Asymptotic Conditional Probabilities for Conjunctive Queries

Nilesh Dalvi, Gerome Miklau, and Dan Suciu

University of Washington

1 Introduction

Two seemingly unrelated applications call for a renewed study of probabilistic properties of logical formulas. One is the study of information about a sensitive query which is disclosed by a public view [10]. The other is a study of queries with uncertain predicates [1]. Both have been studied using a certain probabilistic model, which, as we show here has some limitations. In this paper we propose a new probabilistic model of databases, considered before for random graphs [12, 9] but not for databases, and study properties of conjunctive queries under this new model. This model provides a characterization of information disclosure between a query and view, with query answerability at one end of the spectrum, and logical independence (or perfect security) at the other.

Motivation 1: Information Disclosure We start by illustrating the limitation of the probabilistic model in [10]. The owner of a database I wishes to publish a view $V(I)$ over the database, and would like to determine whether certain sensitive information is disclosed by the view. The sensitive data is expressed in terms of a query, called the *sensitive query*, $Q(I)$. The *query-view security problem* requires one to check whether the view V does not leak any secret information about the query. In [10] this problem is modeled by comparing the *a priori* knowledge an adversary possesses about $Q(I)$, with the knowledge about $Q(I)$ given $V(I)$. The adversary's knowledge is described as the probability of $Q(I)$ attaining a certain value, when I is chosen randomly. If both the view and the sensitive query are boolean, the *a priori* probability is $\mathbf{P}(Q)$, while the *a posteriori* probability is the conditional probability $\mathbf{P}(Q \mid V)$. When the two values are identical, then the query is said to be *perfectly secure* w.r.t. the view. The work in [10] is focused on deciding, for conjunctive queries Q and V, when perfect security holds. Notice that the definition is for one fixed domain and probability distribution, although the results in [10] show that it is largely independent of both.

The problem is that perfect security is often too restrictive for practical purposes, rejecting as insecure query-view pairs that are probably acceptable in practice. This is illustrated in the following example:

Example 1. Suppose we have a sensitive database *Employee*(*name*, *department*, *email*), and suppose we would like to publish a view V consisting of all departments but hiding all employee names. Limiting our discussion to boolean queries and views, suppose we want to publish that one of the departments

is called *Amateur Astronomy*, but would like to hide the fact that one of the employees is *John Smith*. Then Q and V would be defined as follows:

$$V \leftarrow \mathit{Employee}(-, \text{“}\mathit{Amateur\ Astronomy}\text{”}, -)$$
$$Q \leftarrow \mathit{Employee}(\text{“}\mathit{John\ Smith}\text{”}, -, -)$$

Here Q is not perfectly secure w.r.t. to the view V. For a quick justification, consider the case where the domains for *name*, *department*, and *email* each consist of a single value (say *js@mystartup.com* for *email*). Then there are only two database instances, $\emptyset$ and $\{(\text{“}\mathit{John\ Smith}\text{”}, \text{“}\mathit{Amateur\ Astronomy}\text{”}, \text{“}\mathit{js@mystartup.com}\text{”})\}$, and assume each has probability 0.5, hence $\mathbf{P}(Q) = 0.5$. By contrast, $\mathbf{P}(Q \mid V) = 1$. In fact, it can be shown that for every domain and probability distribution[1] $\mathbf{P}$ we have $\mathbf{P}(Q \mid V) > \mathbf{P}(Q)$, and therefore the query is not perfectly secure w.r.t. the view. However, the difference between the two probabilities is tiny for large domains, and for practical purposes the information disclosure should be considered negligible. In practice, users are usually willing to publish the set of *department*'s, even if the employee *name*'s are sensitive. Thus, the notion of *perfect security* is a higher standard than what is currently used in practice.

Capturing *practical security*, as opposed to *perfect security*, was left open in [10].

Motivation 2: Query Answering in the Presence of Uncertainties For integrating large numbers of data sources the Local As View (LAV) approach has been proposed [7]. A global data instance I is specified indirectly, through a number of view definitions $V(I)$, one corresponding to each local data source. Only the materialized views $v = V(I)$ are known, but not the instance I, and any instance J is considered *possible* as long as[2] $v = V(J)$. A tuple t is a *certain answer* to a query $Q(I)$ if $t \in Q(J)$ for every possible instance J.

When integrating and querying unfamiliar data sources however, one often needs to deal with uncertainties. Uncertain facts in databases have been addressed for example in [5], where they have been modeled as *probabilistic databases*. A probabilistic database is a probability distribution $\mathbf{P}(I)$ over all instances I. There are several ways in which one can derive such a probability distribution. In one application, for example, predicates in a user query are interpreted as uncertain, and the degree to which a tuple in the database matches the predicate is transformed into a probability [1].

We propose to use probabilistic databases for query answering using views, thus allowing uncertainties to be handled during data integration. We still have local sources described as views, and are given the materialized views only. But now we have a probability distribution $\mathbf{P}(J)$ on all possible database instances J, and the probability of a tuple t is the sum of $\mathbf{P}(J)$ over all J's for which $t \in Q(J)$. All certain tuples will have probability 1, but other tuples may have probability close to 1, and should be considered as probable answers.

[1] We require $\mathbf{P}$ to be $\neq 0$ everywhere.
[2] Or $v \subseteq V(J)$ for the Open World Assumption.

Example 2. Continuing Example 1, suppose the company publishes two views, one with all *(name, department)* pairs in the database, and the other with all *(department, email)* pairs. Suppose we receive an email from *js@mystartup.com*, and would like to use these two views to find out the name of the person who sent out that email. As before, we restrict the discussion to boolean views and queries, and model the problem with the two boolean views below, and a boolean query (asking whether *"John Smith"* is the sender):

$$V' \leftarrow \textit{Employees}(\textit{"John Smith"}, \textit{"Amateur Astronomy"}, -)$$
$$V'' \leftarrow \textit{Employees}(-, \textit{"Amateur Astronomy"}, \textit{"js@mystartup.com"})$$
$$Q \leftarrow \textit{Employees}(\textit{"John Smith"}, -, \textit{"js@mystartup.com"})$$

In order for Q to be a certain answer we need to have the logical implication $V'V'' \rightarrow Q$, which does not hold in this example. That is, Q is not a certain answer given V. We argue, however, that a system should report *"John Smith"* as a possible answer to the query. To see why, assume for the moment that there are, on average, 5 employees per department. Then *"John Smith"* is an answer with probability 20%. Clearly, in some applications it is critical to return it as an answer. Assume now that we know nothing about the database, except that V' and V'' are true, and that the *Employees* table has a number of records which is much smaller than the size of the domains for the three attributes. Then, if the domain size is very large, the probability of *"John Smith"* being an answer approaches 1. Indeed, if we populate a fixed (say 100) number of tuples in *Employee* with random values from a huge domain, the probability that two tuples have the same *department* value is close to 0. Hence, the probability that the two tuples satisfying V' and V'' are in fact the same tuple approaches 1, and *"John Smith"* is an answer with a probability close to 1.

This surprising example justifies our quest for an investigation of the asymptotic conditional probability of queries.

Contributions In this paper we show that a certain new probability model provides a reasonable definition for both practical security and probable answers. In this model individual tuples have a uniform probability of occurring in the database, but the probability of each tuple t is now such that the expected size of the relation instance R is a given constant S (different constants may be used for different relation names). As the domain size n grows to ∞, the expected database size remains constant. Hence, in the case of directed graphs (i.e. a single, binary relation R), the probability that two given nodes are connected by an edge is S/n^2. Denoting by $\mu_n[Q]$ the probability that a boolean query Q is true on a domain of size n, our goal is to compute $\mu_n[Q \mid V]$ as $n \rightarrow \infty$.

For Information Disclosure we will propose, as a definition of practical security, the condition $\lim_n \mu_n[Q \mid V] = 0$. This is justified as follows. The adversary faces a large domain. For example, if he is trying to guess whether *"John Smith"* is an employee, then he has only a tiny probability of success: $1/n$ where n is the size of the domain. On the other hand, the size of the database is much smaller, and the adversary often knows a good approximation. This definition relaxes the previous definition of perfect security for sensitive queries Q (see Sec. 3).

For Query Answering we will propose, as a definition of probable answer, the condition $\lim_n \mu_n[Q \mid V] = 1$. Again, this relaxes the definition of certain answers (see Sec. 3).

The key technical contribution in this paper is to show that $\lim_n \mu_n[Q \mid V]$ for *conjunctive queries* Q and V always exists and to provide an algorithm for computing it. The key technical lemma is to show that, for each conjunctive query Q there exists two number c, d s.t. $\mu_n[Q] = c/n^d + O(1/n^{d+1})$. Moreover, both d and c can be computed algorithmically. Since $\mu_n[Q \mid V] = \mu_n[QV]/\mu_n[V]$, the main result follows easily.

Our main result leads to the following classification for query Q and view V, describing a spectrum of information disclosure and answerability.

Perfect query-view security $\mu_n[Q \mid V] = \mu_n[Q]$ for all n large enough. Here V provides no information about Q. This is the condition studied in [10].

Practical query-view security $\lim_{n\to\infty} \mu_n[Q \mid V] = 0$. This implies that the difference of probabilities is zero in the limit (since $\lim_n \mu_n[Q] = 0$ for all practical purposes). For finite n, V may in fact contain some information for answering Q, but it is considered negligible under our model.

Practical Disclosure $0 < \lim_{n\to\infty} \mu_n[Q \mid V] < 1$. Disclosure is non-negligible in this case. Our main result allows us to compute this quantity in terms of expected database size S.

Probable Query Answer $\lim_{n\to\infty} \mu_n[Q \mid V] = 1$. For any n, the answer to Q is not determined by V. However as $n \to \infty$, Q is almost surely true.

Certain Query Answer $\mu_n[Q \mid V] = 1$ for all n. Here V determines the answer to Q. That is, true is a certain answer to boolean query Q, given V.

Related Work When restricted to graphs, our random database model is an instance of the random graphs introduced by Erdös and Rényi [2]. A random graph on n is a graph on n vertices where each edge is chosen randomly and independently with probability $p(n)$. The study of convergence laws on random graphs was initiated independently by Fagin [3] and Glebskiĭ et al. [6]. They consider random graphs with $p(n)$ a constant and proved a 0-1 law for statements of first order logic, i.e. the asymptotic probabilities always converge to either 0 or 1. These results hold for a pure relational vocabulary, without constants or function symbols.

The work was later extended to a class of edge probabilities of the form $p(n) = \beta n^{-\alpha}$, where $\alpha, \beta \geq 0$. The results of Shelah and Spencer [12] and Lynch [9] show that a convergence law holds for all $\alpha \geq 1$ and for all irrational α between 0 and 1, although the limit need not be 0 or 1. For the case of one binary predicate our random databases correspond to the case $\alpha = 2$ and $\beta = S$ (the expected size).

The problem of evaluating asymptotic conditional probabilities has received relatively less attention. Fagin [3] shows that conditional probabilities do not always converge for first order probabilities. Liogon'kiĭ [8] proves that even the

problem of determining if conditional probabilities converge is undecidable for first order logic, but is decidable when restricted to only unary predicates.

Paper organization In Sec. 2 we review probability distributions for databases. In Sec. 3 we introduce the probabilistic model analyzed here. Sec. 4 contains the main theorems, with proofs deferred to the Appendix. We illustrate our main results on several examples in Sec. 5. We conclude in Sec. 6.

2 Basic Definitions and Background

We fix a vocabulary of relation names $R_1, R_2, \ldots R_m$. The number of attributes in relation R_i is its arity, denoted $A(R_i)$. For a finite domain D, a tuple for R_i is an element of $D^{A(R_i)}$, and we denote *Tup* the disjoint union of $D^{A(R_i)}$, for $i = 1, \ldots, m$. A database instance I over D is any subset of *Tup*, and *inst*(D) denotes the set of all database instances over D.

The probability distributions over database instances that we consider are always derived by choosing tuples independently. Each tuple $t \in$ *Tup* is assigned a probability $\mathbf{P}[t]$ that it will occur in the database instance. This induces the following probability distribution on instances $I \in$ *inst*(D):

$$\mathbf{P}[I] = \prod_{t \in I} \mathbf{P}[t] \cdot \prod_{t \notin I} (1 - \mathbf{P}[t]) \tag{1}$$

The problem considered in this paper concerns the probability of a query. Our discussion will be restricted to *conjunctive queries*, possibly with the inequality operators $\neq$. Thus, a query is a conjunction of predicates, where each predicate is either a relational predicate $R(t_1, \ldots, t_k)$, called a *subgoal*, or an inequality predicate $x \neq t$; here x is a variable, while $t, t_1, \ldots, t_k$ are either variables or constants. We use letters from the end of the alphabet for variables, e.g. x, y, z, u, v, and from the beginning of the alphabet for constants, e.g. a, b, c.

Our results are presented only for boolean queries. They also apply to non-boolean queries under the Open World Assumption (OWA), using the simple transformation illustrated for the query below:

$$q(x, y) \leftarrow R(x, a, z), S(z, y)$$

Here q is non-boolean. Suppose its answer includes the tuples (a, b) and (c, b). This is expressed as $q(a, b) \wedge q(c, b)$, which becomes the following boolean query:

$$Q \leftarrow R(a, a, z_1), S(z_1, b), R(c, a, z_2), S(z_2, b)$$

Statements about q and the fact that its answers include (a, b) and (c, b) are thus rephrased into statements about the boolean query Q. In the rest of this paper we will consider only boolean queries, unless otherwise stated.

Given a boolean query Q and a probability distribution over database instances, the following expression represents the probability that Q is true on a randomly chosen database instance I:

$$\mathbf{P}[Q] = \sum_{\{I \in inst(D) | Q(I) = true\}} \mathbf{P}[I] \tag{2}$$

3 Probabilistic Model

We now introduce our new twist to the probabilistic model, in which we let the domain size tend to ∞ while keeping the expected size of each relation instance fixed. Let D_n denote a domain of size n.

For each relation R_i in the vocabulary, fix a number S_i representing the expected size of R_i. We then define a specific probability distribution, denoted μ_n (instead of $\mathbf{P}$), having the following properties:

1. For each relation R_i, each tuple (element of $(D_n)^{A(R_i)}$) belongs to R_i independently and with equal probability.
2. For each relation R_i, the expected size of R_i is S_i, independent of n.

It follows that, for every tuple t of R_i, $\mu_n[t] = S_i/n^{A(R_i)}$.

Given a boolean query Q, its probability, $\mu_n[Q]$ given by the formula (2), is the probability that Q is true on an instance I randomly chosen from $inst(D_n)$. Similarly, define $\mu_n[Q_1 \mid Q_2]$ to be the conditional probability that a database chosen randomly from $inst(D_n)$ satisfies Q_1, given that it satisfies Q_2. It is equal to $\mu_n[Q_1Q_2]/\mu_n[Q_2]$. We are concerned with the following two asymptotic probabilities:

Definition 1. *For conjunctive query Q, the* **asymptotic probability** *of Q is $\mu[Q] = \lim_{n\to\infty} \mu_n[Q]$, if the limit exists.*

For conjunctive queries Q_1, Q_2, the **conditional asymptotic probability** *is $\mu[Q_1 \mid Q_2] = \lim_{n\to\infty} \mu_n[Q_1 \mid Q_2]$, if the limit exists.*

It is known[12] that $\mu[Q]$ exists for every pure relational FO formula Q (and, hence, for any conjunctive query without constants), and that it is not necessarily 0 or 1 (hence this model does not have a 0/1-law). To see the latter, consider the vocabulary of a single binary relation R, and the query $Q \leftarrow R(x, y)$. The query checks $R \neq \emptyset$. For each $n > 0$, $\mu_n[t] = S/n^2$ for any tuple t, and $\mu_n[Q] = 1 - (1 - S/n^2)^{n^2}$. Hence, $\mu[Q] = 1 - e^{-S}$.

Queries like the above are not interesting in database applications. More generally, call a subgoal of Q_1 *trivial* if it has no constants, and all its variables are distinct and do not occur in any other subgoals of Q_1 (they may occur however in inequality predicates). Trivial subgoals can be eliminated from a query Q_1, by splitting it into into a query Q without trivial subgoals, and several statements of the form $R \neq \emptyset$. For example, if $Q_1 \leftarrow R(x, y), T(u, a, b)$ (where $R(x, y)$ is a trivial subgoal), then $\mu_n[Q_1] = \mu_n[R \neq \emptyset]\mu_n[Q]$ where $Q \leftarrow T(u, a, b)$, and we have computed $\mu_n[R \neq \emptyset]$ above. For that reason we will assume throughout the paper that queries do not have trivial subgoals. It follows from our main results that in that case $\mu[Q] = 0$; conversely, $\mu[Q] > 0$ only if Q is a conjunction of queries of the form "R_i is non-empty".

Finally, we can define:

Definition 2. *Let Q and V be two boolean conjunctive queries.*

1. *Q is practically secure w.r.t. V if $\mu[Q \mid V] = 0$.*

2. Q is a probable answer given V if $\mu[Q \mid V] = 1$.

This definition relaxes previous definitions from the literature. Indeed, if Q has no trivial subgoals then $\mu[Q] = 0$; hence, if it is perfectly secure w.r.t V (i.e. $\mu_n[Q \mid V] = \mu_n[Q]$ for all n) then it is practically secure w.r.t V ($\mu[Q \mid V] = 0$). Similarly, if Q is a certain answer given V ($\mu_n[Q \mid V] = 1$ for all n) then it is a probable answer ($\mu[Q \mid V] = 1$).

4 Main Results

Throughout this section we consider conjunctive queries with inequality predicates $\neq$, and without trivial subgoals.

4.1 Main Result: Part I

Half of our main result is captured by the following theorem:

Theorem 1. *For every conjunctive query Q, there exists two numbers c, d such that:*

$$\mu_n(Q) = c(1/n)^d + O((1/n)^{d+1})$$

where d is an integer, $d \geq 1$. We denote c and d by $c_Q = \text{coeff}(Q)$ and $d_Q = \exp(Q)$ respectively.

It follows that $\mu[Q] = 0$. The number *exp*(Q) depends only on the query Q, while *coeff*(Q) depends both on the query Q and on $S_1, \ldots, S_m$, the expected cardinalities of the database relations. The second half of our main result shows how to compute *exp*(Q) and *coeff*(Q): we postpone it until we introduce the necessary notations.

Theorem 1 also implies the existence of the *conditional asymptotic probability* of formulas Q_1 and Q_2:

Corollary 1. *For any two boolean conjunctive queries Q_1, Q_2 the conditional asymptotic probability, $\mu(Q_1|Q_2)$, always exists and is as follows:*

$$\mu(Q_1|Q_2) = \begin{cases} 0 & \exp(Q_1Q_2) < \exp(Q_2) \\ \text{coeff}(Q_1Q_2)/\text{coeff}(Q_2) & \exp(Q_1Q_2) = \exp(Q_2) \end{cases}$$

We next show how to compute *coeff*(Q) and *exp*(Q).

4.2 Intuition

Here we illustrate the main intuition behind Theorem 1 and also motivate the notations needed to express *exp*(Q) and *coeff*(Q). We use a relational schema consisting of two tables, R and T, with arities 2 and 3 respectively, and expected sizes S_1 and S_2. Thus, given a domain $D_n = \{a_1, \ldots, a_n\}$, the probability of a

tuple $R(a_i, a_j)$ is $p_1 = S_1/n^2$ and the probability of a tuple $T(a_i, a_j, a_k)$ is $p_2 = S_2/n^3$. We consider the following three queries:

$$Q_1 \leftarrow R(a, x)$$
$$Q_2 \leftarrow R(a, x), T(x, y, b), R(y, c)$$
$$Q_3 \leftarrow T(a, b, x), T(a, y, c)$$

Here a, b, c are constants and we will assume that they occur in the domain D_n (hence $n \geq 3$). For each query Q, our goal is to express its probability as $\mu_n[Q] = c/n^d + O(1/n^{d+1})$, focusing on computing $c_Q = exp(Q)$ and $d_Q = coeff(Q)$.

Let's start with Q_1. There are n possible ways to substitute it's variable x with constants in the domain D_n, and, for each substitution $\{a_i/x\}$, the probability of the tuple (a, a_i) is $p_1 = S_1/n^2$, hence:

$$\mu_n[Q_1] \approx n \times p_1 = n \times \frac{S_1}{n^2} = \frac{S_1}{n} \tag{3}$$

suggesting $d_Q = 1$, $c_Q = S_1$. Of course, this is not a rigorous calculation, since we have approximated the probability of $R(a, a_1) \vee \ldots \vee R(a, a_n)$ with the sum of their probabilities. A rigorous calculation confirms the values for c_Q and d_Q:

$$\mu_n[Q_1] = 1 - (1 - p_1)^n = np_1 + \frac{n(n-1)}{2}p_1^2 + \ldots = \frac{S_1}{n} + O(\frac{1}{n^2})$$

Consider now Q_2. Using the same informal reasoning, there are n^2 possible substitutions for the variables x, y, and for each substitution $\{a_i/x, a_j/y\}$, the probability that all three tuples $R(a, a_i)$, $T(a_i, a_j, b)$, and $R(a_j, c)$ appear in the database is $S_1/n^2 \cdot S_2/n^3 \cdot S_1/n^2$. Thus:

$$\mu_n[Q_2] \approx n^2 \times \frac{S_1 S_2 S_1}{n^{2+3+2}} = \frac{S_1^2 S_2}{n^5} \tag{4}$$

which suggests $d_Q = 5$ and $c_Q = S_1^2 S_2$. A rigorous, but much more complex calculation (which is omitted) confirms that $\mu_n[Q_2] = S_1^2 S_2/n^5 + O(1/n^6)$.

Formulas (3) and (4) suggest the following definition:

Definition 3. *Let Q be a conjunctive query, and let $goals(Q)$ denote the set of its subgoals. We define several parameters, for each subgoal $g \in goals(Q)$ and for the entire query Q. For a subgoal g, we assume g is a predicate on the relation R_i, i.e. $g = R_i(t_1, \ldots, t_k)$. Recall that S_i is the expected cardinality of R_i.*

$$A(g) = A(R_i) \quad \textit{the "arity" of subgoal } g$$
$$C(g) = S_i \quad \textit{the "coefficient" of subgoal } g$$
$$V(Q) = \quad \textit{the number of distinct variables in } Q$$
$$A(Q) = \sum\{A(g) \mid g \in goals(Q)\}$$
$$D(Q) = A(Q) - V(Q) \quad \textit{the "exponent" of } Q$$
$$C(Q) = \prod\{C(g) \mid g \in goals(Q)\} \quad \textit{the "coefficient" of } Q$$

For our running example we have:

$$\begin{array}{ll} D(Q_1) = 2-1 = 1 & C(Q_1) = S_1 \\ D(Q_2) = (2+3+2)-2 = 5 & C(Q_2) = S_1 S_2 S_1 = S_1^2 S_2 \\ D(Q_3) = (3+3)-2 = 4 & C(Q_3) = S_2^2 \end{array}$$

Generalized to any query Q, our informal argument says that there are $n^{V(Q)}$ substitutions of its variable, each leading to an event with probability $C(Q)/n^{A(Q)}$; thus $\mu_n[Q] \approx C(Q)/n^{D(Q)}$. This suggests $exp(Q) = D(Q)$ and $coeff(Q) = C(Q)$. However, this is not true on Q_3, i.e. $\mu_n[Q_3] \neq S_2^2/n^4 + O(1/n^5)$, because the two subgoals in Q_3 unify to $T(a,b,c)$, hence $\mu_n[Q_3] \geq \mu_n[T(a,b,c)] = S_2/n^3$. We will show that $\mu_n[Q_3] = S_2/n^3 + O(1/n^4)$. The example Q_3 suggests that we need to consider unifications between subgoals in the query. We do that next.

4.3 Unifications

We assume here a conjunctive query Q with $\neq$ predicates. From Q, we generate a set of queries Q_0 by unifying some of the subgoals. Each Q_0 is obtained by (1) applying some substitution to Q, (2) dropping all $\neq$ predicates and (3) eliminating duplicate subgoals. While generate this set, we do not consider two Q_0 which are isomorphic. The steps are formally defined below.

Substitutions A *substitution*, η, is a mapping from variables to variables and constants. Importantly, we restrict substitutions to use only constants already in Q, thus, formally $\eta : Var(Q) \rightarrow Var(Q) \cup Const(Q)$. A substitution may not be defined on Q, if it violates some of the $\neq$ predicates. We denote $Q \models \eta$ if the substitution is defined on Q, and in that case $\eta(Q)$ denotes the result of applying η to the subgoals of Q (we drop the $\neq$ predicates). For example, if $Q \leftarrow R(a,x), R(x,y), R(y,z), x \neq y$ then the substitution $\eta = \{b/x, y/y, y/z\}$ is defined on Q and by applying it we obtain the query $Q_0 = \eta(Q)$, $Q_0 \leftarrow R(a,b), R(b,y), R(y,y)$. By contrast, the substitution $\eta' = \{x/x, x/y, z/z\}$ is not defined on Q.

Each substitution η defines a partition P on $goals(Q)$, such that two subgoals g, g' are in the same equivalence class if $\eta(g) = \eta(g')$. We call η a *unifier* for the partition P. Notice that $\eta(Q)$ has exactly $|P|$ subgoals, i.e. one for each equivalence class in P. For a trivial illustration, consider the query $Q \leftarrow R(a,x,b), R(x,y,v), R(z,z,w)$ and the substitution $\eta = \{z/x, z/y, b/v, b/w\}$. Then $\eta(Q) \leftarrow R(a,z,b), R(z,z,b)$, and η defines the partition $P = \{ \{R(a,x,b)\}, \{R(x,y,v), R(z,z,w)\}\}$, since the last two subgoals are mapped to the same subgoal by η.

A substitution η_0 is called the *most general unifier* for a partition P if for any other unifier η for P there exists a substitution θ s.t. $\eta = \theta \circ \eta_0$. In this case we call $\eta(Q)$ a *most general unifying query* of Q. Continuing our example, η is not a most general unifier for Q: the mgu is given by $\eta_0 = \{z/x, z/y, w/v\}$, which results in $\eta_0(Q) \leftarrow R(a,z,b), R(z,z,w)$. Indeed, η is obtained as $\theta \circ \eta_0$, for $\theta = \{b/w\}$.

Dropping $\neq$ All the $\neq$ predicates are dropped from the unifying query $Q_0 = \eta(Q)$. However, the $\neq$ predicates in Q are not ignored: they determine which substitutions we may apply to obtain all unifying queries.

Eliminate Duplicate Subgoals Since $goals(Q_0)$ is a set, this is an obvious operation. Considering again $Q \leftarrow R(a,x,b), R(x,y,v), R(z,z,w)$ and the substitution $\eta = \{z/x, z/y, b/v, b/w\}$, if we apply mechanically η to Q we obtain $Q_0 \leftarrow R(a,z,b), R(z,z,b), R(z,z,b)$. The subgoal $R(z,z,b)$ is a duplicate, however, and should be eliminated, i.e. $Q_0 \leftarrow R(a,z,b), R(z,z,b)$. While this sounds evident, we insist on it because the functions $D(-)$ and $C(-)$ return different (and wrong) results if we fail to eliminate duplicates.

Drop isomorphic queries When generating all unifying queries Q_0, we do not include two queries that are identical up to variable renaming.

We now formally define the set of unifying queries.

Definition 4. *Let Q be a conjunctive query. Define:*

$$UQ(Q) = \text{ the set of all unifying queries } Q_0 \text{ of } Q,\ Q_0 = \eta(Q)$$
$$MGUQ(Q) = \{Q_0 \mid Q_0 \in UQ(Q) \text{ and } Q_0 \text{ is a most general unifying query}\}$$

Note that any two distinct queries $Q_0, Q_0' \in UQ(Q)$ are not isomorphic.

4.4 Main Result: Part II

The second half of our main result, complementing Theorem 1 is:

Theorem 2. *Let Q be a conjunctive query possibly with $\neq$ predicates, and without trivial subgoals. Then the exponent* $\exp(Q)$ *and the coefficient* $\text{coeff}(Q)$ *in Theorem 1 are given by:*

$$\exp(Q) = \min\{D(Q_0) \mid Q_0 \in UQ(Q)\} \tag{5}$$
$$\text{coeff}(Q) = \sum\{C(Q_0) \mid Q_0 \in UQ(Q), D(Q_0) = \exp(Q)\} \tag{6}$$

Thus, to compute *exp*(Q) we have to iterate over all unifying queries Q_0 and take the minimum value of $D(Q_0)$. To compute *coeff*(Q) we have to iterate over all unifying queries Q_0 that achieve the minimum $D(Q_0)$. While this is an exponential time algorithm, as the next section shows, this cannot be avoided. Before illustrating the theorem, we prove that in both formulas (5) and (6) it suffices to iterate over $MGUQ(Q)$ rather than $UQ(Q)$. The algorithm becomes much more efficient, but remains exponential in the worst case. The correctness follows easily from the following:

Lemma 1. *Let $Q_0 \in UQ(Q) - MGUQ(Q)$. Then there exists $Q_1 \in MGUQ(Q)$ s.t. $D(Q_1) < D(Q_0)$.*

Proof. Let $Q_0 = \eta(Q)$, and let η define a partition P with k sets. Let η_1 be the most general unifier for the same partition, and denote $Q_1 = \eta_1(Q)$; $Q_1 \in MGUQ(Q)$. There exists θ s.t. $Q_0 = \theta(Q_1)$. Both Q_0 and Q_1 have exactly

k distinct subgoals, hence $A(Q_0) = A(Q_1)$. Moreover, θ is not an isomorphism (since $Q_0 \notin MGUQ(Q)$), hence it either maps at least one variable to a constant, or it maps two distinct variables to the same variable. In both cases $V(Q_0) < V(Q_1)$, hence $D(Q_0) = A(Q_0) - V(Q_0) > D(Q_1) = A(Q_1) - V(Q_1)$.

In all examples below we will compute *exp*(Q) and *coeff*(Q) by using $MGUQ(Q)$ instead of $UQ(Q)$ in Theorem 2. The proof, however, will be for $UQ(Q)$.

We now illustrate Theorem 2 on several examples. For our three queries above, we have $MGUQ(Q_1) = \{Q_1\}$, $MGUQ(Q_2) = \{Q_2\}$, and $MGUQ(Q_3) = \{Q_3, Q_3'\}$ where $Q_3' \leftarrow T(a, b, c)$. This confirms the values for *exp*(Q_i), *coeff*(Q_i) we have found above for $i = 1, 2$. For Q_3, we have $D(Q_3') = 3 < D(Q_3) = 4$, hence *exp*$(Q_3) = 3$ and *coeff*$(Q_3) = C(Q_3') = S_2$.

For a slightly more complex example, consider the following two queries:

$$Q_4 \leftarrow R(a, x), R(y, b)$$
$$Q_5 \leftarrow R(a, x), R(y, b), x \neq b$$

Here $MGUQ(Q_4) = \{Q_4, Q_4'\}$, where $Q_4' \leftarrow R(a, b)$. We have $D(Q_4) = D(Q_4') = 2 =$ *exp*(Q_4), hence *coeff*$(Q_4) = C(Q_4) + C(Q_4') = S_1^2 + S_1$, according to Equation (6). By contrast, $MGUQ(Q_5) = \{Q_4\}$, since Q_4' is not a unifying query for Q_5. It follows:

$$\mu_n[Q_4] = \frac{S_1^2 + S_1}{n^2} + O(\frac{1}{n^3}) \qquad \mu_n[Q_5] = \frac{S_1^2}{n^2} + O(\frac{1}{n^3})$$

4.5 Complexity of evaluating *coeff* and *exp*

Theorem 3. *Given a conjunctive query Q expected sizes of the relations, and a number k, deciding* $\exp(Q) \leq k$ *is NP-hard and evaluating* $\text{coeff}(Q)$ *is #P-hard in the size of the query.*

The proof is omitted. Although evaluating these parameters is hard in the general case, there are several cases where its very efficient. An example is a query where no two sub-goals can be unified. For instance, a query with no relation occuring multiple times, or same relation always occuring with different constants.

5 Applications

We illustrate here our main results with five examples, corresponding to the five classes of query-view pairs described in Sec. 1. Recall that $\mu[Q \mid V] = \lim_{n\to\infty} \mu_n[Q \mid V]$

Perfect query-view security This class is defined by $\mu_n[Q \mid V] = \mu_n[Q]$ for all n large enough. An example is:

$$V \leftarrow R(a, x); \quad Q \leftarrow R(b, x)$$

We showed[10] that $\mathbf{P}(Q \mid V) = \mathbf{P}(Q)$ for all domains D and tuple-independent probability distributions $\mathbf{P}$. The view leaks absolutely nothing about the query.

Practical query-view security This class is defined by $\mu[Q \mid V] = 0$. Consider the following example:

$$V \leftarrow R(a, y); \quad Q \leftarrow R(x, b)$$

We have $exp(V) = 2-1 = 1$, and $exp(QV) = 2$ (see query Q_4 in Sec. 4.4). Hence $\mu[Q \mid V] = 0$. Example 1 in Sec. 1 is a variation. There:

$$V \leftarrow R(x, b, z); \quad Q \leftarrow R(a, y', z')$$

and we have $exp(V) = 3 - 2 = 1$, and $exp(QV) = 2$, since $UQ(QV) = \{ \{ R(x, b, z),\ R(a, y', z') \},\ \{R(a, b, z)\} \}$. Again, $\mu[Q \mid V] = 0$. In both cases, although V leaks a tiny amount of information about Q, it is safe to publish V while keeping Q secret as long as the domain D is very large.

Practical disclosure This class is defined by $0 < \mu[Q \mid V] < 1$. For an illustration, consider

$$V \leftarrow R(a, y), R(x, b); \quad Q \leftarrow R(a, b)$$

We have $exp(V) = 2$, $coeff(V) = S + S^2$ (see query Q_4 in Sec. 4.3). For QV we note that $MGUQ(QV) = \{\{ R(a, y),R(x, b),R(a, b)\},\ \{R(a, b),R(x, b)\},\ \{R(a, b),R(a, y)\},\ \{R(a, b)\}\}$, and that the minimum $D(-)$ is attained only by the last query, $R(a, b)$. Hence $exp(QV) = 2$, $coeff(QV) = S$. It follows that $\mu[QV] = 1/(1 + S)$. Depending on the application, this may be considered to be an important leakage. For example, if the database has $S = 1000$ tuples, then an attacker has a chance of 0.1% of guessing the answer to Q.

This is an important example in practice. Suppose $R(name, phone)$ represents names and phone numbers, and the owner wants to publish all names, then separately all phone numbers, and wonders if the association between names and phone numbers remains secret. Clearly, it does not, since an attacker can pick a random name and phone number and associate them with about $1/S$ chance of success. Unless S is huge, this is an important leakage.

Probable query answering Recall that this class is defined by $\mu_n[Q \mid V] = 1$. We illustrate this with an abstraction of Example 2 in Sec. 1:

$$V \leftarrow R(a, b, z), R(x, b, c); \quad Q \leftarrow R(a, y, c)$$

Here $MGUQ(V) = \{\{R(a, b, z), R(x, b, c)\}, \{R(a, b, c)\}\}$ and the minimum $D(-)$ is attained only by $R(a, b, c)$, hence $exp(V) = 3$ and $coeff(V) = S$. In a similar way, $MGUQ(QV) = \{\{R(a, b, z),R(x, b, c),R(a, y, c)\},\ \{R(a, b, c),R(a, b, z)\},\ \{R(a, b, c),\ R(a, y, c)\},\ \{R(a, b, c),\ R(x, b, c)\},\ \{R(a, b, c)\}\}$, and the minimum $D(-)$ is also attained only by the last query, hence $exp(QV) = 3$, $coeff(QV) = S$. It follows that $\mu[Q \mid V] = 1$.

This may also be important in practice. Although V does not logically imply Q, one may argue that for practical purposes if we know that V is true

then we also know Q. For example, suppose we integrate two data sources $R_1(name, phone)$ and $R_2(phone, email)$, by describing them as projections of a global relation $R(name, phone, email)$. Suppose a user wants to find the email address of "John Smith", and that R_1 contains ("John Smith", 1234) and R_2 contains $(1234, js@com)$. The tuple ("John Smith", js@com) is not a certain answer, however it is a very probable answer, and should normally be returned to the user.

Certain answers Recall that this class is defined by $\mu_n[Q \mid V] = 1$ for all n. An example is:

$$V \leftarrow R(a, x, x); \quad Q \leftarrow R(a, y, z)$$

6 Conclusion

Our results show that for conjunctive queries, asymptotic conditional probabilities always exist, and can be evaluated algorithmically. Our results also hold when constants are allowed in the logic, which is required when queries need to refer to specific objects, and when converting non-boolean queries to boolean queries. We have shown that this model has interesting applications both to information disclosure and query answering.

References

1. N. Dalvi and D. Suciu. Efficient query evaluation on probabilistic databases. In *Conference on Very Large Data Bases*, 2004.
2. P. Erdös and A. Rényi. On the evolution of random graphs. *Magyar Tud. Akad. Mat. Kut. Int. Kozl.*, 5:17–61, 1960.
3. R. Fagin. Probabilities on finite models. *Journal of Symbolic Logic*, 41(1):50–58, 1976.
4. C. Fortuin, P. Kasteleyn, and J. Ginibre. Correlation inequalities on some partially ordered sets. *Comm.in Math. Physics*, 22:89–103, 1971.
5. N. Fuhr and T. Rlleke. A probabilistic relational algebra for the integration of information retrieval and database systems. *ACM Transactions on Information Sysytems*, 15(1):32–66, 1997.
6. Y. V. Glebskiĭ, D. I. Kogan, M. I. Liogon'kiĭ, and V. A. Talanov. Range and degree of realizability of formulas in the restricted predicate calculus. *Kibernetika*, 2:17–28, 1969. [Engl. Transl. Cybernetics, vol. 5, 142–154 (1972)].
7. A. Halevy. Answering queries using views: A survey. *VLDB Journal*, 10(4):270–294, 2001.
8. M. I. Liogon'kiĭ. On the conditional satisfyability ratio of logical formulas. *Mathematical Notes of the Academy of the USSR*, 6:856–861, 1969.
9. J. F. Lynch. Probabilities of sentences about very sparse random graphs. *Random Struct. Algorithms*, 3(1):33–54, 1992.
10. G. Miklau and D. Suciu. A formal analysis of information disclosure in data exchange. In *ACM SIGMOD International Conference on Management of Data*, pages 563–574, June 2004.
11. C. E. Shannon. Communication theory of secrecy systems. In *Bell System Technical Journal*, 1949.
12. J. Spencer and S. Shelah. Zero-one laws for sparse random graphs. *J. Amer. Math. Soc.*, pages 97–115, 1988.

A Appendix

We prove here Theorems 1 and 2.

An *event* is a set of tuples, $e \subseteq Tup$, and we denote $\mu_n[e]$ the probability that all tuples are in a randomly chosen database instance. Since all tuples are independent events, we have $\mu_n[e] = \prod_{t \in e} \mu_n[t]$. If $e_1, \ldots, e_m$ are events then $e_1 \vee \ldots \vee e_m$ denotes the event that at least one of them happens, i.e. a randomly chosen database instance contains all tuples in e_i, for some $i = 1, \ldots, m$. The proof of theorems 1 and 2 relies on the following inequalities, representing a lower bound and an upper bound for $\mu_n[e_1 \vee \ldots \vee e_m]$, and which are standard in probability theory:

$$\sum_{i=1,m} \mu_n[e_i] - \sum_{1 \le i < j \le m} \mu_n[e_i e_j] \le \mu_n[e_1 \vee \ldots \vee e_m] \le \sum_{i=1,m} \mu_n[e_i] \tag{7}$$

The event $e_i e_j$ represents the fact that all tuples in e_i and e_j are chosen; it is equivalent to the event $e_i \cup e_j$.

Given a conjunctive query Q_0, denote $Q_0^{\neq}$ the query obtained by adding all possible $\neq$ predicates, between any two distinct variables in Q_0, and between any variable and constant in Q. For example, if $Q_0 \leftarrow R(a, x), R(x, y)$ then $Q_0^{\neq} \leftarrow R(a, x), R(x, y), x \neq y, x \neq a, y \neq a$. The proof of the main result consists proving the following two equalities, then applying Eq.(7) to each of them.

$$Q \equiv \bigvee \{Q_0^{\neq} \mid Q_0 \in UQ(Q)\} \tag{8}$$

$$Q_0^{\neq} \equiv \bigvee \{\theta(Q_0^{\neq}) \mid Q_0^{\neq} \models \theta\} \tag{9}$$

In Eq.(9) the substitution θ ranges over all substitutions of the variables in Q_0 with constants[3] in D_n. This equation is the standard semantics of conjunctive queries, and we will not illustrate or discuss it further. Instead, we focus on Eq.(8). We first illustrate it on $Q \leftarrow R(a, x), R(x, y)$. This query has five unifiers, $UQ(Q) = \{Q, Q_1, Q_2, Q_3, Q_4\}$:

$$Q_1 \leftarrow R(a, a), R(a, y),\ Q_2 \leftarrow R(a, x), R(x, a),\ Q_3 \leftarrow R(a, z), R(z, z),\ Q_4 \leftarrow R(a, a)$$

We have seen $Q^{\neq}$; similarly $Q_1^{\neq} \leftarrow R(a, a), R(a, y), y \neq a$, etc. Eq.(8) says:

$$Q \equiv Q^{\neq} \vee Q_1^{\neq} \vee Q_2^{\neq} \vee Q_3^{\neq} \vee Q_4^{\neq}$$

Now we prove (8). The containment in one direction is easy: $Q_0 \subseteq Q$ for $Q_0 \in UQ(Q)$ follows from the standard homomorphism theorem (since $Q_0 = \eta(Q)$), and $Q_0^{\neq} \subseteq Q_0$ is also immediate. For the other direction, consider one database instance I where Q is true, and let θ be the substitution that makes Q true. We will find some $Q_0 \in UQ(Q)$, s.t. $Q_0^{\neq}$ is also true in I. Let $const(Q)$ be all constants in Q, and $C = \{c_1, \ldots, c_m\}$ be all constants in $\theta(Q)$ that are not in $const(Q)$. Let $z_1, \ldots, z_m$ be m fresh variables, one for each constant in C. Define the following substitution η on Q's variables. If $\theta(x) \in const(Q)$, then $\eta(x) = \theta(x)$; otherwise, if $\theta(x) = c_i$, $i = 1, \ldots, m$, then $\eta(x) = z_i$. Let $Q_0 = \eta(Q)$. By definition $UQ(Q)$ contains some isomorphic copy of Q_0, so assume w.l.o.g. $Q_0 \in UQ(Q)$. The valuation θ_0 defined by $\theta_0(z_i) = c_i$, $i = 1, m$ is defined on $Q_0^{\neq}$, and $\theta_0(Q_0) = \theta(Q)$, proving that $Q_0^{\neq}$ is true on the instance I.

We now sketch the proof of Theorems 1 and 2, by proving an upper bound and a lower bound for $\mu_n[Q]$.

Upper bound We apply the upper bound in (7) twice: first to Eq.(8), then, for each unifying query $Q_0 \in UQ(Q)$, to Eq.(9). We obtain:

$$\mu_n[Q] \le \sum_{Q_0 \in UQ(Q)} \sum_{\theta : Q_0^{\neq} \models \theta} \mu_n[\theta(Q_0^{\neq})]$$

For each substitution θ that is defined on $Q_0^{\neq}$ we have $\mu_n[\theta(Q_0^{\neq})] = C(Q_0)/n^{A(Q_0)}$ (see Definition 3 for notations). This is because $\theta(Q_0^{\neq})$ is a set of tuples having one distinct tuple for each subgoal in Q_0: the $\neq$ predicates prevent θ from mapping two subgoals to the same tuple. Moreover, there are $n^{V(Q_0)} - O(n^{V(Q_0)-1})$ substitutions θ that are defined on $Q_0^{\neq}$. Hence, for each unifier Q_0, the inner sum above is $C(Q_0)/n^{D(Q_0)} - O(1/n^{D(Q_0)+1})$. When summing up over all unifiers, the

[3] Unlike our definition in Sec. 4.3, here we do allow the substitution θ to use constants that do not appear in the query.

dominant terms are those with the lowest $D(Q_0)$, hence we have proven the following upper bound (see Theorem 2 for $exp(Q)$ and $coeff(Q)$):

$$\mu_n[Q] \leq \frac{coeff(Q)}{n^{exp(Q)}} + O(\frac{1}{n^{exp(Q)+1}})$$

Lower bound This is harder, because we have to prove that the second order terms in the lower bound of Eq.(7) are negligible: more precisely we show that the total contribution of these terms is $O(1/n^{exp(Q)+1})$. We first apply the lower bound to Eq.(8). The second order terms are here expressions of the form $\mu_n[Q_0^{\neq}Q_1^{\neq}]$, where $Q_0, Q_1 \in UQ(Q)$. Here $Q_0^{\neq}Q_1^{\neq}$ represents the conjunction of the two boolean queries, and is obtained by first renaming all variables in Q_0 and Q_1 to make them disjoint, and then taking the union of all predicates in the two queries, both subgoals and $\neq$ predicates. The number of such expressions depends only on Q, not on n, so it suffices to show that each such expression is $O(1/n^{exp(Q)+1})$. This follows from the following lemma, and our already proven upper bound:

Lemma 2. *Let Q_0 and Q_1 be two non-isomorphic conjunctive queries, without $\neq$ predicates. Then:*

$$\exp(Q_0^{\neq}Q_1^{\neq}) \geq \min(D(Q_0), D(Q_1)) + 1$$

Indeed, for $Q_0, Q_1 \in UQ(Q)$ the upper bound we have already shown gives us $\mu_n[Q_0^{\neq}Q_1^{\neq}] = O(1/n^{exp(Q_0^{\neq}Q_1^{\neq})})$, and the lemma implies that $exp(Q_0^{\neq}Q_1^{\neq}) \geq exp(Q)+1$. We now prove the lemma. Assume the contrary, that $exp(Q_0^{\neq}Q_1^{\neq}) \leq D(Q_0)$ and $exp(Q_0^{\neq}Q_1^{\neq}) \leq D(Q_1)$. The first assumption implies that there exists a unifier $Q_0' = \eta(Q_0^{\neq}Q_1^{\neq})$ s.t. $D(Q_0') \leq D(Q_0)$. Since $goals(\eta(Q_0^{\neq})) \subseteq goals(\eta(Q_0^{\neq}Q_1^{\neq}))$ we have $D(\eta(Q_0^{\neq})) \geq D(\eta(Q_0^{\neq}Q_1^{\neq}))$; with the equality holding only if $goals(\eta(Q_0^{\neq}))$ $= goals(\eta(Q_0^{\neq}Q_1^{\neq}))$, because there are no trivial subgoals in $\eta(Q_0^{\neq}Q_1^{\neq})$. (One can verify that if Q has no trivial subgoals, then $\eta(Q)$ has no trivial subgoals either.) Moreover, η maps all subgoals of $Q_0^{\neq}$ to distinct subgoals, because of the $\neq$ predicates, hence $D(Q_0) \geq D(\eta(Q_0^{\neq}))$, and equality holds only if η is an isomorphism. We have thus shown that $D(Q_0) \geq D(\eta(Q_0^{\neq}Q_1^{\neq})) \geq exp(Q_0^{\neq}Q_1^{\neq})$ and, given our first assumption, all three numbers are equal. This implies that $\eta(Q_0^{\neq}Q_1^{\neq})$ is an isomorphic copy of Q_0, which means that η is an injective function from Q_1 to (an isomorphic copy of) Q_0. Similarly, using the second assumption we prove the existence of an injective function from Q_0 to Q_1, implying that Q_0 and Q_1 are isomorphic, and contradicting the lemma's assumption.

We have shown so far $\sum_{Q_0 \in UQ(Q)} \mu_n[Q_0^{\neq}] - O(1/n^{expQ+1}) \leq \mu_n[Q]$. Given the upper bound, it suffices to consider in the sum only unifiers Q_0 for which $D(Q_0) = exp(Q)$: the others result in lower order terms. We apply now Eq.(9) to $Q_0^{\neq}$, and then the lower bound in (7). The higher order terms are now of the form $\mu_n[\theta(Q_0^{\neq})\theta'(Q_0^{\neq})]$, and we will show that their combined effect is $O(1/n^{exp(Q)+1})$. The number of such terms is now dependent on n. Denote $e = \theta(Q_0^{\neq})$ and $e' = \theta'(Q_0^{\neq})$. Both e and e' are sets of tuples, and they have both the same number of tuples, namely equal to the number of subgoals in Q_0, because both θ and θ' are injective (due to the $\neq$ predicates). We examine their overlap. Consider two tuples $t \in e$ and $t' \in e'$ s.t. $t = t'$. They cannot come from two distinct subgoals in Q_0, because in that case those two subgoals were unifiable, and, after unifying them, one obtains $Q_1 \in UQ(Q)$ s.t. $D(Q_1) < D(Q_0)$, contradicting the fact that $D(Q_0) = exp(Q)$. So t and t' correspond to the same subgoal in Q_0. Consider all subgoals in Q_0 that are mapped to the same tuples by θ and θ'. Define a new boolean query Q_1 consisting of precisely these subgoals; hence $goals(Q_1) \subset goals(Q_0)$ (we cannot have equality because $\theta \neq \theta'$). The intuition here is that, when Q_1 has few subgoals (or, e.g., is empty), then $\mu_n[ee']$ is very small, since e and e' are largely independent; when Q_1 has many subgoals, then we use the fact that there cannot be too many pairs of valuations θ, θ' that agree on all subgoals in Q_1. For these we need the following inequalities, which are easily checked. (1) $\mu_n[\theta(Q_0^{\neq})\theta'(Q_0^{\neq})] = O(1/n^{2A(Q_0)-A(Q_1)})$, and (2) the number of pairs of substitutions θ, θ' which agree precisely on the subgoals in Q_1 is $O(n^{2(V(Q_0)-V(Q_1))})$. Now we can add the second order terms and obtain:

$$\begin{aligned}\sum_{Q_0(\neq) \models \theta,\theta'} \mu_n[\theta(Q_0^{\neq})\theta'(Q_0^{\neq})] &= \sum_{Q_1 : goals(Q_1) \subset goals(Q_0)} O(\frac{n^{2(V(Q_0)-V(Q_1))}}{1/n^{2A(Q_0)-A(Q_1)}}) \\ &= \sum_{Q_1 : goals(Q_1) \subset goals(Q_0)} O(\frac{1}{n^{2D(Q_0)-D(Q_1)}}) \leq O(1/n^{D(Q_0)+1})\end{aligned}$$

For the last inequality we have used the fact that $D(Q_1) < D(Q_0)$, since $goals(Q_1) \subset goals(Q_0)$ and Q_0 has no trivial subgoals.