Uniform Reliability of Self-Join-Free Conjunctive Queries

Antoine Amarilli, Benny Kimelfeld

Uniform Reliability of Self-Join-Free Conjunctive Queries

Antoine Amarilli
LTCI, Télécom Paris, Institut Polytechnique de Paris, France

Benny Kimelfeld
Technion - Israel Institute of Technology, Haifa, Israel

Abstract

The *reliability* of a Boolean Conjunctive Query (CQ) over a tuple-independent probabilistic database is the probability that the CQ is satisfied when the tuples of the database are sampled one by one, independently, with their associated probability. For queries without self-joins (repeated relation symbols), the data complexity of this problem is fully characterized in a known dichotomy: reliability can be computed in polynomial time for *hierarchical* queries, and is #P-hard for non-hierarchical queries. Hierarchical queries also characterize the tractability of queries for other tasks: having read-once lineage formulas, supporting insertion/deletion updates to the database in constant time, and having a tractable computation of tuples' Shapley and Banzhaf values.

In this work, we investigate a fundamental counting problem for CQs without self-joins: how many sets of facts from the input database satisfy the query? This is equivalent to the *uniform* case of the query reliability problem, where the probability of every tuple is required to be 1/2. Of course, for hierarchical queries, uniform reliability is in polynomial time, like the reliability problem. However, it is an open question whether being hierarchical is necessary for the uniform reliability problem to be in polynomial time. In fact, the complexity of the problem has been unknown even for the simplest non-hierarchical CQs without self-joins.

We solve this open question by showing that uniform reliability is #P-complete for every non-hierarchical CQ without self-joins. Hence, we establish that being hierarchical also characterizes the tractability of unweighted counting of the satisfying tuple subsets. We also consider the generalization to query reliability where all tuples *of the same relation* have the same probability, and give preliminary results on the complexity of this problem.

2012 ACM Subject Classification Theory of computation → Database query processing and optimization (theory)

Keywords and phrases Hierarchical conjunctive queries, query reliability, tuple-independent database, counting problems, #P-hardness

Digital Object Identifier 10.4230/LIPIcs.ICDT.2021.17

Related Version *Full Version*: `https://arxiv.org/abs/1908.07093` [1]

Funding The work of Benny Kimelfeld was supported by the Israel Science Foundation (ISF), Grant 768/19, and the German Research Foundation (DFG) Project 412400621 (DIP program).

Acknowledgements The authors are very grateful to Kuldeep S. Meel and Dan Suciu for insightful discussions on the topic of this paper.

1 Introduction

Probabilistic databases [23] extend the usual model of relational databases by allowing database facts to be uncertain, in order to model noisy and imprecise data. The evaluation of a Boolean query Q over a probabilistic database D is then the task of computing the probability that Q is true under the probability distribution over possible worlds given by D. This computational task has been considered by Grädel, Gurevich and Hirsch [10] as a special case of computing the *reliability* of a query in a model which is nowadays known as

Tuple-Independent probabilistic Databases (TIDs) [6, 23]. In a TID, every fact is associated with a probability of being true, and the truth of every fact is an independent random event. While the TID model is rather weak, query evaluation over TIDs can also be used for probabilistic inference over models with correlations among facts, such as Markov Logic Networks [11, 13]. Hence, studying the complexity of query evaluation on TIDs is the first step towards understanding which forms of probabilistic data can be tractably queried.

To this end, Grädel et al. [10] showed the first Boolean Conjunctive Query (referred to simply as a *CQ* hereafter) for which query evaluation is #P-hard on TIDs. Later, Dalvi and Suciu [6] established a dichotomy on the complexity of evaluating CQs without self-joins (i.e., without repeated relation symbols) over TIDs: if the CQ is *safe* (or *hierarchical* [8, 23] as we explain next), the problem is solvable in polynomial time; otherwise, the problem is #P-hard. (This result was later extended to the class of all CQs and unions of CQs [7].)

The class of *hierarchical* CQs is defined by requiring that, for every two variables x and y, the sets of query atoms that feature x must contain, be contained in, or be disjoint from, the set of atoms that feature y. Interestingly, this class of hierarchical queries was then found to characterize the tractability boundary of other query evaluation tasks for CQs without self-joins, over databases without probabilities (and under conventional complexity assumptions). Olteanu and Huang [18] showed that a query is hierarchical if and only if, for every database, the *lineage* of the query is a read-once formula. Livshits, Bertossi, Kimelfeld and Sebag [15] proved that the hierarchical CQs are precisely the ones that have a tractable *Shapley value* as a measure of responsibility of facts to query answers (a result that was later generalized to CQs with negation [15]); they also conjecture that this complexity classification also holds for another measure of responsibility, namely the *causal effect* [21]. (We discuss these measures again later in this section.) Berkholz, Keppeler and Schweikardt [3] showed that the hierarchical CQs are (up to conventional assumptions of fine-grained complexity) precisely the ones for which we can use an auxiliary data structure to update the query answer in constant time in response to the insertion or deletion of a tuple.

In this paper, we show that the property of being hierarchical also captures the complexity of a fundamental counting problem for CQs without self-joins: *how many sets of facts from the input database satisfy the query?* This problem, which we refer to as *uniform reliability*, is equivalent to query evaluation over a TID where the probability of *every* fact is equal to $\frac{1}{2}$. In particular, it follows from the aforementioned dichotomy that this problem can be solved in polynomial time for every self-join-free hierarchical CQ Q. Yet, if Q is not hierarchical, it does not necessarily mean that Q is intractable already in this uniform setting. Indeed, it was not known whether enforcing uniformity makes query evaluation on TIDs easier, and the complexity of uniform reliability was already open for the simplest case of a non-hierarchical CQ: $Q_1 \coloneq \mathsf{R}(x), \mathsf{S}(x, y), \mathsf{T}(y)$. The proofs of #P-hardness of Dalvi and Suciu [6] require TIDs with deterministic facts (probability 1), in addition to $\frac{1}{2}$, already in the case of Q_1. Here, we address this problem and show that the dichotomy is also true for the uniform reliability problem. In particular, uniform reliability is #P-complete for every non-hierarchical CQ without self-joins (and solvable in polynomial time for every hierarchical CQ without self-joins).

The uniform reliability problem that we study is a basic combinatorial problem on CQs, and a natural restricted case of query answering on TIDs, but it also has a direct application for quantifying the impact (or responsibility) of a fact f on the result of a CQ Q over ordinary (non-probabilistic) databases. One notion of tuple impact is the aforementioned causal effect, defined as the difference between two quantities: the probability of Q conditioning on the existence of f, minus the probability of Q conditioning on the absence of f [21]. This causal

effect was recently shown [3] to be the same as the *Banzhaf power index*, studied in the context of wealth distribution in cooperative game theory [9] and applied, for instance, to voting in the New York State Courts [12]. One notion of causal effect (with so-called *endogenous* facts) is defined by viewing the ordinary database as a TID where the probability of every fact is $\frac{1}{2}$. Therefore, computing the causal effect amounts to solving two variations of uniform reliability, corresponding to the two quantities. In fact, it is easy to see that all of our results apply to each of these two variations.

Uniform reliability also relates to the aforementioned computation of a tuple's Shapley value, a measure of wealth distribution in cooperative game theory that has been applied to many use cases [20, 22]. Livshits et al. [15] showed that computing a tuple's Shapley value can be reduced to a generalized variant of uniform reliability. Specifically, for CQs, computing the Shapley value (again for *endogenous* facts) amounts to calculating the number of subinstances that satisfy Q *and* have precisely m tuples (for a given number m). This generalization of uniform reliability is tractable for every hierarchical CQ without self-joins [15]. Clearly, our results here imply that this generalization is intractable for every non-hierarchical CQ without self-joins, allowing us to conclude that the complexity dichotomy also applies to this generalization.

Our investigation can be viewed as a first step towards the study of problems that lie in between uniform reliability and probabilistic query answering over TIDs. For instance, a natural variant is the one where the probability of each tuple of the database is the same, but not necessarily $\frac{1}{2}$. This problem can arise, for example, in scenarios of network reliability, where all connections are equally important and have the same independent probability of failure. A more general case is the one where the probabilities for every relation are the same, but different relations may be associated with different probabilities. This corresponds to data integration scenarios where every relation is a resource with a different level of trust (e.g., enterprise data vs. Web data vs. noisy sensor data). In this paper, we formalize this generalization and ask which combinations of CQs and probability assignments make the problem intractable. We do not completely answer this question, but propose preliminary results for the query Q_1 mentioned earlier: we show that some combinations of probabilities can be easily proved hard using our main result, while others can be proved hard with other techniques.

Related work. As explained earlier, our work is closely related to existing literature on query evaluation over probabilistic databases. The dichotomy of Dalvi and Suciu [6] for CQs without self-joins requires tuples with probabilities $\frac{1}{2}$ and 1. This is also the case for their generalized dichotomy on CQs without self-joins where some relations can be required to be deterministic (i.e., all tuples have probability 1) while tuples in the remaining relations can have arbitrary probabilities (including 1). The later generalization of the dichotomy by Dalvi and Suciu [7] to CQs with self-joins and to UCQs required an unbounded class of probabilities, not just $\frac{1}{2}$ and 1. In very recent work, Kenig and Suciu [14] have strengthened the generalized dichotomy and showed that probabilities $\frac{1}{2}$ and 1 suffice for UCQs as well.[1] In that work, they also investigate uniform reliability (that we study here, i.e., where $\frac{1}{2}$ is the only nonzero probability allowed) and prove #P-hardness for the so-called unsafe "final type-I" queries. As they explain in their discussion on the work of this paper (which was posted as a preprint before theirs), their result on uniform reliability complements ours, and it is not clear if any of these two results can be used to prove the other.

[1] Kenig and Suciu refer to this case as TID with probabilities from $\{0, \frac{1}{2}, 1\}$; we mean the same thing, as in this paper we assume that tuples with probability zero are simply ignored.

The work of this paper also relates to rewriting techniques used in the case of DNF formulas to reduce weighted model counting to unweighted model counting [5]. Nevertheless, the results and techniques for this problem are not directly applicable to ours, since model counting for CQs translates to DNFs of a very specific shape (namely, those that can be obtained as the lineage of the query).

Another superficially related problem is that of *symmetric model counting* [2]. This is a variant of uniform reliability where each relation consists of *all possible tuples* over the corresponding domain, and so each fact carries the same weight: these assumptions are often helpful to make model counting tractable. The assumption that we make is much weaker: we do not deal with symmetric databases, but rather with arbitrary databases where all facts of the database (but *not* necessarily all possible facts over the domain) have the same uniform probability of $\frac{1}{2}$. For this reason, the tractability results of Beame et al. [2] do not carry over to our setting. In terms of hardness results, [2, Theorem 3.1] shows the $\#\mathrm{P}_1$-hardness of symmetric model counting (hence of uniform reliability) for a specific FO^3 sentence, and [2, Corollary 3.2] shows a $\#\mathrm{P}_1$-hardness result for *weighted* symmetric model counting for a specific CQ (without assuming self-join-freeness). Hence, these results do not determine the complexity of uniform reliability for self-join-free CQs as we do here.

There is a closer connection to existing dichotomy results on counting *database repairs* [16, 17]. In this setting, the input database may violate the primary key constraints of the relations, and a repair is obtained by selecting one fact from every collection of conflicting facts (i.e., distinct facts that agree on the key): the *repair counting problem* asks how many such repairs satisfy a given CQ. In particular, it can easily be shown that for a CQ Q, there is a reduction from the uniform reliability of Q to repair counting of another CQ Q'. Yet, this reduction can only explain cases of tractability (namely, where Q is hierarchical) which, as explained earlier, are already known. We do not see how to design a reduction in the other direction, from repair counting to uniform reliability, in order to show our hardness result.

Finally, our work relates to the study of the Constraint Satisfaction Problem (CSP). However, there are two key differences. First, we study query evaluation in terms of homomorphisms *from* a fixed CQ, whereas the standard CSP phrasing talks about homomorphisms *to* a given template. Second, the standard counting variant of CSP (namely, #CSP), for which Bulatov has proved a dichotomy [4], is about counting *the number of homomorphisms*, whereas we count *the number of subinstances* for which a homomorphism exists. For these reasons, it is not clear how results on CSP and #CSP can be helpful towards our main result.

Organization. We give preliminaries in Section 2. In Section 3, we formally state the studied problem and main result, that is, the dichotomy on the complexity of uniform reliability for CQs without self-joins. We prove this result in Sections 4–6. We discuss a generalization to arbitrary uniform probabilities in Section 7, and conclude in Section 8. Missing proofs can be found in the full version of this paper [1].

2 Preliminaries

We begin with some preliminary definitions and notation that we use throughout the paper. We first define databases and conjunctive queries, before introducing the task of probabilistic query evaluation, and the uniform reliability problem that we study.

Databases. A (relational) *schema* $\mathbf{S}$ is a collection of *relation symbols* with each relation symbol ρ in $\mathbf{S}$ having an associated arity. We assume a countably infinite set Const of *constants* that are used as database values. A *fact* over $\mathbf{S}$ is an expression of the form

$\rho(c_1, \ldots, c_k)$ where ρ is a relation symbol of $\mathbf{S}$, where k is the arity of ρ, and where $c_1, \ldots, c_k$ are values of Const. An *instance* I over $\mathbf{S}$ is a finite set of facts. In particular, we say that an instance J is a *subinstance* of an instance I if we have $J \subseteq I$.

Conjunctive queries. This paper focuses on queries in the form of a Boolean Conjunctive Query, which we refer to simply as a *CQ*. Intuitively, a CQ Q over the schema $\mathbf{S}$ is a relational query definable as an existentially quantified conjunction of atoms. Formally, a CQ is a first-order formula of the form $Q \text{ :- } \rho_1(\vec{\tau}_1), \ldots, \rho_n(\vec{\tau}_m)$ where each $\rho_i(\vec{\tau}_i)$ is an *atom* of Q, formed of a relation symbol of $\mathbf{S}$ and of a tuple $\vec{\tau}_i$ of constants and (existentially quantified) variables, with the same arity as ρ_i. In the context of a CQ Q, we omit the schema $\mathbf{S}$ and implicitly assume that $\mathbf{S}$ consists of the relation symbols that occur in Q (with the arities that they have in Q); in that case, we may also refer to an instance I over $\mathbf{S}$ as an instance *over* Q. We write $I \models Q$ to state that the instance I satisfies Q. We denote by the set of all subinstances J of I that satisfy Q by:

$$\mathrm{Mod}(Q, I) := \{J \subseteq I \mid J \models Q\}.$$

A *self-join* in a CQ Q is a pair of distinct atoms over the same relation symbol. For example, in $Q \text{ :- } \mathsf{R}(x, y), \mathsf{S}(x), \mathsf{R}(y, z)$, the first and third atoms constitute a self-join. Our analysis in this paper is restricted to CQs *without self-joins*, that we also call *self-join-free*.

Let Q be a CQ. For each variable x of Q, we denote by $\mathrm{atoms}(x)$ the set of atoms $\rho_i(\vec{\tau}_i)$ of Q where x occurs. We say that Q is *hierarchical* [6] if for all variables x and x' one of the following three relations hold: $\mathrm{atoms}(x) \subseteq \mathrm{atoms}(x')$, $\mathrm{atoms}(x') \subseteq \mathrm{atoms}(x)$, or $\mathrm{atoms}(x) \cap \mathrm{atoms}(x') = \emptyset$. The simplest non-hierarchical self-join-free CQ is Q_1, which we already mentioned in the introduction:

$$Q_1 \text{ :- } \mathsf{R}(x), \mathsf{S}(x, y), \mathsf{T}(y) \tag{1}$$

Probabilistic query evaluation. The problem of *probabilistic query evaluation* over tuple-independent databases [23] is defined as follows.

▶ **Definition 2.1.** *The problem of* probabilistic query evaluation *(or PQE) for a CQ Q, denoted $\mathsf{PQE}(Q)$, is that of computing, given an instance I over Q and an assignment $\pi : I \to [0, 1]$ of a probability $\pi(f)$ to every fact f, the probability that Q is true, namely:*

$$\Pr(Q, I, \pi) := \sum_{J \in \mathrm{Mod}(Q,I)} \prod_{f \in J} \pi(f) \times \prod_{f \in I \setminus J} (1 - \pi(f)) .$$

We again study the *data complexity* of this problem, and we assume that the probabilities attached to the instance I are rational numbers represented by their integer numerator and denominator.

PQE was first studied by Grädel, Gurevich and Hirsch [10] as *query reliability* (which they also generalize beyond Boolean queries). They identified a Boolean CQ Q with self-joins such that the reliability of Q is #P-hard to compute. Dalvi and Suciu [6, 7] then studied the PQE problem, culminating in their dichotomy for the complexity of PQE on unions of conjunctive queries with self-joins [7]. In this paper, we only consider their earlier study of CQs without self-joins [6]. They characterize, under conventional complexity assumptions, the self-join-free CQs where PQE is solvable in PTIME. They state the result in terms of safe query plans ("safe CQs"), but the term "hierarchical" was adopted in later publications [8, 23]:

▶ **Theorem 2.2.** [6] *Let Q be a CQ without self-joins. If Q is hierarchical, then $\mathsf{PQE}(Q)$ is solvable in polynomial time. Otherwise, $\mathsf{PQE}(Q)$ is #P-hard.*

Recall that #P is the complexity class of problems that count witnesses of an NP-relation (e.g., satisfying assignments of a logical formula, vertex covers of a graph, etc.). A function F is #P-hard if every function in #P has a polynomial-time *Turing reduction* (or *Cook reduction*) to F.

We stress that Theorem 2.2 applies to CQs *without* self-joins. In the presence of self-joins, being hierarchical is still necessary for tractability, but no longer sufficient [23, Theorem 4.23, Proposition 4.25].

Uniform reliability. We study the query reliability problem (which we equivalently refer to as PQE), and focus on the uniform variant of this problem, where the probability of every fact is $\frac{1}{2}$. Equivalently, the task is to count the subinstances that satisfy the query (up to division/multiplication by 2^n where n is the number of facts in the instance). Formally:

▶ **Definition 2.3.** *The problem of* uniform reliability *for a CQ Q, denoted $\mathsf{UR}(Q)$, is that of determining, given an instance I over Q, how many subinstances of I satisfy Q. In other words, $\mathsf{UR}(Q)$ is the problem of computing $|\mathrm{Mod}(Q, I)|$ given I. We study the* data complexity *of this problem, i.e., Q is fixed and the complexity is a function of the input I.*

3 Problem Statement and Main Result

Let Q be a CQ without self-joins. It follows from Theorem 2.2 that, if Q is hierarchical, then $\mathsf{UR}(Q)$ is solvable in polynomial time. Indeed, there is a straightforward reduction from $\mathsf{UR}(Q)$ to $\mathsf{PQE}(Q)$: given an instance I for Q, let $\pi : I \to [0, 1]$ be the function that assigns to every fact of I the probability $\pi(f) = \frac{1}{2}$. Then we have:

$$|\mathrm{Mod}(Q, I)| = 2^{|I|} \times \Pr(Q, I, \pi)$$

because every subset of Q has the same probability, namely $2^{-|I|}$.

However, the other direction is not evident. If Q is non-hierarchical, we know that $\mathsf{PQE}(Q)$ is #P-hard, but we do not know whether the same is true of $\mathsf{UR}(Q)$. Indeed, this does not follow from Theorem 2.2 (as uniform reliability is a restriction of PQE), and it does not follow from the proof of the theorem either. Specifically, the reduction that Dalvi and Suciu [6] used to show hardness consists of two steps.

1. Proving that $\mathsf{PQE}(Q_1)$ is #P-hard (where Q_1 is defined in (1)).
2. Constructing a polynomial-time Turing reduction from $\mathsf{PQE}(Q_1)$ to $\mathsf{PQE}(Q)$ for every non-hierarchical CQ Q without self-joins.

In both steps, the constructed instances I consist of facts with two probabilities: $\frac{1}{2}$ and 1 (i.e., *deterministic facts*). If all facts had probability $\frac{1}{2}$, then we would get a reduction to our $\mathsf{UR}(Q)$ problem. However, the proof crucially relies on deterministic facts, and we do not see how to modify it to give the probability $\frac{1}{2}$ to all facts. This is true for both steps. Even for the first step, the complexity of $\mathsf{UR}(Q_1)$ has been unknown so far. For the second step, it is not at all clear how to reduce from $\mathsf{UR}(Q_1)$ to $\mathsf{UR}(Q)$, even if $\mathsf{UR}(Q_1)$ is proved to be #P-hard.

In this paper, we resolve the question and prove that $\mathsf{UR}(Q)$ is #P-complete whenever Q is a non-hierarchical CQ without self-joins. Hence, we establish that the dichotomy of Theorem 2.2 also holds for uniform reliability. Our main result is:

▶ **Theorem 3.1.** *Let Q be a CQ without self-joins. If Q is hierarchical, then $\mathsf{UR}(Q)$ is solvable in polynomial time. Otherwise, $\mathsf{UR}(Q)$ is #P-complete.*

As explained earlier, the tractability of $\mathsf{UR}(Q)$ for hierarchical queries follows from Theorem 2.2. (Note that the theorem assumes the self-join freeness of the query.) Membership of $\mathsf{UR}(Q)$ in #P is straightforward. Thus, our technical contribution is the following:

▶ **Theorem 3.2.** *Let Q be a non-hierarchical CQ without self-joins. Then $\mathsf{UR}(Q)$ is #P-hard.*

A preliminary observation is that this claim follows from the hardness of a specific family of non-hierarchical self-join-free CQs. We call them the *$Q_{r,s,t}$-queries*, and they have the following form for some natural numbers $r, s, t > 0$:

$$Q_{r,s,t} \text{ :- } \mathsf{R}_1(x), \ldots, \mathsf{R}_r(x), \mathsf{S}_1(x,y), \ldots, \mathsf{S}_s(x,y), \mathsf{T}_1(y), \ldots, \mathsf{T}_t(y) \tag{2}$$

These queries always have two variables and no constants, and relations of arity 1 or 2. Note that $Q_{1,1,1}$ is the same as Q_1 (introduced as Equation (1) on page 5). We can show that, for any non-hierarchical self-join-free CQ Q, there is a reduction to the $\mathsf{UR}(Q)$ problem from the problem $\mathsf{UR}(Q_{r,s,t})$ for some $r, s, t > 0$. Formally:

▶ **Proposition 3.3.** *Let Q be a non-hierarchical CQ without self-joins. We can compute natural numbers $r, s, t > 0$ such that there is a polynomial-time Turing reduction from $\mathsf{UR}(Q_{r,s,t})$ to $\mathsf{UR}(Q)$.*

Proof sketch. We only sketch the case where Q has no constants, as the general case is similar. As Q is non-hierarchical, we can find two variables x and y such that the sets $\mathrm{atoms}(x)$ and $\mathrm{atoms}(y)$ intersect and are incomparable. We take $r := |\mathrm{atoms}(x) \setminus \mathrm{atoms}(y)|$, $s := |\mathrm{atoms}(x) \cap \mathrm{atoms}(y)|$, and $t := |\mathrm{atoms}(y) \setminus \mathrm{atoms}(x)|$. We then reduce from an instance of $\mathsf{UR}(Q_{r,s,t})$ to an instance of $\mathsf{UR}(Q)$.

To do so, given an input instance I, we rewrite the S_i-facts $\mathsf{S}_i(a,b)$ for $1 \leq i \leq s$ by facts of the i-th relation of $\mathrm{atoms}(x) \cap \mathrm{atoms}(y)$, by reusing a and b at the positions where x and y are respectively used in Q for that relation, and filling the other positions with some fixed constant c. We rewrite the R_i-facts and the T_i-facts in the same manner to facts corresponding to the relations of $\mathrm{atoms}(x) \setminus \mathrm{atoms}(y)$ and to relations of $\mathrm{atoms}(y) \setminus \mathrm{atoms}(x)$ respectively, using the same fixed constant c for positions of the new facts where neither x nor y was used in Q. Last, for every relation of Q which is not used in $\mathrm{atoms}(x) \cup \mathrm{atoms}(y)$, we create one fact in the instance where all positions are filled with the fixed constant c. This rewriting is in polynomial time.

We can then show that UR for $Q_{r,s,t}$ on I reduces to UR for Q on the rewritten instance, which establishes the result. Note that the correctness of this process relies on the self-join-freeness of Q. ◀

From Proposition 3.3 we conclude that it suffices to prove hardness for the $Q_{r,s,t}$-queries:

▶ **Theorem 3.4.** *For all $r, s, t > 0$, the problem $\mathsf{UR}(Q_{r,s,t})$ is #P-hard.*

This implies in particular that $\mathsf{UR}(Q_1)$ is #P-hard. We prove this theorem in Sections 4–6, and then investigate a generalization in Section 7.

4 Defining the Main Reduction

In this section and the two next ones, we show the #P-hardness of $\mathsf{UR}(Q_{r,s,t})$ (Theorem 3.4). Fix the arbitrary values $r, s, t > 0$ from the theorem statement. We reduce from the #P-hard problem of counting the number of independent sets of a bipartite graph. The input to this problem is a bipartite graph $G = (R \cup T, S)$ where $S \subseteq R \times T$, and the goal is to calculate

the number P of *independent-set pairs* (R', T') with $R' \subseteq R$ and $T' \subseteq T$, that is, pairs such that $R' \times T'$ is disjoint from S. This problem is the same as computing the number of falsifying assignments of a so-called *monotone partitioned 2-DNF formula*, i.e., a monotone Boolean formula in disjunctive normal form over variables from two disjoint sets $\mathcal{X}$ and $\mathcal{Y}$ where every clause is the conjunction of one variable of $\mathcal{X}$ and one variable of $\mathcal{Y}$. Counting the satisfying assignments of such formulas is #P-hard [19], so it is also #P-hard to count falsifying assignments, and thus to count independent-set pairs.

Let us fix $G = (R \cup T, S)$ as the input to the problem. Our proof consists of three parts. First, in the present section, we introduce several gadgets and use them to build the various instances of $\mathsf{UR}(Q_{r,s,t})$ to which we reduce. Second, in Section 5, we explain how to obtain a linear equation system that connects the number P of independent-set pairs of G to the results of $\mathsf{UR}(Q_{r,s,t})$ on our instances. Last, in Section 6, we argue that the matrix of this system is invertible, so we can recover P and conclude the reduction, showing Theorem 3.4.

Defining the Gadgets. We define the gadgets that we will use in the reduction as building blocks for our instances of $\mathsf{UR}(Q_{r,s,t})$. Recall that R_i, S_i, and T_i are the relations that occur in $Q_{r,s,t}$ (see Equation (2)). For all i, we collectively refer to a fact over R_i, S_i and T_i as an R_*-fact, an S_*-fact, and a T_*-fact, respectively. In our reduction, we will use multiple copies of the gadgets, instantiated with specific elements that will intuitively serve as endpoints to the gadgets. There are two types of gadgets:

- The *(a,b)-gadget* is an instance with two elements a and b (which are intuitively the endpoints), and the following facts (noting that they satisfy the query):

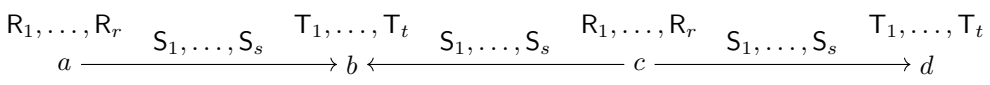
$$\mathsf{R}_1(a), \ldots, \mathsf{R}_r(a), \qquad \mathsf{S}_1(a,b), \ldots, \mathsf{S}_s(a,b), \qquad \mathsf{T}_1(b), \ldots, \mathsf{T}_t(b)$$

We will need to count the possible worlds of this gadget and of subsequent gadgets, because these quantities will be important in the reduction to understand the link between the independent-set pairs of G and the subinstances of our instances of $\mathsf{UR}(Q_{r,s,t})$ that satisfy the query. To this end, we denote by λ_{R} the number of possible worlds of the (a,b)-gadget that violate $Q_{r,s,t}$ when we fix the R_*-facts on a to be present. We easily compute: $\lambda_{\mathsf{R}} = 2^{s+t} - 1$. Similarly, we denote by λ_{T} the number of possible worlds that violate $Q_{r,s,t}$ when we fix the T_*-facts on b to be present. We have: $\lambda_{\mathsf{T}} = 2^{s+r} - 1$. Last, we denote by $\bar{\lambda}_{\mathsf{T}} = 2^{s+r}$ the number of possible worlds when we fix the T_*-facts to be absent (all these possible worlds violate $Q_{r,s,t}$), and denote by $\bar{\lambda}_{\mathsf{R}} = 2^{s+t}$ the number of possible worlds when we fix the R_*-facts to be absent (again, all violate $Q_{r,s,t}$).

- The *(a,b,c,d)-gadget* is an instance with elements a, b, c, and d, and the following facts:

$$\mathsf{R}_1(a), \ldots, \mathsf{R}_r(a), \quad \mathsf{S}_1(a,b), \ldots, \mathsf{S}_s(a,b), \quad \mathsf{T}_1(b), \ldots, \mathsf{T}_t(b), \quad \mathsf{S}_1(c,b), \ldots, \mathsf{S}_s(c,b),$$
$$\mathsf{R}_1(c), \ldots, \mathsf{R}_r(c), \quad \mathsf{S}_1(c,d), \ldots, \mathsf{S}_s(c,d), \quad \mathsf{T}_1(d), \ldots, \mathsf{T}_t(d).$$

We illustrate the gadget below, where every vertex represents a domain element, every edge represents a pair of elements occurring in a fact, and unary and binary facts are simply written as relation names, respectively above their element and above their edge:



We denote by γ its number of possible worlds that violate $Q_{r,s,t}$ where we fix the R_*-facts on a and the T_*-facts on d to be present. We denote by δ_{R} its number of possible worlds that violate $Q_{r,s,t}$ when we fix the R_*-facts on a to be present and the T_*-facts on d to be absent,

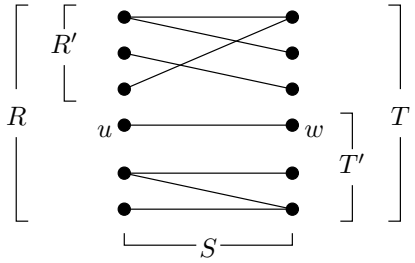


Figure 1 Example of the bipartite graph $G = (R \cup T, S)$ and an independent set (R', T').

and symmetrically denote by δ_{T} its number of possible worlds that violate $Q_{r,s,t}$ when fixing the R_*-facts on a to be absent and the T_*-facts on d to be present. Last, we denote by $\delta_\bot$ its number of possible worlds that violate $Q_{r,s,t}$ when we fix the R_*-facts on a and the T_*-facts on d to be absent. We will study the quantities γ, δ_{R}, δ_{T}, $\delta_\bot$ in two lemmas in Section 6.

Defining the Reduction. Having defined the various gadgets that we will use, let us describe the instances that we construct from our input bipartite graph $G = (R \cup T, S)$ (see Figure 1 for an illustration of the graph). The *vertices* of G are the elements of R and T, and its *edges* are the pairs in S.

Let us now write $m := |S|$, and define the following large (but polynomial) values. We will use them as parameters when building the various UR instances to which we will reduce.

$$M_1 := 4ms + 1$$
$$M_2 := M_1 + 2m(t+s)M_1 + 1$$
$$M_3 := M_1 + M_2 + |R|\,(t+s)M_2 + 1$$

Fix $M := (|R|+1) \times (|T|+1) \times (m+1)^3$, the number of instances to which we will reduce. Now, for each $0 \leq p < M$, we construct the instance D_p on the schema $Q_{r,s,t}$, featuring:

- One element u for each vertex $u \in R$ of G, with all the facts $\mathsf{R}_1(u), \ldots, \mathsf{R}_r(u)$
- One element w for each vertex $w \in T$ of G, with all the facts $\mathsf{T}_1(w), \ldots, \mathsf{T}_t(w)$
- For every edge $(u, w) \in S$ of G, create:
  - p copies of the $(u, *, *, w)$-gadget connecting u and w (using fresh elements for b and c in each copy, as denoted by the $*$'s).
  - $M_1 \times p$ copies of the $(u, *)$-gadget (using a fresh element for b in each copy).
- For each element $u \in R$, create $M_2 \times p$ copies of the $(u, *)$-gadget.
- For each element $w \in T$, create $M_3 \times p$ copies of the $(*, w)$-gadget.

It is clear that this construction is in polynomial time in the input G for each $0 \leq p < M$, because the values M_1, M_2, M_3 are polynomial, so the construction is in polynomial-time overall because M is polynomial. Observe that the construction of D_p is designed to ensure that any match of the query $Q_{r,s,t}$ on a possible world of D_p will always be contained in the facts of one of the gadgets (plus the facts on the elements u and w of the first two bullet points). This means that we can determine if the query is true in the possible worlds simply by looking separately at the facts of each gadget (and at the facts on the u and w).

Now, coming back to our reduction, for each $0 \leq p < M$ we denote by N_p the number of subinstances of D_p that violate $Q_{r,s,t}$. Each of these values can be computed in polynomial time using our oracle for $\mathsf{UR}(Q_{r,s,t})$: for $0 \leq p < M$, we build D_p, call the oracle to obtain the number $|\mathrm{Mod}(Q_{r,s,t}, D_p)|$ of subinstances that satisfy $Q_{r,s,t}$, and compute: $N_p := 2^{|D_p|} - |\mathrm{Mod}(Q_{r,s,t}, D_p)|$.

Hence, in our reduction, given the input bipartite graph G, we have constructed the instances D_p and used our oracle to compute the number N_p of subinstances of each D_p that violate $Q_{r,s,t}$, for each $0 \leq p < M$, and this process is in PTIME. In the next section, we explain how we can use a linear equation system to recover from the numbers N_p the answer to our original problem on $G = (R \cup T, S)$, i.e., the number P of independent-set pairs of G.

5 Obtaining the Equation System

To define the linear equation system, it will be helpful to introduce some parameters about subsets of vertices of the bipartite graph. For any $R' \subseteq R$ and $T' \subseteq T$, we write the following:

- $\mathrm{c}(R', T')$ to denote the number of edges of S that are *contained* in $R' \times T'$, that is, they have both endpoints in $R' \cup T'$. Formally,

$$\mathrm{c}(R', T') := |(R' \times T') \cap S|\,.$$

- $\mathrm{d}(R', T')$ to denote the number of edges of S that are *dangling from* R', that is, they have one endpoint in R' and the other in $T \setminus T'$. Formally,

$$\mathrm{d}(R', T') := |(R' \times (T \setminus T')) \cap S|\,.$$

- $\mathrm{d}'(R', T')$ to denote the number of edges of S that are *dangling from* T', that is, they have one endpoint in $R \setminus R'$ and the other in T'. Formally,

$$\mathrm{d}'(R', T') := |((R \setminus R') \times T') \cap S|\,.$$

- $\mathrm{e}(R', T')$ to denote the number of edges of S that are *excluded* from $R' \cup T'$, that is, they have no endpoint in $R' \cup T'$. Formally,

$$\mathrm{e}(R', T') := |S \setminus (R' \times T')|\,.$$

It is immediate by definition that, for any R' and T', every edge of S is either contained in $R' \times T'$, dangling from R', dangling from T', or excluded from $R' \cup T'$. Hence, we clearly have

$$\mathrm{c}(R', T') + \mathrm{d}(R', T') + \mathrm{d}'(R', T') + \mathrm{e}(R', T') = m.$$

Observe that a pair (R', T') is an independent-set pair of G iff $\mathrm{c}(R', T') = 0$. Thus, given the input G to the reduction, our goal is to compute the following quantity:

$$P = |\{(R', T') \mid R' \subseteq R,\ T' \subseteq T,\ \mathrm{c}(R', T') = 0\}| = \sum_{R' \subseteq R,\ T' \subseteq T,\ \mathrm{c}(R', T') = 0} 1 \tag{3}$$

Let us define the variables on the input graph G that we will use to express P, and that we will recover from the values N_p.

Picking Variables. Our goal is to construct a linear equation system relating the quantity that we wish to compute, namely P, and the quantities provided by our oracle, namely N_p for $0 \leq p < M$. Instead of using P directly, we will construct a system connecting N_p to quantities on G that we now define, from which we will be able to recover P. We call these quantities *variables* because they are unknown and our goal in the reduction is to compute them from the N_p to recover P.

Let us introduce, for each $0 \leq i \leq |R|$, for each $0 \leq j \leq |T|$, for each $0 \leq c, d, d' \leq m$, the variable $X_{i,j,c,d,d'}$, that stands for the number of pairs (R', T') with $|R'| = i$, with $|T'| = j$, and with c- and d- and d′-values exactly as indicated. (We do not need e as a parameter here because it is determined from c, d, d′.) Formally:

$$X_{i,j,c,d,d'} := |\{(R', T') \mid R' \subseteq R\,,\, T' \subseteq T\,,\, |R'| = i\,,\, |T'| = j,\\ \mathrm{c}(R', T') = c\,,\, \mathrm{d}(R', T') = d\,,\, \mathrm{d}'(R', T') = d'\}|$$

For technical reasons, let us define, for all i, j, c, d, d', other variables, which are the ones that we will actually use in the equation system:

$$Y_{i,j,c,d,d'} := (2^r - 1)^{|R|-i} \times (2^t - 1)^{|T|-j} \times X_{i,j,c,d,d'}$$

Getting our Answer from the Variables. Let us now explain why we can compute our desired value P (the number of independent-set pairs of G) from the variables $Y_{i,j,c,d,d'}$. Refer back to Equation (3), and let us split this sum according to the values of the parameters $i = |R'|$, $j = |T'|$, and $\mathrm{d}(R', T')$, $\mathrm{d}'(R', T')$. Using our variables $X_{i,j,c,d,d'}$, this gives:

$$P = \sum_{0 \leq i \leq |R|} \sum_{0 \leq j \leq |T|} \sum_{0 \leq d, d' \leq m} X_{i,j,0,d,d'}$$

We can insert the variables $Y_{i,j,c,d,d'}$ instead of $X_{i,j,c,d,d'}$ in the above, obtaining:

$$P = \sum_{0 \leq i \leq |R|} \sum_{0 \leq j \leq |T|} \sum_{0 \leq d, d' \leq m} \frac{Y_{i,j,0,d,d'}}{(2^r - 1)^{|R|-i} \times (2^t - 1)^{|T|-j}} \tag{4}$$

This equation justifies that, to compute the quantity P that we are interested in, it suffices to compute the value of the variables $Y_{i,j,0,d,d'}$ for all $0 \leq i \leq |R|$, $0 \leq j \leq |T|$, and $0 \leq d, d' \leq m$. If we can compute all these quantities in polynomial time, then we can use the equation above to compute P in polynomial time, completing the reduction.

Designing the Equation System. We will now design a linear equation system that connects the quantities N_p for $0 \leq p < M$ computed by our oracle to the quantities $Y_{i,j,c,d,d'}$ for all $0 \leq i \leq |R|$, $0 \leq j \leq |T|$, $0 \leq c, d, d' \leq m$ that we wish to compute. To do so, write the vector $\vec{N} = (N_0, \ldots, N_{M-1})$, and the vector $\vec{Y} = (Y_{0,0,0,0,0}, \ldots, Y_{|R|,|T|,m,m,m})$. We will describe an M-by-M matrix A so that we have the equation $\vec{N} = A\vec{Y}$. We will later justify that the matrix A is invertible, so that we can compute $\vec{Y}$ from $\vec{N}$ and conclude the proof.

To define the matrix A, let us consider arbitrary subsets $R' \subseteq R$ and $T' \subseteq T$, and an arbitrary $0 \leq p < M$, and let us denote by $\mathcal{D}_p(R', T')$ the set of subinstances of D_p where the set of vertices of R on which we have kept *all* R_*-facts is precisely R', and where the set of vertices on which we have kept *all* T_*-facts is precisely T'. In other words, an instance $I' \subseteq D_p$ is in $\mathcal{D}_p(R', T')$ if *(a)* I' contains all R_*-facts on elements of R' and all T_*-facts on elements of T', *and (b)* for each vertex in $R \setminus R'$, there is at least one R_*-fact missing from I', and for each vertex in $T \setminus T'$, there is at least one T_*-fact missing from I'. It is clear that the $\mathcal{D}_p(R', T')$ form a partition of the subinstances of D_p, so that:

$$N_p = \sum_{R' \subseteq R, T' \subseteq T} |\{I' \in \mathcal{D}_p(R', T') \mid I' \not\models Q_{r,s,t}\}| \tag{5}$$

Let us now study the number in the above sum for each R' and T', that is, the number of instances in $\mathcal{D}_p(R', T')$ that violate the query. We can show the following by performing some accounting over all gadgets in the construction.

▷ Claim 5.1. For any $0 \leq p < M$, for any choice of R' and T', writing $i := |R'|$, $j := |T'|$, $c := \mathrm{c}(R', T')$, $d := \mathrm{d}(R', T')$, $d' := \mathrm{d}'(R', T')$, $e := \mathrm{e}(R', T') = m - c - d - d'$, we have:

$$|\{I' \in \mathcal{D}_p(R', T') \mid I' \not\models Q_{r,s,t}\}| = (2^r - 1)^{|R|-i} \times (2^t - 1)^{|T|-j} \times \alpha(i, j, c, d, d')^p$$

Where $\alpha(i, j, c, d, d')$ is defined as the following quantity:

$$\gamma^c \times \delta_{\mathsf{R}}^d \times \delta_{\mathsf{T}}^{d'} \times \delta_{\perp}^e \times \lambda_{\mathsf{R}}^{M_1(c+d)+M_2 i} \times \lambda_{\mathsf{T}}^{M_3 j} \times \bar{\lambda}_{\mathsf{R}}^{M_1(d'+e)+M_2(|R|-i)} \times \bar{\lambda}_{\mathsf{T}}^{M_3(|T|-j)}.$$

Let us substitute this value in Equation (5). Note that this value only depends on the cardinalities of R' and T' and the values of $\mathrm{c}, \mathrm{d}, \mathrm{d}', \mathrm{e}$, but not on the specific choice of R' and T'. Thus, splitting the sum accordingly, we can obtain the following:

▷ Claim 5.2. For any $0 \leq p < M$, we have that:

$$N_p = \sum_{0 \leq i \leq |R|} \sum_{0 \leq j \leq |T|} \sum_{0 \leq c,d,d' \leq m} Y_{i,j,c,d,d'} \times \alpha(i, j, c, d, d')^p$$

This equation can be expressed as a matrix equation $\vec{N} = A\vec{Y}$, with A the matrix whose cells contain $\alpha(i, j, c, d, d')^p$. Note that A is indeed an M-by-M matrix, where each row corresponds to a value of p with $0 \leq p < M$, and every column corresponds to a tuple (i, j, c, d, d'), for which there are M choices by definition of M. The matrix A relates the vector $\vec{N}$ computed from our oracle calls and the variables $\vec{Y}$ that we wish to determine to solve our problem on the graph G. It only remains to show that A is an invertible matrix, so that we can compute its inverse A^{-1} in polynomial time, use it to recover $\vec{Y}$ from $\vec{N}$, and from there recover P via Equation 4, concluding the reduction. Now, A is clearly a Vandermonde matrix, so we need just argue that its coefficients $\alpha(i, j, c, d, d')$ are different. We do this in the next section.

6 Showing that the Matrix is Invertible

In this section, we conclude our proof of Theorem 3.4 by showing the following:

▷ Claim 6.1. For all $(i, j, c, d, d') \neq (i_2, j_2, c_2, d_2, d'_2)$ with $0 \leq i, i_2 \leq |R|$, $0 \leq j, j_2 \leq |T|$, $0 \leq c, c_2, d, d_2, d', d'_2 \leq m$, we have $\alpha(i, j, c, d, d') \neq \alpha(i_2, j_2, c_2, d_2, d'_2)$.

This implies that the Vandermonde matrix A is invertible, and concludes the definition of the reduction and the proof of Theorem 3.4.

A Closer Look at γ, δ_{R}, δ_{T} and $\delta_{\perp}$. To show our claim, we will need to look deeper in the definition of α, which involves γ, δ_{R}, δ_{T} and $\delta_{\perp}$. Remember that these are the number of possible worlds of the $(*, *, *, *)$-gadgets defined in Section 4. To show the invertibility of the matrix, we will first need to understand what is the exponent of the number 2 in the decomposition of these numbers as a product of primes. This is abstracted away in the following lemma, which we prove by computing explicitly the numbers of possible worlds.

▶ **Lemma 6.2.** *The number γ is odd, and we have, for some odd quantities δ'_{R}, δ'_{T}, $\delta'_{\perp}$:*

$$\delta_{\mathsf{R}} = 2^s \times \delta'_{\mathsf{R}}$$
$$\delta_{\mathsf{T}} = 2^s \times \delta'_{\mathsf{T}}$$
$$\delta_{\perp} = (2^s)^2 \times \delta'_{\perp}$$

Second, we will need the following lemma on these quantities:

▶ **Lemma 6.3.** *For all $r, s, t \geq 1$, we have: $\delta_{\mathsf{R}} \times \delta_{\mathsf{T}} \neq \gamma \times \delta_{\perp}$.*

Proof sketch. We do a case distinction on the possible worlds accounted for in $\delta_{\mathsf{R}} \times \delta_{\mathsf{T}}$ and those accounted for in $\gamma \times \delta_{\perp}$, picking an order on the nodes that simplifies the comparison between the two case distinctions. By focusing on the cases where the number of possible worlds is different, we can explicitly compute the difference between these two quantities and show that it is non-zero, specifically it is $(2^s)^3 \times (2^r - 1) \times (2^t - 1)$. (We suspect that there may be a more elegant proof avoiding the need for this case distinction.) ◀

We can now use the previous lemmas to show Claim 6.1. Fix i, j, c, d, d' and i_2, j_2, c_2, d_2, d'_2. As usual, we denote $e = m - c - d - d'$ and $e_2 = m - c_2 - d_2 - d'_2$. By contraposition, we show that if $\alpha(i, j, c, d, d') = \alpha(i_2, j_2, c_2, d_2, d'_2)$ then the parameters are equal.

Equality on i and j, and Two Equations for c, d, d'. Let us rewrite the definition of α (given in Claim 5.1), using Lemma 6.2 and substituting the definition of $\bar{\lambda}_{\mathsf{R}}$ and $\bar{\lambda}_{\mathsf{T}}$ from Section 4:

$$\begin{aligned}\alpha(i,j,c,d,d') = \gamma^c \times (2^s)^d (\delta'_{\mathsf{R}})^d \times (2^s)^{d'} (\delta'_{\mathsf{T}})^{d'} \times (2^{2s})^e {\delta'_{\perp}}^e \\ \times \lambda_{\mathsf{R}}^{M_1(c+d)+M_2 i} \times \lambda_{\mathsf{T}}^{M_3 j} \times (2^{t+s})^{M_1(d'+e)+M_2(|R|-i)} \times (2^{r+s})^{M_3(|T|-j)}\end{aligned}$$

Recall that, by Lemma 6.2, the quantities γ, δ'_{R}, δ'_{T} and $\delta'_{\perp}$ are odd, and the quantities λ_{R} and λ_{T} as defined in Section 4 are odd. We get a similar equation for $\alpha(i_2, j_2, c_2, d_2, d'_2)$. From the integer equality $\alpha(i, j, c, d, d') = \alpha(i_2, j_2, c_2, d_2, d'_2)$, the coefficients of two in the prime number decompositions of these numbers must also be equal. This yields:

$$\begin{aligned}&s(d\ + d'\ + 2e\ ) + (t+s) \times (M_1(d'\ + e\ )) + (t+s) \times M_2(|R| - i\ ) + (r+s) \times M_3(|T| - j\ ) \\ &= s(d_2 + d'_2 + 2e_2) + (t+s) \times (M_1(d'_2 + e_2)) + (t+s) \times M_2(|R| - i_2) + (r+s) \times M_3(|T| - j_2)\end{aligned}$$

We now use the fact that, as $d, d', e \leq m$, we have $s(d + d' + 2e) \leq 4ms$. By definition of M_1, we have $s(d + d' + 2e) < M_1$. We also have $d' + e \leq 2m$, so that (using the previous inequality) we have $s(d + d' + 2e) + (t + s) \times (M_1(d' + e)) < M_2$ by definition of M_2. Last, we have $|R| - i \leq |R|$, so that (using the two previous inequalities) we have $s(d + d' + 2e) + (t + s) \times (M_1(d' + e)) + (t + s) \times M_2 \times (|R| - i) < M_3$ by the definition of M_3. Similar inequalities hold for the right-hand-side of the above equation. Thus, we can reason about the quotient of the equation by M_3, about the quotient by M_2 of its remainder modulo M_3, about the quotient by M_1 of the remainder modulo M_2 of the remainder modulo M_3, and about the remainder modulo M_1 of the remainder modulo M_2 of the remainder modulo M_3. This gives us four equations (where we also simplify by the constant factors s, $t + s$, $r + s$):

$$\begin{aligned}d + d' + 2e &= d_2 + d'_2 + 2e_2 \\ d' + e &= d'_2 + e_2 \\ |R| - i &= |R| - i_2 \\ |T| - j &= |T| - j_2\end{aligned}$$

The last two equations imply that $i = i_2$ and $j = j_2$, so we have shown that two quantities are equal, out of the five that define α. The two first equations imply $d' + e = d'_2 + e_2$ (second

equation), and $d + e = d_2 + e_2$ (subtracting the second equation from the first equation). Rewriting $e = m - c - d - d'$, rewriting e_2 likewise, and simplifying, we get:

$$c + d = c_2 + d_2$$
$$c + d' = c_2 + d'_2$$

These two equations do not suffice to justify that $(c, d, d') = (c_2, d_2, d'_2)$, so more reasoning is needed to get one additional equation and argue that these quantities must be equal.

Getting the Last Equation. Let us write the equality $\alpha(i, j, c, d, d') = \alpha(i_2, j_2, c_2, d_2, d'_2)$ and simplify all the (non-zero) values now known to be equal thanks to $i = i_2$, $j = j_2$:

$$\gamma^c \cdot (\delta_{\mathsf{R}})^d \cdot (\delta_{\mathsf{T}})^{d'} \cdot (\delta_\perp)^e \cdot \lambda_{\mathsf{R}}^{M_1(c+d)} \cdot \bar{\lambda}_{\mathsf{R}}^{M_1(d'+e)} = \gamma^{c_2} \cdot (\delta_{\mathsf{R}})^{d_2} \cdot (\delta_{\mathsf{T}})^{d'_2} \cdot (\delta_\perp)^{e_2} \cdot \lambda_{\mathsf{R}}^{M_1(c_2+d_2)} \cdot \bar{\lambda}_{\mathsf{R}}^{M_1(d'_2+e_2)}$$

The previously shown equations also imply that the λ_{R} and $\bar{\lambda}_{\mathsf{R}}$ factors simplify, so we get:

$$\gamma^{c-c_2} \times (\delta_{\mathsf{R}})^{d-d_2} \times (\delta_{\mathsf{T}})^{d'-d'_2} \times (\delta_\perp)^{e-e_2} = 1$$

The equation $c + d = c_2 + d_2$ (shown above) implies that $c - c_2 = d_2 - d$, and subtracting that equation from $c + d' = c_2 + d'_2$ (shown above) gives $d' - d = d'_2 - d_2$, so that $d' - d'_2 = d - d_2$. As we know $d + e = d_2 + e_2$ (above), we have $e - e_2 = d_2 - d$. So using $c - c_2 = d_2 - d$, $d' - d'_2 = d - d_2$, and $e - e_2 = d_2 - d$, we have:

$$(\gamma \times \delta_\perp)^{d_2-d} = (\delta_{\mathsf{R}} \times \delta_{\mathsf{T}})^{d_2-d}$$

Now, by Lemma 6.3, we have $\gamma \times \delta_\perp \neq \delta_{\mathsf{R}} \times \delta_{\mathsf{T}}$. Thus, this equation implies that $d = d_2$. In combination with the equations that we showed above, this completely specifies the system: from $c + d = c_2 + d_2$ we get $c = c_2$, and from $c + d' = c_2 + d'_2$ we get $d' = d'_2$. Thus, we have $(c, d, d', i, j) = (c_2, d_2, d'_2, i_2, j_2)$. This establishes Claim 6.1 and shows that all coefficients $\alpha(c, d, d', i, j)$ of the Vandermonde matrix A are different, so it is invertible. This concludes the proof of Theorem 3.4, and hence of our main result (Theorems 3.1 and 3.2).

7 Extending to Uniform Probabilities

Having proved our main result (Theorem 3.1), we now turn to a natural variant of the probabilistic query evaluation problem: what if, instead of imposing that all probabilities are $\frac{1}{2}$ (which amounts to uniform reliability), we impose that all tuples of the same relation have the same probability? Let us formally define this variant:

▶ **Definition 7.1.** *Let Q be a CQ without self-joins, and let φ be a function mapping each relation symbol ρ of Q to a rational number $0 < \varphi(\rho) \leq 1$. The problem $\mathsf{PQE}_\varphi(Q)$ is the problem $\mathsf{PQE}(Q)$ on input instances I over Q whose probability function $\pi : I \to [0, 1]$ is defined by φ, i.e., for every fact $f \in I$, we have $\pi(f) = \varphi(\rho)$ where ρ is the relation symbol used in f.*

In this section, we will focus on this problem for the hard query $Q_1 : \mathsf{R}(x), \mathsf{S}(x, y), \mathsf{T}(y)$ from Equation 1, and write the problem directly as $\mathsf{PQE}_{r,s,t}(Q_1)$ where $0 < r, s, t \leq 1$ are the respective images of R, S and T under φ. In this language, the uniform reliability problem for Q_1 is (up to renormalization) the problem $\mathsf{PQE}_{\frac{1}{2},\frac{1}{2},\frac{1}{2}}(Q_1)$, which we have shown to be #P-hard in Theorem 3.1 because Q_1 is non-hierarchical. The usual #P-hardness proof for

$\mathsf{PQE}(Q_1)$ [6], where we reduce from the problem of counting the satisfying assignment of a monotone partitioned DNF formula [19], is actually a hardness proof for $\mathsf{PQE}_{\frac{1}{2},1,\frac{1}{2}}(Q_1)$. The natural question is whether hardness can be shown for other values of r, s, and t.

Let us first observe that we can use our main hardness result on uniform reliability (Theorem 3.2) to show hardness in a specific case via an easy reduction. While we do not hope that the technique can generalize, it still classifies some of the cases:

▶ **Corollary 7.2.** $\mathsf{PQE}_{2^{-r},2^{-s},2^{-t}}(Q_1)$ *is #P-hard for all natural numbers* $r, s, t > 0$.

Proof. Consider the query $Q_{i,j,k}$, following Equation 2. We have shown in Theorem 3.4 that uniform reliability for $Q_{i,j,k}$ is #P-hard. We reduce this problem to the PQE variant that we consider. To do this, consider an input instance I to $\mathsf{UR}(Q_{i,j,k})$. We call an element a of I *useless for* R_* if some R_*-fact does not hold on a, we define a being *useless for* T_* analogously, and we call a pair (a, b) of I *useless for* S_* if some S_*-fact does not hold about the pair. We call an R_*-fact *useless* if it holds about an element that is useless for R_*, and extend this definition to T_*-facts and to S_*-facts (for element pairs). It is clear that no match of $Q_{i,j,k}$ on I can involve a useless fact. Indeed, when a match of the query involves a fact f of the form $\mathsf{R}_*(a)$, then the match witnesses that all other R_*-facts hold on a, so that a is not useless for R_*, and f is not useless. The same reasoning applies to S_*-facts and T_*-facts.

Hence, let us compute in linear time the subinstance I' of I where we only keep the facts that are not useless: this is doable in polynomial time. Now, any subset of I is defined by picking a subset J' of I' and a subset J'' of $I \setminus I'$. Further, as $I \setminus I'$ only consists of useless facts, it is clear that if some subset J' does not satisfy $Q_{i,j,k}$ then $J := J' \cup (I \setminus I')$ still does not, because the facts added in $I \setminus I'$ cannot be part of a query match in I, hence in J. All this reasoning shows that the answer to $\mathsf{UR}(Q_{i,j,k})$ on I is the answer to the same problem on I', multiplied by the number of possible choices for J'', that is, $2^{|I \setminus I'|}$. Hence, to show hardness, it suffices to reduce the uniform reliability problem on I' to our PQE problem.

Now, we can rewrite I' in linear time to I'' by replacing the set of R_*-facts on every element a where they exist by a single R-fact, and doing the same for T_*-facts and for S_*-facts (on element pairs). As we have removed useless facts, all facts of I' are thus taken into account in the rewriting. Now, define the probability assignment π on I'' by mapping the R, S, and T-facts to 2^{-i}, 2^{-j}, and 2^{-k} respectively. This means that (I'', π) is an instance to the $\mathsf{PQE}_{2^{-i},2^{-j},2^{-k}}(Q_1)$ problem that we are reducing to. Now, there is a clear correspondence from the subsets of I' to the possible worlds of I'', which is defined by rewriting to the signature R, S, T as we did in the reduction; and the number of preimages of each possible world of I'' is exactly equal to its probability according to π, up to renormalization by a constant factor of $2^{-|I'|}$. Thus, the answer to $\mathsf{UR}(Q_{i,j,k})$ on I' is exactly the answer to $\mathsf{PQE}_{2^{-i},2^{-j},2^{-k}}(Q_1)$ on I'' up to renormalization. This concludes the proof. ◀

Note that this does not classify the complexity of $\mathsf{PQE}_{r,s,t}(Q_1)$ for arbitrary values of r, s, t. Conversely, $\mathsf{PQE}_{r,s,1}(Q_1)$ and $\mathsf{PQE}_{1,s,t}(Q_1)$ are clearly solvable in polynomial time:

▶ **Proposition 7.3.** $\mathsf{PQE}_{r,s,1}(Q_1)$ *and* $\mathsf{PQE}_{1,s,t}(Q_1)$ *are in PTIME for all* $0 < r, s, t \leq 1$.

Proof sketch. When the R-facts have probability 1, we can rewrite Q_1 to $\mathsf{S}(x, y), \mathsf{T}(y)$ which is hierarchical, hence safe (Theorem 2.2). The other case is symmetric. ◀

We conjecture that these are the only tractable cases. Specifically, we conjecture the following generalization of the #P-hardness of $\mathsf{UR}(Q_1)$:

▶ **Conjecture 7.4.** $\mathsf{PQE}_{r,s,t}(Q_1)$ *is #P-hard for all* $0 < r, s, t \leq 1$ *with* $r < 1$ *and* $t < 1$.

We leave this for future work, but can prove Conjecture 7.4 in the case of $s = 1$. Formally:

▶ **Theorem 7.5.** $\mathsf{PQE}_{r,1,t}(Q_1)$ *is #P-hard for every* $0 < r < 1$ *and* $0 < t < 1$.

Proof sketch. Like in the previous section, we reduce from the problem of counting the independent sets of a bipartite graph. We do a simpler coding using simpler gadgets, and we show like before that the number of independent sets (parameterized by their number of vertices to the left and right) can be connected to the answer to the uniform reliability problem on a family of instances with different gadget instantiations. To argue that the matrix of the equation system is invertible, we notice that it is the Kronecker product of two invertible Vandermonde matrices. ◀

8 Conclusion

While query evaluation over TIDs has been studied for over a decade, the basic case of a uniform distribution, namely uniform reliability, had been left open. We have settled this open question for the class of CQs without self-joins, and shown a dichotomy on computational complexity of counting satisfying database subsets: this task is tractable for hierarchical queries, and #P-hard otherwise. We have also embarked on the investigation of the more general variant of CQ evaluation over TIDs with uniform probabilities. We have shown tractability for some combinations of probabilities by a straightforward reduction from uniform reliability, and shown hardness for others using different proof techniques.

In future work, we plan to investigate whether and how the proof of our main result can be simplified, with the goal of showing hardness for all combinations of uniform probabilities not known to be tractable, at least for the query Q_1 (Conjecture 7.4). An interesting special case is when these probabilities are $1/2$ and 1, that is, some relations are deterministic while the others define a uniform distribution over all their subsets; note that hardness in this setting is a stronger statement than hardness in the model of Dalvi and Suciu [7], as in this model the tuples of the probabilistic relations can have arbitrary probabilities.

Another question is whether our results could extend to more general query classes. A natural question would be to study the question for CQs with self-joins. More generally, we could study the case of UCQs with self-joins, and try to match the known dichotomy for non-uniform probabilities [7]. Following our work, considerable progress in this direction has been done recently by Kenig and Suciu [14], which addresses the case of PQE with probabilities of $1/2$ and 1, and leaves open the case of uniform reliability for arbitrary UCQs.

References

1 Antoine Amarilli and Benny Kimelfeld. Uniform reliability of self-join-free conjunctive queries. In *ICDT*, 2021.

2 Paul Beame, Guy Van den Broeck, Eric Gribkoff, and Dan Suciu. Symmetric weighted first-order model counting. In *PODS*, pages 313–328. ACM, 2015.

3 Christoph Berkholz, Jens Keppeler, and Nicole Schweikardt. Answering conjunctive queries under updates. In *PODS*, pages 303–318. ACM, 2017.

4 Andrei A. Bulatov. The complexity of the counting constraint satisfaction problem. *J. ACM*, 60(5):34:1–34:41, 2013.

5 Supratik Chakraborty, Dror Fried, Kuldeep S Meel, and Moshe Y Vardi. From weighted to unweighted model counting. In *IJCAI*, 2015.

6 Nilesh Dalvi and Dan Suciu. Efficient query evaluation on probabilistic databases. *VLDB Journal*, 16(4):523–544, 2007.

7 Nilesh Dalvi and Dan Suciu. The dichotomy of probabilistic inference for unions of conjunctive queries. *J. ACM*, 59(6), 2012.

8 Nilesh N. Dalvi, Christopher Ré, and Dan Suciu. Probabilistic databases: Diamonds in the dirt. *Commun. ACM*, 52(7):86–94, 2009.

9 Pradeep Dubey and Lloyd S. Shapley. Mathematical properties of the Banzhaf power index. *Mathematics of Operations Research*, 4(2):99–131, 1979.

10 Erich Grädel, Yuri Gurevich, and Colin Hirsch. The complexity of query reliability. In *PODS*, pages 227–234. ACM Press, 1998.

11 Eric Gribkoff and Dan Suciu. SlimShot: In-database probabilistic inference for knowledge bases. *PVLDB*, 9(7):552–563, 2016.

12 B. Grofman and H. Scarrow. *Iannucci and Its Aftermath: The Application of the Banzhaf Index to Weighted Voting in the State of New York*, pages 168–183. Physica-Verlag HD, Heidelberg, 1979. doi:10.1007/978-3-662-41501-6_10.

13 Abhay Kumar Jha and Dan Suciu. Probabilistic databases with MarkoViews. *PVLDB*, 5(11):1160–1171, 2012.

14 Batya Kenig and Dan Suciu. A dichotomy for the generalized model counting problem for unions of conjunctive queries. *CoRR*, abs/2008.00896, 2020.

15 Ester Livshits, Leopoldo E. Bertossi, Benny Kimelfeld, and Moshe Sebag. The Shapley value of tuples in query answering. In *ICDT*, volume 155 of *LIPIcs*, pages 20:1–20:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

16 Dany Maslowski and Jef Wijsen. A dichotomy in the complexity of counting database repairs. *J. Comput. Syst. Sci.*, 79(6):958–983, 2013.

17 Dany Maslowski and Jef Wijsen. Counting database repairs that satisfy conjunctive queries with self-joins. In *ICDT*, pages 155–164. OpenProceedings.org, 2014.

18 Dan Olteanu and Jiewen Huang. Using OBDDs for efficient query evaluation on probabilistic databases. In *SUM*, volume 5291 of *Lecture Notes in Computer Science*, pages 326–340. Springer, 2008.

19 J. Scott Provan and Michael O. Ball. The complexity of counting cuts and of computing the probability that a graph is connected. *SIAM Journal on Computing*, 12(4), 1983.

20 Alvin E Roth. *The Shapley value: Essays in honor of Lloyd S. Shapley*. Cambridge University Press, 1988.

21 Babak Salimi, Leopoldo E. Bertossi, Dan Suciu, and Guy Van den Broeck. Quantifying causal effects on query answering in databases. In *TAPP*, 2016.

22 L.S. Shapley. Stochastic games. *Proceedings of the National Academy of Sciences of the United States of America*, 39:1095–1100, 1953.

23 Dan Suciu, Dan Olteanu, Christopher Ré, and Christoph Koch. *Probabilistic Databases*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2011.