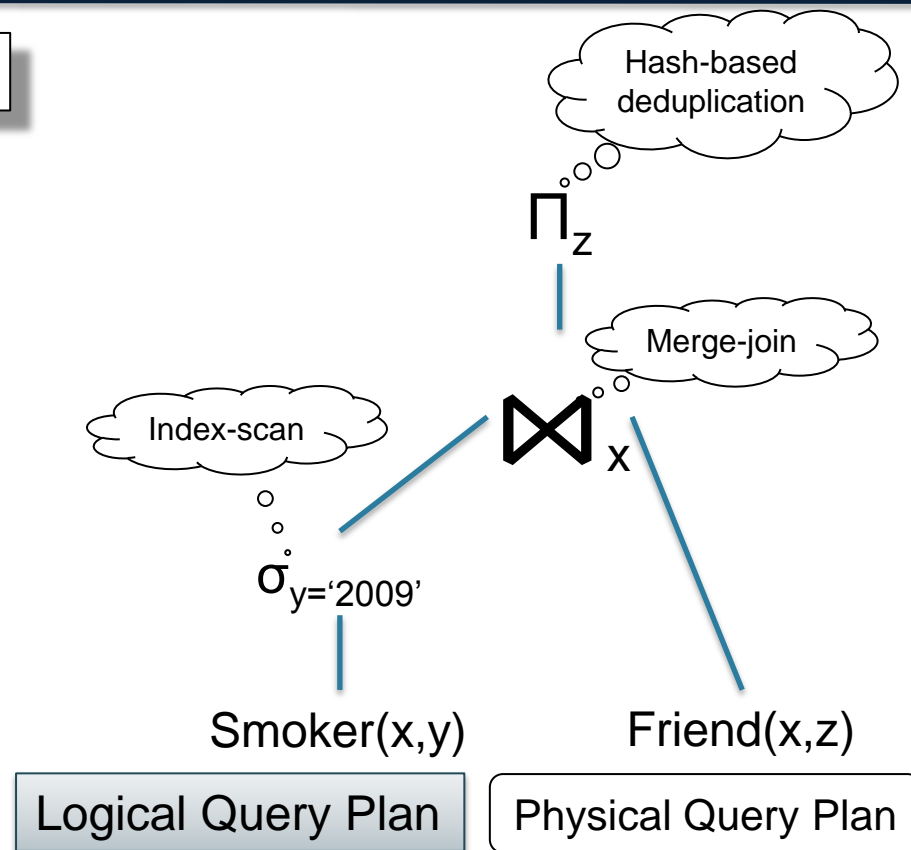
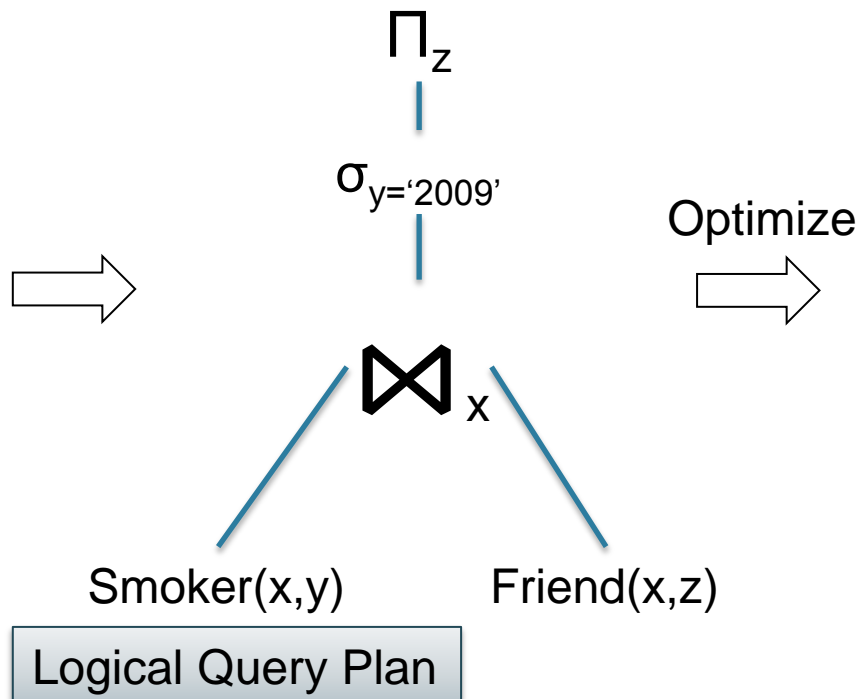
What Everyone Should Know about Databases

Declarative Query
“what”



Query Plan
“how”

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



What Every **Researcher** Should Know about Databases

Problem: compute $Q(D)$

Moshe Vardi [Vardi'82]
2008 ACM SIGMOD Contribution Award



This talk: query = **blue**, data = **red**

What Every **Researcher** Should Know about Databases

Problem: compute $Q(D)$

Moshe Vardi [Vardi'82]
2008 ACM SIGMOD Contribution Award

- Data complexity:
fix Q , complexity = $f(D)$



This talk: query = **blue**, data = **red**

What Every **Researcher** Should Know about Databases

Problem: compute $Q(D)$

Moshe Vardi [Vardi'82]
2008 ACM SIGMOD Contribution Award

- Data complexity:
fix Q , complexity = $f(D)$
- Query complexity: (expression complexity)
fix D , complexity = $f(Q)$
- Combined complexity:
complexity = $f(D, Q)$



This talk: query = **blue**, data = **red**

Probabilistic Databases

- **A probabilistic database** = relational database where each tuple is a random variable
- **Semantics** = probability distribution over possible worlds (deterministic databases)
- In this talk: tuples are independent events

Example

Probabilistic database **D**:

Friend

x	y	P
A	B	p_1
A	C	p_2
B	C	p_3

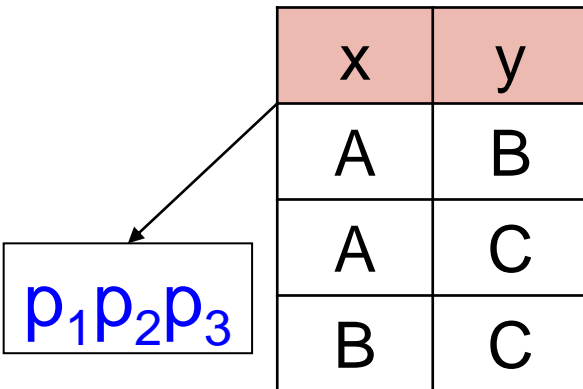
Example

Probabilistic database **D**:

Friend

x	y	P
A	B	p_1
A	C	p_2
B	C	p_3

Possible worlds semantics:



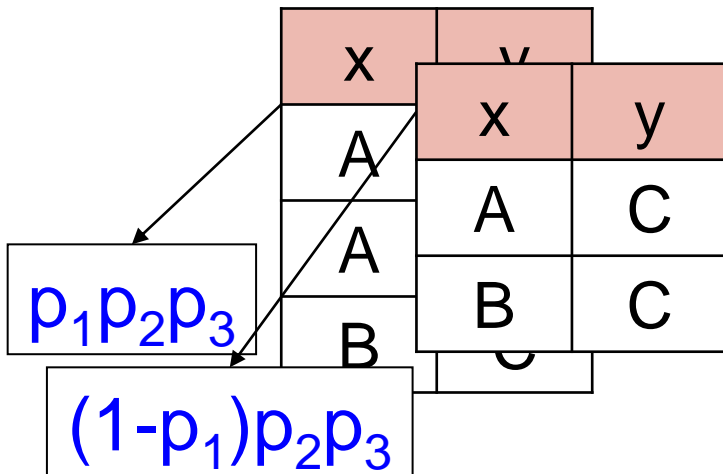
Example

Probabilistic database **D**:

Friend

x	y	P
A	B	p_1
A	C	p_2
B	C	p_3

Possible worlds semantics:



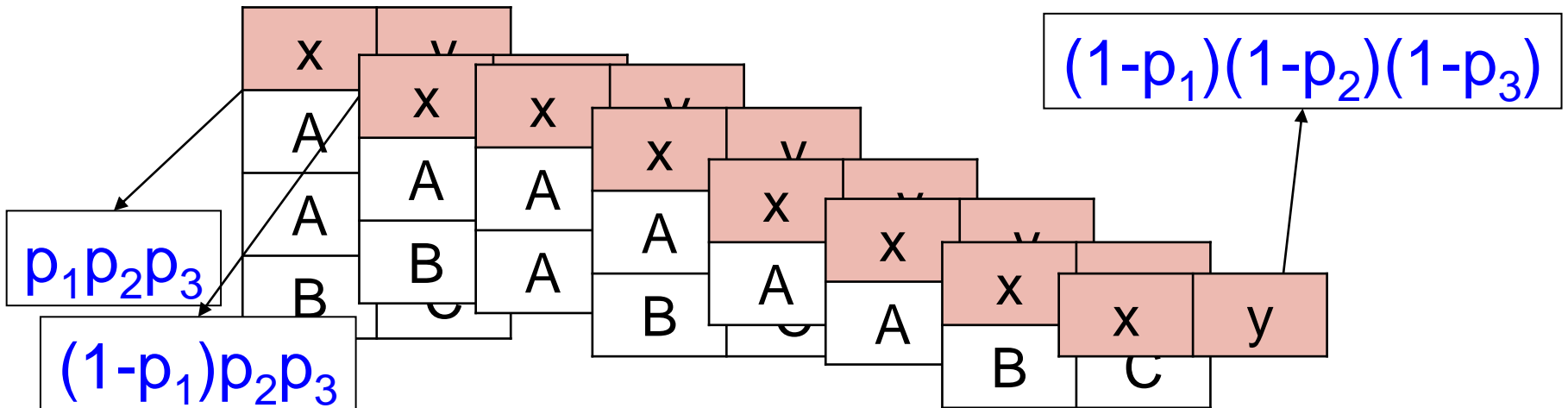
Example

Probabilistic database **D**:

Friend

x	y	P
A	B	p_1
A	C	p_2
B	C	p_3

Possible worlds semantics:



Query Semantics

Fix a Boolean query Q , probabilistic database D :

$P(Q | D) = P_D(Q) =$ marginal probability of Q
on possible words of D

$$Q = \exists x \exists y \text{ Smoker}(x) \wedge \text{Friend}(x,y)$$

An Example

$$P(Q \mid D) =$$

Smoker

x	P
A	p_1
B	p_2
C	p_3

Friend

x	y	P
A	D	q_1
A	E	q_2
B	F	q_3
B	G	q_4
B	H	q_5

$$Q = \exists x \exists y \text{ Smoker}(x) \wedge \text{Friend}(x,y)$$

An Example

$$P(Q \mid D) = 1 - (1 - q_1) * (1 - q_2)$$

Smoker

x	P
A	p_1
B	p_2
C	p_3

Friend

x	y	P
A	D	q_1
A	E	q_2
B	F	q_3
B	G	q_4
B	H	q_5

$$Q = \exists x \exists y \text{ Smoker}(x) \wedge \text{Friend}(x,y)$$

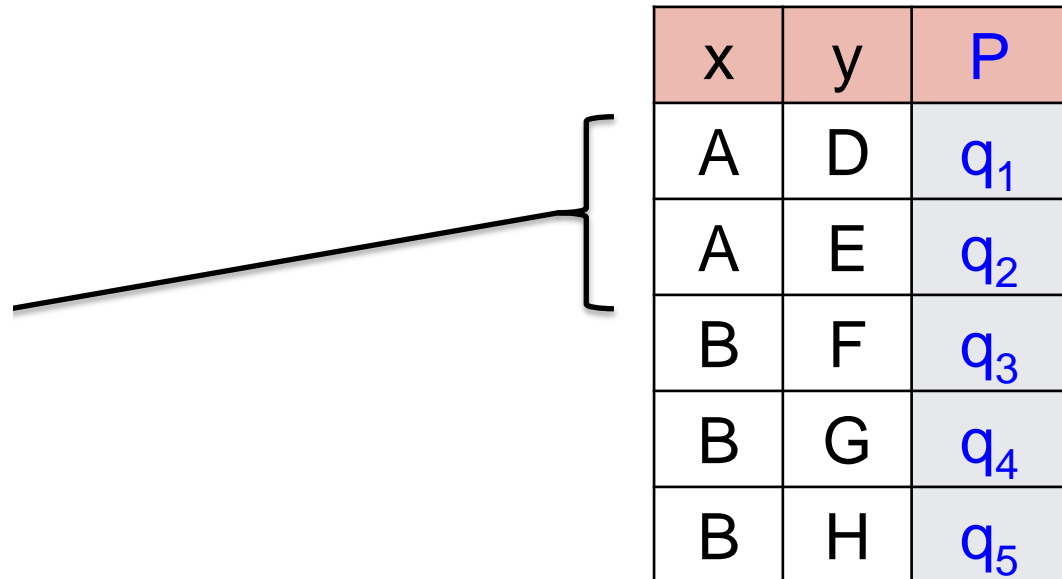
An Example

$$P(Q \mid D) = p_1 * [1 - (1 - q_1) * (1 - q_2)]$$

Smoker

x	P
A	p_1
B	p_2
C	p_3

Friend



x	y	P
A	D	q_1
A	E	q_2
B	F	q_3
B	G	q_4
B	H	q_5

$$Q = \exists x \exists y \text{ Smoker}(x) \wedge \text{Friend}(x,y)$$

An Example

$$P(Q \mid D) = p_1 * [1 - (1 - q_1) * (1 - q_2)] \\ 1 - (1 - q_3) * (1 - q_4) * (1 - q_5)$$

Smoker

x	P
A	p_1
B	p_2
C	p_3

Friend

x	y	P
A	D	q_1
A	E	q_2
B	F	q_3
B	G	q_4
B	H	q_5

$$Q = \exists x \exists y \text{ Smoker}(x) \wedge \text{Friend}(x,y)$$

An Example

$$P(Q \mid D) =$$

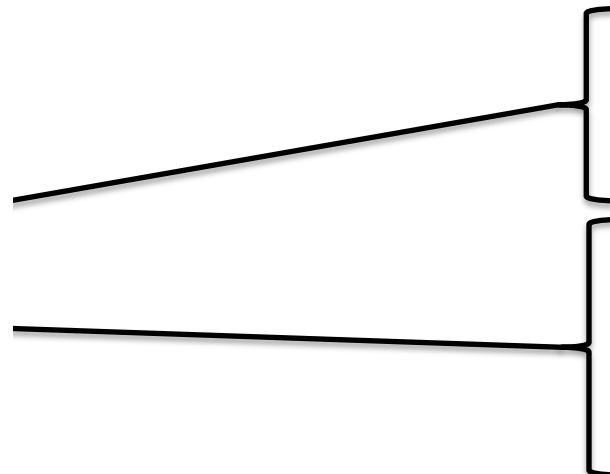
$$p_1 * [1 - (1 - q_1) * (1 - q_2)]$$

$$p_2 * [1 - (1 - q_3) * (1 - q_4) * (1 - q_5)]$$

Smoker

x	P
A	p_1
B	p_2
C	p_3

Friend



x	y	P
A	D	q_1
A	E	q_2
B	F	q_3
B	G	q_4
B	H	q_5

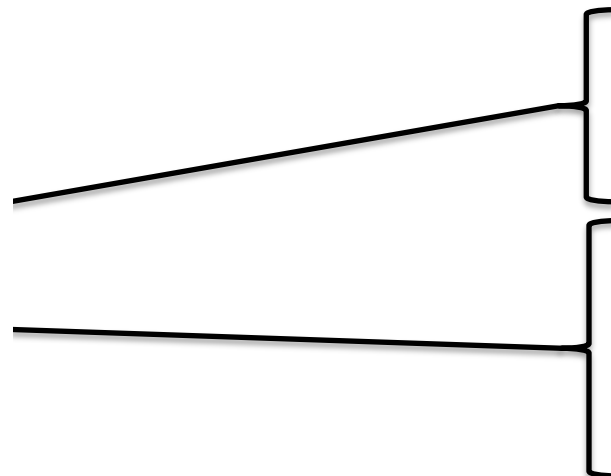
$$Q = \exists x \exists y \text{ Smoker}(x) \wedge \text{Friend}(x,y)$$

An Example

$$P(Q \mid D) = 1 - \{1 - p_1 * [1 - (1 - q_1) * (1 - q_2)]\} * \{1 - p_2 * [1 - (1 - q_3) * (1 - q_4) * (1 - q_5)]\}$$

Friend

x	y	P
A	D	q_1
A	E	q_2
B	F	q_3
B	G	q_4
B	H	q_5



Smoker

x	P
A	p_1
B	p_2
C	p_3

$$Q = \exists x \exists y \text{ Smoker}(x) \wedge \text{Friend}(x,y)$$

An Example

$$P(Q \mid D) = 1 - \{1 - p_1 * [1 - (1 - q_1) * (1 - q_2)]\} * \\ \{1 - p_2 * [1 - (1 - q_3) * (1 - q_4) * (1 - q_5)]\}$$

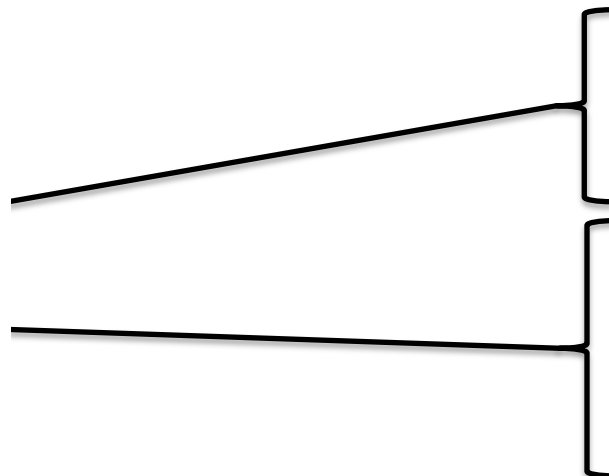
One can compute $P(Q \mid D)$ in PTIME
in the size of the database D

Friend

x	y	P
A	D	q_1
A	E	q_2
B	F	q_3
B	G	q_4
B	H	q_5

Smoker

x	P
A	p_1
B	p_2
C	p_3

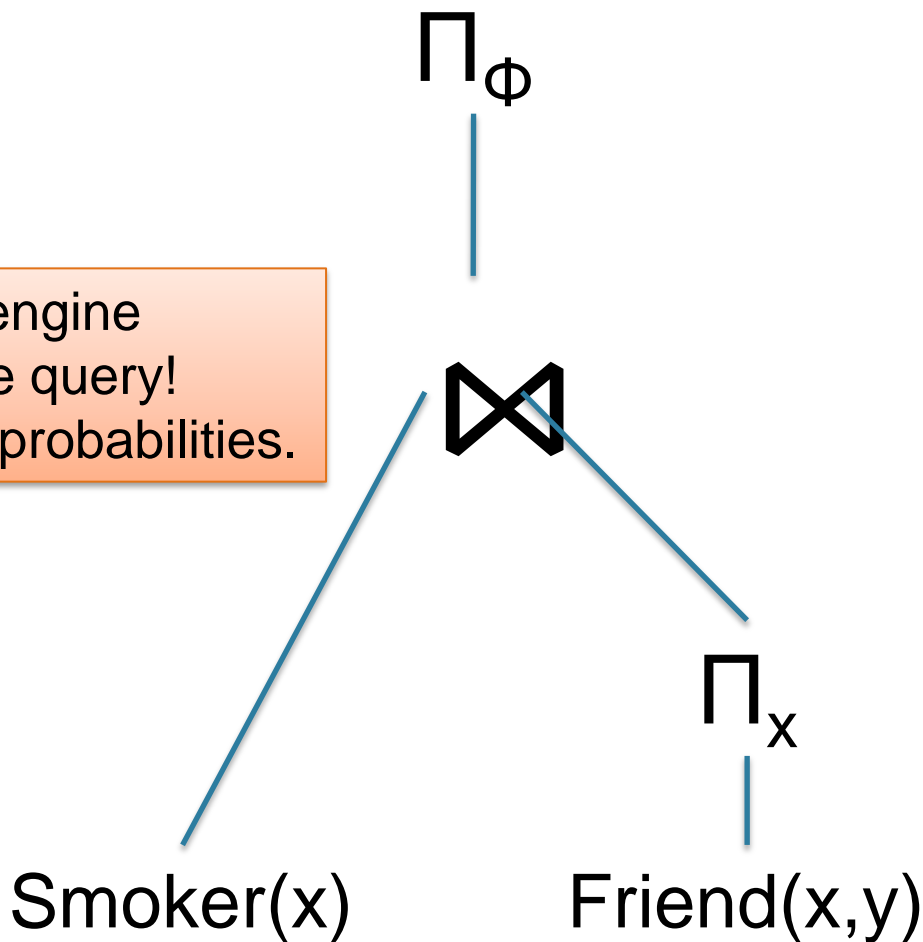


$$Q = \exists x \exists y \text{ Smoker}(x) \wedge \text{Friend}(x,y)$$

An Example

Use the SQL engine to compute the query!
Aggregate on probabilities.

x	P
A	p_1
B	p_2
C	p_3



x	y	P
A	D	q_1
A	E	q_2
B	F	q_3
B	G	q_4
B	H	q_5

$$Q = \exists x \exists y \text{ Smoker}(x) \wedge \text{Friend}(x,y)$$

An Example

Use the SQL engine to compute the query!
Aggregate on probabilities.

x	P
A	p_1
B	p_2
C	p_3

Smoker(x)

Π_{ϕ}



x	P
A	$1-(1-q_1)(1-q_2)$
B	$1-(1-q_4)(1-q_5)(1-q_6)$

Π_x

Friend(x,y)

x	y	P
A	D	q_1
A	E	q_2
B	F	q_3
B	G	q_4
B	H	q_5

$$Q = \exists x \exists y \text{ Smoker}(x) \wedge \text{Friend}(x,y)$$

An Example

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

Use the SQL engine to compute the query!
Aggregate on probabilities.

x	P
A	p_1
B	p_2
C	p_3

Smoker(x)

Π_{ϕ}



x	P
A	$1 - (1 - q_1)(1 - q_2)$
B	$1 - (1 - q_4)(1 - q_5)(1 - q_6)$

Π_x

Friend(x,y)

x	y	P
A	D	q_1
A	E	q_2
B	F	q_3
B	G	q_4
B	H	q_5

Problem Statement

Given: probabilistic database D , query Q

Compute: $P(Q \mid D)$

Data complexity: fix Q , complexity = $f(|D|)$

Approaches to Compute $P(Q \mid D)$

- Propositional inference:
 - Ground the query $Q \rightarrow F_{Q,D}$, compute $P(F_{Q,D})$
 - This is **Weighted Model Counting** (later...)
 - Works for every query Q
 - But: may be exponential in $|D|$ (data complexity)
- Lifted inference:
 - Compute a query plan for Q , execute plan on D
 - Always polynomial time in $|D|$ (data complexity)
 - But: does not work for all queries Q

Lifted Inference Rules

Preprocess Q (omitted from this talk; see [Suciu'11]),
then apply these rules (some have preconditions)

$$P(\neg Q) = 1 - P(Q) \quad \text{negation}$$

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

Independent
join / union

$$P(\exists z Q) = 1 - \prod_{A \in \text{Domain}} (1 - P(Q[A/z]))$$
$$P(\forall z Q) = \prod_{A \in \text{Domain}} P(Q[A/z])$$

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

Example

$$Q = \forall x \forall y (\text{Smoker}(x) \vee \text{Friend}(x,y))$$

$$= \forall x (\text{Smoker}(x) \vee \forall y \text{Friend}(x,y))$$

∇-Rule

$$P(Q) = \prod_{A \in \text{Domain}} P(\text{Smoker}(A) \vee \forall y \text{Friend}(A,y))$$

- Check independence:
Smoker(Alice) ∨ ∇y Friend(Alice,y)
Smoker(Bob) ∨ ∇y Friend(Bob,y)

Example

$$Q = \forall x \forall y (\text{Smoker}(x) \vee \text{Friend}(x,y))$$

$$= \forall x (\text{Smoker}(x) \vee \forall y \text{Friend}(x,y))$$

∇-Rule

$$P(Q) = \prod_{A \in \text{Domain}} P(\text{Smoker}(A) \vee \forall y \text{Friend}(A,y))$$

• Check independence:
Smoker(Alice) ∨ ∇y Friend(Alice,y)
Smoker(Bob) ∨ ∇y Friend(Bob,y)

$$P(Q) = \prod_{A \in \text{Domain}} [1 - (1 - P(\text{Smoker}(A))) \times (1 - P(\forall y \text{Friend}(A,y)))]$$

∇-Rule

Example

$$Q = \forall x \forall y (\text{Smoker}(x) \vee \text{Friend}(x,y))$$

$$= \forall x (\text{Smoker}(x) \vee \forall y \text{Friend}(x,y))$$

∇-Rule

$$P(Q) = \prod_{A \in \text{Domain}} P(\text{Smoker}(A) \vee \forall y \text{Friend}(A,y))$$

Check independence:
Smoker(Alice) ∨ ∇y Friend(Alice,y)
Smoker(Bob) ∨ ∇y Friend(Bob,y)

$$P(Q) = \prod_{A \in \text{Domain}} [1 - (1 - P(\text{Smoker}(A))) \times (1 - P(\forall y \text{Friend}(A,y)))]$$

∇-Rule

$$P(Q) = \prod_{A \in \text{Domain}} [1 - (1 - P(\text{Smoker}(A))) \times (1 - \prod_{B \in \text{Domain}} P(\text{Friend}(A,B)))]$$

∇-Rule

Example

$$Q = \forall x \forall y (\text{Smoker}(x) \vee \text{Friend}(x,y))$$

$$= \forall x (\text{Smoker}(x) \vee \forall y \text{Friend}(x,y))$$

∇-Rule

$$P(Q) = \prod_{A \in \text{Domain}} P(\text{Smoker}(A) \vee \forall y \text{Friend}(A,y))$$

Check independence:
Smoker(Alice) ∨ ∇y Friend(Alice,y)
Smoker(Bob) ∨ ∇y Friend(Bob,y)

$$P(Q) = \prod_{A \in \text{Domain}} [1 - (1 - P(\text{Smoker}(A))) \times (1 - P(\forall y \text{Friend}(A,y)))]$$

∇-Rule

$$P(Q) = \prod_{A \in \text{Domain}} [1 - (1 - P(\text{Smoker}(A))) \times (1 - \prod_{B \in \text{Domain}} P(\text{Friend}(A,B)))]$$

Lookup the probabilities
in the database

∇-Rule

Example

$$Q = \forall x \forall y (\text{Smoker}(x) \vee \text{Friend}(x,y))$$

$$= \forall x (\text{Smoker}(x) \vee \forall y \text{Friend}(x,y))$$

∇-Rule

$$P(Q) = \prod_{A \in \text{Domain}} P(\text{Smoker}(A) \vee \forall y \text{Friend}(A,y))$$

Check independence:
Smoker(Alice) ∇ ∇y Friend(Alice,y)
Smoker(Bob) ∇ ∇y Friend(Bob,y)

$$P(Q) = \prod_{A \in \text{Domain}} [1 - (1 - P(\text{Smoker}(A))) \times (1 - P(\forall y \text{Friend}(A,y)))]$$

∇-Rule

$$P(Q) = \prod_{A \in \text{Domain}} [1 - (1 - P(\text{Smoker}(A))) \times (1 - \prod_{B \in \text{Domain}} P(\text{Friend}(A,B)))]$$

Lookup the probabilities
in the database

∇-Rule

Runtime = $O(n^2)$.

Discussion: CNF vs. DNF

Databases		KR/AI	
Conjunctive Queries CQ	$FO(\exists, \wedge)$	Positive Clause	$FO(\forall, \vee)$
Union of Conjunctive Queries UCQ	$FO(\exists, \wedge, \vee) = \exists$ Positive-DNF	Positive FO	$FO(\forall, \wedge, \vee) = \forall$ Positive-CNF
UCQ with “safe negation” UCQ⁻	\exists DNF	First Order CNF	\forall CNF
$Q = \exists x, \exists y, \text{Smoker}(x) \wedge \text{Friend}(x, y)$		$Q = \forall x \forall y (\text{Smoker}(x) \vee \text{Friend}(x, y))$	

$$\exists x, \exists y, \text{Smoker}(x) \wedge \text{Friend}(x, y) = \neg \forall x, \forall y, (\neg \text{Smoker}(x) \vee \neg \text{Friend}(x, y))$$

Discussion

Lifted Inference Sometimes Fails.


$$H_0 = \forall x \forall y (\text{Smoker}(x) \vee \text{Friend}(x,y) \vee \text{Jogger}(y))$$

The \forall -rule does not apply: $H_0[\text{Alice}/x]$ and $H_0[\text{Bob}/x]$ are dependent:

$$H_0[\text{Alice}/x] = \forall y (\text{Smoker}(\text{Alice}) \vee \text{Friend}(\text{Alice},y) \vee \text{Jogger}(y))$$

$$H_0[\text{Bob}/x] = \forall y (\text{Smoker}(\text{Bob}) \vee \text{Friend}(\text{Bob},y) \vee \text{Jogger}(y))$$

Computing $P(H_0 \mid D)$ is #P-hard in $|D|$
(Proof: later...)



Dependent

Discussion

Lifted Inference Sometimes Fails.


$$H_0 = \forall x \forall y (\text{Smoker}(x) \vee \text{Friend}(x,y) \vee \text{Jogger}(y))$$

The \forall -rule does not apply: $H_0[\text{Alice}/x]$ and $H_0[\text{Bob}/x]$ are dependent:

$$H_0[\text{Alice}/x] = \forall y (\text{Smoker}(\text{Alice}) \vee \text{Friend}(\text{Alice},y) \vee \text{Jogger}(y))$$

$$H_0[\text{Bob}/x] = \forall y (\text{Smoker}(\text{Bob}) \vee \text{Friend}(\text{Bob},y) \vee \text{Jogger}(y))$$

Computing $P(H_0 \mid D)$ is #P-hard in $|D|$
(Proof: later...)



Dependent

Consequence: assuming $\text{PTIME} \neq \text{\#P}$, H_0 is not liftable!

Summary

- Database D = relations
- Query Q = FO
- Query plans, query optimization
- Data complexity: fix Q , complexity $f(D)$
- Probabilistic DB's = independent tuples
- Lifted inference: simple, but fails sometimes