Materialized Views in Probabilistic Databases

For Information Exchange and Query Optimization

Christopher Ré
University of Washington
chrisre@cs.washington.edu

Dan Suciu
University of Washington
suciu@cs.washington.edu

ABSTRACT

Views over probabilistic data contain correlations between tuples, and the current approach is to capture these correlations using explicit *lineage*. In this paper we propose an alternative approach to materializing probabilistic views, by giving conditions under which a view can be represented by a block-independent disjoint (BID) table. Not all views can be represented as BID tables and so we propose a novel partial representation that can represent all views but may not define a unique probability distribution. We then give conditions on when a query's value on a partial representation will be uniquely defined. We apply our theory to two applications: query processing using views and information exchange using views. In query processing on probabilistic data, we can ignore the lineage and use materialized views to more efficiently answer queries. By contrast, if the view has explicit lineage, the query evaluation must reprocess the lineage to compute the query resulting in dramatically slower execution. The second application is information exchange when we do not wish to disclose the entire lineage, which otherwise may result in shipping the entire database. The paper contains several theoretical results that completely solve the problem of deciding whether a conjunctive view can be represented as a BID and whether a query on a partial representation is uniquely determined. We validate our approach experimentally showing that representable views exist in real and synthetic workloads and show over three magnitudes of improvement in query processing versus a lineage based approach.

1. INTRODUCTION

A view over probabilistic data contains correlations between tuples which make views expensive to represent. Currently, materialized views are based on a complete approach (e.g. [20, 36, 38]) which can represent any conjunctive view but requires storing auxiliary information (e.g. lineage [20]). Lineage (or provenance [17]) has many advantages, especially for scientific applications [8]. However, there are important drawbacks with lineage for information exchange and query optimization using views. In query optimization using views, to compute probabilities correctly we must determine how tuples are correlated. Inspecting the lineage of tuples in a view at query execution time is expensive since the lineage of a single tuple can be as large as the database. In contrast, if we can prove that all tuples in the view are independent, we do not need to examine the lineage at execution time. Further, we can optimize our query to use much more efficient query processing techniques (e.g. safe plans [19, 18, 35, 36]). In some applications of exchanging views of probabilistic data, using lineage to describe the data semantics is not an option. For example, we may not want to disclose any lineage (e.g. B2B applications) or the size of the lineage may be prohibitive. We study if a view can be understood without the full lineage.

In this paper, we consider the block-independent disjoint (**BID**) formalism. Informally, each relation is partitioned into blocks that are disjoint but, across blocks, tuples are independent. The **BID** formalism captures many representations previously discussed in the literature (e.g. tuple independent [18, 30], p-or tables [25], ?-tables, x-relations [20] and is essentially the same as [6]). We study the **view representation problem** (Problem 1) which is to decide: Given a conjunctive view V, is the output of V **representable** as a **BID** table? Not all views can be represented as **BID** tables and so we propose a **partial representation** which can represent any view but may not specify how all tuples in a view correlate. Because all correlations among tuples are not defined, many probability distributions can agree with a single partially representable view which can cause a query's value to fail to be uniquely defined. This motivates us to study the **partial view answering problem** (Problem 2) which is: Given a query Q using partially representable views, is Q's value uniquely defined?

We validate our solutions using data given to us by iLike.com [13], a service provided by the music site GarageBand which provides users with recommendations for music and friends. iLike has three characteristics which make it a good candidate for materialized views. First, the data are uncertain because the recommendations are based on similarity functions. Second, the database is large (e.g. tens of gigabytes) and backs an interactive website with many users; hence, query performance is important. Third, the database hosts more than one service, implying that there are integration issues. Probabilistic materialized views are a tool that addresses all three of these issues. Interestingly, materialized views are present in iLike. Since there is no support for un-

VLDB '07, September 23-28, 2007, Vienna, Austria.

certain views, the materialized views are constructed in an *ad-hoc* manner and require care to use correctly. Our experiments show that 80% of the materialized views in iLike are representable by BID tables and more than 98.5% of their workload could benefit from our optimizations. We also experiment with synthetic workloads and find that the majority of queries can benefit from our techniques. Using the techniques described in this paper, we are able to achieve speed ups of more than three orders of magnitude on large probabilistic data sets (e.g. TPC 1G with additional probabilities). A summary of our contributions:

- We solve the *view representation problem* for the case of conjunctive views over BID databases (Def. 2.2) which capture several representations in the literature [11, 18, 36, 39, 41]. Specifically:
  - We give a sound and complete algorithm for finding a representation when one exists and prove that the decision is Π_2^P Complete in general. (Sec. 4.1)
  - We give a polynomial time approximation for the *view representation problem* which is sound, but not complete. We show that it is complete for all queries without self-joins. (Sec. 4.2)
- We solve the *probabilistic materialized view answering problem* for the case of conjunctive queries and conjunctive views.
  - We propose a partial representation system to handle views that are not representable. (Sec. 5.1)
  - We give a sound and complete algorithm to decide if Q correctly uses a partially represented view and prove this decision is Π_2^P-Complete. (Sec. 5.2)
  - We give a polynomial time approximation for the *probabilistic materialized view answering problem* that is sound, but not complete. We show that it is complete for queries without repeated probabilistic views. (Sec. 5.3)
- We validate our techniques experimentally (Sec. 7), showing that representable views exist in practice, that our techniques yield several orders magnitude improvement and that our practical algorithms are almost always complete.

2. PROBLEM DEFINITION

Our running example is a scenario in which a user, Alice, maintains a restaurant database that is extracted from web data and she wishes to send data to Bob. Alice's data are uncertain because they are the result of *information extraction* [33, 26, 28] and *sentiment analysis* [23]. A natural way for her to send data to Bob is to write a view, materialize its result and send the result to Bob. We begin with some preliminaries about probabilistic databases based on the *possible worlds model* [5] and discuss the data in Fig. 1.

2.1 Representation Formalism

DEFINITION 2.1 (REPRESENTATION SYNTAX). *A **block independent disjoint table description** (BID table description) is a relational schema with the attributes partitioned into three classes separated by semicolons:*

$$\mathtt{R}(K_1, \ldots, K_n; A_1, \ldots, A_m; \mathbf{P})$$

*where $K = \{K_1, \ldots, K_n\}$ is called the **possible worlds key**, $A = \{A_1, \ldots, A_m\}$ is called the **value attribute set** and $\mathbf{P}$ is a single distinguished attribute called the **probability attribute**; its type is a value in the half-open interval $(0, 1]$.*

Semantics. Representations contain probability attributes ($\mathbf{P}$) but the worlds they represent do not. For example, a BID table description $\mathtt{R}(K; A; \mathbf{P})$ with distinguished attribute $\mathbf{P}$ corresponds to a **BID symbol** $\mathtt{R}^p(K; A)$ with no attribute $\mathbf{P}$[1]. A representation **yields** a distribution on *possible worlds* over *BID symbols*.

DEFINITION 2.2. *Given an instance of a BID table description $\mathtt{R}(K; A; \mathbf{P}) = \{t_1^r, \ldots, t_n^r\}$, a **possible world** is a subset of tuples, I, without the attribute $\mathbf{P}$ that satisfies the key constraint $K \to A$. Let $\mathbf{AK}(I) = \{k \mid \exists i\, t_i^r[K] = k \wedge \forall t \in I\ t_i^r[K] \neq t[K]\}$. The probability of possible world I, denoted $\mu(I)$, is defined as:*

$$\mu(I) = \overbrace{\Big(\prod_{i=1:t_i^r[KA]\in I}^{n} t_i^r[\mathbf{P}]\Big)}^{Present\ Tuples} \underbrace{\Big(\prod_{k\in\mathbf{AK}(I)} \Big(1 - \sum_{j=1:t_j^r[K]=k}^{n} t_j^r[\mathbf{P}]\Big)\Big)}_{Absent\ Tuples}$$

Informally, Def. 2.2 says three things: The marginal probability of each tuple, $\mu(t)$, satisfies $\mu(t) = t^r[\mathbf{P}]$, any set of tuples, $t_1, \ldots, t_m$ with distinct keys are independent (i.e. $\mu(t_1 \wedge \cdots \wedge t_m) = \prod_{i=1}^m \mu(t_i)$) and any two distinct tuples s, t that share the same *possible worlds key* are disjoint, $\mu(s \wedge t) = 0$.

We refer to a schema that contains both deterministic and BID table descriptions as a **BID schema** and an instance of a BID schema as a **BID instance**.

DEFINITION 2.3. *A possible world I for a BID instance $\mathfrak{I}$ with $\mathtt{R}_1, \ldots \mathtt{R}_n$ consists of a n possible worlds $I_1, \ldots, I_n$ with measures $\mu_1, \ldots, \mu_n$, one for each $\mathtt{R}_i$, and its probability is defined as $\mu_{\mathfrak{I}}(I) = \prod_{i=1}^n \mu_i(I_i)$, we will simply write μ when $\mathfrak{I}$ is clear from the context.*

DEFINITION 2.4 (CONJUNCTIVE VIEW). *A view is a conjunctive query of the form:*

$$V^p(\vec{h}) :- g_1, \ldots, g_n \tag{1}$$

*where each g_i is a **subgoal**, that is either a BID symbol (e.g. $\mathtt{R}^p$) in which case we say g_i is probabilistic or a deterministic table (e.g. $\mathtt{S}$). The set of variables in V^p is denoted $\mathbf{var}(V^p)$ and list of the head variables $\vec{h} \subseteq \mathbf{var}(V^p)$.*

The natural schema associated with $V^p(\vec{h})$ is the set of head variables denoted by H. For example, a view with head $V^p(x, y, x)$ has $\vec{h} = (x, y, x)$ while $H = \{x, y\}$.

[1]To emphasize this distinction, we will use a typewriter faced symbol with a superscripted p (e.g. $\mathtt{R}^p(K; A)$) to denote the BID symbol corresponding to the BID table description $\mathtt{R}(K; A; \mathbf{P})$. For deterministic tables, this distinction is immaterial.

Chef	Restaurant	**P**	
TD	D. Lounge	0.9	(w_1)
TD	P. Kitchen	0.7	(w_2)
MS	C. Bistro	0.8	(w_3)

W(Chef,Restaurant;;**P**) (WorksAt)

Restaurant	Dish
D. Lounge	Crab Cakes
P. Kitchen	Crab Cakes
P. Kitchen	Lamb
C. Bistro	Fish

S(Restaurant,Dish) (Serves)

Chef	Dish	Rating	**P**	
TD	Crab Cakes	High	0.8	(r_{11})
		Med	0.1	(r_{12})
		Low	0.1	(r_{13})
TD	Lamb	High	0.3	(r_{21})
		Low	0.7	(r_{22})
MS	Fish	High	0.6	(r_{31})
		Low	0.3	(r_{32})

R(Chef,Dish;Rating;**P**) (Rated)

Figure 1: Sample Restaurant Data in Alice's Database. In WorksAt each tuple is *independent*. There is no uncertainty about Serves. In Rated, each (Chef,Dish) pair has one true rating. Note that there may be more than one uncertain attribute as described in Def. 2.1.

DEFINITION 2.5 (VIEW SEMANTICS). *Given a view $V^p(H)$, the marginal probability of a tuple t in the output of V^p is denoted $\mu(V^p(t))$ and satisfies:*

$$\mu(V^p(t)) = \sum_{I:I \models V^p(t)} \mu(I)$$

The output of V^p on the representation $\mathfrak{I}$ is a set tuples with marginal probabilities denoted $\mathcal{O}$ and defined by $\mathcal{O} = \{(t,p) \mid \mu_{\mathfrak{I}}(V^p(t)) = p > 0\}$, we denote the marginal probability of $t \in \mathcal{O}$ as $t[\mathbf{P}]$.

2.2 Running Example

Sample data is shown in Fig. 1 for Alice's schema that contains three relations described in BID syntax: W (WorksAt), S (Serves) and R (Rating). The relation W records chefs, who may work at multiple restaurants in multiple cities. The tuples of W are extracted from text and so are uncertain. For example, ('TD', 'D. Lounge') (w_1) in W signifies that we extracted that 'TD' works at 'D.Lounge' with probability 0.9. Our syntax tells us that all tuples are independent because they all have different possible worlds keys. The relation R records the rating of a chef's dish (e.g. 'High' or 'Low'). Each (Chef,Dish) pair has only one true rating. Thus, r_{11} and r_{13} are *disjoint* because they rate the pair ('TD', 'Crab Cakes') as both 'High' and 'Low'. Because distinct (Chef,Dish) pair ratings are extracted independently, they are associated to ratings independently.

Semantics. In W, there are 2^3 *possible worlds*. For example, the probability of the singleton subset {('TD', 'D. Lounge')} is $0.9 * (1 - 0.7) * (1 - 0.8) = 0.054$. This representation is called a p-?-table [25], ?-table [20] or tuple independent [18]. The BID table R yields $3 * 2 * 3 = 18$ possible worlds since each (Chef,Dish) pair is associated with at most one rating. When the probabilities shown sum to 1, there is at least one rating for each pair. For example, the probability of the world $\{r_{11}, r_{21}, r_{31}\}$ is $0.8 * 0.3 * 0.6 = 0.144$. R is a slight generalization of p-or-set-table [25] or x-table [20].

2.2.1 Representable Views

Alice wants to ship her data to Bob, who wants a view with all chefs and restaurant pairs that make a highly rated dish. Alice obliges by computing and sending the following view:

$$V_1^p(c,r) :- \mathtt{W}^p(c,r), \mathtt{S}(r,d), \mathtt{R}^p(c,d;\text{'High'}) \qquad (2)$$

In the following example data, we calculate the probability of a tuple appearing in the output both numerically in the **P** column and symbolically in terms of other tuple probabilities of Fig. 1. We calculate the probabilities symbolically only for exposition; the output of a query or view is a set of tuples matching the head with associated probability scores.

EXAMPLE 2.1 (OUTPUT OF V_1^p FROM EQ. (2)).

	C	R	**P**	*(Symbolic Probability)*
t_1^o	*TD*	*D. Lounge*	0.72	$w_1 r_{11}$
t_2^o	*TD*	*P.Kitchen*	0.602	$w_2(1-(1-r_{11})(1-r_{21}))$
t_3^o	*MS*	*C.Bistro*	0.32	$w_3 r_{31}$

The output tuples, t_1^o and t_2^o, are *not* independent because both depend on r_{11}. This lack of independence is problematic for Bob because a BID instance cannot represent this type of correlation. Hence we say V_1^p is not a **representable view**. For Bob to understand the data, Alice must ship the lineage of each tuple. For example, it would be sufficient to ship the symbolic probability polynomials in Ex. 2.1.

Consider a second view where we can be much smarter about the amount of information necessary to understand the view. In V_2^p, Bob wants to know which working chefs make and serve a highly rated dish:

$$V_2^p(c) :- \mathtt{W}^p(c,r), \mathtt{S}(r,d), \mathtt{R}^p(c,d;\text{'High'}) \qquad (3)$$

EXAMPLE 2.2 (OUTPUT OF V_2^p FROM EQ. (3)).

c	**P**	*(Symbolic Probability)*
TD	0.818	$r_{11}(1-(1-w_1)(1-w_2)) + (1-r_{11})(w_2 r_{21})$
MS	0.48	$w_3 r_{31}$

Importantly, Bob can understand the data in Ex. 2.2 with no auxiliary information because the set of events contributing to 'TD' and those contributing to 'MS' are *independent*. It can be shown that over any instance, all tuples contributing to distinct c values are independent. Thus, V_1^p is a **representable view**. This motivates us to understand the following fundamental question:

PROBLEM 1 (VIEW REPRESENTABILTY). *Given a view V^p, can the output of V^p be represented as a BID table?*

It is interesting to note that efficiency and representability are distinct concepts: Computing the output of V_1^p can be done in polynomial time but its result is not representable. On the other hand, computing V_2^p can be shown to be #P-Hard [18, 35], but V_2^p is *representable*. To process V_2^p, we must use Monte Carlo procedures (e.g. [36]), which are orders of magnitude more expensive than traditional SQL processing. We do not discuss evaluation further because our focus is not on efficiently evaluating views, but on representing the output of views.

2.2.2 Partially Representable Views

To optimize and share a larger class of views, we would like to use the results of views that are not *representable*. The output of a non-representable view still has meaning: It contains the marginal probabilities that each tuple appears in the view but may not describe how all tuples correlate. Consider the output of V_1^p in Ex. 2.1: Tuples that agree on c are correlated, but tuples with different c values are independent. To capture this, we define a **partially representable view** (def. 5.3) which has schema $V^p(K_I; D; A)$. The intuition is that tuples that differ on K_I are guaranteed to be independent, and distinct tuples which agree on $K_I D$ are guaranteed to be disjoint. Tuples that agree on K_I but not on D may be correlated.

EXAMPLE 2.3. *Recall the view V_1^p in Eq. (2), whose natural schema is $V^p(C, R)$. It is partially representable as $V_1^p(C; R; \emptyset)$ (i.e. $K_I = \{C\}$, $D = \{R\}$ and $A = \emptyset$). Tuples that differ on C are independent, but those that agree on C but differ on R may be correlated in unknown ways. Thus, the materialized view does have some meaning for Bob, but does not contain sufficient information to completely determine a probability distribution on its possible worlds.*

Trivially, any view $V^p(H)$ can be partially represented by letting $K_I = A = \emptyset$ and $D = H$. Thus the interesting question is: What is the complexity to decide, for a given K_I, D, A, if a view V^p is *partially representable*?

2.2.3 Using Views to Answer Queries

In an information exchange setting, we do not have access to the base data and so for partially representable views may not know how all tuples are correlated. Thus, to answer a query Q, we need to ensure that the value of Q does not depend on correlations that are not captured by the partial representation. To illustrate, we attempt to use the output of V_1^p, a partially representable view, to answer two queries.

EXAMPLE 2.4. *Suppose Bob has a local relation, $\mathtt{L}^p(d; r)$, where d is a possible worlds key for $\mathtt{L}$. Bob wants to answer the following queries:*

$$Q_u(d) :- \mathtt{L}^p(d; r), V_1^p(c, r) \text{ and } Q_n(c) :- V_1^p(c, r)$$

Since the materialized view V_1^p does not uniquely determine a probability distribution on its possible worlds, it is not immediately clear that Bob can answer these queries without access to Alice's database and to V_1^p's definition. However, the query Q_u is uniquely defined. For any fixed d_0, the set of r_i values such that $\mathtt{L}^p(d_0; r_i)$ partition the possible worlds:

$$\mu(Q_u(d_0)) = \sum_{r_i : I \models \mathtt{L}^p(d_0; r_i)} \mu(\mathtt{L}^p(d_0; r_i) \wedge \exists c\ V_1^p(c, r_i))$$

The partial representation $V_1^p(C; R; \emptyset)$, tells us that distinct values for c in V_1^p are independent. Thus, each term in the summation is uniquely defined implying Q_u is uniquely defined. In contrast, Q_n is not uniquely defined because Q_n('TD') is true when either of t_1^o or t_2^o are present; our partial representation does not capture the correlation of the pair (t_1^o, t_2^o).

This example motivates the following problem:

PROBLEM 2 (VIEW ANSWERING). *Let V^p be a partially representable view. Given a query Q using V^p, is the value of Q uniquely defined?*

If V^p is representable, then this problem is trivial because V^p uniquely defines a probability distribution.

3. PRELIMINARIES

We define notations we need in the remainder of the paper.

DEFINITION 3.1. *A **valuation** v for a view V^p is a mapping from variables and constants in V^p to constants which is identity on constants ([2]). Given a valuation we let* $\mathbf{im}(v)$ *denote the image of v.*

We define a subset of valuations called **disjoint aware valuations** which associate each possible worlds key value to exactly one attribute value. Recall the definition of a view in Eq. (1); if g_i is probabilistic, we let $\vec{k}_i$ (resp. $\vec{a}_i$) denote the list of variables or constants in the possible world key attribute positions (resp. value attribute positions)[2]. For a subgoal g_i, $\mathbf{pred}(g_i)$ denotes the predicate associated to g_i.

DEFINITION 3.2 (DISJOINT AWARE VALUATION). *A valuation, v, for a conjunctive view is **disjoint aware** if $\forall i, j$*

$$\mathbf{pred}(g_i) = \mathbf{pred}(g_j) \text{ and } v(\vec{k}_i) = v(\vec{k}_j) \implies v(\vec{a}_i) = v(\vec{a}_j)$$

EXAMPLE 3.1 (DISJOINT AWARE VALUATIONS).

$$V_4^p() :- \mathtt{R}^p(x, y; \text{'High'}), \mathtt{R}^p(x, z; \text{'Low'}) \tag{4}$$

Any valuation v such that $v(y) \neq v(z)$ is disjoint aware but if $v(y) = v(z)$, it is not.

We need to consider pairs of valuations that do not use disjoint events, called **compatible valuations**.

DEFINITION 3.3. *A pair of disjoint aware valuations (v, w) for a view V^p is called **compatible** if the pair satisfies $\forall i, j$:*

$$\mathbf{pred}(g_i) = \mathbf{pred}(g_j) \text{ and } v(\vec{k}_i) = w(\vec{k}_j) \implies v(\vec{a}_i) = w(\vec{a}_j)$$

3.1 Generalizing Functional Dependencies

We consider three notions of functional dependencies over variables in the body of the view: standard functional dependencies, denoted $V^p \models \vec{a} \to \vec{b}$, representation functional dependencies, denoted $V^p \models \vec{a} \to_r \vec{b}$, and possible worlds dependencies, denoted $V^p \models \vec{a} \to_p \vec{b}$.

DEFINITION 3.4. *For a pair of valuations (v, w) for V^p, let $\dagger(v, w)$ denote the property:*

$$v(\vec{a}) = w(\vec{a}) \implies v(\vec{b}) = w(\vec{b})$$

Then we write:

- $V^p \models \vec{a} \to \vec{b}$ *if $\dagger(v, w)$ for any valuations.*
- $V^p \models \vec{a} \to_r \vec{b}$ *if $\dagger(v, w)$ for any disjoint aware valuations.*
- $V^p \models \vec{a} \to_p \vec{b}$ *if $\dagger(v, w)$ for any compatible valuations.*

[2]We assume deterministic tables do not have keys.

Algorithm 1 Decision Procedure for $V^p \models \vec{a} \rightarrow_p \vec{b}$

Input: $V^p(\vec{h}) :- g_1, \ldots, g_m$ and
$\vec{a}, \vec{b} \subseteq \mathbf{var}(V^p) \cup \mathbf{const}(V^p)$

Output: `'Yes'` iff $V^p \models \vec{a} \rightarrow_p \vec{b}$ else `'No'`

1: η is a function that is identity on $\vec{a}$ and $\mathbf{const}(V^p)$ and maps all other variables to distinct fresh variables.
2: $VV(\vec{b}', \eta(\vec{b})) :- g_1, \ldots, g_m, \eta(g_1), \ldots, \eta(g_m)$
3: Let $VV'(\vec{b}, \vec{b}') = \mathbf{DJA}(\ VV(\vec{b}, \eta(\vec{b}))\ )$
4: **if** Chase Succeeds **and** $\vec{b} \neq \vec{b}'$ **then**
5: **return** `'No'` (* $V^p \not\models \vec{a} \rightarrow_p \vec{b}$ *)
6: **else**
7: **return** `'Yes'` (* $V^p \models \vec{a} \rightarrow_p \vec{b}$ *)

$V^p \models \vec{a} \rightarrow \vec{b}$ if and only if $\vec{b} \subseteq \vec{a}$ as sets of variables[3]. To see how these definitions relate, it is immediate that:

$$V^p \models \vec{a} \rightarrow \vec{b} \implies V^p \models \vec{a} \rightarrow_r \vec{b} \implies V^p \models \vec{a} \rightarrow_p \vec{b}$$

We show by example that both reverse implications fail:

EXAMPLE 3.2. *Consider a BID table* $\mathtt{U}(K; A; \mathbf{P})$ *and a deterministic binary table* $\mathtt{D}(B, C)$.

$$V_5^p(x, y, z) :- \mathtt{U}^p(x; y), \mathtt{U}^p(x; z), \mathtt{D}(y, z) \tag{5}$$

Here $V_5^p \not\models xy \rightarrow z$ *but* $V_5^p \models xy \rightarrow_r z$ *since, any disjoint aware valuation for* V_5 *must satisfy* $v(y) = v(z)$.

$$V_6^p() :- \mathtt{U}^p(x; y) \tag{6}$$

In this case, $V_6^p \not\models x \rightarrow_r y$ *but* $V_6^p \models x \rightarrow_p y$.

3.2 A Chase Procedure

Consider a view with body $\mathtt{U}^p(x; y), \mathtt{U}^p(x; z), \mathtt{T}^p(y; u), \mathtt{T}^p(z; v)$. Clearly, any disjoint aware valuation must equate y and z. Hence, when evaluated over any possible world the query is equivalent to $\mathtt{U}^p(x; y), \mathtt{U}^p(x; y), \mathtt{T}^p(y; u), \mathtt{T}^p(y; v)$, where we have equated $y = z$. We can repeat the argument to equate u and v; this process is called a *chase* [22]:

PROPOSITION 3.1. *There is a polynomial time procedure that takes as input a conjunctive view* V^p *and produces as output a view* $\mathbf{DJA}(V^p)$ *(**D**is**J**oint **A**ware version of* V^p*) and surjective homomorphism* θ *from* V^p *to* $\mathbf{DJA}(V^p)$ *such that* $\mathbf{DJA}(V^p)$ *is equivalent to* V^p *over all possible worlds and the identity homomorphism on* $\mathbf{DJA}(V^p)$ *is disjoint aware. The Chase may fail if no such view exists.*

EXAMPLE 3.3. *Consider* V_5^p *from Ex. 3.2, then* [4]

$$\mathbf{DJA}(V_5^p) = W(x, y, y) :- \mathtt{U}^p(x; y), \mathtt{D}(y, y)$$

The chase equates y *and* z *because both have possible worlds key* x*. In contrast, the chase will fail on*

$$V(c, r) :- \mathtt{R}^p(c, r; \text{'High'}), \mathtt{R}^p(c, r; \text{'Low'})$$

The chase cannot unify the constants 'High' and 'Low'.

We show how to decide $V^p \models \vec{a} \rightarrow_r \vec{b}$ and $V^p \models \vec{a} \rightarrow_p \vec{b}$.

[3]This fails when we add dependencies in Sec. 4.3.
[4]We are *not* doing minimization, only removing duplicates.

PROPOSITION 3.2. *If* $\mathbf{DJA}(V^p)$ *exists, let* θ *be the chase homomorphism, then the following holds:*

$$\mathbf{DJA}(V^p) \models \theta(\vec{a}) \rightarrow \theta(\vec{b}) \iff V^p \models \vec{a} \rightarrow_r \vec{b}$$

We use Prop. 3.2 to decide $V^p \models \vec{a} \rightarrow_r \vec{b}$. For example, $V_5^p \models xy \rightarrow_r z$ Eq. (5) because, the chase homomorphism θ is given by $\theta(x, y, z) = (x, y, y)$. Thus, $\theta(z) \subseteq \theta(xy)$.

We use Alg. 1 to efficiently decide $V^p \models \vec{a} \rightarrow_p \vec{b}$.

PROPOSITION 3.3. *Algorithm 1 is a polynomial time sound and complete algorithm to decide if* $V^p \models \vec{a} \rightarrow_p \vec{b}$.

PROOF SKETCH. Observe that if the chase fails there must be some contradiction in the view so there are no disjoint aware valuations for V^p implying $V^p \models \vec{a} \rightarrow_p \vec{b}$ trivially holds. If the chase succeeds, then by Prop. 3.1, VV' has the property that the identity valuation is disjoint aware. Thus, if our test outputs `'No'`, there is a disjoint aware valuation vv for VV^p such that $vv(\vec{a}) = vv(\vec{a})$ but $vv(\vec{b}) \neq vv(\vec{b}')$. Let v (resp. w) be the restriction of vv to $g_1, \ldots, g_m$ (resp. $\eta(g_1), \ldots, \eta(g_n)$). More precisely, (v, w) is a *compatible* pair of valuations for V^p such that $v(\vec{a}) = w(\vec{a})$ but $v(\vec{b}) \neq w(\vec{b})$. Thus, $V^p \not\models \vec{a} \rightarrow_p \vec{b}$. The reverse direction and efficiency of the procedure follow directly from the Chase and Prop. 3.1. □

EXAMPLE 3.4. *Consider the view:*

$$V_7^p() :- \mathtt{R}^p(x, y; u), \mathtt{U}^p(x; z), \mathtt{T}^p(x, z; u) \tag{7}$$

We want to check $V_7^p \models x \rightarrow_p u$*. Alg. 1 forms* VV *by making copies of* V_7^p *and equating* x *in the copies as follows:*

$$VV_7^p(u, u') :- \mathtt{R}^p(x, y\ ; u), \mathtt{U}^p(x; z), \mathtt{T}^p(x, z\ ; y), \mathtt{R}^p(x, y'; u'), \mathtt{U}^p(x; z'), \mathtt{T}^p(x, z'; y')$$

The Chase first uses $K_{\mathtt{U}} \rightarrow A_{\mathtt{U}}$ *to derive that* $z = z'$*, then uses* $K_{\mathtt{T}} \rightarrow A_{\mathtt{T}}$ *to force* $y = y'$ *and finally,* $K_{\mathtt{T}} \rightarrow A_{\mathtt{T}}$ *to make* $u = u'$*. Thus, the algorithm says* `'Yes'`*. If we drop any subgoal, we can no longer derive* $u = u'$ *and so the algorithm will say* `'No'`*.*

4. PROBLEM 1: REPRESENTABILITY

The goal of this section is to give a solution to Problem 1, deciding if a view is representable, when the views are described by conjunctive queries and the representation formalism is BID. Since representability is a property of a view on an infinite family of representations, it is not immediately clear that the property is decidable. Our main result is that testing representability is decidable and is Π_2^P-Complete in the size of the view definition. The high complexity motivates us to give an efficient sound (but not complete) test in Sec. 4.2. For the important special case when all probabilistic symbols used in the view definition are distinct, we show that this test is complete as well. We then discuss how our technical result relates to prior art in Sec. 4.3.

4.1 Statement of Main Results

There are two key properties of BID representations: Tuples that differ on a possible worlds key are independent, which we will call **block independent**, and distinct tuples that share a possible worlds key must be disjoint, which we call **disjoint in blocks**.

DEFINITION 4.1. *Given a view with schema $V^p(H)$ defined in terms of BID symbols, $K \subseteq H$, we say V^p is K-**block independent** if for any BID instance $\mathfrak{I}$, denoting $\mathcal{O}$ as the output of V^p on $\mathfrak{I}$ (Def. 2.5), $\forall I \subseteq \mathcal{O}$ satisfying $\forall s,t \in I\ s[K] = t[K] \implies s = t$, the following holds:*

$$\mu(\bigwedge_{s\in I} V^p(s[H])) = \prod_{s\in I} s[\mathbf{P}]$$

*For $K' \subseteq H$, we say $V^p(K', H-K')$ is K'-**disjoint in blocks** if for $i = 1,2$, $\mu(V^p(t_i)) > 0$, $t_1[K'] = t_2[K']$ and $t_1 \neq t_2$ then:*

$$\mu(V^p(t_1[H]) \wedge V^p(t_2[H])) = 0$$

*We say a view V^p is **representable** if there is some K such that V^p is K-block independent and K-disjoint in blocks.*

In other words, V^p is *representable*, i.e. K-block independent and K-disjoint in blocks, if and only if we can represent the output of V^p as a BID table with BID table description $\mathtt{V}(K; H-K; \mathbf{P})$. Deciding if the preceding definition holds for a view is a formal definition of the *view representability* problem. We first consider $V^p(H)$ and $K \subseteq H$ as given and return to the problem of deducing K from the view definition in Sec. 4.1.3.

4.1.1 Block Independence

Intuitively, two tuples in a view are *not* independent if their value depends on two tuples with the same possible worlds key value.

DEFINITION 4.2. *A tuple t is **disjoint critical** for a Boolean view $V^p()$ if and only if there exists a possible world I such that $I \models V^p()$ and $I - \{t\} \not\models V^p()$. A pair of tuples (s,t) each with the same arity as a probabilistic BID symbol $\mathtt{R}^p_i(K_i; A_i)$ such that $s[K_i] = t[K_i]$ is K-**doubly critical** for a view V^p if $\exists s^o, t^o$ such that $s^o[K] \neq t^o[K]$ and s (resp. t) is **disjoint critical** for $V^p(s^o)$ (resp. $V^p(t^o)$).*

In the above definition, it is important to note that that we do not require that s and t be different tuples, only that they agree on the possible worlds key of some probabilistic relation.

EXAMPLE 4.1. *Recall from Ex. 2.1 that the view V^p_1 returns three tuples $\{t^o_1, t^o_2, t^o_3\}$. For each $t \in \{t^o_1, t^o_2, t^o_3\}$, the tuples referenced by the symbolic probability for $V^p_1(t)$ are* disjoint critical *for $V^p_1(t)$. For example, r_{11} is disjoint critical for $V^p_1(t^o_2)$ because we can take $I = \{w_2, r_{11}, \mathtt{S}(\text{'D.Lounge'}, \text{'Crab Cakes'})\}$ and $I \models V^p_1(t^o_2)$ but $I - \{r_{11}\} \not\models V^p_1(t^o_2)$. Further, r_{11} is also critical for $V^p_1(t^o_1)$. Since $t^o_1[CR] \neq t^o_2[CR]$, the pair (r_{11}, r_{11}) is an example of a CR-*doubly critical tuple. *Interestingly, there are no C-*doubly critical tuples.

LEMMA 4.1. *Given a view $V^p(H)$ and $K \subseteq H$, there are no K-doubly critical tuples if and only if V^p is K-block independent.*

The proof of this lemma requires a detailed examination of the multilinear polynomials produced by a view on a probabilistic instance and we leave it for the full paper [37]. Lem. 4.1 is the basis for Alg. 2, which decides K-block independence by looking for K-doubly critical tuples.

Algorithm 2 K-Block Independence

Input: A conjunctive view $V^p(H)$ and $K \subseteq H$
Output: `'Yes'` iff V^p is K-Block Independent

1: Let $n = |\mathbf{var}(V^p)|$, $C = \{c_1, \ldots, c_{n^2}\}$ be fresh constants
2: Let $\vec{h}$ denote head variables, $\vec{k}$ variables at positions K
3: $\mathbf{D} = \{u | u$ disjoint aware valuation for V^p s.t. $\forall x \in \mathbf{var}(V^p)\ u(x) \in C \cup \mathbf{const}(V^p)\}$.
4: **if** $\forall v, w \in \mathbf{D}, \forall s \in \mathbf{im}(v), \forall t \in \mathbf{im}(w)$. $v(\vec{k}) \neq w(\vec{k})$, $s \in v(\mathtt{R}^p_i)$, $t \in w(\mathtt{R}^p_i)$ **and** $s[K_i] = t[K_i] \implies$ $\mathbf{im}(v) - \{s\} \models V^p(v(\vec{h}))$ **or** $\mathbf{im}(w) - \{t\} \models V^p(w(\vec{h}))$
5: **then return** `'Yes'` **else** `'No'`

EXAMPLE 4.2 (EX. 4.1 CONTINUED). *In Ex. 4.1, we observed that there are no C-doubly critical tuples for V^p_1, which implies that V^p_1 is C-block independent. Also, we observed that V^p_1 is not CR-block independent, because of r_{11}, a CR-doubly critical tuple.*

Algorithm Lemma 4.1 gives us the following test for checking k-block independence: *for every two instances I, J, every tuples $s \in I, t \in J$, and every two output tuples $I \models V^p(s^o)$, $J \models V^p(t^o)$ s.t. $s^o[K] \neq t^o[K]$, check that $I - \{s\} \models V^p(s^o)$ or $J - \{t\} \models V^p(t^o)$.* This is not yet an algorithm, because I and J range over infinitely many instances. However, we can prove that it suffices to range over instances consisting of the constants in the view plus at most n^2 additional constants. This leads us to Algorithm 2, and also proves that the problem is in Π^P_2. In addition, we can also prove that the problem is hard for Π^P_2 (by reduction from $\forall\exists$ `3CNF`), hence:

THEOREM 4.1. *Algorithm 2 is sound and complete. Further, checking that no K-doubly critical exists for a conjunctive view is Π^P_2-Complete.*

4.1.2 Disjoint in Blocks

Having established a test for block independence, we now state how to decide if a query is disjoint within blocks. The idea here is simple: A view fails to be K-disjoint within blocks if and only if there exist distinct tuples which agree on K but can occur in some possible world together. We give a polynomial time algorithm based on a Chase (Sec. 3.2) and the following lemma:

LEMMA 4.2. *Given a conjunctive view $V^p(H)$ and $K \subseteq H$ then $V^p \models K \rightarrow_p H$[5] if and only if V^p is K-disjoint in blocks.*

To see the forward direction, consider any two tuples s, t which disagree on K. It must be the case that every valuation such that $v(\vec{h}) = s[H]$ and $w(\vec{h}) = t[H]$ use at least one tuple that is disjoint else $V^p \not\models K \rightarrow_p H$. To see the reverse direction, observe that if (v, w) is compatible then $\mathbf{im}(v) \cup \mathbf{im}(v) = I$ satisfies the constraints and is a possible world. Hence, s and t are both answers to V^p on I, which is a contradiction to our assumption that V^p is disjoint in blocks.

THEOREM 4.2. *Algorithm 3 is a sound and complete `PTIME` algorithm to decide given V^p, K and H, if $V^p \models K \rightarrow_p H$ and hence if V^p is K-disjoint in blocks*

[5]We use $V^p \models K \rightarrow_p H$ to mean $V^p \models \vec{k} \rightarrow_p \vec{h}$ where $\vec{k}$ ($\vec{h}$) is the list of variables and constants at K (resp. H).

Algorithm 3 K-Disjoint in Blocks

Input: $V^p(H)$ and $K \subseteq H$

1: **return** $V^p \models K \rightarrow_p H$ (* See Alg. 1 *)

EXAMPLE 4.3. *Consider the following view:*

$$V_8^p(d;r) :- \mathtt{L}^p(d;r), V_2^p(c,r) \quad (8)$$

*where $K = \{D\}$ and $A = \{R\}$. Any compatible pair of disjoint aware valuations that agree on d must agree on r, else they would be inconsistent. Thus, V_8^p is D-*disjoint in blocks*. To see a negative example, observe that V_1^p Eq. (2) is not C-disjoint in blocks because the pair of valuations, $v(c,r,d) =$ ('TD', 'D.Lounge', 'Crab Cakes') and $w(c,r,d) =$ ('TD', 'P.Kitchen', 'Crab Cakes'), is* compatible *and $v(c) = w(c)$ but $v(r) \neq w(r)$.*

4.1.3 Finding Possible Worlds Keys

In previous sections, we assumed that the BID schema for V^p was part of the input; we now consider how to infer the schema for V^p from its definition. Interestingly, we can efficiently find K such that if $V^p(K'; H-K')$ is representable for *any* K' then $V^p(K; H-K)$ is representable. Formally, we efficiently find a **candidate key** K for V^p.

DEFINITION 4.3. *K is a **candidate key** for V^p if V^p is representable if and only if V^p is K-block independent.*

The central observation to find a candidate K for a fixed V^p is the following:

PROPOSITION 4.1. *If V^p is K-disjoint in blocks and K'-block independent then $V^p \models K \rightarrow_r K'$.*

PROOF SKETCH. Suppose that $V^p \not\models K \not\rightarrow_r K'$ then there is a representation on which the output of V^p contains tuples s,t such that $s[K] = t[K]$ but $s[K'] \neq t[K']$, $s[\mathbf{P}] > 0$ and $t[\mathbf{P}] > 0$. Since s,t agree on K but, $s \neq t$ and V^p is K-disjoint in blocks this implies s,t are disjoint. On the other hand, they disagree on K' which since V^p is K'-block independent implies s,t are independent. Since a pair of events with positive probability cannot be both independent and disjoint; we reach a contradiction. □

This proposition says something interesting: Informally, up to $\rightarrow_r$ equivalence, there is a *unique* choice of K for which $V^p(K; H-K)$ can be representable. Since we can infer these dependencies in `PTIME` (Alg. 3), Prop. 4.1 suggests the efficient algorithm in Alg. 4.

Algorithm 4 Finding a candidate key for V^p

Input: V^p, a conjunctive view
Output: Candidate key K for V^p

1: $W^p(H_W) \leftarrow \mathbf{DJA}(V^p)$
2: $K \leftarrow H_W$
3: **for each** $A \in H$ **do**
4: **if** $V^p \models K - \{A\} \rightarrow_p H$ **then** (* see Alg. 1 *)
5: $K \leftarrow K - \{A\}$
6: **return** K (* K is a minimal possible worlds key *)

THEOREM 4.3. *When there are no functional dependencies in the representation, Algorithm 4 correctly finds a candidate key K.*

Algorithm 5 Finding a K-Collision for V^p

Input: $V^p(H) :- g_1, \ldots, g_n$ and K
Output: `'Yes'` iff V^p has a collision

1: **for each** $i,j \in 1, \ldots, n$ **do**
2: (* Make a fresh copy of V^p *)
$V_1^p(K,\ H-K)\ :-\ g_1, \ldots, g_n$
$V_2^p(K', H-K') :-\ g'_1, \ldots, g'_n$
3: **if** g_i is probabilistic **and pred**$(g_i) =$ **pred**(g'_j) **then**
4: Unify $g_i[K_i] = g'_j[K_j]$.
5: Let $W_1 \leftarrow \mathbf{DJA}(V_1), W_2 \leftarrow \mathbf{DJA}(V_2)$
6: **if** Chase Succeeds **and** $W_1[K] \neq W_2[K']$ **then**
7: **return** `'Yes'` (* There is a Collision. *)
8: **return** `'No'` (* There is no Collision. *)

Algorithm 6 Practical K-Block Independence

Input: $V^p(H)$ a conjunctive view and $K \subseteq H$
Output: `'Yes'` only if V^p is K-Block Independent
return `'Yes'` if $V^p(K; H-K)$ has no K-Collision.

To get an intuition for Thm. 4.3, we observe that the returned $K \subseteq H$ satisfies $V^p \models K \rightarrow_r K'$ for any representable $V^p(K'; H)$. Prop. 3.2 implies that the chase homomorphism, θ satisfies $\theta(K') \subseteq \theta(K)$. If $V^p(K; H-K)$ is not representable, we show that, $\theta(K') \subset \theta(K)$. This allows us to construct a strict subset of K, call it K_0, such that $V \models K_0 \rightarrow_p H$, which is a contradiction to K's minimality. In particular, take $K_0 = \theta^{-1}(K') \cap K$, which is valid because θ is surjective (Prop. 3.1).

4.1.4 A Solution for Problem 1

We have now established all the necessary ingredients to solve problem 1 for conjunctive views, which we summarize in the following theorem:

THEOREM 4.4. *Given a conjunctive view $V^p(H)$, deciding if there is some K such that output of V^p can be represented as a single BID relation $V(K; H-K; \mathbf{P})$ is decidable. Further, it is Π_2^P Complete.*

The algorithm first runs Alg. 4 which returns a candidate key K, which we use as input to Alg. 2.

4.2 Practical Algorithm for Representability

Since the intractable portion of the representability check is deciding K-block independence, we give a polynomial time approximation for K-block independence that is sound, i.e. it says a view is representable only if it is representable. However, it may not be complete, declaring that a view is not representable, when in fact it is. The central notion is a *$\vec{k}$-collision*, which intuitively says there are two output tuples which may depend on input tuples that are not independent (i.e. the same tuple or disjoint).

DEFINITION 4.4. *A **$\vec{k}$-collision** for a view*

$$V^p(\vec{k}, \vec{a}) :- g_1, \ldots, g_n$$

is a pair of disjoint aware valuations (v,w) such that $v(\vec{k}) \neq w(\vec{k})$ but there exists i,j such that g_i that is probabilistic, **pred**$(g_i) =$ **pred**(g_j) *and $v(\vec{k}_i) = w(\vec{k}_j)$.*

THEOREM 4.5. *For a view $V^p(H)$ and $K \subseteq H$, if algorithm 6 outputs 'Yes' then V^p is guaranteed to be K-block independent. Further, if V^p does not contain repeated probabilistic subgoals then algorithm 6 is complete. The algorithm is PTIME.*

When V^p does not contain repeated probabilistic subgoals the algorithm is complete because every probabilistic tuple in the image of a valuation must be critical. In particular, the image of g_i and g_j in the definition of collision are critical.

EXAMPLE 4.4. *Consider $V_2^p(C)$ in Eq. (3), if we unify any pair of probabilistic subgoals, we are forced to unify the head, c. This means that a collision is never possible and we conclude that V_2^p is C-block independent. Notice that we can unify the* S *subgoal for distinct values of c, since* S *is deterministic, this is not a collision. In $V_1^p(c,r)$ Eq. (2), the following pair (v,w), $v(c,r,d) =$ ('TD', 'D.Lounge','Crab Cakes') and $w(c,r,d) =$ ('TD', 'P.Kitchen', 'Crab Cakes'), is a collision because $v(c,r) \neq w(c,r)$ and we have unified the keys of the $\mathtt{R}^p$ subgoal. Since there are no repeated probabilistic subgoals, we are sure that V_1^p is not CR-block independent.*

4.3 Extensions and Discussion

Extensions. In a BID instance, tuples in distinct views must be *independent*. The following pair of views illustrates the problem:

$$V_x^p(x) :- \mathtt{T}^p(x,y,z;) \text{ and } V_y^p(y) :- \mathtt{T}^p(x,y,z;)$$

Each view is representable by itself. However, all tuples in T contribute to each view, so the pair of views is not representable. It is straightforward to extend our test to handle independence of tuples in distinct views and is left for the full paper. Additionally, we extend our results to handle dependencies in the representation in the full paper.

Relation to Query Evaluation. We have observed that efficient query evaluation for a view and representability are distinct concepts. To see this, observe that Thm. 4.1 shows that any single Boolean view is *representable*. Some Boolean queries have high complexity (#P) [18, 35]. When a query has a PTIME algorithm, it is called **safe**. This implies that not every representable view is safe. On the other hand, Ex. 2.1 gives an example of a non-representable view that has a safe plan. However, not all queries have safe plans, but for conjunctive queries there are efficient schemes to approximate probabilities to essentially any desired precision [36]. Using the result of approximation schemes for materialized view optimizations and providing error guarantees is an interesting open question.

Complex Correlations. The problem of K-Block Independence is to decide: For tuples s,t, is it the case that $\mu(V_1^p(s) \wedge V_2^p(t)) = \mu(V_1^p(s))\mu(V_2^p(t))$? In [32], a similar problem was studied where V_1^p is a secret query and V_2^p is a public view and our goal is to determine if the secret query and public view are independent. It was shown that this problem is Π_2^P Complete[6]. That work used a more restrictive tuple independent model in which the FKG inequality [3] $\mu(V_1^p(s) \wedge V_2^p(t)) \geq \mu(V_1^p(s))\mu(V_2^p(t))$ holds. Fig. 2 shows that this inequality no longer holds in our setting by showing a view V_9^p and family of representations such that tuples in

[6]However, there seems to be no direct reduction to our problem in the single view case.

K_1	A_1	**P**
c	H	c_H
	T	c_T

$\mathtt{M}_1(K_1; A_1; \mathbf{P})$

K_2	A_2	**P**
a	H	a_H
	T	a_T
b	H	b_H
	T	b_T

$\mathtt{M}_2(K_2; A_2; \mathbf{P})$

K_2	**P**
a	$a_H c_T + a_T c_H$
b	$b_H c_T + b_T c_H$

$V_9^p(k_2) :- \mathtt{M}_1^p(k_1; x), \mathtt{M}_2^p(k_2; x)$

	a_H	b_H	c_H	$\mu(V_9^p(a) \wedge V_9^p(b))$
(I)	0.5	0.5	0.5	0.25
(P)	0.9	0.9	0.5	0.41
(N)	0.9	0.1	0.5	0.09

On all three representations, $\mu(V_9^p(a))\mu(V_9^p(b)) = 0.25$.

Figure 2: Sample Data for Discussion. If all probabilities are 0.5, $V_9^p(a)$ and $V_9^p(b)$ appear to be independent. $l \in \{a,b,c\}$ $l_H = 1 - l_T$.

V_9^p are positively correlated, negatively correlated or even independent depending on how we set the probabilities in the representations. This technical difference is significant because the proof in [32] is an inductive argument that relies on the FKG inequality. Since the FKG inequality does not hold, we must use a completely different technique.

EXAMPLE 4.5. *Consider the family of representations given in Fig. 2. Consider the query:*

$$V_9^p(k_2) :- \mathtt{M}_1^p(k_1; x), \mathtt{M}_2^p(k_2; x) \tag{9}$$

The probabilities are described symbolically in the figure. Thus, $\mu(V_9^p(a) \wedge V_9^p(b)) = a_H b_H c_T + a_T b_T c_H$. In case (I), the tuples appear to be pairwise independent, but this does not hold for every distribution. For example in case (P) the two tuples are positively correlated, while in (N) they are negatively correlated. These correlations are possible even though V_9^p is a very simple conjunctive view. They are the result of the more sophisticated BID representation system, which allows disjoint events.

5. PROBLEM 2: QUERYING USING VIEWS

In this section, we study problem 2: Given a conjunctive query Q written using a materialized view V^p is the value of $\mu(Q)$ uniquely defined? Of course, if V^p is representable this problem is trivial: Q's value is always uniquely defined.

5.1 Partially Representable Views

In contrast to an ordinary probabilistic materialized view that represents a unique probability distribution, a partially represented view represents *many* probability distributions, each of which we call *agreeable*.

DEFINITION 5.1. *A **partial BID view description** is a relational schema with the attributes partitioned into four classes:*

$$V(K_I; D; A; \mathbf{P})$$

*where K_I is called the **independence key**, $K_I D$ is called the **disjointness key**, A is called the **value attribute set** and $\mathbf{P}$ is called the **probability attribute**, a distinguished attribute taking values in the half-open interval $(0,1]$.*

EXAMPLE 5.1. *Recall that $V_1^p(C,R)$ from Eq. (2) is not representable. We show that it is partially representable with syntax: $V_1(C; R; \emptyset; \mathbf{P})$.*

The intuition is that a partial representation preserves marginal probabilities but may not specify all correlations: If a set of tuples differ on K_I, they are independent. If two distinct tuples agree on $K_I D$, they are disjoint. However, if two tuples agree on K_I but disagree on D, they may be correlated in complicated ways.

DEFINITION 5.2 (SEMANTICS). *Given a view $V^p(H)$ and a partition K_I, D, A of H, we say V^p is **partially representable** if V^p is K_I-block independent and $K_I D$-disjoint in blocks.*

DEFINITION 5.3. *A possible world is a set of tuples, I, that satisfies $K_I D \rightarrow A$. We say a distribution on possible worlds, μ, **agrees** with $V^p(K_I; D; A)$ if for any set of tuples $I \subseteq \mathcal{O}$, the output of V^p on $\mathfrak{I}$ (Def. 2.5) such that $s, t \in I$, $s[K_I] = t[K_I] \implies s = t$ then*

$$\mu(\bigwedge_{t \in I} V^p(t)) = \prod_{t \in I} \mu(V^p(t))$$

In particular, if $D = \emptyset$, then Def. 5.3 coincides with Def 2.2 and so the partial representation uniquely defines a probability distribution. Any view has a trivial partial representation with $K_I = A = \emptyset$ and $D = H$. In the previous section, we showed that checking K-block independence is Π_2^P-Complete. Thus, the following is immediate:

THEOREM 5.1. *Given a conjunctive view V^p with head H and K, D, A such that their disjoint union is H i.e. $K \oplus D \oplus A = H$, deciding if the output of V^p is partially representable as $V(K; D; A; \mathbf{P})$ is Π_2^P-Complete.*

5.2 Statement of Main Results

Intuitively, a query's value fails to be uniquely defined if it depends on two tuples whose correlation is not specified by the partial representation. Due to space constraints, we present queries that use a single partially representable view.

DEFINITION 5.4. *A **critical pair** for a Boolean query $Q()$ is a pair of distinct tuples (s, t) such that there exists a possible world I satisfying*

$$I - \{s, t\} \not\models Q(), I \models Q() \text{ and } I - \{s\} \models Q() \iff I - \{t\} \models Q()$$

*Given a partially representable view $V^p(K_I; D; A)$, a pair of tuples (s, t) is called V^p-**intertwined** if $s, t \in V^p$ and $s[K_I] = t[K_I]$ but $s[D] \neq t[D]$.*

In contrast to K-doubly critical tuples, the possible world I must be the same for s, t.

EXAMPLE 5.2 (RUNNING EXAMPLE). *Consider the partial representation in Ex. 5.1 and the queries:*

$$Q_1() :- V_1^p(c, r) \text{ and } Q_2(c) :- V_1^p(c, \text{'D.Lounge'})$$

t_1^o and t_2^o are a critical pair of tuples for Q_1 and are V_1^p-intertwined. For any fixed c_0, there is no critical pair of tuples of V_1^p-intertwined tuples for $Q_2(c_0)$.

We state the link between intertwined tuples and distributions that agree with a view.

PROPOSITION 5.1. *Given a partially representable view $V^p(K_I; D; A)$, μ be a distribution that agrees with V^p and $s, t \in V^p$ that are V^p-intertwined such that $\mu(s) \neq 1$ and $\mu(t) \neq 1$ then there exists a distribution ν that agrees with V^p, such that $\mu(s \wedge t) \neq \nu(s \wedge t)$ and $\mu(s \vee t) \neq \nu(s \vee t)$.*

5.2.1 Critical Intertwined Captures Uniqueness

A query $Q()$ is **uniquely defined** if for any two agreeable distributions,μ, ν, we have $\mu(Q()) = \nu(Q())$. We establish that the existence of a critical pair of intertwined tuples captures when a query fails to be *uniquely defined*.

LEMMA 5.1. *There exist a critical pair of intertwined tuples for a conjunctive query $Q()$ if and only if $Q()$ is not uniquely defined.*

To see the forward direction consider a conjunctive query Q. If there is a pair of critical tuples (s, t) for $Q()$, then there are two cases: $I - \{s\} \models Q()$, in which case Q is satisfied when either of s, t are present, or $I - \{s\} \not\models Q()$, in which case $Q()$ is satisfied only when s and t are both present. Since I is a possible world, we can create a representation $\mathfrak{I}$ such that s, t are the only tuples with $\mu \neq 1$. For a possible world J of $\mathfrak{I}$, $J \models Q() \iff J \models s \wedge t$ or$J \models Q() \iff J \models s \vee t$, by Prop. 5.1 neither is uniquely defined. The reverse direction is an inductive proof that gives less information and is in the full paper.

EXAMPLE 5.3 (CONTINUING EX. 5.2). *A distribution, μ, that always agrees with V_1^p is the result of inlining of V_1^p in Q_1. Here $\mu(Q_1()) \approx 0.905$. A second distribution that agrees with V_1^p, ν, is to assume independence. Thus, $\nu(Q_1) = 1 - (1 - 0.72)(1 - 0.602)(1 - 0.32) \approx 0.924$. As we saw in Ex. 5.2, Q_1 does have a critical pair of intertwined tuples. On the other hand, for each c value, the query Q_2 is uniquely defined, in the example its value is* 0.72.

THEOREM 5.2. *Given a query Q using a partially representable view V^p, deciding if Q's value is uniquely defined is Π_2^p Complete.*

Let $n = |\mathbf{var}(Q)|$ and C be a set of n^2 fresh constants; a complete algorithm checks that for all possible worlds with domains in $\mathbf{const}(Q) \cup C$, there is not a critical pair of intertwined tuples; this algorithm is in Π_2^p.

5.3 Practical Test for Uniqueness

DEFINITION 5.5. *Given a schema with a single partially representable view V^p, an **intertwined collision** for a query $Q(H)$ is a pair of compatible valuations (v, w) such that $v(\vec{h}) = w(\vec{h})$ and there exists a pair of subgoals, (g_i, g_j), such that $\mathbf{pred}(g_i) = \mathbf{pred}(g_j) = V^p$, $v(\vec{ki}_i) = w(\vec{ki}_j)$ and $v(\vec{d}_i) \neq w(\vec{d}_j)$ where $\vec{ki}_i$ ($\vec{ki}_j$) is the list of variables at K_I in g_i (resp. g_j) and $\vec{d}_i$ ($\vec{d}_j$) is the list of variables at D in g_i (resp. g_j).*

The algorithm to find an intertwined collision is a straightforward extension of finding a K-collision. The key difference is that we use the Chase to ensure that the valuations we find are compatible, not individually disjoint aware.

THEOREM 5.3. *If no intertwined collisions exist for a conjunctive query Q, then its value is uniquely defined. If the partially representable view symbol V^p is not repeated, this test is complete. The test can be implemented in* `PTIME`.

PROOF SKETCH. We sketch the soundness argument in the special case of a Boolean query $Q()$. We show that if there exists a critical intertwined pair (s, t) for Q, then

there must be an intertwined collision. Let I be the instance provided by Def. 5.4. Suppose, $I - \{s\} \models Q()$. Since $I - \{s,t\} \not\models Q()$, the image of any valuation v that witnesses $I - \{s\} \models Q()$ must contain t. By symmetry, the image of any valuation that witnesses $I - \{t\} \models Q()$ must contain w. It is easy to see that (v, w) is compatible and hence (v, w) is an intertwined collision. If $I - \{s\} \not\models Q()$ then there is a single valuation v which uses both s, t. Thus, (v, v) is an intertwined collision. □

EXAMPLE 5.4. *In V_1^p, $K_I = \{C\}$ and $D = \{R\}$. An intertwined collision for Q_1 is $v(c,r) =$ ('TD', 'D.Lounge') and $w(c,r) =$ ('TD', 'P.Kitchen'), thus Q's value is not uniquely defined. On the other hand, in Q_2, trivially there is no intertwined collision and so Q_2's value is uniquely defined.*

5.4 Discussion

Optimization. In an optimizer, we would like *syntactic independence* [10], which is the ability to rewrite a query Q that does not use a materialized view V into an equivalent Q' that does use V. The same theory applies, but we must additionally check that Q' correctly uses a view as described in Sec. 5.2. A key difference in query optimization is that we usually have access to the view definitions. When the view definitions are present, a partial representation for a view essentially strips the view's lineage. If a query's value is uniquely defined, its value is the same as inlining the view definition. In the full paper, we show that deciding uniqueness in this setting is Π_2^p complete and give `PTIME` approximations.

Best Parital Representation. A view $V^p(H)$ may have many partial representations, which differ in how H is partitioned into K, D, A. If $V^p(K; D; A)$ and $V^p(K'; D'; A')$ are valid partial representations then there is a valid partial representation $V^p(K \cup K'; D'', A'')$ for some D'', A''. Thus, there is a single best (i.e. largest) choice of K, but this K is difficult to find (Thm. 5.1). In practice, a good choice for K is the intersection of the possible worlds key of each probabilistic subgoal. This choice is correct, i.e. there is some choice of D, A such that $V^p(K; D; A)$ is representable, but may there may be a larger K. Once we have chosen K, the next step is to choose D and A. Ideally, D should be empty and so V^p is representable, but this is not always possible. In general, there is no best choice for D, A (smallest D), e.g.

$$V^p(x, y, z) :- \mathtt{R}^p(x, y; z), \mathtt{S}^p(x, z; y)$$

This view is not representable, so $|D| \geq 1$. However, $|D| = 1$ is possible with either $V^p(x; y; z)$ or $V^p(x; z; y)$.

View Selection. Informally, the view selection problem [12] is to select given a set of queries $\mathcal{Q}$, the workload and a space budget B, choose a set of views $\mathcal{V}$ to materialize within the space budget B to minimize the cost of $\mathcal{Q}$. In the probabilistic setting, we now also check each $q \in \mathcal{Q}$ is uniquely defined using $\mathcal{V}$. The new twist is that the cost function has a large step: If a query Q can be executed using a *safe plan* [18, 35] over a view V, the cost of executing Q is dramatically lower.

6. RELATED WORK

Materialized views are a fundamental technique used to optimize queries [1, 10, 24, 27] and as a means to share, protect and integrate data [32, 40] that are currently implemented by all major database vendors. Because the complexity of deciding when a query can use a view is high, there has been a considerable amount of work on making query answering using views algorithms scalable [24, 34]. In the same spirit, we provide efficient practical algorithms for our representability problems.

Recently, probabilistic databases have received attention because of their ability to deal with uncertainty resulting from data cleaning tasks [4, 36], information extraction [9, 26] and sensor data [21, 29]. This has resulted in several systems [7, 21, 38, 41] with accompanying work on probabilistic query processing [11, 18, 35, 36, 39]. Prior art has considered using a representation system [20, 36, 38] that can represent every conjunctive view. Typically, these systems use base tables that are similar to BID representations but then introduce auxiliary information (e.g. lineage [41] or factors [38]) to track correlations introduced by query processing.

In prior art [20], the following question is studied: Given a class of queries $\mathcal{Q}$ is a particular representation formalism closed for all $Q \in \mathcal{Q}$? In contrast, our test is more fine-grained: For any fixed conjunctive Q, is the *BID* formalism closed under Q? Also relevant for expanding the class of practical algorithm is the recent work in [31].

7. EXPERIMENTS

In this section we answer three main questions: To what extent do representable and partially representable views occur in real and synthetic data sets? How much do probabilistic materialized views help query processing? How expensive are our proposed algorithms for finding representable views?

7.1 Experimental Setup

Data Description. We experimented with a variety of real and synthetic data sets including: a database from iLike.com [13], the Northwind database (NW) [14], the Adventure Works Database from SQL Server 2005 (AW)[15] and the TPC-H benchmark (TPCH) [16]. We manually created several probabilistic schemata based on the Adventure Works [15], Northwind [14] and TPC-H data which are described in Fig. 4.

Queries and Views. We interpreted all queries and views with scalar aggregation as probabilistic existence operators (i.e. computing the probability a tuple is present). iLike, Northwind and Adventure Works had predefined views as part of the schema. We created materialized views for TPC-H using an exhaustive procedure to find all subqueries that were representable, did not contain cross products and joined at least two probabilistic relations.

Real data: iLike.com. We were given query logs and the relational schema of iLike.com, which is interesting for three reasons: It is a real company, a core activity of iLike is manipulating uncertain data (e.g. similarity scores) and the schema contains materialized views. iLike's data, though not natively probabilistic, is easily mapped to a BID representation. The schema contains over 200 tables of which a handful contain uncertain information. The workload trace contains over 7 million queries of which more than 100,000 manipulated uncertain information contained in 5 views. Of these 100,000 queries, we identified less than 10 query types which ranged from simple selections to complicated many way joins.

Performance Data. All performance experiments use the TPC-H data set with a probabilistic schema containing uncertainty in the `part`, `orders`, `customer`, `supplier`

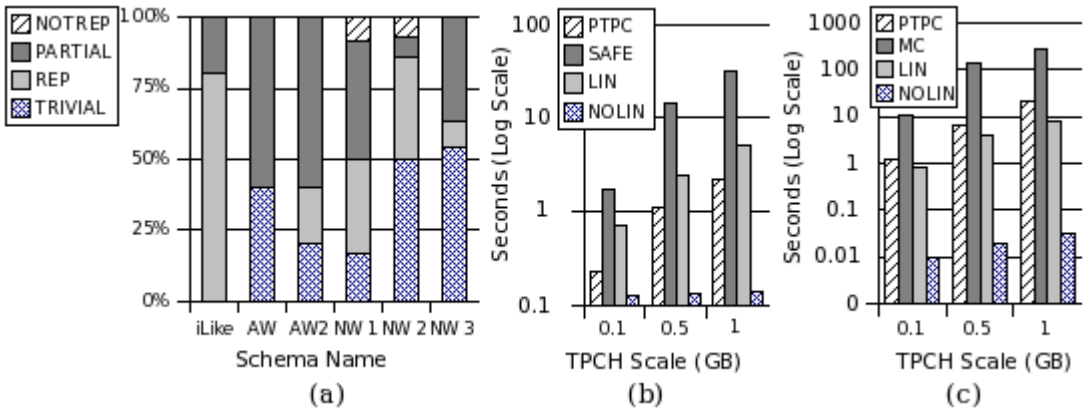

Figure 3: (a) Percentage by workload that are representable, non-trivially partially representable or not representable. We see that almost all views have some non-trivial partial representation. (b) Running times for Query 10 which is safe. (c) Retrieval times for Query 5 which is not safe. Performance data is TPC-H (0.1, 0.5, 1G) data sets. All running times in seconds and on logarithmic scale.

Schema	Tables (w/**P**)
AW	18 (6)
AW2	18 (3)
NW1	16 (2)
NW2	16 (5)
NW3	16 (4)
TPC-H	8 (5)

(a)

Size (w/**P**)	Tuples (w/**P**)
0.1 (440M)	3.3M (2.4M)
0.5 (2.1G)	16M (11.6M)
1.0 (4.2G)	32M (23.2M)

(b)

Figure 4: Schema and TPC Data statistics. (a) Number of tables referenced by at least one view and number of probabilistic tables (i.e. with attribute P). (b) Size and (w/P) are in Gb. The number of deterministic and probabilistic tuples is in millions.

and `lineitem` tables. We used the TPC-H tool `dbgen` to generate relational data. The data in each table marked as probabilistic was then transformed by uniformly at random injecting additional tuples such that each key value was expected to occur 2.5 times. We allowed for *entity uncertainty*, that is, the sum of probabilities for a possible worlds key may be less than 1.

System Details. Our experimental machine was a Windows Server 2003 machine running SQL Server 2005 with 4GB of RAM, 700G Ultra ATA drive and dual Xeon (3GHz) processors. The Mystiq engine is a middleware system that functions as a preprocessor and uses a complete approach [7, 36]. The materialized view tools are implemented using approximately 5000 lines of OCaml. After importing all probabilistic materialized views, we tuned the database using only the SQL Server Database Engine Tuning Advisor.

Execution Time Reporting Method. We reduced query time variance by executing each query seven times, dropping the highest and lowest times and averaging the remaining five times. In all reported numbers, the variance of the five runs was less than 5% of query execution time.

7.2 Question 1: Do Representable and Partially Representable views exist?

In Fig. 3(a), we show the percentage of views in each workload that is trivially representable because there are no probabilities in the view *(TRIVIAL)*, representable *(REP)*, non-trivially partially representable *(PARTIAL)* or only trivially partially representable *(NOTREP)*. In iLike's workload, 4 of the 5 views (80%) are representable. Further, 98.5% of the over 100k queries that manipulate uncertain data use the representable views. In synthetic data sets, representable views exist as well. In fact, 50% or more of the views in each data set except for AW are representable. Overall, 63% of views are representable. 45% of the representable views are non-trivially representable. Additionally, almost all views we examined have a non-trivial partial representations (over 95%). We conclude that that representable and partially representable views exist and can be used in practice.

7.3 Question 2: Do our techniques make query processing more efficient?

The TPC data set is the basis for our performance experiments because it is reproducible and the data can be scaled arbitrarily. We present queries 5 and 10, because they both have many joins (6 and 4) and they are contrasting: Query 10 is *safe* [18, 35], and so can be efficiently evaluated by a modified SQL query. Query 5 is *unsafe* and so requires expensive Monte Carlo techniques. Graphs 3(b) and 3(c) report the time taken to execute the query and retrieve the results. For query 10, this is the total time for execution because it is safe. In contrast, query 5 requires additional Monte Carlo techniques to compute output probabilities.

Graph Discussion. In Fig. 3(b), we see running times of query 10 without probabilistic semantics *(PTPC)*, as a safe plan *(SAFE)*, with a subview materialized and retaining lineage *(LIN)* and the same subview without lineage *(NOLIN)*. *LIN* is equivalent to a standard materialized view optimization; the lineage information is computed and stored as a table. In *NOLIN*, we discard the lineage and retain only the probability that a tuple appears in the view. The graph confirms that materializing the lineage yields an order of magnitude improvement for safe queries because we do not need to compute three of the four joins at query execution

time. Interestingly, the bars for *NOLIN* show that precomputing the probabilities and ignoring the lineage yields an additional order of magnitude improvement. This optimization is correct because the materialized view is *representable*. This is interesting because it shows that being aware of when we can remove lineage is helpful even for safe plans.

As a baseline, Fig. 3(c) shows the query execution times for query 5 without probabilistic semantics but using the enlarged probabilistic tables *(PTPC)*. Fig. 3(c) also shows the cost of retrieving the tuples necessary for Monte Carlo simulation *(MC)*. Similarly, we also see the cost when materializing a view and retaining lineage *(LIN)* and when we precompute the probabilities and discard the lineage *(NOLIN)*. For *(MC)* and *(LIN)*, the extra step of Monte Carlo Simulation is necessary which for TPC 0.1 (resp. TPC 0.5, TPC 1) requires an additional 13.62 seconds (resp. 62.32s, 138.21s). Interestingly, query 5 using the materialized view does *not* require Monte Carlo Simulation because the rewritten query is safe. Thus, the time for *NOLIN* is an end-to-end running time and so we conclude that our techniques offer four order of magnitude improvement over materializing the lineage alone ($8.2s + 138.21s$ with lineage v. 0.03s without).

7.4 Question 3: How costly are our algorithms?

All views listed in this paper were correctly classified by our practical algorithm (Alg. 6), which always executes in well under 1 second. Finding all representable or partially representable sub-views for all but two queries completed in under 145 seconds; the other two queries completed in under an hour. Materializing views for unsafe queries completed under 1.5 hours for all results reported in the paper. However, this is an offline process and can be parallelized because it can utilize multiple separate Monte Carlo processes.

8. CONCLUSION

We have formalized and solved the problems of representability and using partial representable views to answer queries in the case of conjunctive views and queries. We have shown that representable and partially representable views exist in real and synthetic data sets and demonstrated that understanding representability is a large optimization win even in complete approaches.

Acknowledgements This work was partially supported by NSF ITR IIS-0428168, NSF IIS- 0454425, Suciu's CAREER NSF IIS-0092955 grant, and a gift from Microsoft.

9. REFERENCES

[1] S. Abiteboul and O. Duschka. Complexity of answering queries using materialized views. pages 254–263, 1998.
[2] Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases*. Addison Wesley Publishing Co, 1995.
[3] N. Alon and J. Spencer. *The Probabilistic Method*. John Wiley, 1992.
[4] P. Andritsos, A. Fuxman, and R. J. Miller. Clean answers over dirty databases. In *ICDE*, 2006.
[5] F. Bacchus, A.J. Grove, J.Y. Halpern, and D.Koller. Generating new beliefs from old. In *Proceedings of UAI*, pages 37–45, 1994.
[6] Daniel Barbara, Hector Garcia-Molina, and Daryl Porter. The management of probabilistic data. *IEEE Trans. Knowl. Data Eng.*, 4(5):487–502, 1992.
[7] J. Boulos, N .Dalvi, B. Mandhani, S. Mathur, C. Ré, and D. Suciu. Mystiq: A system for finding more answers by using probabilities. In *SIGMOD*, 2005. system demo.
[8] P. Buneman, A. Chapman, and J. Cheney. Provenance management in curated databases. In *SIGMOD Conference*, pages 539–550, 2006.
[9] M.J. Cafarella, C. Ré, D. Suciu, and O. Etzioni. Structured querying of web text data: A technical challenge. In *CIDR*, pages 225–234. www.crdrdb.org, 2007.
[10] S. Chaudhuri, R. Krishnamurthy, S. Potamianos, and K. Shim. Optimizing queries with materialized views. *icde*, 00:190, 1995.
[11] R. Cheng, D. Kalashnikov, and S. Prabhakar. Evaluating probabilistic queries over imprecise data. In *SIGMOD*, 2003.
[12] R. Chirkova, A. Halevy, and D. Suciu. A formal perspective on the view selection problem. In *Proceedings of VLDB*, Rome, Italy, September 2001.
[13] Garage Band Corp. `www.ilike.com`.
[14] Microsoft Corp. Northwind for sql server 2000.
[15] Microsoft Corp. Sql server 2005 samples (feb. 2007).
[16] Transaction Processing Performance Council. Tpc-h (ad-hoc, decision support) benchmark. http://www.tpc.org/.
[17] Y. Cui and J. Widom. Lineage tracing for general data warehouse transformations. *VLDBJ*, 12(1):41–58, 2003.
[18] N. Dalvi and D. Suciu. Efficient query evaluation on probabilistic databases. In *VLDB*, Toronto, Canada, 2004.
[19] N. Dalvi and D. Suciu. Management of probabilisitic data: Foundations and challenges. In *PODS*, pages 1–12, 2007.
[20] A. Das Sarma, O. Benjelloun, A. Halevy, and J. Widom. Working models for uncertain data. In *ICDE*, 2006.
[21] A. Deshpande, C. Guestrin, S. Madden, J.M. Hellerstein, and W. Hong. Model-driven data acquisition in sensor networks. In *VLDB*, pages 588–599, 2004.
[22] A. Deutsch, L. Popa, and V. Tannen. Physical data independence, constraints and optimization with universal plans. In *VLDB*, 1999.
[23] O. Etzioni, M .Banko, and M.J. Cafarella. Machine reading. In *AAAI*. AAAI Press, 2006.
[24] J. Goldstein and P. Larson. Optimizing queries using materialized views: a practical, scalable solution. In *SIGMOD 2001*, pages 331–342, New York, NY, USA, 2001. ACM Press.
[25] T.J. Green and V. Tannen. Models for incomplete and probabilistic information. *IEEE Data Engineering Bulletin*, 29(1):17–24, March 2006.
[26] R. Gupta and S. Sarawagi. Curating probabilistic databases from information extraction models. In *Proc. of the 32nd Int'l Conference on Very Large Databases (VLDB)*, 2006.
[27] A. Halevy. Answering queries using views: A survey. *VLDB Journal*, 10(4):270–294, 2001.
[28] T.S. Jayram, R. Krishnamurthy, S. Raghavan, S. Vaithyanathan, and H. Zhu. Avatar information extraction system. *IEEE Data Engineering Bulletin*, 29(1), 2006.
[29] N. Khoussainova, M. Balazinska, and D. Suciu. Probabilistic rfid data management. Technical Reprot TR2007-03-01, University of Washington, Seattle, Washington, March 2007.
[30] L. Lakshmanan, N. Leone, R. Ross, and V.S. Subrahmanian. Probview: A flexible probabilistic database system. *ACM Trans. Database Syst.*, 22(3), 1997.
[31] A. Machanavajjhala and J .Gehrke. On the efficiency of checking perfect privacy. In Stijn Vansummeren, editor, *PODS*, pages 163–172. ACM, 2006.
[32] G. Miklau and D. Suciu. A formal analysis of information disclosure in data exchange. In *SIGMOD*, 2004.
[33] O.Etzioni, M.J. Cafarella, D. Downey, S. Kok, A. Popescu, T. Shaked, S. Soderland, D.S. Weld, and A. Yates. Web-scale information extraction in knowitall: (preliminary results). In *WWW*, pages 100–110, 2004.
[34] R. Pottinger and A. Halevy. Minicon: A scalable algorithm for answering queries using views. *The VLDB Journal*, 10(2-3):182–198, 2001.
[35] C. Ré, N. Dalvi, and D. Suciu. Query evaluation on probabilistic databases. *IEEE Data Engineering Bulletin*, 29(1):25–31, 2006.
[36] C. Ré, N. Dalvi, and D. Suciu. Efficient top-k query evaluation on probabilistic data. In *Proceedings of ICDE*, 2007.
[37] C. Ré and D. Suciu. Materialized views in probabilsitic databases for information exchange and query optimization (full version). Technical Report TR2007-03-02, University of Washington, Seattle, Washington, March 2007.
[38] P. Sen and A. Deshpande. Representing and querying correlated tuples in probabilistic databases. In *Proceedings of ICDE*, 2007.
[39] M. Soliman, I.F. Ilyas, and K. Chen-Chaun Chang. Top-k query processing in uncertain databases. In *Proceedings of ICDE*, 2007.
[40] J. D. Ullman. Information integration using logical views. In *ICDT*, volume 1186 of *Lecture Notes in Computer Science*, pages 19–40. Springer, 1997.
[41] J. Widom. Trio: A system for integrated management of data, accuracy, and lineage. In *CIDR*, pages 262–276, 2005.