Data Modeling in Dataspace Support Platforms

Anish Das Sarma[1], Xin (Luna) Dong[2], and Alon Y. Halevy[3]

[1] Stanford University,
anish@cs.stanford.edu
[2] AT&T Labs-Research, New Jersey, USA
lunadong@research.att.com
[3] Google Inc.
halevy@google.com

Abstract. Data integration has been an important area of research for several years. However, such systems suffer from one of the main drawbacks of database systems: the need to invest significant modeling effort upfront. Dataspace Support Platforms (DSSP) envision a system that offers useful services on its data without any setup effort, and improve with time in a pay-as-you-go fashion. We argue that in order to support DSSPs, the system needs to model uncertainty at its core. We describe the concepts of probabilistic mediated schemas and probabilistic mappings as enabling concepts for DSSPs.

1 Introduction

Data integration and exchange systems offer a uniform interface to a multitude of data sources, and the ability to share data across multiple systems. These systems have recently enjoyed significant research and commercial success [10,11]. Current data integration systems are essentially a natural extension of traditional database systems in that queries are specified in a structured form and data is modeled in one of the traditional data models (relational, XML). In addition, the data integration system has exact knowledge of how the data in the sources map to the schema used by the data integration system. Consequently, to set up a data integration application there is a need for significant upfront effort in creating the mediated schema and the schema mappings.

Dataspace Support Platforms (DSSP) [12] envision data integration systems where the amount of upfront effort is much smaller. The system should be able to bootstrap itself and provide some useful services with no human intervention. Over time, through user feedback or as sources are added and the data management needs become clearer, the system evolves in a *pay-as-you-go* fashion.

To support DSSPs, we cannot rely on the same data modeling paradigms that form the basis for data integration systems. In particular, we cannot assume that the mediated schema is given to us in advance and that the schema mappings between the sources and the mediated schema will be *accurate*. We argue that a DSSP needs to support uncertainty at its very core: both in the mediated schema and in the schema mappings.

This article describes some of the formal foundations for data integration with uncertainty. We define probabilistic schema mappings and probabilistic mediated schemas, and show how to answer queries in their presence. With these foundations, it is possible to completely automatically bootstrap a pay-as-you-go integration system.

A.T. Borgida et al. (Eds.): Mylopoulos Festschrift, LNCS 5600, pp. 122–138, 2009.

This article is largely based on previous papers [3,5]. The proofs of the theorems we state and the experimental results validating some of our claims can be found therein.

2 Uncertainty in Data Integration

We begin by describing some of the possible sources of uncertainty in a dataspace-like data integration system, and then explain how such a system should differ from a traditional data integration system.

2.1 Sources of Uncertainty

Uncertainty can arise in several ways in a dataspace system.

Uncertain mediated schema: The mediated schema is the set of schema terms in which queries are posed. They do not necessarily cover all the attributes appearing in any of the sources, but rather the aspects of the domain that the application builder wishes to expose to the users. Uncertainty in the mediated schema can arise for several reasons. First, as we describe in Section 4, if the mediated schema is automatically inferred from the data sources during bootstrapping, there will be some uncertainty about the results. Second, when domains are broad, there will be some uncertainty about how to model them. For example, if we model all the topics in Computer Science there will be some uncertainty about the degree of overlap between different topics.

Uncertain schema mappings: Schema mappings specify the semantic relationships between the terms in the sources and the terms used in the mediated schema. However, schema mappings can be inaccurate. In a dataspace, we expect that many of the initial schema mappings will be automatically derived, and hence will be inaccurate. In many applications it is impossible to create and maintain precise mappings between data sources. This can be because the users are not skilled enough to provide precise mappings, such as in personal information management [6], because people do not understand the domain well and thus do not even know what correct mappings are, such as in bioinformatics, or because the scale of the data prevents generating and maintaining precise mappings, such as in integrating data on the scale of the web [16].

Uncertain data: Data sources in a dataspace may not always be well structured, and therefore some of the data may be obtained by automatic methods. Furthermore, systems that include many sources (e.g., the Web) may contain unreliable or inconsistent data. Even in enterprise settings, it is common for informational data such as gender, race, and income level to be dirty or missing, even when the transactional data is precise.

Uncertain queries: We expect that much of the early interaction with a dataspace will be through keyword queries, since the users are not aware of a (non-existent) schema. The system needs to translate these queries into some structured form so they can be reformulated with respect to the data sources. At this step, the system may generate multiple candidate structured queries and have some uncertainty about which query captures the real intent of the user.

2.2 System Architecture

We now describe the architecture of a data integration module for DSSPs and contrast it to a traditional data integration system.

The first and most fundamental characteristic of this system is that it is based on a probabilistic data model. This means that we attach probabilities to tuples that we process in the system, schema mappings, the mediated schemas, and possible interpretations of keyword queries posed to the system. In contrast, a traditional data integration system includes a single mediated schema and assumes a single (and correct) schema mapping between the mediated schema and each source. The data in the sources is also assumed to be correct.

Unlike a traditional data integration systems that assume that the query is posed in a structured fashion (i.e., can be translated to some subset of SQL), here we assume that queries are posed as keywords. Hence, whereas traditional data integration systems begin by reformulating a query onto the schemas of the data sources, a DSSP needs to first reformulate a keyword query into a set of candidate structured queries. We refer to this step as *keyword reformulation*. Note that keyword reformulation is different from techniques for keyword search on structured data (e.g., [14,1]) in that (a) it does not assume access to all the data in the sources or that the sources support keyword search, and (b) it tries to distinguish different structural elements in the query in order to pose more precise queries to the sources (e.g., realizing that in the keyword query "Chicago weather", "weather" is an attribute label and "Chicago" is an instance name). That being said, keyword reformulation should benefit from techniques that support answering keyword search on structured data.

The query answering model in a DSSP is also different. Instead of necessarily finding *all* answers to a given query, our goal is typically to find the top-k answers, and rank these answers most effectively.

The architecture of the system is shown in Figure 1. The system contains a number of data sources and a mediated schema (we omit probabilistic mediated schemas from this figure). When the user poses a query Q, which can be either a structured query on the mediated schema or a keyword query, the system returns a set of answer tuples,

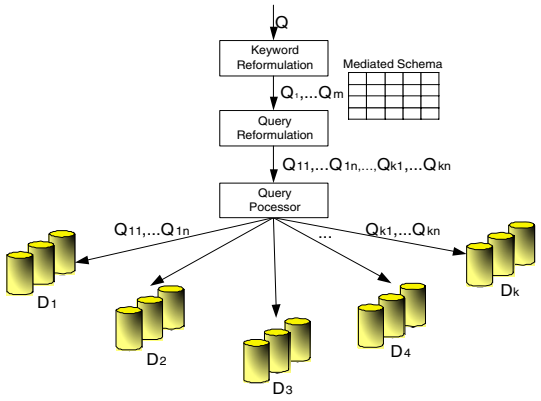

Fig. 1. Architecture of a data-integration system that handles uncertainty

each with a probability. If Q is a keyword query, the system first performs keyword reformulation to translate it into a set of candidate structured queries on the mediated schema. Otherwise, the candidate query is Q itself.

We discuss probabilistic schema mappings in Section 3, and probabilistic mediated schemas in Section 4.

2.3 Source of Probabilities

A critical issue in any system that manages uncertainty is whether we have a reliable source of probabilities. Whereas obtaining reliable probabilities for such a system is one of the most interesting areas for research, there is quite a bit to build on. For keyword reformulation, it is possible to train and test reformulators on large numbers of queries such that each reformulation result is given a probability based on its performance statistics. In the case of schema matching, it is standard practice for schema matchers to also associate numbers with the candidates they propose [4,20]. The issue here is that the numbers are meant only as a ranking mechanism rather than true probabilities. However, as schema matching techniques start looking a larger number of schemas, one can ascribe probabilities (or approximations thereof) to their measures (see Section 3.4). Finally, information extraction techniques are also often based on statistical machine learning methods, thereby lending their predictions a probabilistic interpretation.

3 Uncertainty in Mappings

The key to resolving heterogeneity at the schema level is to specify schema mappings between data sources. These mappings describe the relationship between the contents of the different sources and are used to reformulate a query posed over one source (or a mediated schema) into queries over the sources that are deemed relevant. However, in many applications we are not able to provide all the exact schema mappings upfront. In this section we introduce probabilistic schema mappings (p-mappings) to capture uncertainty on mappings between schemas.

We start by presenting a running example for this section that also motivates p-mappings (Section 3.1). Then we present a formal definition of probabilistic schema mapping and its semantics (Section 3.2). Section 3.3 describes algorithms for query answering with respect to probabilistic mappings and discusses the complexity.

3.1 Motivating Probabilistic Mappings

Example 1. Consider a data source S, which describes a person by her email address, current address, and permanent address, and the mediated schema T, which describes a person by her name, email, mailing address, home address and office address:

```
S=(pname, email-addr, current-addr, permanent-addr)
T=(name, email, mailing-addr, home-addr, office-addr)
```

A semi-automatic schema-mapping tool may generate three possible mappings between S and T, assigning each a probability. Whereas the three mappings all map pname to name, they map other attributes in the source and the target differently.

Possible Mapping	Prob
m_1 = {(pname, name), (email-addr, email), (current-addr, mailing-addr), (permanent-addr, home-addr)}	0.5
m_2 = {(pname, name), (email-addr, email), (permanent-addr, mailing-addr), (current-addr, home-addr)}	0.4
m_3 = {(pname, name), (email-addr, mailing-addr), (current-addr, home-addr)}	0.1

(a)

*p*name	*e*mail-addr	*c*urrent-addr	*p*ermanent-addr
Alice	alice@	Mountain View	Sunnyvale
Bob	bob@	Sunnyvale	Sunnyvale

(b)

Tuple (mailing-addr)	Prob
('Sunnyvale')	0.9
('Mountain View')	0.5
('alice@')	0.1
('bob@')	0.1

(c)

Fig. 2. The running example: (a) a probabilistic schema mapping between S and T; (b) a source instance D_S; (c) the answers of Q over D_S with respect to the probabilistic mapping

Figure 2(a) describes the three mappings using sets of attribute correspondences. For example, mapping m_1 maps pname to name, email-addr to email, current-addr to mailing-addr, and permanent-addr to home-addr. Because of the uncertainty about which mapping is correct, we consider all of these mappings in query answering.

Suppose the system receives a query Q composed on the mediated schema and asking for people's mailing addresses:

```
Q: SELECT mailing-addr FROM T
```

Using the possible mappings, we can reformulate Q into different queries:

```
Q1: SELECT current-addr FROM S
Q2: SELECT permanent-addr FROM S
Q3: SELECT email-addr FROM S
```

If the user requires all possible answers, the system generates a single aggregation query based on Q_1, Q_2 and Q_3 to compute the probability of each returned tuple, and sends the query to the data source. Suppose the data source contains a table D_S as shown in Figure 2(b), the system will retrieve four answer tuples, each with a probability, as shown in Figure 2(c).

If the user requires only the top-1 answer (i.e., the answer tuple with the highest probability), the system decides at runtime which reformulated queries to execute. For example, after executing Q_1 and Q_2 at the source, the system can already conclude that ('Sunnyvale') is the top-1 answer and can skip query Q_3. □

3.2 Definition and Semantics

3.2.1 Schema Mappings. We begin by reviewing non-probabilistic schema mappings. The goal of a schema mapping is to specify the semantic relationships between a

source schema and a *target schema*. We refer to the source schema as $\bar{S}$, and a relation in $\bar{S}$ as $S = \langle s_1, \ldots, s_m \rangle$. Similarly, we refer to the target schema as $\bar{T}$, and a relation in $\bar{T}$ as $T = \langle t_1, \ldots, t_n \rangle$.

We consider a limited form of schema mappings that are also referred to as *schema matching* in the literature. Specifically, a schema matching contains a set of *attribute correspondences*. An attribute correspondence is of the form $c_{ij} = (s_i, t_j)$, where s_i is a *source attribute* in the schema S and t_j is a *target attribute* in the schema T. Intuitively, c_{ij} specifies that there is a relationship between s_i and t_j. In practice, a correspondence also involves a function that transforms the value of s_i to the value of t_j. For example, the correspondence (c-degree, temperature) can be specified as temperature=c-degree $*1.8+32$, describing a transformation from Celsius to Fahrenheit. These functions are irrelevant to our discussion, and therefore we omit them. This class of mappings are quite common in practice and already expose many of the novel issues involved in probabilistic mappings. Broader classes of p-mappings are discussed in [5]. Formally, relation mappings and schema mappings are defined as follows.

Definition 1 (Schema Mapping). *Let $\bar{S}$ and $\bar{T}$ be relational schemas. A* relation mapping *M is a triple (S, T, m), where S is a relation in $\bar{S}$, T is a relation in $\bar{T}$, and m is a set of attribute correspondences between S and T.*

When each source and target attribute occurs in at most one correspondence in m, we call M a one-to-one relation mapping.

A schema mapping *$\overline{M}$ is a set of one-to-one relation mappings between relations in $\bar{S}$ and in $\bar{T}$, where every relation in either $\bar{S}$ or $\bar{T}$ appears at most once.* □

A pair of instances D_S and D_T *satisfies* a relation mapping m if for every source tuple $t_s \in D_S$, there exists a target tuple $t_t \in D_t$, such that for every attribute correspondence $(p, q) \in m$, the value of attribute p in t_s is the same as the value of attribute q in t_t.

Example 2. Consider the mappings in Example 1. The source database in Figure 2(b) (repeated in Figure 3(a)) and the target database in Figure 3(b) satisfy m_1. □

3.2.2 Probabilistic Schema Mappings

Intuitively, a probabilistic schema mapping describes a probability distribution over a set of *possible* schema mappings between a source schema and a target schema.

Definition 2 (Probabilistic Mapping). *Let $\bar{S}$ and $\bar{T}$ be relational schemas. A* probabilistic mapping (p-mapping)*, pM, is a triple $(S, T, \mathbf{m})$, where $S \in \bar{S}$, $T \in \bar{T}$, and $\mathbf{m}$ is a set $\{(m_1, Pr(m_1)), \ldots, (m_l, Pr(m_l))\}$, such that*

- *for $i \in [1, l]$, m_i is a one-to-one mapping between S and T, and for every $i, j \in [1, l]$, $i \neq j \Rightarrow m_i \neq m_j$.*
- *$Pr(m_i) \in [0, 1]$ and $\sum_{i=1}^{l} Pr(m_i) = 1$.*

A schema p-mapping*, $\overline{pM}$, is a set of p-mappings between relations in $\bar{S}$ and in $\bar{T}$, where every relation in either $\bar{S}$ or $\bar{T}$ appears in at most one p-mapping.* □

We refer to a non-probabilistic mapping as an *ordinary mapping*. A schema p-mapping may contain both p-mappings and ordinary mappings. Example 1 shows a p-mapping (see Figure 2(a)) that contains three possible mappings.

*p*name	*e*mail-addr	*c*urrent-addr	*p*ermanent-addr
Alice	alice@	Mountain View	Sunnyvale
Bob	bob@	Sunnyvale	Sunnyvale

(a)

*n*ame	*e*mail	*m*ailing-addr	*h*ome-addr	*o*ffice-addr
Alice	alice@	Mountain View	Sunnyvale	office
Bob	bob@	Sunnyvale	Sunnyvale	office

(b)

*n*ame	*e*mail	*m*ailing-addr	*h*ome-addr	*o*ffice-addr
Alice	alice@	Sunnyvale	Mountain View	office
Bob	email	bob@	Sunnyvale	office

(c)

Tuple (mailing-addr)	Prob
('Sunnyvale')	0.9
('Mountain View')	0.5
('alice@')	0.1
('bob@')	0.1

(d)

Tuple (mailing-addr)	Prob
('Sunnyvale')	0.94
('Mountain View')	0.5
('alice@')	0.1
('bob@')	0.1

(e)

Fig. 3. Example 3: (a) a source instance D_S; (b) a target instance that is by-table consistent with D_S and m_1; (c) a target instance that is by-tuple consistent with D_S and $< m_2, m_3 >$; (d) $Q^{table}(D_S)$; (e) $Q^{tuple}(D_S)$

3.2.3 Semantics of Probabilistic Mappings.

Intuitively, a probabilistic schema mapping models the uncertainty about which of the mappings in pM is the correct one. When a schema matching system produces a set of candidate matches, there are two ways to interpret the uncertainty: (1) a single mapping in pM is the correct one and it applies to all the data in S, or (2) several mappings are partially correct and each is suitable for a subset of tuples in S, though it is not known which mapping is the right one for a specific tuple. Figure 3(b) illustrates the first interpretation and applies mapping m_1. For the same example, the second interpretation is equally valid: some people may choose to use their current address as mailing address while others use their permanent address as mailing address; thus, for different tuples we may apply different mappings, so the correct mapping depends on the particular tuple.

We define query answering under both interpretations. The first interpretation is referred to as the *by-table* semantics and the second one is referred to as the *by-tuple* semantics of probabilistic mappings. Note that one cannot argue for one interpretation over the other; the needs of the application should dictate the appropriate semantics. Furthermore, the complexity results for query answering, which will show advantages for by-table semantics, should not be taken as an argument in the favor of by-table semantics.

We next define query answering with respect to p-mappings in detail and the definitions for schema p-mappings are the obvious extensions. Recall that given a query and an ordinary mapping, we can compute *certain answers* to the query with respect to the mapping. Query answering with respect to p-mappings is defined as a natural extension of certain answers, which we next review.

A mapping defines a relationship between instances of S and instances of T that are *consistent* with the mapping.

Definition 3 (Consistent Target Instance). *Let $M = (S, T, m)$ be a relation mapping and D_S be an instance of S.*

An instance D_T of T is said to be consistent with D_S and M*, if for each tuple $t_s \in D_S$, there exists a tuple $t_t \in D_T$, such that for every attribute correspondence $(a_s, a_t) \in m$, the value of a_s in t_s is the same as the value of a_t in t_t.* □

For a relation mapping M and a source instance D_S, there can be an infinite number of target instances that are consistent with D_S and M. We denote by $Tar_M(D_S)$ the set of all such target instances. The set of answers to a query Q is the intersection of the answers on all instances in $Tar_M(D_S)$.

Definition 4 (Certain Answer). *Let $M = (S, T, m)$ be a relation mapping. Let Q be a query over T and let D_S be an instance of S.*

A tuple t is said to be a certain answer of Q with respect to D_S and M*, if for every instance $D_T \in Tar_M(D_S)$, $t \in Q(D_T)$.* □

By-table semantics: We now generalize these notions to the probabilistic setting, beginning with the by-table semantics. Intuitively, a p-mapping pM describes a set of possible worlds, each with a possible mapping $m \in pM$. In by-table semantics, a source table can fall in one of the possible worlds; that is, the possible mapping associated with that possible world applies to the whole source table. Following this intuition, we define target instances that are *consistent with* the source instance.

Definition 5 (By-table Consistent Instance). *Let $pM = (S, T, \mathbf{m})$ be a p-mapping and D_S be an instance of S.*

An instance D_T of T is said to be by-table consistent with D_S and pM*, if there exists a mapping $m \in \mathbf{m}$ such that D_S and D_T satisfy m.* □

Given a source instance D_S and a possible mapping $m \in \mathbf{m}$, there can be an infinite number of target instances that are consistent with D_S and m. We denote by $Tar_m(D_S)$ the set of all such instances.

In the probabilistic context, we assign a probability to every answer. Intuitively, we consider the certain answers with respect to each possible mapping in isolation. The probability of an answer t is the sum of the probabilities of the mappings for which t is deemed to be a certain answer. We define by-table answers as follows:

Definition 6 (By-table Answer). *Let $pM = (S, T, \mathbf{m})$ be a p-mapping. Let Q be a query over T and let D_S be an instance of S.*

Let t be a tuple. Let $\bar{m}(t)$ be the subset of $\mathbf{m}$, such that for each $m \in \bar{m}(t)$ and for each $D_T \in Tar_m(D_S)$, $t \in Q(D_T)$.

Let $p = \sum_{m \in \bar{m}(t)} Pr(m)$. If $p > 0$, then we say (t, p) is a by-table answer of Q with respect to D_S and pM. □

By-tuple semantics: If we follow the possible-world notions, in by-tuple semantics, different tuples in a source table can fall in different possible worlds; that is, different possible mappings associated with those possible worlds can apply to the different source tuples.

Formally, the key difference in the definition of by-tuple semantics from that of by-table semantics is that a consistent target instance is defined by a mapping *sequence* that assigns a (possibly different) mapping in $\mathbf{m}$ to each source tuple in D_S. (Without losing generality, in order to compare between such sequences, we assign some order to the tuples in the instance).

Definition 7 (By-tuple Consistent Instance). *Let $pM = (S, T, \mathbf{m})$ be a p-mapping and let D_S be an instance of S with d tuples.*

An instance D_T of T is said to be by-tuple consistent with D_S and pM, *if there is a sequence $\langle m^1, \ldots, m^d \rangle$ such that d is the number of tuples in D_S and for every $1 \leq i \leq d$,*

- *$m^i \in \mathbf{m}$, and*
- *for the i^{th} tuple of D_S, t_i, there exists a target tuple $t'_i \in D_T$ such that for each attribute correspondence $(a_s, a_t) \in m^i$, the value of a_s in t_i is the same as the value of a_t in t'_i.* □

Given a mapping sequence $seq = \langle m^1, \ldots, m^d \rangle$, we denote by $Tar_{seq}(D_S)$ the set of all target instances that are consistent with D_S and seq. Note that if D_T is by-table consistent with D_S and m, then D_T is also by-tuple consistent with D_S and a mapping sequence in which each mapping is m.

We can think of every sequence of mappings $seq = \langle m^1, \ldots, m^d \rangle$ as a separate event whose probability is $Pr(seq) = \Pi_{i=1}^{d} Pr(m^i)$. If there are l mappings in pM, then there are l^d sequences of length d, and their probabilities add up to 1. We denote by $\mathbf{seq}_d(pM)$ the set of mapping sequences of length d generated from pM.

Definition 8 (By-tuple Answer). *Let $pM = (S, T, \mathbf{m})$ be a p-mapping. Let Q be a query over T and D_S be an instance of S with d tuples.*

Let t be a tuple. Let $\overline{seq}(t)$ be the subset of $\mathbf{seq}_d(pM)$, such that for each $seq \in \overline{seq}(t)$ and for each $D_T \in Tar_{seq}(D_S)$, $t \in Q(D_T)$.

Let $p = \sum_{seq \in \overline{seq}(t)} Pr(seq)$. If $p > 0$, we call (t, p) a by-tuple answer of Q with respect to D_S and pM. □

The set of by-table answers for Q with respect to D_S is denoted by $Q^{table}(D_S)$ and the set of by-tuple answers for Q with respect to D_S is denoted by $Q^{tuple}(D_S)$.

Example 3. Consider the p-mapping pM, the source instance D_S, and the query Q in the motivating example.

In by-table semantics, Figure 3(b) shows a target instance that is consistent with D_S (repeated in Figure 3(a)) and possible mapping m_1. Figure 3(d) shows the by-table answers of Q with respect to D_S and pM. As an example, for tuple $t =$('Sunnyvale'), we have $\bar{m}(t) = \{m_1, m_2\}$, so the possible tuple ('Sunnyvale', 0.9) is an answer.

In by-tuple semantics, Figure 3(c) shows a target instance that is by-tuple consistent with D_S and the mapping sequence $< m_2, m_3 >$. Figure 3(e) shows the by-tuple answers of Q with respect to D_S and pM. Note that the probability of tuple t=('Sunnyvale') in the by-table answers is different from that in the by-tuple answers. We describe how to compute the probabilities in detail in the next section. □

3.3 Query Answering

The following theorems are proved in [5] and consider by-table and by-tuple query answering in turn.

Theorem 1. *Let Q be an SPJ query and let $\overline{pM}$ be a schema p-mapping. Answering Q with respect to $\overline{pM}$ in by-table semantics is in PTIME in the size of the data and the mapping.* □

Theorem 2. *Let Q be an SPJ query and let $\overline{pM}$ be a schema p-mapping. The problem of finding the probability for a by-tuple answer to Q with respect to $\overline{pM}$ is #P-complete with respect to data complexity and is in PTIME with respect to mapping complexity.* □

Although by-tuple query-answering is hard, [5] shows several important restricted cases where it can still be done in polynomial time.

3.4 Creating P-Mappings

In [3] we address the problem of generating a p-mapping between a source schema and a target schema. We assume an input of a set of weighted correspondences between the source attributes and the target attributes. These weighted correspondences are created by a set of schema matching modules. There are two main challenges in creating p-mappings: (1) There may not be any p-mapping *consistent* with a given set of weight correspondences, and (2) when there is a consistent p-mapping, there may be multiple consistent p-mappings, and the question is which of them to choose. The algorithm we describe in [3] is based on normalizing weighted correspondences to ensure consistency, followed by choosing the p-mapping that maximizes the *entropy* of the probability assignment.

4 Uncertainty in Mediated Schema

The mediated schema is the set of schema terms (e.g., relations, attribute names) in which queries are posed. They do not necessarily cover all the attributes appearing in any of the sources, but rather the aspects of the domain that are important for the integration application. When domains are broad, and there are multiple perspectives on them (e.g., a domain in science that is constantly under evolution), then there will be uncertainty about which is the correct mediated schema and about the meaning of its terms. Also, when the mediated schema is created automatically by inspecting the sources in a pay-as-you-go system, there will be uncertainty about the mediated schema.

In this section we first motivate the need for probabilistic mediated schemas (p-med-schemas) with an example (Section 4.1). In Section 4.2 we formally define p-med-schemas and relate them with p-mappings in terms of expressive power and semantics of query answering.

4.1 P-Med-Schema Motivating Example

Let us begin with an example motivating p-med-schemas. Consider a setting in which we are trying to automatically infer a mediated schema from a set of data sources, where each of the sources is a single relational table. In this context, the mediated schema can be thought of as a "clustering" of source attributes, with similar attributes being grouped into the same cluster. The quality of query answers critically depends on the quality of this clustering. Because of the heterogeneity of the data sources being integrated, one is typically unsure of the semantics of the source attributes and in turn of the clustering.

Example 4. Consider two source schemas both describing people:

```
S1(name, hPhone, hAddr, oPhone, oAddr)
S2(name, phone, address)
```

In S2, the attribute phone can either be a home phone number or be an office phone number. Similarly, address can either be a home address or be an office address.

Suppose we cluster the attributes of S1 and S2. There are multiple ways to cluster the attributes and they correspond to different mediated schemas. Below we list a few (in the mediated schemas we abbreviate hPhone as hP, oPhone as oP, hAddr as hA, and oAddr as oA):

M1({name}, {phone, hP, oP}, {address, hA, oA})
M2({name}, {phone, hP}, {oP}, {address, oA}, {hA})
M3({name}, {phone, hP}, {oP}, {address, hA}, {oA})
M4({name}, {phone, oP}, {hP}, {address, oA}, {hA})
M5({name}, {phone}, {hP}, {oP}, {address}, {hA}, {oA})

None of the listed mediated schemas is perfect. Schema M_1 groups multiple attributes from S1. M_2 seems inconsistent because phone is grouped with hPhone while address is grouped with oAddress. Schemas M_3, M_4 and M_5 are partially correct but none of them captures the fact that phone and address can be either home phone and home address, or office phone and office address.

Even if we introduce probabilistic schema mappings, none of the listed mediated schemas will return ideal answers. For example, using M_1 prohibits returning correct answers for queries that contain both hPhone and oPhone because they are taken to be the same attribute. As another example, consider a query that contains phone and address. Using M_3 or M_4 as the mediated schema will unnecessarily favor home address and phone over office address and phone or vice versa. A system with M_2 will incorrectly favor answers that return a person's home address together with office phone number. A system with M_5 will also return a person's home address together with office phone, and does not distinguish such answers from answers with correct correlations.

A probabilistic mediated schema will avoid this problem. Consider a probabilistic mediated schema $\mathbf{M}$ that includes M_3 and M_4, each with probability 0.5. For each of them and each source schema, we generate a probabilistic mapping (Section 3). For example, the set of probabilistic mappings $\mathbf{pM}$ for S_1 is shown in Figure 4(a) and (b).

Now consider an instance of S_1 with a tuple

```
('Alice', '123-4567', '123, A Ave.',
          '765-4321', '456, B Ave.')
```

Possible Mapping	Probability
{(name, name), (hP, hPP), (oP, oP), (hA, hAA), (oA, oA)}	0.64
{(name, name), (hP, hPP), (oP, oP), (oA, hAA), (hA, oA)}	0.16
{(name, name), (oP, hPP), (hP, oP), (hA, hAA), (oA, oA)}	0.16
{(name, name), (oP, hPP), (hP, oP), (oA, hAA), (hA, oA)}	0.04

(a)

Possible Mapping	Probability
{(name, name), (oP, oPP), (hP, hP), (oA, oAA), (hA, hA)}	0.64
{(name, name), (oP, oPP), (hP, hP), (hA, oAA), (oA, hA)}	0.16
{(name, name), (hP, oPP), (oP, hP), (oA, oAA), (hA, hA)}	0.16
{(name, name), (hP, oPP), (oP, hP), (hA, oAA), (oA, hA)}	0.04

(b)

Answer	Probability
('Alice', '123-4567', '123, A Ave.')	0.34
('Alice', '765-4321', '456, B Ave.')	0.34
('Alice', '765-4321', '123, A Ave.')	0.16
('Alice', '123-4567', '456, B Ave.')	0.16

(c)

Fig. 4. The motivating example: (a) p-mapping for S_1 and M_3, (b) p-mapping for S_1 and M_4, and (c) by-table query answers w.r.t. $\mathbf{M}$ and $\mathbf{pM}$. Here we denote {phone, hP} by hPP, {phone, oP} by oPP, {address, hA} by hAA, and {address, oA} by oAA.

and a query

```
SELECT name, phone, address
FROM People
```

The by-table answer generated by our system with respect to $\mathbf{M}$ and $\mathbf{pM}$ is shown in Figure 4(c). (As we describe in detail in the following sections, we allow users to compose queries using any attribute in the source.) Compared with using one of M_2 to M_5 as a mediated schema, our method generates better query results in that (1) it treats answers with home address and home phone and answers with office address and office phone equally, and (2) it favors answers with the correct correlation between address and phone number. □

4.2 Probabilistic Mediated Schema

Consider a set of source schemas $\{S_1, \ldots, S_n\}$. We denote the attributes in schema $S_i, i \in [1, n]$, by $attr(S_i)$, and the set of all source attributes as $\mathcal{A}$. That is, $\mathcal{A} =$

$attr(S_1) \cup \ldots \cup attr(S_n)$. We denote a mediated schema for the set of sources $\{S_1, \ldots, S_n\}$ by $M = \{A_1, \ldots, A_m\}$, where each of the A_i's is called a *mediated attribute*. The mediated attributes are *sets* of attributes from the sources, i.e., $A_i \subseteq \mathcal{A}$; for each $i, j \in [1, m], i \neq j \Rightarrow A_i \cap A_j = \emptyset$.

Note that whereas in a traditional mediated schema an attribute has a name, we do not deal with naming of an attribute in our mediated schema and allow users to use any source attribute in their queries. (In practice, we can use the most frequent source attribute to represent a mediated attribute when exposing the mediated schema to users.) If a query contains an attribute $a \in A_i, i \in [1, m]$, then when answering the query we replace a everywhere with A_i.

A *probabilistic mediated schema* consists of a set of mediated schemas, each with a probability indicating the likelihood that the schema correctly describes the domain of the sources. We formally define probabilistic mediated schemas as follows.

Definition 9 (Probabilistic Mediated Schema). *Let $\{S_1, \ldots, S_n\}$ be a set of schemas. A* probabilistic mediated schema (p-med-schema) for $\{S_1, \ldots, S_n\}$ *is a set*

$$\mathbf{M} = \{(M_1, Pr(M_1)), \ldots, (M_l, Pr(M_l))\}$$

where

- *for each $i \in [1, l]$, M_i is a mediated schema for $S_1, \ldots, S_n$, and for each $i, j \in [1, l], i \neq j$, M_i and M_j correspond to different clusterings of the source attributes;*
- *$Pr(M_i) \in (0, 1]$, and $\Sigma_{i=1}^{l} Pr(M_i) = 1$.* □

Semantics of queries: Next we define the semantics of query answering with respect to a p-med-schema and a set of p-mappings for each mediated schema in the p-med-schema. Recall that answering queries with respect to p-mappings returns a set of answer tuples, each with a probability indicating the likelihood that the tuple occurs as an answer. We consider by-table semantics here. Given a query Q, we compute answers by first answering Q with respect to each possible mapping, and then for each answer tuple t summing up the probabilities of the mappings with respect to which t is generated.

We now extend this notion for query answering that takes p-med-schema into consideration. Intuitively, we compute query answers by first answering the query with respect to each possible mediated schema, and then for each answer tuple taking the sum of its probabilities weighted by the probabilities of the mediated schemas.

Definition 10 (Query Answer). *Let S be a source schema and $\mathbf{M} = \{(M_1, Pr(M_1)), \ldots, (M_l, Pr(M_l))\}$ be a p-med-schema. Let $\mathbf{pM} = \{pM(M_1), \ldots, pM(M_l)\}$ be a set of p-mappings where $pM(M_i)$ is the p-mapping between S and M_i. Let D be an instance of S and Q be a query.*

Let t be a tuple. Let $Pr(t|M_i), i \in [1, l]$, be the probability of t in the answer of Q with respect to M_i and $pM(M_i)$. Let $p = \Sigma_{i=1}^{l} Pr(t|M_i) * Pr(M_i)$. If $p > 0$, then we say (t, p) is a by-table answer with respect to $\mathbf{M}$ and $\mathbf{pM}$.

We denote all by-table answers by $Q_{\mathbf{M},\mathbf{pM}}(D)$. □

We say that query answers A_1 and A_2 are *equal* (denoted $A_1 = A_2$) if A_1 and A_2 contain exactly the same set of tuples with the same probability assignments.

Expressive power: A natural question to ask at this point is whether probabilistic mediated schemas provide any added expressive power compared to deterministic ones. Theorem 3 shows that if we consider *one-to-many* schema mappings, where one source attribute can be mapped to multiple mediated attributes, then any combination of a p-med-schema and p-mappings can be equivalently represented using a deterministic mediated schema with p-mappings, but may not be represented using a p-med-schema with deterministic schema mappings. Note that we can easily extend the definition of query answers to one-to-many mappings as one mediated attribute can correspond to no more than one source attribute.

Theorem 3. *The following two claims hold.*

1. *Given a source schema S, a p-med-schema $\mathbf{M}$, and a set of p-mappings $\mathbf{pM}$ between S and possible mediated schemas in $\mathbf{M}$, there exists a deterministic mediated schema T and a p-mapping pM between S and T, such that $\forall D, Q$: $Q_{\mathbf{M},\mathbf{pM}}(D) = Q_{T,pM}(D)$.*
2. *There exists a source schema S, a mediated schema T, a p-mapping pM between S and T, and an instance D of S, such that for any p-med-schema $\mathbf{M}$ and any set $\mathbf{m}$ of deterministic mappings between S and possible mediated schemas in $\mathbf{M}$, there exists a query Q such that $Q_{\mathbf{M},\mathbf{m}}(D) \neq Q_{T,pM}(D)$.* □

In contrast, Theorem 4 shows that if we restrict our attention to one-to-one mappings, then a probabilistic mediated schema *does* add expressive power.

Theorem 4. *There exists a source schema S, a p-med-schema $\mathbf{M}$, a set of one-to-one p-mappings $\mathbf{pM}$ between S and possible mediated schemas in $\mathbf{M}$, and an instance D of S, such that for any deterministic mediated schema T and any one-to-one p-mapping pM between S and T, there exists a query Q such that, $Q_{\mathbf{M},\mathbf{pM}}(D) \neq Q_{T,pM}(D)$.* □

Constructing one-to-many p-mappings in practice is much harder than constructing one-to-one p-mappings. When we are restricted to one-to-one p-mappings, p-med-schemas grant us more expressive power while keeping the process of mapping generation feasible.

4.3 Creating P-Med-Schemas

In [3], we study the problem of creating p-med-schemas. First the algorithm constructs a graph with source attributes as nodes, and weighted edges representing pairwise similarities. Then, it groups similarity edges into *certain* and *uncertain* based on their weights. Clusterings due to inclusion of all certain edges and combinations of inclusion/exclusion of uncertain edges constitute possible mediated schemas. Probabilities on each mediated schema are assigned based on *consistency* with respect to the data sources.

To enable users to see a single mediated schema, reference [3] also describes how to *consolidate* (without loss) the p-med-schema with one-to-one p-mappings into an equivalent single mediated schema with one-to-many p-mappings.

5 Related Work

Probabilistic Mappings: There have been various models proposed to capture uncertainty on mappings between attributes. [8] proposes keeping the top-K mappings between two schemas, each with a probability (between 0 and 1) of being true. [9] proposes assigning a probability for matching of every pair of source and target attributes. This notion corresponds to weighted correspondences described in Section 3.4.

Magnani and Montesi [17] have empirically shown that top-k schema mappings can be used to increase the recall of a data integration process and Gal [7] described how to generate top-k schema matchings by combining the matching results generated by various matchers. The probabilistic schema mappings we described above are different as they contain all possible schema mappings that conform to the schema matching results and assigns probabilities to these mappings to reflect the likelihood that each mapping is correct. Nottelmann and Straccia [19] proposed generating probabilistic schema matchings that capture the uncertainty on each matching step. The probabilistic schema mappings we create not only capture our uncertainty on results of the matching step, but also take into consideration various combinations of attribute correspondences and describe a *distribution* of possible schema mappings where the probabilities of all mappings sum up to 1.

Mediated Schemas: He and Chang [13] considered the problem of generating a mediated schema for a set of web sources. Their approach was to create a mediated schema that is statistically maximally *consistent* with the source schemas. To do so, they assume that the source schemas are created by a *generative model* applied to some mediated schema, which can be thought of as a probabilistic mediated schema. The probabilistic mediated schema we described in this chapter has several advantages in capturing heterogeneity and uncertainty in the domain. We can express a wider class of attribute clusterings, and in particular clusterings that capture attribute correlations. Moreover, we are able to combine attribute matching and co-occurrence properties for the creation of the probabilistic mediated schema, allowing for instance two attributes from one source to have a nonzero probability of being grouped together in the mediated schema. Also, the approach for p-med-schema creation described in this chapter is independent of a specific schema-matching technique, whereas the approach in [13] is tuned for constructing generative models and hence must rely on statistical properties of source schemas.

Magnani et al. [18] proposed generating a set of alternative mediated schemas based on probabilistic relationships between *relations* (such as an Instructor relation intersects with a Teacher relation but is disjoint with a Student relation) obtained by sampling the overlapping of data instances. Here we focus on matching attributes within relations. In addition, our approach allows exploring various types of evidence to improve matching and we assign probabilities to the mediated schemas we generate.

Chiticariu et. al. [2] studied the generation of multiple mediated schemas for an existing set of data sources. They consider multi-table data sources, not considered in this chapter, but explore interactive techniques that aid humans in arriving at the mediated schemas.

6 Conclusions

The investigation of data integration with uncertainty is only beginning. This chapter described some of the fundamental concepts on which such DSSP data integration systems will be built, but there is a lot more to do.

The main challenge is to build actual data integration systems that incorporate uncertainty and thereby uncover a new set of challenges, such as efficiency and understanding what are the common types of uncertainty that arise in data integration applications, so techniques can be tailored for these cases.

Our work focussed only on bootstrapping a pay-as-you-go integration system. The next challenge is to find methods to improve it over time (see [15] for a first work on doing so).

References

1. Agrawal, S., Chaudhuri, S., Das, G.: DBXplorer: A system for keyword-based search over relational databases. In: Proc. of ICDE, pp. 5–16 (2002)
2. Chiticariu, L., Kolaitis, P.G., Popa, L.: Interactive generation of integrated schemas. In: Proc. of ACM SIGMOD (2008)
3. Das Sarma, A., Dong, L., Halevy, A.: Bootstrapping pay-as-you-go data integration systems. In: Proc. of ACM SIGMOD (2008)
4. Doan, A., Halevy, A.Y.: Semantic integration research in the database community: A brief survey. AI Magazine 26(1), 83–94 (2005)
5. Dong, X., Halevy, A.Y., Yu, C.: Data integration with uncertainty. In: Proc. of VLDB (2007)
6. Dong, X., Halevy, A.Y.: A platform for personal information management and integration. In: Proc. of CIDR (2005)
7. Gal, A.: Why is schema matching tough and what can we do about it? SIGMOD Record 35(4), 2–5 (2007)
8. Gal, A., Modica, G., Jamil, H., Eyal, A.: Automatic ontology matching using application semantics. AI Magazine 26(1), 21–31 (2005)
9. Gal, A., Anaby-Tavor, A., Trombetta, A., Montesi, D.: A framework for modeling and evaluating automatic semantic reconciliation (2003)
10. Halevy, A.Y., Ashish, N., Bitton, D., Carey, M.J., Draper, D., Pollock, J., Rosenthal, A., Sikka, V.: Enterprise information integration: successes, challenges and controversies. In: SIGMOD (2005)
11. Halevy, A.Y., Rajaraman, A., Ordille, J.J.: Data integration: The teenage years. In: VLDB (2006)
12. Halevy, A.Y., Franklin, M.J., Maier, D.: Principles of dataspace systems. In: PODS (2006)
13. He, B., Chang, K.C.: Statistical schema matching across web query interfaces. In: Proc. of ACM SIGMOD (2003)
14. Hristidis, V., Papakonstantinou, Y.: DISCOVER: Keyword search in relational databases. In: Proc. of VLDB, pp. 670–681 (2002)
15. Jeffery, S., Franklin, M., Halevy, A.: Pay-as-you-go user feedback for dataspace systems. In: Proc. of ACM SIGMOD (2008)
16. Madhavan, J., Cohen, S., Dong, X., Halevy, A., Jeffery, S., Ko, D., Yu, C.: Web-scale data integration: You can afford to pay as you go. In: Proc. of CIDR (2007)

17. Magnani, M., Montesi, D.: Uncertainty in data integration: current approaches and open problems. In: VLDB workshop on Management of Uncertain Data, pp. 18–32 (2007)
18. Magnani, M., Rizopoulos, N., Brien, P., Montesi, D.: Schema integration based on uncertain semantic mappings. In: LNCS, pp. 31–46 (2005)
19. Nottelmann, H., Straccia, U.: Information retrieval and machine learning for probabilistic schema matching. Information Processing and Management 43(3), 552–576 (2007)
20. Rahm, E., Bernstein, P.A.: A survey of approaches to automatic schema matching. VLDB Journal 10(4), 334–350 (2001)