A Compositional Framework for Complex Queries over Uncertain Data

Michaela Götz
Cornell University
Ithaca, NY
goetz@cs.cornell.edu

Christoph Koch
Cornell University
Ithaca, NY
koch@cs.cornell.edu

ABSTRACT

The ability to flexibly compose confidence computation with the operations of relational algebra is an important feature of probabilistic database query languages. Computing confidences is computationally hard, however, and has to be approximated in practice. In a compositional query language, even very small errors caused by approximation can lead to an entirely incorrect result: A selection operation on an approximated probability can incorrectly keep or drop a tuple even if the probability value has been approximated to a very narrow confidence interval.

In this paper, we study the query evaluation problem for compositional query languages for probabilistic databases with particular focus on providing overall result quality guarantees in the face of approximate intermediate results. We present a framework for evaluating compositional queries based on a new representation system that can capture uncertainty about probabilities. More specifically, we consider probability intervals instead of exact probabilities, interpreting tuples obtained by selection on approximate values as unreliable.

We study the complexity of query evaluation over our new model. We present efficient confidence computation algorithms which compute bounds that are close to tight for important classes. For deciding a selection predicate, we show that no efficient randomized algorithm exists unless BPP⊃NP. Still we are able to efficiently guess robust predicates with a good error bound. Putting all these pieces together in our framework, we evaluate queries using a decomposition into a relational algebra plan and an approximation plan. The latter allows to successively improve accuracy and error bounds, while the relational algebra plan only has to be executed once.

1. INTRODUCTION

In many applications of data management, uncertainty is an inherent feature of the data. For example, large collections of uncertain data arise in sensor networks [17] because the measurements of sensors are inaccurate and can correspond to several high-level events. Other examples include information retrieval [7], scientific databases [6], and data cleaning [2]. In this paper we think of uncertain data as a finite set of possible worlds – one for each alternative of the uncertain data – together with a probability distribution over these worlds. This model is known as probabilistic database.

The Query Language. We consider a powerful query language over probabilistic databases that allows to study "what if"–scenarios and to make decisions based on a comparison of the probabilities of events. It consist of positive relational algebra and *confidence computation*. The confidence operation computes for each tuple its probability. Decisions can be made based on these confidence values through *selection predicates involving confidences of events*. The class of predicates that we consider is very general. It consists of Boolean combinations of inequalities involving arithmetic expressions over confidences and constants. To give a very simple example one can select all tuples whose confidence is above a certain threshold. This language was introduced in [14].

A Motivating Example: Data Cleaning. One application of our framework is efficient data cleaning. Given a probabilistic database, integrity constraints can be enforced by assigning the worlds violating the constraints a probability of zero and normalizing the probabilities of the remaining worlds. In general removing worlds violating constraints is $\#P$–hard. A few heuristics have been proposed [15]. Other lines of work restrict the constraints to soft key constraints [11] or to constraints involving aggregation over probabilistic XML [5].

Our framework enables efficient enforcement of equality generating dependencies (egds) [1, 10]. The egds are taken into consideration when approximating confidences without adding any overhead to the complexity of query evaluation.

Complexity. Computing the confidence of a tuple is #P–hard [6, 8]. However, an efficient, accurate, and reliable randomized approximation algorithm exists. This algorithm [13] can output an estimate which is arbitrarily close to the true confidence with high probability in polynomial time. We call such an algorithm a fully polynomial time randomized approximation scheme (FPRAS).[1] If we would like to compute the confidence p of a tuple we run the approximation algorithm that outputs an estimate $\hat{p}$ that is close to p with high probability. Here, $\hat{p}$ is a random variables describing

[1] There are also deterministic approximation algorithms, see [18].

ICDT 2009, March 23–25, 2009, Saint Petersburg, Russia.

the output of the algorithm. The guarantee of the algorithm given δ and ϵ is that with probability at least $1 - \delta$ the estimate $\hat{p}$ is close to the true value p, i.e.

$$\Pr\left[\frac{\hat{p}}{1+\epsilon} \leq p \leq \frac{\hat{p}}{1-\epsilon}\right] \geq 1 - \delta.$$

Also, deciding a predicate is hard. We prove that already deciding simple predicates of the form "confidence OP c", where OP $\in \{<, \leq, \geq, >\}$ is #P–hard. We further show that no non-trivial predicate can be estimated with an error probability bounded by 1/3 unless BPP⊃NP, which is considered unlikely. BPP is the class of decision problems for which a poly–time randomized algorithm exists outputting the correct answer with probability $\geq 2/3$. This negative result is a worst case result. There are cases in which we can output the value of a predicate correctly with high probability.

Beyond Probabilistic Databases. A major challenge arises in the composition of randomized approximation operations. The approximate results cannot be captured as a probabilistic database any longer. This is because the selection of tuples based on their approximate confidence is unreliable. The following example illustrates this.

Example 1.1. Let us assume that we want to select all tuples with confidence at least c. We can use a FPRAS to compute an estimate $\hat{p}$ of the confidence p of a tuple that is fairly accurate with high probability, i.e. $\Pr\left[\frac{\hat{p}}{1+\epsilon} \leq p \leq \frac{\hat{p}}{1-\epsilon}\right]$ $\geq 1 - \delta$. Now, there are three possible cases:

1.If $\frac{\hat{p}}{1+\epsilon} \geq c$ then with probability at least $1-\delta$ we should select the tuple.

2. If $\frac{\hat{p}}{1-\epsilon} < c$ then with probability at least $1-\delta$ we should NOT select the tuple.

3. Otherwise we cannot make a guess of whether or not the tuple should be selected.

In the first case the probability of the tuple is between $1 - \delta$ and 1. In the second case the probability of the tuple is between 0 and δ. In the third case we can only give the trivial bounds on the probability which are 0 and 1. □

We introduce a succinct representation system generalizing the one of [3] that can capture interval probabilities instead of exact probabilities. The semantics is that any probability distribution consistent with the intervals is possible. Modeling uncertainty of the probabilities has been done for probabilistic databases [16, 21] and XML [9, 22]. In all prior work the dependencies considered are so basic that confidence computation becomes easy. In our work we consider arbitrary dependencies.

Complexity with Interval Probabilities. Computing the confidence exactly is #P–hard even if the probabilities are known exactly, but a FPRAS is known. We show that only knowing interval probabilities makes this problem harder. However, we present an efficient randomized approximation algorithm for the case that the interval probabilities were introduced by the unreliability of selection predicates.

Framework. We present a framework for efficiently evaluating arbitrarily composed queries. None of the existing systems [3, 6, 20] can do this. In this framework queries are decomposed into an approximation plan and a relational algebra plan. The former allows to successively improve the accuracy and the error bounds. The latter has to be executed only once and standard optimization techniques can be employed.

In summary our contributions are as follows:

- We study a powerful query language over uncertain data. This language is formally introduced in Section 3. We design a representation system extending the one of [3] in Section 2 such that arbitrarily composed queries are transformations from one instance to another. In our representation system we can represent uncertainty of the probability distribution over the possible worlds by considering probability intervals instead of exact probabilities.
- We present a framework for query processing in Section 4 in which queries are decomposed into an approximation plan and a relational algebra plan. The former allows to successively improve the accuracy and the error bounds. The latter has to be executed only once and standard optimization techniques can be employed. The latter allows to successively improve accuracy and error bounds, in order to provide overall result quality guarantees in the face of approximate intermediate results. While the relational algebra plan only has to be executed once.
- We show that in general confidence approximation over our new representation system with interval probabilities is hard, see Section 6. However, for evaluating queries over probabilistic databases we develop an efficient approximation algorithm that can handle the interval probabilities introduced through unreliable selections.
- We analyze the complexity of evaluating selection predicates involving probabilities in Section 6.1. We show that it is #P–hard for a simple type of predicate. We also show that for any non-trivial predicate, there is no randomized algorithm with a good error bound unless BPP⊃NP. We generalize the results in [14] to efficiently guess robust predicates with a good error bound in Sec. 6.2.

2. DATA MODEL

A basic model for uncertain data is a *probabilistic database*.

Definition 2.1. A probabilistic database with schema $\Sigma = (R_1[\vec{A_1}], \ldots, R_k[\vec{A_k}])$ is a *finite* set of possible worlds together with a probability distribution $\vec{p}$ over the worlds. Each world is associated with a relational database over the schema Σ. □

The intuition behind this definition is that we know that there is only one world describing the reality, but we do not know which one. Some worlds are more likely than others.

2.1 Representation System

In Example 1.1 we saw that a selection of tuples based on their approximate confidences results in a probabilistic database, in which the probability distribution is not known exactly. We introduce a representation system in which one can represent uncertainty of the probability distribution. We extend the system of U–DBs [3]. This representation system corresponds to a probabilistic version of conditional tables with variables that range over finite domains and with a global condition that is always true. A valuation of the variables corresponds to a possible world containing all tuples whose conditions evaluate to true.

A	B	D
a_1	b_1	$X_1 \wedge X_2$
a_1	b_1	$\neg X_3$
a_2	b_1	$\neg X_1 \wedge \neg X_3$
a_1	b_2	$\neg X_1 \wedge \neg X_3$
a_1	b_2	X_2
a_1	b_2	X_3

(a) U–relation

Vars	$\Pr_{\min}$	$\Pr_{\max}$
X_1	0.8	1
X_2	0	0.1
X_3	0.5	0.5

(b) RUPD

Figure 1: U–DB.

Definition 2.2. Given a set of binary random variables, a RUPD (short for Representation of an Unreliable Probability Distribution) describes for each variable X a lower bound $\Pr_{\min}[X = 1]$ and an upper bound $\Pr_{\max}[X = 1]$ on an unknown probability $\Pr[X = 1]$. A U–relation U_i is a table with schema $\Sigma = U_i[\vec{A}_i, D_i]$, where the attribute D_i contains a clause over the variables in the RUPD. An (unreliable) U–DB consists of a set of binary random variables, a RUPD, and U–relations $U_1, \ldots, U_k$. □

Fig. 1 shows an example of a U–DB. Note that a tuple can occur multiple times with different clauses. Conceptually, for each tuple over $\vec{A}_i$ we have a DNF which is the disjunction of all clauses in D_i associated with that tuple. While such a table in which tuples have DNF conditions is not strictly a U-relation according to the original definition in [**?**], we will sometimes refer to this generalized notion as U-relation too.

Remark 1. The RUPD does not determine exactly the probability of a variable X being true. It gives bounds on the probability of a variable. These bounds imply bounds for the probability of X being false.

$$\Pr[X = 0] \geq \Pr_{\min}[X = 0] = 1 - \Pr_{\max}[X = 1]$$
$$\Pr[X = 0] \leq \Pr_{\max}[X = 0] = 1 - \Pr_{\min}[X = 1]$$

Conventions. We call a U–relation certain if the conditions of all variables are always true. In this case we omit the column D_i. For brevity we will denote $\Pr[X = 1]$ by $\Pr[X]$ and $\Pr[X = 0]$ by $\Pr[\neg X]$. We refer to variables X where the exact probability is unknown, i.e. $\Pr_{\min}[X = 1] \neq \Pr_{\max}[X = 1]$, as *unreliable variables*.

We next explain the semantics of a U–DB. Each variable can have any probability within its bounds.

Definition 2.3. A RRPD W^* (short for Representation of a Reliable Probability Distribution) describes for each binary variable X a probability $\Pr_{W^*}[X]$.

We say a RRPD W^* is an instantiation of a RUPD if for all variables X: $\Pr_{\min}[X] \leq \Pr_{W^*}[X] \leq \Pr_{\max}[X]$. □

Conceptually, one of the instantiations of a given RUPD is the correct one, but it is unknown which one. Having defined the RRPD instantiations of the RUPD, we next show that each RRPD corresponds to a probabilistic database.

A total valuation θ from all variables to values in $\{0, 1\}$ determines a possible world. In each U–relation all tuples are contained in that world, whose clause evaluates to true under θ. Note that two different total valuations can describe the same databases. A DNF ψ describes a set of possible worlds. This set contains all possible worlds described by a valuation θ under which ψ evaluates to true. We denote this set by $\omega(\psi)$.

Every RRPD W^* induces a probability distribution over the possible worlds. The probability of a world described by the total valuation θ, denoted by $p_{W^*}(\theta)$, is

$$p_{W^*}(\theta) = \prod_X \Pr_{W^*}[X = \theta(X)].$$

This means that the variables are independent. As we will see later, despite this independence assumption we can represent every probabilistic database as a U–DB. The probability that a DNF ψ is true is

$$p_{W^*}(\psi) = \sum_{\theta \in \omega(\psi)} p_{W^*}(\theta).$$

We can also calculate the probability that a tuple t is contained in a world randomly drawn according to the probability distribution induced by a W^*. This probability is referred to as the confidence of the tuple, $p_{W^*}(t)$. Let ψ_t be the DNF of the tuple t which is the disjunction of all clauses associated with t. Now, $p_{W^*}(t) = p_{W^*}(\psi_t)$.

Example 2.1. Fig. 1(a) depicts a U–DB. Note that only variable X_3 is reliable.

Consider the following instantiation RRPD W^* of the RUPD: $\Pr_{W^*}[X_1] = 0.9$, $\Pr_{W^*}[X_2] = 0.1$, $\Pr_{W^*}[X_3] = 0.5$. Indeed, these probabilities fall into the bounds of the RUPD in Fig. 1(b). Consider the total valuation θ: $\theta(X_1) = 1$, $\theta(X_2) = 1$, $\theta(X_3) = 0$. The corresponding world contains tuples $\langle a_1, b_1 \rangle$ and $\langle a_1, b_2 \rangle$. The probability of θ is $p_{W^*}(\theta) = 0.9 \cdot 0.1 \cdot 0.5 = 0.045$. We can also calculate the confidence of the tuple $\langle a_1, b_2 \rangle$, which is the confidence of the DNF $(\neg X_1 \wedge \neg X_3) \vee X_2 \vee X_3$: $p_{W^*}(\langle a_1, b_2 \rangle) = 1 - p_{W^*}(X_1 \wedge \neg X_2 \wedge \neg X_3) = 1 - 0.405 = 0.595$.

Note that the U–relation is equivalent to a U–relation in which $\langle a_1, b_2 \rangle$ occurs only once with condition $(\neg X_1 \wedge \neg X_3) \vee X_2 \vee X_3$. □

Remark 2. When we analyze the efficiency of operations over U–DBs we do this with respect to the number of variables.

Our representation system of U–DBs has two desirable properties:

Proposition 2.1 (Expressiveness). *Every probabilistic database can be represented as a U–DB.*

This result was known for U–DBs containing variables of finite domains [3]. We show that binary variables are sufficient to represent every probabilistic database.

Proof. Given a probabilistic database with probability distribution $\vec{p}$ over n possible worlds. We construct a U–DB as follows. For world i and every tuple in that world we insert this tuple in the U–relation with condition $X_1 = 0 \wedge \cdots \wedge X_{i-1} = 0 \wedge X_i = 1$. In the RUPD we assign a variable X_i the probability $\Pr_{\min}[X_i = 1] = \Pr_{\max}[X_i = 1] = \frac{p^{(i)}}{1 - \sum_{j=1}^{i-1} p^{(j)}}$. Thus this RUPD has a unique RRPD instantiation W^*.

We claim that the set of possible worlds described by the U–DB is the same as in the probabilistic database. A total valuation θ for which X_i is the first variable that is set to one by θ (i.e., $\theta(X_i) = 1$ and $\forall j < i : \theta(X_j) = 0$) selects exactly those tuples of world i.

We need to make sure that the possible worlds have the same probabilities, i.e., that RRPD W^* induces a probability distribution over the possible worlds that is equal to $\vec{p}$.

$$
\begin{aligned}
\Pr[i^{\mathrm{th}}\mathrm{world}] &= \sum_{\substack{\theta:\theta(X_i)=1\wedge\\ \forall j<i:\theta(X_j)=0}} p_{W^*}(\theta)\\
&= \sum_{\substack{\theta:\theta(X_i)=1\wedge\\ \forall j<i:\theta(X_j)=0}} \prod_k \Pr_{W^*}[X_k=\theta(X_k)]\\
&= \Pr_{W^*}[X_i=1]\prod_{j<i}\Pr_{W^*}[X_j=0]\\
&= \frac{p^{(i)}}{1-\sum_{k=1}^{i-1}p^{(k)}}\prod_{j<i}\left(1-\frac{p^{(j)}}{1-\sum_{k=1}^{j-1}p^{(k)}}\right)\\
&= \frac{p^{(i)}}{1-\sum_{k=1}^{i-1}p^{(k)}}\prod_{j<i}\frac{1-\sum_{k=1}^{j}p^{(k)}}{1-\sum_{k=1}^{j-1}p^{(k)}} = p^{(i)}
\end{aligned}
$$

□

The transformation from a probabilistic database into a U–DB requires as many variables as there are possible worlds. Often there are more succinct ways to represent a probabilistic database as a U–DB.

Proposition 2.2 (Succinctness [3]). *U–DBs are succinct. The number of worlds represented by a U–DB can be exponential in the size of the U–DB.*

As we will see in the next section, our model is expressive enough to capture the unreliable result of a selection if the predicates are evaluated on approximate confidences. We can view queries as transformations from one U–DB to another.

3. QUERY LANGUAGE

3.1 Syntax

Our query language consists of the following operators: selection σ, projection π, product $\times$, union $\cup$, and confidence computation $\widehat{p}$. As selection predicates $\phi(t)$ we allow Boolean combinations of inequalities involving arithmetic expressions over confidences and constants. The inequalities compare an arithmetic expression with zero using the relations $\{=,\neq,>,<,\leq,\geq\}$. The arithmetic expressions are of the form $g(t, p(\psi_1(t)), \ldots, p(\psi_k(t)))$, where $\psi_i(t)$ is a DNF (such as ψ_t) and $g(\cdot)$ is an arithmetic expression constructed using the variables, constants, and the operations $+, -, \cdot, /$.

Queries are arbitrary compositions of the operators. They are executed on a U–DB and the result is another U–DB.

3.2 Semantics

We define the semantics of a compositional query on a U–DB inductively, by defining the semantics of each operation on an intermediate result of a query Q. Our U–DB consists of a RUPD and U–relations $U_1[\vec{A}_1, D_1], \ldots, U_k[\vec{A}_k, D_k]$. An operation gets as input one or two U–relations. The output is a new U–relation that will be added to the existing ones. All operations have access to the RUPD and can update it. Updates are restricted to the insertion of new variables and their probabilities.

$$
\begin{aligned}
[\![\pi_{\vec{B}}(Q)]\!] &= \{\langle t,c\rangle \mid \exists s\langle s,c\rangle \in [\![Q]\!] \wedge \pi_{\vec{B}}(s)=t\}\\
[\![\sigma_\phi(Q)]\!] &= \{\langle t,c\rangle \mid \langle t,c\rangle \in [\![Q]\!] \wedge \phi(t)=1\}\\
[\![Q_1\cup Q_2]\!] &= \{\langle t,c\rangle \mid \langle t,c\rangle \in [\![Q_1]\!] \vee \langle t,c\rangle \in [\![Q_2]\!]\}\\
[\![Q_1\times Q_2]\!] &= \{\langle s,t,(c_1\wedge c_2)\rangle \mid \langle s,c_1\rangle \in [\![Q_1]\!],\\
&\qquad\qquad \langle t,c_2\rangle \in [\![Q_2]\!], c_1\wedge c_2 \text{ satisfiable}\}
\end{aligned}
$$

Figure 2: Semantics of operations in positive relational algebra over an unreliable U–relation.

Positive Relational Algebra. In this paragraph we consider the operations σ, π, $\times$, $\cup$. We restrict ourselves to selection predicates that do not involve any confidences. The general selection will be discussed later.[2]

We define the semantics $[\![\cdot]\!]$ of these operations inductively as listed in Fig. 2. Here Q, Q_1, Q_2 denote U–relations, which are the results of queries executed on our U–DB. For example, the semantics of a selection $\sigma_\phi(Q)$ corresponds to the following intuition: Q is the result of a query over our U–DB. We compute all possible worlds of Q, execute the selection in every single world, and add the result as another table in that world. If we then represent this probabilistic database as a U–DB again, we obtain exactly the one defined through $[\![\sigma_\phi(Q)]\!]$. Based on the definition of the semantics it is straightforward to evaluate the operations efficiently by re–writing them as positive relational algebra expressions over the U–relations.

Remark. We do not consider the difference operation, because one would have to transform $\neg\psi$ back into a DNF which takes exponential time in the number of variables.

Confidence Approximation. The randomized operation $\widehat{p}(Q,\epsilon,\delta)$ estimates for each tuple t its confidence with accuracy ϵ and error bound δ according to the RUPD. For the case that all probabilities are certain, the operation $\widehat{p}(Q,\epsilon,\delta)$ is a FPRAS. We naturally extend the definition of a FPRAS to cases in which the probability distribution is not known exactly.

The operation $\widehat{p}(Q,\epsilon,\delta)$ estimates a lower bound $\widehat{p}_{\min}(t)$ and an upper bound $\widehat{p}_{\max}(t)$. We require that (1) the bounds are correct with high probability and (2) the bounds are almost tight.

(1) *Low Error.* For any probability distribution induced by an RRPD instantiation W^* of the RUPD the bounds are valid with high probability, i.e.

$$\Pr\left[p_{W^*} \in \left[\frac{\widehat{p}_{\min}}{1+\epsilon}, \frac{\widehat{p}_{\max}}{1-\epsilon}\right]\right] \geq 1-\delta. \qquad (1)$$

(2) *High Accuracy.* There are instantiations $W_{\min}$ and $W_{\max}$ of the RUPD such that the bounds are ϵ–tight with high probability, i.e.

$$\Pr\left[\frac{\widehat{p}_{\min}}{1-\epsilon} \geq p_{W_{\min}}\right] \geq 1-\delta \qquad (2)$$

$$\Pr\left[\frac{\widehat{p}_{\max}}{1+\epsilon} \leq p_{W_{\max}}\right] \geq 1-\delta \qquad (3)$$

The reader might want to object that requirement (1) is too strong because it is sufficient to obtain bounds on the con-

[2] Given a query we can determine whether a selection predicate involves confidences using typed attributes. An attribute has type "uncertain" if it is the result of a confidence computation and it has type "certain" otherwise.

fidence with respect to the true RRPD instantiation of the RUPD. However, this RRPD is unknown. That is why the bounds have to hold with respect to all RRPD instantiations of the RUPD.

The result is stored in a certain database of the schema $\Sigma = (R[Q.\vec{A}, \widehat{p}_{\min}, \widehat{p}_{\max}, \epsilon, \delta])$ that is added to the U–DB.

Next we discuss how to select tuples based on the confidences of events in our query language.

Selection Based on Confidence. The randomized operation $\hat{\sigma}_\phi(Q)$ guesses for each tuple t whether it satisfies the condition ϕ (e.g. whether its confidence is above 0.8). We denote by $\hat{\phi}(t) \in \{0, 1, \text{“Don't Know”}\}$ the guess of $\phi(t)$. The guarantee is that the guess is correct with probability at least $1 - \delta$ for all RRPDs W^* that are instantiations of the RUPD. The result of this operation is a tuple independent unreliable probabilistic database. Tuple independence means that the conditions of the tuples contain disjoint set of variables. This result can be represented as a U–relation with schema $\Sigma = (R[Q.\vec{A}, D])$. The condition of a tuple t is $Z_t = 1$, where Z_t is a new variable. Those variables are added to the RUPD with the following probabilities:

$$\Pr_{\min}[Z_t] = \begin{cases} 1 - \delta, & \text{if } \hat{\phi}(t) = 1, \text{ error bound} = \delta \\ 0, & \text{if } \hat{\phi}(t) = 0, \text{ error bound} = \delta \\ 0, & \text{if “Don't Know”} \end{cases}$$

$$\Pr_{\max}[Z_t] = \begin{cases} 1, & \text{if } \hat{\phi}(t) = 1, \text{ error bound} = \delta \\ \delta, & \text{if } \hat{\phi}(t) = 0, \text{ error bound} = \delta \\ 1, & \text{if “Don't Know”} \end{cases}$$

3.3 Conditioning with Constraints

In our motivating example in the introduction we claimed that in our framework the enforcement of equality generating dependencies (egds) can be done in an efficient way. We consider egds of the form $\forall t_1, t_2 : \xi_1 \rightarrow \xi_2$, where ξ_1 and ξ_2 are Boolean combinations of predicates comparing tuples and constants using (in–) equalities $(=, >, \leq, \neq, \dots)$. The conditional functional dependencies [4] that allow to restrict the validity of a dependency to a certain class of tuples fall into our class of constraints.

The egds are taken into consideration when approximating tuple confidences and evaluating the selection predicates based on them. The complexity of these operations remains the same when imposing egds on the data. The confidence of a tuple t under the constraint ξ for any RRPD W^* is

$$p_{W^*}(t|\xi) = \frac{p_{W^*}(\psi_t) - p_{W^*}(\psi_{t \wedge \neg\xi})}{1 - p_{W^*}(\psi_{\neg\xi})}, \tag{4}$$

where ψ_t is the disjunction of the clauses associated with tuple t in the U–relation that describes the worlds in which tuple t is contained, $\psi_{\neg\xi}$ is a DNF that describes the worlds in which the constraint ξ is violated, and $\psi_{t\wedge\neg\xi}$ is a DNF that describes the worlds in which tuple t is contained and the constraint ξ is violated.

We can estimate $p(t|\xi)$ by estimating the confidences of equation(4). We use the operations $\widehat{p}(\psi_t, \epsilon, \delta), \widehat{p}(\psi_{t\wedge\neg\xi}, \epsilon, \delta)$ and $\widehat{p}(\psi_{\neg\xi}, \epsilon, \delta)$ to obtain bounds that are ϵ–tight with probability at least $1 - \delta$.

We compute bounds on $p(t|\xi)$ as follows

$$\widehat{p}_{\min}(t|\xi) = \frac{\frac{\widehat{p}_{\min}(\psi_t)}{1+\epsilon} - \frac{\widehat{p}_{\max}(\psi_{t\wedge\neg\xi})}{1-\epsilon}}{1 - \frac{\widehat{p}_{\min}(\psi_{\neg\xi})}{1+\epsilon}} \tag{5}$$

$$\widehat{p}_{\max}(t|\xi) = \frac{\frac{\widehat{p}_{\max}(\psi_t)}{1-\epsilon} - \frac{\widehat{p}_{\min}(\psi_{t\wedge\neg\xi})}{1+\epsilon}}{1 - \frac{\widehat{p}_{\max}(\psi_{\neg\xi})}{1-\epsilon}} \tag{6}$$

The next proposition says that these bounds are correct with probability at least $1 - 3\delta$.

PROPOSITION 3.1. *Computing the confidence according to Equation (5) and (6) with approximate confidences yields the following guarantee*

$$\Pr\left[p(t|\xi) \in [\widehat{p}_{\min}(t|\xi), \widehat{p}_{\max}(t|\xi)]\right] \geq 1 - 3\delta.$$

The proof follows from the fact that if the bounds for $p(t|\xi)$ are wrong then at least one of the bounds for $p(\psi_t)$, $p(\psi_{t\wedge\neg\xi})$ or $p(\psi_{\neg\xi})$ is wrong. But such a failure occurs with at most probability δ.

Next we explain how to rewrite the confidence computation with constraints $p(U|\xi)$ as a query involving only positive relational algebra and confidence computation. Let us start by describing how to compute the DNFs $\psi_t, \psi_{\neg\xi}$ and $\psi_{t\wedge\neg\xi}$. For each tuple t its condition ψ_t is the disjunction of the clauses recorded in the column D for t. Let α describe the predicate that evaluates to true if the conjunction of the clauses of two tuples is satisfiable. Given a set of constraints $\xi = \{\dots, \mathrm{eq}_i, \dots\}$ with egds eq_i of the form $\forall t_1, t_2 : \xi_1 \rightarrow \xi_2$, where ξ_1 and ξ_2 are Boolean combinations of predicates comparing tuples and constants using (in–) equalities $(=, >, \leq, \neq, \dots)$. We can compute $\psi_{\neg\xi}$ by computing the cross product between U and U and selecting tuples violating one of the constraints in ξ. A projection to an empty set of attributes yields $\psi_{\neg\xi}$, i.e.

$$\psi_{\neg\xi} = \bigcup_i \pi_\emptyset\left(U \bowtie_{\mathrm{eq}_i \text{ violated} \wedge \alpha} U\right).$$

We can compute $\psi_{t\wedge\neg\xi}$ for each $t \in U$ by computing the conjunction of the ψ_t and $\psi_{\neg\xi}$. Using α as a join predicate between U and $\psi_{\neg\xi}$, we filter out those tuples that are never in conflict with the constraints ξ.

Putting these computations together we can issue a single query (involving renaming operations explained below) computing bounds on the conditional confidence:

$$\begin{aligned} &\widehat{p}(U|\xi) \\ &= \pi_{\phi_4}\left(\rho_{\phi_1}\left(\widehat{p}(U, \epsilon, \delta)\right)\right. \\ &\bowtie_\alpha \rho_{\phi_2}\left(\widehat{p}\left(\bigcup_i \pi_\emptyset\left(U \bowtie_{\mathrm{eq}_i \text{ violated}\wedge\alpha} U\right)\epsilon, \delta\right)\right) \\ &\bowtie_\alpha \rho_{\phi_3}\left(\widehat{p}\left(U \bowtie_\alpha \bigcup_i \pi_\emptyset\left(U \bowtie_{\mathrm{eq}_i \text{ violated}\wedge\alpha} U\right), \epsilon, \delta\right)\right)\left.\right) \end{aligned}$$

We define renaming operations as follows:

$$\begin{array}{lll}
\phi_1: & \widehat{p}_{\min} & \rightarrow \widehat{p}_{\min}(\psi_t), \\
 & \widehat{p}_{\max} & \rightarrow \widehat{p}_{\max}(\psi_t), \\
 & \delta & \rightarrow \delta_t, \\
\phi_2: & \widehat{p}_{\min} & \rightarrow \widehat{p}_{\min}(\psi_{\neg\xi}), \\
 & \widehat{p}_{\max} & \rightarrow \widehat{p}_{\max}(\psi_{\neg\xi}), \\
 & \delta & \rightarrow \delta_{\neg\xi}, \\
\phi_3: & \widehat{p}_{\min} & \rightarrow \widehat{p}_{\min}(\psi_{\neg\xi\wedge t}), \\
 & \widehat{p}_{\max} & \rightarrow \widehat{p}_{\min}(\psi_{\neg\xi\wedge t}), \\
 & \delta & \rightarrow \delta_{\neg\xi\wedge t}, \\
\phi_4: & \vec{A}, \frac{\frac{\widehat{p}_{\min}(\psi_t)}{1+\epsilon} - \frac{\widehat{p}_{\max}(\psi_{t\wedge\neg\xi})}{1-\epsilon}}{1-\frac{\widehat{p}_{\min}(\psi_{\neg\xi})}{1+\epsilon}} & \rightarrow \widehat{p}_{\min}, \\
 & \frac{\frac{\widehat{p}_{\max}(\psi_t)}{1-\epsilon} - \frac{\widehat{p}_{\min}(\psi_{t\wedge\neg\xi})}{1+\epsilon}}{1-\frac{\widehat{p}_{\max}(\psi_{\neg\xi})}{1-\epsilon}} & \rightarrow \widehat{p}_{\max}, \\
 & \delta_{\neg\xi} + \delta_t + \delta_{t\wedge\neg\xi} & \rightarrow \delta.
\end{array}$$

An arrow name$_1$ → name$_2$ means that name$_1$ is renamed to name$_2$.

Sec. 5 and Sec. 6 will be concerned with efficient evaluation of the randomized operations. Before that we outline our compositional framework for query evaluation under the assumption that we can efficiently evaluate all operations.

4. COMPOSITIONAL FRAMEWORK FOR QUERY EVALUATION

In our framework for approximate query evaluation we can separate the handling of the data from the approximation of the confidences. The data handling is done as follows: given a query that involves confidence computations, these are not carried out immediately, instead the values for the attributes $\widehat{p}_{\min}, \widehat{p}_{\max}, \epsilon, \delta$ are left empty for each tuple. For a selection $\hat{\sigma}$ we select all tuples, create a new variable X_t for each tuple t and put $X_t = 1$ as condition in the U–relation. We add all these variables to the RUPD and leave their probability bounds empty. The operations in relational algebra can be carried out as they are not affected by the approximation of confidences.

A provenance tree of the approximations is created as follows: We successively contract edges in the query plan between a relational algebra operation and an approximation operation, i.e. we merge the nodes to one node whose label is the one of the approximation operation and we keep all adjacent edges. With this provenance tree we can carry out the approximations in a bottom up way and fill in the results at the positions left open by execution of the relational algebra plan.

Example 4.1. A search engine wants to make all books searchable. Towards that goal it uses optical character recognition (OCR) techniques to produce candidate words for a given piece of scanned text. The result is a table OCR = (Image, Word, D_1). Furthermore, human feedback is provided and represented in a certain table FEEDBACK = (ID, Image, Word). Another table captures the trustworthiness of the humans. Error probabilities have been recorded based on some test. Those are represented in the table TRUST = (ID, D_2).

The search engine combines these sources to compute probability distributions over words for each image as depicted in Fig. 3. First, the set of candidates for each image in the OCR are restricted by selecting only those that have a probability above 0.2. The result is joined with the FEEDBACK and with the TRUST. The result is a table (Image, Word, ID, D_2) where D_2 represents the error probability. On this table the integrity constraint ξ is enforced which assures that Image is a key. Thus all possible worlds that contain two different words for an image are assigned a probability of zero. The probabilities of the remaining worlds are normalized. We can compute the probability for each tuple to obtain new probability distributions over the words for each image.

Fig. 3 shows that given a query we can extract a relational algebra query plan and an approximation provenance tree. We can resort to standard optimization techniques for the relational algebra query plan and we can successively improve the accuracy and the error probability of the randomized operations.

Improving the Error Bound. Suppose the approximations have been carried out in a bottom up fashion according to the provenance tree. Imagine the quality of the result is unsatisfactory, for example the accuracy of a confidence computation is too low or the error bound δ is not small enough. In order to improve the quality of the result we can either improve the quality of the last approximation that yielded the result, or we can go further down in the provenance tree to improve the quality of descendant approximations which improve the approximation of the root in the provenance tree (which created the result). For example, an unsatisfactory quality of the confidence approximation in Fig. 3 might be caused by a large error bound of the child in the provenance tree. In general, if we decide to improve some descendant approximation then all ancestors of this node in the provenance tree have to be re-computed from scratch, because the RUPD has been updated. However, we do not need to evaluate the relation algebra query plan again.

5. APPROXIMATING THE CONFIDENCE

We propose algorithms for approximating the confidence of a DNF formula ψ given a RUPD. Our goal is given ϵ and δ to compute bounds $\widehat{p}_{\min}(\psi), \widehat{p}_{\max}(\psi)$ such that with probability at least $1 - \delta$ the bounds are correct for any RRPD instantiation W^* of the RUPD, see Equation(1). The confidence $p_{W^*}(\psi)$ is the probability that the DNF evaluates to true under a random assignment of variables to values in $\{0, 1\}$ according to W^*.

Ideally, these bounds are close to the optimal bounds, i.e. we would like that with high probability the lower bound is only a $(1 - \epsilon)$ factor away from the optimal lower bound and the upper bound is only a $(1 + \epsilon)$ factor away from the optimal upper bound. The optimal upper bound is the maximum of $p(\psi)$ taken over all RRPD instantiations of the RUPD. The optimal lower bound is the minimum of $p(\psi)$ taken over all RRPD instantiations of the RUPD. See Equations (2) and (3).

Notational Conventions. Given a RUPD, that records for each variable X a lower bound $\Pr_{\min}[X]$ and an upper bound $\Pr_{\max}[X]$ on the probability of X being true. We refer to a RRPD that is an instantiation of the RUPD as

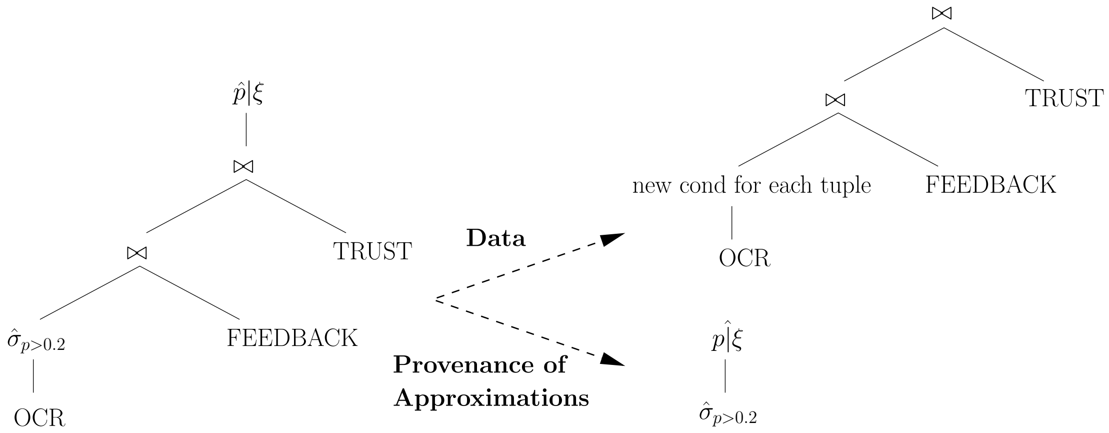

Figure 3: Extracting a relational algebra query plan and an approximation provenance tree from a query.

W^* – it records for each variable X a value $\Pr_{W^*}[X]$ in $[\Pr_{\min}[X], \Pr_{\max}[X]]$. We denote by $p_{W^*}(\psi)$ the probability of ψ given the RRPD W^*. We refer to a RRPD under which the probability of each variable X is either $\Pr_{\min}[X]$ or $\Pr_{\max}[X]$ as W'. We denote by $p_{W'}(\psi)$ the probability of ψ given the RRPD W'.

In Sec 5.1 we briefly review the Karp–Luby algorithm that approximates confidences for RRPDs. We show how to invoke the Karp–Luby algorithm an exponential number of times to obtain bounds that are correct and tight with high probability, see Sec. 5.2. We show that there only knowing interval probabilities instead of exact probabilities make this approximation problem NP–hard, see Sec. 5.2.2. However, we design efficient approximation algorithms for certain classes of DNFs and RUPDs in Sec. 5.2.3. Most notably, if the uncertainty of the probabilities is introduced through selections then the confidence computation remains efficient. Furthermore, we show how to efficiently compute (not so tight) bounds for the general case in Sec. 5.2.4.

5.1 Reliable U–DBs

This section reviews how to approximate the confidence of a DNF ψ if all variables involved are reliable (i.e. $\Pr_{\min}[X] = \Pr_{\max}[X], \forall X$). The exact probabilities of the variables are recorded in a RRPD W^*. The techniques presented in this section are standard and have already been used in a similar form in [6, 14]. Since our solutions for the general case of unreliable variables build on these techniques, we will briefly discuss them.

First we define an estimator in Alg. 1 that in expectation calculates the confidence of a DNF divided by the sum of probabilities of the clauses P. This estimator is based on the one in [13] that computes the number of solutions of a DNF formula. In essence the solutions are weighted according to their probability.

Proposition 5.1. *In expectation* Estimator(ψ, W^*) *outputs* $\frac{p_{W^*}(\psi)}{P}$, *where* $P = \sum_{clause\ c \in \psi} p_{W^*}(c)$.

Algorithm 1: Estimator (DNF ψ, RRPD W^*)

```
Fix an order of the clauses in $\psi$.
Let $P = \sum_{\text{clause } c \in \psi} p_{W^*}(c)$.
Choose $c$ from $\psi$ with probability $p_{W^*}(c)/P$.
Choose a total valuation $\theta \in \omega(c)$ with probability
  $p_{W^*}(\theta)/p_{W^*}(c)$. That is for each variable $Y$ whose
  truth value is not determined by $c$ sample a value
  according to $W^*$.
if $c$ is the clause in $\psi$ with the smallest order that
  evaluates to true under $\theta$ then return 1
else return 0
```

Proof. Given a DNF ψ. Let X denote the random variable of the output of the Estimator. Let $X_{c,\theta}$ be the indicator variable that is 1 if and only if c is the first clause satisfied by θ. We have:

$$\begin{aligned}
E[X] &= \sum_{c \in \psi} \frac{p_{W^*}(c)}{P} \sum_{\theta \in \omega(c)} \frac{p_{W^*}(\theta)}{p_{W^*}(c)} X_{c,\theta} \\
&= \sum_{c \in \psi} \sum_{\theta \in \omega(c)} \frac{p_{W^*}(\theta)}{P} X_{c,\theta} \\
&= \sum_{\theta} \frac{p_{W^*}(\theta)}{P} \sum_{c:\theta \in \omega(c)} X_{c,\theta} \\
&= \sum_{\theta:\exists c \in \psi:\theta \in \omega(c)} \frac{p_{W^*}(\theta)}{P} \\
&= \frac{p_{W^*}(\psi)}{P}
\end{aligned}$$

□

Algorithm 1 gives rise to a fully-polynomial-time randomized approximation scheme (FPRAS). All we need to do is to average the results of multiple samples from the Estimator multiplied by P, see Algorithm 2. The number of samples is polynomial in all parameters.

Proposition 5.2. Karp-Luby$(\psi, \epsilon, \delta, W^*)$ *outputs an estimate* $\hat{p}_{W^*}(\psi)$ *of* $p_{W^*}(\psi)$. *The guarantee is that*

$$\Pr\left[\frac{\hat{p}_{W^*}(\psi)}{1+\epsilon} \leq p_{W^*}(\psi) \leq \frac{\hat{p}_{W^*}(\psi)}{1-\epsilon}\right] \geq 1 - \delta.$$

Algorithm 2: Karp-Luby(DNF ψ, ϵ, δ, RRPD W^*)

```
1 Let $M = \left\lceil \frac{3|\psi| \ln \frac{2}{\delta}}{\epsilon^2} \right\rceil$.
2 Let $S = 0$.
3 Let $P = \sum_{\text{clause } c \in \psi} p_{W^*}(c)$.
4 for $1 \leq i \leq M$ do
5     $X_i \leftarrow$ Estimator $(\psi, W^*)$
6     $S \leftarrow S + X_i$
7 return $P \cdot S/M$
```

The algorithm only needs $\left\lceil \frac{3|\psi| \ln \frac{2}{\delta}}{\epsilon^2} \right\rceil$ *samples from the* Estimator*, where* $|\psi|$ *is the number of clauses in the DNF* ψ*.*

Proof. Let X_i be a random variable denoting the result of the Estimator in round i of the for loop 5. Hence, let S denote the random variable summing up all X_is: $S = \sum_{i=1}^{M} X_i$. By linearity of expectation and Prop. 5.2 we have that $E[S] = M \cdot p(\psi)/P$. The Chernoff bound guarantees that

$$\Pr[|S - E[S]| \geq \epsilon E[S]] \leq 2e^{-\epsilon^2 E[S]/3}.$$

For us this means

$$\Pr\left[\left| M\frac{\hat{p}_{W^*}(\psi)}{P} - M\frac{p_{W^*}(\psi)}{P} \right| \geq \epsilon \frac{Mp_{W^*}(\psi)}{P}\right] \leq 2e^{-\frac{\epsilon^2 Mp_{W^*}(\psi)}{3P}}$$

which is equivalent to

$$\Pr[|\hat{p}_{W^*}(\psi) - p(\psi)| \geq \epsilon \cdot p_{W^*}(\psi)] \leq 2e^{-\frac{\epsilon^2 Mp_{W^*}(\psi)}{3P}}.$$

We can re-write the upper bound using the fact that for all $c \in \psi : p_{W^*}(c) \leq p_{W^*}(\psi)$, which implies that $P = \sum_{c \in \psi} p_{W^*}(c) \leq |\psi| p_{W^*}(\psi)$. Hence, $p_{W^*}(\psi)/P \geq 1/|\psi|$.

$$\Pr[|\hat{p}_{W^*}(\psi) - p_{W^*}(\psi)| \geq \epsilon p_{W^*}(\psi)] \leq 2e^{-\frac{\epsilon^2 M}{3|\psi|}}$$

Since $M = \left\lceil \frac{3|\psi| \ln \frac{2}{\delta}}{\epsilon^2} \right\rceil$ the claim follows. □

This proposition states that Karp-Luby outputs correct bounds with high probability, i.e. Equation (1) holds. It also follows from this proposition that $\widehat{p}_{W^*}(\psi)$ is ϵ–tight with high probability.

5.2 Unreliable U–DBs

Above we have seen an algorithm for approximating the tuple confidence if we know the exact probabilities of all variables. However in our general framework we might only have bounds on the probabilities recorded in the RUPD and hence we cannot apply this algorithm.

Our goal is to compute bounds $\widehat{p}_{\min}(\psi), \widehat{p}_{\max}(\psi)$ such that for all instantiations RRPD W^* of the RUPD the bounds are correct with high probability, i.e. Equation (1) holds. The bounds should be close to optimal, i.e., there are instantiations $W_{\min}$ and $W_{\max}$ of the RUPD such that the bounds are ϵ–tight with high probability, i.e. Equations (2) and (3) hold.

5.2.1 An Accurate Approximation Algorithm

Given a RUPD and a DNF ψ whose probability we want to estimate, Algorithm 3 proceeds as follows: For each RRPD W' in which each variable X has a probability of either $\Pr_{\min}[X]$ or $\Pr_{\max}[X]$, it uses the Karp-Luby algorithm to obtain an estimate of the confidence. It outputs the maximum and the minimum bound. This yields an algorithm with a running time exponential in the number of variables.

Algorithm 3: Conf (DNF ψ,RUPD W, ϵ, δ)

```
Let min–est = 1, max–est = 0.
Let $S = 0$.
for any RRPD instantiation $W'$ of the RUPD s.t. $\Pr_{W'}[X] \in \{\Pr_{\min}[X], \Pr_{\max}[X]\}$ do
    Let $\hat{p}$ =Karp-Luby $(\psi, W', \epsilon, \delta)$.
    min–est $\leftarrow$ min($\frac{\hat{p}}{1+\epsilon}$,min–est)
    max–est $\leftarrow$ max($\frac{\hat{p}}{1-\epsilon}$,max–est)
return min–est, max–est, error bound $\delta$
```

The next proposition states that Algorithm 3 is correct, i.e. that it is in fact sufficient to focus only on RRPDs in which the probabilities of the variables take on either the maximum value or the minimum value (but nothing in between). In particular there is a RRPD $W'_{\max}$ in which each variable X has a probability of either $\Pr_{\min}[X]$ or $\Pr_{\max}[X]$ that maximizes the confidence of ψ and there is a RRPD $W'_{\min}$ in which each variable X has a probability of either $\Pr_{\min}[X]$ or $\Pr_{\max}[X]$ that minimizes the confidence of ψ among all instantiations of the RUPD. Hence, if we want to compute bounds on the confidence it is not necessary to check all RRPD instantiations W^* of the RUPD.

Proposition 5.3. *For any RRPD instantiation* W^* *of the RUPD the probability of a DNF* ψ *is bounded by*

$$\min_{W'} p_{W'}(\psi) \leq p_{W^*}(\psi) \leq \max_{W'} p_{W'}(\psi),$$

where the minimum and the maximum are taken over all RRPDs W' *such that* $\Pr_{W'}[X] \in \{\Pr_{\min}[X], \Pr_{\max}[X]\}$*.*

Proof. Assume for contradiction that there is a DNF ψ, a RUPD, and an instantiation RRPD W^* of the RUPD such that the probability of ψ is greater than the probability of ψ under any RRPD W'. (The case where the probability of ψ is smaller than the probability of ψ under any W' is analogous.) We use a hybrid argument to show that starting from the RRPD W^* with each step we can change the probability of one variable X to $\Pr_{\min}[X]$ or $\Pr_{\max}[X]$, such that we only increase the probability of the DNF ψ. We end up with an RRPD W' in which each variable X has a probability of either $\Pr_{\min}[X]$ or $\Pr_{\max}[X]$, s.t. $p_{W'}(\psi) \geq p_{W^*}(\psi)$ contradicting our assumption.

We start with hybrid $H_0 = W^*$. The i^{th} hybrid replaces the probabilities of the first i variables with either $\Pr_{\min}$ or $\Pr_{\max}$. Hence the n^{th} hybrid is the desired W'. We prove that there is a way to create H_{i+1}, such that $p_{H_{i+1}}(\psi) \geq p_{H_i}(\psi)$.

Let X be the i^{th} variable. We can rewrite $\psi = X \wedge \psi_1 \vee \neg X \wedge \psi_2$, such that X is neither contained in ψ_1 nor in ψ_2. Hence, $p_{H_i}(\psi) = \Pr_{W^*}[X] p_{H_i}(\psi_1) + (1 - \Pr_{W^*}[X]) p_{H_i}(\psi_2)$. If $p_{H_i}(\psi_1) > p_{H_i}(\psi_2)$, then replacing $\Pr_{W^*}[X]$ by $\Pr_{\max}[X]$ can only increase the probability of ψ. Similarly, if $p_{H_i}(\psi_1) \leq p_{H_i}(\psi_2)$, then replacing $\Pr_{W^*}[X]$ by $\Pr_{\min}[X]$ can only increase the probability of ψ. In both cases we obtain H_{i+1} such that $p_{H_{i+1}}(\psi) \geq p_{H_i}(\psi)$. □

Together with Proposition 5.2 it follows that Algorithm 3 returns correct bounds with probability at least $1-\delta$. Furthermore, the bounds are ϵ–tight with high probability.

5.2.2 Hardness

We would like to further reduce the search space and find a RRPD $W'_{\max}$ that maximizes the probability of ψ and we would like to find a RRPD $W'_{\min}$ that minimizes the probability of ψ efficiently. Then we would only have to run KARP-LUBY twice to obtain efficient approximations of the optimal bounds.

Unfortunately, this is not possible. Even deciding whether there is a RRPD W' such that $p_{W'}(\psi)$ is below some threshold τ is NP–hard.

PROPOSITION 5.4. *Given a RUPD, a DNF ψ, and a threshold τ. Deciding whether there is a RRPD W' in which each variable X has a probability of either* $\Pr_{\min}[X]$ *or* $\Pr_{\max}[X]$ *such that $p_{W'}(\psi) < \tau$ is NP–hard.*

PROOF. We reduce SAT to our problem. Let χ be a CNF for which we want to determine whether there is a satisfying assignment.

We construct a RUPD over all variables X in χ with probabilities of $\Pr_{\min}[X] = 0$ and $\Pr_{\max}[X] = 1$. Let $\psi = \neg\chi$.

We claim that there is an assignment that satisfies χ if and only if there is a RRPD instantiation W' of the RUPD recording for each variable X a probability of either $\Pr_{\min}[X]$ or $\Pr_{\max}[X]$ such that $p_{W'}(\psi) < \tau$.

We show both directions. If χ is satisfiable then there is a total valuation θ' : Vars $\rightarrow \{0,1\}$ such that χ evaluates to true. Consider the RRPD W': $\Pr_{W'}(X) = \theta'(X)$. Indeed, W' is an instantiation of the RUPD and each variable X has a probability of either $\Pr_{\min}[X]$ or $\Pr_{\max}[X]$. Only one total valuation from variables to $\{0,1\}$ has non-zero probability with respect to W'. This is θ' which has a probability of 1 and ψ evaluates to false under θ'. Hence,

$$\begin{aligned} p_{W'}(\psi) &= \sum_{\substack{\theta:\ \text{Vars} \rightarrow \{0,1\} \\ \psi \text{ evaluates to true under } \theta}} p_{W'}(\theta) \\ &= p_{W'}(\theta') = 0 < \tau. \end{aligned}$$

If χ is not satisfiable then for all assignments θ : Vars $\rightarrow \{0,1\}$ the CNF χ evaluates to false. Hence, under all total valuations θ the DNF ψ evaluates to true. For any RRPD W', in which each variable has a probability of either $\Pr_{\min}$ or $\Pr_{\max}$ exactly one total valuation θ has non-zero probability. This is the valuation that assigns the truth value 1 to a variable if and only if the probability under W' that the variable takes on value 1 is 1. This total valuation has probability 1 and under this total valuation ψ evaluates to true. Hence, for any RRPD W'

$$p_{W'}(\psi) = \sum_{\substack{\theta:\ \text{Vars} \rightarrow \{0,1\} \\ \psi \text{ evaluates to true under } \theta}} p_{W'}(\theta) = 1 \geq \tau$$

□

The hardness of deciding whether there is a RRPD W' such that $p_{W'}(\psi) < \tau$ is not caused by the difficulty of computing $p_{W'}(\psi)$. In our reduction these confidences are efficiently computable. Instead the hardness is caused by the number of possible instantiations W'. This shows that relaxing the model of probabilistic databases to allow for uncertainty of the probability distribution over the possible worlds makes the problem of approximating confidences harder.

The hardness of reducing of the search space is a worst case result. Next, we describe how we exploit structural properties of a DNF ψ and easy RUPDs in order to reduce the search space.

5.2.3 Improvements

Given a RUPD and a DNF ψ, we are looking for an RRPD $W'_{\max}$ in which each variable X has a probability of either $\Pr_{\min}[X]$ or $\Pr_{\max}[X]$ that maximizes the probability of ψ. (The minimization problem is analogous.)

For n variables there are 2^n possible RRPDs W' recording for each variable a probability of either $\Pr_{\min}[X]$ or $\Pr_{\max}[X]$. We seek to reduce this search space. More precisely, we want to find variables X for which we can efficiently determine whether $\Pr_{W'_{\max}}[X] = \Pr_{\min}[X]$ or whether $\Pr_{W'_{\max}}[X] = \Pr_{\max}[X]$.

We describe situations in which we can efficiently determine $\Pr_{W'_{\max}}[X]$. In these situations we use the following rewriting $\psi = X \wedge \psi_1 \vee \neg X \wedge \psi_2$, such that X is neither contained in ψ_1 nor in ψ_2.

1. If variable X only occurs positively in the DNF ψ, then $\Pr_{W'_{\max}}[X] = \Pr_{\max}[X]$.

   Here is an argument why this is correct: Since variable X occurs only as $X = 1$ in the clauses, all clauses in ψ_2 are also contained in ψ_1. Hence, $p(\psi) = \Pr[X]p(\psi_1) + (1 - \Pr[X])p(\psi_2)$ increases as $\Pr[X]$ increases. Therefore, $p(\psi)$ is maximized if $\Pr_{W'_{\max}} = \Pr[X]$.

2. We can analyze the $p_{W'}(\psi_1)$ and $p_{W'}(\psi_2)$ in a *worst case fashion*. If according to any RRPD instantiation W^* of the RUPD, $p_{W^*}(\psi_1) > p_{W^*}(\psi_2)$, then we can conclude that $\Pr_{W'_{\max}}[X] = \Pr_{\max}[X]$. Similarly, if according to any RRPD instantiation W^* of the RUPD, $p_{W^*}(\psi_1) < p_{W^*}(\psi_2)$, then we can conclude that $\Pr_{W'_{\max}}[X] = \Pr_{\min}[X]$.

   From the following inequalities that hold for all RRPDs W' and for all ψ we can derive sufficient conditions for the cases above that we can check efficiently:

$$p_{W'}(\psi_1) \leq \sum_{\text{clause } c \in \psi_1} \prod_{Y \in c} \Pr_{\max}[Y] \prod_{\neg Y \in c} (1 - \Pr_{\min}[Y])$$

$$p_{W'}(\psi_1) \geq \max_{\text{clause } c \in \psi_1} \left(\prod_{Y \in c} \Pr_{\min}[Y] \prod_{\neg Y \in c} (1 - \Pr_{\max}[Y]) \right)$$

3. We can determine $\Pr_{W'_{\max}}[X]$ *iteratively*. If we were able to determine $\Pr_{W'_{\max}}[Y]$ for all variables Y in ψ_1 and ψ_2, then we can use the (ϵ,δ)–approximation scheme KARP-LUBY to estimate $p_{W'_{\max}}(\psi_1)$ and also $p_{W'_{\max}}(\psi_2)$. If

$$\frac{\widehat{p}_{W'_{\max}}(\psi_1)}{1+\epsilon} > \frac{\widehat{p}_{W'_{\max}}(\psi_2)}{1-\epsilon},$$

   then we can conclude that with probability $\geq 1-\delta$ $\Pr_{W'_{\max}}[X] = \Pr_{\max}[X]$. Similarly, if

$$\frac{\widehat{p}_{W'_{\max}}(\psi_1)}{1-\epsilon} < \frac{\widehat{p}_{W'_{\max}}(\psi_2)}{1+\epsilon},$$

   then we can conclude that $\Pr_{W'_{\max}}[X] = \Pr_{\min}[X]$ with probability $\geq 1-\delta$.

```
Algorithm 4: CONF (DNF $\psi$, RUPD , $\epsilon, \delta$)
Let $W^*$: $\Pr_{W^*}[X] = \Pr_{\min}[X] + \frac{\Pr_{\max}[X]-\Pr_{\min}[X]}{2}$
Let $\hat{p} =$ KARP-LUBY $(\psi, W^*, \epsilon, \delta)$
return
    lower bound $\frac{\hat{p}}{1+\epsilon} - \sum_X \frac{\Pr_{\max}[X]-\Pr_{\min}[X]}{2}$,
    upper bound $\frac{\hat{p}}{1-\epsilon} + \sum_X \frac{\Pr_{\max}[X]-\Pr_{\min}[X]}{2}$,
    and error bound $\delta$
```

Whether we are able to determine $\Pr_{W'_{\max}}[X]$ for a given variable X depends on the structure of the DNF that influences how far the iterative reasoning works and the probabilities recorded by the RUPD that can make a worst case analysis possible.

We suggest to determine $\Pr_{W'_{\max}}[X]$ when possible and then apply Algorithm 3 discussed in the previous section. If we were able to determine $\Pr_{W'_{\max}}[X]$ for k out of n variables, then the complexity of the algorithm goes down from 2^n to 2^{n-k}. However, whenever the iterative reasoning was applied to determine the value $\Pr_{W'_{\max}}[X]$ for a variable, then the overall error probability increases by at most δ.

Consider the special case where the unreliability is induced by selection predicates. That is, we consider queries $\widehat{p}(Q)$, where the input of Q is a reliable U–DB, in which $\Pr_{\min}[X] = \Pr_{\max}[X]$. Recall, that we refer to variables X with $\Pr_{\min}[X] \neq \Pr_{\max}[X]$ as unreliable variables.

OBSERVATION 5.1. *Given a U–DB with reliable variables only. Let Q be an arbitrary query in our query language. Then in the result of Q all unreliable variables X occur only positively in the DNFs of the tuples.*

A select operation involving confidences introduces uncertain variables, but only assigns them the value 1 in the clauses of the tuples. All other operations do not change the assignments.

Using our improvement 1. for all these variable $\Pr_{W'_{\max}}[X] = \Pr_{\max}[X]$ (and similarly, $\Pr_{W'_{\min}}[X] = \Pr_{\min}[X]$). Hence, in Algorithm 3 we only need to estimate the confidence of the DNF under $W'_{\max}$ and $W'_{\min}$ in order to obtain valid and almost tight bounds of the confidence under the RUPD.

REMARK 5.1. *Given a U–DB with reliable variables only. Let Q be an arbitrary query in our query language. Then Algorithm 3 only needs to invoke the Karp–Luby algorithm twice. Once for the instantiation* $W'_{\max} : \Pr_{W'_{\max}}[X] = \Pr_{\max}[X]$ *and once for the instantiation* $W'_{\min} : \Pr_{W'_{\min}}[X] = \Pr_{\min}[X]$. *This yields an efficient approximation algorithm that outputs correct and tight bounds with high probability.*

5.2.4 An Efficient Randomized Algorithm

The approach above tries to calculate bounds that are close to the optimal bounds on the confidence. In order to obtain an efficient algorithm for confidence computation given any DNF and any RUPD we relax this requirement.

We propose Algorithm 4, which chooses for each variable X a probability at the center of the interval

$$[\Pr_{\min}[X], \Pr_{\max}[X]],$$

runs the KARP-LUBY algorithm, and then adjusts the returned bounds.

PROPOSITION 5.1. *Algorithm 4 is correct, that is, with probability* $\geq 1-\delta$ *for all instantiations RRPD* W^* *of the RUPD,* $p_{W^*}(\psi)$ *is within the returned bounds. Its running time is polynomial in all parameters. However, the bounds are not tight.*

6. A RANDOMIZED ALGORITHM FOR EVALUATING PREDICATES

Given an approximate selection query $\widehat{\sigma}_\phi(U)$. The predicate $\phi(t)$ is a Boolean combination of inequalities involving arithmetic expressions over the confidences of DNFs such as $p(\psi_1)/p(\psi_2) \geq 2 \wedge p(\psi_3) < 0.3$.

Before we present our algorithm we carry out an analysis of the hardness of this problem. The hardness holds even if the probabilities of all variables are known exactly.

6.1 Hardness of Evaluating Predicates

Let us start by analyzing the hardness of simple predicates of the form $p(\psi)$ OP c, where OP $\in \{>, \geq, <, \leq\}$ and $0 < c < 1$.

We show that evaluating such a predicate is #P–hard.

THEOREM 6.1. *Evaluating any predicate of the form*

$$p(\psi) \text{ OP } c$$

for OP $\in \{>, \geq, <, \leq\}$ *is #P–hard.*

PROOF. In order to show that a problem A is #P–hard, we need to find an efficient Turing reduction from another #P–hard problem B [19]. In this reduction we are given an instantiation of B and we are granted access to an oracle that solves A. If we manage to solve the problem for the instantiation efficiently, then problem A is also #P–hard.

Our reduction is from counting the number of satisfying assignments of a DNF formula which is #P–complete [19].

Given a DNF ψ whose number of satisfying assignments we want to compute. Let n be the number of variables in ψ. We execute binary search on $[0, 2^n]$, in order to find the number of satisfying assignments to ψ. If we can efficiently decide whether the number of satisfying assignments is OP a certain threshold then we can find the number of satisfying assignments efficiently because the binary search needs at most $\mathcal{O}(n)$ iterations.

We can use our oracle for $p(\psi)$ OP c in order to decide whether the number of satisfying assignments is OP m as follows. We consider the case OP is $\geq$. The other cases are similar. We construct a RRPD W^* over all n variables in ψ with a probability of 1/2. Here the confidence of ψ is equal to the number of satisfying assignments divided by 2^n. Furthermore, RRPD W^* records a probability for a new variable X.

We consider two cases. If $c < \frac{m}{2^n}$, then the new variable X has a probability of $\frac{c \cdot 2^n}{m}$. We have that the number of satisfying assignments of ψ is $\geq m$ if and only if $p(\psi) \geq \frac{m}{2^n}$, which is equivalent to $p(\psi \wedge X) \geq \frac{m}{2^n} \cdot \Pr[X] = c$.

If $c \geq \frac{m}{2^n}$, then the new variable X has a probability of $\frac{c - \frac{m}{2^n}}{1 - \frac{m}{2^n}}$. We have that the number of satisfying assignments of ψ is $\geq m$ if and only if $p(\psi) \geq \frac{m}{2^n}$, which is equivalent to $p(\psi \vee X) \geq \frac{m}{2^n}(1 - \Pr[X]) + \Pr[X] = c$.

This completes the reduction. □

Problems in #P are in the strict sense counting problems. A more relaxed notion of #P equivalence [12] also includes other functional problems. One can show that that the confidence computation is #P–equivalent for probabilities in $\mathbb{Q}$ [8, 11]. Along the same lines, one can show that deciding a predicate is in #P. Together with Theorem 6.1 we have the following corollary.

Corollary 6.2. *Evaluating any predicate of the form*

$$p(\psi) \text{ OP } c$$

for OP $\in \{>, \geq, <, \leq\}$ *is #P–equivalent.*

Next we show that evaluating any non-trivial predicate is hard.

Theorem 6.3. *Evaluating any non-trivial predicate is either NP–hard or coNP–hard. Thus unless P=NP or P=coNP there is no efficient algorithm evaluating predicates.*

Proof. Let ϕ be some non-trivial predicate over the confidence of the DNF formulas $\psi_1, \ldots, \psi_k$.

Before we describe the reduction showing that evaluating ϕ is as least as hard as deciding SAT we make preliminary observations about the predicate ϕ. The predicate $\phi(\psi_1, \ldots, \psi_k)$ is a Boolean combination of inequalities of the form $g(p(\psi_1), \ldots, p(\psi_k))$ OP 0, where OP $\in \{<, \leq, =, \neq \geq, >\}$ and $g(\cdot)$ is an arithmetic expression.

Since ϕ is non-trivial there are confidence values $p_1, \ldots, p_k$ such that varying p_i changes the value of the predicate. We claim that the value of the predicate cannot change infinitely often if we fix all variables but p_i and we vary p_i from 0 to 1.

The functions g can be rewritten as rational functions over the variable p_i. Now how often the value of the inequality $g(p(\psi_1), \ldots, p(\psi_k))$ OP 0 changes is bounded by the sum of the degrees of the polynomials in the numerator and denominator. Given two predicates, how often the AND or the OR of the two predicates changes the truth value is bounded by the sum of how often each predicate changes its truth value.

Consider our predicate ϕ again. By pushing down negations of the Boolean combination ($>$ becomes $\leq$, $=$ becomes $\neq$, etc.) and inductively analyzing the ANDs and ORs we get that the truth value cannot change infinitely often if we vary p_i from 0 to 1.

Hence, if we fix $p_1, \ldots, p_k$ and vary p_i then either (a) there is a value $\tau \in (0, 1]$ and there is a Boolean value b such that: for $p_i \in [0, \tau)$ the predicate ϕ evaluates to b and for $p_i = \tau$ the predicate ϕ evaluates to $\neg b$ or (b) there is a value $\tau \in [0, 1]$ and there is a Boolean value b such that: for $p_i \in [0, \tau]$ the predicate ϕ evaluates to b and there is a value ϵ such that for $p_i \in (\tau, \tau + \epsilon]$ the predicate ϕ evaluates to $\neg b$.

Case (a): Let χ be a CNF formula over variables $Y_1, \ldots Y_{n'}$ for which we want to determine whether it is satisfiable. We create a RUPD over all variables of ψ each with a probability of 1/2 and over a new variable X with a probability of τ. Furthermore, the RUPD records probabilities for $k-1$ new variables $X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_k$ with values $p_1, \ldots, p_{i-1}, p_{i+1}, \ldots, p_k$. Let $\psi = \neg\chi \wedge X$, which we can efficiently represent as a DNF. By construction we have that $p(\psi) \in [0, \tau]$. Hence, the following statements are equivalent

1. $\phi(X_1, \ldots, X_{i-1}, \psi, X_{i+1}, \ldots, X_k) = b$
2. $p(\psi) < \tau$
3. $p(\neg\chi) < 1$
4. χ is satisfiable

Case (b): Let χ be a CNF formula over variables $Y_1, \ldots Y_{n'}$ for which we want to determine whether it is satisfiable. We create a RUPD over all variables of ψ each with a probability of 1/2 and over a new variable X with a probability of $\frac{\tau}{1-\epsilon'}$, where $\epsilon' = \min(\epsilon, 1-\tau, 2^{-n'}, 1 - \frac{\tau}{\tau+\epsilon})$. Furthermore, the RUPD records probabilities for $k-1$ new variables $X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_k$ with values $p_1, \ldots, p_{i-1}$, p_{i+1}, $\ldots$, p_k. Let $\psi = \neg\chi \wedge X$. By construction we have that $p(\psi) \in [0, \tau + \epsilon]$.

We claim that $\phi(X_1, \ldots, X_{i-1}, \psi, X_{i+1}, \ldots, X_k)$ evaluates to b if and only if χ is satisfiable.

If χ is satisfiable then $p(\neg\chi) < 1$. Since all variables of χ have a probability 1/2 this implies $p(\neg\chi) \leq 1 - 2^{-n'} \leq 1 - \epsilon'$. Hence $p(\psi) = p(\neg\chi \wedge X) \leq (1-\epsilon')(\frac{\tau}{1-\epsilon'}) = \tau$. Thus, the predicate evaluates to b.

If χ is not satisfiable then $p(\neg\chi) = 1$. Hence $p(\neg\chi \wedge X) = p(X) = \frac{\tau}{1-\epsilon'} \leq \frac{\tau}{1-(1-\frac{\tau}{\tau+\epsilon})} \leq \tau + \epsilon$. Furthermore, $p(\psi) = \frac{\tau}{1-\epsilon'} > \tau$. Thus, the predicate evaluates to $\neg b$.

For both cases (a) and (b) we conclude that if $b = 1$ then evaluating ϕ is NP-hard. If $b = 0$ then evaluating ϕ is coNP-hard. □

Despite this negative result we could ask ourselves whether there is an efficient randomized algorithm if we are willing to accept errors. Before we answer this question, let us introduce the complexity class BPP.

Definition 6.4. The class of decision problems for which a PTIME randomized algorithm exists which outputs the correct answer with probability $\geq 2/3$ is called BPP. □

In fact, the constant 2/3 is arbitrary, it could be any constant greater than 1/2. (Note, that any algorithm with error probability $\geq 1/2$ is useless, because we can simply flip a coin.) As an immediate consequence of Theorem 6.3 we have the following corollary.

Corollary 6.5. *There is no efficient algorithm that evaluates a non-trivial predicate correctly with probability* $\geq 2/3$ *unless BPP⊃NP.*

It is considered unlikely that BPP ⊃ NP [8]. Since BPP = coBPP it is also considered unlikely that BPP ⊃ coNP.

We established that deciding simple predicates of the form $p(\psi)$ OP c is at least as hard as any enumeration problem in #P. Furthermore, there is no efficient randomized algorithm deciding any non-trivial predicate unless BPP⊃NP.

6.2 Efficiently Evaluating Predicates

The results in this section generalize the results of [14] to unreliable U–DBs. We cannot hope to efficiently evaluate all predicates with bounded error. Instead we will present an efficient algorithm that refuses to output a guess for some inputs, but for other inputs outputs a guess of the predicate that is correct with high probability $1 - \delta$.

We want to guess the value of a predicate that is based on confidences of DNFs $\psi_1, \ldots, \psi_k$ given a tuple t. Our guess, $\hat{\phi}(t)$ has to be correct with probability at least $1 - \delta$. To do so, for all i we approximate the confidence of ψ_i running the confidence computation for M_i rounds of sampling. The result is $\hat{p}^i_{\min}, \hat{p}^i_{\max}$. The guarantee is that for all ϵ and for

Algorithm 5: PREDICATE APPROXIMATION ($\phi, \psi_1, \ldots, \psi_k$, RUPD, δ, bound on running time $l_{\max}$, update rate r)

```
1  for $i \in [k]$ do $M_i = 0, M'_i = r|\psi_i|$
2  Let the number of rounds be $l = 0$
3  while $l \leq l_{\max}$ do
4      for $i \in [k]$ do
5          $\hat{p}^i_{\min}, \hat{p}^i_{\max} \leftarrow$ IMPROVE-CONF ($\psi_i, M'_i$)
6      Maximize $\epsilon$ // Using binary search
       subject to $\exists \hat{\phi} \forall \vec{x} \in \{\frac{\hat{p}^1_{\min}}{1+\epsilon}, \frac{\hat{p}^1_{\max}}{1-\epsilon}\} \times ... \times \{\frac{\hat{p}^k_{\min}}{1+\epsilon}, \frac{\hat{p}^k_{\max}}{1-\epsilon}\}$ :
       $\phi(\vec{x}) = \hat{\phi}$
7      if $\epsilon \neq \perp \wedge \epsilon > 0 \wedge 2ke^{\epsilon^2 \cdot l \cdot r/3} \leq \delta$ then
8          return guess $\hat{\phi}$
9      $l \leftarrow l + 1$
10     for $i \in [k]$ do $M_i \leftarrow M_i + M'_i$
11 return "Don't know"
```

all $\delta \geq 2e^{\frac{M_i \cdot \epsilon^2}{3|\psi_i|}}$:

$$\Pr\left[p(\psi_i) \in \left[\frac{\hat{p}^i_{\min}}{1+\epsilon}, \frac{\hat{p}^i_{\max}}{1-\epsilon}\right]\right] \geq 1 - \delta$$

Note that this guarantee holds for the basic Algorithm 2 of Sec. 5.1 that computes the average of M_i calls to the ESTIMATOR. This guarantee also holds for the general algorithm of Sec. 5.2 that computes up to 2^n of these averages each over M_i calls to the ESTIMATOR. This guarantee holds for the algorithm sketched in Sec. 5.2.3 that computes 2 averages each over M_i calls to the ESTIMATOR. We refer to any of these algorithms as CONF(ψ_i, M_i). They all have the nice property that successive improvements are possible. Let us assume the algorithms keep track of the averages and the number of samples. Then we can request an IMPROVE-CONF(ψ_i, RUPD, M'_i) which runs CONF for M'_i more rounds and outputs $\hat{p}^i_{\max}, \hat{p}^i_{\max}$ as the result of $M_i + M'_i$ rounds of sampling.

The next lemma upper bounds the error probability if we decide the predicate based on these estimates.

LEMMA 6.6. *[14] Let ϕ be a predicate over confidences of the DNFs $\psi_1, \ldots, \psi_k$. Let $\hat{p}^i_{\min}, \hat{p}^i_{\max}$ be the result of calling* CONF(ψ_i, M_i).

If for some ϵ the member points in the axis–parallel orthotope:

$$\left(\frac{\hat{p}^1_{\min}}{1+\epsilon}, \frac{\hat{p}^1_{\max}}{1-\epsilon}\right) \times \cdots \times \left(\frac{\hat{p}^k_{\min}}{1+\epsilon}, \frac{\hat{p}^k_{\max}}{1-\epsilon}\right)$$

all agree on a value $\hat{\phi}$ for ϕ, then

$$\Pr[\hat{\phi} \neq \phi(t)] \leq \sum_{i=1}^{k} 2e^{\frac{M_i \cdot \epsilon^2}{3|\psi_i|}}$$

In [14] it is remarked that one only has to check the corner points of the orthotope.

Algorithm 5 successively improves the estimates until the error bound $\sum_{i=1}^k 2e^{\frac{M_i \cdot \epsilon^2}{3|\psi_i|}}$ is at most the desired bound δ. In practice, one would try to figure out a trade–off between the overhead of checking whether the error-bound δ is achieved and the possible overhead of computing too many rounds and set the parameter r accordingly.

PROPOSITION 6.1. *Algorithm 5 is correct.*

PROOF SKETCH. By Lemma 6.6 and the correctness of CONF and IMPROVE-CONF for all DNFs the true confidence $p(\psi_i)$ is between $\frac{\hat{p}^i_{\min}}{1+\epsilon}$ and $\frac{\hat{p}^i_{\max}}{1-\epsilon}$ with probability at least $1 - 2e^{\frac{M_i \cdot \epsilon^2}{3}}$. Furthermore, if all confidences $p(\psi_i)$ are in fact in that interval then the predicate evaluates to $\hat{\phi}$. Hence, if the algorithm returns a guess $\hat{\phi}$ then this guess is correct with probability at least $1 - \sum_{i=1}^k 2e^{\frac{M_i \cdot \epsilon^2}{3}} = 1 - \sum_{i=1}^k 2e^{\frac{l \cdot r \cdot \epsilon^2}{3}} = 1 - 2ke^{\epsilon^2 \cdot l \cdot r/3}$ which is at least $1 - \delta$ (see line 7). □

The Limits of Predicate Approximation.

As we have seen in Sec. 6.1, evaluating predicates is hard. Thus, we cannot assume that it is always possible to achieve a given error bound δ. PREDICATE APPROXIMATION has an upper bound on the number of improvements $l_{\max}$. If after $l_{\max}$ calls to IMPROVE-CONF for each DNF it was not possible to achieve an error bound of $\leq \delta$ then the algorithm outputs "Don't know". Let us look at an example illustrating the impossibility of predicate approximation. Consider the predicate $\phi(\psi) = p(\psi) > 0.5$. Suppose that $p(\psi) = 0.5$. In this case, it is necessary to precisely compute the confidence instead of using an approximation. Also it can be impossible to come up with a good guess for this predicate if ψ involves unreliable variables and the lower bound is smaller than 0.5 and the upper bound is greater than 0.5. Here the variables are too unreliable to compute the predicate at all. Those difficult predicates are *not robust.*

DEFINITION 6.7. Given a predicate $\phi(\psi_1, \ldots, \psi_k)$ and an RUPD. ϕ is called ϵ_0–robust if, for all RRPD instantiations W^* of the RUPD and for all points $(\hat{p}_1, \ldots, \hat{p}_k)$ in the ϵ_0–neighborhood of the confidence values,

$$\phi(p_{W^*}(\psi_1), \ldots, p_{W^*}(\psi_k)) = \phi(\hat{p}_1, \ldots, \hat{p}_k)$$

where the ϵ_0–neighborhood of a point $p_1, \ldots, p_k$ contains all points $(\hat{p}_1, \ldots, \hat{p}_k)$ such that for all i: $p_i(1 - \epsilon_0) \leq \hat{p}_i \leq p_i(1 + \epsilon_0)$. □

We cannot determine the level of robustness of a predicate without computing the exact bounds on the confidences. Therefore, to keep our approximation algorithm efficient, we need to bound the running time. In particular, Algorithm 5 outputs guesses for $\frac{3\ln(\delta/2k)}{l_{\max}r}$–robust predicates with high probability.

PROPOSITION 6.1. *Let $\hat{\phi}$ be the guess returned by* PREDICATE APPROXIMATION *($\phi, \psi_1, \ldots, \psi_k$, RUPD, $\delta, l_{\max}$) after l executions of the while loop. Then with probability $\geq 1 - \delta$ the predicate is not $\frac{3\ln(\delta/2k)}{(l-1)r}$–robust.*

If the algorithm returns "Don't know" then with probability $\geq 1 - \delta$ the predicate is not $\frac{3\ln(\delta/2k)}{(l_{\max})r}$–robust.

This proposition justifies the running time of our algorithm. Because of the level of robustness the algorithm cannot terminate earlier.

7. ACKNOWLEDGMENTS

This material is based upon work supported by the AFOSR under Award No. FA9550-07-1-0437, by the U.S. Department of Homeland Security under Grant Award Number 5-36423.5750, and by the National Science Foundation under

Grants No. CNS-0627680 and IIS-0812272. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the sponsors.

8. REFERENCES

[1] Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases*. Addison-Wesley, 1995.

[2] Periklis Andritsos, Ariel Fuxman, and Renee J. Miller. Clean answers over dirty databases: A probabilistic approach. In *Proc. ICDE*, 2006.

[3] Lyublena Antova, Thomas Jansen, Christoph Koch, and Dan Olteanu. “Fast and Simple Relational Processing of Uncertain Data”. In *Proc. ICDE*, 2008.

[4] Philip Bohannon, Wenfei Fan, Floris Geerts, Xibei Jia, and Anastasios Kementsietsidis. “Conditional Functional Dependencies for Data Cleaning”. In *Proc. ICDE*, 2007.

[5] Sara Cohen, Benny Kimelfeld, and Yehoshua Sagiv. “Incorporating Constraints In Probabilistic XML”. In *Proc. PODS*, 2008.

[6] Nilesh Dalvi and Dan Suciu. “Efficient Query Evaluation on Probabilistic Databases”. In *Proc. VLDB*, 2004.

[7] Norbert Fuhr and Thomas Rölleke. “A Probabilistic Relational Algebra for the Integration of Information Retrieval and Database Systems”. *ACM Trans. Inf. Syst.*, **15**(1), 1997.

[8] Erich Grädel, Yuri Gurevich, and Colin Hirsch. “The Complexity of Query Reliability”. In *Proc. PODS*, 1998.

[9] Edward Hung, Lise Getoor, and V. S. Subrahmanian. “Probabilistic Interval XML”. In *Proc. ICDT*, 2003.

[10] T. Imielinski and W. Lipski. “Incomplete Information in Relational Databases”. *J. ACM*, **31**(4), 1984.

[11] Abhay Jha, Vibhor Rastogi, and Dan Suciu. “Query Evaluation with Soft-Key Constraints”. In *Proc. PODS*, 2008.

[12] David S. Johnson. A catalog of complexity classes. *Handbook of theoretical computer science (vol. A): algorithms and complexity*, pages 67–161, 1990.

[13] Richard M. Karp and Michael Luby. “Monte-Carlo Algorithms for Enumeration and Reliability Problems”. In *Proc. FOCS*, 1983.

[14] Christoph Koch. “Approximating Predicates and Expressive Queries on Probabilistic Data”. In *Proc. PODS*, 2008.

[15] Christoph Koch and Dan Olteanu. “Conditioning Probabilistic Databases”. In *Proc. VLDB*, 2008.

[16] Laks V. S. Lakshmanan, Nicola Leone, Robert B. Ross, and V. S. Subrahmanian. Probview: A flexible probabilistic database system. *ACM Trans. Database Syst.*, 22(3):419–469, 1997.

[17] Christopher Ré, Julie Letchner, Magdalena Balazinska, and Dan Suciu. “Event Queries on Correlated Probabilistic Streams”. In *Proc. SIGMOD*, 2008.

[18] Luca Trevisan. A note on approximate counting for k-dnf. In *Proc. APPROX-RANDOM*, 2004.

[19] Leslie G. Valiant. “The Complexity of Enumeration and Reliability Problems”. *SIAM J. Comput.*, **8**(3), 1979.

[20] Jennifer Widom. “TRIO: A System for Managing Data, Uncertainty, and Lineage”. In Charu Agarwal, editor, *Managing and Mining Uncertain Data*, 2008. To appear.

[21] Wenzhong Zhao, Alex Dekhtyar, and Judy Goldsmith. Query algebra operations for interval probabilities. In *Proc. DEXA*, 2003.

[22] Wenzhong Zhao, Alex Dekhtyar, and Judy Goldsmith. Databases for interval probabilities. *Int. J. Intell. Syst.*, 19(9):789–815, 2004.