Bootstrapping Pay-As-You-Go Data Integration Systems*

Anish Das Sarma†
Stanford University
California, USA
anish@cs.stanford.edu

Xin Dong†
AT&T Labs–Research
New Jersey, USA
lunadong@research.att.com

Alon Halevy
Google Inc.
California, USA
halevy@google.com

ABSTRACT

Data integration systems offer a uniform interface to a set of data sources. Despite recent progress, setting up and maintaining a data integration application still requires significant upfront effort of creating a mediated schema and semantic mappings from the data sources to the mediated schema. Many application contexts involving multiple data sources (e.g., the web, personal information management, enterprise intranets) do not require full integration in order to provide useful services, motivating a *pay-as-you-go* approach to integration. With that approach, a system starts with very few (or inaccurate) semantic mappings and these mappings are improved over time as deemed necessary.

This paper describes the first completely self-configuring data integration system. The goal of our work is to investigate how advanced of a starting point we can provide a pay-as-you-go system. Our system is based on the new concept of a *probabilistic mediated schema* that is automatically created from the data sources. We automatically create probabilistic schema mappings between the sources and the mediated schema. We describe experiments in multiple domains, including 50-800 data sources, and show that our system is able to produce high-quality answers with no human intervention.

Categories and Subject Descriptors

H [**Information Systems**]

General Terms

Algorithms, Design, Experimentation

Keywords

data integration, pay-as-you-go, mediated schema, schema mapping

*This work was supported by a Microsoft Graduate Fellowship and by NSF under grants IIS-0415175, IIS-0324431 and IIS-0414762.
†Work was partially done while these authors were at Google.

SIGMOD'08, June 9–12, 2008, Vancouver, BC, Canada.

1. INTRODUCTION

Data integration systems offer a single-point interface to a set of data sources. A data integration application is typically built by creating a mediated schema for the domain at hand, and creating semantic mappings between the schemas of the data sources and the mediated schema. The user (or application) poses queries using the terminology of the mediated schema, and the query is reformulated onto the sources using the semantic mappings.

Despite recent progress in the field, setting up and maintaining a data integration application still requires significant upfront and ongoing effort. Hence, reducing the effort required to set up a data integration application, often referred to as *on-the-fly* integration, has been a recurring challenge for the field. In fact, as pointed out in [12], many application contexts (e.g., the web, personal information management, enterprise intranets) do not require full integration in order to provide useful services. This observation led to proposing a *pay-as-you-go* approach to integration, where the system starts with very few (or inaccurate) semantic mappings and these mappings are improved over time as deemed necessary.

This paper describes the first completely self-configuring data integration system. The goal of our work is to investigate how advanced of a starting point we can provide a pay-as-you-go system, and how well a completely automated system can perform. We evaluate our system on several domains, each consisting of 50-800 heterogeneous tables obtained from the Web. The key contribution of the paper is that we can obtain very good query precision and recall compared to the alternatives of (1) treating all the sources as text, or (2) performing full manual integration.

To completely automate data integration, we need to automatically create a mediated schema from the sources and automatically create semantic mappings between the sources and the mediated schema. Automatic creation of schema mappings has received considerable attention [5, 7, 8, 9, 13, 14, 18, 21, 23, 26, 28]. Recently [10] introduced the notion of *probabilistic schema mappings* which provides a foundation for answering queries in a data integration system with uncertainty about semi-automatically created mappings. To complete the puzzle, we show how to automatically create a mediated schema from a set of data sources.

The specific contributions of the paper are the following. First, we show how to automatically create a mediated schema from a set of data sources. In doing so, we introduce the concept of a *probabilistic* mediated schema, which is a set of mediated schemas with probabilities attached to each. We show that probabilistic mediated schemas offer benefits in modeling uncertainty about the semantics of attributes in the sources. We describe how to create a deterministic mediated schema from the probabilistic one, which is the schema exposed to the user.

Our second contribution is an algorithm for creating probabilis-

tic schema mappings from the data sources to the mediated schema. Since a mapping is constructed from a set of weighted attribute correspondences between a source schema and the mediated schema, and such weighted correspondences do not uniquely determine a semantic mapping [10], we construct a probabilistic mapping that is consistent with the correspondences and obtains the maximal entropy.

As our final contribution, we describe a set of experimental results establishing the efficacy of our algorithms. We compare the precision and recall of our system with several alternative approaches including: (1) a perfect integration where the mediated schema and mappings are created manually, (2) a document-centric approach where we perform keyword search on the sources, and (3) posing queries directly on the sources without a mediated schema. We show that our automatic methods achieve F-measure of around 0.92 compared to (1) and significantly outperform (2) and (3) and several variants of our algorithms. Hence, we believe that our approach can substantially reduce the amount of time taken to create a data integration application.

The paper is organized as follows. Section 2 gives an overview of our approach. Section 3 formally introduces the notion of a probabilistic mediated schema and presents some basic results about them. Sections 4–6 present algorithms for the various steps in setting up a data integration system: constructing the probabilistic mediated schema, generating probabilistic mappings for each source, and consolidating the schema and mappings, respectively. Section 7 provides an experimental evaluation of our system. Section 8 presents related work and we conclude in Section 9.

2. OVERVIEW

We begin with an overview of our approach and point out the technical challenges we address in the paper. Creating a data integration application involves two activities that require significant human effort: creating the mediated schema and creating semantic mappings between the data sources and the mediated schema. Both activities require knowledge of the domain as well as an understanding of the queries that can be frequently asked.

Our goal is to create a data integration application *without* any human involvement. The resulting integration should give *best-effort* answers, and should let the administrator improve the system in a pay-as-you-go fashion. To accomplish this goal, we need to automatically create a mediated schema and semantic mappings from the sources to that schema.

To support best-effort answers and improvement over time, we build our system on a probabilistic data model. Recent work has introduced probabilistic schema mappings [10], which enable a data integration system to model its uncertainty on which schema mapping is correct. While we define probabilistic schema mappings formally in the next section, intuitively, a probabilistic schema mapping consists of a *set* of mappings with a probability attached to each mapping. Previous research has also considered the problem of automatically creating (non-probabilistic) schema mappings.

The mediated schema in a data integration application consists of the set of relations and attributes that we wish to expose to users of the system. The mediated schema need not include all the attributes that appear in the sources, nor does it include only the intersection of the attributes that appear in all of the sources.

To build a mediated schema automatically, a natural strategy is to start from attributes in the source schemas, group those that have the same meaning, and consider each result group as an attribute in the mediated schema. Because of the heterogeneity of the sources, we are typically unsure of the semantics of the source attributes and in turn of the clustering results. Furthermore, since attributes can have ambiguous meanings and some attributes can overlap in their meaning, this approach to creating a mediated schema results in a significant amount of uncertainty.

Our approach is based on constructing a *probabilistic mediated schema*. The following example illustrates the advantages of a probabilistic mediated schema in our setting.

EXAMPLE 2.1. *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 .5. For each of them and each source schema, we generate a probabilistic mapping (Section 5). For example, the set of probabilistic mappings* $\mathbf{pM}$ *for* S_1 *is shown in Figure 1(a) and (b).*

Now consider an instance of S_1 *with a tuple*

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

and a query

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

The answer generated by our system with respect to $\mathbf{M}$ *and* $\mathbf{pM}$ *is shown in Figure 1(c). (As we describe in detail in Section 3, we*

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)

Figure 1: The motivating example: (a) p-mapping for S_1 and M_3, (b) p-mapping for S_1 and M_4, and (c) query answers w.r.t. M and pM. Here we denote {phone, hP} by hPP, {phone, oP} by oPP, {address, hA} by hAA, and {address, oA} by oAA.

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. □

Building on the concept of a probabilistic mediated schema, our approach consists of three steps:

Construct a probabilistic mediated schema: We begin by constructing a set of schemas with a probability associated with each one (Section 4).

Find best probabilistic schema mappings: Given the probabilistic mediated schema, we need to construct the appropriate semantic mappings (Section 5). The key challenge in this step is that the tools for automatic schema mapping typically produce *weighted correspondences* between pairs of attributes (one from each schema). But such correspondences neither uniquely determine a specific schema mapping, nor uniquely determine a distribution of possible schema mappings. Therefore, we need to choose one distribution that seems to *best* capture the automatically generated attribute correspondences.

Create a single mediated schema to expose to the user: In this step we create a single mediated schema for the user and create semantic mappings to it by adjusting the mappings created in the previous step (Section 6). The consolidated mediated schema is such that it returns the same answers as we would have obtained over the probabilistic mediated schema.

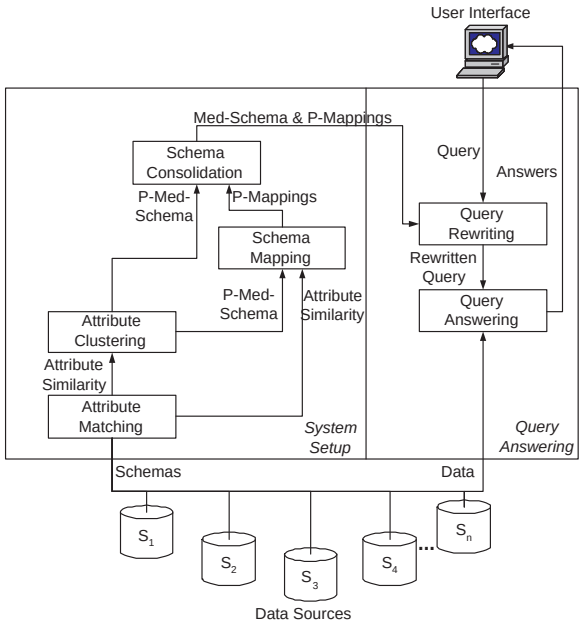

Figure 2: Architecture of our automatic-setup data integration system.

This step is not strictly necessary. For example, in some situations we may prefer to present the user with the set of mediated schemas and have her choose one that best suits the application's needs. We also show that under certain conditions, a probabilistic mediated schema actually adds expressive power to the system.

Figure 2 depicts the architecture of our system. At set-up time, we start with attribute matching, based on which we generate the probabilistic mediated schema and mappings. We then consolidate them and generate the final mediated schema and mappings. At query-answering time, for each data source we rewrite a query according to the mappings and answer the rewritten queries on the source data. We then combine the results from different data sources by taking the disjunction of the probabilities of each answer tuple; that is, if answer t has probability $p_i, i \in [1, n]$, for the i-th data source, the final probability of t is $1 - \Pi_{i=1}^{n}(1 - p_i)$. Here we assume that the different data sources are independent. Dealing with data sources where some may be derived from others is beyond the scope of this paper.

3. PROBABILISTIC MEDIATED SCHEMAS

In this section we formally define probabilistic mediated schemas and the semantics of queries posed over them. We also show precisely the relationship between probabilistic mediated schemas and deterministic (i.e., non-probabilistic) mediated schemas.

In our discussion, we consider a set of source schemas $\{S_1, \ldots, S_n\}$ that are assumed to be roughly from the same domain. We consider the case where each schema contains a *single* table with a set of attributes. 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, $\mathcal{A} = attr(S_1) \cup \cdots \cup attr(S_n)$. We focus on this case because it already exposes many interesting problems and is a common case in practice (e.g., integration on the web); we describe the challenges in integrating multiple-table sources in future work (Section 9).

We begin with deterministic mediated schemas. We denote a mediated schema for a 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 3.1 (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 $\Sigma_{i=1}^{l} Pr(M_i) = 1$.* □

Probabilistic schema mappings: Before we can define the semantics of answers posed over mediated schemas, we review the definition of probabilistic schema mappings, originally introduced in [10]. In this paper we mostly consider *one-to-one schema mappings*. Given a mediated schema M and a data source S, a schema mapping consists of a set of attribute correspondences, where each correspondence matches a source attribute in S to an attribute in the mediated schema M. The mapping is one-to-one if each of the attributes of the source or the mediated schema is involved in at most one attribute correspondence.

A probabilistic schema mapping describes a probabilistic distribution of possible mappings between a source and a mediated schema. Formally, they are defined as follows:

DEFINITION 3.2 (PROBABILISTIC MAPPING). *Let S be a source schema and M be a mediated schema. A* probabilistic schema mapping (p-mapping) *between S and M is a set*

$$pM = \{(m_1, Pr(m_1)), \ldots, (m_l, Pr(m_l))\}$$

such that

- *for each $i \in [1, l]$, m_i is a schema mapping between S and M, and for every $i, j \in [1, l], i \neq j \Rightarrow m_i \neq m_j$;*
- *$Pr(m_i) \in (0, 1]$, and $\Sigma_{i=1}^{l} Pr(m_i) = 1$.* □

We focus on one-to-one mappings because they are common in practice and it is more feasible to generate such mappings than more complex mappings. As we show later, our algorithm actually produces one-to-many schema mappings when it consolidates a probabilistic mediated schema into a deterministic one. A one-to-many mapping maps a source attribute to a set (e.g., concatenation) of attributes in the mediated schema.

Semantics of queries: We measure the quality of the p-med-schema and the p-mappings we generate by the accuracy of query answering results. Our goal is to return all correct answers possibly with wrong answers, but rank correct answers higher. That is, we want to obtain high *precision*, *recall* and high *Top-k precision*.

However, before we can answer queries in this setting, we need to define the semantics of queries. We define the semantics of a p-med-schema by defining query answering with respect to a p-med-schema and a set of p-mappings. Our definition is the natural extension of query answering with respect to p-mappings [10].

We consider select-project-join (SPJ) queries, a core set of SQL queries. 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. In this paper we consider *by-table* semantics, which assumes there is one single possible mapping that is correct and it applies to all tuples in the source table. 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 3.3 (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 3.4 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. (To maintain the flow of the paper, we provide only proof sketches for some theorems in the body of the paper, and defer complete proofs to the appendix.)

THEOREM 3.4 (SUBSUMPTION). *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 $Q_{\mathbf{M},\mathbf{m}}(D) \neq Q_{T,pM}(D)$.

□

Proof sketch: To prove (1), we show that we can create a single new mediated schema T, and rewrite each original schema mapping in $\mathbf{pM}$ between S and a mediated schema in $\mathbf{M}$ to a corresponding schema mapping between S and T. For the second part, we give an example S, T, and a p-mapping between them such that no p-med-schema with deterministic mappings can represent it. □

In contrast, Theorem 3.5 shows that if we restrict our attention to one-to-one mappings, then a probabilistic mediated schema *does* add expressive power.

THEOREM 3.5. *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, $Q_{\mathbf{M},\mathbf{pM}}(D) \neq Q_{T,pM}(D)$.* □

Proof sketch: We prove the theorem by constructing a p-med-schema $M = \{M_1, M_2\}$ and showing that for any single mediated schema T and any p-mapping pM, a query Q referring to an attribute that is clustered differently in M_1 and M_2 would miss answers from those generated with respect to one of M_1 and M_2 when posed over T. □

Constructing one-to-many p-mappings in practice is much harder than constructing one-to-one p-mappings. And, 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. MEDIATED SCHEMA GENERATION

This section describes how we create the probabilistic mediated schema. We begin by showing how to create a single mediated schema, and then we extend the algorithm to create multiple mediated schemas with probabilities attached to each.

4.1 Creating a single mediated schema

Consider a set of source table schemas $S_1, \ldots, S_n$. We are interested in creating a mediated schema M which best represents the domain the tables are about. Our strategy is to create M by clustering attributes in source tables. We want M to contain all "important" attributes from source tables, and we want to ensure that semantically similar attributes from different tables are combined into one cluster. For example, if two source tables have attributes phone-no and phone, we would like to put them in the same mediated attribute.

Our mediated-schema generation algorithm assumes there is some pairwise attribute similarity measure, s. The similarity $s(a_i, a_j)$ between two source attributes a_i and a_j depicts how closely the two attributes represent the same real-world concept. There has been a significant amount of work in designing pairwise similarity functions [26]. Improving on these techniques is not the focus of our work. Instead, our algorithm is designed so it can leverage any existing technique.

We create a mediated schema in three steps. First, we remove infrequent attributes from the set of all source attributes; that is, attribute names that do not appear in a large fraction of source tables. This step ensures that our mediated schema contains only information that is relevant and central to the domain. In the second step we construct a weighted graph whose nodes are the attributes that survived the filter of the first step. An edge in the graph is labeled with the pairwise similarity between the two nodes it connects. We include an edge in the graph only if its weight is above a certain threshold τ. Finally, we cluster the nodes in the resulting weighted graph to obtain the mediated schema. A cluster is defined to be a connected component of the graph.

0: **Input**: Source schemas $S_1, \ldots, S_n$.
Output: A set of possible mediated schemas.
1: Compute $\mathcal{A} = \{a_1, \ldots, a_m\}$, the set of all source attributes;
2: **for each** ($j \in [1, m]$)
Compute frequency $f(a_j) = \frac{|\{i \in [1,n] | a_j \in S_i\}|}{n}$;
3: Set $\mathcal{A} = \{a_j | j \in [1, m], f(a_j) \geq \theta\}$; *//$\theta$ is a threshold*
4: Construct a weighted graph $G(V, E)$, where (1) $V = \mathcal{A}$, and (2) for each $a_j, a_k \in \mathcal{A}$, $s(a_j, a_k) \geq \tau - \epsilon$, there is an edge (a_j, a_k) with weight $s(a_j, a_k)$;
5: Mark all edges with weight less than $\tau + \epsilon$ as *uncertain*;
6: **for each** (uncertain edge $e = (a_1, a_2) \in E$)
Remove e from E if (1) a_1 and a_2 are connected by a path with only certain edges, or (2) there exists $a_3 \in V$, such that a_2 and a_3 are connected by a path with only certain edges and there is an uncertain edge (a_1, a_3);
7: **for each** (subset of uncertain edges)
Omit the edges in the subset and compute a mediated schema where each connected component in the graph corresponds to an attribute in the schema;
8: **return** distinct mediated schemas.

Algorithm 1: Generate all possible mediated schemas.

4.2 Creating a p-med-schema

We now show how to extend the algorithm we just described to create a probabilistic mediated schema $\mathbf{M}$. Given source tables $S_1, \ldots, S_n$, we first construct the multiple schemas $M_1, \ldots, M_p$ in $\mathbf{M}$, and then assign each of them a probability.

We exploit two pieces of information available in the source tables: (1) pairwise similarity of source attributes; and (2) statistical co-occurrence properties of source attributes. Whereas the first piece of information tells us when two attributes are likely to be similar, the second tells us when two attributes are likely to be different. Consider for example, source table schemas

```
S1: (name,address,email-address)
S2: (name,home-address)
```

Pairwise string similarity would indicate that attribute address can be similar to both email-address and home-address. However, since the first source table contains address and email-address together, they cannot refer to the same concept. Hence, the first table suggests address is different from email-address, making it more likely that address refers to home-address.

Algorithm 1 describes how we create the multiple mediated schemas in $\mathbf{M}$ from $S_1, \ldots, S_n$ and a pairwise similarity function s. Steps 1–3 of the algorithm find the attributes that occur frequently in the sources. Steps 4 and 5 construct the graph of these high-frequency attributes. Unlike the graph constructed in Section 4.1, we allow an error ϵ on the threshold τ for edge weights. We thus have two kinds of edges: *certain edges*, having weight at least $\tau + \epsilon$, and *uncertain edges*, having weight between $\tau - \epsilon$ and $\tau + \epsilon$.

Steps 6-8 describe the process of obtaining multiple mediated schemas. Specifically, a mediated schema in $\mathbf{M}$ is created for every subset of the uncertain edges. For every subset, we consider the graph resulting from omitting that subset from the graph. The mediated schema includes a mediated attribute for each connected component in the resulting graph. Since, in the worst case, the number of resulting graphs is exponential in the number of uncertain edges, the parameter ϵ needs to be chosen carefully. In addi-

0: **Input:** Possible mediated schemas $M_1, \ldots, M_l$ and source schemas $S_1, \ldots, S_n$.
Output: $Pr(M_1), \ldots, Pr(M_l)$.
1: **for each** ($i \in [1, l]$)
Count the number of source schemas that are consistent with M_i, denoted as c_i;
2: **for each** ($i \in [1, l]$) Set $Pr(M_i) = \frac{c_i}{\sum_{i=1}^{l} c_i}$.

Algorithm 2: Assign probabilities to possible mediated schemas.

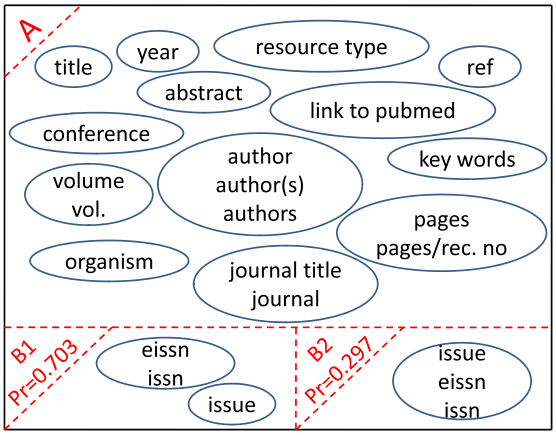

Figure 3: The p-med-schema for a set of bibliography sources. Each oval in the graph represents an attribute in the mediated schemas. The p-med-schema contains two possible schemas, the first containing attributes in regions A and B1, and the second containing attributes in regions A and B2. They have probabilities 0.703 and 0.297 respectively.

tion, Step 6 removes uncertain edges that when omitted will not lead to different mediated schemas. Specifically, we remove edges that connect two nodes already connected by certain edges. Also, we consider only one among a set of uncertain edges that connect a particular node with a set of nodes that are connected by certain edges.

Our next step is to compute probabilities for possible mediated schemas that we have generated. As a basis for the probability assignment, we first define when a mediated schema is *consistent with* a source schema. The probability of a mediated schema in **M** will be the proportion of the number of sources with which it is consistent.

DEFINITION 4.1 (CONSISTENCY). *Let M be a mediated schema for sources $S_1, \ldots, S_n$. We say M is* consistent with *a source schema $S_i, i \in [1, n]$, if there is no pair of attributes in S_i that appear in the same cluster in M.*

Intuitively, a mediated schema is consistent with a source only if it does not group distinct attributes in the source (and hence distinct real-world concepts) into a single cluster. Algorithm 2 shows how to use the notion of consistency to assign probabilities on the p-med-schema.

EXAMPLE 4.2. *We applied our algorithm on a data set containing 649 source tables about bibliographies extracted from HTML tables on the web (we shall describe the data set in more detail in Section 7.1). We used a string similarity measure for the pairwise attribute comparison. We used a frequency threshold θ of 10%; hence, only attributes that appeared in at least 10% of the tables were clustered. We used an edge weight threshold of $\tau = 0.85$ and error bar of $\epsilon = 0.02$.*

Figure 3 shows the p-med-schema generated by our algorithm. As we have an uncertain edge between issue *and* issn*, the result p-med-schema contains two possible mediated schemas, where one groups* eissn, issn *and* issue *and the other keeps* issue *apart.*

First of all, we generated a very good clustering, with semantically similar attributes grouped together (e.g., author, author(s), *and* authors *are grouped and* pages *and* pages/rec. no *are grouped). Second, among the several hundreds of distinct source attribute names, our mediated schema contains mostly attributes that are relevant to the domain. Note that many of our source tables are about Biology and Chemistry, so although attributes* organism *and* link to pubmed *are not central to bibliographies in general, they occur in a large fraction of source tables and are still selected. Finally, notice that the three attributes* issue, eissn, *and* issn *are clustered differently in M_1 and M_2. Since a large number of source schemas contain both* issue *and* issn *(or* eissn*), they are consistent with only M_1 but not M_2; thus, M_1 has a higher probability than M_2.* □

5. P-MAPPING GENERATION

We now address the problem of generating a p-mapping between a source schema and a mediated schema. We begin by computing weighted correspondences between the source attributes and the mediated attributes. However, as we explain shortly, there can be *multiple* p-mappings that are consistent with a given set of weighted correspondences. Of all such p-mappings we choose the one that maximizes the entropy of the probability assignment.

5.1 Computing weighted correspondences

A *weighted correspondence* between a pair of attributes specifies the degree of semantic similarity between them. Let $S(a_1, \ldots, a_m)$ be a source schema and $M(A_1, \ldots, A_n)$ be a mediated schema. We denote by $C_{i,j}, i \in [1, m], j \in [1, n]$, the weighted correspondence between a_i and A_j and by $p_{i,j}$ the weight of $C_{i,j}$. Our first step is to compute a weighted correspondence between every pair of attributes. Recall that the A_j's are clusters of attributes. We compute the weighted correspondence from the similarity between a_i and each attribute in A_j as follows[1]:

$$p_{i,j} = \sum_{a \in A_j} s(a_i, a).$$

Whenever the similarity $p_{i,j}$ is below a certain threshold, we set it to 0, thereby ensuring that clearly incorrect mappings are not generated.

Although weighted correspondences tell us the degree of similarity between pairs of attributes, they do not tell us *which* mediated attribute a source attribute should map to. For example, whereas a source attribute phone is more similar to the mediated attribute {phone, hPhone} than to {oPhone}, it could still make sense to map phone to either of them in a schema mapping. In fact, given a set of weighted correspondences, there could be a *set* of p-mappings that are consistent with it. We can define the one-to-many relationship between sets of weighted correspondences and p-mappings by specifying when a p-mapping is *consistent with* a set of weighted correspondences.

[1]Although we could have used avg or max instead of sum as well, the sum of pairwise similarities looks at the cluster as a whole to determine how well a_i is connected with the cluster.

DEFINITION 5.1 (CONSISTENT P-MAPPING). *A p-mapping* pM *is* consistent with *a weighted correspondence* $C_{i,j}$ *between a pair of source and target attributes if the sum of the probabilities of all mappings* $m \in pM$ *containing correspondence* (i,j) *equals* $p_{i,j}$*; that is,*

$$p_{i,j} = \sum_{m \in pM, (i,j) \in m} \Pr(m).$$

A p-mapping is consistent with *a set of weighted correspondences* $\mathbf{C}$ *if it is consistent with each weighted correspondence* $C \in \mathbf{C}$. □

However, not every set of weighted correspondences admits a consistent p-mapping. The following theorem shows under which conditions a consistent p-mapping exists, and establishes a normalization factor for weighted correspondences that will guarantee the existence of a consistent p-mapping.

THEOREM 5.2. *Let* $\mathbf{C}$ *be a set of weighted correspondences between a source schema* $\mathbf{S}(a_1, \ldots, a_m)$ *and a mediated schema* $\mathbf{M}(A_1, \ldots, A_n)$.

- *There exists a consistent p-mapping with respect to* $\mathbf{C}$ *if and only if (1) for every* $i \in [1,m]$, $\sum_{j=1}^{n} p_{i,j} \leq 1$ *and (2) for every* $j \in [1,n]$, $\sum_{i=1}^{m} p_{i,j} \leq 1$.
- *Let*

$$M' = max\{max_i\{\sum_{j=1}^{n} p_{i,j}\}, max_j\{\sum_{i=1}^{m} p_{i,j}\}\}.$$

Then, for each $i \in [1,m]$, $\sum_{j=1}^{n} \frac{p_{i,j}}{M'} \leq 1$ *and for each* $j \in [1,n]$, $\sum_{i=1}^{m} \frac{p_{i,j}}{M'} \leq 1$. □

Proof sketch: The second part of the theorem is trivial and the first part is proved as follows.

Only if: Suppose in contrast, there exists $i_0 \in [1,m]$ such that $\sum_{j=1}^{n} p_{i_0,j} > 1$. Let pM be the p-mapping that is consistent with $\mathbf{C}$. Let $\bar{m}$ be the set of mappings in pM that map a_{i_0} to some A_j's. Then, $\sum_{m \in \bar{m}} Pr(m) = \sum_{j=1}^{n} p_{i_0,j} > 1$ and so the sum of all probabilities in pM is greater than 1, contradicting the definition of p-mappings. Similarly, we can prove the second condition.

If: We transform the problem of constructing a consistent p-mapping to a problem of finding a set of bipartite matchings in a graph. The vertex sets in the bipartite graph correspond to source and mediated attributes respectively, and edges correspond to $p_{i,j}$'s. We prove the existence of a solution to the transformed problem by induction on the maximum number of edges for any vertex. □

Based on Theorem 5.2, we normalize the weighted correspondences we generated as described previously by dividing them by M'; that is,

$$p'_{i,j} = \frac{p_{i,j}}{M'}.$$

5.2 Generating p-mappings

To motivate our approach to generating p-mappings, consider the following example. Consider a source schema (A, B) and a mediated schema (A', B'). Assume we have computed the following weighted correspondences between source and mediated attributes: $p_{A,A'} = 0.6$ and $p_{B,B'} = 0.5$ (the rest are 0).

There are an infinite number of p-mappings that are consistent with this set of weighted correspondences and below we list two:

pM_1:

```
m1: (A,A'), (B,B'): 0.3
m2: (A,A'): 0.3
m3: (B,B'): 0.2
m4: empty: 0.2
```

pM_2:

```
m1: (A,A'), (B,B'): 0.5
m2: (A,A'): 0.1
m3: empty: 0.4
```

In a sense, pM_1 seems better than pM_2 because it assumes that the similarity between A and A' is independent of the similarity between B and B'.

In the general case, among the many p-mappings that are consistent with a set of weighted correspondences $\mathbf{C}$, we choose the one with the *maximum entropy*; that is, the p-mappings whose probability distribution obtains the maximum value of $\sum_{i=1}^{l} -p_i * log p_i$. In the above example, pM_1 obtains the maximum entropy.

The intuition behind maximum entropy is that when we need to select among multiple possible distributions on a set of exclusive events, we choose the one that does not favor any of the events over the others. Hence, we choose the distribution that does not *introduce new information* that we didn't have apriori. The principle of maximum entropy is widely used in other areas such as natural language processing [4, 24].

To create the p-mapping, we proceed in two steps. First, we enumerate all possible one-to-one schema mappings between S and M that contain a subset of correspondences in $\mathbf{C}$. Second, we assign probabilities on each of the mappings in a way that maximizes the entropy of our result p-mapping.

Enumerating all possible schema mappings given $\mathbf{C}$ is trivial: for each subset of correspondences, if it corresponds to a one-to-one mapping, we consider the mapping as a possible mapping.

Given the possible mappings $m_1, \ldots, m_l$, we assign probabilities $p_1, \ldots, p_l$ to $m_1, \ldots, m_l$ by solving the following constraint optimization problem (OPT):

`maximize` $\sum_{k=1}^{l} -p_k * \log p_k$ `subject to:`

1. $\forall k \in [1,l], 0 \leq p_k \leq 1$,
2. $\sum_{k=1}^{l} p_k = 1$, and
3. $\forall i,j : \sum_{k \in [1,l], (i,j) \in m_k} p_k = p_{i,j}$.

We can apply existing technology (such as [11]) in solving the OPT optimization problem. Although finding maximum-entropy solutions in general is costly, our experiments show that the execution time is reasonable for a one-time process; in addition, we can reduce the search space by considering *group p-mappings* [10], which divides the weighted correspondences into groups to localize the uncertainty.

6. P-MEDIATED-SCHEMA CONSOLIDATION

To complete the fully automatic setup of the data integration system, we consider the problem of consolidating a probabilistic mediated schema into a single mediated schema and creating p-mappings to the consolidated schema. We require that the answers to queries over the consolidated schema be equivalent to the ones over the probabilistic mediated schema.

The main reason to consolidate the probabilistic mediated schema into a single one is that the user expects to see a single schema. In addition, consolidating to a single schema has the advantage of more efficient query answering: queries now need to be

0: **Input:** Mediated schemas $M_1, \ldots, M_l$.
Output: A consolidated single mediated schema T.
1: Set $T = M_1$.
2: **for** $(i = 2, \ldots, l)$ modify T as follows:
3: **for each** (attribute A' in M_i)
4: **for each** (attribute A in T)
5: Divide A into $A \cap A'$ and $A - A'$;
6: **return** T.

Algorithm 3: Consolidate a p-med-schema.

rewritten and answered based on only one mediated schema. We note that in some contexts, it may be more appropriate to show the application builder a set of mediated schemas and let her select one of them (possibly improving on it later on).

Consolidating a p-med-schema: Consider a p-med-schema $\mathbf{M} = \{(M_1, Pr(M_1)), \ldots, (M_l, Pr(M_l))\}$. We consolidate $\mathbf{M}$ into a single mediated schema T. Intuitively, our algorithm (see Algorithm 3) generates the "coarsest refinement" of the possible mediated schemas in $\mathbf{M}$ such that every cluster in any of the M_i's is equal to the union of a set of clusters in T. Hence, any two attributes a_i and a_j will be together in a cluster in T if and only if they are together in every mediated schema of $\mathbf{M}$. The algorithm initializes T to M_1 and then modifies each cluster of T based on clusters from M_2 to M_l.

EXAMPLE 6.1. *Consider a p-med-schema $M = \{M_1, M_2\}$, where M_1 contains three attributes $\{a_1, a_2, a_3\}$, $\{a_4\}$, and $\{a_5, a_6\}$, and M_2 contains two attributes $\{a_2, a_3, a_4\}$ and $\{a_1, a_5, a_6\}$. The target schema T would then contain four attributes: $\{a_1\}$, $\{a_2, a_3\}$, $\{a_4\}$, and $\{a_5, a_6\}$.* □

Note that in practice the consolidated mediated schema is the same as the mediated schema that corresponds to the weighted graph with only certain edges. Here we show the general algorithm for consolidation, which can be applied even if we do not know the specific pairwise similarities between attributes.

Consolidating p-mappings: Next, we consider consolidating p-mappings specified w.r.t. $M_1, \ldots, M_l$ to a p-mapping w.r.t. the consolidated mediated schema T. Consider a source S with p-mappings $pM_1, \ldots, pM_l$ for $M_1, \ldots, M_l$ respectively. We generate a single p-mapping pM between S and T in three steps. First, we modify each p-mapping $pM_i, i \in [1, l]$, between S and M_i to a p-mapping pM_i' between S and T. Second, we modify the probabilities in each pM_i'. Third, we consolidate all possible mappings in pM_i''s to obtain pM. The details are as follows.

1. **For each** $i \in [1, l]$**, modify p-mapping** pM_i**:** Do the following for every possible mapping m in pM_i:
   - For every correspondence $(a, A) \in m$ between source attribute a and mediated attribute A in M_i, proceed as follows. (1) Find the set of all mediated attributes B in T such that $B \subset A$. Call this set $\overline{B}$. (2) Replace (a, A) in m with the set of all (a, B)'s, where $B \in \overline{B}$.

   Call the resulting p-mapping pM_i'.
2. **For each** $i \in [1, l]$**, modify probabilities in** pM_i'**:** Multiply the probability of every schema mapping in pM_i' by $Pr(M_i)$, which is the probability of M_i in the p-med-schema. (Note that after this step the sum of probabilities of all mappings in pM_i' is not 1.)
3. **Consolidate** pM_i'**'s:** Initialize pM to be an empty p-mapping (i.e., with no mappings). For each $i \in [1, l]$, *add* pM_i' to pM as follows:
   - For each schema mapping m in pM_i' with probability p: *if m is in pM, with probability p'*, modify the probability of m in pM to $(p + p')$; *if m is not in pM*, then add m to pM with probability p.

   The resulting p-mapping, pM, is the final consolidated p-mapping. The probabilities of all mappings in pM add to 1.

Note that Step 2 can map one source attribute to multiple mediated attributes; thus, the mappings in the result pM are one-to-many mappings, and so typically different from the p-mapping generated directly on the consolidated schema. The following theorem shows that the consolidated mediated schema and the consolidated p-mapping are equivalent to the original p-med-schema and p-mappings.

THEOREM 6.2 (MERGE EQUIVALENCE). *For all queries Q, the answers obtained by posing Q over a p-med-schema $\mathbf{M} = \{M_1, \ldots, M_l\}$ with p-mappings $pM_1, \ldots, pM_l$ is equal to the answers obtained by posing Q over the consolidated mediated schema T with consolidated p-mapping pM.* □

PROOF. Consider a mediated schema $M_i, i \in [1, l]$, and the associated p-mapping pM_i. Let pM_i' be the p-mapping obtained by modifying pM_i in our algorithm. Let Q be a query. If an attribute in Q is mapped to a source attribute under pM_i, based on our construction of pM_i' it will also be mapped to the same source attribute under pM_i'. Therefore, Q posed over M_i with p-mapping pM_i returns the same set of answer tuples as Q posed over T with p-mapping pM_i', whereas each returned tuple's probability is multiplied by $Pr(M_i)$. Finally, the probability of an answer tuple when Q is posed over the p-med-schema is the weighted sum of the probabilities of the tuple returned by posing Q over each M_i, weighted by probability $Pr(M_i)$. Hence Q returns the same answer over the original and consolidated schemas. □

7. EXPERIMENTS

We now describe experiments that validate the performance of our algorithms. Our main goal is to examine the quality of answers obtained from a completely automatic setup of a data integration system. In addition, we describe experiments validating the use of a probabilistic mediated schema, and showing that our setup time scales well with the number of sources.

7.1 Experimental setup

We built a data integration system, referred to as UDI, based on the techniques described in the previous sections. UDI takes a set of data sources and automatically creates a mediated schema and a probabilistic schema mapping between each data source and the mediated schema. UDI accepts select-project queries on the exposed mediated schema and returns answers ranked by their probabilities. We did not consider joins, as our mediated schema contained a single table. For each given query, UDI transforms it into a set of queries on the data sources according to the probabilistic schema mappings, retrieves answers from individual data sources, and then combines the answers assuming that the data sources are independent (as we described in Section 2).

For the purposes of our evaluation, it suffices to store each source as a single table in a database, rather than access data sources at query time. We used MySQL for storing the data, and implemented the query processor in Java. We used the SecondString tool [2] to compute the Jaro Winkler similarity [27] of attribute names in pairwise attribute comparison. We used Knitro [1] to solve the entropy maximization problem in p-mapping construction. We conducted

Table 1: Number of tables in each domain and keywords that identify the domain. Each domain contains 50 to 800 data sources.

Domain	#Src	Keywords
Movie	161	*movie* and *year*
Car	817	*make* and *model*
People	49	*name*, one of *job* and *title*, and one of *organization*, *company* and *employer*
Course	647	one of *course* and *class*, one of *instructor*, *teacher* and *lecturer*, and one of *subject*, *department* and *title*
Bib	649	*author*, *title*, *year*, and one of *journal* and *conference*

our experiments on a Windows Vista machine with Intel Core 2 GHz CPU and 2GB memory.

For our experiments, we set the pairwise similarity threshold for creating the mediated schema to 0.85, the error bar for uncertain edges to 0.02, the frequency threshold for considering attributes in the mediated schema to 10%, and the correspondence threshold to 0.85. Our experiments showed similar results even when the above constants were varied by 20%.

Data and queries: We evaluated our system using real data sets from five domains: Movie, Car, People, Course, and Bibliography. The tables were selected from a larger corpus of HTML tables on the web for which attribute labels were clearly present. We selected the tables for each domain by searching for tables that contained certain keywords (see the third column of Table 1). Each of the tables typically contain tens to a few hundreds of tuples. Table 1 also shows the number of tables extracted for each domain.

For each domain, we chose 10 queries, each containing one to four attributes in the `SELECT` clause and zero to three predicates in the `WHERE` clause. The attributes in the `SELECT` and `WHERE` clauses are attributes from the exposed mediated schema. Each predicate contains an attribute, an operator, and a value, where the operator can be $=, \neq, <, \leq, >, \geq$ and `LIKE`. When we selected the queries, we varied selectivity of the predicates and likelihood of the attributes being mapped correctly to cover all typical cases.

Overview of experiments: Our main goal is to see how well we can do without any human intervention in setting up a data integration system. Hence, Section 7.2 compares the answers obtained by UDI with those that would be obtained from a data integration system in which the mediated schema and schema mappings were created manually. In the absence of UDI, the typical approach imagined to bootstrap pay-as-you-go data integration systems is to consider all the data sources as a collection of text documents and apply keyword search techniques. Section 7.3 compares UDI to this approach and to several variations of UDI where some of its features are omitted. Section 7.4 demonstrates the value of probabilistic mediated schemas and Section 7.5 shows the quality of the mediated schema we create. Finally, Section 7.6 touches on efficiency issues in UDI.

Performance measure: In our experiments we used three standard metrics: *precision*, *recall* and *F-measure*. Let $\bar{A}$ be the set of answers that our system generates and $\bar{B}$ be the set of answers in the golden standard. The three metrics are defined as follows: (1) *Precision*: $P = \frac{|\bar{A} \cap \bar{B}|}{|A|}$; (2) *Recall*: $R = \frac{|\bar{A} \cap \bar{B}|}{|B|}$; and (3) *F-measure*: $F = \frac{2*P*R}{P+R}$.

To show how well our system ranks the answers, we plotted the *recall/precision curve (R-P curve)* for certain domains. An R-P curve varies recall on the X-axis and precision on the Y-axis. An ideal R-P curve is a horizontal line with a precision of 1.

Table 2: Precision, recall and F-measure of query answering of the UDI system compared with a manually created integration system. The results show that UDI obtained a high accuracy in query answering.

Domain	Precision	Recall	F-measure
Golden standard			
People	1	.849	.918
Bib	1	.852	.92
Approximate golden standard			
Movie	.95	1	.924
Car	1	.917	.957
Course	.958	.984	.971
People	1	1	1
Bib	1	.955	.977

Since most of the approaches we compared against do not return ranked answers, to be fair to these approaches we do not remove duplicates before measuring precision/recall, although we did observe similar results even with duplicates eliminated. Only for the R-P curve experiment duplicates were removed, as the experiment needs tuple probabilities.

7.2 UDI v.s. manual integration

To compare UDI with manual integration, we constructed a *golden standard* by manually creating mediated schemas and schema mappings in two domains (People and Bib). To answer queries, we followed the traditional data integration approach, reformulating the query using the schema mappings, and taking the union of the results obtained from the relevant data sources.

Since the manual integration with the number of sources we have is a significant undertaking, for the other three domains we compared UDI with an approximation to a golden standard. Specifically, we retrieved all answers generated by UDI as well as the answers obtained by directly posing the query over each data source, and then manually removed incorrect answers. Note that the approximate golden standard will still be high in precision but may loose recall compared with the true golden standard. We executed ten queries in each domain and report the average precision, recall and F-measure of the returned results.

Results: Table 2 shows that we obtain high precision and recall for all domains. In comparison to the true golden standard, we obtained a recall of about 0.85 on the two domains, and in comparison to the approximate golden standard, we obtained a recall of over 0.9 in all cases and over 0.95 in four of the domains. Extrapolating from the discrepancy in the Bib and People domains between the true and approximate golden standards, we expect that we would obtain recall around 0.8-0.85 with respect to the golden standard on all domains. Our precision and recall results validate the main point of our work: we are able to completely automatically set up a data integration system to obtain high-quality results, and therefore be in an excellent starting point to improve the data integration system with time.

We believe that the main method to improve our results is to employ a better schema matcher. Our matcher considered only similarity of attribute names and did not look at values in the corresponding columns or other clues. Hence, we did not detect that location and address are similar attributes. We also suffered some loss of recall because we set a high threshold to choose attribute

correspondences in order to reduce the number of correspondences considered in the entropy maximization. While there is a chance that a p-mapping generated automatically can contain incorrect mappings, leading to low precision, this did not happen very often in our experiments. This is also due to the high threshold we applied to correspondences, therefore preserving mostly correct ones.

7.3 Competing automatic approaches

Next, we compared our system with alternative approaches for bootstrapping a data integration system. The first approach is to consider the data sources as a collection of documents and perform keyword search. We tested three variants of this approach. In each one, given a query Q, we generated a keyword query Q' by taking all attribute names in the SELECT clause and values in the WHERE clause of Q. Using MySQL's keyword search engine, we tried the following variants:

- KEYWORDNAIVE: return tuples with *any* of the keywords in the query Q'.
- KEYWORDSTRUCT: classify keywords in Q' according to the schema of S: if a keyword K occurs in an attribute name of S, consider K as a *structure term*; otherwise, consider K as a *value term*. Return tuples with *any* of the value terms.
- KEYWORDSTRICT: classify keywords in Q' as in KEYWORDSTRUCT and return tuples with *all* value terms.

The second alternative approach, SOURCE, answers Q directly on every data source that contains all the attributes in Q, and takes the union of returned answers.

Finally, we considered the TOPMAPPING approach, where we use the consolidated mediated schema but consider only the schema mapping with the highest probability, rather than all the mappings in the p-mapping.

Results: Figure 4 shows that UDI obtains better results than the other approaches. We make the following three observations.

First, not surprisingly, all variants of KEYWORD performed poorly on all domains. While we fully expected the precision and recall of keyword-based solutions to be poor, the main point of the experiment was to measure how poorly they did compared to UDI, since keyword search engines offer a simple and general solution for searching any kind of information.

Second, the SOURCE approach always obtained high precision, but its recall was low. The reason for this is that in essence, SOURCE considers only attribute-identity mappings between terms in the query and terms in the sources. Therefore, SOURCE will miss any answer that needs a more subtle mapping. In the Course domain, the precision of SOURCE is below 1 because a numeric comparison performed on a string data type generates incorrect answers.

Third, the precision of TOPMAPPING varied a lot from domain to domain. When the mapping with the highest probability was indeed a correct mapping, TOPMAPPING obtained high precision; but otherwise, TOPMAPPING returned incorrect answers and resulted in low precision. In any case, the recall of TOPMAPPING was low since it did not consider other correct mappings (in the Bib domain, TOPMAPPING failed to return any correct answers). The recall of TOPMAPPING is even lower than SOURCE because the highest-probability mapping often did not produce all identity mappings, picked by SOURCE.

7.4 Contribution of p-med-schema

To examine the contribution of using a probabilistic mediated schema in improving query answering results, we considered two approaches that create a *single* mediated schema:

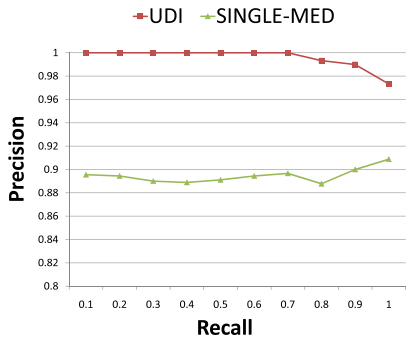

Figure 6: R-P curves for the Movie domain. The experimental results show that UDI ranks query answers better.

- SINGLEMED: create a deterministic mediated schema based on the algorithm in Section 4.1.
- UNIONALL: create a deterministic mediated schema that contains a singleton cluster for each frequent[2] source attribute.

Figure 5 compares UDI with the above methods. We observed that although SINGLEMED and UNIONALL perform better than the alternatives considered in the previous section, they still do not perform as well as UDI. Specifically, SINGLEMED obtained similar precision as UDI but a lower recall, because it missed some correct mediated schemas and the accompanying mappings. UNIONALL obtained high precision but much lower recall. This is because UNIONALL does not group source attributes with the same semantics, resulting in correspondences with low weights; thus, we may miss some correct attribute correspondences in p-mapping generation. In addition, not grouping similar attributes leads to an explosion in the number of possible mappings in p-mappings. In the Bib domain, UNIONALL ran out of memory in system setup.

We observed from Figure 5 that the average F-measure of UDI was only slightly better than SINGLEMED. This is because UDI beat SINGLEMED in recall only for queries that contain ambiguous attributes. For other queries the recall was the same using both the approaches.

We took a closer look at how UDI and SINGLEMED rank their answers by plotting the R-P curve in the Movie domain in Figure 6 (the other domains exhibited similar behavior). Recall was varied on the x-axis by taking top-K answers based on probabilities. For instance, to compute the UDI precision for 50% recall, we find K such that the top-K answers in UDI have 50% recall. We then compute the precision for these K answers.

Although UDI and SINGLEMED obtained similar precision in this domain, UDI ranked the returned answers better: the R-P curve of UDI has a better shape in the sense that with a fixed recall, it has a higher precision. Note the precision at recall of 1 is different from those of Figure 5 for two reasons. First, to rank tuples, we eliminated duplicates and combined their probabilities; hence, the answer set does not contain duplicates, unlike in the answers used for Figure 5. Moreover, in the UDI domain, many incorrect answers were ranked below all the correct answers, so we got a precision higher than that in Figure 5.

7.5 Quality of mediated schema

Next, we tested the quality of the probabilistic mediated schema against a manually created schema. Recall that each mediated schema corresponds to a clustering of source attributes. Hence,

[2]We use the same threshold to decide if a source attribute occurs frequently.

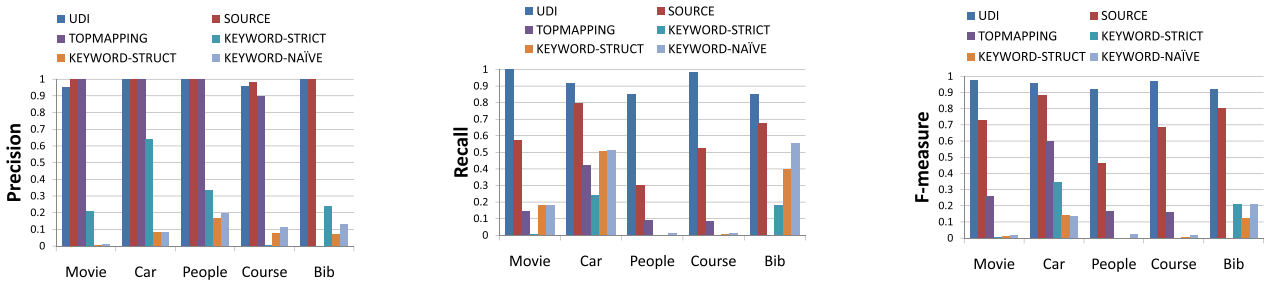

Figure 4: Performance of query answering of the UDI system and alternative approaches. The UDI system obtained the highest F-measure in all domains.

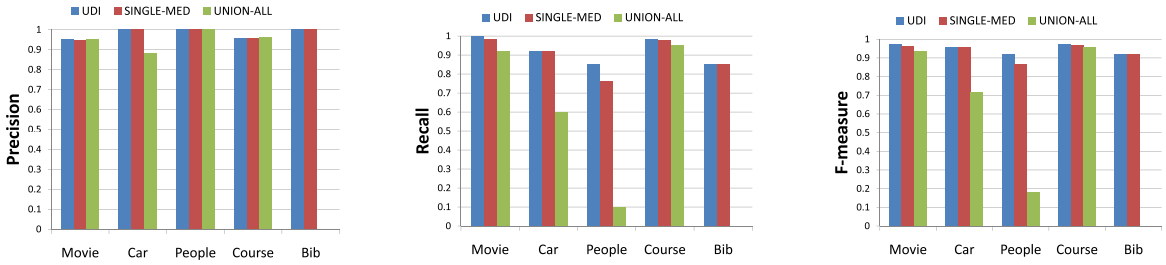

Figure 5: Performance of query answering of the UDI system and approaches that generate deterministic mediated schemas. The experimental results show that using a probabilistic mediated schema improves query answering performance. Note that we did not plot the measures for UNIONALL in the Bib domain as this approach ran out of memory in system setup.

Table 3: Precision, recall and F-measure of p-med-schemas generated by UDI.

Domain	Precision	Recall	F-measure
Movie	.97	.62	.76
Car	.68	.88	.77
People	.76	.86	.81
Course	.83	.58	.68
Bib	.77	.81	.79
Avg	.802	.75	.762

we measured its quality by computing the *precision*, *recall* and *F-measure* of the clustering, where we counted how many *pairs* of attributes are correctly clustered. To compute the measures for probabilistic mediated schemas, we computed the measures for each individual mediated schema and summed the results weighted by their respective probabilities.

Table 3 shows that we obtained high precision and recall, averaging 0.8 and 0.75 respectively, over five domains. We expect that if we used a more sophisticated pair-wise attribute matching algorithm, our results would be significantly better.

7.6 Setup efficiency

Finally, we measured the time taken to set up the data integration system. To examine the effect of the number of data sources on the efficiency of the system, we started with a subset of the data sources in a domain and gradually added more data sources. We report our results on the Car domain, as it contains the largest number of data sources. We observed similar trends for other domains.

Figure 7 shows the time to set up the system, which includes four steps: (1) importing source schemas, (2) creating a p-med-schema, (3) creating a p-mapping between each source schema and each possible mediated schema, and (4) consolidating the p-med-schema and the p-mappings. For the entire data set, consisting of 817 data sources, it took roughly 3.5 minutes in total to configure the integration system. Considering the typical amount of time it takes to set up data integration applications, few minutes is a negligible amount of time. Furthermore, Figure 7 shows that the setup time increased linearly with the number of data sources. We note that the most time-consuming step in system setup is to solve the maximum-entropy problem.

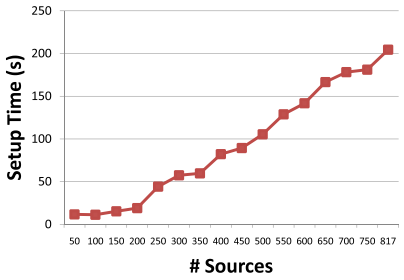

Figure 7: System setup time for the Car domain. When the number of data sources was increased, the setup time increased linearly.

We also measured the time to answer queries in our system. With 817 data sources, UDI answered queries in no more than 2 seconds. Since UDI is storing all the data locally and not communicating with live data sources, this number cannot be considered representative of a real data integration system. Instead, the number illustrates that answering queries over the mediated schema and

the p-mappings we create does not add significant overhead.

8. RELATED WORK

We briefly describe related work on automated creation of mediated schemas and on schema-mapping creation. In contrast to previous work that focused on each of these problems in isolation, ours is the first that handled the entire process of setting up a data integration application. The goal of our work is to be able to offer high-quality answers to queries without any human involvement.

Creating mediated schema: Most of the previous work on automatically creating mediated schemas focused on the theoretical analysis of the semantics of merging schemas and the choices that need to be made in the process [3, 6, 15, 17, 22, 25]. The goal of these work was to make as many decisions automatically as possible, but where some ambiguity arises, refer to input from a designer.

The work closest to ours is by He and Chang [14] who 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. Our probabilistic mediated schemas have 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 described in our motivating example in Section 2. 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, our approach is independent of a specific schema-matching technique, whereas their approach is tuned for constructing generative models and hence must rely on statistical properties of source schemas.

Magnani et al. [20] 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. Our technique focuses 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.

Schema mapping: Schema mapping has been studied extensively in the last ten years. Previous work has studied how to explore various clues, including attribute names, descriptions, data types, constraints, and data values, to understand the semantics of attributes and match attributes (see [26] for a survey and [5, 7, 8, 9, 14, 18, 28] for some work since then). In addition, some work considered how to create schema mappings by choosing a set of attribute correspondences that best conform to certain heuristic constraints involving the structure of the schema [8, 21]. Our algorithm takes existing schema matching techniques as a blackbox for attribute comparison, based on which we then create mediated schemas and probabilistic schema mappings.

Recent work has proposed notions to capture the uncertainty in data integration. Dong et al. [10] proposed the concept of probabilistic schema mapping and studied query answering with respect to such mappings, but they did not describe how to create such mappings. Magnani and Montesi [19] have empirically shown that top-k schema mappings can be used to increase the recall of a data integration process and Gal [13] described how to generate top-k schema matchings by combining the matching results generated by various matchers. The probabilistic schema mappings we generate are different as it contains 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. In Section 7 we have compared our system to TOPMAPPING, where we choose a single mapping from the sources to the mediated schema. A further refinement would be to choose the top-k mappings selected using one of the techniques above. Finally, Nottelmann and Straccia [23] 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.

9. CONCLUSIONS

We showed that it is possible to automatically set up a data integration application that obtains answers with high precision and recall. In doing so, we established a fairly advanced starting point for pay-as-you-go data integration systems. At its core, our system is built on modeling uncertainty in data integration systems. The main novel element we introduced to build our system is a probabilistic mediated schema, which is constructed automatically by analyzing the source schemas. We showed a set of experiments on five domains and hundreds of data sources that validated our approach.

Of course, setting up the data integration is just the first step in the process. Aside from improvements to this process, our future work will consider how to improve the data integration system with time. We believe that the foundation of modeling uncertainty will help pinpoint where human feedback can be most effective in improving the semantic integration in the system, in the spirit of [16]. In addition, we plan to extend our techniques to dealing with multiple-table sources, including mapping multi-table schemas, normalizing mediated schemas, and so on.

10. REFERENCES

[1] Knitro optimization software. http://www.ziena.com/knitro.htm.

[2] Secondstring. http://secondstring.sourceforge.net/.

[3] C. Batini, M. Lenzerini, and S. B. Navathe. A comparative analysis of methodologies for database schema integration. In *ACM Computing Surveys*, pages 323–364, 1986.

[4] A. L. Berger, S. A. D. Pietra, and V. J. D. Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, (1):39–71, 1996.

[5] J. Berlin and A. Motro. Database schema matching using machine learning with feature selection. In *Proc. of the 14th Int. Conf. on Advanced Information Systems Eng. (CAiSE02)*, 2002.

[6] P. Buneman, S. Davidson, and A. Kosky. Theoretical aspects of schema merging. In *Proc. of EDBT*, 1992.

[7] R. Dhamankar, Y. Lee, A. Doan, A. Y. Halevy, and P. Domingos. iMAP: Discovering complex semantic matches between database schemas. In *Proc. of ACM SIGMOD*, 2004.

[8] H. Do and E. Rahm. COMA - a system for flexible combination of schema matching approaches. In *Proc. of VLDB*, 2002.

[9] A. Doan, J. Madhavan, P. Domingos, and A. Y. Halevy. Learning to map between ontologies on the Semantic Web. In *Proc. of the Int. WWW Conf.*, 2002.

[10] X. Dong, A. Y. Halevy, and C. Yu. Data integration with uncertainty. In *Proc. of VLDB*, 2007.

[11] M. Dudik, S. J. Phillips, and R. E. Schapire. Performance guarantees for regularized maximum entropy density estimation. In *Proc. of the 17th Annual Conf. on Computational Learning Theory*, 2004.

[12] M. Franklin, A. Y. Halevy, and D. Maier. From databases to dataspaces: a new abstraction for information management. In

SIGMOD Record, pages 27–33, 2005.
[13] A. Gal. Why is schema matching tough and what can we do about it? *SIGMOD Record*, 35(4):2–5, 2007.
[14] B. He and K. C. Chang. Statistical schema matching across web query interfaces. In *Proc. of ACM SIGMOD*, 2003.
[15] R. Hull. Relative information capacity of simple relational database schemata. In *Proc. of ACM PODS*, 1984.
[16] S. Jeffery, M. Franklin, and A. Halevy. Pay-as-you-go user feedback for dataspace systems. In *Proc. of ACM SIGMOD*, 2008.
[17] L. A. Kalinichenko. Methods and tools for equivalent data model mapping construction. In *Proc. of EDBT*, 1990.
[18] J. Kang and J. Naughton. On schema matching with opaque column names and data values. In *Proc. of ACM SIGMOD*, 2003.
[19] M. Magnani and D. Montesi. Uncertainty in data integration: current approaches and open problems. In *VLDB workshop on Management of Uncertain Data*, pages 18–32, 2007.
[20] M. Magnani, N. Rizopoulos, P. Brien, and D. Montesi. Schema integration based on uncertain semantic mappings. *Lecture Notes in Computer Science*, pages 31–46, 2005.
[21] S. Melnik, H. G. Molina, and E. Rahm. Similarity flooding: A versatile graph matching algorithm. In *Proc. of ICDE*, pages 117–128, 2002.
[22] R. J. Miller, Y. Ioannidis, and R. Ramakrishnan. The use of information capacity in schema integration and translation. In *Proc. of VLDB*, 1993.
[23] H. Nottelmann and U. Straccia. Information retrieval and machine learning for probabilistic schema matching. *Information Processing and Management*, 43(3):552–576, 2007.
[24] S. D. Pietra, V. D. Pietra, and J. Lafferty. Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4):380–393, 1997.
[25] R. Pottinger and P. Bernstein. Creating a mediated schema based on initial correspondences. In *IEEE Data Eng. Bulletin*, pages 26–31, Sept 2002.
[26] E. Rahm and P. A. Bernstein. A survey of approaches to automatic schema matching. *VLDB Journal*, 10(4):334–350, 2001.
[27] S. E. Fienberg W. Cohen, P. Ravikumar. A comparison of string distance metrics for name-matching tasks. In *Proc. of IJCAI*, 2003.
[28] J. Wang, J. Wen, F. H. Lochovsky, and W. Ma. Instance-based schema matching for Web databases by domain-specific query probing. In *Proc. of VLDB*, 2004.

APPENDIX

A. PROOFS

Proof of Theorem 3.4:

1. Let the target mediated schema T contain the set of all source attributes that appear in some schema of $\mathbf{M}$, each in its own cluster. We construct the set of mappings in the result p-mapping pM as follows. For each mapping m_i with probability p_i in $\mathbf{pM}$ between S and mediated schema M_j in $\mathbf{M}$ having probability $p(M_j)$, we add a mapping m_{ij} to pM, with probability $p_i * p(M_j)$. m_{ij} is constructed as follows. If m_i maps attribute $B \in S$ to some cluster A in M_j, then m_{ij} maps B to all the singleton clusters in T that have an element in A. Hence if a query asks for some attribute in A in $\mathbf{M}$, due to M_j in the input it would retrieve B with probability $p_i * p(M_j)$ because of the probabilities of the corresponding mediated schema and mapping, and in our constructed target also it would retrieve B with the same probability because of m_{ij}.
2. Consider the following example. Let S be a source with two attributes a, and b, mediated schema T also have two singleton clusters $c_1 = \{a\}$ and $c_2 = \{b\}$. Suppose $\mathbf{pM}$ has two mappings m_1 with probability 0.5 mapping a in S to c_1 and b to c_2, and m_2 with probability 0.5 mapping a in S to c_2 and b to c_1. There exists no p-med-schema $\mathbf{M}$ and deterministic mappings m which would give results equivalent to the above for all queries. Intuitively, this is because, we would need to construct two distinct mediates schemas, each with two singleton clusters, and encoding the two mappings above. However, there exists just one distinct mediated schema with all singleton clusters for the set of two attributes $\{a, b\}$. □

Proof of Theorem 3.5: Consider a single data source $S(a_1, a_2)$ and a single-tuple instance of S: $D = \{(x_1, x_2)\}$.

Consider a p-med-schema $M = \{M_1, M_2\}$, where M_1 contains two mediated attributes $A_1 = \{a_1\}$ and $A_2 = \{a_2\}$ (i.e., singleton clusters), and M_2 contains one mediated attribute $A_3 = \{a_1, a_2\}$. We have $p(M_1) = 0.7$ and $p(M_2) = 0.3$.

Consider a set $\mathbf{pM} = \{pM_1, pM_2\}$ of p-mappings for M_1 and M_2 respectively. Both pM_1 and pM_2 are indeed deterministic (contain one single mapping with probability 1): pM_1 maps A_1 and A_2 to a_1 and a_2 respectively, and M_2 maps A_3 to a_1.

We claim that there does not exist a single mediated schema T with a p-mapping pM between S and T such that $Q_{T,pM}(D) = Q_{\mathbf{M},\mathbf{pM}}(D)$ for all Q. If there did exist such a T and pM, T would need to have a_1 and a_2 in different clusters. Otherwise, for query $Q1$: `SELECT a1, a2 FROM T`. Answer $Q1_{T,pM}(D)$ contains either tuple (x_1, x_1) or (x_2, x_2), but not (x_1, x_2), which occurs in $Q1_{\mathbf{M},\mathbf{pM}}(D)$. Thus, T has two attributes $A_4 = \{a_1\}$ and $A_5 = \{a_2\}$. Because pM is a one-to-one mapping, each possible mapping in pM can map a_1 to one of A_4 and A_5. Consider query $Q2$: `SELECT a1 FROM T`, $Q2_{\mathbf{M},\mathbf{pM}}(D)$ contains tuple (x_1) with probability 1. Thus, every possible mapping in pM must map a_1 to A_4 and so cannot map a_1 to A_5. Now consider query $Q3$: `SELECT a2 FROM T`. Answer $Q3_{\mathbf{M},\mathbf{pM}}(D)$ contains tuple (x_1) with probability .3; however, $Q3_{T,pM}(D)$ does not contain (x_1), leading to contradiction. □.

Proof of Theorem 5.2: The second part of the theorem is trivial and we prove the first part.

Only if: Suppose in contrast, there exists $i_0 \in [1, m]$ such that $\sum_{j=1}^{n} p_{i_0,j} > 1$. Let pM be the p-mapping that is consistent with $\mathbf{P}$. Let $\bar{m}$ be the set of mappings in pM that map a_{i_0} to some A_j's. Then, $\sum_{m \in \bar{m}} Pr(m) = \sum_{j=1}^{n} p_{i_0,j} > 1$ and so the sum of all probabilities in pM is greater than 1, contradicting the definition of p-mappings. Similarly, we can prove the second condition.

If: We prove the theorem assuming all $p_{i,j}$'s are rational numbers. If some of $p_{i,j}$'s are irrational, we can convert the problem to an equivalent problem with rational $p_{i,j}$'s. Details of this conversion are omitted.

First note that there exists a rational number q such that $\forall i, j$, $\frac{p_{i,j}}{q}$ is a positive integer. The existence of such a q follows from the fact that all $p_{i,j}$'s are rational. We transform the set of correspondences $\mathbf{C}$ to a new set of correspondences $\mathbf{C}'$ as follows: for each correspondence $C_{i,j} \in \mathbf{C}$ with probability $p_{i,j}$, there is a correspondence $C'_{i,j} \in \mathbf{C}'$ with weight $c_{i,j} = \frac{p_{i,j}}{q}$, which is an integer. Intuitively, between each pair of attributes there are $c_{i,j}$ *unit correspondences*, each of which has probability q.

Let M be the maximum number of correspondences for any source or mediated attribute. That is,

$M = max\{max_i(\sum_j c_{i,j}), max_j(\sum_i c_{i,j})\}$.

Therefore, using the condition from the theorem we have $M \cdot q \leq 1$. Hence it is sufficient for us to find a p-mapping that has $M + 1$ mappings, among which M mappings each has probability q and one mapping is empty and has probability $(1 - M \cdot q)$. For each mapping to have probability q, we need to pick at most one unit correspondence for every source or mediated attribute. Considering attributes as nodes, and correspondences as edges, we can reduce

our problem of finding such a p-mapping to the following bi-partite matching problem:

> Consider a bipartite graph $G(V_1, V_2, E)$, where E is a multi-set of edges between a vertex in V_1 and a vertex in V_2. Suppose all vertices are associated with at most M edges. Find M bipartite matchings between V_1 and V_2 such that every edge in E appears in exactly one bi-partite matching.

We prove the existence of a solution to the above problem by induction on M. Suppose $M = 1$, then every vertex is involved in at most one edge, and hence the set of all edges constitutes a bipartite matching. Let us now suppose there exists a solution for the problem when $M \leq k$; we prove the existence of a solution when $M = (k + 1)$. Suppose $M = (k + 1)$, consider all vertices that have $(k + 1)$ edges. Pick exactly one edge from every such vertex and add it to a set B. (First pick edges between a pair of nodes that each have $(k + 1)$ edges, and then pick edges with one endpoint having $(k+1)$ edges.) Since B contains at most one edge per vertex, it is a matching. And since the remaining graph has at most k edges per node, using the induction hypothesis we can find a set of k bipartite matchings S. The set of matchings $S \cup \{B\}$ is a solution to the problem. □