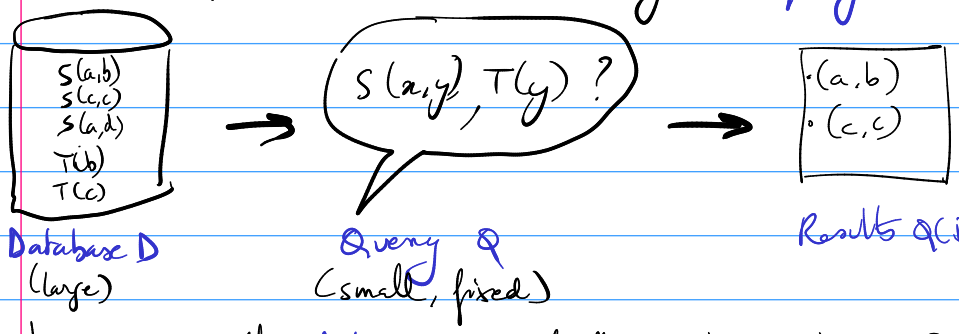
The (In)Tractability of Query Evaluation over Probabilistic Databases

Introduction and Background

Central problem in database theory : query evaluation



Database D (large) — Query Q (small, fixed) — Results Q(D)

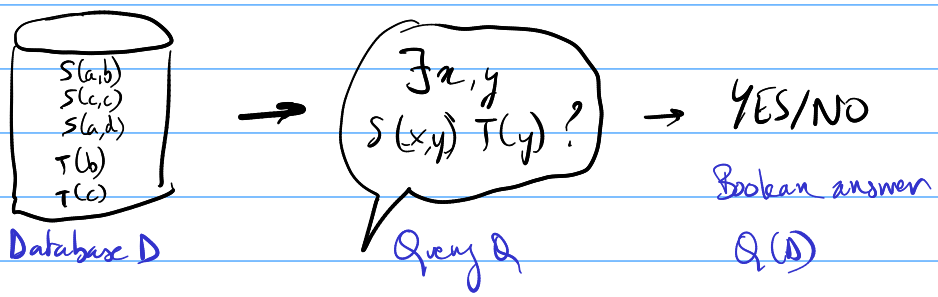
We study the data complexity of this problem, where Q is fixed

⇒ What is the complexity, given D, of computing Q(D) ?

In this talk we will only distinguish two complexity regimes: → tractable (in PTIME)

→ intractable (NP-hard or worse)

Up to a PTIME transformation (testing each possible output), we can assume that the query is Boolean



Database D — Query Q — Boolean answer Q(D)

For most database query languages (CQs, UCQs, Datalog ...) this problem is always tractable

In this talk: we study a probabilistic version of this problem, where the data is uncertain

→ Practical motivations : real-world data is often uncertain, noisy, etc

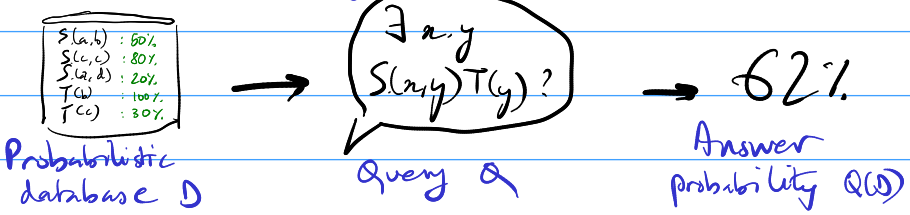
→ Theoretical motivations : the corresponding counting problem is interesting

Probabilistic database :

S(a,b)	: 50%
S(c,c)	: 80%
S(a,d)	: 20%
T(b)	: 100%
T(c)	: 30%

→ each fact is annotated by a probability
→ shows how likely the fact is to exist
→ we assume independence across facts

The Probabilistic Query Evaluation problem for a query Q (PQE(Q))



Probabilistic database D — Query Q — Answer probability Q(D)

Unfortunately the problem PQE(Q) can now become intractable
⇒ For which queries Q can we tractably solve PQE(Q) ?

Connections to :
- Graphical models : inference in Bayesian networks, Markov networks...
- Knowledge compilation : the study of tractable formalisms to represent Boolean functions
- Model counting of Boolean formulas, counting complexity...

Structure of the talk

- Preliminaries
- Upper bounds
  - For tractable self-join-free conjunctive queries
  - Via provenance circuits
  - For tractable UCQs
  - Intensional-extensional conjecture
- Lower bounds
  - #P-hardness for intractable self-join-free conjunctive queries
  - Extending to uniform reliability
  - (#P-hardness of intractable UCQs)
  - #P-hardness of unbounded UCQ^{∞}
- Conclusion

Preliminaries : database theory

- Relational signature σ : predicate names and arity — R with arity 1, S with arity 2 ...
- Constants — $a, b, c \ldots$
- Fact F over σ : predicate and constants — $R(a)$, $S(a,b)$, $S(c,c)$
- Instance I : finite set of facts — $\{R(a), S(a,a)\}$
  also called : database, structure, model ...

- Variables — $x, y, z \ldots$
- Atom over σ : like a fact but with variables — $R(x)$
- Conjunctive query (CQ) : existentially quantified conjunction of atoms — $\exists x,y\ R(x) \wedge S(x,y)$ or just $R(x), S(x,y)$
- Important special case : Self-Join-Free CQ (SJFCQ) — $R(x)\,S(y)$ ok; $S(x,y)\,S(y,z)$ not ok
  → every relation is used at most once
- Union of conjunctive queries (UCQ) : disjunction of CQs — $R(x)S(x,y) \vee S(x,y)T(y)$
  equivalently : positive Boolean first-order formula
  Note : we can reuse variables across disjuncts : $S(x,y) \vee T(y)$ vs $S(x,y) \vee T(z)$

- Homomorphism from a CQ Q to an instance I — $Q : R(x)S(x,y)$ $I = \{R(a), S(a,a)\}$ $h(x) := a$, $h(y) := a$
  witnesses that I satisfies Q : $I \models Q$

Also characterizes CQ entailment : Q implies Q' $(\forall I, I \models Q \Rightarrow I \models Q')$
iff Q' has a homomorphism into Q

Preliminaries : probabilistic databases

- Tuple-independent database (TID) : $J = (I, \pi)$:
  - an instance I
  - a function $\pi : I \to [0,1]$

S(a,b) : 50%
S(c,c) : 80%
S(a,d) : 20%
T(b) : 100%
T(c) : 30%

- Possible world : a subinstance $I' \subseteq I$

A TID is a concise representation of a probability distribution over its possible worlds, where each fact has the indicated probability and we assume independence.

S(a,b) : 50% → 50%
~~S(c,c) : 80%~~ → 20%
S(a,d) : 20% → 20%
T(b) : 100% → 100%
~~T(c) : 30%~~ → 70%

⇒ proba of possible world :
50% × 20% × 20% × 100% × 70%

Formula for $I' \subseteq I$:

$$Pr_J(I') = \underbrace{\prod_{F \in I'} \pi(F)}_{\text{kept facts}} \times \underbrace{\prod_{F \in I \setminus I'} (1 - \pi(F))}_{\text{~~discarded~~ facts}}$$

- Probability of a query Q on a TID $J = (I, \pi)$

$$Pr_J(Q) = \sum_{\substack{I' \subseteq I \\ I' \models Q}} Pr_J(I')$$

S(a,b) : 50%
S(c,c) : 80%
S(a,d) : 20%
T(b) : 100%
T(c) : 30%

Query : $S(x,y)\,T(y)$

Proba :

$1 - (1 - 50\%) \times (1 - 80\% \times 30\%)$

$= 62\%$

- Probabilistic Query Evaluation problem for query Q, PQE(Q) :
  - input : TID J
  - output : compute $Pr_J(Q)$.

- Special case : Uniform Reliability for Q, UR(Q) :
  - input : instance I
  - output : compute how many subinstances satisfy Q : $|\{I' \subseteq I \mid I' \models Q\}|$

⇒ UR(Q) is PQE(Q) in the case where all probabilities are $\frac{1}{2}$ (or 0) up to factor $\frac{1}{2^{|I|}}$.

Goal : For which queries Q is PQE(Q) tractable ?

Tractable cases for SJFCQs : examples

Example 1 : $Q_1 = S(x,y)$

How to solve PQE(Q_1) ?

$S(a,b)$: 50%
$S(c,c)$: 80%
$S(a,d)$: 20%.

→ $Pr_D(Q_1) = Pr_D(\text{some } S\text{-fact in } D)$

$= Pr_D\left(\bigvee_{F \text{ S-fact of } D} F \text{ is kept}\right)$ — independent OR

$= 1 - Pr_D\left(\bigwedge_{F \text{ S-fact of } D} F \text{ is discarded}\right)$ — independent AND

$= 1 - \prod_{F \text{ S-fact of } D} Pr_D(F \text{ is discarded})$

$= 1 - \prod_{F \text{ S-fact of } D} (1 - \pi(F))$

$1 - (1-50\%) \times (1-80\%) \times (1-20\%) = 92\%$

Example 2 : $Q_2 = S(x,y), S'(x',y')$.

→ We know how to compute $Pr_D(S(x,y))$ (above) and $Pr_D(S'(x',y'))$ (ditto)

and : $Pr_D(Q_2) = Pr_D(S(x,y) \text{ AND } S'(x',y'))$

↑ independent AND
→ uses self-join-freeness
→ uses the fact that there are no shared variables

so $Pr_D(Q_2) = Pr_D(S(x,y)) \times Pr_D(S'(x',y'))$

Example 3 : $Q_3 = R(x)S(x,y)$

→ The variable x occurs in each atom so it partitions the database facts

and : $Pr_D(Q_3) = Pr_D\left(\bigvee_{a \text{ constant of } D} \exists x R(a) \wedge S(a,x)\right)$ ← independent OR
→ uses self-join-freeness
→ each database fact is "relevant" for at most one disjunct depending on the value of its first element

$= 1 - \prod_{a \text{ const of } D} \left(1 - Pr_D(\exists x R(a) \wedge S(a,x))\right)$

now $Pr_D(\exists x R(a) \wedge S(a,x))$ is $\pi(R(a)) \times Pr_D(\exists x S(a,x))$ ← independent OR

$= \pi(R(a)) \times Pr_D\left(\bigvee_{b \text{ constant of } D} S(a,b)\right)$

so $Pr_D(Q_3) = 1 - \prod_{R(a) \text{ fact of } I} \left(1 - \pi(R(a)) \times \left(1 - \prod_{S(a,b) \text{ fact of } I} (1 - \pi(S(a,b)))\right)\right)$ which is PTIME...

Tractable cases for SJFCQ : tractability criterion

Def: A hierarchical expression is a (Boolean) CQ built from:

- atoms $R(x)$
- conjunction $Q_1 \wedge Q_2$
- existential quantification $\exists x\, Q(x, y_1, \dots, y_n)$

where the quantified variable occurs in each atom under the quantification

⇒ A CQ is hierarchical if it can be written as a hierarchical expression

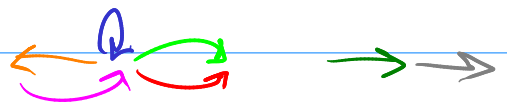
Examples: $\exists x \left(R(x) \wedge \exists y\, S(x,y) \right) \wedge \exists z\, T(z)$ ok

$\exists x \left(R(x) \wedge \exists y \left(S(x,y) \wedge T(y) \right) \right)$ not ok

Example: On an arity-1 signature, all SJFCQs admit hierarchical expressions.

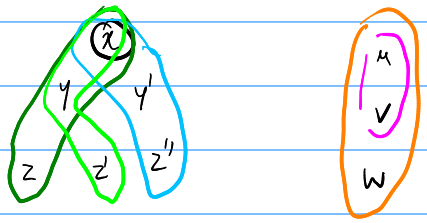
$\left(\exists x\, R_1(x) \wedge R_2(x) \right) \wedge \left(\exists y\, R_3(y) \wedge R_4(y) \right) \wedge \dots$

On an arity-2 signature, the hierarchical SJFCQs consist of (possibly multiple) star patterns



Each star pattern has a "center" and all facts of the pattern involve the center

On a higher-arity signature, hierarchical SJFCQs are more complex to represent



$$\left(\exists x \left(\exists y \left(\exists z\, R_1(xyz) \right) \wedge \left(\exists z'\, R_2(xyz') \right) \right) \wedge \left(\exists y' \exists z''\, R_3(xy'z'') \right) \right) \wedge \left(\exists u \exists v\, R_4(u,v) \wedge \exists w\, R_5(u,v,w) \right)$$

The hierarchical SJFCQs are all tractable:

Dalvi, Suciu VLDBJ '07

Thm: Let Q be a SJFCQ. If Q is hierarchical then PQE(Q) is in PTIME.

Tractable cases for SJFCQ : tractability proof

Note : We extend the definition of hierarchical expressions to allow atoms featuring constants.

Claim: A hierarchical expression can be decomposed by applying the rules:

- conjunction $Q_1 \wedge \cdots \wedge Q_n$ of hierarchical expressions that share no variable
- existential quantification $\exists x\, Q$ where x occurs in all atoms of Q and $Q[x/a]$ is a hierarchical expression
- atoms involving only constants

Proof : Immediate (top-down rephrasing of the bottom-up definition)

Now the probability of a hierarchical expression can be computed via the rules:

- conjunction : $Pr_D(Q_1 \wedge \cdots \wedge Q_n) = Pr_D(Q_1) \times \cdots \times Pr_D(Q_n)$
  independent AND → uses self-join-freeness
- existential quantification : $Pr_D(\exists x\, Q) = Pr_D(\bigvee_a Q[x/a])$
  $= 1 - \prod_a (1 - Pr_D(Q[x/a]))$
  independent OR → uses self-join-freeness → uses the fact that x occurs in all atoms
- atoms $Pr_D(F) = \pi(F)$ for F a fact (atom with only constants)

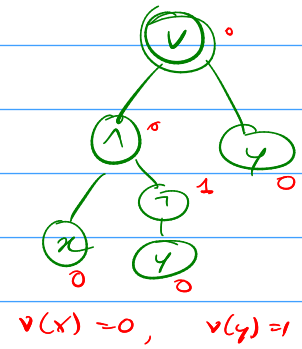
⇒ This computes the probability of a hierarchical SJFCQ in linear time in the input TID assuming that arithmetic operations take constant time

Let us see this result in a different way, using Boolean circuits

→ define the problem of computing the probability of a circuit
→ define tractable circuit classes for the problem
→ explain how to reduce PQE to computing the probability of a provenance circuit
→ explain why, for SJFCQs, we can make the circuit tractable

Circuits, probabilities, and tractable circuits

- Boolean circuit : DAG where the vertices (gates) are :
  - Variables :
  - Boolean operators $\wedge$, $\vee$, $\neg$ (indegree 1)
  - Distinguished output gate
- Valuation of variables X : function $\nu : X \to \{0,1\}$
  → gives an evaluation of all gates of the circuit



$\nu(x) = 0$, $\nu(y) = 1$

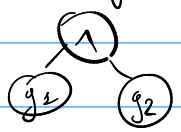
- Probability computation on a circuit C :
  Given a probabilistic valuation $\pi : X \to [0,1]$
  what is the probability that the circuit evaluates to 1, assuming independence ?

$\pi(x)$: 20%
$\pi(y)$: 70%
proba : 76%

→ This problem is intractable (SAT reduces to it)

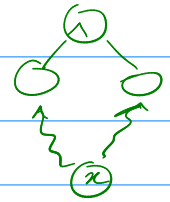
- Deterministic decomposable circuit (d-D) :

- each AND gate is decomposable :

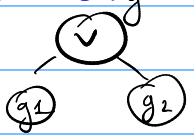


no variable has a path in the circuit to g_1 and g_2
→ the gate g_1 depends on X_1 and g_2 depends on X_2 with $X_1 \cap X_2 = \emptyset$

We do not have :

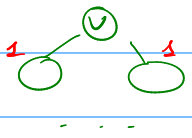


- each OR gate is deterministic :



no valuation makes both g_1 and g_2 true
→ they are mutually exclusive

We do not have :

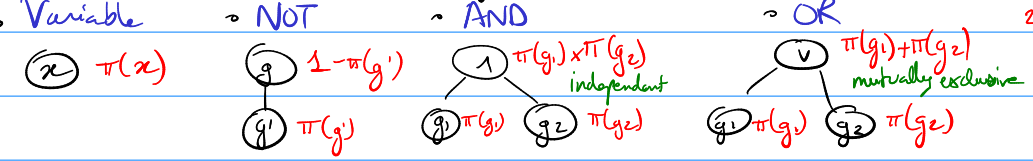


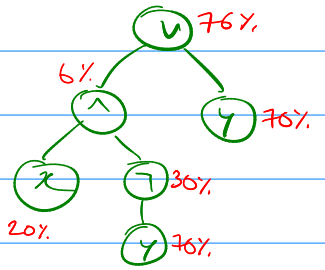
valuation ν

Prop : We can perform probability computation on a d-D circuit in linear time (assuming constant-time arithmetics).

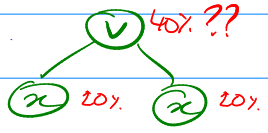
Proof : Compute the probability on each gate bottom-up

- Variable : $\pi(x)$
- NOT : $1 - \pi(g')$
- AND : $\pi(g_1) \times \pi(g_2)$ independent
- OR : $\pi(g_1) + \pi(g_2)$ mutually exclusive





Remark : does not work for general circuits, because of correlations



Provenance circuits

- Provenance of a query Q on an instance I:
  Boolean function $Prov(Q, I)$ having as variables the facts of I
  such that for any valuation $\nu : I \to \{0,1\}$,
  letting $I_\nu = \{F \in I \mid \nu(F) = 1\}$ be the subinstance of facts mapped to 1
  we have $I_\nu \models Q$ iff $Prov(Q,I)$ evaluates to 1 under ν

  $I = \{S(a,b), S(c,c), S(a,d), T(b), T(c)\}$
  $Q : S(x,y)\ T(y)$
  $Prov(Q,I) : (S(a,b) \wedge T(b)) \vee (S(c,c) \wedge T(c))$

→ The provenance describes the subinstances of I that satisfy Q

- Provenance circuit of Q on I: Boolean circuit that expresses $Prov(Q,I)$

Thm: "We can perform PQE via provenance circuits":
Let $J = (I, \pi)$ be a TID, let Q be a query, let C be a provenance circuit of Q on I:
the probability $Pr_J(Q)$ is precisely the answer of probability evaluation of C with π

Proof: By definition

See Green, Karvounarakis, Tannen, PODS'07

Thm: For any fixed UCQ Q, given an instance I, we can compute in PTIME a provenance circuit

Proof: write it in Disjunctive Normal Form (DNF)
with a disjunct for every image of a homomorphism of Q in I

$I = \{S(a,b), S(c,c), S(a,d), T(b), T(c)\}$
$Q : S(x,y)\ T(y)$
homomorphism 1: $x \mapsto a,\ y \mapsto b$ ⇒ disjunct $S(a,b) \wedge T(b)$
homomorphism 2: $x \mapsto c,\ y \mapsto c$ ⇒ disjunct $S(c,c) \wedge T(c)$

This can be useful, eg, to explain why the query holds but does not help for PQE unless the circuit is in a tractable class (here, d-Ds).

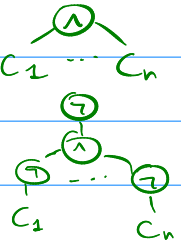
Via Olteanu, Huang, SUM'08

Thm: For any hierarchical SJFCQ Q, given any instance I,
we can compute in linear time a d-D provenance circuit of Q on I

→ We get the tractability of PQE as a corollary.

Proof: Recall which rules we used:

- Conjunction: $Pr_D(Q_1 \wedge \dots \wedge Q_n)$ (independent AND)
- existential: $Pr_D(\bigvee_a Q[x/a])$ (independent OR)
- facts: variable gate



The subcircuits C_i depend on facts from different relations, so this $\wedge$ is decomposable

(Using de Morgan's law)
The subcircuits C_i depend on different facts (depending on the values of the root variables), so this $\wedge$ is decomposable

Note: We can get circuits in more restricted formalisms (read-once formulas, OBDDs...)

Tractable cases for UCQs : independent $\wedge$ and $\vee$

We go from SJFCQs to UCQs → union in addition to conjunction
→ repeated relation symbols

Def : A UCQ is an expression built with atoms, $\wedge$, $\vee$, and existential quantification

$(\exists x R(x) \wedge \exists y T(y)) \vee \exists xy S(xy)$

Standard way to write a UCQ : union of CQs :

$(\exists xy\ R(x) \wedge T(y)) \vee \exists xy\ S(xy)$

We decompose the CQs into connected components based on shared variables (called components)
We get $(C_{11} \wedge C_{12} \wedge \dots) \vee (C_{21} \wedge C_{22} \wedge \dots) \vee \dots$ where the C_{ij} are components (with quantifiers)
This is called the DNF representation of the UCQ

$(\exists x R(x) \wedge \exists y T(y)) \vee \exists xy S(xy)$

Far less standard representation : CNF representation of the UCQ
Obtained using de Morgan's law
We get $(C'_{11} \vee C'_{12} \vee \dots) \wedge (C'_{21} \vee C'_{22} \vee \dots) \wedge \dots$

$(\exists x R(x) \vee \exists xy S(xy)) \wedge (\exists y T(y) \vee \exists xy S(xy))$

For SJFCQs we simplified $\wedge$ as independant AND
→ no longer possible in general : repeated relation symbols
If it is possible, we do it :

Rule : If we can write Q as $Q' \wedge Q''$ where Q' and Q'' are Boolean queries that share no relation symbols, then $Pr(Q) = Pr(Q') \times Pr(Q'')$.

Ex : $Pr((\exists x R(x) \vee \exists y T(y)) \wedge \exists xy S(xy)) = Pr(\exists x R(x) \vee \exists y T(y)) \times Pr(\exists xy S(xy))$

Rule : If we can write Q as $Q' \vee Q''$ where Q' and Q'' are Boolean queries that share no relation symbols, then

$$Pr(Q) = 1 - (1 - Pr(Q'))(1 - Pr(Q''))$$

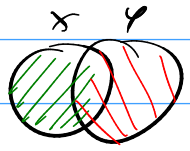
→ can be tested by putting Q in CNF, and in DNF, and counting connected symbol-components.

Tractable case for UCQs : inclusion-exclusion

Say we have a UCQ where we cannot apply these two rules

Idea : use inclusion-exclusion principle on the CNF to get to unions of components (and then see what we can do ...)

Inclusion-exclusion (sets) : $|X \cup Y| = |X| + |Y| - |X \cap Y|$



Inclusion-exclusion (probabilistic) : $Pr(Q \vee Q') = Pr(Q) + Pr(Q') - Pr(Q \wedge Q')$

→ independence NOT required !

Inclusion-exclusion (dual, for CNF) : $Pr(Q \wedge Q') = Pr(Q) + Pr(Q') - Pr(Q \vee Q')$.

Example : $Pr\big(\overbrace{(\exists x R(x) \vee \exists y S(y))}^{Q_1} \wedge \overbrace{(\exists y S(y) \vee \exists z T(z))}^{Q_2}\big)$

$= Pr(\overbrace{\exists x R(x) \vee \exists y S(y)}^{\text{independent AND}}) + Pr(\overbrace{\exists y S(y) \wedge \exists z T(z)}^{\text{independent AND}})$

$- Pr(\underbrace{\exists x R(x) \vee \exists y S(y) \vee \cancel{\exists y S(y)} \vee \exists z T(z)}_{\text{independent OR}})$ (redundant (see later))

OK, which case remains ?

→ A union of components $c_1 \vee \cdots \vee c_n$ which is not an independent OR

Idea : Remember hierarchical expressions !

If some c_i does not have a root variable, we fail (intractable)

Otherwise, $\exists x c_1 \vee \exists x c_2 \vee \cdots \vee \exists x c_n \equiv \exists x\, c_1 \vee c_2 \vee \cdots \vee c_n$

(specific to union!) (now x occurs in all atoms !)

Idea : Can we write this as $\bigvee_a [c_1 \vee \cdots \vee c_n][x/a]$?

Tractable cases for UCQs : separator variables

Example: $(\exists x y\, R(x), S(x,y)) \vee (\exists x y\, R'(x), S(x,y))$

$\equiv \exists x \left(\exists y\, R(x) S(x,y) \vee \exists y\, R'(x) S(x,y)\right)$

$\equiv \bigvee_{a \text{ in domain}} \exists y\, R(a) S(a,y) \vee \exists y R'(a) S(a,y)$

Is this an independent OR? → YES, a database fact contributes to at most one disjunct

So we can simply compute $Pr(\exists y\, R(a) S(a,y) \vee \exists y\, R'(a) S(a,y))$

components : only shared variables count

this is $Pr((R(a) \vee R'(a)) \wedge (R(a) \vee \exists y S(a,y)) \wedge (R'(a) \vee \exists y S(a,y)) \wedge (\exists y S(a,y) \vee \exists y S(a,y))$

redundant (later) — redundant

$= Pr((R(a) \vee R(a')) \wedge \exists y S(a,y))$: ok

independent OR — independent AND

So $(\exists x y\, R(x), S(x,y)) \vee (\exists x y\, R'(x), S(x,y))$ is tractable

Counter-example : $(\exists x y\, R(x), S(x,y)) \vee (\exists x y\, S(y,x), T(x))$

$\equiv \exists x (\exists y\, R(x) S(x,y)) \vee (\exists y\, S(y,x), T(x))$

$\equiv \bigvee_{a \text{ in domain}} \exists y\, R(a) S(a,y) \vee \exists y\, S(y,a), T(a)$

instance: $R(u)\ S(u,v)\ T(v)$

On the instance: $(\exists y\, R(u) S(u,y) \vee \exists y S(y,u), T(u))$ (no match) $\vee (\exists y R(v) S(v,y) \vee \exists y\, S(y,v), T(v))$ (no match)

PROBLEM : disjuncts are not independent !!

In fact, this query is intractable... the order of variables matters !

Def : A separator variable for a union of components $C_1 \vee \cdots \vee C_n$ is a root variable x such that we can write $\exists x\, C_1 \vee \cdots \vee C_n$

(formally, we have chosen one root variable per component : x occurs in all atoms)

such that for every relation S there is at least one position i s.t. x occurs at position i in every S-atom

→ Guarantees that $\bigvee_a Q[x/a]$ is really an independent OR

Dalvi, Suciu, JACM '12

Tractable cases for UCQs : Summary of the algo so far

- If Q is a fact $Q = R(a_1, \dots a_n)$ (atom with no variables) then look up its probability [base]
- If you can write $Q = Q' \wedge Q''$ with Q' and Q'' syntactically independent (Q in CNF), then $Pr(Q) = Pr(Q') \times Pr(Q'')$ [independent AND]
- If you can write $Q = Q' \vee Q''$ with Q' and Q'' syntactically independent (Q in DNF) then $Pr(Q) = 1 - (1 - Pr(Q'))(1 - Pr(Q''))$ [independent OR]
- If you have $Q = Q_1 \wedge \dots \wedge Q_n$ in CNF ($n > 1$) then [inclusion-exclusion]

$$Pr(Q) = Pr(Q_1) + Pr(Q_2) + \dots + Pr(Q_n) - Pr(Q_1 \vee Q_2) - Pr(Q_1 \vee Q_3) - \dots + \dots - \dots$$

- If you have $Q = C_1 \vee \dots \vee C_k$ as a union of components and we can write Q as $\exists x\, C_1 \vee \dots \vee C_k$ with x a separator variable then return $1 - \prod_{a \text{ in domain}} 1 - Pr(Q[x/a])$ [independent project]
- Else FAIL

Making the algorithm complete : 2 small fixes and one big fix

Small fix 1 : minification

At every step, make the query minimal by removing redundant disjuncts/conjuncts

ex : $R(x) \vee R(x) S(x,y) T(y)$

Small fix 2 : ranking

Transformation on instance and database to make sure all atoms "use variables in the same order"

ex : $R(x,y) R(y,x)$ PTIME but no separator variable

→ Fix an order on the domain of instance I

→ Replace $R(a,a)$ by $R_=(a)$, for $a<b$ replace $R(a,b)$ by $R_<(a,b)$, for $a>b$ replace $R(a,b)$ by $R_>(b,a)$

→ Consider all possible rankings on the query variables $x=y$, $x<y$, $x>y$

→ Get disjuncts $R_<(x,y) R_>(x,y) \vee R_=(x)$

Note : $R(x) S(x,y) \vee S(x,y) T(y)$ is ranked

Tractable cases for UCQs : Cancellations in inclusion-exclusion

Big fix : In inclusion-exclusion, some terms may cancel out.

Recall that $Pr(Q_1 \wedge \dots \wedge Q_n) = Pr(Q_1) + Pr(Q_2) + \dots + Pr(Q_n)$
$- Pr(Q_1 \vee Q_2) - \dots$
$+ \dots$

Some of those queries may end up being equivalent and cancel out.

ex : $(Q_2 \vee Q_3) \wedge (Q_0 \vee Q_3) \wedge (Q_1 \vee Q_3) \wedge (Q_0 \vee Q_1 \vee Q_2)$

written $23 \wedge 03 \wedge 13 \wedge 012$

Inclusion-exclusion : $Pr(23 \wedge 03 \wedge 13 \wedge 012) =$

$Pr(23) + Pr(03) + Pr(13) + Pr(012)$
$- Pr(023) - Pr(123) - Pr(0123) - Pr(013) - Pr(0123) - Pr(0123)$
$+ Pr(0123) + Pr(0123) + Pr(0123) + Pr(0123)$
$- Pr(0123)$

It may be the case that $P(0123)$ is intractable to compute!

Hence, fix : In inclusion-exclusion, regroup the equivalent queries and recurse only on those with non-zero coefficient

$\Rightarrow$ Now the algorithm is complete

Tractable cases for UCQs : provenance circuits

Recall the case of SJFCQs :

> **Thm** : For any hierarchical SJFCQ Q, given any instance I, we can compute in linear time a d-D provenance circuit of Q on I
> → We get the tractability of PQE as a corollary.

Is the same true for UCQs?

Recall the rules :

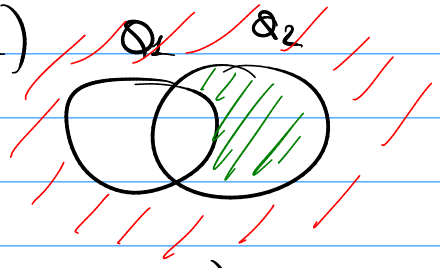
- independent AND : $Q = Q' \wedge Q''$ (indep) ⇒ ok (decomposable AND)
- independent OR : $Q = Q' \vee Q''$ (indep) ⇒ ok (decomposable AND + negation)
- independent project : $Q = \bigvee_a Q[x/a]$ (indep) ⇒ ok (ditto)
- inclusion-exclusion (with cancellations) $Pr(Q_1 \wedge Q_2 \wedge \dots \wedge Q_n) = Pr(Q_1) + \dots - \dots + \dots$

⇒ problem! This is not a logical operation but arithmetics on the probabilities

In simple cases, we can do it at a logical level :

$$Q_1 \wedge Q_2 \equiv \neg\big(\neg(Q_1 \vee \neg(Q_1 \vee Q_2)) \vee \neg Q_2\big)$$

($Q_1 \vee \neg(Q_1 \vee Q_2)$: mutually exclusive; $\neg(\dots) \vee \neg Q_2$: mutually exclusive)



→ expresses $Pr(Q_1 \wedge Q_2)$ as a function of $Pr(Q_1)$, $Pr(Q_2)$, $Pr(Q_1 \vee Q_2)$

$$\begin{aligned} Pr(Q_1 \wedge Q_2) &= 1 - \big(1 - (Pr(Q_1) + 1 - Pr(Q_1 \vee Q_2)) + 1 - Pr(Q_2)\big) \\ &= Pr(Q_1) + 1 - Pr(Q_1 \vee Q_2) - 1 + Pr(Q_2) \\ &= Pr(Q_1) + Pr(Q_2) - Pr(Q_1 \vee Q_2) \end{aligned}$$

We can apply this recursively, but does not handle cancellations

⇒ Open problem!

See Monet, PODS'20

Intensional-extensional conjecture : There are UCQs Q such that PQE(Q) is tractable but we cannot build a d-D provenance circuit for Q on an input instance in PTIME

(I hope the conjecture can be disproved)

Beame, Li, Roy, Suciu, TODS'17
Bova, Gyssens, PODS'17

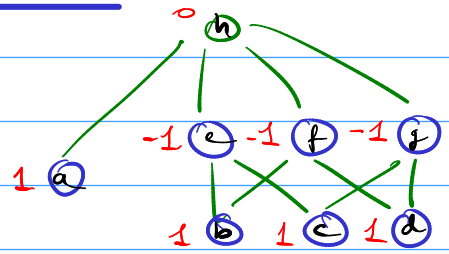
Remark : known to be true for weaker formalisms than d-Ds :

- FBDDs and dec-DNNFs and DLDDs
- structured d-DNNFs

Also open for d-DNNFs (d-Ds with negations at leaves), open if d-D ≡ d-DNNFs
Some characterizations known for weaker formalisms, e.g., OBDDs (inversion-free UCQs)

Wilfrid Mikaël Monat and Louis Jachiet

Stand-alone rephrasing of the open problem

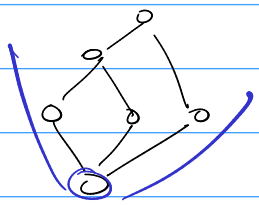


Take a join-semilattice $(V, <)$

Define the Möbius function for $v \in V$

$$\mu(v) = 1 - \sum_{w<v} \mu(w)$$

Let $S \subseteq V$ be the set of vertices v having $\mu(v) \neq 0$



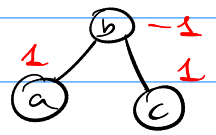
Inductively define the reachable subsets:

- For each $v \in S$, its upset $\{w \in V \mid v < w\}$ is reachable
- If S_1 and S_2 are reachable and disjoint then $S_1 \sqcup S_2$ is reachable
- If S_1 and S_2 are reachable and $S_2 \subseteq S_1$ then $S_1 \setminus S_2$ is reachable

The join-semilattice $(V, <)$ is good if V is reachable.

⇒ Are all join-semilattices good?

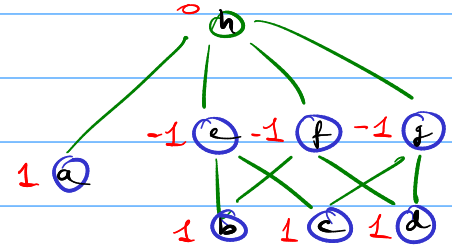
ex:



$\{a, b\}$, $\{b\}$, $\{c, b\}$

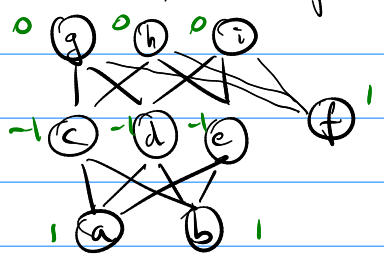
$\{a\} = \{a, b\} \setminus \{b\}$

$\{a, b, c\} = \{a\} \cup \{c, b\}$



$$\hat{b} - \hat{e} + \hat{c} - \hat{f} + \hat{a} - \hat{g} + \hat{d} = V$$

False on DAGs: (found by bruteforce search)



Lower bounds for non-hierarchical SJFCQs : basics

- #P : class of counting problems expressible as the number of accepting paths of a polynomial-time nondeterministic Turing machine

#SAT : count the satisfying assignments

→ A counting analogue of NP : NP asks "is the count ≥ 1 ?"

→ but there are PTIME problems whose counting analogue is #P-hard (see later)

Remark : If (non-probabilistic) query evaluation for a query Q is in PTIME, then PQE(Q) is in #P

Proof : The Turing machine first guesses a valuation (need to bias by the probabilities) then evaluates the query on the possible world

(Technically in $FP^{\#P}$ because the output of PQE is a fraction and we must normalize by the denominator)

- #P-hard problem: every problem in #P reduces to it by a PTIME (Turing) reduction

Provan, Ball, SICOMP'83

Our initial #P-hard problem : the #PP2DNF problem aka counting independent sets in bipartite graphs

Input : • Variables partitioned into two sets $X = \{X_1, \dots, X_n\}$ and $Y = \{Y_1, \dots, Y_m\}$

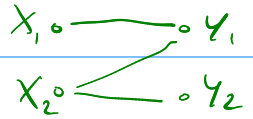
• Clauses $X_{i_1} \wedge Y_{j_1} \vee \dots \vee X_{i_k} \wedge Y_{j_k}$

- no negation
- one X-variable and one Y-variable per clause

Output : How many satisfying assignments for the formula $(X_{i_1} \wedge Y_{j_1}) \vee \dots \vee (X_{i_k} \wedge Y_{i_k})$?

Example : formula $(X_1 \wedge Y_1) \vee (X_2 \wedge Y_1) \vee (X_2 \wedge Y_2)$

→ 8 satisfying assignments out of 16

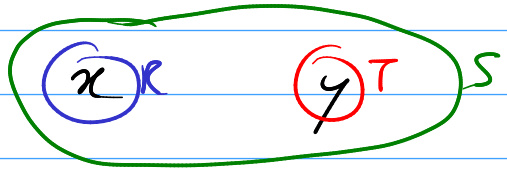


Dalvi, Suciu, VLDBJ'07

Lower bounds for non-hierarchical SJFCQs: R-S-T queries

Let's first show hardness for the simplest non-hierarchical SJFCQ:

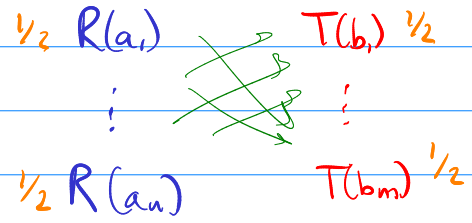
$Q: R(x), S(x,y), T(y)$



Theorem: PQE for the SJFCQ $Q: R(x)\, S(x,y)\, T(y)$ is #P-hard

Proof: consider a #PP2DNF instance $\{x_1, \dots x_n\}, \{y_1, \dots, y_m\}, x_{i_1} \wedge y_{j_1}, \dots, x_{i_k} \wedge y_{j_k}$

Create the TID:



plus facts $S(a_{i_p}, b_{j_p})$ with proba 1 for $1 \le p \le k$

→ there is a probability-preserving bijection between possible worlds and valuations

→ a valuation satisfies the formula (x_{i_p} and y_{j_p} both true for some $1 \le p \le k$) iff the possible world satisfies Q (contains facts $R(a_{i_p})\, S(a_{i_p}, b_{j_p})\, T(b_{j_p})$ for some $1 \le p \le k$)

Let us extend to other single non-hierarchical SJFCQs:

Def: An R-S-T query is a query of the form:

$$Q_{rst}(x,y): R_1(x), \dots, R_r(x), S_1(x,y), \dots, S_s(x,y), T_1(y), \dots, T_t(y)$$

for some $r, s, t > 0$.

Corollary: PQE(Q) is #P-hard for any R-S-T query

Proof: Same reduction as before, add facts $R_2(a), \dots, R_r(a)$
$S_2(a,b), \dots, S_s(a,b)$
$T_2(b), \dots, T_t(b)$

for all elements a, b, with proba 1.

Lower bounds for non-hierarchical SJFCQs : hard patterns

We want to show #P-hardness for all non-hierarchical SJFCQs

Lemma : A SJFCQ Q is hierarchical iff the following holds:

Letting atoms(x) for a variable x of Q be the set of atoms that contain x,

For any two variables x and y,

the sets atoms(x) and atoms(y) are either :

- disjoint : atoms(x) $\cap$ atoms(y) $= \emptyset$
- comparable : atoms(x) $\subseteq$ atoms(y) or atoms(y) $\subseteq$ atoms(x)

$Q : R(x), S(x,y), T(z)$

atoms(x) $= \{R(x), S(x,y)\}$

atoms(y) $= \{S(x,y)\}$ (disj. from atoms(x))

atoms(z) $= \{T(z)\}$

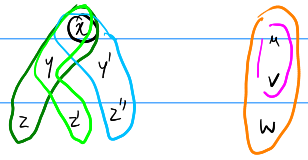
$Q : R(x), S(x,y), T(y)$

atoms(x) $= \{R(x), S(x,y)\}$

atoms(y) $= \{T(y), S(x,y)\}$

intersect but not comparable

Proof : • hierarchical $\Rightarrow$ condition, by induction



- One atom A: true, atoms(x) $= \{A\}$ for all vars
- $\exists x\, Q(x, y_1, \dots, y_n)$: true, atoms is unchanged
- $Q_1 \wedge Q_2$: • if a variable x occurs both in Q_1 and Q_2, then it must occur in all atoms (because it will be quantified later) so atoms(x) and atoms(y) cannot be a counterexample for such a variable x
  • for x occurring only in Q_1 and y occurring only in Q_2, atoms(x) $\cap$ atoms(y) $= \emptyset$
  • for x and y occurring only in Q_1, conclude by induction hypothesis
  • (ditto for x and y only in Q_2)

• condition $\Rightarrow$ hierarchical, also by induction

Take the variables $x_1, \dots, x_n$ such that atoms(x_1), ..., atoms(x_n) are maximal under inclusion

We must have atoms(x_i) $\cap$ atoms(x_j) $= \emptyset$ or atoms(x_i) $=$ atoms(x_j) for all i, j

(otherwise, by the condition, they are comparable, and different, contradicting maximality)

Write each atom with no variables as a conjunct

The maximal sets cover all other atoms:

each distinct set is a conjunct where you quantify the variables achieving the set, and recurse

$x : A_1 A_2 A_3 A_4$ $\quad u : A_5$

$y : A_1 A_2 A_3 A_4$ $\quad v : A_5$

$z : A_1 A_2 \quad w : A_3$

$z' : A_1$

$$\exists x \exists y \left(A_4 \wedge (\exists z A_2 \wedge (\exists z' A_1)) \wedge (\exists w A_3) \right) \wedge \left(\exists u \exists v\, A_5 \right)$$

Corollary : If Q is a non-hierarchical SJFCQ, it has two variables x, y such that

- atoms(x) $\cap$ atoms(y) $\neq \emptyset$ So there are • atoms with x and y Call these atoms to $R(x,y)$
- atoms(x) $\not\subseteq$ atoms(y) • atoms with x but not y $S(x,y)$
- atoms(y) $\not\subseteq$ atoms(x) • atoms with y but not x $T(x,y)$

(Chan - empty)

Lower bound for non-hierarchical SJFCQ : reducing from R-S-T

Theorem: Let Q be a SJFCQ with nonempty atom sets

$\mathcal{R}(x)$ atoms using x but not y

$\mathcal{S}(x,y)$ atoms using x and y

$\mathcal{T}(y)$ atoms using y but not x

for some variables x, y. Let $r = |\mathcal{R}|$, $s = |\mathcal{S}|$, $t = |\mathcal{T}|$.

Then $\mathrm{PQE}(Q_{rst})$ reduces in PTIME to $\mathrm{PQE}(Q)$

Proof: Take a TID J on which we want to solve $\mathrm{PQE}(Q_{rst})$.

Let c be a new constant.

Build J' from J by changing the facts:

- For the p-th atom $R(x, z_1, \dots, z_n)$ of $\mathcal{R}$
  replace each fact $R_p(a)$ by $R(a, c, \dots, c)$
  keeping the same probability
- Do similarly for $\mathcal{R}, \mathcal{S}, \mathcal{T}$
- For each atom in $\mathcal{Z} = Q \setminus (\mathcal{R} \cup \mathcal{S} \cup \mathcal{T})$
  letting Z be the relation, add a fact $Z(c)$ with proba 1.

$R(x), R'(x,w,x), S(x,w,y), T(w,y), U(w,w)$

$\mathcal{R}$ $\mathcal{S}$ $\mathcal{T}$ $\mathcal{Z}$

$R_1(a)^{1/2}, R_2(a)^{1/2}, S_1(a,b)^{1}, T_1(b)^{1/2}$

becomes

$R(a)^{1/2}, R'(a,c,a)^{1/2}, S(a,c,a)^{1}, T(c,b)^{1/2}$

$U(c,c)^{1}$

Correctness: there is a probability-preserving bijection between the possible worlds of J and J' and
- if a possible world of J satisfies Q_{rst} then the translation of any match witnesses that the corresponding world of J' satisfies Q
- if a possible world of J' satisfies Q, restricting to $\mathcal{R} \cup \mathcal{S} \cup \mathcal{T}$ and taking the preimage of a match, the corresponding world of J satisfies Q

Dalvi, Suciu, VLDBJ'07

Conclusion: For any SJFCQ Q:

- either Q is hierarchical and we can solve $\mathrm{PQE}(Q)$ in PTIME on any instance I by building a d-D circuit representing $\mathrm{Prov}(Q, I)$
- or $\mathrm{PQE}(Q)$ is #P-hard

A., Kimelfeld
ICDT'21

#P-hardness of Uniform Reliability for Non-Hierarchical SJFCQ

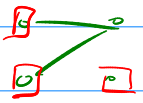
Remember Uniform Reliability (UR): all facts have probability 1/2
We know it is tractable for hierarchical SJFCQs (like PQE)
But is it also #P-hard for non-hierarchical SJFCQs?
(Reduction used probas 1/2, 0, and 1)

Theorem: UR(Q) is #P-hard for any non-hierarchical SJFCQ Q

- Also works for other fixed probabilities than 1/2 (if $0<p<1$)
- Example use case of the interpolation method used in many hardness proofs

By the previous techniques it essentially suffices to show:

Theorem: UR(Q) is #P-hard for $R(x)\,S(x,y)\,T(y)$



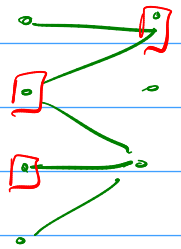
We will reduce from counting the independent sets of a bipartite graph (#BIS ≡ #PP2DNF)
Let $G=(R \cup T, S)$ be the input and X be the set of independent sets: we want $|X|$

Principle of the interpolation method:

1. Partition the subsets of $R \cup T$ based on some properties
   $$2^{R \cup T} = X_0 \sqcup X_1 \sqcup \cdots \sqcup X_n$$
   where $X = X_{i_1} \sqcup \cdots \sqcup X_{i_k}$
2. Code G into TIDs with gadgets having a degree of freedom
   $J_0, J_1, \ldots, J_n$
3. Relate the answer of the target problem, $Pr_{J_d}(Q)$ for each d, to the cardinalities $|X_0|, \ldots, |X_n|$ as a linear equation
4. Show invertibility of the matrix of the system

#P-hardness of UR : 1. Partitioning

Consider the bipartite graph $G = (R \cup T, S)$
and subsets of vertices $R' \subseteq R$, $T' \subseteq T$



The "important quantities" of R' and T' are:

- number of edges $\bullet \!\!-\!\!\!-\!\!\bullet$ c_{00}
- number of edges $\bullet \!\!-\!\!\!-\!\!\square$ or $\square \!\!-\!\!\!-\!\!\bullet$ c_{01}
- number of edges $\square \!\!-\!\!\!-\!\!\square$ c_{11}

$c_{00} = 1$

$c_{01} = 3$

$c_{11} = 1$

Note that $c_{00} + c_{01} + c_{11} = |S|$ (number of edges), so 2 "degrees of freedom"

Write $X_{\vec{c}}$ the set of vertex subsets with quantities as indicated
($|X_{\vec{c}}| = 0$ if $c_{00} + c_{01} + c_{11} \neq |S|$)

We have : $2^{R \cup T} = \bigsqcup_{\vec{c}} X_{\vec{c}}$

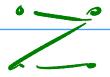
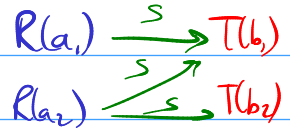
The independent sets are precisely in the $X_{\vec{c}}$ where $c_{11} = 0$

$$X = \bigsqcup_{c_{00} c_{01}} X_{c_{00} c_{01} 0}$$

$$|X| = \sum_{c_{00} c_{01}} |X_{c_{00} c_{01} 0}|$$

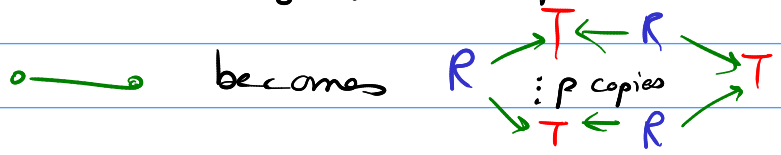
So let us show how to compute all the $X_{\vec{c}}$

#P-hardness of UR : 2. Coding

It is easy to code the bipartite graph  to a PQE instance 

but all facts now have probability 1/2 ... What can vary ?

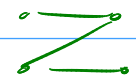
Idea : Code each edge by p copies of a "back-and-forth path"



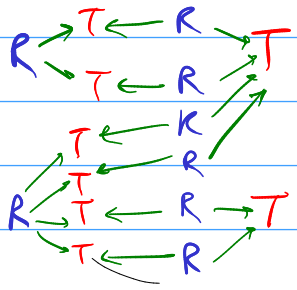
For a choice of p, call J_p the result of coding the input G in this way

#P-hardness of UR: 3. Relating

Given a TID J_p, our UR oracle tells us how many subinstances violate Q: N_p ($2^{|J_p|}$ − oracle answer)



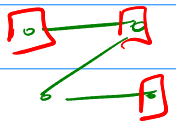
$p = 2$

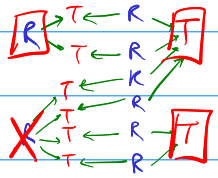


N_2 = how many subinstances have no $R \to T$

How to relate the N_p and the $|X_c|$?

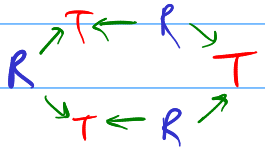
- In J_p, when choosing a possible world, we must decide if we keep/discard the endpoints R and T





$$N_p = \sum_{\substack{R' \subseteq R \\ T' \subseteq T}} N_p(R', T')$$

- Now, $N_p(R', T')$ is the product of the number of violating possible worlds for each gadget



→ uses the fact that each "possible" query match is "within" only one gadget (+endpoints)

- Each such number depends only on p (here $p=2$) and whether the endpoints are kept or not (4 possible types: $\square - \square$ $\square - \times$ $\times - \square$ $\times - \times$)

Call $N_{p,bb'}$ this number, for $b, b' \in \{0,1\}$ the type

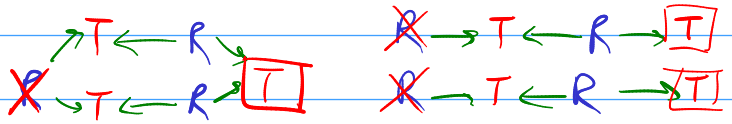
by symmetry $N_{p,01} = N_{p,10}$

- How many gadgets of each type we have is given by the "important quantities" $c_{00}(R', T')$, $c_{01}(R', T')$, $c_{11}(R', T')$

$$N_p(R', T') = N_{p,00}^{c_{00}} \times N_{p,01}^{c_{01}} \times N_{p,11}^{c_{11}}$$

#P_hardness of UR : 3. Relating (cont'd)

- Let's compute: $N_{pbb'}$. First note that $N_{pbb'} = (N_{bb'})^p$



- Let's compute!

N_{00} — $\not R \to T \leftarrow R \to \not T$ — $2 \times 2 \times (8-1) = 28 = 4 \times 7$

N_{01} — $\not R \to T \leftarrow R \to T$ — $2 \times (8+3) = 22 = 2 \times 11$ (R missing)

(or $R \to T \leftarrow R \to \not T$ — symmetry)

N_{11} — $R \to T \leftarrow R \to T$ — $3 \times 3 + 4 + 2 \times 2 = 17$ (middle S missing; both R and T missing; one present one missing)

Now let's put the equations together

$$N_p = \sum_{\substack{R' \in R \\ T' \in T}} N_p(R', T') = \sum_{\substack{R' \in R \\ T' \in T}} N_{p\,00}^{c_{00}} \times N_{p\,01}^{c_{01}} \times N_{p\,11}^{c_{11}}$$

$$= \sum_{\substack{R' \in R \\ T' \in T}} \left(N_{00}^{c_{00}} \times N_{01}^{c_{01}} \times N_{11}^{c_{11}} \right)^p$$

The sum argument only depends on the $c_{bb'}$ → inject the $|X_{\vec{c}}|$!

$$N_p = \sum_{c_{00}, c_{01}, c_{10}} |X_{c_{00} c_{01} c_{10}}| \times \left(N_{00}^{c_{00}} \times N_{01}^{c_{01}} \times N_{11}^{c_{11}} \right)^p$$

This is in fact the equation system:

$$\begin{pmatrix} N_0 \\ N_1 \\ N_2 \\ \vdots \\ N_B \end{pmatrix} = \begin{pmatrix} \alpha(0,0,0)^0 & \cdots & \alpha(|S|,|S|,|S|)^0 \\ \vdots & & \\ \alpha(000)^B & & \alpha(|S|,|S|,|S|)^B \end{pmatrix} \times \begin{pmatrix} |X_{000}| \\ \vdots \\ |X_{|S||S||S|}| \end{pmatrix}$$

where $B = (|S|+1)^3 - 1$

and

$\alpha(ijk) = N_{00}^i N_{01}^j N_{11}^k$

#P-hardness of UR : 4. Showing invertibility

The matrix M has coefficients $M_{p,(i,j,k)} = \alpha(ijk)^p$

We recognize a Vandermonde matrix on values $\alpha(ijk)$

Lemma : A Vandermonde matrix is invertible iff its "base values" are all different

$\begin{pmatrix} v_0^0 & v_1^0 \\ v_0^1 & v_1^1 \end{pmatrix}$

$v_0 \neq v_1$

Proof : If we could write a linear combination of the rows $\lambda_0 R_0 + \dots + \lambda_{n-1} R_{n-1} = 0$ $(\lambda_0, \dots, \lambda_{n-1}) \neq (0, \dots, 0)$

then, looking at each column, each base value is a different root of the polynomial $\lambda_0 x^0 + \dots + \lambda_{n-1} x^{n-1}$ of degree $n-1$

$\Rightarrow$ impossible, a non-zero polynomial of degree $n-1$ has at most $n-1$ roots

$\Rightarrow$ are all the $\alpha(ijk)$ values distinct ??

Let's assume $\alpha(ijk) = \alpha(i'j'k')$

so $N_{00}^{i} \times N_{01}^{j} \times N_{11}^{k} = N_{00}^{i'} \times N_{01}^{j'} \times N_{11}^{k'}$

and $4^i 7^i \times 2^j 11^j \times 17^k = 4^{i'} 7^{i'} \times 2^{j'} 11^{j'} \times 17^{k'}$

Now by uniqueness of the prime number decomposition we get $(ijk) = (i'j'k')$

→ so the matrix M of the system is invertible

→ so we can use the equation system to recover the desired values $|X_{ijk}|$ from the oracle results N_p (invert the matrix in PTIME)

→ and we get $|X| = \sum_{ij} |X_{ij}|$ showing #P-hardness

Note : here we used symmetry and got lucky with the numbers

General proof : more elaborate asymmetric coding plus rounding argument plus lemma showing that the "determinant" $N_{00}N_{11} - N_{01}N_{10}$ is $\neq 0$.

Dalvi, Suciu JACM '12

#P-hardness for UCQs where the algorithm fails

Recall that we had a complicated algorithm to compute the probability of UCQs
→ which failed when it could not find a separator variable

Thm: If the algorithm fails on a query Q then $\text{PQE}(Q)$ is #P-hard.

Very technical proof. High level ideas:

- Focus on queries on which the algorithm fails immediately: immediately unsafe query
  If the algorithm eventually fails on a query Q, then there is a query Q' on which it fails immediately s.t. $\text{PQE}(Q')$ reduces to $\text{PQE}(Q)$.
- Rewrite immediately unsafe queries to make them leveled:
  allows us to assume that the domain is k-partite and each atom $R(x_1, \dots, x_\ell)$ maps variable x_i to elements of one part only
  → show that the leveling is still immediately unsafe
  → we can assume that queries are leveled
- Define forbidden queries, the minimal immediately unsafe leveled queries, under some order (sequence of relation arities, and number of atoms)
  → The simpler queries to which a forbidden query "rewrites" are always safe by minimality
  → Show that forbidden queries have been simplified to have only two levels, ie. any leveled immediately unsafe query can be simplified further
  → A forbidden query is a disjunction of components with atoms $R(x)$, $S(x,y)$, $T(y)$ on a partitioned variable set $X \cup Y$
- Show that forbidden queries are #P-hard: different types (1-1, 1-2, 2-1, 2-2)
  → Very technical; uses the interpolation method

What about uniform reliability:

Kenig, Suciu PODS '21

Thm: If $\text{PQE}(Q)$ is #P-hard for a UCQ Q, this still holds with probas $\{0, \frac{1}{2}, 1\}$
Further, $\text{UR}(Q)$ is #P-hard for final type-1 queries.

Open problem: Is $\text{UR}(Q)$ #P-hard for every UCQ Q where $\text{PQE}(Q)$ is #P-hard?

A., Ceylan
ICDT '20

#P-hardness of PQE for unbounded UCQ$^\infty$

We now extend our study of PQE to queries beyond UCQs.
We keep a key property: closure under homomorphisms

Def: A query Q is closed under homomorphisms if for any instances I and I',
if $I \models Q$ and I has a homomorphism to I' ($I \xrightarrow{\text{hom}} I'$)
then $I' \models Q$.

In particular: UCQs, but also languages like Datalog or regular path queries (RPQs)

Def: A regular path query Q on an arity-2 signature σ is a regexp over σ
An instance I satisfies Q if there is a "path" of facts
$R_1(a_1,a_2)\, R_2(a_2,a_3)\, \dots,\, R_n(a_n,a_{n+1})$ such that the word $R_1 \cdots R_n$
is in the language of the regexp Q

RS^*T
$R(a_1,a_2)\, S(a_2,a_3)\, S(a_3,a_4)\, T(a_4,a_5)$

We write UCQ$^\infty$ the queries closed under homomorphism.
Why? Because they are like UCQs with infinitely many disjuncts
RS^*T : $R(x_1,x_2)T(x_2,x_3) \vee R(x_1,x_2)S(x_2,x_3)T(x_3,x_4) \vee R(x_1,x_2)S(x_2,x_3)S(x_3,x_4)T(x_4,x_5)$

Def: A UCQ$^\infty$ is unbounded if it is not expressible as a UCQ (infinitely many non-redundant disjuncts)
ex: RS^*T

Rem: A UCQ$^\infty$ is unbounded iff it has infinitely many minimal models
i.e. $I \models Q$ but for any $I' \subsetneq I$, $I' \not\models Q$

For technical reasons we restrict to arity-2 signatures (probabilistic graphs)
Work in progress: lifting the restriction

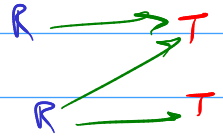
Theorem: For any unbounded UCQ$^\infty$ Q on an arity-2 signature, PQE(Q) is #P-hard.

Work in progress: what about UR?

Proof structure: first look at bipartite R-S-T instances then arbitrary instances

#P-hardness for unbounded UCQ^∞: bipartite R-S-T instances

Def: A bipartite RST instance I consists of:
- a bipartite domain $dom(I) = U \sqcup V$,
- R-facts on a subset $U' \subseteq U$ and T-facts on a subset $V' \subseteq V$
- S-facts on a subset $E \subseteq U \times V$



Def: A bipartite RST TID is a TID whose underlying instance is bipartite RST

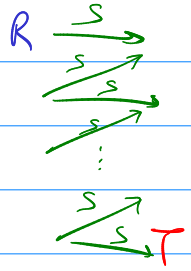
We will reduce from hard UCQ^∞ (bounded or unbounded) on bipartite RST instances

Q_1: $R(x)S(xy)T(y)$ #P-hard

Q_2: $R(x)S(xy)S(x'y)S(x'y')T(y')$

$\vdots$

Q_n: $R(x_1)S(x_1y_1)S(x_2y_1)S(x_2y_2)\cdots S(x_ny_n)T(y_n)$

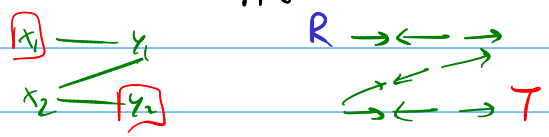


Lemma: For all $n > 0$, $PQE(Q_n)$ on bipartite RST TIDs is #P-hard

Proof: Usual hardness proof but using paths $\rightarrow\leftarrow\rightarrow\leftarrow\rightarrow\cdots\rightarrow$

Example for Q_2: reduce from #PP2DNF

Notice that non-satisfying valuations can create matches of Q_m but $m > n$

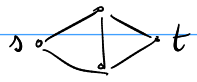


#P-hardness for unbounded UCQ$^\infty$: bipartite R-S-T instances (cont'd)

Now consider the unbounded query $Q_\infty = \bigvee_{n>0} Q_n$. $R(\rightarrow\leftarrow)^* T$

Lemma PQE(Q_∞) on bipartite RST TIDs is #P-hard

Note: cannot do the usual reduction! (because of the Q_m matches with $m > n$)



Def: #U-ST-CONN Undirected source-to-target connectivity problem

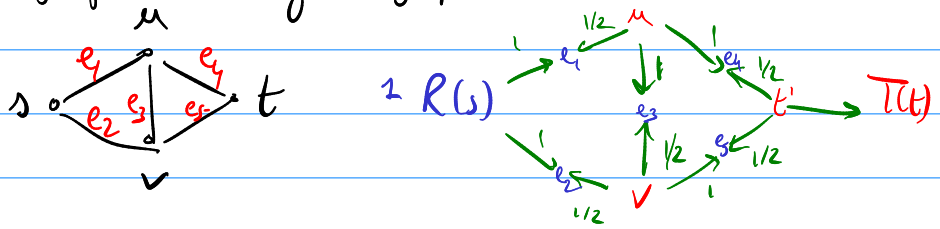
Input: undirected graph G, distinguished vertices s,t

Output: probability that s and t are connected (same connected component) when all edges have proba 1/2

Valiant SICOMP'79

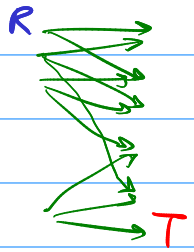
Prop: U-ST-CONN is #P-hard

Proof of lemma: given graph G code it to J:



Note: $\rightarrow\leftarrow$ (1, 1/2) or (1/2, 1) is arbitrary

- one extra edge $t' \rightarrow T(t)$
- is bipartite



Correctness: Probability-preserving bijection between possible worlds

⇒ if a possible world connects s and t then following any path gives a query match

⇒ likewise, any match of Q_∞ in a possible world of J gives a path

Now let's move to arbitrary graphs, reducing from bipartite RST TIDs

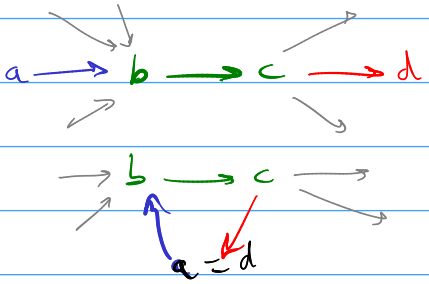
#P-hardness for unbounded UCQ$^\infty$: RST-patterns

We consider instances over a signature σ of arity 2 (edge colors).
Edge orientation ignored for simplicity. Arity-1 facts also ignored.

Def: A RST-pattern in an instance I is a choice of 3 facts

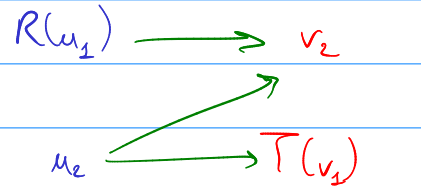
$U(a,b)$
$V(b,c)$
$W(c,d)$

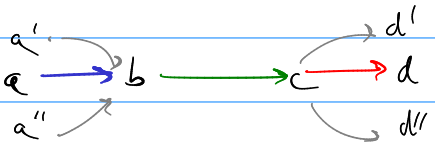
Further, the 4 elements are distinct except maybe $a=d$.

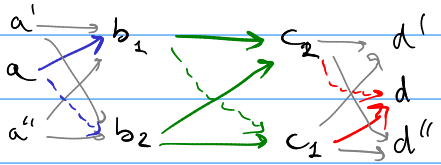


We can use RST patterns to code an RST bipartite instance:

Def: Let I' be a bipartite RST instance
$\{u_1, \dots, u_n\}$ left vertices
$\{v_1, \dots, v_m\}$ right vertices
R-facts $R(u_1), \dots, R(u_i)$
T-facts $T(v_1), \dots T(v_j)$
S-facts $S(u,v)$ for $(u,v) \in E$
Let I be a graph instance
and a,b,c,d be an RST-pattern
The coding of I' in I is the graph instance
where we copy b to $b_1, \dots, b_n$
copy c to $c_1, \dots, c_m$
copy $U(a,b)$ to $U(a,b_1), \dots, U(a,b_i)$
copy $V(b,c)$ according to E
copy $W(c,d)$ to $W(c_1,d), \dots, W(c_j,d)$
copy all other facts "everywhere"







note: dashes ---- means "all facts except $V(b,c)$"

To code a bipartite RST TID, we give the proba of each $R(u_p)$ to $U(a,b_p)$
of each $T(v_q)$ to $W(c_q,d)$
of each $S(u,v)$ to its V-fact
and all other probas are 1.

Remark: If $I' \xrightarrow{\text{hom}} I''$ (on bipartite RST instances) [preserves homomorphisms]
then coding$(I', I) \xrightarrow{\text{hom}}$ coding(I'', I)

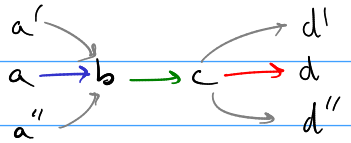
#P-hardness for unbounded UCQ$^\infty$: RST patterns (cont'd)

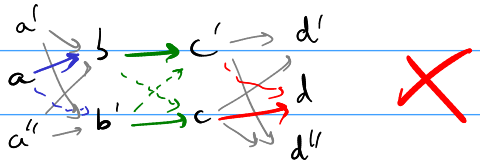
Consider a UCQ$^\infty$ Q. Fix an instance I and RST pattern $U(a,b)\,V(b,c)\,W(c,d)$

Def: The RST pattern is critical if:
- $I \models Q$
- $\text{coding}(I'_0, I) \not\models Q$ for I'_0 the bipartite RST instance

$R \rightarrow$

$\rightarrow T$





Intuition: "the query cares about connecting $\rightarrow$ and $\rightarrow$ by $(\rightarrow\leftarrow)^*\rightarrow$"

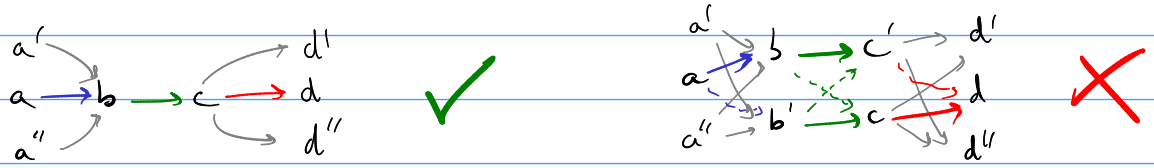
Two results to show

Thm: [Hardness of critical patterns]
If we have a critical pattern for Q then $\text{PQE}(Q)$ is #P-hard

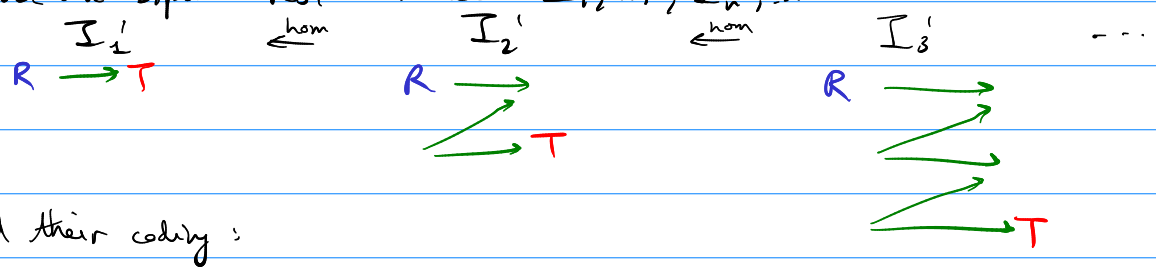
Thm: [Finding a critical pattern]
Every unbounded UCQ$^\infty$ on graphs has a critical pattern.

#P-hardness for unbounded UCQ$^\infty$: hardness of critical patterns

Fix the query Q, instance I, and critical pattern:

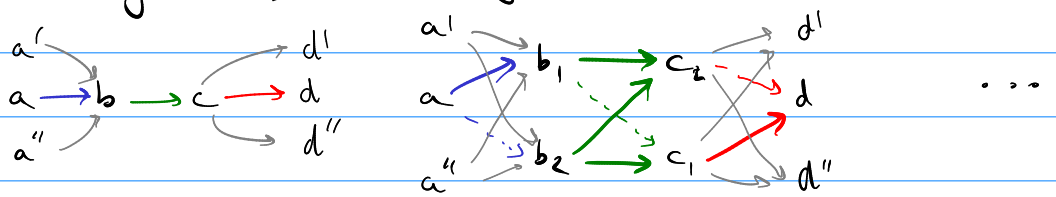


Consider the bipartite RST instances $I'_1, \dots, I'_n, \dots$



and their codings:

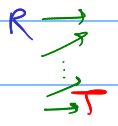
Coding (I'_1, I) $\xleftarrow{\text{hom}}$ Coding (I'_2, I)



Two cases:

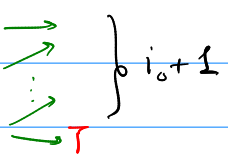
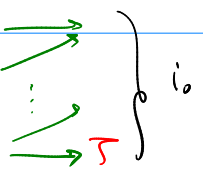
1. Coding (I'_i, I) satisfies the query ✓

Reduce from PQE(Q_∞) for the connectivity query Q_∞ on a bipartite RST TID I':

- A possible world of I' violating Q_∞ has a homomorphism to  so the world in the coding violates Q
- A possible world of I' satisfying Q_∞ has a homomorphism from some I'_i  so the world in the coding satisfies Q

2. There is a threshold i_0, Coding (I'_i, I) satisfies the query for $i \le i_0$ ✓; Coding (I_i, I) violates the query for $i > i_0$ ✗

Reduce from PQE(Q_{i_0}) for the distance-i_0 query Q_{i_0} on a bipartite RST TID I':

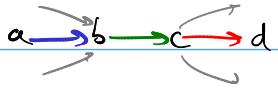
- A possible world of I' violating Q_{i_0} has a homomorphism to  (i_0+1)
- A possible world of I' satisfying Q_{i_0} has a homomorphism from  (i_0)

#P-hardness for unbounded UCQ$^\infty$: finding a critical pattern

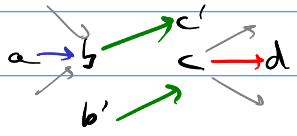
Fix an unbounded UCQ$^\infty$ Q and instance I with $I \models Q$

First find a tight pattern then get a critical pattern

Def: The dissociation of an RST pattern $U(a,b)\,V(b,c)\,W(c,d)$ in I is obtained by:



- copying b to b' and c to c'
- moving all facts $\{b,c\}$ including $S(b,c)$ to $\{b,c'\}$ and $\{b',c\}$



Note: this is not a coding!

Note: the resulting instance has a homomorphism back to I

Def: An RST pattern $U(a,b)\,V(b,c)\,W(c,d)$ in I is tight if $I \models Q$ but the dissociation of the pattern violates Q.

Two remaining claims:

Thm: [Finding a tight pattern]

Any unbounded UCQ$^\infty$ on graphs has a tight pattern

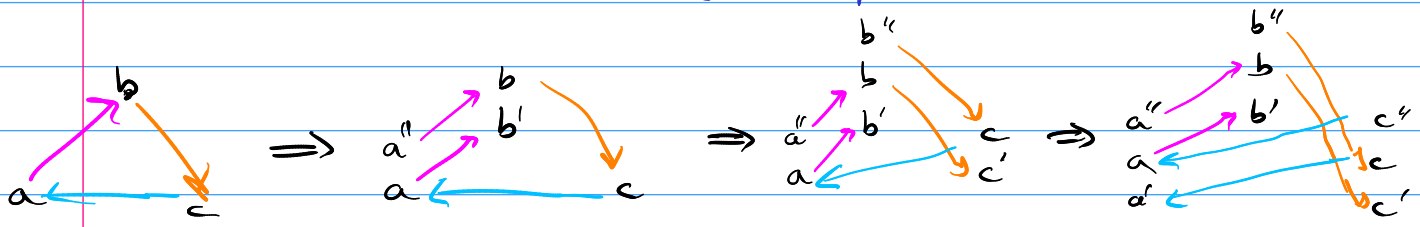
(This is where we really use the unboundedness of the query.)

Thm: [From tight to critical]

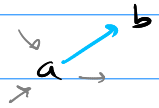
If a query has a tight pattern then it has a critical pattern

#P-hardness for unbounded UCQ$^\infty$: finding a tight pattern

High-level idea:
- Take large minimal model of the query
- Iteratively dissociate edges
- Argue for termination
- Show that if the query is still true at the end then this violates minimality



Why does it terminate?

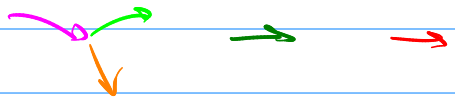
Def: Leaf edge $\{a,b\}$ 

all facts involving b also involve a

Dissociation takes a non-leaf edge (content of an RST-pattern) and replaces it by two leaf edges
⇒ the number of non-leaf edges decreases

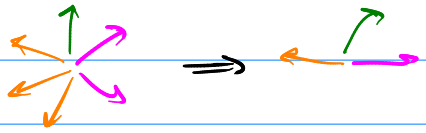
Cor: Any iterative dissociation process on an instance must terminate

What are instances with no RST pattern left to dissociate? → Only leaf edges
[↔ hierarchical queries]

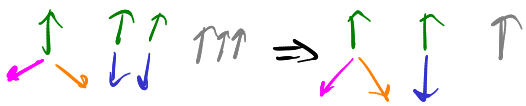


Lemma: An instance I with only leaf edges has a self-homomorphism to a constant-sized subset: call S_0 the maximal size.

- Finitely many possible fact combinations for an edge → (depends on signature)
  → let K be this number
- A connected component (star pattern) has a homomorphism to a subset of K edges (merge isomorphic branches)



- A disjoint union of star patterns has a homomorphism to a constant number of stars (merge isomorphic stars)



#P-hardness for unbounded UCQ^∞: finding a tight pattern (cont'd)

Let's conclude the proof: any unbounded UCQ^∞ Q has a tight pattern.

As Q is unbounded, it has a minimal model I of size $> S_0$ $\quad I \models Q$

Iteratively dissociate until it is no longer possible: $I = I_0 \xleftarrow{\text{hom}} I_1 \xleftarrow{\text{hom}} \cdots \xleftarrow{\text{hom}} I_n$

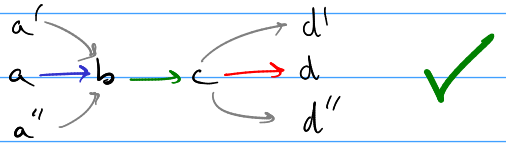
There are two possibilities:

1. $I_{i_0} \models Q$ but $I_{i_0+1} \not\models Q$ for some i_0
   → we have found a tight pattern
2. $I_n \models Q$
   → let us show a contradiction
   We know that I_n has only leaf edges
   so $I_n \xrightarrow{\text{hom}} I_{n'}$ with $I_{n'} \subseteq I_n$ and $|I_{n'}| \leq S_0$ [use lemma]
   as $I_n \models Q$, we have $I_{n'} \models Q$
   as $I_{n'} \subseteq I_n$, we have $I_{n'} \xrightarrow{\text{hom}} I_n$
   Thus, as $I_n \xrightarrow{\text{hom}} I_0 = I$ (homomorphism composition)
   we have $I_{n'} \xrightarrow{\text{hom}} I$, with $|I_{n'}| \leq S_0$ and $I_{n'} \models Q$
   Take I' the image of $I_{n'}$ in I by the homomorphism
   we have $I' \subseteq I$ with $|I'| \leq |I_{n'}| \leq S_0$ and $I' \models Q$
   → This is impossible: it contradicts the minimality of I.

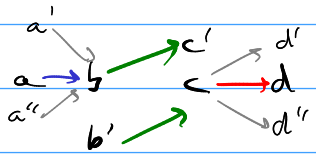
⇒ So we have found our tight pattern.

#P-hardness for unbounded UCQ$^{\infty}$: from tight to critical

We know our query Q has a tight pattern

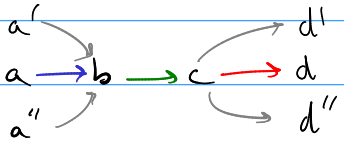


✓

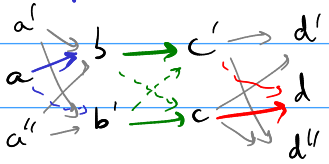


✗

but what we need for hardness is a critical pattern



✓

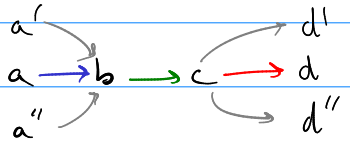


✗

Idea: Minimality criterion:

Take a tight pattern minimizing the following weight (lexicographically):

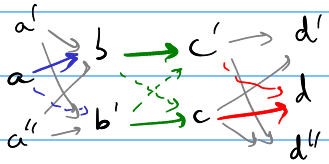
- the number of facts in $b \to c$
- the total number of incident facts in



Prop: Such a minimal tight pattern is in fact critical

for any choice of $U(a,b)$ $V(b,c)$ $W(c,d)$ in $a \to b$ $b \to c$ $c \to d$.

Proof: assume by contradiction that the query still holds when doing the coding



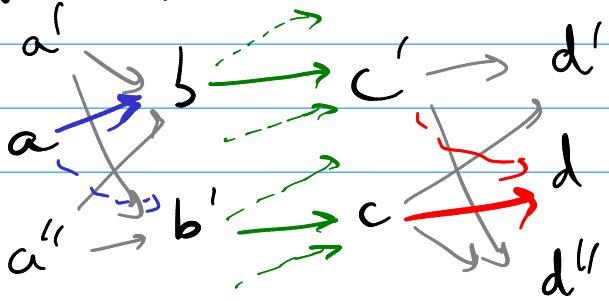
✓

The edges $b \dashrightarrow c$ and $b' \dashrightarrow c'$ have one fact less than $b \to c$ in the original instance: $V(b,c)$ is missing.

→ So we can dissociate them without breaking the query

(or it would give a tight pattern of smaller weight)

We get:

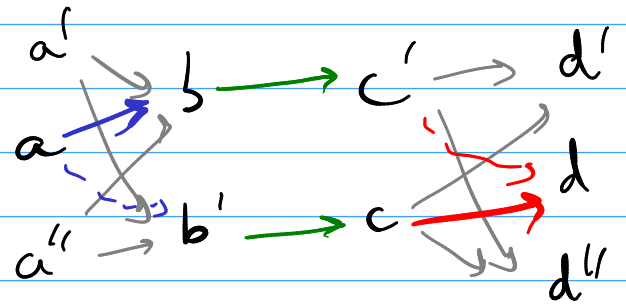


✓

(cont'd)

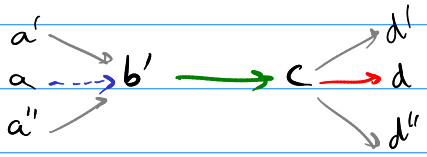
#P-hardness for unbounded UCQ$^\infty$: from tight to critical

We can merge the ⇢ into → with a homomorphism and Q still holds:



✓

Now consider



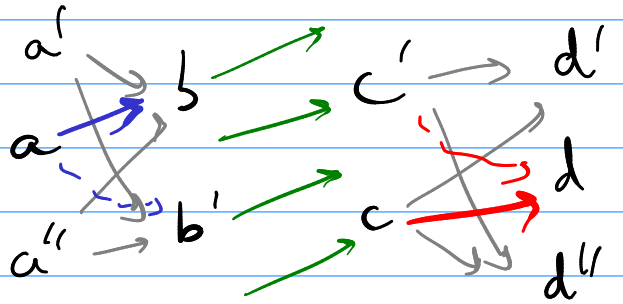
This has smaller weight than the minimal tight pattern:

the incident fact $U(a,b)$ is missing from $a \dashrightarrow b$

→ so we can dissociate without breaking the query

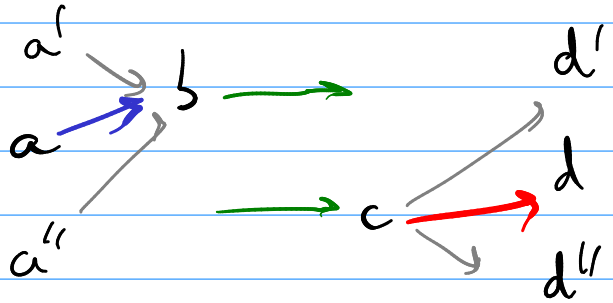
Same reasoning on $b \to c'$

We get:



✓

We homomorphically merge b and b', and c and c', and get:



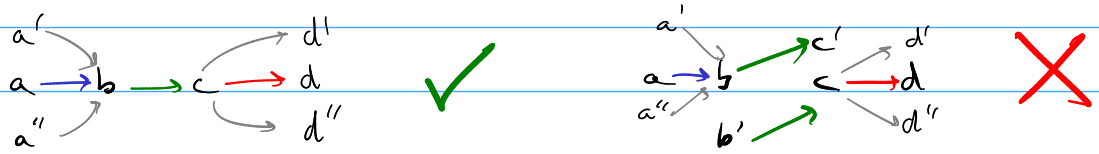
✓

But this is the dissociation, which by assumption violates the query (tightness) ⇒ contradiction!

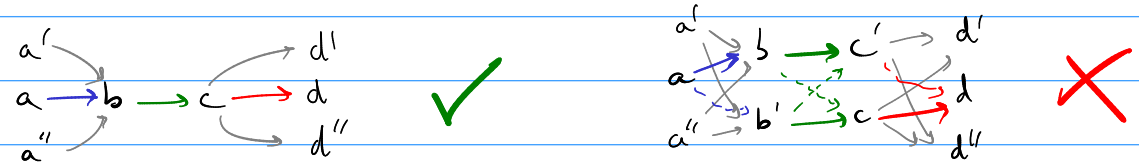
So in fact the query was wrong all along, and we have found our critical pattern.

#P-hardness for unbounded UCQ^∞: recap

We have found a tight pattern (by iteratively dissociating)



We have shown that a minimal tight pattern is in fact critical



We have shown how to use a tight pattern to show hardness
(by reducing from PQE(Q_k) or PQE(Q_∞) on bipartite RST TIDs)

So we have indeed shown that PQE(Q) is #P-hard for our unbounded UCQ^∞ Q

Bonus 1 : Treelike instances

Not many queries are tractable ⇒ restrict the shape of the data

Def : Treewidth of a graph

Treewidth of a relational instance ⇒ that of its Gaifman graph

We can allow expressive queries in Monadic second-order logic
→ first-order logic (in particular UCQs, negation) plus quantification over sets

Courcelle, IEC '90

Thm : Evaluating a fixed MSO query on an input instance whose treewidth is $\leq k$ for a constant k (treelike instance) can be done in linear time data complexity

Proof : Encode the query to a tree automaton running on a tree encoding of the tree decomposition

A., Bourhis, Senellart, ICALP'15, PODS'16

Thm : We can build in linear time a provenance d-D of the query on the instance

Corollary : PQE for MSO queries over treelike instances is in linear time

Bonus 2 : Lower bound on unbounded treewidth

What if the treewidth is not bounded? Can we use cliquewidth etc. ?

⇒ Restrict to arity-2 instances for technical reasons

Def : Treewidth-constructible instance family

Infinite family $\mathcal{I}$ of arity-2 instances such that, given $k \in \mathbb{N}$, we can build in PTIME in k an instance $I \in \mathcal{I}$ such that treewidth$(I) \geq k$

ex : All planar instances, all instances containing a triangle ...

Intuition: Prevent that the growth in treewidth is "too slow" wrt the instance size

Def : For a query Q and family $\mathcal{I}$, $PQE(Q, \mathcal{I})$ is $PQE(Q)$ where the input TID must be an instance of $\mathcal{I}$ (with arbitrary probabilities)

→ in particular we can close $\mathcal{I}$ under subgraphs

A., Monet, submitted, 2022 (see A, Bourhis, Senellart, PODS'16)

Theorem : For $Q_{\neq}$ the query testing if there are two incident facts ($\exists x y z \, R(x,y) \wedge S(y,z)$), for any treewidth-constructible family $\mathcal{I}$, $PQE(Q_{\neq}, \mathcal{I})$ is #P-hard under ZPP reductions

Key tool : finding arbitrary degree-3 planar graphs in a high-treewidth graph

via Chekuri and Chuzhoy, JACM '16

Thm : There is $c \in \mathbb{N}$ such that the following holds:
Given a degree-3 planar graph H and a graph G with treewidth $|H|^c$, we can embed G in H as a topological minor in ZPP.

Then show that $Q_{\neq}$ is #P-hard on arbitrary subdivisions of an input graph

A., Monet, Senellart ICDT '18

Also lower bound on circuits : no polynomial-sized d-SDNNFs

Summary and outlook

What is the complexity of probabilistic query evaluation?

- On SJFCQs, well-understood
  - Hierarchical ⇒ tractable, via circuits
  - Non-hierarchical ⇒ intractable, even for UR
- On UCQs, understood, but...
  - PTIME algorithm which may fail (?) Circuits?
  - #P-hardness when the algorithm fails (?) UR?
- On UCQ$^\infty$, intractability except for UCQs (?) Higher-arity? (?) UR?

Not covered:

- Queries with negation [Fink, Olteanu, TODS '16]
- Queries with inequalities [Olteanu, Huang, SIGMOD '09]
- Provenance circuit bounds for other lineage classes [Jha, Suciu, TCS '13]
- Results on infinite domains [Carmeli, Grohe, Lindner, Standke, PODS '21]
- Support for updates [Berkholz, Merz, PODS '21]
- Symmetric model counting [Beame, Van den Broeck, Gribkoff, Suciu, PODS '15]
- Practical implementation: see ProvSQL [Senellart, ongoing]

Other directions for future research:

- Approximations (beyond additive approximations)
- Reusability of techniques? (enumeration, Shapley values...) — Livshits, Bertossi, Kimelfeld, Sebag, LMCS '21
- Connection to reasoning, e.g. probabilistic programming

Thanks for your attention! Questions?