Probabilistic Data Exchange

Ronald Fagin
IBM Research – Almaden
fagin@almaden.ibm.com

Benny Kimelfeld
IBM Research – Almaden
kimelfeld@us.ibm.com

Phokion G. Kolaitis
UC Santa Cruz and
IBM Research – Almaden
kolaitis@cs.ucsc.edu

ABSTRACT

The work reported here lays the foundations of data exchange in the presence of probabilistic data. This requires rethinking the very basic concepts of traditional data exchange, such as solution, universal solution, and the certain answers of target queries. We develop a framework for data exchange over probabilistic databases, and make a case for its coherence and robustness. This framework applies to arbitrary schema mappings, and finite or countably infinite probability spaces on the source and target instances. After establishing this framework and formulating the key concepts, we study the application of the framework to a concrete and practical setting where probabilistic databases are compactly encoded by means of annotations formulated over random Boolean variables. In this setting, we study the problems of testing for the existence of solutions and universal solutions, materializing such solutions, and evaluating target queries (for unions of conjunctive queries) in both the exact sense and the approximate sense. For each of the problems, we carry out a complexity analysis based on properties of the annotation, in various classes of dependencies. Finally, we show that the framework and results easily and completely generalize to allow not only the data, but also the schema mapping itself to be probabilistic.

Categories and Subject Descriptors

H.2.4 [**Database Management**]: Systems—*Relational databases*; H.2.4 [**Database Management**]: Systems—*Query processing*; H.2.5 [**Database Management**]: Heterogeneous Databases—*Data translation*

General Terms

Theory

Keywords

Data exchange, data integration, probabilistic database, probabilistic schema mapping, probabilistic solution, universal probabilistic solution, conjunctive query, certain answer, computational complexity

ICDT 2010, March 22–25, 2010, Lausanne, Switzerland.

1. INTRODUCTION

Data exchange is the problem of transforming data that conform to one schema, the *source schema*, into data that conform to another schema, the *target schema*, in a way that is consistent with various *dependencies* (i.e., constraints expressed in some logical formalism over the two schemas). The source and target schemas, along with the dependencies, define a *schema mapping*, and the results of the consistent transformation of a source instance are called *solutions*. Traditional data exchange is based on the assumption that source data are *certain*. However, the need to account for *uncertainty* in data has long been recognized [4, 19]. In view of the advent of the Web and related modern applications, models of uncertain data (typically *probabilistic databases*) have recently gained significant renewed focus [9–11, 24, 31, 33, 43, 44]. It is, therefore, essential to rethink the conceptual framework of data exchange in the context of uncertainty in the source data.

Our goal in this paper is to lay the foundations of data exchange in the presence of probabilistic data. This is accomplished in two main parts. First, in Sections 2–4, we establish a framework that extends and generalizes traditional data exchange to probabilistic (source and target) databases. This framework is general, in the sense that it imposes essentially *no restriction at all* on the types of dependencies or on the probabilistic databases (which are finite or countably infinite spaces of ordinary finite databases, where each database is assigned a probability). Then, in Section 5, we apply our framework to a concrete and practical setting, where the dependencies are from widely-studied classes, and where the probabilistic databases are compactly encoded in various conventional manners (e.g., as in [2, 6, 10, 31, 43]).

Furthermore, in Section 6, we extend the framework and the results to allow the schema mapping (and the data) to be probabilistic. In principle, we could use this extended setting right from the beginning. The reason for not doing so is that it would significantly increase the complexity of the presentation, while the key challenges and ideas arise already when only the data are probabilistic.

Formally, a *schema mapping* is a triple $(\mathbf{S}, \mathbf{T}, \Sigma)$, where $\mathbf{S}$ and $\mathbf{T}$ are the *source* and *target* schemas, respectively, and Σ is a set of dependencies formulated as logical assertions over $\mathbf{S}$ and $\mathbf{T}$. A *source instance* is an instance I over $\mathbf{S}$, and a *target instance* is an instance J over $\mathbf{T}$; moreover, J is allowed to include *labeled nulls*, which are essentially variables that are not bound to specific values. A target instance J is a *solution* if the pair $\langle I, J\rangle$ satisfies Σ. In this paper, source and target instances are replaced with *probabilistic instances* (abbrev. *p-instances*): a *source p-instance* is a probability space $\tilde{\mathcal{I}}$ over the source instances, and a *target p-instance* is a probability space $\tilde{\mathcal{J}}$ over the target instances.

The first task is, naturally, to define a *probabilistic solution* (abbrev. *p-solution*) for a source p-instance w.r.t. a schema mapping

$(\mathbf{S}, \mathbf{T}, \Sigma)$. Essentially, we define a target p-instance $\tilde{\mathcal{J}}$ to be a *p-solution* for a p-instance $\tilde{\mathcal{I}}$ if there exists a probability space over source-solution pairs (I, J) (i.e., J is a solution for I w.r.t. Σ), such that the marginals coincide with the p-instance $\tilde{\mathcal{I}}$ on the one hand, and with the p-instance $\tilde{\mathcal{J}}$ on the other. Our definition of a p-solution is based on the classical concept of a *bivariate* (joint) probability space with given marginals (research of this concept goes back to the 1950s [18,36]), but with the additional requirement that the *support* (i.e., the set of samples with a nonzero probability) is contained in a fixed relation (in this case, the source-solution relation). To explore the coherence of this definition, we formulate two intuitive properties that every reasonable concept of a solution should satisfy. Each of these properties says that a p-solution *properly reflects* the uncertainty of the source data. Rather surprisingly, we show that each of the two properties is actually a characterization of a p-solution.

We then proceed to the adaptation of the notion of a *universal solution*. Our definition of a *universal p-solution* is similar to that of a *p-solution* (given above), except that we require the existence of a probability space over pairs (I, J), such that J is a *universal* solution for I (and, again, the marginals coincide with $\tilde{\mathcal{I}}$ and $\tilde{\mathcal{J}}$). On the surface, this definition does not imply any desired semantic property. In traditional data exchange, a universal solution J is a "good" solution in the sense that it generalizes all the other solutions, since every solution contains a homomorphic image of J. We want a similar property to characterize a universal p-solution. For that, we need to figure out the meaning of *generalization* between p-instances.

There are various ways of formally modeling the generalization relationship between p-instances; we consider three natural definitions, where each of the three extends the traditional concept (existence of a homomorphism) to p-instances. One definition is (again) in terms of a bivariate distribution, and the other two are based on the notion of a *stochastic order* (see, e.g., [45]). We show that the three are different from one another (and moreover, in the finite case, testing whether they hold belong to different complexity classes). So, we do not have one robust formalization of the generalization relationship between p-solutions. *A priori*, each of the three relationships could imply a different alternative definition of a universal p-solution, namely, one that "generalizes" all the p-solutions. Quite remarkably, the three definitions are *equivalent* to the above definition of a universal p-solution. Furthermore, as we show next when we consider the concept of answering target queries, a universal p-solution is also characterized by its usefulness in answering target conjunctive queries (as in the deterministic case [15]). These results indicate that the concept of a *universal p-solution* is very robust.

Since a solution in our framework (namely, a p-solution) is inherently probabilistic, evaluating target queries amounts to *querying probabilistic databases*. In particular, for a source p-instance $\tilde{\mathcal{I}}$ and a query q, every p-solution $\tilde{\mathcal{J}}$ gives a (potentially different) confidence value for each possible answer $\mathbf{a}$. Consistently with the approach of *certain answers* in traditional data exchange, the *confidence* of $\mathbf{a}$ is defined to be the infimum of the confidence values for $\mathbf{a}$ over all p-solutions. We show that (when a p-solution exists) this is the same as the probability that $\mathbf{a}$ is a certain answer for a random source instance of $\tilde{\mathcal{I}}$. We show that a universal p-solution can be used for answering unions of conjunctive queries (UCQs), namely, evaluation thereon gives the correct confidence values. Moreover, if a p-solution can be used this way in the evaluation of conjunctive queries, then this p-solution is necessarily universal.

We then proceed to study algorithmic and computational aspects of data exchange for finite probabilistic databases. Specifically, we consider the following problems: testing for the existence of solutions and universal solutions, materializing such solutions, and evaluating target unions of conjunctive queries. It follows from our results that these problems are not harder than their counterparts in the traditional (deterministic) setting. That holds, though, under the assumption that the source p-instance is represented in an explicit manner (i.e., by specifying each possible world I along with its probability). This is at odds with conventional practice, which is to associate a measure of confidence (or a probabilistic event) with each fact. Such a representation (along with some statistical assumptions) is typically logarithmic-scale compact. So, following existing representations (e.g., *ULDBs* [2, 43], *probabilistic c-tables* [24] and *probabilistic trees* [44]), we explore a setting where the source p-instance is represented compactly by *annotating* facts with *conditions*, which are formulas over a set of (Boolean and probabilistically independent) *random event variables*. We consider two types of annotations. In a *DNF instance* the annotation is in disjunctive normal form; in a *tuple-independent instance* different facts are probabilistically independent, and the annotation effectively specifies the probability of each fact, as done in [6, 10, 11]).

Our analysis is based on *data complexity*, which is common in studying the complexity aspects of data exchange, (e.g., [15–17, 20]). Thus, we hold fixed a schema mapping and a query (when relevant), and the input consists of an annotated (i.e., DNF or tuple-independent) source instance. In our analysis, we consider the types of dependencies that were studied in [15]. Thus, we allow *st-tgds* (source-to-target tgds), *t-tgds* (target tgds)[1], and *t-egds* (target egds). We consider also the effect on the complexity when the st-tgds and/or t-tgds are restricted to being *full*. We divide the computational problems into *categories* that correspond to all possible combinations of dependency and annotation types. We start with the problems of testing whether a (universal) solution exists and of materializing one that is encoded as a DNF instance. For each category, we show that either the corresponding problem is tractable for all schema mappings (in the category) or that there exists a schema mapping for which the problem is intractable. We then consider target-query evaluation and, in particular, show that every nontrivial UCQ is #P-hard in some schema mapping of the most restrictive category (namely, independent facts and full st-tgds). Due to this hardness, we study the complexity of *approximate* query evaluation (which, in practice, is often good enough), and give the following complete classification. For each category, we prove one of the following:

- For every schema mapping and for every target UCQ there exists an efficient algorithm (randomized or deterministic) for approximate query evaluation.
- For every nontrivial target UCQ there exists a schema mapping in which query evaluation is hard to approximate.

Finally, we show how to generalize the framework and all of the aforementioned results to accommodate probabilistic schema mappings (in addition to probabilistic data). The combination of a probabilistic schema mapping with a source p-instance requires having a joint probability distribution over sets of dependencies and source instances; that is, a probability space on pairs (Σ, I), where Σ is a set of dependencies and I is a source instance. We call such a probability distribution a *probabilistic problem* (*p-problem*, in short). In general, a p-problem allows for every correlation between the probabilistic mapping and the source p-instance; a special case is

[1]We make the now standard assumption of *weak acyclicity* [15].

the *product space* where the probabilistic schema mapping and the source p-instance are assumed to be independent.

We show that the framework and all aforementioned results *completely* generalize to p-problems, under the proper adaptation of the definitions. In particular, we use the notions of a *p-solution*, a *universal p-solution* and an *answer confidence* (for a target query) *for a p-problem* $\tilde{\mathcal{P}}$ rather than *for a source p-instance* $\tilde{\mathcal{I}}$. Moreover, the results of Section 5 are generalized by annotating the dependencies specifying the mapping similarly to source facts (i.e., using formulas over event variables); event variables can be shared between facts and dependencies, thereby allowing correlations between the probabilistic source data and mappings to be represented.

To the best of our knowledge, this work is the first to study data exchange over probabilistic databases. In [13, 14, 41, 42], the problem of data exchange (and specifically data integration) for deterministic databases and probabilistic mappings is studied. The relationship between that work and this paper is discussed in Section 6.2.

The proofs of the results presented in this paper will appear in a full version.

2. PRELIMINARIES

2.1 Schemas and Instances

We assume fixed countably infinite sets Const of *constants* and Var of *nulls*, such that $\mathsf{Const} \cap \mathsf{Var} = \emptyset$. A *schema* is a finite sequence $\mathbf{R} = \langle R_1, \ldots, R_k \rangle$ of distinct relation symbols, where each R_i has a fixed arity $r_i > 0$. An *instance* I (*over* $\mathbf{R}$) is a sequence $\langle R_1^I, \ldots, R_k^I \rangle$, such that each R_i^I is a finite relation of arity r_i over $\mathsf{Const} \cup \mathsf{Var}$ (i.e., R_i^I is a finite subset of $(\mathsf{Const} \cup \mathsf{Var})^{r_i}$). We call R_i^I the *R_i-relation* of I. We may abuse this notation and use R_i to denote both the relation symbol and the relation R_i^I that interprets it. We use $\mathsf{dom}(I)$ to denote the set of all constants and nulls that appear in I. We say that I is a *ground instance* if $\mathsf{dom}(I)$ does not contain nulls. We denote by $\mathsf{Inst}(\mathbf{R})$ and $\mathsf{Inst}^{\mathsf{c}}(\mathbf{R})$ the classes of all instances and ground instances, respectively, over $\mathbf{R}$. We use $R(t_1, \ldots, t_r)$ to denote that $(t_1, \ldots, t_r)$ is a tuple in a relation R and call it a *fact*. We identify an instance with the set of its facts.

Let K_1 and K_2 be instances over the same schema. A *homomorphism* $h : K_1 \to K_2$ is a mapping from $\mathsf{dom}(K_1)$ to $\mathsf{dom}(K_2)$, such that *(1)* $h(c) = c$ for all constants $c \in \mathsf{dom}(K_1)$, *and (2)* for all facts $R(\mathbf{t})$ of K_1, the fact $R(h(\mathbf{t}))$ is in K_2 (for $\mathbf{t} = (t_1, \ldots, t_r)$, the tuple $h(\mathbf{t})$ is $(h(t_1), \ldots, h(t_r))$). By $K_1 \to K_2$ we denote the existence of a homomorphism $h : K_1 \to K_2$.

2.2 Schema Mappings

We now describe our formalism of a *schema mapping*, which follows that of [15]. Suppose that $\mathbf{S} = \langle S_1, \ldots, S_n \rangle$ and $\mathbf{T} = \langle T_1, \ldots, T_m \rangle$ are two schemas with no relation symbols in common. We denote by $\langle \mathbf{S}, \mathbf{T} \rangle$ the schema that is obtained by concatenating $\mathbf{S}$ and $\mathbf{T}$. Similarly, if I and J are instances of $\mathbf{S}$ and $\mathbf{T}$, respectively, then $\langle I, J \rangle$ is the instance $K \in \mathsf{Inst}(\langle \mathbf{S}, \mathbf{T} \rangle)$ that satisfies $S_i^K = S_i^I$ and $T_j^K = T_j^J$ for $1 \leq i \leq n$ and $1 \leq j \leq m$; in other words, since we identify an instance with the set of its facts, $\langle I, J \rangle$ is essentially the union of I and J.

We assume some formalism for expressing constraints over a given schema $\mathbf{R}$. If $I \in \mathsf{Inst}(\mathbf{R})$ and Σ is a set of formulas in this formalism, then $I \models \Sigma$ denotes that I satisfies every formula of Σ.

A *schema mapping* is a triple $(\mathbf{S}, \mathbf{T}, \Sigma)$, where $\mathbf{S}$ (the *source schema*) and $\mathbf{T}$ (the *target schema*) are schemas without common relation symbols, and Σ is a set of formulas over the schema $\langle \mathbf{S}, \mathbf{T} \rangle$. Each formula of Σ is called a *dependency*. A *source instance* is a ground instance I over $\mathbf{S}$, and a *target instance* is an instance J over $\mathbf{T}$ (that is, $I \in \mathsf{Inst}^{\mathsf{c}}(\mathbf{S})$ and $J \in \mathsf{Inst}(\mathbf{T})$). We say that the target instance J is a *solution for* I (*w.r.t.* Σ) if $\langle I, J \rangle \models \Sigma$. A solution J for I w.r.t. Σ is *universal* if $J \to J'$ for all solutions J' for I w.r.t. Σ (in other words, every solution contains a homomorphic image of J).

2.3 Probability Spaces

All the probability spaces we consider are countable (finite or countably infinite). We call such spaces *p-spaces* and use the following notation. A p-space is a pair $\tilde{\mathcal{U}} = (\Omega(\tilde{\mathcal{U}}), p_{\tilde{\mathcal{U}}})$, such that $\Omega(\tilde{\mathcal{U}})$ is a countable set and $p_{\tilde{\mathcal{U}}} : \Omega(\tilde{\mathcal{U}}) \to [0, 1]$ is a function that satisfies $\sum_{u \in \Omega(\tilde{\mathcal{U}})} p_{\tilde{\mathcal{U}}}(u) = 1$. Each member u of $\Omega(\tilde{\mathcal{U}})$ is a *sample*, and $\Omega(\tilde{\mathcal{U}})$ is the *sample space*. We say that the p-space $\tilde{\mathcal{U}}$ is *over* $\Omega(\tilde{\mathcal{U}})$. The *support* of $\tilde{\mathcal{U}}$, denoted $\Omega_+(\tilde{\mathcal{U}})$, is the set of all samples $u \in \Omega(\tilde{\mathcal{U}})$ such that $p_{\tilde{\mathcal{U}}}(u) > 0$. We say that $\tilde{\mathcal{U}}$ is *finite* if its support $\Omega_+(\tilde{\mathcal{U}})$ is finite. A subset $X \subseteq \Omega(\tilde{\mathcal{U}})$ is called an *event*. The *probability* of the event X, denoted $\Pr_{\tilde{\mathcal{U}}}(X)$, is the sum $\sum_{u \in X} p_{\tilde{\mathcal{U}}}(u)$. We may omit the subscript $\tilde{\mathcal{U}}$ from $\Pr_{\tilde{\mathcal{U}}}(X)$ when it is clear from the context. We use $\mathcal{U}$ (i.e., without the tilde sign) to denote the *random variable* that represents a sample of $\tilde{\mathcal{U}}$. An event is often represented by a logical formula over $\mathcal{U}$ (e.g., $\varphi(\mathcal{U})$ is the same as $\{u \in \Omega(\tilde{\mathcal{U}}) \mid \varphi(u)\}$). We often abuse the above notation and identify $\tilde{\mathcal{U}}$ with its sample space $\Omega(\tilde{\mathcal{U}})$ (e.g., $u \in \tilde{\mathcal{U}}$ means that u is a member of $\Omega(\tilde{\mathcal{U}})$).

3. EXCHANGING PROBABILISTIC DATA

Our goal is to study data exchange in the presence of uncertainty in the source instance. We use the convention of modeling uncertain data as a *probabilistic database* [10, 11, 24, 33, 43]. The challenges of this generalization of data exchange arise right in the beginning: What is the meaning of a *solution* for a probabilistic source instance? The first observation is that such a solution should by itself be probabilistic (because if the source database is uncertain, then so is the target database). Next, we formalize the notion of a probabilistic database.

Let $\mathbf{R}$ be a schema. A *probabilistic database*, or a *probabilistic instance* (*over* $\mathbf{R}$), abbrev. *p-instance*, is a p-space $\tilde{\mathcal{I}}$ over $\mathsf{Inst}(\mathbf{R})$. If $\tilde{\mathcal{I}}$ is a p-space over $\mathsf{Inst}^{\mathsf{c}}(\mathbf{R})$, then $\tilde{\mathcal{I}}$ is a *ground p-instance*. Note that the sample space $\mathsf{Inst}(\mathbf{R})$ (or $\mathsf{Inst}^{\mathsf{c}}(\mathbf{R})$) is countable due to our assumption that Const and Var are countable (and that ordinary instances are finite).

Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ be a schema mapping. A *source p-instance* is a ground p-instance $\tilde{\mathcal{I}}$ over $\mathbf{S}$, and a *target p-instance* is a p-instance $\tilde{\mathcal{J}}$ over $\mathbf{T}$. In other words, a source p-instance and a target p-instance are p-spaces over the source and target instances, respectively.

EXAMPLE 3.1. Let $\mathcal{M}$ be the schema mapping $(\mathbf{S}, \mathbf{T}, \Sigma)$, where $\mathbf{S}$ and $\mathbf{T}$ are defined as follows. Note that, for convenience, the columns are named.

$\mathbf{S}$: $Researcher$(name, university), $RArea$(researcher, topic)

$\mathbf{T}$: $UArea$(university, department, topic)

and Σ contains the single dependency

$$\forall r, u, t (Researcher(r, u) \wedge RArea(r, t) \rightarrow \exists d \; UArea(u, d, t)) .$$

Figure 1 depicts a set of possible facts (e.g., r_{e}, a_{eir} and u_{ir}) for each of the three relations. Note that $\perp_1$, $\perp_2$ and $\perp_3$ are nulls

Researcher facts	
r_{e}	*Researcher*(Emma, UCSD)
r_{j}	*Researcher*(John, UCSD)

RArea facts	
a_{eir}	*RArea*(Emma, IR)
a_{edb}	*RArea*(Emma, DB)
a_{jdb}	*RArea*(John, DB)
a_{jai}	*RArea*(John, AI)

Source p-instance $\tilde{\mathcal{I}}$	
$I_1 = \{r_{\mathsf{e}}, r_{\mathsf{j}}, a_{\mathsf{eir}}, a_{\mathsf{jdb}}\}$	0.3
$I_2 = \{r_{\mathsf{e}}, r_{\mathsf{j}}, a_{\mathsf{eir}}, a_{\mathsf{jai}}\}$	0.3
$I_3 = \{r_{\mathsf{e}}, r_{\mathsf{j}}, a_{\mathsf{edb}}, a_{\mathsf{jai}}\}$	0.2
$I_4 = \{r_{\mathsf{e}}, r_{\mathsf{j}}, a_{\mathsf{edb}}, a_{\mathsf{jdb}}\}$	0.1
$I_5 = \{r_{\mathsf{e}}, a_{\mathsf{edb}}\}$	0.1

UArea facts	
u_{ir}	*UArea*(UCSD, $\perp_1$, IR)
u_{ai}	*UArea*(UCSD, $\perp_2$, AI)
u_{db}	*UArea*(UCSD, $\perp_3$, DB)

Target p-instance $\tilde{\mathcal{J}}_1$	
$J_1 = \{u_{\mathsf{ir}}, u_{\mathsf{db}}\}$	0.3
$J_2 = \{u_{\mathsf{ir}}, u_{\mathsf{ai}}\}$	0.3
$J_3 = \{u_{\mathsf{db}}, u_{\mathsf{ai}}\}$	0.2
$J_4 = \{u_{\mathsf{db}}\}$	0.2

Target p-instance $\tilde{\mathcal{J}}_2$	
$J_5 = \{u_{\mathsf{ir}}, u_{\mathsf{db}}\}$	0.35
$J_6 = \{u_{\mathsf{ir}}, u_{\mathsf{ai}}, u_{\mathsf{db}}\}$	0.45
$J_7 = \{u_{\mathsf{ir}}, u_{\mathsf{ai}}\}$	0.2

Target p-instance $\tilde{\mathcal{J}}_3$	
$J_8 = \{u_{\mathsf{ir}}, u_{\mathsf{db}}\}$	0.3
$J_9 = \{u_{\mathsf{ai}}, u_{\mathsf{db}}\}$	0.3
$J_{10} = \{u_{\mathsf{ir}}, u_{\mathsf{ai}}\}$	0.4

Figure 1: Source and target p-instances for the schema mapping $\mathcal{M}$ of Example 3.1

(and the rest of the data values are constants). Using the facts, the figure depicts four finite p-instances $\tilde{\mathcal{I}}$, $\tilde{\mathcal{J}}_1$, $\tilde{\mathcal{J}}_2$ and $\tilde{\mathcal{J}}_3$, where $\tilde{\mathcal{I}}$ is a source p-instance and each $\tilde{\mathcal{J}}_i$ is a target p-instance. Each sample of a p-instance is represented by a two-entry row, where the left entry shows the instance and the right one shows its probability. For example, the probability $p_{\tilde{\mathcal{I}}}(I_1)$ of $I_1 = \{r_{\mathsf{e}}, r_{\mathsf{j}}, a_{\mathsf{eir}}, a_{\mathsf{jdb}}\}$ is 0.3. Observe that in each of $\tilde{\mathcal{I}}$, $\tilde{\mathcal{J}}_1$, $\tilde{\mathcal{J}}_2$ and $\tilde{\mathcal{J}}_3$, the probabilities in the rows sum up to 1. □

The challenge is to identify when a target p-instance constitutes a solution for a source p-instance. In principle, we have a binary relationship between deterministic instances I and J (namely, J is a solution for I), and we want to generalize it to p-instances $\tilde{\mathcal{I}}$ and $\tilde{\mathcal{J}}$. The *probabilistic match* is a systematic way of extending a binary relationship between objects into a binary relationship between p-spaces thereof. Next, we give the formal definition of a probabilistic match. In Section 3.2, we apply it to define our notion of a solution in the probabilistic setting, which we call a *p-solution*. Then, we show that this definition is semantically coherent, by considering two natural and desirable requirements for a notion of a solution and showing that each of these requirements actually characterizes a p-solution.

3.1 Probabilistic Match

Our notion of a probabilistic match between p-spaces is based on the classical concept of joint (or *bivariate*) probability spaces with specified marginals [18, 36]. Our new twist on this old notion is that we require the joint distribution to have a support contained in a given binary relation.

DEFINITION 3.2. ***(Probabilistic Match)*** Let $\tilde{\mathcal{U}}$ and $\tilde{\mathcal{W}}$ be two p-spaces and let $R \subseteq \Omega(\tilde{\mathcal{U}}) \times \Omega(\tilde{\mathcal{W}})$ be a binary relation. A *probabilistic match of $\tilde{\mathcal{U}}$ in $\tilde{\mathcal{W}}$ w.r.t. R* (or, for short, an *R-match of $\tilde{\mathcal{U}}$ in $\tilde{\mathcal{W}}$*) is a p-space $\tilde{\mathcal{P}}$ over $\Omega(\tilde{\mathcal{U}}) \times \Omega(\tilde{\mathcal{W}})$ that satisfies the following two conditions.

1. The support of $\tilde{\mathcal{P}}$ is contained in R (i.e., $\Pr(\mathcal{P} \in R) = 1$).
2. The marginals of $\tilde{\mathcal{P}}$ are $\tilde{\mathcal{U}}$ and $\tilde{\mathcal{W}}$. This means that:
   (i) $\sum_{w \in \Omega(\tilde{\mathcal{W}})} p_{\tilde{\mathcal{P}}}(u, w) = p_{\tilde{\mathcal{U}}}(u)$ for all $u \in \tilde{\mathcal{U}}$, *and*
   (ii) $\sum_{u \in \Omega(\tilde{\mathcal{U}})} p_{\tilde{\mathcal{P}}}(u, w) = p_{\tilde{\mathcal{W}}}(w)$ for all $w \in \tilde{\mathcal{W}}$. □

Note that an R-match of $\tilde{\mathcal{U}}$ in $\tilde{\mathcal{W}}$ can be viewed as a probability space over R, whose marginals coincide with $\tilde{\mathcal{U}}$ and $\tilde{\mathcal{W}}$. A special case of a probabilistic match is the *product space* $\tilde{\mathcal{U}} \times \tilde{\mathcal{W}}$, where R is the set $\Omega(\tilde{\mathcal{U}}) \times \Omega(\tilde{\mathcal{W}})$ and the two coordinates are probabilistically independent (that is, $p_{\tilde{\mathcal{U}} \times \tilde{\mathcal{W}}}(u, w) = p_{\tilde{\mathcal{U}}}(u) \cdot p_{\tilde{\mathcal{W}}}(w)$ for all $u \in \tilde{\mathcal{U}}$ and $w \in \tilde{\mathcal{W}}$). Two other special cases, for a relation $R \subseteq \Omega(\tilde{\mathcal{U}}) \times \Omega(\tilde{\mathcal{W}})$, are the following.

- An R-match $\tilde{\mathcal{P}}$ is *left-trivial* if for every $u \in \Omega_+(\tilde{\mathcal{U}})$ there is exactly one $w \in \Omega(\tilde{\mathcal{W}})$ such that $p_{\tilde{\mathcal{P}}}(u, w) > 0$; equivalently, $\Pr_{\tilde{\mathcal{P}}}(u, w) = \Pr_{\tilde{\mathcal{U}}}(u)$ whenever $\Pr_{\tilde{\mathcal{P}}}(u, w) > 0$.
- Similarly, $\tilde{\mathcal{P}}$ is *right-trivial* if for every $w \in \Omega_+(\tilde{\mathcal{W}})$ there is exactly one $u \in \Omega(\tilde{\mathcal{U}})$ such that $p_{\tilde{\mathcal{P}}}(u, w) > 0$; equivalently, $\Pr_{\tilde{\mathcal{P}}}(u, w) = \Pr_{\tilde{\mathcal{W}}}(w)$ whenever $\Pr_{\tilde{\mathcal{P}}}(u, w) > 0$.

EXAMPLE 3.3. Consider the p-instances $\tilde{\mathcal{I}}$, $\tilde{\mathcal{J}}_1$ and $\tilde{\mathcal{J}}_2$ of Figure 1. The two bipartite graphs of Figure 2 depict (finite) probabilistic matches of $\tilde{\mathcal{I}}$ in $\tilde{\mathcal{J}}_1$ (in the graph on the left side of Figure 2) and in $\tilde{\mathcal{J}}_2$ (in the graph on the right side of Figure 2). The relations R are the ones given by the (solid and dashed) edges that connect each of the two pairs of p-spaces. The probability of each pair (I, J) is written inside a rectangular box on the corresponding edge, unless this probability is zero and then the edge is represented

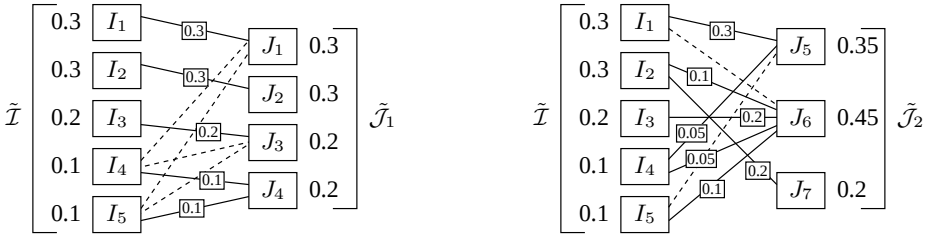

Figure 2: $\mathsf{SOL}_{\mathcal{M}}$-matches of $\tilde{\mathcal{I}}$ in $\tilde{\mathcal{J}}_1$ and of $\tilde{\mathcal{I}}$ in $\tilde{\mathcal{J}}_2$ for the source p-instance $\tilde{\mathcal{I}}$ and the target p-instances $\tilde{\mathcal{J}}_1$ and $\tilde{\mathcal{J}}_2$ of Example 3.1

as a dashed line. (Recall that the probability of (I, J) is necessarily zero if no edge connects I and J.)

Observe that the probabilistic match of $\tilde{\mathcal{I}}$ in $\tilde{\mathcal{J}}_1$ on the left side of Figure 2 is left-trivial, since every node on the left side is incident to exactly one nonzero edge. Thus, it is immediate to verify that Item (i) in Definition 3.2 holds. Note that this match is not right-trivial (since J_4 is incident to two nonzero edges). Actually, there cannot be any right-trivial match of $\tilde{\mathcal{I}}$ in $\tilde{\mathcal{J}}_1$, simply because $\Omega_+(\tilde{\mathcal{J}}_1)$ contains fewer samples than $\Omega_+(\tilde{\mathcal{I}})$.

A more complex example of a probabilistic match is the match of $\tilde{\mathcal{I}}$ in $\tilde{\mathcal{J}}_2$ on the right side of Figure 2. Note that this match is neither left-trivial nor right-trivial. Consider the instance I_4 in the right side of Figure 2. In Item (i) in Definition 3.2, when the role of u is played by I_4, the sum on the left side of Item (i), which is the sum of probabilities of the edges adjacent to I_4, is $0.05 + 0.05 = 0.1$, which is exactly the probability of I_4, which is the value on the right side of Item (i). Consider now the instance J_6 in the right side of Figure 2. In Item (ii) in Definition 3.2, when the role of w is played by J_6, the sum on the left side of Item (ii), which is the sum of probabilities of the edges adjacent to J_6, is $0+0.1+0.2+0.05+0.1 = 0.45$, which is exactly the probability of J_6, which is the value on the right side of Item (ii). □

3.2 p-Solution

We are now ready to define the concept of a p-solution. For a schema mapping $\mathcal{M}$, we denote by $\mathsf{SOL}_{\mathcal{M}}$ the binary relation that comprises pairs $(I, J) \in \mathsf{Inst}^{\mathsf{c}}(\mathbf{S}) \times \mathsf{Inst}(\mathbf{T})$, such that J is a solution for I.

DEFINITION 3.4. ***(p-Solution)*** Let $\mathcal{M}$ be a schema mapping and let $\tilde{\mathcal{I}}$ be a source p-instance. A *p-solution (for $\tilde{\mathcal{I}}$ w.r.t. Σ)* is a target p-instance $\tilde{\mathcal{J}}$, such that there is a $\mathsf{SOL}_{\mathcal{M}}$-match of $\tilde{\mathcal{I}}$ in $\tilde{\mathcal{J}}$. □

Note that by a $\mathsf{SOL}_{\mathcal{M}}$-match we mean, of course, an R-match where the role of R is played by $\mathsf{SOL}_{\mathcal{M}}$.

EXAMPLE 3.5. Consider again the schema mapping $\mathcal{M}$ of Example 3.1 and the source and target p-instances that are depicted in Figure 1 and described in Example 3.1. Figure 2 shows two $\mathsf{SOL}_{\mathcal{M}}$-matches of $\tilde{\mathcal{I}}$: the one on the left side is in $\tilde{\mathcal{J}}_1$, and the one on the right side is in $\tilde{\mathcal{J}}_2$. For example, there are edges from I_4 to J_1, J_3, and J_4, since J_1, J_3, and J_4 are each solutions for I_4 w.r.t. Σ (the edges from I_4 to J_1 and J_3 each have probability 0, which is allowed). There is no edge from I_4 to J_2, since J_2 is not a solution for I_4 w.r.t. Σ. Thus, both $\tilde{\mathcal{J}}_1$ and $\tilde{\mathcal{J}}_2$ are p-solutions. Later, we will show that $\tilde{\mathcal{J}}_3$ is not a p-solution (i.e., there is no $\mathsf{SOL}_{\mathcal{M}}$-match of $\tilde{\mathcal{I}}$ in $\tilde{\mathcal{J}}_3$). □

Defining a p-solution by means of a $\mathsf{SOL}_{\mathcal{M}}$-match is a straightforward application of the probabilistic-match mechanism. Next, we give a semantic justification to this definition. We start with an example. Consider the schema mapping $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ and the source and target p-instances of Example 3.1. As shown in Examples 3.3 and 3.5, $\tilde{\mathcal{J}}_1$ and $\tilde{\mathcal{J}}_2$ are p-solutions. One may claim that $\tilde{\mathcal{J}}_3$ should be deemed a p-solution as well (even though we later show that it is not) due to the following statement (that can be easily verified). For each sample I of $\tilde{\mathcal{I}}$ (which has the probability $\Pr_{\tilde{\mathcal{I}}}(I)$ of being the selected instance), there is a probability of $\Pr_{\tilde{\mathcal{I}}}(I)$, or even higher, that a sample of $\tilde{\mathcal{J}}$ is a solution for I. Next, we show that this property is not enough, and moreover, that $\tilde{\mathcal{J}}_3$ should *not* be a p-solution.

For an arbitrary target p-instance $\tilde{\mathcal{J}}$, let $p_{\mathsf{db}}(\tilde{\mathcal{J}})$ be the probability that, in $\tilde{\mathcal{J}}$, database (DB) research is done in UCSD. The source p-instance $\tilde{\mathcal{I}}$ says that there is a probability of 0.7 that at least one researcher of UCSD is in the DB area (as obtained by summing up the probabilities of all the instances that contain a_{edb}, a_{jdb} or both). By the schema mapping $\mathcal{M}$ we would like a p-solution $\tilde{\mathcal{J}}$ to say that DB research is done in UCSD with a probability of 0.7, that is, $p_{\mathsf{db}}(\tilde{\mathcal{J}}) = 0.7$. Moreover, since Σ allows DB research at UCSD even if the source does not contain a DB researcher at UCSD, we should allow $p_{\mathsf{db}}(\tilde{\mathcal{J}})$ to be larger than 0.7, in addition to allowing it to equal 0.7. Now, $p_{\mathsf{db}}(\tilde{\mathcal{J}}_1)$ is exactly 0.7 and $p_{\mathsf{db}}(\tilde{\mathcal{J}}_2)$ is 0.8, as desired. However, this is not the case for $\tilde{\mathcal{J}}_3$, since $p_{\mathsf{db}}(\tilde{\mathcal{J}}_3) = 0.6$.

To generalize the above example, consider a schema mapping $\mathcal{M}$, a source p-instance $\tilde{\mathcal{I}}$ and an event E of $\tilde{\mathcal{I}}$ (e.g., the event "one or more researchers are in the DB area in UCSD," which means that a_{edb} or a_{jdb} or both are in the source instance), We say that a target instance J is *consistent* with E if J is a solution for at least one instance I of E. Then, as illustrated above, the following property is desired from a p-solution $\tilde{\mathcal{J}}$. For all events E of $\tilde{\mathcal{I}}$, the probability that $\tilde{\mathcal{J}}$ is consistent with E is at least the probability of E. An analogous desired property is the following. For all events F of $\tilde{\mathcal{J}}$ (e.g., the event "the DB area in UCSD is nonempty", which means that u_{db} is in the target instance), the probability that a random instance of $\tilde{\mathcal{I}}$ has a solution in F is at least the probability of F. It can rather easily be shown that existence of a $\mathsf{SOL}_{\mathcal{M}}$-match guarantees these two properties. Rather surprisingly, each of the two properties implies the existence of a $\mathsf{SOL}_{\mathcal{M}}$-match; thus, as shown in the next theorem, each of the two is a characterization of a p-solution.

THEOREM 3.6. *Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ be a schema mapping. Let $\tilde{\mathcal{I}}$ be a source p-instance and let $\tilde{\mathcal{J}}$ be a target p-instance. The following are equivalent.*

1. *$\tilde{\mathcal{J}}$ is a p-solution (that is, a $\mathsf{SOL}_{\mathcal{M}}$-match of $\tilde{\mathcal{I}}$ in $\tilde{\mathcal{J}}$ exists).*
2. *For all $E \subseteq \mathsf{Inst}^{\mathsf{c}}(\mathbf{S})$,*
$$\Pr_{\tilde{\mathcal{J}}}\left(\bigvee_{I \in E} \langle I, \mathcal{J}\rangle \models \Sigma\right) \geq \Pr_{\tilde{\mathcal{I}}}(E)\,.$$
3. *For all $F \subseteq \mathsf{Inst}(\mathbf{T})$,*
$$\Pr_{\tilde{\mathcal{I}}}\left(\bigvee_{J \in F} \langle \mathcal{I}, J\rangle \models \Sigma\right) \geq \Pr_{\tilde{\mathcal{J}}}(F)\,.$$

Note that, following the above discussion about $\tilde{\mathcal{J}}_3$, the fact that Part 2 of Theorem 3.6 is necessary for being a p-solution shows that $\tilde{\mathcal{J}}_3$ is not a p-solution for $\tilde{\mathcal{I}}$ (by using the event E saying that there is a DB researcher in UCSD). Theorem 3.6 is proved via the following characterization of the existence of a probabilistic match in the spirit of Hall's Marriage Theorem [25].

LEMMA 3.7. *Let $\tilde{\mathcal{U}}$ and $\tilde{\mathcal{W}}$ be two p-spaces and let $R \subseteq \Omega(\tilde{\mathcal{U}}) \times \Omega(\tilde{\mathcal{W}})$ be a binary relation. There exists an R-match of $\tilde{\mathcal{U}}$ in $\tilde{\mathcal{W}}$ if and only if for all events U of $\tilde{\mathcal{U}}$ it holds that $\Pr_{\tilde{\mathcal{U}}}(U) \leq \Pr_{\tilde{\mathcal{W}}}\big(\bigvee_{u \in U} R(u, \mathcal{W})\big)$.*

The proof for finite p-spaces is by an application of the *max-flow min-cut* theorem. For countably infinite graphs, the max-flow min-cut property does not necessarily hold. Nevertheless, recent results [3] show that, under some restrictions, this property holds for countably infinite graphs, and these restrictions are met by the graphs that are relevant to us. Hence, our proof for finite p-spaces extends to (countably) infinite p-spaces. We also have a direct proof that is based only on the finite variant of max-flow min-cut.

4. UNIVERSAL P-SOLUTIONS AND QUERY ANSWERING

In this section, we generalize the concepts of a *universal solution*, and that of answering target queries, to the probabilistic setting.

4.1 Universal p-Solutions

Recall that the notion of a probabilistic match provides a systematic way of extending any binary relationship between (deterministic) objects to a relationship between probability spaces thereof. In the case of universal solutions, this is applied as follows. Consider a schema mapping $\mathcal{M}$. Denote by $\mathsf{USOL}_{\mathcal{M}}$ the relationship between pairs I and J of source and target instances, respectively, such that $\mathsf{USOL}_{\mathcal{M}}(I, J)$ holds if and only if J is a universal solution for I. Then a *universal p-solution* is defined as follows.

DEFINITION 4.1. ***(Universal p-Solution)*** Let $\mathcal{M}$ be a schema mapping. Let $\tilde{\mathcal{I}}$ and $\tilde{\mathcal{J}}$ be source and target p-instances, respectively. We say that $\tilde{\mathcal{J}}$ is a *universal p-solution* (*for* $\tilde{\mathcal{I}}$ *w.r.t.* Σ) if there is a $\mathsf{USOL}_{\mathcal{M}}$-match of $\tilde{\mathcal{I}}$ in $\tilde{\mathcal{J}}$. □

EXAMPLE 4.2. The $\mathsf{SOL}_{\mathcal{M}}$-match of $\tilde{\mathcal{I}}$ in $\tilde{\mathcal{J}}_1$ (where $\mathcal{M}$, $\tilde{\mathcal{I}}$ and $\tilde{\mathcal{J}}_1$ are described in Example 3.1) on the left side of Figure 2 is actually a $\mathsf{USOL}_{\mathcal{M}}$-match, since an edge from I_m to J_n has a nonzero probability only if J_n is a universal solution for I_m. Thus, $\tilde{\mathcal{J}}_1$ is a universal p-solution for $\tilde{\mathcal{I}}$. The $\mathsf{SOL}_{\mathcal{M}}$-match of $\tilde{\mathcal{I}}$ in $\tilde{\mathcal{J}}_2$ on the right side of Figure 2 is not a $\mathsf{USOL}_{\mathcal{M}}$-match since, for example, there is an edge (with probability 0.1) between I_2 and J_6, yet J_6 is not a universal solution for I_2. Later, we will show that $\tilde{\mathcal{J}}_2$ is, indeed, *not* a universal p-solution for $\tilde{\mathcal{I}}$. □

We now give a proposition about the existence of a p-solution and a universal p-solution. This proposition is straightforward, and we record it for later use.

PROPOSITION 4.3. *Let $\mathcal{M}$ be a schema mapping and let $\tilde{\mathcal{I}}$ be a source p-instance. A p-solution exists if and only if a solution exists for all $I \in \Omega_+(\tilde{\mathcal{I}})$. Similarly, a universal p-solution exists if and only if a universal solution exists for all $I \in \Omega_+(\tilde{\mathcal{I}})$.*

We note that when a p-solution for a source p-instance $\tilde{\mathcal{I}}$ exists, there is a straightforward construction of a p-solution that is left-trivial. A similar comment holds for universal p-solutions.

In the deterministic case, a universal solution is deemed a good choice of a solution, since it is a *most general* one, where the notion of generality is defined by means of a homomorphism; that is, J_1 *generalizes* J_2 if $J_1 \rightarrow J_2$. We would like to have a similar characterization of a universal p-solution. For that, we need a notion for a relationship between p-instances that corresponds to that of homomorphism in ordinary data. One such definition can be obtained by applying the probabilistic match. Let $\mathbf{T}$ be a schema. We denote by $\mathsf{HOM}_{\mathbf{T}}$ the binary relation that includes all the pairs $(J_1, J_2) \in (\mathsf{Inst}(\mathbf{T}))^2$, such that $J_1 \rightarrow J_2$. Consider two p-instances $\tilde{\mathcal{J}}_1$ and $\tilde{\mathcal{J}}_2$ over $\mathbf{T}$. We use $\tilde{\mathcal{J}}_1 \xrightarrow{\text{mat}} \tilde{\mathcal{J}}_2$ to denote that there is a $\mathsf{HOM}_{\mathbf{T}}$-match of $\tilde{\mathcal{J}}_1$ in $\tilde{\mathcal{J}}_2$.

REMARK 4.4. The definition of $\tilde{\mathcal{J}}_1 \xrightarrow{\text{mat}} \tilde{\mathcal{J}}_2$, restricted to finite p-instances, is similar yet different from that of *homomorphism* given in [13] where, in our terminology, only right-trivial $\mathsf{HOM}_{\mathbf{T}}$-matches are allowed (in particular, there is no homomorphism from $\tilde{\mathcal{J}}_1$ to $\tilde{\mathcal{J}}_2$ in the sense of [13] if the cardinality of $\Omega_+(\tilde{\mathcal{J}}_1)$ is strictly larger than that of $\Omega_+(\tilde{\mathcal{J}}_2)$). □

The $\mathsf{HOM}_{\mathbf{T}}$-match extends the notion of homomorphism to p-instances in the systematic way of applying the probabilistic match. There are, though, other natural ways of generalizing this notion. Next, we consider two such ways, which are based on the classical notion of a *stochastic order*. We then explore their relationships to the existence of a $\mathsf{HOM}_{\mathbf{T}}$-match. First, we need some definitions.

A stochastic order is traditionally an order over numeric random variables (cf. [45]). Here, we extend this notion from numbers to general preordered elements, in a straightforward manner. Formally, let O be a countable set and let $\preceq$ be a preorder over O (i.e., $\preceq$ is a reflexive and transitive binary relation $\preceq$ over O). The *stochastic extension* of $\preceq$ is the preorder $\preceq'$ over the set of all the p-spaces over O, where for all p-spaces $\tilde{\mathcal{U}}$ and $\tilde{\mathcal{W}}$, the interpretation of $\tilde{\mathcal{U}} \preceq' \tilde{\mathcal{W}}$ is

$$\forall\, o \in O \ (\Pr(\mathcal{U} \preceq o) \geq \Pr(\mathcal{W} \preceq o)) \ .$$

Let $\mathbf{T}$ be a schema. It is well known that the existence-of-a-homomorphism relationship can be viewed as a preorder over $\mathsf{Inst}(\mathbf{T})$ (see, e.g., [26]), and there are basically two ways to define this preorder. In the first, we use the preorder $\preceq_{\mathsf{sp}}$, where $J \preceq_{\mathsf{sp}} J'$ is interpreted as $J \rightarrow J'$, namely, "J is at most as *specific* as J'." The second preorder, $\preceq_{\mathsf{ge}}$, has the complement interpretation: "J is at most as *general* as J'," that is, $J \preceq_{\mathsf{ge}} J'$ means $J' \rightarrow J$. Having the two preorders $\preceq_{\mathsf{sp}}$ and $\preceq_{\mathsf{ge}}$ over instances, we automatically obtain two preorders over p-instances, namely, the stochastic extensions, which we denote by $\overset{\preceq_{\mathsf{sp}}}{\Longrightarrow}$ and $\overset{\preceq_{\mathsf{ge}}}{\Longleftarrow}$, respectively.[2] Thus, $\tilde{\mathcal{J}}_1 \overset{\preceq_{\mathsf{sp}}}{\Longrightarrow} \tilde{\mathcal{J}}_2$ if $\Pr(\mathcal{J}_1 \rightarrow J) \geq \Pr(\mathcal{J}_2 \rightarrow J)$ for all instances J over $\mathbf{T}$, and $\tilde{\mathcal{J}}_2 \overset{\preceq_{\mathsf{ge}}}{\Longleftarrow} \tilde{\mathcal{J}}_1$ if $\Pr(J \rightarrow \mathcal{J}_2) \geq \Pr(J \rightarrow \mathcal{J}_1)$ for all instances J over $\mathbf{T}$. For uniformity of presentation, we write $\tilde{\mathcal{J}}_1 \overset{\succeq_{\mathsf{ge}}}{\Longrightarrow} \tilde{\mathcal{J}}_2$ instead of $\tilde{\mathcal{J}}_2 \overset{\preceq_{\mathsf{ge}}}{\Longleftarrow} \tilde{\mathcal{J}}_1$.

We now have three ways of extending the relationship $J_1 \rightarrow J_2$ (existence of a homomorphism) from instances J_1 and J_2 to p-instances $\tilde{\mathcal{J}}_1$ and $\tilde{\mathcal{J}}_2$. The first is $\tilde{\mathcal{J}}_1 \xrightarrow{\text{mat}} \tilde{\mathcal{J}}_2$, namely, there exists a $\mathsf{HOM}_{\mathbf{T}}$-match of $\tilde{\mathcal{J}}_1$ in $\tilde{\mathcal{J}}_2$. The second is $\tilde{\mathcal{J}}_1 \overset{\preceq_{\mathsf{sp}}}{\Longrightarrow} \tilde{\mathcal{J}}_2$, namely, $\tilde{\mathcal{J}}_1$ is at most as specific as $\tilde{\mathcal{J}}_2$. The third is $\tilde{\mathcal{J}}_1 \overset{\succeq_{\mathsf{ge}}}{\Longrightarrow} \tilde{\mathcal{J}}_2$, namely, $\tilde{\mathcal{J}}_1$ is at least as general as $\tilde{\mathcal{J}}_2$. Observe that the three are indeed extensions of $\rightarrow$, in the following sense. If $\tilde{\mathcal{J}}_1$ and $\tilde{\mathcal{J}}_2$ are deterministic instances J_1 and J_2 (i.e., the probability of J_i in $\tilde{\mathcal{J}}_i$ is 1, for $i = 1, 2$), then each of $\tilde{\mathcal{J}}_1 \xrightarrow{\text{mat}} \tilde{\mathcal{J}}_2$, $\tilde{\mathcal{J}}_1 \overset{\preceq_{\mathsf{sp}}}{\Longrightarrow} \tilde{\mathcal{J}}_2$ and $\tilde{\mathcal{J}}_1 \overset{\succeq_{\mathsf{ge}}}{\Longrightarrow} \tilde{\mathcal{J}}_2$ is equivalent to $J_1 \rightarrow J_2$.

The following theorem shows that $\xrightarrow{\text{mat}}$ is a strictly stronger relationship than $\overset{\preceq_{\mathsf{sp}}}{\Longrightarrow}$ and $\overset{\succeq_{\mathsf{ge}}}{\Longrightarrow}$; that is, $\tilde{\mathcal{J}}_1 \xrightarrow{\text{mat}} \tilde{\mathcal{J}}_2$ implies both $\tilde{\mathcal{J}}_1 \overset{\preceq_{\mathsf{sp}}}{\Longrightarrow} \tilde{\mathcal{J}}_2$ and $\tilde{\mathcal{J}}_1 \overset{\succeq_{\mathsf{ge}}}{\Longrightarrow} \tilde{\mathcal{J}}_2$, and there are cases where neither of the opposite implications holds. Moreover, it shows that $\overset{\preceq_{\mathsf{sp}}}{\Longrightarrow}$ and $\overset{\succeq_{\mathsf{ge}}}{\Longrightarrow}$ are incomparable. Finally, the theorem shows that for finite p-instances $\tilde{\mathcal{J}}_1$ and $\tilde{\mathcal{J}}_2$, testing $\tilde{\mathcal{J}}_1 \overset{\preceq_{\mathsf{sp}}}{\Longrightarrow} \tilde{\mathcal{J}}_2$ and testing $\tilde{\mathcal{J}}_1 \overset{\succeq_{\mathsf{ge}}}{\Longrightarrow} \tilde{\mathcal{J}}_2$ are not even in the same complexity class as testing $\tilde{\mathcal{J}}_1 \xrightarrow{\text{mat}} \tilde{\mathcal{J}}_2$ (assuming NP $\neq$ coNP) since the first two tests are DP-hard[3] (yet decidable) while the third is NP-complete.

THEOREM 4.5. *The following hold.*

1. *For all p-instances $\tilde{\mathcal{J}}_1$ and $\tilde{\mathcal{J}}_2$, if $\tilde{\mathcal{J}}_1 \xrightarrow{\text{mat}} \tilde{\mathcal{J}}_2$ then $\tilde{\mathcal{J}}_1 \overset{\preceq_{\mathsf{sp}}}{\Longrightarrow} \tilde{\mathcal{J}}_2$ and $\tilde{\mathcal{J}}_1 \overset{\succeq_{\mathsf{ge}}}{\Longrightarrow} \tilde{\mathcal{J}}_2$.*
2. *There are p-instances $\tilde{\mathcal{J}}_1$ and $\tilde{\mathcal{J}}_2$, such that $\tilde{\mathcal{J}}_1 \overset{\preceq_{\mathsf{sp}}}{\Longrightarrow} \tilde{\mathcal{J}}_2$ and $\tilde{\mathcal{J}}_1 \overset{\succeq_{\mathsf{ge}}}{\not\Longrightarrow} \tilde{\mathcal{J}}_2$; similarly, there are p-instances $\tilde{\mathcal{J}}_1$ and $\tilde{\mathcal{J}}_2$, such that $\tilde{\mathcal{J}}_1 \overset{\succeq_{\mathsf{ge}}}{\Longrightarrow} \tilde{\mathcal{J}}_2$ and $\tilde{\mathcal{J}}_1 \overset{\preceq_{\mathsf{sp}}}{\not\Longrightarrow} \tilde{\mathcal{J}}_2$. Hence, due to Part 1, neither $\overset{\preceq_{\mathsf{sp}}}{\Longrightarrow}$ nor $\overset{\succeq_{\mathsf{ge}}}{\Longrightarrow}$ implies $\xrightarrow{\text{mat}}$.*

[2] The choice of the notation $\overset{\preceq_{\mathsf{sp}}}{\Longrightarrow}$ and $\overset{\preceq_{\mathsf{ge}}}{\Longleftarrow}$ (rather than, e.g., $\preceq'_{\mathsf{sp}}$ and $\preceq'_{\mathsf{ge}}$) is for clarity of presentation.

[3] Recall that DP is the class of problems that can be formed as a difference of two problems in NP [37].

3. *Testing $\tilde{\mathcal{J}}_1 \xrightarrow{\text{mat}} \tilde{\mathcal{J}}_2$, given two finite p-instances $\tilde{\mathcal{J}}_1$ and $\tilde{\mathcal{J}}_2$, is in NP.*[4]

4. *Testing each of $\tilde{\mathcal{J}}_1 \xrightarrow{\preceq_{\text{sp}}} \tilde{\mathcal{J}}_2$ and $\tilde{\mathcal{J}}_2 \xrightarrow{\succeq_{\text{ge}}} \tilde{\mathcal{J}}_1$, given finite p-instances $\tilde{\mathcal{J}}_1$ and $\tilde{\mathcal{J}}_2$, is in EXPTIME and NEXPTIME, respectively, and there is a schema $\mathbf{T}$ over which both tests are DP-hard.*

We can now give three additional definitions of a universal p-solution as a most general p-solution, where generality is according to each of the three relationships $\xrightarrow{\text{mat}}$, $\xrightarrow{\preceq_{\text{sp}}}$ and $\xrightarrow{\succeq_{\text{ge}}}$. Theorem 4.5 shows that the three relationships between p-instances are inherently different; hence, we might expect to get *different* definitions of a universal p-solution. Surprisingly, it turns out that all three definitions are equivalent to existence of a $\mathsf{USOL}_{\mathcal{M}}$-match! This is shown in the following theorem. This theorem also shows that, for a solution $\tilde{\mathcal{J}}$, either *all* $\mathsf{SOL}_{\mathcal{M}}$-matches are $\mathsf{USOL}_{\mathcal{M}}$-matches (and then $\tilde{\mathcal{J}}$ is universal) or *none* of them is a $\mathsf{USOL}_{\mathcal{M}}$-match.

THEOREM 4.6. *Let $\mathcal{M}$ be a schema mapping. Let $\tilde{\mathcal{I}}$ be a source p-instance and let $\tilde{\mathcal{J}}$ be a p-solution. The following are equivalent.*

1. *$\tilde{\mathcal{J}}$ is a universal p-solution (i.e., there is a $\mathsf{USOL}_{\mathcal{M}}$-match of $\tilde{\mathcal{I}}$ in $\tilde{\mathcal{J}}$).*
2. *$\tilde{\mathcal{J}} \xrightarrow{\text{mat}} \tilde{\mathcal{J}}'$ for all p-solutions $\tilde{\mathcal{J}}'$.*
3. *$\tilde{\mathcal{J}} \xrightarrow{\preceq_{\text{sp}}} \tilde{\mathcal{J}}'$ for all p-solutions $\tilde{\mathcal{J}}'$.*
4. *$\tilde{\mathcal{J}} \xrightarrow{\succeq_{\text{ge}}} \tilde{\mathcal{J}}'$ for all p-solutions $\tilde{\mathcal{J}}'$.*
5. *Every $\mathsf{SOL}_{\mathcal{M}}$-match of $\tilde{\mathcal{I}}$ in $\tilde{\mathcal{J}}$ is a $\mathsf{USOL}_{\mathcal{M}}$-match.*

In Section 4.2.1, we give a query-based characterization of a universal p-solution (Proposition 4.9). Taken together with Theorem 4.6, these results show that the notion of a universal p-solution is remarkably robust.

4.2 Query Answering

We now generalize the concept of answering target queries in data exchange. A k-ary *query* over a schema $\mathbf{R}$ is function Q that maps every instance $J \in \mathsf{Inst}(\mathbf{R})$ to a set $Q(J) \subseteq \mathsf{dom}(J)^k$, such that Q is invariant under isomorphism of instances. Note that for $k = 0$, the result $Q(J)$ is either $\{()\}$ (denoted **true**) or $\emptyset$ (denoted **false**). Such a query is called *Boolean*. A *conjunctive query* (abbrev. *CQ*) and a *union of conjunctive queries* (abbrev. *UCQ*) are special cases of queries. For completeness, we next formally define a CQ and a UCQ.

A CQ has the form $\exists \mathbf{y} \varphi(\mathbf{x}, \mathbf{y}, \mathbf{c})$, where $\mathbf{x}$ and $\mathbf{y}$ are tuples of variables, $\mathbf{c}$ is a tuple of constants (from Const) and $\varphi(\mathbf{x}, \mathbf{y}, \mathbf{c})$ is a conjunction of atomic formulas over the schema $\mathbf{R}$. We make the safety requirement that all the variables of $\mathbf{x}$ must participate in $\varphi(\mathbf{x}, \mathbf{y}, \mathbf{c})$. A UCQ has the form $\exists \mathbf{y}(\varphi_1(\mathbf{x}, \mathbf{y}, \mathbf{c}) \vee \cdots \vee \varphi_k(\mathbf{x}, \mathbf{y}, \mathbf{c}))$, where $\exists \mathbf{y}(\varphi_i(\mathbf{x}, \mathbf{y}, \mathbf{c}))$ is a CQ for all $1 \leq i \leq k$. Given an instance K over $\mathbf{R}$, the set $Q(K)$ of answers comprises all the possible assignments for $\mathbf{x}$ that result in a clause that is true over K.

We follow the conventional notion [9–11] of querying probabilistic databases. Thus, for a query Q and a p-instance $\tilde{\mathcal{K}}$ (where both Q and $\tilde{\mathcal{K}}$ are over a schema $\mathbf{R}$), every tuple $\mathbf{a} \in (\mathsf{Const} \cup \mathsf{Var})^k$ has a *confidence* value, which is the probability $\Pr(\mathbf{a} \in Q(\mathcal{K}))$. In practice, the tuples $\mathbf{a}$ often come from some finite[5] set of possible answers, which can be given to the user (along with the confidence values); alternatively, the user may request k answers with the top probabilities [38].

Let $(\mathbf{S}, \mathbf{T}, \Sigma)$ be a schema mapping and let Q be a k-ary query over $\mathbf{T}$. In the deterministic case, *answering* Q means that, given a (deterministic) source instance I, we produce the *certain answers*, namely, the tuples $\mathbf{a} \in \mathsf{Const}^k$ that belong to $Q(J)$ for all solutions J for I. We denote this set by $\mathrm{certain}(Q, I, \Sigma)$. Next, we generalize the concept of certain answers to the case of probabilistic source instances. Let $\tilde{\mathcal{I}}$ be a source p-instance. Given $\mathbf{a}$, each p-solution $\tilde{\mathcal{J}}$ gives a (possibly different) probability $\Pr(\mathbf{a} \in Q(\mathcal{J}))$. Consistent with the deterministic case, we would like to characterize $\mathbf{a}$ with a property that is guaranteed in every p-solution. Therefore, we define the *confidence* of $\mathbf{a}$, denoted $conf_Q(\mathbf{a})$, as follows. If there are no solutions, then $conf_Q(\mathbf{a}) = 1$. Otherwise, it is the infimum of the confidences (probabilities) of $\mathbf{a}$ over all the p-solutions, namely,

$$conf_Q(\mathbf{a}) \stackrel{\text{def}}{=} \inf_{\text{p-solutions } \tilde{\mathcal{J}}} \Pr\left(\mathbf{a} \in Q(\mathcal{J})\right) .$$

If Q is Boolean, we write $conf_Q$ instead of $conf_Q(())$.

The following proposition shows that the confidence of an answer $\mathbf{a}$ is the same as the probability that $\mathbf{a}$ is certain in a random source instance (given that a p-solution exists). This equality is interesting, because the two numbers describe apparently different quantities: one is the infimum, over all p-solutions, of the probability of an event defined over the p-solutions (specifically, the probability of having $\mathbf{a}$ as an answer), whereas the other is the probability of an event defined over the source p-instance (specifically, the probability of having $\mathbf{a}$ in the certain answers). In particular, this proposition shows the robustness of our generalization of the notion of target-query answering.

PROPOSITION 4.7. *Let $(\mathbf{S}, \mathbf{T}, \Sigma)$ be a schema mapping, let Q be a query over $\mathbf{T}$, and let $\tilde{\mathcal{I}}$ be a source p-instance, such that a p-solution exists. For all tuples $\mathbf{a}$ of constants,*

$$conf_Q(\mathbf{a}) = \Pr_{\tilde{\mathcal{I}}}(\mathbf{a} \in \mathrm{certain}(Q, \mathcal{I}, \Sigma)) .$$

As a part of the proof of Proposition 4.7, we construct a p-solution $\tilde{\mathcal{J}}$, such that $\Pr(\mathbf{a} \in Q(\mathcal{J}))$ is equal to the probability on the right-hand side of the equality. Thus, the infimum in the definition of confidence is always realized by some p-solution (hence, it can be replaced with *minimum*).

EXAMPLE 4.8. Consider again the schema mapping $\mathcal{M}$, and the p-instances $\tilde{\mathcal{I}}$, $\tilde{\mathcal{J}}_1$ and $\tilde{\mathcal{J}}_2$ of Example 3.1. Recall from Example 3.5 that both $\tilde{\mathcal{J}}_1$ and $\tilde{\mathcal{J}}_2$ are p-solutions. Let Q be the following target CQ, which extracts all the universities where both IR and AI research is conducted.

$$Q(u) \text{:-} \exists d_1, d_2 \left(UArea(u, d_1, \mathsf{IR}) \wedge UArea(u, d_2, \mathsf{AI}) \right)$$

For $\tilde{\mathcal{J}}_1$, there is only one possible answer, which is $\mathbf{a} = (\mathsf{UCSD})$. Since $\Pr(\mathbf{a} \in Q(\mathcal{J}_1)) = 0.3$, we get that $conf_Q(\mathbf{a}) \leq 0.3$. Hence, the value of the left-hand side of the equality in Proposition 4.7 is at most 0.3. What about the right-hand side, which is the probability that $\mathbf{a}$ is a certain answer? Since $\mathbf{a}$ is a certain answer only for I_2, and I_2 has the probability 0.3, the right-hand side of the equality in Proposition 4.7 is 0.3. Hence, by Proposition 4.7, $conf_Q(\mathbf{a})$ is 0.3, and so is realized by $\tilde{\mathcal{J}}_1$; that is, $\tilde{\mathcal{J}}_1$ is a p-solution $\tilde{\mathcal{J}}$ such that $\Pr(\mathbf{a} \in Q(\mathcal{J}))$ is minimal. In contrast, for $\tilde{\mathcal{J}}_2$ we have $\Pr(\mathbf{a} \in Q(\mathcal{J}_2)) = 0.65$, which is strictly larger than $conf_Q(\mathbf{a})$. □

Earlier, in Example 4.2, we noted that the $\mathsf{SOL}_{\mathcal{M}}$-match of $\tilde{\mathcal{I}}$ in $\tilde{\mathcal{J}}_2$ on the right side of Figure 2 is not a $\mathsf{USOL}_{\mathcal{M}}$-match. Thus, by Part 5 of Theorem 4.6, $\tilde{\mathcal{J}}_2$ is *not* a universal p-solution for $\tilde{\mathcal{I}}$.

[4] Recall that there are fixed schemas over which testing $\tilde{\mathcal{J}}_1 \xrightarrow{\text{mat}} \tilde{\mathcal{J}}_2$ is NP-hard even if $\tilde{\mathcal{J}}_1$ and $\tilde{\mathcal{J}}_2$ are deterministic [8].

[5] This is the case when $\tilde{\mathcal{K}}$ is a finite p-space.

Moreover, recall from Example 4.8 that $\Pr(\mathbf{a} \in Q(\mathcal{J}_2))$ is strictly larger than $conf_Q(\mathbf{a})$. The following section shows how this latter fact gives another proof that $\tilde{\mathcal{J}}_2$ is not universal.

4.2.1 UCQs over Universal p-Solutions

In the deterministic case, a universal solution can be used for answering target UCQs in the sense that the result of applying the query to the universal solution (and then restricting to the tuples of constants) is the set of all certain answers [15]. Moreover, a solution that has this property for every CQ is necessarily universal [15]. The following proposition shows that, although the concepts of deterministic and probabilistic query answering are inherently different, this property of universal solutions generalizes to universal p-solutions. That is, the confidence of an answer for a UCQ is obtained by querying a universal p-solution (when one exists), and a p-solution that has this property for every (Boolean) CQ is necessarily universal. The second part is proved by using the third characterization of Theorem 4.6.

PROPOSITION 4.9. *Let $(\mathbf{S}, \mathbf{T}, \Sigma)$ be a schema mapping and let $\tilde{\mathcal{I}}$ be a source p-instance. The following hold.*

1. *If $\tilde{\mathcal{J}}$ is a universal p-solution and Q is a UCQ over $\mathbf{T}$, then $conf_Q(\mathbf{a}) = \Pr(\mathbf{a} \in Q(\mathcal{J}))$ for all tuples $\mathbf{a}$ of constants.*
2. *If $\tilde{\mathcal{J}}$ is a p-solution such that $conf_Q = \Pr(Q(\mathcal{J}))$ holds for all Boolean CQs Q, then $\tilde{\mathcal{J}}$ is a universal p-solution.*

In the next section, we study computational aspects of probabilistic data exchange. In particular, we consider the tasks of testing whether a (universal) p-solution exists, materializing one (when it exists), and evaluating target UCQs. By Proposition 4.3, a p-solution exists if and only if there is a solution for I w.r.t. Σ for all $I \in \Omega_+(\tilde{\mathcal{I}})$. By the discussion that follows Proposition 4.3, if a p-solution exists, then we can materialize one, using solutions for the instances of $\Omega_+(\tilde{\mathcal{I}})$, by a straightforward construction. A similar comment applies to universal (p-) solutions. Proposition 4.7 implies that we can compute $conf_Q(\mathbf{a})$ by determining whether $\mathbf{a} \in \mathrm{certain}(Q, I, \Sigma)$ for each $I \in \Omega_+(\tilde{\mathcal{I}})$, and taking the sum of the probabilities of the instances I for which the answer is "yes."

Consequently, in the case of finite p-instances, these tasks in the probabilistic setting are not harder than their traditional counterparts. Nevertheless, this analysis is based on the assumption that source p-instances are represented in an explicit manner (i.e., by specifying each possible instance along with its probability). This is not a practical assumption, as evidenced by existing models of probabilistic databases (e.g., [2, 6, 10, 11, 43]) that usually employ a (typically logarithmic-scale) compact encoding of the possible worlds. So, the next section studies the above computational problems under some typical compact representations of probabilistic databases.

5. COMPACT REPRESENTATION

In this section, we explore complexity aspects of data exchange in a concrete setting where dependencies are in the form of *tgds* and *egds* [5, 15] (the formal definitions are in Section 5.2), and p-instances are represented compactly by annotating facts with probabilistic *conditions* [19, 23, 24] rather than explicitly specifying the whole probability space.

5.1 Annotated Instances

We consider p-instances that are represented by means of *Boolean pc-tables* [24] (which are the probabilistic version of *c-tables* [27])

I_p^α

	Fact f	Condition $\alpha(f)$
r_{e}	*Researcher*(Emma, UCSD)	$\mathbf{true}$
r_{j}	*Researcher*(John, UCSD)	$e_1 \vee e_2 \vee e_3 \vee e_4$
a_{eir}	*RArea*(Emma, IR)	$e_1 \vee e_2$
a_{edb}	*RArea*(Emma, DB)	$\neg e_1 \wedge \neg e_2$
a_{jdb}	*RArea*(John, DB)	$e_1 \vee (\neg e_2 \wedge \neg e_3 \wedge e_4)$
a_{jai}	*RArea*(John, AI)	$(\neg e_1 \wedge e_2) \vee (\neg e_1 \wedge e_3)$

$$\mathsf{EVar}(\alpha) = \{e_1, e_2, e_3, e_4\}$$

$$p : \mathsf{EVar}(\alpha) \to [0, 1]$$
$$p(e_1) = 3/10, \quad p(e_2) = 3/7, \quad p(e_3) = p(e_4) = 1/2$$

Figure 3: A DNF instance I_p^α

where the condition assigned to each fact is a logical formula over *event variables*—probabilistically independent Boolean (Bernoulli) random variables. In pc-tables conditions can be phrased as arbitrary propositional-logic formulas, which renders the most basic operations as intractable, since, for one, it is NP-complete even to decide whether a given fact occurs with a nonzero probability. Thus, our focus is on two restricted representations that correspond to (or subsume) various representations in the literature. In the first, conditions are in disjunctive normal form (DNF), and in the second, the facts are probabilistically independent. Next, we give the formal definitions.

We assume an infinite set EVar of *event variables*. Let $\mathbf{R}$ be a schema. A *DNF instance* (*over* $\mathbf{R}$) comprises an instance I over $\mathbf{R}$, a function α that maps every fact f of I to a DNF formula $\alpha(f)$ over EVar, and a function $p : \mathsf{EVar}(\alpha) \to [0, 1]$ where $\mathsf{EVar}(\alpha)$ is the set of all the event variables that appear in the image of α. The DNF instance given by I, α and p is denoted by I_p^α. A DNF instance I_p^α naturally encodes a p-instance, which we denote by $\text{p-space}(I_p^\alpha)$, where a sample I' is obtained as follows. First, a random truth assignment $\tau : \mathsf{EVar}(\alpha) \to \{\mathbf{true}, \mathbf{false}\}$ is chosen for the event variables of I; this assignment is obtained by independently picking a random Boolean value $\tau(e)$, with probability $p(e)$ for $\mathbf{true}$, for each member e of $\mathsf{EVar}(\alpha)$. Second, all the facts f such that τ satisfies the formula $\alpha(f)$ are selected as members of I' (alternatively, I' is obtained from I by removing all the facts f such that $\alpha(f)$ is violated). Thus, $\text{p-space}(I_p^\alpha)$ is the finite p-instance $\tilde{\mathcal{I}}$ such that $\Omega_+(\tilde{\mathcal{I}})$ comprises instances with facts from I, and for all $I' \subseteq I$ the probability $p_{\tilde{\mathcal{I}}}(I')$ is that of obtaining I' in the above process (namely, the sum of the probabilities of all the assignments

I_p^α

	Fact f	$\alpha(f)$	$p(\alpha(f))$
r_{e}	*Researcher*(Emma, UCSD)	e'_0	1.0
r_{j}	*Researcher*(John, UCSD)	e'_1	0.9
a_{eir}	*RArea*(Emma, IR)	e'_2	0.6
a_{edb}	*RArea*(Emma, DB)	e'_3	0.4
a_{jdb}	*RArea*(John, DB)	e'_4	0.4
a_{jai}	*RArea*(John, AI)	e'_5	0.5

Figure 4: A tuple-independent instance I_p^α

that satisfy every formula of I' and none of $I \setminus I'$).

EXAMPLE 5.1. Figure 3 depicts a DNF instance I_p^α. The table on the top of the figure has a row for each fact, and the right column contains the condition of the corresponding fact. As shown in the middle part of the figure, $\mathsf{EVar}(\alpha)$ contains the four event variables $e_1, \ldots, e_4$. Finally, the function p is specified in the bottom.

Note that the facts of I are those that are depicted in the upper row of Figure 1, that is, the facts of the p-instance $\tilde{\mathcal{I}}$. The reader can verify that I_p^α encodes[6] *exactly* the p-instance $\tilde{\mathcal{I}}$; that is, $\tilde{\mathcal{I}} = \text{p-space}(I_p^\alpha)$ (which means that $\tilde{\mathcal{I}}$ and $\text{p-space}(I_p^\alpha)$ have the same support, and the same probability for each instance in their support). As an example, let us compute the probability of the instance $I_5 = \{r_\mathsf{e}, a_\mathsf{edb}\}$ (from Figure 1). In general, an instance can be produced by multiple truth assignments, but I_5 is produced by only the assignment that maps all four variables to **false**, because $r_\mathsf{j} \notin I_5$. Let τ be that assignment. Observe that τ indeed produces I_5 since it violates the condition of every fact other than r_e and a_edb. Therefore, the probability of I_5 is the probability of τ, namely, $\frac{7}{10} \times \frac{4}{7} \times \frac{1}{2} \times \frac{1}{2} = \frac{28}{280} = 0.1$. As another example, the reader can verify that the assignments τ that map e_1 to **true** are exactly those that result in the instance $I_1 = \{r_\mathsf{e}, r_\mathsf{j}, a_\mathsf{eir}, a_\mathsf{jdb}\}$; therefore, the probability of I_1 is $p(e_1) = 0.3$. □

In [24] it is shown that every finite p-instance can be represented by means of Boolean pc-tables (i.e., Boolean pc-tables are "complete"). In particular, every finite p-instance $\tilde{\mathcal{I}}$ is equal to $\text{p-space}(I_p^\alpha)$ for some DNF instance I_p^α, since every formula in propositional logic can be transformed into DNF. Note that this translation may entail an exponential blowup. But, one can efficiently translate into DNF instances other representations like *block-independent disjoint* databases [38–40] and *probabilistic rdb's* [31].

A special case of a DNF instance is one where tuples are probabilistically independent. Formally, a *tuple-independent* instance is a DNF instance I_p^α, such that for all facts $f \in I$, the condition $\alpha(f)$ is a *distinct* atomic event variable e_f (i.e., $e_f \neq e_g$ for $f \neq g$); in particular, the facts of I_p^α are probabilistically independent. We require a tuple-independent instance I_p^α to be such that the function p is strictly positive (i.e., $p(e) > 0$ for all $e \in \mathsf{EVar}(\alpha)$). This is not a restriction, since a fact with zero-probability event can simply be removed.

EXAMPLE 5.2. Figure 4 depicts a tuple-independent instance I_p^α. Each row shows a fact f, the unique variable $e'_i = \alpha(f)$ and the probability $p(e'_i)$. The facts of Figure 4 are the same as those of Figure 1 (and those of Figure 3, which is discussed in Example 5.1). Let $\tilde{\mathcal{I}}$ be as in Figure 1. The probability of each fact f in I_p^α is the marginal probability of f in $\tilde{\mathcal{I}}$ (i.e., $p(\alpha(f))$ is the sum of the probabilities of the instances $I \in \Omega_+(\tilde{\mathcal{I}})$ with $f \in I$). However, unlike Figure 3, the instance I_p^α of Figure 4 does *not* encode $\tilde{\mathcal{I}}$ (that is, $\text{p-space}(I_p^\alpha) \neq \tilde{\mathcal{I}}$). Moreover, no tuple-independent instance encodes $\tilde{\mathcal{I}}$, simply because the facts of $\tilde{\mathcal{I}}$ are not independent. As an example, the facts a_eir and a_edb are mutually exclusive in $\tilde{\mathcal{I}}$ (hence, they are not independent). □

In terms of representations of probabilistic data in the literature, tuple-independent instances are sets of *p-?-tables* [24], and they are the same as the *tuple-independent probabilistic structures* of [11] (called *probabilistic databases* in [12]). We could avoid using event variables in tuple-independent instances, and just write a number next to each fact (as done in [11, 12]). However, it is convenient for us to syntactically view these instances as special cases of DNF instances.

Consider a schema mapping $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$. A *source DNF instance* is a DNF instance I_p^α over $\mathbf{S}$, such that I is a ground instance, and a *target DNF instance* is a DNF instance J_q^β over $\mathbf{T}$ (J is not necessarily ground). Special cases are source and target tuple-independent instances. Clearly, if I_p^α and J_q^β are source and target DNF instances, then $\text{p-space}(I_p^\alpha)$ and $\text{p-space}(J_q^\beta)$ are source and target p-instances, respectively.

5.2 Tuple/Equality-Generating Dependencies

We consider two specific types of dependencies that were studied in past research on data exchange (e.g., [15, 16]); each dependency is a *tuple-generating dependency* (*tgd*) or an *equality-generating dependency* (*egd*) [5]. More particularly, let $(\mathbf{S}, \mathbf{T}, \Sigma)$ be a schema mapping. A *source-to-target tgd* (*st-tgd*) is a formula of the form

$$\forall \mathbf{x} \left(\varphi_\mathbf{S}(\mathbf{x}) \rightarrow \exists \mathbf{y} \psi_\mathbf{T}(\mathbf{x}, \mathbf{y})\right) ,$$

a *target tgd* (*t-tgd*) is one of the form

$$\forall \mathbf{x} \left(\varphi_\mathbf{T}(\mathbf{x}) \rightarrow \exists \mathbf{y} \psi_\mathbf{T}(\mathbf{x}, \mathbf{y})\right) ,$$

and a *target egd* (*t-egd*) has the form

$$\forall \mathbf{x} \left(\varphi_\mathbf{T}(\mathbf{x}) \rightarrow (x_1 = x_2)\right) .$$

In the above formulas, $\varphi_\mathbf{S}(\mathbf{x})$ is a conjunction of atomic formulas over $\mathbf{S}$, and each of $\varphi_\mathbf{T}(\mathbf{x})$ and $\psi_\mathbf{T}(\mathbf{x}, \mathbf{y})$ is a conjunction of atomic formulas over $\mathbf{T}$. Moreover, all the variables of $\mathbf{x}$ appear in both $\varphi_\mathbf{S}(\mathbf{x})$ and $\varphi_\mathbf{T}(\mathbf{x})$, and $\mathbf{x}$ contains the variables x_1 and x_2. As a special case, *full* st-tgds and *full* t-tgds are ones that do not contain existentially quantified variables (i.e., $\mathbf{y}$ is empty).

5.3 Complexity Results

We use *data complexity* for analyzing the computational problems that we address. In particular, we assume that the schema mapping $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ is fixed, and the input consists of the source DNF instance I_p^α. If a query Q is involved, then it is fixed as well. For all variables $e \in \mathsf{EVar}(\alpha)$, the number $p(e)$ is a rational number represented by a pair of integers (the numerator and the denominator). Finally, we consider only schema mappings where the set Σ of dependencies is the union of finite sets Σ_1 and Σ_2, such that Σ_1 contains only st-tgds and t-egds, and Σ_2 is a *weakly acyclic* set of t-tgds (see [15] for the formal definition of weak acyclicity).

The complexity results are shown in Table 1. We study five computational problems, and give their complexity for each of the two types of source p-instances: the top five rows of Table 1 consider source DNF instances, and the bottom five rows are for source tuple-independent instances. Each row is associated with a specific problem. Each column corresponds to a class of schema mappings. For example, the column entitled "*full st-tgds, t-egds*" considers schema mappings $(\mathbf{S}, \mathbf{T}, \Sigma)$ such that Σ contains only full st-tgds and t-egds. An upper bound (e.g., "PTIME" or "FP") refers to all schema mappings in the corresponding column, whereas a lower bound (e.g., "no FPRAS if P $\neq$ RP" or "$\notin$ FP if NP $\neq$ RP") means that there *exists* a schema mapping, in the corresponding column, for which the result holds. By "coNP-complete" we mean that the problem is in coNP for all schema mappings in the corresponding column, and there is a schema mapping, in the corresponding column, where the problem is coNP-hard. The meaning of "$\text{FP}^{\#\text{P}}$-complete" is similar (we later give the definition of $\text{FP}^{\#\text{P}}$). Next, we explain the problems we study and the complexity results.

Existence of p-solutions. The first problem is that of deciding whether a p-solution exists. This problem is the same as deciding whether a *universal* p-solution exists, as it follows from [15] and

[6] The translation of $\tilde{\mathcal{I}}$ into I_p^α follows standard techniques of encoding finite p-spaces by annotations (see, e.g., [24, 44]).

I_p^α	Problem	*st-tgds, t-egds, w.a. t-tgds*	*st-tgds, t-egds*	*st-tgds, w.a. t-tgds*	*full st-tgds, t-egds, full t-tgds*	*full st-tgds, full t-tgds*	*full st-tgds, t-egds*	*st-tgds*	*full st-tgds*
DNF	*Existence of a (U.) p-Solution*	coNP-complete	coNP-complete	trivial	coNP-complete	trivial	PTIME	trivial	trivial
	Materializing a p-Solution	$\notin$ FP if P $\neq$ NP	$\notin$ FP if P $\neq$ NP	FP	$\notin$ FP if P $\neq$ NP	FP	FP	FP	FP
	Materializing a U. p-Solution	$\notin$ FP if P $\neq$ NP	$\notin$ FP if P $\neq$ NP	$\notin$ FP if P $\neq$ NP	$\notin$ FP if P $\neq$ NP	$\notin$ FP if P $\neq$ NP	FP	FP	FP
	Target UCQ: Exact	$\mathrm{FP}^{\#\mathrm{P}}$-complete							
	Target UCQ: Approx.	no FPRAS if RP $\neq$ NP	no FPRAS if RP $\neq$ NP	no FPRAS if RP $\neq$ NP	no FPRAS if RP $\neq$ NP	no FPRAS if RP $\neq$ NP	FPRAS	FPRAS	FPRAS
Tuple-Independent	*Existence of a (U.) p-Solution*	PTIME	PTIME	trivial	PTIME	trivial	PTIME	trivial	trivial
	Materializing a p-Solution	FP							
	Materializing a U. p-Solution	$\notin$ FP if RP $\neq$ NP	$\notin$ FP if RP $\neq$ NP	$\notin$ FP if RP $\neq$ NP	$\notin$ FP if RP $\neq$ NP	$\notin$ FP if RP $\neq$ NP	FP	FP	FP
	Target UCQ: Exact	$\mathrm{FP}^{\#\mathrm{P}}$-complete							
	Target UCQ: Approx.	no FPRAS if RP $\neq$ NP	no FPRAS if RP $\neq$ NP	no FPRAS if RP $\neq$ NP	no FPRAS if RP $\neq$ NP	no FPRAS if RP $\neq$ NP	FPAS	FPAS	FPAS

Table 1: Complexity of testing for the existence of a (universal) p-solution, materializing a candidate (universal) p-solution as a DNF instance, and (exact and approximate) evaluation of target UCQs

Proposition 4.3 that (for the class of schema mappings we study) a p-solution exists if and only if a universal one exists. This problem corresponds to the rows of Table 1 entitled "*Existence of a (U.) p-Solution*." By "trivial" we mean that a p-solution always exists. These are the cases where Σ is the union of a set of st-tgds and a weakly acyclic set of t-tgds (and Σ has no t-egds). Observe that for tuple-independent instances, existence of p-solutions is always tractable or trivial. For DNF instances, however, the nontrivial cases are coNP-complete, except for the tractable case where Σ contains full st-tgds and t-egds.

Materialization. The second problem corresponds to the rows entitled "*Materializing a p-Solution*," and is that of materializing a *candidate p-solution*, namely, a target p-instance $\tilde{\mathcal{J}}$ that forms a p-solution *if one exists*. We restrict to generation of candidate p-solutions $\tilde{\mathcal{J}}$ that are represented as DNF instances J_q^β (i.e., $\tilde{\mathcal{J}} = \text{p-space}(J_q^\beta)$). The third problem is the universal version of the second, namely, generation of a *candidate universal p-solution*, and it corresponds to the rows entitled "*Materializing a U. p-Solution*." For these problems, the table contains three types of results: FP,[7] not in FP unless P $=$ NP, and not in FP unless RP $=$ NP.[8]

Table 1 shows that, for source DNF instances, we can sometimes efficiently materialize a candidate universal p-solution (e.g., when Σ has only st-tgds) whereas in other cases we cannot efficiently materialize even a (not necessarily universal) candidate p-solution. If source instances are tuple-independent, then materializing a candidate p-solution is always tractable. However, for materializing candidate universal p-solutions, the intractable cases for source DNF instances remain intractable for tuple-independent instances. The positive results are obtained by combining the chase algorithm [5, 15, 35] with the known concept of maintaining conditions (or provenance) in relational operators, which is used in [23, 24, 27] for showing closure of annotated databases under relational algebra.[9] The lower bounds are proved using the inapproximability of determining the number of assignments satisfying a monotone 2-CNF formula (see, e.g., [49]), and the Monte-Carlo algorithm of [29] as a reduction technique.

Answering target UCQs. The fourth problem is that of evaluating unions of conjunctive queries, and it corresponds to the rows of Table 1 entitled "*Target UCQ: Exact*." Formally, for a schema mapping $(\mathbf{S}, \mathbf{T}, \Sigma)$ and a UCQ Q over $\mathbf{T}$, the problem is the following. Given a source DNF instance I_p^α and a tuple $\mathbf{a}$ of constants, compute $conf_Q(\mathbf{a})$. As shown in the table, in every studied case (even when there are only full st-tgds and source instances are tuple-independent) there is a schema mapping such that this problem is $\mathrm{FP}^{\#\mathrm{P}}$-complete. Recall that $\mathrm{FP}^{\#\mathrm{P}}$ is the class of functions that are efficiently computable using an oracle to some function in #P.[10] Note that a function F is $\mathrm{FP}^{\#\mathrm{P}}$-hard[11] if there is a polynomial-time *Turing reduction* (or *Cook reduction*) from every function in $\mathrm{FP}^{\#\mathrm{P}}$ to F. Actually, we can show even more: in the most restricted case (source tuple-independent instances and only full st-tgds), for every *nontrivial* target UCQ Q there exists a schema mapping such that evaluating Q is $\mathrm{FP}^{\#\mathrm{P}}$-hard, where a *trivial* UCQ is a Boolean UCQ that is equivalent to $\mathbf{true}$. To show this lower bound, we use hardness results of [11, 12]; membership in $\mathrm{FP}^{\#\mathrm{P}}$ is shown by adapting some of the techniques given in [21].

Given this intractability, the best that one can hope for when looking for tractable classes of schema mappings (in terms of target-query evaluation) is an evaluation in an *approximate* manner; in practice, such an evaluation is often good enough. So, the fifth problem is that of approximately evaluating target UCQs, and it is considered in the rows of Table 1 entitled "*Target UCQ: Approx.*" Formally, let $(\mathbf{S}, \mathbf{T}, \Sigma)$ be a schema mapping, and let Q be a UCQ over $\mathbf{T}$. A *fully polynomial randomized approximation scheme* (ab-

[7] FP is the class of polynomial-time computable functions.

[8] RP comprises the sets that are efficiently recognizable by a randomized algorithm with a bounded one-sided error (i.e., the answer may mistakenly be "no"). NP=RP is equivalent to NP$\subseteq$BPP [32] (where BPP comprises the sets that are efficiently recognizable by a randomized algorithm with a bounded two-sided error) and implies that BPP contains the whole polynomial hierarchy [48].

[9] A similar construction is used in [22] for the task of propagating *trust conditions* through data exchange between peers in a network.

[10] #P [47] is the class of functions that count the number of accepting paths of the input of an NP machine.

[11] Using an oracle to a #P-hard (or $\mathrm{FP}^{\#\mathrm{P}}$-hard) function, one can efficiently solve every problem in the polynomial hierarchy [46].

brev. *FPRAS*) for Q is a randomized algorithm A that gets as input a DNF instance I_p^α over $\mathbf{S}$, a tuple $\mathbf{a}$, and a number $\epsilon > 0$, and returns a (random) value $A(I_p^\alpha, \mathbf{a})$ such that

$$\Pr_A\left(\frac{p}{1+\epsilon} \leq A(I_p^\alpha, \mathbf{a}) \leq (1+\epsilon)p\right) \geq \frac{2}{3},$$

where $p = conf_Q(\mathbf{a})$.[12] Moreover, A is required to run in polynomial time in the size of I_p^α and in $1/\epsilon$. An even stronger notion is that of *FPAS*, where the approximation algorithm is deterministic (i.e., the reliability factor $2/3$ is replaced with 1).

Table 1 shows that for source DNF instances, there is an FPRAS for a UCQ when Σ contains only st-tgds, or full st-tgds with t-egds. For such Σ, there is even an FPAS if source instances are tuple-independent. To prove these results, we use techniques for approximating the number of satisfying assignments for a DNF formula [29, 34] (as done in, e.g., [12, 30]). For the rest of the studied cases, there is always a schema mapping Σ and a UCQ Q such that no FPRAS exists unless RP = NP. Actually, this holds even if we fix the approximation ratio ϵ (that is, the running time of the algorithm is no longer required to depend polynomially on $1/\epsilon$). Moreover, this holds for *all* nontrivial UCQs Q, except for the cell of the column entitled "*st-tgds, t-egds*" in the bottom row of the table (in the "tuple-independent" part), where this result holds for all UCQs Q except for *near-trivial* ones. A UCQ Q over a schema $\mathbf{T}$ is *near-trivial* if it is a statement about non-emptiness of the relations; more precisely, it is a Boolean UCQ such that $Q(J_1) = Q(J_2)$ whenever J_1 and J_2 are instances such that $R^{J_1} = \emptyset$ if and only if $R^{J_2} = \emptyset$ for all relation symbols R of $\mathbf{T}$. Note that this notion is weaker than UCQ triviality; this weakening is necessary, since it can be shown that over tuple-independent source instances, there is an FPAS for every near-trivial UCQ if Σ contains only st-tgds and t-egds.

General comments. Table 1 shows that the studied problems are often hard. On the positive side, observe that for the rightmost three columns, all the problems (except for exact query answering) are tractable. Not all the possible combinations of (full) st-tgds, (full) t-tgds and t-egds are mentioned in Table 1. However, this table actually *covers* all possible combinations, in the following sense. Each missing combination lies between two combinations that have the same complexity results in the table. For example, the combination "*st-tgds, full t-tgds*" (which is not in the table) is between "*full st-tgds, full t-tgds*" and "*st-tgds, w.a. t-tgds,*" and the complexity results for these two combinations are exactly the same; hence, these results also hold for the missing "*st-tgds, full t-tgds*."

6. PROBABILISTIC MAPPINGS

In this section, we generalize the framework and results of the previous sections to accommodate uncertainty in the schema mapping. More formally, in this generalization not only is the source data probabilistic, but the set of dependencies specifying the schema mapping is probabilistic as well. Moreover, we will allow the source p-instance and the probabilistic mapping to be arbitrarily correlated. Next, we give the basic definitions. Later, we discuss the generalization of the results of the previous sections to this new setting.

Let $\mathbf{S}$ and $\mathbf{T}$ be two schemas with no relation symbols in common. We assume that there is a fixed countably infinite set $\mathsf{Dep}_{\mathbf{ST}}$ of formulas over $\langle \mathbf{S}, \mathbf{T} \rangle$, such that every set Σ of dependencies specifying a schema mapping is a finite subset of $\mathsf{Dep}_{\mathbf{ST}}$. We denote by $\mathsf{Dep}^*_{\mathbf{ST}}$ the (countable) set of all finite subsets Σ of $\mathsf{Dep}_{\mathbf{ST}}$.

[12]Note that the choice of the reliability factor $2/3$ is arbitrary, since one can improve it to $(1-\delta)$ by taking the median of $O(\log \delta)$ trials [28].

Up until now, we considered schema mappings that are specified by triples $(\mathbf{S}, \mathbf{T}, \Sigma)$ where $\Sigma \in \mathsf{Dep}^*_{\mathbf{ST}}$. Here, as a starting point, we are interested in replacing the fixed Σ with a p-space $\tilde{\mathbf{\Sigma}}$ over $\mathsf{Dep}^*_{\mathbf{ST}}$. Thus, both the source instance $\tilde{\mathcal{I}}$ and the schema mapping $(\mathbf{S}, \mathbf{T}, \tilde{\mathbf{\Sigma}})$ are probabilistic. However, separating the probabilistic schema mapping from the source p-instance necessitates the assumption of probabilistic independence (or some other specific correlation) between the two. In practice, such an assumption is often a limitation. Therefore, in this section we *do not* use these two notions; instead, we use a generalized definition that is based on the notion of a *probabilistic problem* (abbrev. *p-problem*). The formal definition is the following.

DEFINITION 6.1. ***(p-Problem)*** Let $\mathbf{S}$ and $\mathbf{T}$ be schemas without common relation symbols. A *p-problem* (*from* $\mathbf{S}$ *to* $\mathbf{T}$) is a p-space $\tilde{\mathcal{P}}$ over $\mathsf{Dep}^*_{\mathbf{ST}} \times \mathsf{Inst}^{\mathsf{c}}(\mathbf{S})$. □

Observe that the marginals of a p-problem $\tilde{\mathcal{P}}$ define a unique probabilistic schema mapping $(\mathbf{S}, \mathbf{T}, \tilde{\mathbf{\Sigma}})$ and a unique source p-instance $\tilde{\mathcal{I}}$; however, $\tilde{\mathcal{P}}$ is not necessarily the product space of $\tilde{\mathcal{I}}$ and $\tilde{\mathbf{\Sigma}}$.

A p-solution $\tilde{\mathcal{J}}$ for a p-problem $\tilde{\mathcal{P}}$ is defined similarly to the case of a fixed Σ, except that now the probabilistic match is from $\tilde{\mathcal{P}}$ to $\tilde{\mathcal{J}}$ (rather than from the source p-instance $\tilde{\mathcal{I}}$ to $\tilde{\mathcal{J}}$). Formally, given a p-problem $\tilde{\mathcal{P}}$ from $\mathbf{S}$ to $\mathbf{T}$, a target p-instance $\tilde{\mathcal{J}}$ is a *p-solution* (for $\tilde{\mathcal{P}}$) if there is a $\mathsf{dSOL}_{\mathbf{ST}}$-match of $\tilde{\mathcal{P}}$ in $\tilde{\mathcal{J}}$, where $\mathsf{dSOL}_{\mathbf{ST}}$ is the binary relation between pairs (Σ, I) and instances J, such that $I \in \mathsf{Inst}^{\mathsf{c}}(\mathbf{S})$, $J \in \mathsf{Inst}(\mathbf{T})$, $\Sigma \in \mathsf{Dep}^*_{\mathbf{ST}}$, and $\langle I, J \rangle \models \Sigma$. (The letter d in $\mathsf{dSOL}_{\mathbf{ST}}$ denotes that dependencies are involved in the relation.) Similarly, $\tilde{\mathcal{J}}$ is a *universal p-solution* (for $\tilde{\mathcal{P}}$) if there is a $\mathsf{dUSOL}_{\mathbf{ST}}$-match of $\tilde{\mathcal{P}}$ in $\tilde{\mathcal{J}}$, where $\mathsf{dUSOL}_{\mathbf{ST}}$ is the relation between pairs (Σ, I) and instances J such that J is a universal solution for I w.r.t. Σ.

6.1 Generalization of the Results

We now discuss the generalization of our results to the notion of a p-problem. Basically, *all* the results generalize to p-problems. For Sections 3 and 4, this generalization is via a rather mechanical replacement of the p-space $\tilde{\mathcal{I}}$ with the p-space $\tilde{\mathcal{P}}$. Generalizing the results of Section 5 is a little more involved.

We start with the results of Sections 3 and 4. In Theorem 3.6, we need to replace every occurrence of $\tilde{\mathcal{I}}$ with $\tilde{\mathcal{P}}$ and, in addition, the event E of Part 2 is a subset of $\mathsf{Dep}^*_{\mathbf{ST}} \times \mathsf{Inst}^{\mathsf{c}}(\mathbf{S})$ (rather than $\mathsf{Inst}^{\mathsf{c}}(\mathbf{S})$). In Theorem 4.6, we replace the source instance $\tilde{\mathcal{I}}$ with a p-problem $\tilde{\mathcal{P}}$; moreover, the sets $\mathsf{SOL}_{\mathcal{M}}$ and $\mathsf{USOL}_{\mathcal{M}}$ are replaced with $\mathsf{dSOL}_{\mathbf{ST}}$ and $\mathsf{dUSOL}_{\mathbf{ST}}$, respectively. In Proposition 4.7 the probability space $\tilde{\mathcal{I}}$ is replaced with $\tilde{\mathcal{P}}$; that is, $conf_Q(\mathbf{a})$ is equal to the probability that a random pair (Σ, I) of $\tilde{\mathcal{P}}$ is such that $\mathbf{a}$ is a certain answer (i.e., $\mathbf{a} \in \mathrm{certain}(Q, I, \Sigma)$). Finally, Proposition 4.9 generalizes, again, by simply replacing $\tilde{\mathcal{I}}$ with $\tilde{\mathcal{P}}$.

We now show how the results of Section 5 are generalized. For that, we need to explain how a p-problem is encoded. Recall that a source p-instance is encoded as a DNF instance I_p^α. We use a similar encoding for a probabilistic mapping. That is, every dependency σ is assigned a DNF formula over EVar (namely, a condition) and each variable is given a probability in $[0, 1]$. Formally, a *DNF schema mapping* $(\mathbf{S}, \mathbf{T}, \Sigma^\gamma_r)$ comprises source and target schemas $\mathbf{S}$ and $\mathbf{T}$ (without common relation symbols), a set $\Sigma \in \mathsf{Dep}^*_{\mathbf{ST}}$ of dependencies, a function γ that assigns to each $\sigma \in \Sigma$ a DNF formula $\gamma(\sigma)$ over EVar, and a function $r : \mathsf{EVar}(\gamma) \to [0, 1]$ (where, as usual, $\mathsf{EVar}(\gamma)$ is the set of all the event variables that appear in the image of γ). Now, we allow the source DNF instance I_p^α and the DNF schema mapping $(\mathbf{S}, \mathbf{T}, \Sigma^\gamma_r)$ to share events, that is,

$\mathsf{EVar}(\alpha)$ and $\mathsf{EVar}(\gamma)$ are not necessarily disjoint. In this case, we require p and r to agree on the common variables (i.e., $p(e) = r(e)$ for all $e \in \mathsf{EVar}(\alpha) \cap \mathsf{EVar}(\gamma)$).

A DNF schema mapping $(\mathbf{S}, \mathbf{T}, \Sigma_r^\gamma)$ and a source DNF instance I_p^α naturally encode a finite p-problem from $\mathbf{S}$ to $\mathbf{T}$, where a sample (Σ', I') is obtained as follows. First, a random truth assignment $\tau : \mathsf{EVar}(\gamma) \cup \mathsf{EVar}(\alpha) \to \{\mathbf{true}, \mathbf{false}\}$ is chosen for all the event variables e of $(\mathbf{S}, \mathbf{T}, \Sigma_r^\gamma)$ and I_p^α, by independently picking a random Boolean value $\tau(e)$, such that the probability for $\mathbf{true}$ is $r(e)$ or $p(e)$ (depending on whether $e \in \mathsf{EVar}(\gamma)$ or $e \in \mathsf{EVar}(\alpha)$). Second, all the members of Σ and I having their condition satisfied by τ are selected as members of Σ' and I', respectively. We denote this probability space by $\text{p-space}(\Sigma_r^\gamma, I_p^\alpha)$. Observe that, since γ and α are allowed to use the same variables, the marginal source p-instance and probabilistic schema mapping are not necessarily probabilistically independent. Moreover, it is easy to show (e.g., by using the encoding of finite probabilistic databases given in [1, 24]) that every finite p-problem can be represented as a combination of a DNF schema mapping $(\mathbf{S}, \mathbf{T}, \Sigma_r^\gamma)$ and a source DNF instance I_p^α.

When analyzing the complexity of the problems considered in Section 5, we make the assumption that the DNF schema mapping $(\mathbf{S}, \mathbf{T}, \Sigma_r^\gamma)$ is fixed (i.e., as in Section 5, data complexity is actually analyzed) and, moreover, that every sample $(\mathbf{S}, \mathbf{T}, \Sigma')$ (obtained by choosing a random truth assignment for $\mathsf{EVar}(\gamma)$) is a union of two finite sets, such that the first contains only st-tgds and t-egds, and the second is a weakly acyclic set of t-tgds. Thus, as in Section 5, the input for a computational problem consists of a source DNF (or tuple-independent) instance I_p^α. Then, all the results of Table 1 remain correct. For example, if Σ contains only st-tgds and t-tgds (and the above assumption about the samples $(\mathbf{S}, \mathbf{T}, \Sigma')$ of $(\mathbf{S}, \mathbf{T}, \Sigma_r^\gamma)$ holds), then testing whether a p-solution for the p-problem $\text{p-space}(\Sigma_r^\gamma, I_p^\alpha)$ exists, given a source DNF instance I_p^α, is coNP-complete, but it is solvable in polynomial time if I_p^α is a tuple-independent instance.

6.2 Probabilistic Mappings in the Literature

Data integration under uncertainty is studied in [13, 14, 41, 42], where the schema mapping (which is called there a *p-mapping*) is probabilistic and the source data are deterministic. We can compare our basic notions to those of [13, 14, 41, 42] by restricting our p-problem to a finite one where the source instance is deterministic.

Given a source instance I and a p-mapping $\tilde{\mathcal{M}}$, a *by-table solution*, as defined in [13, 42], is a special case of what we call a p-solution, namely, a finite p-solution $\tilde{\mathcal{J}}$ such that there is a *left-trivial* $\mathsf{dSOL}_{\mathbf{ST}}$-match of the corresponding p-problem $\tilde{\mathcal{P}}$ (namely, the product space $\tilde{\mathcal{M}} \times I$) in $\tilde{\mathcal{J}}$. (See Section 3.1 for the definition of a left-trivial probabilistic match.) Thus, the notion of a by-table solution is more restrictive than our notion of a p-solution (even if we restrict to deterministic source instances); namely, a by-table solution is a p-solution but not necessarily vice versa. In particular, the characterization of Theorem 3.6 for a p-solution does not hold for a by-table solution. As an example, for a by-table solution $\tilde{\mathcal{J}}$, the set $\Omega_+(\tilde{\mathcal{J}})$ must be at most as large as $\Omega_+(\tilde{\mathcal{M}})$ whereas no restriction of this nature exists for our p-solution.

A different type of solution studied (explicitly or implicitly) in [13, 14, 41] is the *by-tuple solution*.[13] A by-tuple solution differs from a by-table solution in that a by-tuple solution allows different tuples to be transformed by different possible worlds of the p-mapping; that is, for each tuple there is a probabilistic choice of the mapping to apply thereon. This notion is restrictive, since it is makes sense only when the mappings are transformations of individual facts. In particular, the mappings of [13, 14, 41] for the by-tuple semantics are essentially *inclusion dependencies* [7].

[13] In [42], only the by-table type is considered.

7. CONCLUSIONS

In this paper, we developed a broad and flexible framework for data exchange over probabilistic data. For that, we had to consider the fundamental notions of traditional data exchange, such as solution, universal solution, and target query evaluation, and generalize them appropriately. In particular, to accommodate source and target p-instances we defined the notion of a *p-solution* in terms of a probabilistic match (namely, the $\mathsf{SOL}_{\mathcal{M}}$-match). We explored the coherence of our basic definitions by scrutinizing them and providing several different characterizations for each of them. We explored the application of the framework to a concrete setting where p-instances are compactly encoded by annotations. Finally, we generalized the framework to allow for probabilistic schema mappings, by introducing the *p-problem* as a construct that represents a joint probability distribution over the data and mappings.

The notion of a probabilistic match allows us to systematically extend other concepts of data exchange into the probabilistic setting. An example is the *core solution* [16, 20]; in fact, it turns out that this extension of the core has various desired properties, which will be presented in a full version of the paper.

Acknowledgments

We thank Laura Haas, Elad Hazan, Renée J. Miller and C. Seshadhri for fruitful discussions. We especially thank Peter Haas for providing valuable comments on this work.

The research of Phokion Kolaitis is partially supported by NSF grants IIS-0430994 and ARRA 0905276.

8. REFERENCES

[1] S. Abiteboul and P. Senellart. Querying and updating probabilistic information in XML. In *EDBT*, pages 1059–1068. ACM, 2006.

[2] P. Agrawal, O. Benjelloun, A. D. Sarma, C. Hayworth, S. U. Nabar, T. Sugihara, and J. Widom. Trio: A system for data, uncertainty, and lineage. In *VLDB*, pages 1151–1154. ACM, 2006.

[3] R. Aharoni, E. Berger, A. Georgakopoulos, A. Perlstein, and P. Sprüssel. The max-flow min-cut theorem for countable networks. To appear in J. Combin. Theory (Series B).

[4] D. Barbará, H. Garcia-Molina, and D. Porter. The management of probabilistic data. *IEEE Trans. Knowl. Data Eng.*, 4(5):487–502, 1992.

[5] C. Beeri and M. Y. Vardi. A proof procedure for data dependencies. *J. ACM*, 31(4):718–741, 1984.

[6] J. Boulos, N. N. Dalvi, B. Mandhani, S. Mathur, C. Ré, and D.Suciu. MYSTIQ: a system for finding more answers by using probabilities. In *SIGMOD*, pages 891–893. ACM, 2005.

[7] M. A. Casanova, R. Fagin, and C. H. Papadimitriou. Inclusion dependencies and their interaction with functional dependencies. *J. Comput. Syst. Sci.*, 28(1):29–59, 1984.

[8] A. K. Chandra and P. M. Merlin. Optimal implementation of conjunctive queries in relational data bases. In *STOC*, pages 77–90. ACM, 1977.

[9] S. Cohen, B. Kimelfeld, and Y. Sagiv. Incorporating constraints in probabilistic XML. In *PODS*, pages 109–118. ACM, 2008.

[10] N. N. Dalvi and D. Suciu. Efficient query evaluation on probabilistic databases. In *VLDB*, pages 864–875. Morgan Kaufmann, 2004.

[11] N. N. Dalvi and D. Suciu. The dichotomy of conjunctive queries on probabilistic structures. In *PODS*, pages 293–302. ACM, 2007.

[12] N. N. Dalvi and D. Suciu. Efficient query evaluation on probabilistic databases. *VLDB J.*, 16(4):523–544, 2007.

[13] X. Dong, A. Halevy, and C. Yu. Data integration with uncertainty. *The VLDB Journal*, 18(2):469–500, 2009.

[14] X. L. Dong, A. Y. Halevy, and C. Yu. Data integration with uncertainty. In *VLDB*, pages 687–698. ACM, 2007.

[15] R. Fagin, P. G. Kolaitis, R. J. Miller, and L. Popa. Data exchange: semantics and query answering. *Theor. Comput. Sci.*, 336(1):89–124, 2005.

[16] R. Fagin, P. G. Kolaitis, and L. Popa. Data exchange: getting to the core. *ACM Trans. Database Syst.*, 30(1):174–210, 2005.

[17] R. Fagin, P. G. Kolaitis, L. Popa, and W.-C. Tan. Composing schema mappings: Second-order dependencies to the rescue. *ACM Trans. Database Syst.*, 30(4):994–1055, 2005.

[18] M. Fréchet. Sur les tableaux de corrélation dont les marges sont donnés. *Annales de l'Université de Lyon*, 4, 1951.

[19] N. Fuhr and T. Rölleke. A probabilistic relational algebra for the integration of information retrieval and database systems. *ACM Trans. Inf. Syst.*, 15(1):32–66, 1997.

[20] G. Gottlob and A. Nash. Efficient core computation in data exchange. *J. ACM*, 55(2), 2008.

[21] E. Grädel, Y. Gurevich, and C. Hirsch. The complexity of query reliability. In *PODS*, pages 227–234. ACM, 1998.

[22] T. J. Green, G. Karvounarakis, Z. G. Ives, and V. Tannen. Update exchange with mappings and provenance. In *VLDB*, pages 675–686. ACM, 2007.

[23] T. J. Green, G. Karvounarakis, and V. Tannen. Provenance semirings. In *PODS*, pages 31–40. ACM, 2007.

[24] T. J. Green and V. Tannen. Models for incomplete and probabilistic information. *IEEE Data Eng. Bull.*, 29(1):17–24, 2006.

[25] P. Hall. On representatives of subsets. *Journal of London Mathematical Society*, 10:26–30, 1935.

[26] P. Hell and J. Nešetřil. *Graphs and Homomorphisms*. Oxford lecture series in mathematics and its applications, 28. Oxford University Press, 2004.

[27] T. Imielinski and W. L. Jr. Incomplete information in relational databases. *J. ACM*, 31(4):761–791, 1984.

[28] M. Jerrum, L. G. Valiant, and V. V. Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theor. Comput. Sci.*, 43:169–188, 1986.

[29] R. M. Karp, M. Luby, and N. Madras. Monte-Carlo approximation algorithms for enumeration problems. *Journal of Algorithms*, 10(3):429–448, 1989.

[30] B. Kimelfeld, Y. Kosharovsky, and Y. Sagiv. Query efficiency in probabilistic XML models. In *SIGMOD Conference*, pages 701–714. ACM, 2008.

[31] B. Kimelfeld and Y. Sagiv. Maximally joining probabilistic data. In *PODS*, pages 303–312. ACM, 2007.

[32] K.-I. Ko. Some observations on the probabilistic algorithms and NP-hard problems. *Inf. Process. Lett.*, 14(1):39–43, 1982.

[33] C. Koch. Approximating predicates and expressive queries on probabilistic databases. In *PODS*, pages 99–108. ACM, 2008.

[34] M. Luby and B. Velickovic. On deterministic approximation of DNF. *Algorithmica*, 16(4/5):415–433, 1996.

[35] D. Maier, A. O. Mendelzon, and Y. Sagiv. Testing implications of data dependencies. *ACM Trans. Database Syst.*, 4(4):455–469, 1979.

[36] D. Morgenstern. Einfache beispiele zweidimensionaler verteilungen. *Mitteilingsblatt für Mathematische Statistik*, 8:234–235, 1956.

[37] C. H. Papadimitriou and M. Yannakakis. The complexity of facets (and some facets of complexity). *J. Comput. Syst. Sci.*, 28(2):244–259, 1984.

[38] C. Re, N. N. Dalvi, and D. Suciu. Efficient top-k query evaluation on probabilistic data. In *ICDE*, pages 886–895. IEEE, 2007.

[39] C. Re and D. Suciu. Efficient evaluation of HAVING queries on a probabilistic database. In *DBPL*, volume 4797 of *Lecture Notes in Computer Science*, pages 186–200. Springer, 2007.

[40] C. Re and D. Suciu. Materialized views in probabilistic databases for information exchange and query optimization. In *VLDB*, pages 51–62. ACM, 2007.

[41] A. D. Sarma, L. Dong, and A. Halevy. Uncertainty in data integration. *Managing and Mining Uncertain Data*, 2009.

[42] A. D. Sarma, X. Dong, and A. Y. Halevy. Bootstrapping pay-as-you-go data integration systems. In *SIGMOD*, pages 861–874. ACM, 2008.

[43] A. D. Sarma, M. Theobald, and J. Widom. Exploiting lineage for confidence computation in uncertain and probabilistic databases. In *ICDE*, pages 1023–1032. IEEE, 2008.

[44] P. Senellart and S. Abiteboul. On the complexity of managing probabilistic XML data. In *PODS*, pages 283–292. ACM, 2007.

[45] M. Shaked and J. Shanthikumar. *Stochastic Orders and Their Applications*. Academic Press, San Diego, CA, 1994.

[46] S. Toda and M. Ogiwara. Counting classes are at least as hard as the polynomial-time hierarchy. *SIAM J. Comput.*, 21(2):316–328, 1992.

[47] L. G. Valiant. The complexity of computing the permanent. *Theor. Comput. Sci.*, 8:189–201, 1979.

[48] S. Zachos. Probabilistic quantifiers and games. *Journal of Computer and System Sciences*, 36(3):433–451, 1988.

[49] D. Zuckerman. On unapproximable versions of NP-complete problems. *SIAM J. Comput.*, 25(6):1293–1304, 1996.