The Complexity Landscape of Probabilistic Query Evaluation

İsmail İlkan Ceylan

17.03.2022

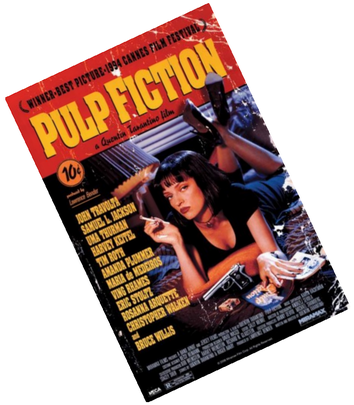
GraphIK, Montpellier

Overview

- Probabilistic databases: a gentle introduction
  - The complexity of probabilistic query evaluation
  - The dichotomy for unions of conjunctive queries over probabilistic databases
  - Weighted model counting and approximability
- Ontology-mediated queries over probabilistic data
  - Hardness of a simple unbounded query
  - Homomorphism-closed queries: Datalog, RPQs, OMQs
  - The dichotomy for homomorphism-closed queries over probabilistic graphs
- Open problems, challenges, outlook

Probabilistic Databases

Probabilistic Databases



StarredIn

deNiro	taxiDriver
foster	taxiDriver
thurman	pulpFiction
travolta	pulpFiction

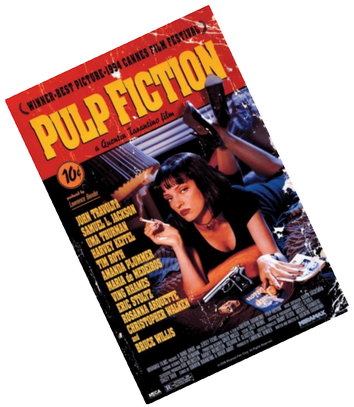
DirectedBy

pulpFiction	tarantino
taxiDriver	scorsese
whiteRibbon	haneke
winterSleep	ceylan

$$\exists x, y \, \text{StarredIn}(x, y) \wedge \text{DirectedBy}(y, \text{tarantino})$$

true - false

Probabilistic Databases



StarredIn		P
deNiro	taxiDriver	0.7
foster	taxiDriver	0.2
thurman	pulpFiction	0.1
travolta	pulpFiction	0.3

DirectedBy		P
pulpFiction	tarantino	0.8
taxiDriver	scorsese	0.6
whiteRibbon	haneke	0.7
winterSleep	ceylan	0.8

$$\exists x, y\ \text{StarredIn}(x, y) \wedge \text{DirectedBy}(y, \text{tarantino})$$

$$0.296$$

Possible Worlds

StarredIn		P
deNiro	taxiDriver	0.7
foster	taxiDriver	0.2
thurman	pulpFiction	0.1
travolta	pulpFiction	0.3

DirectedBy		P
pulpFiction	tarantino	0.8
taxiDriver	scorsese	0.6
whiteRibbon	haneke	0.7
winterSleep	ceylan	0.8

A PDB compactly encodes a set of possible worlds (i.e., classical databases):

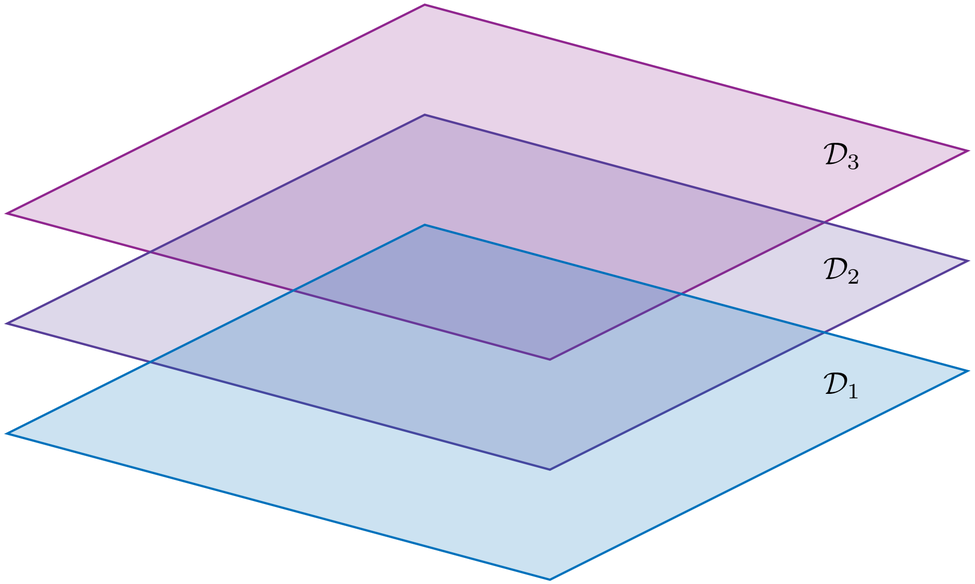
$$D_1 = \{\text{StarredIn}(\text{deNiro}, \text{taxiDriver}), \text{StarredIn}(\text{foster}, \text{taxiDriver})\}$$

A (tuple-independent) PDB defines a probability distribution over the possible worlds D:

$$P(D) = \prod_{t \in D} P(t) \prod_{t \notin D} (1 - P(t))$$

$$P(D_1) = 0.7 \cdot 0.2 \cdot (1 - 0.1) \cdot (1 - 0.3) \cdot (1 - 0.8) \cdots$$

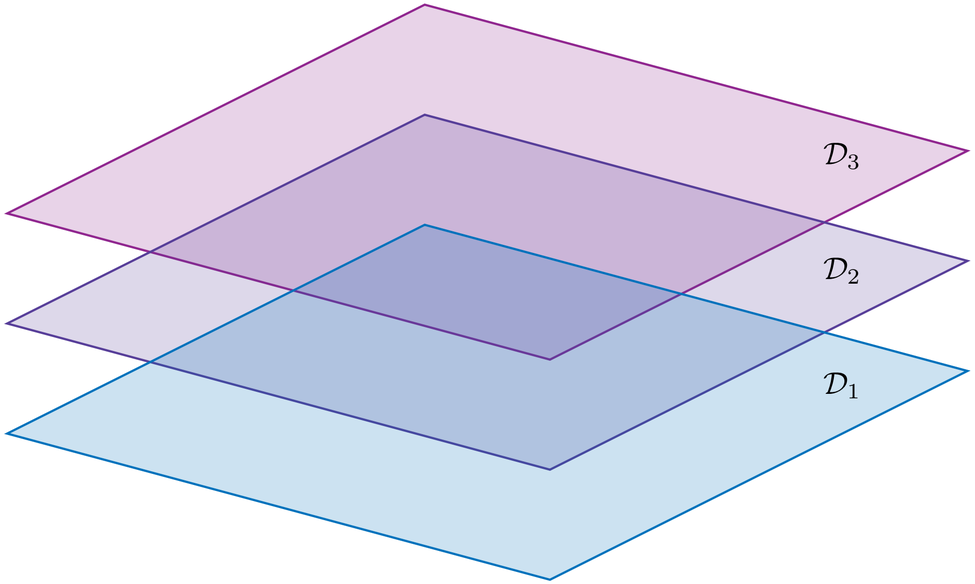
Query Evaluation in Probabilistic Databases



$$P(D) = \prod_{t \in D} P(t) \prod_{t \notin D} (1 - P(t))$$

$$P(Q) = \sum_{D \models Q} P(D)$$

Query Evaluation in Probabilistic Databases



$$P(D) = \prod_{t \in D} P(t) \prod_{t \notin D} (1 - P(t))$$

$$P(Q) = \sum_{D \models Q} P(D)$$

Possible world semantics:

- Views every PDB fact as an independent random variable.
- Closed-world: only accounts for the facts in the PDB, everything else has probability 0!
- Computationally demanding nevertheless - exponentially many possible worlds!

Probabilistic Query Evaluation

Problem: Probabilistic query evaluation

Input: A PDB and a Boolean query Q

Output: $P(Q)$

How Hard is Probabilistic Query Evaluation?

Actors	P
deNiro	0.9
foster	0.8
thurman	0.7
travolta	1

StarredIn		P
deNiro	taxiDriver	0.7
foster	taxiDriver	0.2
thurman	pulpFiction	0.1
travolta	pulpFiction	0.3

Movie	P
pulpFiction	0.8
taxiDriver	0.6
whiteRibbon	0.7
winterSleep	0.8

Our focus is on data complexity: PQE(Q) for some fixed query Q.

$$Q_1 := \exists\, x, y\, \mathsf{StarredIn}(x, y) \wedge \mathsf{Movie}(y)$$

$$Q_2 := \exists\, x, y\, \mathsf{Actor}(x) \wedge \mathsf{StarredIn}(x, y) \wedge \mathsf{Movie}(y)$$

Computing $P(Q_1)$ is easy on any PDB, whereas computing $P(Q_2)$ is hard!

Complexity Background

Complexity Background

#P: Class of function problems recognized by a poly-bounded non-deterministic TM that outputs the number of accepting computation paths.

Complexity Background

#P: Class of function problems recognized by a poly-bounded non-deterministic TM that outputs the number of accepting computation paths.

#SAT is a canonical #P-complete problem:

Complexity Background

#P: Class of function problems recognized by a poly-bounded non-deterministic TM that outputs the number of accepting computation paths.

#SAT is a canonical #P-complete problem:

Problem: #SAT

Input: A propositional formula ϕ

Output: The number of satisfying assignments, i.e., $\#\phi$.

Complexity Background

#P: Class of function problems recognized by a poly-bounded non-deterministic TM that outputs the number of accepting computation paths.

#SAT is a canonical #P-complete problem:

Problem: #SAT

Input: A propositional formula ϕ

Output: The number of satisfying assignments, i.e., $\#\phi$.

#SAT remains #P-hard problem even for restricted fragments of propositional formulas.

Complexity Background

Complexity Background

PP: Class of languages recognized by a poly-bounded non-deterministic TM that that accepts an input iff *more than half* of the computation paths do so.

Complexity Background

PP: Class of languages recognized by a poly-bounded non-deterministic TM that that accepts an input iff *more than half* of the computation paths do so.

MAJSAT a canonical PP-complete problem:

Complexity Background

PP: Class of languages recognized by a poly-bounded non-deterministic TM that that accepts an input iff *more than half* of the computation paths do so.

MAJSAT a canonical PP-complete problem:

Problem: MAJSAT

Input: A propositional formula ϕ

Output: Is the number of satisfying assignments to ϕ more than half?

Complexity Background

PP: Class of languages recognized by a poly-bounded non-deterministic TM that that accepts an input iff *more than half* of the computation paths do so.

MAJSAT a canonical PP-complete problem:

Problem: MAJSAT

Input: A propositional formula ϕ

Output: Is the number of satisfying assignments to ϕ more than half?

The relation of these classes to well-known classes:

$$\mathsf{P} \subseteq \mathsf{NP} \subseteq \mathsf{PP}, \mathsf{PH} \subseteq \mathsf{P}^{\mathsf{PP}} = \mathsf{P}^{\#\mathsf{P}} \subseteq \mathsf{NP}^{\mathsf{PP}} \subseteq \mathsf{PSpace}$$

How Hard is Probabilistic Query Evaluation?

Actors	P
deNiro	0.9
foster	0.8
thurman	0.7
travolta	1

StarredIn		P
deNiro	taxiDriver	0.7
foster	taxiDriver	0.2
thurman	pulpFiction	0.1
travolta	pulpFiction	0.3

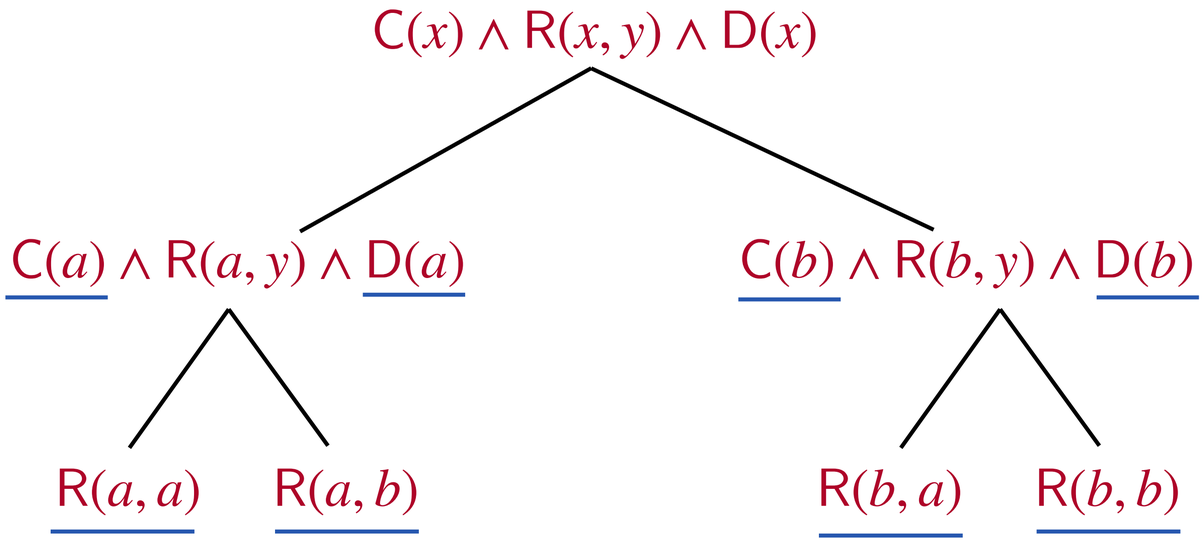
Movie	P
pulpFiction	0.8
taxiDriver	0.6
whiteRibbon	0.7
winterSleep	0.8

$$Q_h := \exists\, x, y\, \mathsf{C}(x) \wedge \mathsf{R}(x, y) \wedge \mathsf{D}(x)$$

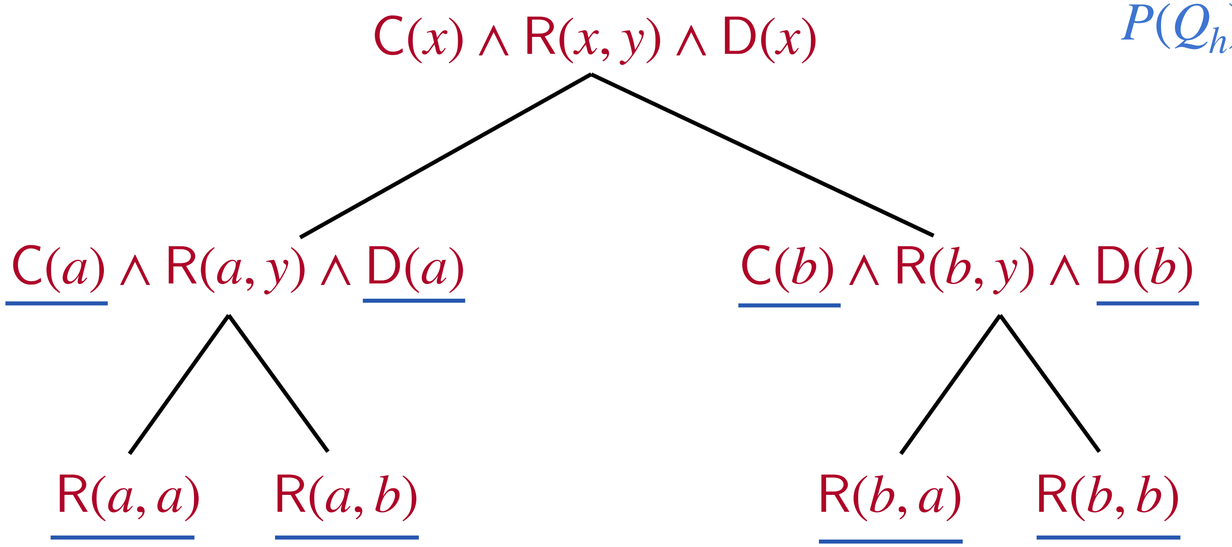
$$Q_{nh} := \exists\, x, y\, \mathsf{C}(x) \wedge \mathsf{R}(x, y) \wedge \mathsf{D}(y)$$

$P(Q_h)$ can be computed in polynomial time, whereas computing $P(Q_{nh})$ is #P-hard!

How Hard is Probabilistic Query Evaluation?



How Hard is Probabilistic Query Evaluation?

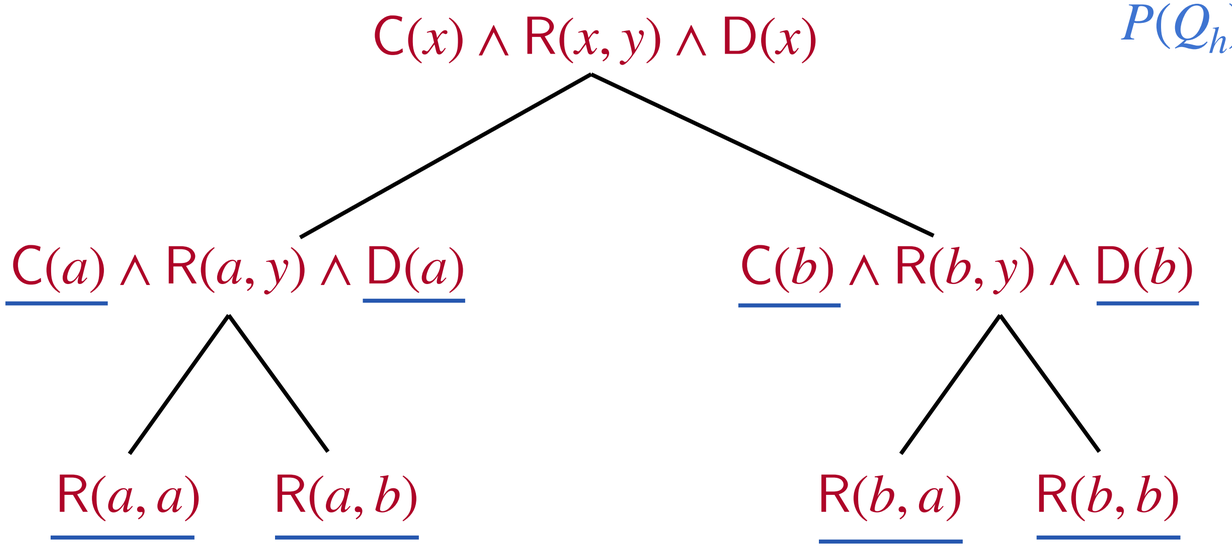


$$P(Q_h) = 1 - \prod_{u \in U} (1 - P(\exists y\, C(u) \wedge R(u,y) \wedge D(u)))$$

$$P(C(u)) \cdot P(\exists y\, R(u,y)) \cdot P(D(u))$$

$$1 - \prod_{v \in U} (1 - P(R(u,v)))$$

How Hard is Probabilistic Query Evaluation?



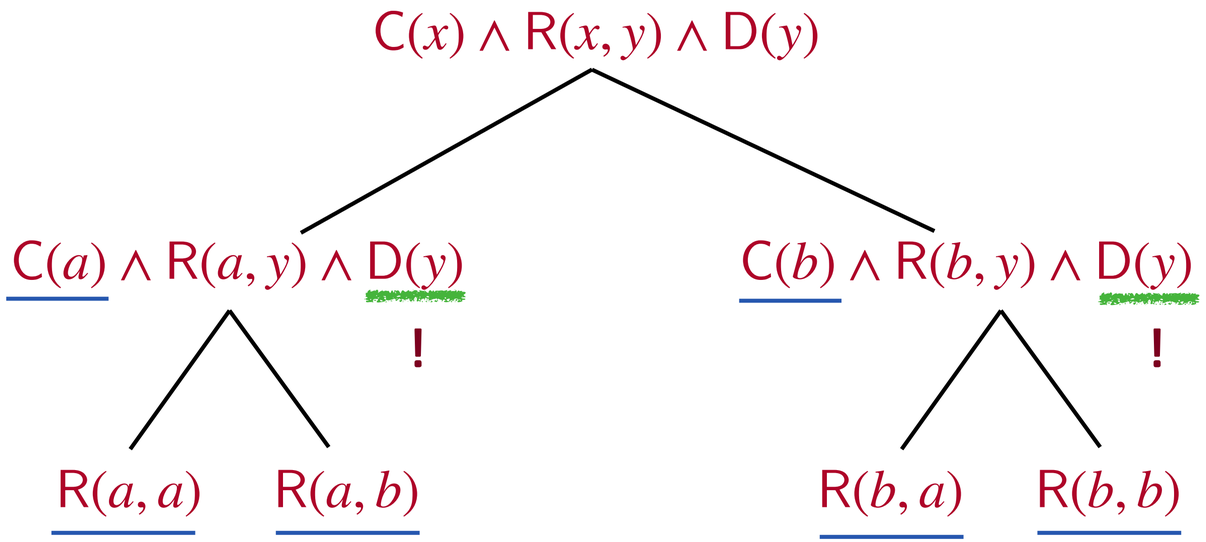
$$P(Q_h) = 1 - \prod_{u \in U} (1 - P(\exists y\, C(u) \wedge R(u,y) \wedge D(u)))$$

$$P(C(u)) \cdot P(\exists y\, R(u,y)) \cdot P(D(u))$$

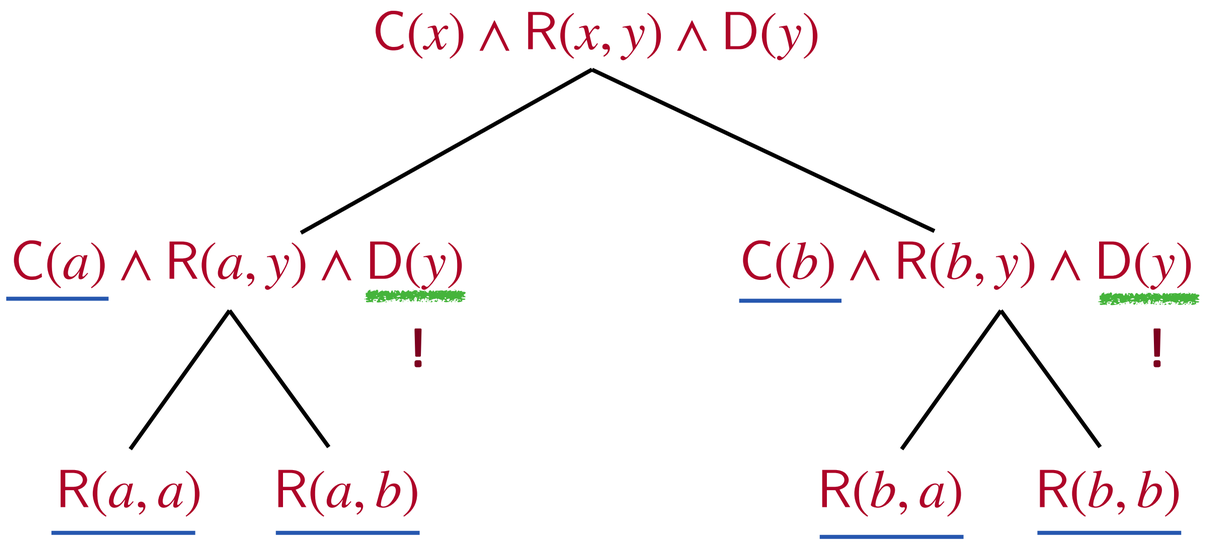
$$1 - \prod_{v \in U} (1 - P(R(u,v)))$$

$P(Q_h)$ can be computed in polynomial time, but what makes computing $P(Q_{nh})$ hard?

How Hard is Probabilistic Query Evaluation?

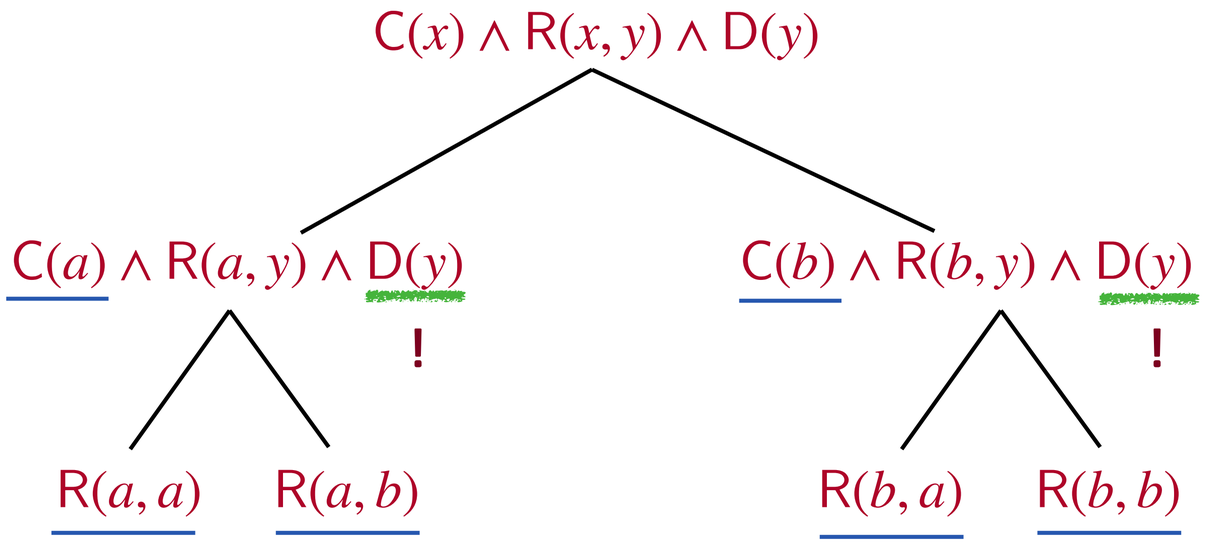


How Hard is Probabilistic Query Evaluation?



Different groundings *share* same D-atoms!

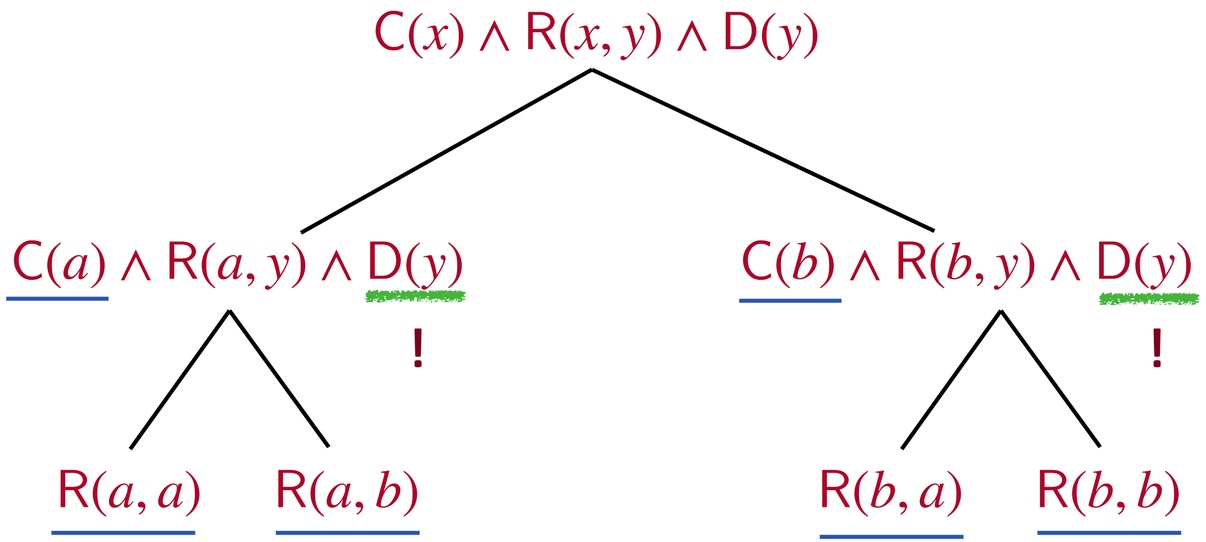
How Hard is Probabilistic Query Evaluation?



Different groundings *share* same D-atoms!

Sub-queries are NOT independent!

How Hard is Probabilistic Query Evaluation?



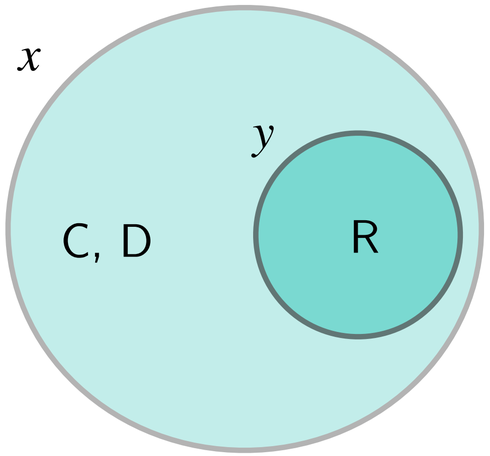
Different groundings *share* same D-atoms!

Sub-queries are NOT independent!

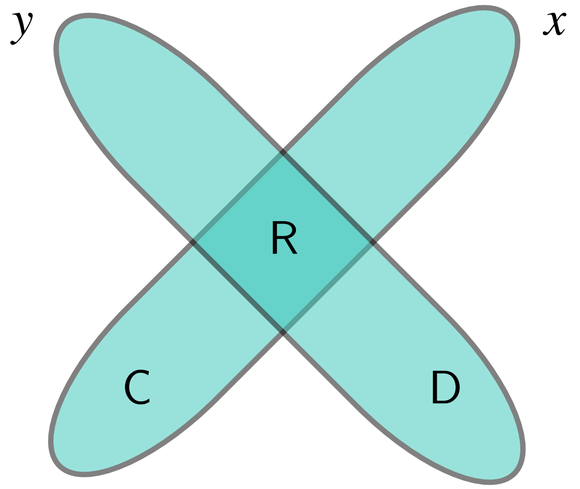
Clash!

How Hard is Probabilistic Query Evaluation?

$Q_h := \exists x, y\, \mathsf{C}(x) \wedge \mathsf{R}(x, y) \wedge \mathsf{D}(x)$

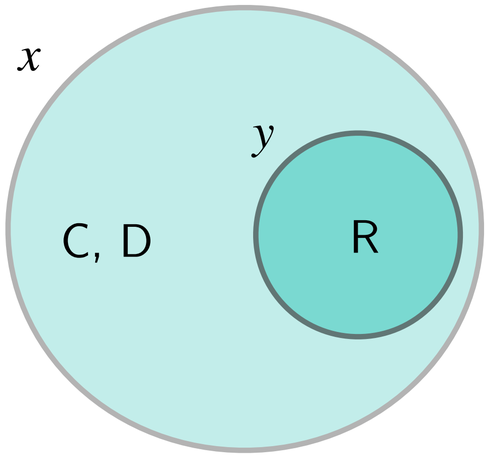


$Q_{nh} := \exists x, y\, \mathsf{C}(x) \wedge \mathsf{R}(x, y) \wedge \mathsf{D}(y)$

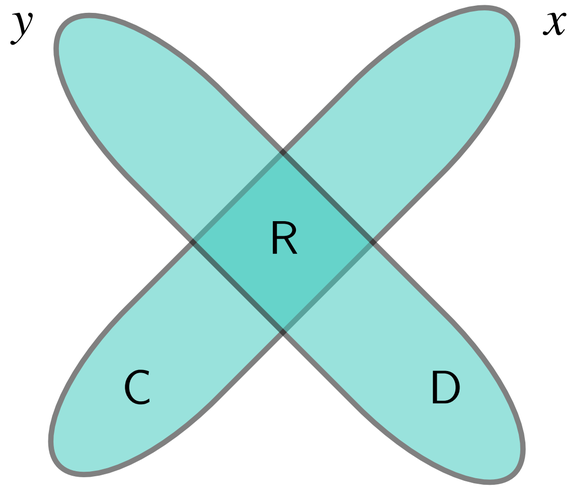


How Hard is Probabilistic Query Evaluation?

$Q_h := \exists x, y\, \mathsf{C}(x) \wedge \mathsf{R}(x, y) \wedge \mathsf{D}(x)$



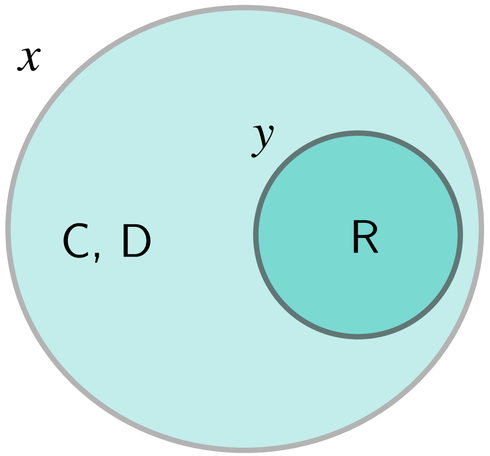
$Q_{nh} := \exists x, y\, \mathsf{C}(x) \wedge \mathsf{R}(x, y) \wedge \mathsf{D}(y)$



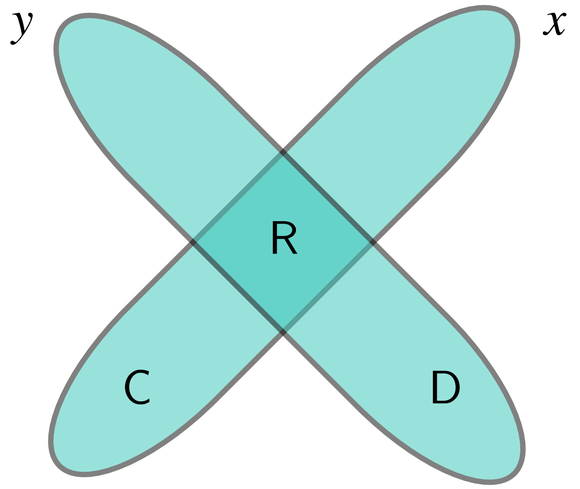
Hierarchical: either the covers of x, y do not intersect, or if they do, one is contained in the other.

How Hard is Probabilistic Query Evaluation?

$Q_h := \exists x, y\, \mathsf{C}(x) \wedge \mathsf{R}(x, y) \wedge \mathsf{D}(x)$



$Q_{nh} := \exists x, y\, \mathsf{C}(x) \wedge \mathsf{R}(x, y) \wedge \mathsf{D}(y)$



Hierarchical: either the covers of x, y do not intersect, or if they do, one is contained in the other.

Separator variable: for hierarchical queries, there is a seperator variable, decomposing the query into independent subqueries.

A Canonical Hardness Result

PQE (Q_{nh}) is #P-hard (unsafe).

A Canonical Hardness Result

PQE (Q_{nh}) is #P-hard (unsafe).

Proof idea. We have the fixed query

$$Q_{nh} := \exists\, x, y\, \mathsf{C}(x) \wedge \mathsf{R}(x, y) \wedge \mathsf{D}(x)$$

...and need to show #P-hardness.

Just like #SAT, #DNF is also #P-hard, since $\#\phi = 2^n - \#(\neg\phi)$

Counting the satisfying assignments of a bipartite monotone 2DNF is hard:

$$\phi = \bigvee_{1 \le j \le m, 1 \le k \le n} x_j \wedge y_k$$

A Canonical Hardness Result

PQE (Q_{nh}) is #P-hard (unsafe).

Proof sketch: Reduce from $\#\phi$, where $\phi = \bigvee_{1 \le j \le m, 1 \le k \le n} x_j \wedge y_k$. Define a PDB:

- For x variables, add $\langle C(x_1) : 0.5 \rangle, \ldots, \langle C(x_m) : 0.5 \rangle$
- For y variables, add $\langle D(y_1) : 0.5 \rangle, \ldots, \langle D(y_n) : 0.5 \rangle$
- For the clauses, add $\langle R(x_j, y_k) : 1 \rangle$ for every clause $(x_j \wedge y_k)$

Each world encodes an assignment of ϕ and has probability 0.5^{m+n}.

It is easy to verify that $\#\phi = P(Q_{nh}) \cdot 2^{m+n}$.

The Dichotomy of Probabilistic Query Evaluation

The Dichotomy of Probabilistic Query Evaluation

Theorem (Small dichotomy): Let Q be any conjunctive query without self-joins: Either Q is hierarchical and hence is safe and PQE(Q) is in PTIME; otherwise it is non-hierarchical, and hence it is unsafe and PQE(Q) is #P-hard.

The Dichotomy of Probabilistic Query Evaluation

Theorem (Small dichotomy): Let Q be any conjunctive query without self-joins: Either Q is hierarchical and hence is safe and PQE(Q) is in PTIME; otherwise it is non-hierarchical, and hence it is unsafe and PQE(Q) is #P-hard.

Theorem [**Dichotomy**, (Dalvi and Suciu, 2013)] For every UCQ query Q, it is either safe and PQE(Q) is in PTIME, or it is unsafe and PQE(Q) is #P-hard.

This result is recently strengthened to PDBs where all facts have probabilities in $\{0,0.5,1\}$: Unsafe queries remain unsafe even under this restriction (Kenig and Suciu, 2021)!

The Dichotomy of Probabilistic Query Evaluation

Theorem (Small dichotomy): Let Q be any conjunctive query without self-joins: Either Q is hierarchical and hence is safe and PQE(Q) is in PTIME; otherwise it is non-hierarchical, and hence it is unsafe and PQE(Q) is #P-hard.

Theorem [**Dichotomy**, (Dalvi and Suciu, 2013)] For every UCQ query Q, it is either safe and PQE(Q) is in PTIME, or it is unsafe and PQE(Q) is #P-hard.

This result is recently strengthened to PDBs where all facts have probabilities in $\{0,0.5,1\}$: Unsafe queries remain unsafe even under this restriction (Kenig and Suciu, 2021)!

Effectiveness: This dichotomy is effective, i.e., there exists a (super-exponential) algorithm which can determine whether a query is safe or unsafe. Exact complexity remains open.

Weighted Model Counting

Problem: WMC

Input: A propositional formula ϕ and a weight function $w : \mathscr{A} \to \mathbb{R}$

Output: The total weight of satisfying assignments to ϕ, given by $\sum_{\nu \models \phi} w(\nu)$.

Weighted Model Counting

Weighted Model Counting

Problem: WMC

Input: A propositional formula ϕ and a weight function $w : \mathscr{A} \to \mathbb{R}$

Output: The total weight of satisfying assignments to ϕ, given by $\sum_{\nu \vDash \phi} w(\nu)$.

Weighted Model Counting

Problem: WMC

Input: A propositional formula ϕ and a weight function $w : \mathscr{A} \to \mathbb{R}$

Output: The total weight of satisfying assignments to ϕ, given by $\sum_{\nu \vDash \phi} w(\nu)$.

We can establish a direct correspondence between WMC and PDBs.

Define the weight function as $w : \mathscr{A} \to [0,1]$ and assume that the weight function factorises as:

$$w(\nu) = \prod_{l \in \nu} w(l) \prod_{\neg l \in \nu} (1 - w(l))$$

A propositional literal corresponds to a probabilistic fact in the PDB; a propositional assignment corresponds to a possible world of the PDB.

Weighted Model Counting

Problem: WMC

Input: A propositional formula ϕ and a weight function $w : \mathscr{A} \to \mathbb{R}$

Output: The total weight of satisfying assignments to ϕ, given by $\sum_{\nu \vDash \phi} w(\nu)$.

Each query has a lineage (or provenance) representation:

$Q = \exists\, x, y\, \mathsf{C}(x) \wedge \mathsf{R}(x, y)$ can be written as $\phi_Q = \bigvee_{u,v \in U} \big(\mathsf{C}(u) \wedge \mathsf{R}(u, v)\big)$

This is a (positive) formula in DNF and the transformation is efficient in data complexity.

Probabilistic UCQ evaluation is essentially weighted DNF counting!

Approximability of Probabilistic Query Evaluation

Approximability of Probabilistic Query Evaluation

Randomized algorithm: Let $f : \mathscr{L} \to \mathbb{R}$ be a function from a class of formulas to weights. Given $\epsilon, \delta > 0$, a randomized algorithm $\hat{f}$ is called an (ϵ, δ)-approximation of f if for all $\phi \in \mathscr{L}$:

$$P(\,|\,\hat{f}(\phi) - f(\phi)\,|\,\leq \epsilon) \geq 1 - \delta\,.$$

Approximability of Probabilistic Query Evaluation

Randomized algorithm: Let $f : \mathscr{L} \to \mathbb{R}$ be a function from a class of formulas to weights. Given $\epsilon, \delta > 0$, a randomized algorithm $\hat{f}$ is called an (ϵ, δ)-approximation of f if for all $\phi \in \mathscr{L}$:

$$P(\,|\,\hat{f}(\phi) - f(\phi)\,|\,\leq \epsilon) \geq 1 - \delta\,.$$

FRPAS: An (ϵ, δ)-approximation algorithm $\hat{f}$ is called a fully polynomial approximation scheme if the running time of algorithm $\hat{f}$ is polynomial in $|\phi|$, ϵ^{-1}, $\log(\delta^{-1})$.

Approximability of Probabilistic Query Evaluation

Randomized algorithm: Let $f : \mathscr{L} \to \mathbb{R}$ be a function from a class of formulas to weights. Given $\epsilon, \delta > 0$, a randomized algorithm $\hat{f}$ is called an (ϵ, δ)-approximation of f if for all $\phi \in \mathscr{L}$:

$$P(\,|\,\hat{f}(\phi) - f(\phi)\,|\,\leq \epsilon) \geq 1 - \delta\,.$$

FRPAS: An (ϵ, δ)-approximation algorithm $\hat{f}$ is called a fully polynomial approximation scheme if the running time of algorithm $\hat{f}$ is polynomial in $|\phi|$, ϵ^{-1}, $\log(\delta^{-1})$.

Approximability: Weighted DNF counting has an FPRAS (Karp, Luby, and Madras 1989) - so does PQE on UCQs!

Inapproximability: Weighted CNF counting is NP-hard to approximate (Roth, 1996).

Probabilistic OMQ Evaluation

Problem: Probabilistic query evaluation

Input: A PDB and a Boolean OMQ (Σ, Q)

Output: $P(\Sigma, Q)$

Ontology-Mediated Queries

Problem: Query evaluation

Input: A database D, a Boolean query Q

Question: $D \models Q$?

Problem: OMQA

Input: A database D, a query Q, and an ontology Σ

Question: $(\Sigma, D) \models Q$?

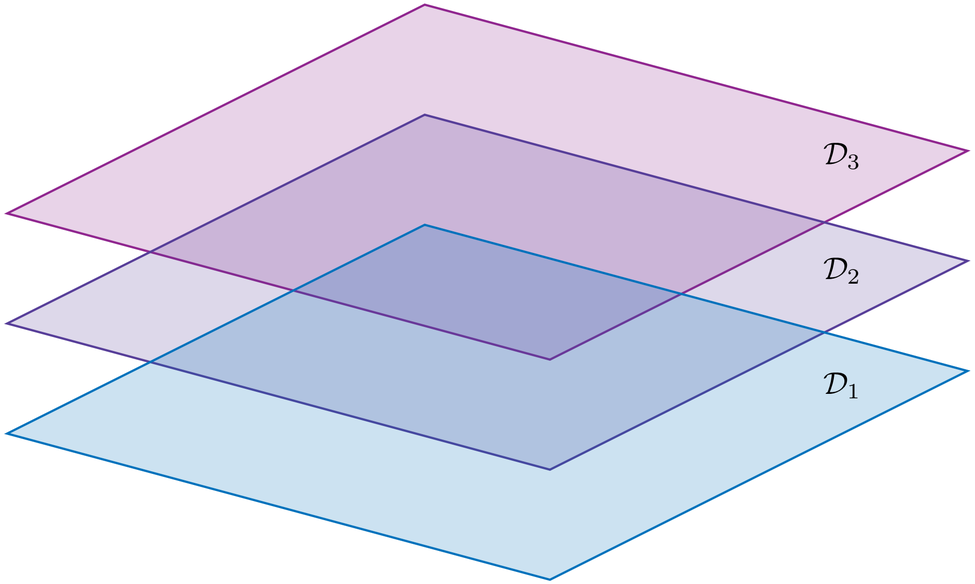
Query evaluation is model checking whereas OMQA evaluation is reasoning.

Semantics: $(\Sigma, D) \models Q$ iff for all I where $I \models \Sigma$ and $I \models D$, it holds that $I \models Q$.

Convention: (Σ, Q) is called an ontology-mediated query (OMQ).

Notation: $D \models (\Sigma, Q)$ is written instead of $(\Sigma, D) \models Q$.

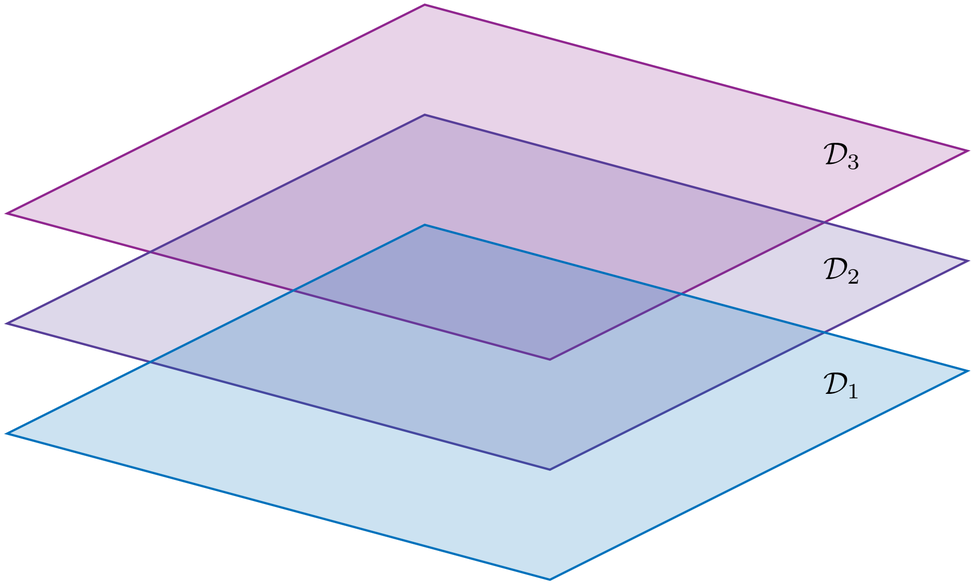
OMQ Evaluation in Probabilistic Databases



$$P(D) = \prod_{t \in D} P(t) \prod_{t \notin D} (1 - P(t))$$

$$P(\Sigma, Q) = \sum_{D \models (\Sigma, Q)} P(D)$$

OMQ Evaluation in Probabilistic Databases



$$P(D) = \prod_{t \in D} P(t) \prod_{t \notin D} (1 - P(t))$$

$$P(\Sigma, Q) = \sum_{D \models (\Sigma, Q)} P(D)$$

The use of an ontology makes a difference:

- Facts alone are independent, but the rules in the ontology introduce dependencies.
- Open-world: Positive probability also for some facts not in the PDB.
- Computationally more demanding!

Probabilistic Ontology-Mediated Query Answering

Q $\exists x, y\, \mathsf{Student}(x) \wedge \mathsf{Teaches}(x, y) \wedge \mathsf{Course}(y)$

Σ $\mathsf{GradStudent}(x) \rightarrow \mathsf{Student}(x)$

$\mathsf{Student}(x) \rightarrow \exists z\, \mathsf{HasMentor}(x, z)$

D $\mathsf{Lecturer}(\mathsf{stefan}) : 0.5,\ \mathsf{GradStudent}(\mathsf{philippo}) : 0.7,$

$\mathsf{Course}(\mathsf{logic}) : 0.8, \mathsf{Teaches}(\mathsf{philippo}, \mathsf{logic}) : 0.4$

Probabilistic Ontology-Mediated Query Answering

$$Q \quad \exists x, y\, \mathsf{Student}(x) \wedge \mathsf{Teaches}(x, y) \wedge \mathsf{Course}(y)$$

$$\Sigma \quad \mathsf{GradStudent}(x) \rightarrow \mathsf{Student}(x)$$

$$\mathsf{Student}(x) \rightarrow \exists z\, \mathsf{HasMentor}(x, z)$$

$$D \quad \mathsf{Lecturer}(\mathsf{stefan}) : 0.5,\ \mathsf{GradStudent}(\mathsf{philippo}) : 0.7,$$

$$\mathsf{Course}(\mathsf{logic}) : 0.8,\ \mathsf{Teaches}(\mathsf{philippo}, \mathsf{logic}) : 0.4$$

The use of an ontology makes a difference:

- Dependence: Being a grad student implies being a student...
- Open-world: Philippo is a student with a positive probability.
- Computationally: Need to account for the entailments of the ontology!

How Hard is Probabilistic OMQ Evaluation?

How Hard is Probabilistic OMQ Evaluation?

Consider the OMQ (Q, Σ), where...

$Q \qquad \mathsf{C}(t)$

$\Sigma \qquad \mathsf{C}(x) \wedge \mathsf{R}(x, y) \rightarrow \mathsf{C}(y)$

Observation: Typical graph reachability query, i.e., a very simple recursive query.

Observation: Very little is known for queries beyond UCQs, particularly, for query languages that feature recursion.

Fact: This query is already #P-hard!

Intuition: We can encode (a probabilistic version) of source-target connectivity.

Probabilistic Ontology-Mediated Query Answering

Problem: Directed source-target reliability

Input: A directed probabilistic graph $G = (V, E)$ where each edge holds independently with probability 0.5, a source node s, and a target node t.

Output: Probability of having a directed st-path

Probabilistic Ontology-Mediated Query Answering

Problem: Directed source-target reliability

Input: A directed probabilistic graph $G = (V, E)$ where each edge holds independently with probability 0.5, a source node s, and a target node t.

Output: Probability of having a directed st-path

Proof idea: We construct the PDB as follows:

- For the source node s, add the fact $\mathsf{C}(s)$ with probability 1.
- For each edge in $(u, v) \in E$, add the fact $\mathsf{R}(u, v)$ with probability 0.5.

The probability of the graph G being st-connected is equal to $P(\Sigma, Q)$!

Can we generalize this observation to a large class of queries featuring recursion?

Homomorphism-closed Queries

Problem: Probabilistic query evaluation

Input: A PDB and an infinite unions of conjunctive queries.

Output: $P(Q)$

Homomorphism-closed Queries over Probabilistic Graphs

Homomorphism-closed query: If G satisfies Q and G has a homomorphism to G' then G' also satisfies Q.

- Generalize CQs and UCQs, but also regular path queries, Datalog, OMQs, etc.
- Do not allow for inequalities or negation
- A homomorphism-closed query is an infinite union of CQs, denoted UCQ^∞.
- A UCQ^∞ query is bounded if the union is finite (it is a UCQ), unbounded otherwise (i.e., reachability query is unbounded).
- Allows pretty wild things, e.g., "There is a path whose length is prime"

Probabilistic graphs: (Tuple-independent) probabilistic database over binary signatures, i.e., graphs with independent edge probabilities.

The Dichotomy for Homomorphism-closed Queries

Could it be the case that all unbounded queries are hard?

Theorem [Hardness, (Amarilli and Ceylan, 2022)] Let Q be an unbounded $\mathrm{UCQ}^{\neq}$ query Q over binary signatures. Then, $\mathrm{PQE}(Q)$ is #P-hard.

The Dichotomy for Homomorphism-closed Queries

Could it be the case that all unbounded queries are hard?

Theorem [Hardness, (Amarilli and Ceylan, 2022)] Let Q be an unbounded $\text{UCQ}^{\neq}$ query Q over binary signatures. Then, PQE(Q) is #P-hard.

The reduction is proceeds on two cases:

(1) unbounded queries with non-iterable edges: reduce from #PP2DNF,

(2) unbounded queries with no non-iterable edges: reduce from #U-ST-CON.

The reductions rely on model-theoretic properties, where (2) is the hard case:

- Based on the existence of certain minimal models with tight patterns.
- Minimal tight models can be used to code edges in the input graph.

The Dichotomy for Homomorphism-closed Queries

The Dichotomy for Homomorphism-closed Queries

(1) For any unbounded UCQ^{∞}, we know that $\text{PQE}(Q)$ is #P-hard.

(2) All bounded queries are UCQs, they are already classified by Dalvi and Suciu

The Dichotomy for Homomorphism-closed Queries

(1) For any unbounded UCQ$^\infty$, we know that PQE(Q) is #P-hard.

(2) All bounded queries are UCQs, they are already classified by Dalvi and Suciu

Theorem [Dichotomy, (Amarilli and Ceylan, 2022)] For every UCQ$^\infty$ query Q over binary signatures, it is either equivalent to a safe UCQ (hence bounded) and PQE(Q) is in PTIME, or it is not and PQE(Q) is #P-hard.

This result is applies to the case where all facts have probabilities in $\{0,0.5,1\}$: Unbounded queries remain unsafe even under this restriction!

The Dichotomy for Homomorphism-closed Queries

(1) For any unbounded UCQ$^\infty$, we know that PQE(Q) is #P-hard.

(2) All bounded queries are UCQs, they are already classified by Dalvi and Suciu

Theorem [Dichotomy, (Amarilli and Ceylan, 2022)] For every UCQ$^\infty$ query Q over binary signatures, it is either equivalent to a safe UCQ (hence bounded) and PQE(Q) is in PTIME, or it is not and PQE(Q) is #P-hard.

This result is applies to the case where all facts have probabilities in $\{0,0.5,1\}$: Unbounded queries remain unsafe even under this restriction!

Effectiveness: Open - To be studied for syntactically well-defined fragments. If boundedness is decidable (e.g., monadic Datalog), that gives a classification of safe/unsafe queries.

On The (In)approximability of Homomorphism-closed Queries

All bounded queries are UCQs...

These queries admit an FPRAS, as does weighted DNF counting!

What is the status of unbounded UCQ$^\infty$ queries?

Largely inapproximable

Complete characterization is lacking

Contains some well-known open problems

E.g., approximability status of #U-ST-CON is open

Outlook

Open Problems, Challenges, Outlook

Probabilistic Query Evaluation

- Dichotomy for FO queries with negation?
- Dichotomy for homomorphism-closed queries on PDBs (with higher-arity signatures)?
- Fine-grained approximability results (e.g., are all unbounded queries inapproximable)?
- Extensions to include real variables: WMI generalizes WMC with real variables. WMI on DNF structures admits an FPRAS (Abboud et al., 2020)!
- Most of the presented results hold even if we restrict probabilities to $\{0,0.5,1\}$: Do the results still hold when all probabilities are 0.5 (counting models of a query)?

Open Problems, Challenges, Outlook

Alternative Data Models and Query Languages

- Open-world probabilistic databases (Ceylan et al., 2022): allow non-zero probabilities for open facts, while assuming closed domain.
- Probabilistic databases with an infinite open-world assumption (Grohe and Lindner, 2019): allow open domains, i.e., infinite universe (domains, such as integers, reals).
- Ontology-mediated querying of data: relaxing the independence assumption, integrating common-sense knowledge, open-domain reasoning.

Open Problems, Challenges, Outlook

Problems Beyond Probabilistic Query Evaluation

- Explaining answers to queries: The most probable database problem (Ceylan et al., 2017): identify the most probable database instance, which entails the query.
- Other explainability problems (SHAP scores) relate to probabilistic query evaluation.

Systems and Implementations

- Exact approaches, largely based on knowledge compilation.
- Approached based on approximate model counting
- Systems dedicated to PDBs.

Thanks!


EPSRC
Engineering and Physical Sciences
Research Council


UNIVERSITY OF
OXFORD

References

- N. Dalvi and D. Suciu. The dichotomy of probabilistic inference for unions of conjunctive queries. J. ACM, 2013.
- D. Suciu, D. Olteanu, C. Ré, and C. Koch, Probabilistic Databases, Synthesis Lectures on Data Management, 2011.
- A. Amarilli, İ. İ. Ceylan. The dichotomy of evaluating homomorphism-closed queries on probabilistic graphs, LMCS, 2022.
- B. Kenig, D. Suciu. A Dichotomy for the Generalized Model Counting Problem for Unions of Conjunctive Queries, PODS, 2021.
- İ. İ. Ceylan, A. Darwiche, G. Van den Broeck, Open-world probabilistic databases: Semantics, algorithms, complexity, AIJ, 2021.
- S. Borgwardt, İ. İ. Ceylan, and T. Lukasiewicz. Ontology-mediated queries for probabilistic databases. *AAAI, 2017*.

References

- R. Karp, M. Luby, and N. Madras. Monte-Carlo approximation algorithms for enumeration problems. J. Algorithms, 1989.
- R. Abboud, İ. İ. Ceylan and R. Dimitrov. On the Approximability of Weighted Model Integration on DNF Structures, KR 2020.
- İ. İ. Ceylan, S. Borgwardt, T. Lukasiewicz, Most Probable Explanations for Probabilistic Database Queries, IJCAI 2017.
- D. Roth. On the Hardness of Approximate Reasoning. AIJ, 1996.
- M. Grohe, P. Lindner, Probabilistic databases with an infinite open-world assumption, PODS 2019.