The Complexity of Query Reliability

Erich Grädel
University of Technology Aachen
graedel@informatik.rwth-aachen.de

Yuri Gurevich
Universtity of Michigan
gurevich@umich.edu

Colin Hirsch
University of Technology Aachen
hirsch@informatik.rwth-aachen.de

Abstract

The reliability of database queries on databases with uncertain information is studied, on the basis of a probabilistic model for unreliable databases.

While it was already known that the reliability of quantifier-free queries is computable in polynomial time, we show here that already for conjunctive queries, the reliability may become highly intractable. We exhibit a conjunctive query whose reliability problem is complete for $\mathrm{FP}^{\#\mathrm{P}}$. We further show, that $\mathrm{FP}^{\#\mathrm{P}}$ is the typical complexity level for the reliability problems of a very large class of queries, including all second-order queries.

We study approximation algorithms and prove that the reliabilities of all polynomial-time evaluable queries can be efficiently approximated by randomized algorithms.

Finally we discuss the extension of our approach to the more general *metafinite database model* where finite relational structures are endowed with functions into an infinite interpreted domain; in addition queries may use aggregate functions like in SQL. Our result that reliability problems of first-order queries have complexity $\mathrm{FP}^{\#\mathrm{P}}$ also holds on this extended model.

1 Introduction

In this paper, we investigate the reliability of queries on databases with uncertain information.

We consider a probabilistic model: An unreliable database is given by a pair $(\mathfrak{A}, \mu)$ where $\mathfrak{A}$ is a finite database (i.e. a finite relational structure) and μ is a function that assigns to every atomic statement $R\bar{a}$ about $\mathfrak{A}$ an error probability $\mu(R\bar{a})$. The value $\mu(R\bar{a})$ is to be interpreted as the probability that the truth value of $R\bar{a}$ on the *observed* database $\mathfrak{A}$ differs from the truth value of $R\bar{a}$ on the *actual* database. Thus, an unreliable database $(\mathfrak{A}, \mu)$ gives rise to a probability space of databases $\mathfrak{B}$ of the same format (that is having the same vocabulary and the same universe) as $\mathfrak{A}$, with a probability distribution ν that assigns to $\mathfrak{B}$ the probability that the actual database is $\mathfrak{B}$. As we will see below, the probabilities $\nu(\mathfrak{B})$ can be easily calculated, given $(\mathfrak{A}, \mu)$ and $\mathfrak{B}$. A very similar model has been considered by de Rougemont [9].

Suppose that a user is given an unreliable database $\mathfrak{D} = (\mathfrak{A}, \mu)$ and wants to evaluate a query $\psi(\bar{x})$. She can evaluate ψ on the observed database $\mathfrak{A}$ and gets $\psi^{\mathfrak{A}} := \{\bar{a} : \mathfrak{A} \models \psi(\bar{a})\}$, but of course, what she really wants is $\psi^{\mathfrak{B}}$, the set of tuples satisfying ψ on the actual database $\mathfrak{B}$. Even if the error probabilities of the atomic statements on $\mathfrak{A}$ are small, it is not clear how reliable the answer $\psi^{\mathfrak{A}}$ will be. In any case, the user would be interested to know how closely $\psi^{\mathfrak{A}}$ approximates the result of ψ on the actual database. Since the actual database is available only as a random variable, the relevant notions measuring the reliability are of a statistical nature.

The *expected error* $H_\psi(\mathfrak{D})$ of ψ on an unreliable database $\mathfrak{D}$ is the expected Hamming distance between the query evaluated on the observed database and the query evaluated on the actual database, that is, the number of tuples where $\psi^{\mathfrak{A}}$ differs from $\psi^{\mathfrak{B}}$. The *reliability* or *fault tolerance* of a k-ary ψ on $\mathfrak{D}$ is

$$R_\psi(\mathfrak{D}) := 1 - \frac{H_\psi(\mathfrak{D})}{n^k}$$

where n is the cardinality of the database. More details are given in Sect. 2.

We study here the complexity of computing the reliability of queries in various query languages. We formulate this more precisely: Fix a query language L; the reliability problem for an L-query $\psi(\bar{x})$ is this: Given an unreliable database $\mathfrak{D}$, compute the reliability $R_\psi(\mathfrak{D})$.

It would be desirable to have efficient algorithms for calculating query reliabilities. One could then combine (efficient) query evaluation with efficient algorithms calculating reliabilities. As answer to a query, the user then not only gets the evaluation on the observed database, but also some information on how good this evaluation can be expected to be.

Note that we measure the complexity of the reliability problem of a query ψ in terms of the size of (an appropriate encoding of) the unreliable database. In database terminology, we study the data complexity (rather than the expression complexity or combined complexity) of query reliability. This makes sense since the queries are usually given by small expressions, whereas the size of the databases may be huge.

De Rougemont [9] proved that the reliability is computable in polynomial time for all *quantifier-free* queries. He further claimed that the same holds for all first-order queries. However, this is unlikely to be true, since we will prove below that there exist first-order queries, in fact even *conjunctive queries*, whose reliability problem is complete for $\mathrm{FP}^{\#\mathrm{P}}$, the class of functions computable in polynomial-time with access to an oracle computing a #P-function. Since it is generally conjectured that #P is a much larger class than P (it comprises the entire polynomial-time hierarchy), we cannot hope for efficient algorithms for the reliability of arbitrary conjunctive queries, let alone first-order queries.

We will then show that $\mathrm{FP}^{\#\mathrm{P}}$ is the typical complexity level for the reliability of database queries. Indeed, for all second-order queries (i.e. for all queries in the polynomial-time hierarchy), the reliability problem is in the class $\mathrm{FP}^{\#\mathrm{P}}$. This strengthens another claim of de Rougemont.

Given that the reliability is easy only for quantifier-free queries, and hard to compute already for conjunctive queries, it is desirable to have efficient *approximation algorithms.* We will prove that there do exist *fully polynomial-time randomized approximation schemes* (FPTRAS) for the probabilities of existential sentences on unreliable databases. This is a rather strong notion of approximability which is often used in the context of counting and probability problems. It means that the result can be efficiently approximated with any desired degree of certainty to any desired degree of accuracy. (A precise definition is given in Sect. 5.) The existence of FPTRAS for the *probabilities* of existential sentences implies a slightly weaker notion of approximability for the *reliability* of any existential or universal query. Our version of approximability gives a bound on the absolute error, rather than the relative error as in an FPTRAS. We then discuss whether this weaker notion (which is sufficient for most practical purposes) can be strengthened to an FPTRAS for reliability. Some observations on the complexity of *absolute reliability* imply that this is unlikely. Not even the reliability problems of all existential queries admit an FPTRAS, unless NP collapses to BPP. We then proceed to show that our weaker version of approximability can be extended to include the reliability of all polynomial-time evaluable queries.

In the last section, we briefly discuss the extension of our approach to the more general *metafinite database model* where finite relational structures are endowed with functions into an infinite interpreted domain; in addition queries may use aggregate functions like in SQL. Our result that reliability problems typically have complexity $\mathrm{FP}^{\#\mathrm{P}}$ also holds on this extended model.

Related Work Considerable research has been conducted on other aspects of probabilistic databases. Zimányi [13] gives an in-depth study of queries and query evaluation methods on probabilistic relational databases, that are given by probabilistic first-order theories. A relational algebra query applied to such a database yields a probabilistic relation, where each tuple is assigned a probability. A sound and complete method for evaluating queries is developed. Notably query decomposition requires an extension to how intermediate results are stored in order to correctly calculate the result probabilities. Lakshmanan and Subrahmanian present another version of probabilistic databases and can even document a successfull implementation [6]. A large number of results specialize on probabilistic deductive databases (see [5, 10] for details and further references).

2 Unreliable databases

Definition 2.1. An *unreliable database* is a pair $\mathfrak{D} = (\mathfrak{A}, \mu)$ where $\mathfrak{D}$ is a finite relational structure and μ a probability function on the set of atomic statements $R\bar{a}$ about $\mathfrak{A}$. The value $\mu(R\bar{a})$ is the probability of the event that the truth value of $R\bar{a}$ on $\mathfrak{A}$ is wrong (i.e., that either $R\bar{a}$ holds in $\mathfrak{A}$ but not in the actual database, or that $R\bar{a}$ holds in the actual database, but not in $\mathfrak{A}$). We call $\mathfrak{A}$ the *observed database.*

For every sentence φ, let $\mathrm{Wrong}(\varphi)$ be the event that the truth-value of φ in the observed database $\mathfrak{A}$ differs from the truth-value of φ in the actual database. Thus $\mu(R\bar{a})$ is the probability of the event $\mathrm{Wrong}(R\bar{a})$. It is supposed that the events $\mathrm{Wrong}(R\bar{a})$ are independent.

An unreliable database $\mathfrak{D} = (\mathfrak{A}, \mu)$ induces a probability space $\Omega(\mathfrak{D})$ of databases $\mathfrak{B}$ that have the same format as $\mathfrak{A}$ (i.e. the the same universe and the same vocabulary), with a probability distribution ν such that $\nu(\mathfrak{B})$ is the probablity that the actual database is $\mathfrak{B}$. We are going to define the space $\Omega(\mathfrak{D})$ precisely and compute the probabilities $\nu(\mathfrak{B})$.

For every atomic statement φ, let

$$\nu(\varphi) = \begin{cases} 1 - \mu(\varphi) & \text{if } \mathfrak{A} \models \varphi \\ \mu(\varphi) & \text{if } \mathfrak{A} \models \neg\varphi. \end{cases}$$

$\nu(\varphi)$ is the probability that φ holds in the actual database. We extend ν to literals (atomic statements and negated atomic statements) in the obvious way, by putting $\nu(\neg\varphi) := 1 - \nu(\varphi)$. Let $\mathrm{Lit}(\mathfrak{B})$ be the set of literals that are true in $\mathfrak{B}$. Then

$$\nu(\mathfrak{B}) = \prod_{\varphi \in \mathrm{Lit}(\mathfrak{B})} \nu(\varphi).$$

Note, that given $(\mathfrak{A}, \mu)$ and a database $\mathfrak{B}$ of the same format as $\mathfrak{A}$, the probability $\nu(\mathfrak{B})$ can be easily computed.

Remark. There is another way of defining an unreliable database. Instead of presenting $\mathfrak{A}$ and μ, just give, for every atomic statement $R(\bar{a})$ the probability $\nu(R\bar{a})$ that $R\bar{a}$ holds in the actual database. This directly leads to the probability distribution over the possible databases and does not start out from a particular observed database. For theoretical considerations it does not really matter which approach is chosen. However, we believe that the definition used here represents the situation in practice more closely.

Query reliability. For every relational query $\psi(\bar{x})$ of arity $k \geq 0$, and every database $\mathfrak{A}$, let $\psi^{\mathfrak{A}} = \{\bar{a} \in A^k : \mathfrak{A} \models \psi(\bar{a})\}$. The *Hamming distance* between $\psi^{\mathfrak{A}}$ and $\psi^{\mathfrak{B}}$ is the cardinality of the symmetric difference $\psi^{\mathfrak{A}} \vartriangle \psi^{\mathfrak{B}}$, i.e. the number of tuples contained in $\psi^{\mathfrak{A}}$ but not in $\psi^{\mathfrak{B}}$ or vice versa.

Definition 2.2. Fix an unreliable database $\mathfrak{D} = (\mathfrak{A}, \mu)$ and a k-ary query ψ. The *expected error* $H_\psi(\mathfrak{D})$ of ψ on $\mathfrak{D}$ is $E(|\psi^{\mathfrak{A}} \vartriangle \psi^{\mathfrak{B}}|)$, the expected Hamming distance between $\psi^{\mathfrak{A}}$ and $\psi^{\mathfrak{B}}$ for randomly selected $\mathfrak{B} \in \Omega(\mathfrak{D})$. The *reliability* or *fault-tolerance* of ψ on $\mathfrak{D}$ is $R_\psi(\mathfrak{D}) := 1 - [H_\psi(\mathfrak{D})/n^k]$.

Note that in the important case where $k = 0$, we have that $0 \leq H_\psi \leq 1$ and $R_\psi = 1 - H_\psi$.

For any fixed query ψ, the expected error H_ψ and the reliability R_ψ are numerical invariants, i.e functions assigning to any unreliable database $\mathfrak{D}$ the numbers $H_\psi(\mathfrak{D})$ and $R_\psi(\mathfrak{D})$ respectively. We are interested in the complexity of these invariants for queries in various query languages.

Complexity. For all complexity considerations, we assume that the error probabilities in the given unreliable databases are rational numbers, encoded in some standard way. FP denotes the class of functions that are computable in polynomial-time. The class #P consists of all functions f from strings over a finite alphabet into $\mathbb{N}$ for which there exists a nondeterministic polynomial-time Turing machine M such that the number of accepting computations on any input for M coincides with the value of f on that input. A function is #P-hard, if there are polynomial-time Turing reductions to it from all problems in #P. If, in addition, the function is in #P, it is #P-complete. For many NP-complete decision problems and also for some problems in P, the related problem of counting the number of witnesses (rather than determining whether there exists at least one) is #P-complete. For background on #P we refer to [3, 7, 11]. Many query reliability problems are closely related to the class #P, although they technically do not belong to this class, since they are not taking values in $\mathbb{N}$. They belong to the class $\mathrm{FP}^{\#\mathrm{P}}$, i.e. they are computable by a polynomial-time algorithm with access to an oracle for a #P-function. Obviously, a function is $\mathrm{FP}^{\#\mathrm{P}}$-hard if and only if it is #P-hard. To prove that a reliability problem in $\mathrm{FP}^{\#\mathrm{P}}$ is in fact $\mathrm{FP}^{\#\mathrm{P}}$-complete, it therefore suffices to reduce a #P-hard function to it.

3 Quantifier-free and conjunctive queries

We first consider queries from very basic languages, with extremely restricted expressive power, so that there is hope to get efficient algorithms for calculating the reliability. Indeed, the quantifier-free queries do admit such algorithms.

Proposition 3.1 (de Rougemont).
Let $\psi(\bar{x})$ be any quantifier-free query. Then R_ψ, the reliability problem for ψ, is computable in polynomial time.

Proof. Let ψ be a k-ary quantifier-free query and $\mathfrak{D} = (\mathfrak{A}, \mu)$ a probabilistic database. Due to the linearity of expectation we can swap expectation and summation. Hence it suffices to show that $H_{\psi(\bar{a})}$ can be calculated in polynomial-time for any tuple $\bar{a}$. The claim then follows due to

$$R_\psi = 1 - \frac{1}{|A|^k} \sum H_{\psi(\bar{a})}.$$

For each tuple $\bar{a}$ we interpret $\psi(\bar{a})$ as a propositional formula by taking the atomic statements as propositional variables $\bar{v} = v_1 \dots v_{n(\psi)}$. Since the number of atoms in ψ is independent of the database we only have a fixed number $n(\psi)$ of propositional variables. Using the error function μ we can assign a probability to each truth assignment of $\bar{v}$. Hence by evaluating ψ once for each such truth assignment and comparing the result against $\psi^{\mathfrak{A}}$ we can calculate the expected error. □

However, as we prove next, it is unlikely that Proposition 3.1 can be generalized to all first-order queries. In fact, not even the expected errors of all *conjunctive queries* can be computed in polynomial time, unless all #P-functions are polynomial-time computable.

Recall that conjunctive queries are queries of the form

$$\exists x_1 \cdots \exists x_k (\varphi_1 \wedge \cdots \wedge \varphi_\ell)$$

where each φ_i is atomic.

Proposition 3.2. *There exist conjunctive queries ψ such that the problem of calculating the expected error H_ψ (and hence the reliability R_ψ) is #P-hard.*

Proof. We will reduce the problem #MONOTONE 2-SAT to the problem of computing H_ψ for a particular conjunctive Boolean query ψ. The problem #MONOTONE 2-SAT was proved to be #P-complete by Valiant [11]. Its input instances are propositional formulae in 2-CNF without negations, i.e. formulae of the form $\bigwedge_{i=1}^n Y_i \vee Z_i$ where Y_i and Z_i are propositional variables. The desired answer is the number of satisfying assignments.

A propositional formula of this form can be modeled by a structure (A, L, R) where the universe A is the (disjoint) union of the set of clauses and the set of propositional variables of the formula, and the atomic statements Luv (resp. Ruv) express that the left (resp. right) variable in clause u is v. Further, we model an assignment of truth values to the propositional variables by the set S of variables that are set to *false* under this assignment.

Given a positive 2-CNF formula $\bigwedge_{i=1}^n Y_i \vee Z_i$ one can construct in polynomial time the unreliable database $(\mathfrak{A}, \mu)$ where $\mathfrak{A} = (A, L, R, S)$ models the given formula together with the assignment that sets all variables to *false* (thus S is the set of all variables in the formula). The error probabilities are defined as follows: All atomic statements Luv, Ruv have error probability 0, and

$$\mu(Sv) = \begin{cases} 1/2 & \text{if } v \text{ is a variable} \\ 0 & \text{otherwise.} \end{cases}$$

Thus the probability space associated with $(\mathfrak{A}, \mu)$ is essentially the uniform distribution over all assignments of truth values to the variables in the given 2-CNF formula.

Now, consider the conjunctive query

$$\psi := \exists x \exists y \exists z (Lxy \wedge Rxz \wedge Sy \wedge Sz)$$

which expresses, on $\mathfrak{A} = (A, L, R, S)$, that the assignment defined by S does *not* satisfy the formula modeled by (A, L, R). Clearly $\mathfrak{A} \models \psi$ and the expected error $H_\psi(\mathfrak{A}, \mu)$ is just the number of assignments that satisfy the given formula, divided by the total number of assignments. Thus, if we could calculate the expected error of ψ in polynomial time, we could solve #MONOTONE 2-SAT (and thus any problem in #P) in polynomial-time. □

It is easy to see that computing H_ψ and R_ψ is in $\mathrm{FP}^{\#\mathrm{P}}$ for all conjunctive ψ. In fact, we will prove a much more general result below.

Remark. The model for unreliable databases studied by de Rougemont [9] is slightly different from ours. In his model, only positive data are unreliable. To put it differently, he only considers unreliable databases $(\mathfrak{A}, \mu)$ where $\mathfrak{A} \models \neg R\bar{a}$ implies that $\mu(R\bar{a}) = 0$. For complexity considerations this gives no essential difference. In particular, Proposition 3.2 holds also for the restricted model. In fact, our proof carries over directly, since the reduction from #MONOTONE 2-SAT assigns positive error probabilites to positive atomic facts only. For further details, see [2].

4 Second-order queries

In this section we prove that the reliability problem is in the class $\mathrm{FP}^{\#\mathrm{P}}$ for all second-order queries. In particular, this includes all Datalog queries (for which the result has already been proved by de Rougemont) and all fixed point queries, but goes far beyond these languages.

The proof is straightforward for queries that can be evaluated in polynomial time[1]. The extension to arbitrary second-order queries is based on two facts.

First, it is a well-known result in finite model theory, that the problems expressible in second-order logic are precisely the problems in the polynomial-time hierarchy PH. The second fact is a theorem from structural complexity theory, due to Regan and Schwentick [8], which gives a very special presentation of arbitrary problems in PH in terms of a single bit of a #P-function.

Theorem 4.1 (Regan, Schwentick).
For every problem D in the polynomial-time hierarchy and every polynomial q, there exist a #P-function f and a polynomial t such that for every input x (of length n) for D, the binary representation of $f(x)$ has the form

$$y0^{q(n)}D(x)0^{q(n)}z$$

where $y, z \in \{0,1\}^*$, $|z| = t(n)$ and where $D(x) = 1$ if $x \in D$ and $D(x) = 0$ otherwise.

Theorem 4.2. *Let ψ be a second-order query. Then the problem of computing the reliability R_ψ is in the class* $\mathrm{FP}^{\#\mathrm{P}}$.

Proof. Let ψ be a k-ary second-order query. Since $R_\psi(\mathfrak{D}) = 1 - [H_\psi(\mathfrak{D})/n^k]$ and

$$H_\psi(\mathfrak{D}) = \sum_{\bar{a} \in A^k} H_{\psi(\bar{a})}(\mathfrak{D})$$

where $H_{\psi(\bar{a})}(\mathfrak{D})$ is the expected error of the Boolean query $\psi(\bar{a})$, it suffices to show that the expected error of *Boolean* queries is in $\mathrm{FP}^{\#\mathrm{P}}$. In the sequel, ψ is a second-order sentence.

We have to show that there exists an $\mathrm{FP}^{\#\mathrm{P}}$-algorithm which, given an unreliable database $\mathfrak{D} = (\mathfrak{A}, \mu)$, computes $H_\psi(\mathfrak{D})$, which in this case is just the expectation that $\psi^\mathfrak{A} \neq \psi^\mathfrak{B}$ for randomly chosen $\mathfrak{B} \in \Omega(\mathfrak{D})$.

We first evaluate ψ on $\mathfrak{A}$ and store the truth value $\psi^\mathfrak{A}$ for later use. Since $\mathrm{PH} \subseteq \mathrm{P}^{\#\mathrm{P}}$ this is within complexity $\mathrm{FP}^{\#\mathrm{P}}$. We then calculate (in polynomial-time) the least natural number g such that $\nu(\mathfrak{B}) \cdot g \in \mathbb{N}$ for all $\mathfrak{B} \in \Omega(\mathfrak{D})$.

We can compute g in polynomial time as follows. Given $(\mathfrak{A}, .\mu)$ we can easily compute the sequence of probabilities $\nu(R\bar{a})$ for all atomic statments of $\mathfrak{A}$. Further we can assume that these probabilities are normalized, i.e. numerator and denominator are relatively prime. Initially let $g' = 1$. Then successively calculate the gcd b between the so-far result g' and the next denominator d. If $b = d$ then d is a factor of g' and we continue the loop. Otherwise take $g' \cdot d/b$ as g' for the next round. In the end let $g = g'$.

[1] In fact even for all UP-queries, where UP (for *unambigous polynomial time*) is the class of languages decided by nondeterminstic polynomial-time Turing machines with at most one accepting computation path.

This indeed yields the smallest possible g. According to the choice of g there can be no $g' \leq g$ which always produces an integer when multiplied with the denominators of the probabilities. The existence of a $g' \leq g$ which always produces integers when multiplied with the probabilities themselves would imply the existence of a factor $h > 1$ occuring in *all* numerators. Since $g' \mid g$ this contradicts the fact that all probabilities were assumed to be given in a normalized representation.

We claim that there exists a nondeterministic algorithm M such that $g \cdot \Pr[\mathfrak{B} \models \psi]$ is computable in polynomial time from the number of accepting computation paths of M on $\mathfrak{D}$. From $g \cdot \Pr[\mathfrak{B} \models \psi]$ and $\psi^\mathfrak{A}$, we can immediately calculate $H_\psi(\mathfrak{D})$.

The algorithm M nondeterministically guesses truth values for all atomic statements $R\bar{a}$ on $\mathfrak{A}$. In the resulting computation tree each leaf sees a different database $\mathfrak{B} \in \Omega(\mathfrak{D})$. At each leaf, the computation tree is then split again $\nu(\mathfrak{B}) \cdot g$ times to reflect the probability of the guessed database. (Recall that $\nu(\mathfrak{B})$ and g are computable in polynomial time.) Then ψ is evaluated on $\mathfrak{B}$.

In the case that ψ can be evaluated in polynomial-time, the algorithm just checks at each leaf, whether $\mathfrak{B} \models \psi$. Thus the number of accepting computation paths will just be $g \cdot \Pr[\mathfrak{B} \models \psi]$.

In the case that ψ is in PH but not in P, we apply Theorem 4.1. Let the polynomial q be chosen in such a way that $2^{q(n)}$ exceeds the number of leaves of the computation tree constructed above (where n is the length of the input $\mathfrak{D}$). At each leaf we apply to $\mathfrak{B}$ the nondeterministic polynomial-time algorithm whose number of accepting computations on $\mathfrak{B}$ has a binary representaion of the form

$$y0^{q(n)}\psi^\mathfrak{B}0^{q(n)}z$$

with $|z| = t(n)$, t a polynomial. The total number of accepting paths of M on $\mathfrak{D}$ is the sum over all these numbers. Note that we add up less than $2^{q(n)}$ numbers of this kind, and in each of these, the relevant bit $\psi^\mathfrak{B}$ is separated from the 'junk' y and z by $q(n)$ 0's. Thus, even after adding up, the 'junk' cannot interfere with the relevant information, i.e. in the total sum we can still see the sum of the relevant bits $\psi^\mathfrak{B}$, which is precisely $g \cdot \Pr[\mathfrak{B} \models \psi]$. ☐

Thus, we have proved that the reliability problem is in $\mathrm{FP}^{\#\mathrm{P}}$ for all second-order queries, and that there even exist conjunctive queries whose reliability problem is $\mathrm{FP}^{\#\mathrm{P}}$-complete.

5 Approximability and absolute reliability

In this section we will establish an approximability result for the reliability of existential and universal queries. We use a rather strong notion of approximation algorithms which is often used for #P-functions, namely *fully polynomial-time randomized approximation schemes*. Let f be a function with values in an ordered numerical domain like $\mathbb{N}$, $\mathbb{Q}$ or $\mathbb{R}$, and let ε, δ be positive rational numbers. We say that f has a *polynomial time randomized (ε, δ) approximization algorithm*, if there is a polynomial-time randomized algorithm M, such that for all inputs $x \in \mathrm{dom}(f)$

$$\Pr\left[\frac{|M(x) - f(x)|}{f(x)} > \varepsilon\right] < \delta. \quad (1)$$

A *fully polynomial-time randomized approximization scheme* (FPTRAS) for f is an algorithm M that takes besides x also ε and δ as inputs, such that (1) holds for all $\varepsilon, \delta > 0$ and the running time of M is polynomially bounded in the length of x, and also in $1/\varepsilon$ and $1/\delta$. It is known that a number of #P-complete functions admit an FPTRAS. We will show that this is also the case for the probabilities of existential queries.

We proceed as follows: We will reduce the problem of calculating the probability of an existential query to the problem of calculating the probability that a given propositional formula in kDNF is true given a probability for each of its variables. For this problem, we can show the existence of an FPTRAS using a result of Karp and Luby [4].

From FPTRAS for the *probabilities* of existential sentences we obtain a slightly weaker notion of approximability for the *reliability* of any existential or universal query. This weaker version is sufficient for most practical purposes. We then discuss whether it can be strengthened to an FPTRAS for reliability.

Definition 5.1. Let C be a class of propositional formulae. Then $\#C$ is the corresponding counting problem and Prob-C is the corresponding probability problem:

- $\#C$: Given a formula $\varphi \in C$, calculate the number of assignments (to the variables in φ) that make φ *true*.
- Prob-C: Given a propositional formula $\varphi \in C$, and a probability function ν that assigns to each variable X of φ a probability $\nu(X) = \Pr[X = 1]$, calculate the induced probability $\nu(\varphi)$ of φ being *true*.

In the following we are interested in DNF, the class of propositional formulae in disjunctive normal form, and kDNF, the class of DNF-formulae where each disjunct has at most k literals.

Theorem 5.2 (Karp, Luby).
The problem #DNF *admits an* FPTRAS.

We use this theorem to derive a similar result for the problems Prob-kDNF.

Theorem 5.3. *For all $k \in \mathbb{N}$, the problem Prob-k*DNF *admits an* FPTRAS.

Proof. Let φ be a kDNF formula and let $\nu : \{X_1, \dots, X_m\} \to [0, 1]$ be a probability function on the variables of φ. We will transform (φ, ν) into an appropriate instance φ'' for $\#DNF$.

For each variable X of φ, with probability $\nu(X) = p/q$ we introduce new propositional variables $\bar{Y} = Y_{\ell-1}, \dots, Y_0$, where $\ell = \text{len}(q)$, the length of the shortest binary representation of q. We write $\text{val}(\bar{Y})$ for the natural number with binary representation $\bar{Y}$.

We first note that for every $b \in \mathbb{N}$ and $\ell \geq \text{len}(b)$, we can efficiently construct DNF-formulae of length $O(\ell^2)$ expressing "$\text{val}(\bar{Y}) < b$" and "$\text{val}(\bar{Y}) \geq b$", respectively. Indeed, if $b_{\ell-1} \cdots b_0$ represents b in binary then "$\text{val}(\bar{Y}) < b$" is expressed by

$$\bigvee_{\substack{i<\ell \\ b_i=1}} \Big(\neg Y_i \wedge \bigwedge_{\substack{i<j<\ell \\ b_j=0}} \neg Y_j\Big).$$

The formula for "$\text{val}(\bar{Y}) \geq b$" is similar.

We now replace in φ every occurence of the literal X by the formula "$\text{val}(\bar{Y}) < p$" and every occurrence of $\neg X_i$ by "$\text{val}(\bar{Y}) \geq p$". This operation is performed for all variables X of φ (taking a sequence of fresh variables $\bar{Y}$ for each X) and the resulting formula is transformed into DNF to obtain φ'. Note that this process may increase the length of φ' exponentially in k but (since the original formula is in kDNF) only polynomially in the length of φ and in the number of bits used for the probabilities.

In the case that all probabilities $\nu(X)$ are dyadic rationals, i.e. of the form $p/2^\ell$, we are done. Indeed, for each variable X of the original formula φ, we have ℓ variables $\bar{Y}$; there are p assignments satisfying the formula "$\text{val}(\bar{Y}) < p$" corresponding to the literal X, and $2^\ell - p$ assignments satisfying "$\text{val}(\bar{Y}) \geq p$" corresponding to the literal $\neg X$. Hence the probability $\nu(\varphi)$ is just number of assignments satisfying φ' divided by the total number of assignments.

However, if the denominators of the probabilities $\nu(X)$ are not powers of two, this is no longer the case. Consider the case that $\nu(X) = p/q$ with $q < 2^\ell$. We still have p assignments satisfying the formula "$\text{val}(\bar{Y}) < p$" (corresponding to X) and $2^\ell - p$ assignments satisfying "$\text{val}(\bar{Y}) \geq p$" (corresponding to $\neg X$). However, we should have only $q - p < 2^\ell - p$ assignments corresponding to $\neg X$. Call an assignments to $\bar{Y}$ *illegal* if it satisfies the formula "$\text{val}(\bar{Y}) \geq q$". The probability $\nu(\varphi)$ is the number of *legal* assignments satisfying φ' divided by the total number of legal assignments. We can easily calculate the total number of legal assignments (the product of all denominators of the probabilities $\nu(X)$) and hence the total number of illegal assignments. To determine the number of legal assignments satisfying φ', we transform φ' into a new formula φ'' that is implied by φ' and, in addition, is satisfied by all illegal assignments. For instance, we can take for φ'' the disjunction of φ' with the formulae "$\text{val}(\bar{Y}) \geq q$" for all sequences $\bar{Y}$ that we have introduced for the variables of φ. Clearly φ'' is in DNF and can be constructed in polynomial time.

The number of legal assignments satisfying φ' is the number of all assignments satisfying φ'' minus the total number of illegal assigments. Hence, by using the FPTRAS of Karp and Luby for #DNF to approximate the number of assignments for φ'', we get an FPTRAS for calculating $\nu(\varphi)$. ☐

We are now in a position to prove our approximability result.

Theorem 5.4. *Let ψ be any boolean existential query. Then the probability $\nu(\psi)$ (that ψ holds in the actual database) admits an* FPTRAS.

Proof. Let $\psi = \exists \bar{y} \varphi(\bar{y})$ be an existential query, where φ is quantifier-free and in kDNF, for some $k \in \mathbb{N}$.

Given a probabilistic database $\mathfrak{D} = (\mathfrak{A}, \mu)$ with n elements, we first calculate (as explained in Sect. 2) the probabilities $\nu(\alpha)$ for all atomic statements α on $\mathfrak{A}$. We then replace the quantifiers in ψ by disjunctions over all possible values:

$$\psi(\bar{x}) = \exists \bar{y} \varphi(\bar{x}, \bar{y}) \longmapsto \bigvee_{\bar{b}} \varphi(\bar{x}, \bar{b}) =: \psi'(\bar{x}).$$

Now we let ψ'' be the propositional formula obtained from ψ' by replacing equalities by their truth values and considering atomic statements $R\bar{c}$ as propositional variables. Note that the number of variables per conjunction does not depend on the size of $\mathfrak{D}$, so ψ'' is a propositional formula in kDNF whose length is polynomial in n, and the function ν defines probabilities for the variables of ψ''. Obviously $\nu(\psi'')$ is just the expectation that ψ holds in a randomly chosen $\mathfrak{B} \in \Omega(\mathfrak{D})$. By Theorem 5.3 we have an FPTRAS for calculating $\nu(\psi'')$. □

Theorem 5.4 implies that the reliability of existential and universal queries can be approximated in the following sense.

Corollary 5.5. *Let ψ be an existential or a universal query. Then there exists a polynomial-time randomized algorithm M such that for all unreliable databases $\mathfrak{D}$ and all $\varepsilon, \delta > 0$*

$$\Pr[|R_\psi(\mathfrak{D}) - M(\mathfrak{D})| > \varepsilon] < \delta.$$

Proof. If ψ is an existential or universal *Boolean* query this is immediate from Theorem 5.4, since for every database $\mathfrak{D} = (\mathfrak{A}, \mu)$ either $H_\psi = \nu(\psi)$ and $R_\psi = 1 - \nu(\psi)$ or vice versa, depending on whether ψ holds in $\mathfrak{A}$, or not.

For k-ary queries with $k > 0$, we can approximate H_ψ by taking the sum of appropriate approximations (with error ε/n^k and probability δ/n^k) for $H_\psi(\bar{a})$ over all k-tuples $\bar{a}$. Since $R_\psi = 1 - H_\psi/n^k$ this gives the desired result. □

The notion of approximability used in Corollary 5.5 is technically weaker than the existence of an FPTRAS, since ε bounds the absolute rather than the relative difference of $M(\mathfrak{D})$ and $R_\psi(\mathfrak{D})$. For practice, this weaker notion is of course sufficient: It is certainly good enough to compute the reliability up to an absolute error of, say, 1/10'000, even if the actual value is very close to 0 (and we hence cannot say anything about the relative error).

Nevertheless it is theoretically interesting, whether the reliability of existential or universal queries admits an FPTRAS. By analyzing the associated decision problem, we see that this is unlikely.

Definition 5.6. Fix any query ψ. Let AR_ψ be the set of all unreliable databases $\mathfrak{D}$ where $R_\psi(\mathfrak{D}) = 1$. We call determining whether a given $\mathfrak{D}$ belongs to AR_ψ the *absolute reliability problem.*

Lemma 5.7. *Let ψ be a quantifier free query. Then* $\mathrm{AR}_\psi \in \mathrm{P}$.

Lemma 5.8. *Let ψ be a polynomial-time evaluable query. Then* $\mathrm{AR}_\psi \in$ Co-NP.

The proofs are simple: In the first case R_ψ is computable in polynomial time. In the second case guess a database $\mathfrak{B}$ and check whether the truth values $\psi^{\mathfrak{A}}$ and $\psi^{\mathfrak{B}}$ differ. For our further observations we take a closer look at existential queries.

Lemma 5.9. *There exist existential queries ψ such that AR_ψ is* Co-NP*-hard.*

Proof. We reduce the problem of 4-colourability of graphs to the complement of AR_ψ for some existential query ψ.

Besides the edge relation E we let our language contain two unary relations R_1, R_2 together giving one of 4 colours for each node. Now consider the query

$$\psi = \exists x \exists y (Exy \land (R_1x \leftrightarrow R_1y) \land (R_2x \leftrightarrow R_2y))$$

expressing that two connected nodes share the same color, i.e. that R_1, R_2 do *not* represent a correct 4-colouring.

Given an arbitrary graph $G = (V, E)$ we now construct a database $\mathfrak{D} = (\mathfrak{A}, \mu)$. The universe V and graph relation E are taken unchanged. For our given database $\mathfrak{A}$ we let $R_1, R_2 = \varnothing$ giving all nodes the same colour. The error function μ is such that $\mu(Euv) = 0$ and $\mu(R_iv) = 1/2$ for all nodes $u, v \in V$, $i = 1, 2$. Note[2] that $\mathfrak{A} \models \psi$. For this construction the existence of a 4-colouring for G is equivalent to $\mathfrak{D} \notin AR_\psi$. □

For our existential non-4-colouring query ψ we can approximate the reliability R_ψ, however we do not know how to approximate the expected error H_ψ. A simple observation shows that the existence of an FPTRAS for the reliability of all existential queries without precondition for $\mathfrak{D}$ (and hence the approximability of both H_φ and R_φ for all existential or universal queries φ) is highly unlikely.

Lemma 5.10. *Let f be any function such that the associated decision problem $\{x : f(x) > 0\}$ is* NP*-hard. If, for some $\varepsilon < 1$, $\delta < 1/2$, f admits a randomized polynomial-time (ε, δ)-approximation algorithm, then* NP $\subseteq$ BPP.

The proof is straightforward.

Note that the decision problem associated with the expected error H_ψ is the complement of the absolute reliability problem AR_ψ. Hence there exist existential queries (such as the non-4-colouring query) whose expected error does not admit an FPTRAS, unless NP $\subseteq$ BPP.

As we cannot expect any positive results using an FPTRAS for reliability calculations we take a look back at Corollary 5.5. Relaxing our bound on the allowed error gave us a positive, if weaker, approximability result. We will now derive a similar result for all polynomial-time evaluable queries. The proof is based on methods used in [4] for developing an FPTRAS for #DNF. We will make some modifications in order to obtain a result for all polynomial-time evaluable queries.

Lemma 5.11 (Karp, Luby).
Let $\{X_i\}$ be a sequence of independent random variables, all identically distributed with values in $[0, 1]$ and expectation $p := \mathrm{E}(X_i) < 0.5$ for all i. Then for all $\varepsilon \in (0, 1)$,

$$\Pr\left[\left|\frac{X_1 + \cdots + X_t}{t} - p\right| > \varepsilon \cdot p\right] < 2 \cdot e^{\left(\frac{-2\varepsilon^2 tp}{9(1-p)}\right)}.$$

Theorem 5.12. *Let ψ be a polynomial-time evaluable query. Then there exists a polynomial-time randomized algorithm M such that for all probabilistic databases $\mathfrak{D}$ and $\varepsilon, \delta > 0$*

$$\Pr[|R_\psi(\mathfrak{D}) - M(\mathfrak{D})| > \varepsilon] < \delta.$$

Proof. The proof is based on methods used by Karp and Luby in [4] for constructing an FPTRAS for #DNF.

In case ψ is not a Boolean query we use the same idea as in the proof of Lemma 5.5 and approximate the query reliability using the sum over all approximations for each possible valuation of the free variables, if with stricter bounds. For the following let ψ be a polynomial-time evaluable Boolean query and let[3] $\xi \in (0, 1/2)$ be a rational.

Let $\mathfrak{D} = (\mathfrak{A}, \mu)$ be a probabilistic database and $\varepsilon, \delta > 0$. We begin with a modification of $\mathfrak{D}$ and ψ. Let R be a new unary relation and let c, d be two new constants. We define a new database $\mathfrak{D}' = (\mathfrak{A}', \mu')$ and query ψ', where

[2] Quietly ignoring the case where $E = \varnothing$.

[3] Note that ξ is chosen prior to knowing any one of $\mathfrak{D}$, ε or δ.

- $\mathfrak{A}' = (\mathfrak{A}, R, c, d)$ with $R = \varnothing$ and $c \neq d$,
- $\mu'(P\bar{a}) = \begin{cases} \xi & \text{if } P = R \text{ and } \bar{a} = c \text{ or } \bar{a} = d, \\ \mu(P\bar{a}) & \text{otherwise,} \end{cases}$
- $\psi' = (\psi \vee Rc) \wedge Rd$.

Now let $\mathfrak{B}'$ be a random variable taking values in $\Omega_{\mathfrak{D}'}$ with distribution ν' as induced by μ'. We define a random variable X through $X = \psi^{\mathfrak{B}'}$ and let $\{X_i\}_{i>0}$ be a sequence of independent identically distributed random variables where each X_i is distributed as X. It follows that for $p := E(X) = \nu'(\psi')$ we have

$$0 < \xi^2 \leq p \leq \xi < 1/2.$$

For the upper bound of p note that on top-level ψ' contains a conjunction with the atom Rd. Since R is empty in $\mathfrak{A}$, $\nu'(Rd) = \mu'(Rd) = \xi$. It follows that $p = q \cdot \xi$ for some $0 \leq q \leq 1$. For the lower bound note that ψ' is *true* if Rc and Rd hold. For the number of repetitions in Lemma 5.11 we let

$$t = t(\varepsilon, \delta) = \left\lceil \frac{9}{2\,\xi\,\varepsilon^2} \ln \frac{1}{\delta} \right\rceil .$$

Due to ξ being fixed, t is bounded by a polynomial in $\frac{1}{\varepsilon}$ and $\frac{1}{\delta}$. Further the X_i are polynomial-time computable, too. It follows that our approximator $\widetilde{X} = \frac{X_1 + \cdots + X_t}{t}$ is polynomial-time computable. The modifications of $\mathfrak{D}$ and ψ lead to our X_i satisfying the preconditions of Lemma 5.11. A few simple transformations on the probability bound in the result of Lemma 5.11 show that for our choice of t we get

$$\Pr\left[\left|\frac{X_1 + \cdots + X_t}{t} - p\right| > \varepsilon \cdot p\right] < \delta. \tag{2}$$

It follows from our construction that

$$\nu(\psi) = \frac{(p - \xi^2)}{(\xi - \xi^2)}\ , \tag{3}$$

hence using

$$\alpha = \frac{(\widetilde{X} - \xi^2)}{(\xi - \xi^2)} \tag{4}$$

as approximation for $\nu(\psi)$ satisfies

$$\Pr\left[|\alpha - \nu(\psi)| > 2 \cdot \varepsilon\right] < \delta. \tag{5}$$

To see why (5) holds, first note that $\xi < 0.5$ and therefore

$$\frac{\xi^2}{\xi - \xi^2} < 1\ . \tag{6}$$

We use (3) and (4) to substitute $\widetilde{X}$ and p in (2) and obtain

$$\Pr[|(\xi - \xi^2)(\alpha - \nu(\psi)| > \varepsilon \cdot (\xi^2 + (\xi - \xi^2)\nu(\psi)] < \delta\ ,$$

or equivalently

$$\Pr\left[\left|\alpha - \nu(\psi)\right| > \varepsilon \cdot \left(\frac{\xi}{\xi - \xi^2} + \nu(\psi)\right)\right] < \delta\ .$$

Due to (6) and $\nu(\psi) < 1$ we are done.

Together with above observations that the X_i are polynomial-time computable and that t bounded by a polynomial in $\frac{1}{\varepsilon}$ (and hence in $2/\varepsilon$) and $1/\delta$, (5) shows that our approximator satisfies the required bounds. Given an ε as allowed error we simply use $\varepsilon/2$ in our algorithm. $\square$

6 Metafinite databases

We have so far considered databases given by finite relational structures and queries represented in a purely logical language (like first-order logic, Datalog, or second-order logic). We briefly discuss the extension to a more general setting, where we may also have *functions into possibly infinite interpreted domains* and *aggregates* or *multiset operations.* The formal framework is that of *metafinite model theory* as presented in [1]. As pointed out in [1], we believe that aggregates are adequately modeled by multiset operations.

Fix an infinite structure $\mathfrak{R}$ together with a collection of multiset operations. In typical applications $\mathfrak{R}$ will be many-sorted, with domains like $\mathbb{N}$, $\mathbb{Q}$ etc. and (as in SQL) with multiset operations like $\sum$, min, max, average, ... over appropriate domains. We assume that R contains two distinguished constants $0, 1$ and functions corresponding to the Boolean operations. Further we require that all functions and multiset operations present in $\mathfrak{R}$ are efficiently computable (with respect to an appropriate machine model).

Definition 6.1. An *unreliable functional database* over $\mathfrak{R}$ is a pair $\mathfrak{D} = (\mathfrak{A}, \nu)$, where $\mathfrak{A} = (A, \mathcal{F})$ consists of a finite set A and a finite set $\mathcal{F}$ of functions of the form $f : A^k \to R$ (where R is the universe of $\mathfrak{R}$). The probability function ν assigns to every equation of the form $f(\bar{a}) = r$ (where $f \in \mathcal{F}$, $\bar{a}$ is a tuple over A whose length matches the arity of f, and $r \in R$) the probability $\nu(f(\bar{a}) = r)$ that this equation holds in the actual database. These events are assumed to be independent for $f(\bar{a}) = r$, $f'(\bar{a}') = r'$ provided that f, f' or $\bar{a}, \bar{a}'$ are distinct. Further we assume that for all f and $\bar{a}$, only finitely many values appear with positive probability — i.e. $\{r \in R : \nu(f(\bar{a}) = r) > 0\}$ is finite — and that the probabilities $\nu(f(\bar{a}) = r)$ are defined consistently, that is

$$\sum_{r \in R} \nu(f(\bar{a}) = r) = 1$$

for all f and all $\bar{a}$.

This assigns a probability $\nu(\mathfrak{B})$ to every database $\mathfrak{B}$ of the same format as $\mathfrak{A}$. It is an immediate consequence of the definition just given, that for every unreliable functional database $(\mathfrak{A}, \mu)$

- The number of databases $\mathfrak{B}$ with $\nu(\mathfrak{B}) > 0$ is finite, and in fact bounded by $2^{p(n)}$ where p is a polynomial and n is the length of an appropriate encoding of $(\mathfrak{A}, \mu)$.
- For every $\mathfrak{B}$, the probability $\nu(\mathfrak{B})$ is efficiently computable.

In the terminology of [1], functional databases are a special class of metafinite algebras. A query on databases of this kind is a global function F, associating with $\mathfrak{A}$ a function $F^{\mathfrak{A}} : A^k \to R$. Note that for the case $k = 0$, the query associates with every database $\mathfrak{A}$ a numerical value $F^{\mathfrak{A}} \in R$.

A query language can be defined as a calculus of terms, with the proviso, that variables range over the finite set A only (not over R)[4].

[4]As shown in [1], by forbidding the use of variables ranging over the infinite interpreted domains in $\mathfrak{R}$, one can avoid most of the inherent evils pertaining to infinite structures.

For instance, the quantifier-free queries on databases $(A, \mathcal{F})$ over $\mathfrak{R}$ contain the terms $f(\bar{x})$ for every name f of a function in $\mathcal{F}$ and are closed under applications of functions in $\mathfrak{R}$ (with fixed interpretations like $+$ or $\cdot$) to terms. Let us further assume that we have in $\mathfrak{R}$ multiset operations $\sum, \prod, \max, \min$ etc. We then get an appropriate notion of *first-order queries* by taking the closure of the quantifier-free queries under applications of multiset operations and applications of functions in $\mathfrak{R}$. Multiset operations play a similar, but more general rôle than quantifiers do in the relational setting. For instance, given a term $F(\bar{x}, y)$, one defines a new term $G(\bar{x})$ of the form $\sum_y F(\bar{x}, y)$. In particular, the operations max and min can be seen as more general variants of existential and universal quantifiers. We refer to [1] for a more detailed treatement and examples.

Most of the methods and results for the finite relational case go through for metafinite databases. For instance, the reliability of quantifier-free queries is still polynomial-time computable. The reliability of first-order queries is in $\mathrm{FP}^{\#\mathrm{P}}$. The proof is essentially the same as for finite relational databases: On each branch of the computation tree one of the finitely many possible databases is guessed; the computation is then split further according to the probability of the guessed database. Finally the query is evaluated and the result compared against the result on the observed database.

The class of *second-order queries* on metafinite databases is defined by extending first-order queries by multiset operations over relations (rather than tuples). For instance, given a term $F(S, \bar{T}, \bar{x})$ with free second-order variables $S, \bar{T}$ and free first-order variables $\bar{x}$, we build a new term $\sum_S F(S, \bar{T}, \bar{x})$. For reasonable choices of second-order operations (like $\sum$, max, min) the expressive power of second-order queries lies between #P and PSPACE. It is in $\mathrm{FP}^{\mathrm{CH}}$ where CH is the so-called counting polynomial hierarchy that has been introduced and studied by Wagner [12]. There are a number of equivalent definitions of the class $\mathrm{FP}^{\mathrm{CH}}$; for instance it is the closure of the polynomial-time computable functions under the #-operation. Other characterizations involve second-order counting quantifiers or closures of some basic arithmetical functions under exponential summation and substitution. For details we refer to [12].

The simple algorithm described above for computing the reliability of first-order queries applies also to second-order queries. It shows that the reliability problem of every second-order query is also in $\mathrm{FP}^{\mathrm{CH}}$.

The following theorem summarizes these results for some particular cases.

Theorem 6.2. *Let $\mathfrak{R}$ be the standard arithmetic over $\mathbb{N}$ or the field of rational numbers, equipped with the multiset operations $\sum, \max, \min$. Further, let ψ be a query on functional databases over $\mathfrak{R}$.*

(i) If ψ is quantifier-free then the reliability of ψ is computable in polynomial time.

(ii) If ψ is first-order, then its reliability problem is in the class $\mathrm{FP}^{\#\mathrm{P}}$.

(iii) If ψ is second-order, then the reliability of ψ is in $\mathrm{FP}^{\mathrm{CH}}$.

Given that metafinite databases can handle numerical values we have the further advantage that the error probabilities can themselves be considered as part of the database. The reliability and the expected error of a query are themselves database queries. In addition to the *complexity* of the reliability problem we can ask whether the reliability of a query is itself *expressible* in a given query language. For instance, it is a result in [1], that the reliability of every quantifier-free query over finite relational databases is first-order definable in an appropriate metafinite setting. Similarly the reliability of any first-order or second-order query over finite relational databases is second-order definable in such a setting.

Acknowledgement

We are grateful to Thomas Schwentick for pointing out Theorem 4.1 to us, and to Jacobo Toran and Lane Hemaspaandra for information and references concerning counting complexity classes.

References

[1] E. Grädel and Y. Gurevich, *Metafinite Model Theory*, Information and Computation **140** (1998), 26–81.

[2] C. Hirsch, *The Reliability of Queries*, Diploma Thesis (RWTH-Aachen, 1998).

[3] D. Johnson, *A Catalog of Complexity Classes*, in: J. van Leeuven (Ed.), *Handbook of Theoretical Computer Science*, Vol. A: Algorithms and Complexity, Elsevier/MIT Press (1990), 67–161.

[4] R. Karp and M. Luby, *Monte Carlo algorithms for enumeration and reliability problems*, Proceedings of the 24th Symposium on Foundations of Computer Science FOCS 1983, 56–64.

[5] V. Lakshmanan and F. Sadri, *Probabilistic deductive databases*, Proceedings of the International Logic Programming Symposium, MIT Press (1994), 197–207.

[6] V. Lakshmanan and V. Subrahmanian, *Probview, A flexible probabilistic database system*, ACM Transactions on Database Systems **22** (1997), 419–469.

[7] C. Papadimitriou, *Computational Complexity*, Addison-Wesley (1994).

[8] K. Regan and T. Schwentick, *On the Power of One Bit of a #P Function*, Proceedings of the Fourth Italian Conference on Theoretical Computer Science (1992), 317-329.

[9] M. de Rougemont, *The reliability of queries*, Proc. 14th ACM Symp. on Principles of Database Systems PODS (1995), 286–291.

[10] V. Subrahmanian, *Stable semantics for probabilistic deductive databases*, Information and Computation **110**(1) (1994), 42–83.

[11] L. Valiant, *The complexity of enumeration and reliability problems*, SIAM J. Computing **8** (1979), 410–421.

[12] K. Wagner, *Some observations on the connection between counting and recursion*, Theoretical Computer Science **47** (1986),131–147.

[13] E. Zimányi, *Query evaluation on probabilistic relational databases*, Theoretical Computer Science **171** (1997), 179–219.