Statistical Schema Matching across Web Query Interfaces*

Bin He
Computer Science Department
University of Illinois at Urbana-Champaign
binhe@uiuc.edu

Kevin Chen-Chuan Chang
Computer Science Department
University of Illinois at Urbana-Champaign
kcchang@cs.uiuc.edu

ABSTRACT

Schema matching is a critical problem for integrating heterogeneous information sources. Traditionally, the problem of matching multiple schemas has essentially relied on finding pairwise-attribute correspondence. This paper proposes a different approach, motivated by integrating large numbers of data sources on the Internet. On this "deep Web," we observe two distinguishing characteristics that offer a new view for considering schema matching: First, as the Web scales, there are ample sources that provide structured information in the same domains (e.g., books and automobiles). Second, while sources proliferate, their aggregate schema vocabulary tends to converge at a relatively small size. Motivated by these observations, we propose a new paradigm, *statistical schema matching*: Unlike traditional approaches using pairwise-attribute correspondence, we take a holistic approach to match all input schemas by finding an underlying generative schema model. We propose a general statistical framework MGS for such hidden model discovery, which consists of hypothesis modeling, generation, and selection. Further, we specialize the general framework to develop Algorithm MGS_{sd}, targeting at *synonym discovery*, a canonical problem of schema matching, by designing and discovering a model that specifically captures synonym attributes. We demonstrate our approach over hundreds of real Web sources in four domains and the results show good accuracy.

1. INTRODUCTION

Schema matching is fundamental for enabling query mediation and data exchange across information sources. This paper attempts to consider the schema matching problem with a new paradigm: Traditionally, such matching has been approached mainly by finding *pairwise-attribute correspondence*, to construct an integrated schema for two or some (small number of) n sources. We observe that there are often challenges (and certainly also opportunities) to deal with large numbers of sources. As a different approach, we take a holistic view of all the attributes in input sources and attempt to find an underlying *unified model* which captures their matchings.

Such scenarios arise, in particular, for integrating databases across the Internet. On this "deep Web," numerous data sources provide structured information (e.g., amazon.com for books; cars.com for automobiles) accessible only via dynamic queries instead of static URL links. Each source accepts queries over its schema attributes (e.g., author and title for amazon.com) through its *query interface*. Thus, schema matching across query interfaces is clearly essential for mediating related sources of the same domains, which this paper specifically focuses on.

On the deep Web, we observe two distinguishing characteristics that offer a new view for schema matching: On one hand, we observe *proliferating sources*: As the Web scales, many data sources provide structured information in the same domains (e.g., there are numerous other "book" sources, such as bn.com). On the other hand, we also observe *converging vocabularies*: The aggregate schema vocabulary of these sources tends to converge at a relatively small size (e.g., these books sources share 12 frequent attributes, which account for 78% of all attribute occurrences).

These observations lead us to hypothesize that, underlying sources of the same domain, there exists a *hidden schema model*, which is a unified generative model that describes how schemas are generated from a finite *vocabulary* of attributes, with some probabilistic behavior. Under this hypothesis, it is natural that we have observed proliferating sources with converging vocabularies.

Motivated by this hypothesis, we explore a new paradigm, which we call *statistical schema matching*. Unlike traditional approaches using pairwise-attribute correspondence, given a set of input sources as observed schemas, we will find hidden models that are consistent, in a statistical sense, with the schemas observed. Using a scenario of matching several book sources, Figure 1 contrasts the two different approaches. Given a set of schemas as input, the traditional schema-matching approaches essentially rely on finding pairwise-attribute correspondence (e.g., author in Source S1 maps to name in S3) for eventually constructing a unified schema for all sources. In contrast, our approach hypothesizes and attempts to find, in the first place, a unified model, which gives the "structure" of the attributes across all sources (e.g., author, name, and writer are the same concept, and so are subject and category).

To realize statistical schema matching, we propose a general abstract framework, MGS, with three steps: (1) *Hypoth-*

*This material is based upon work partially supported by NSF Grant IIS-0133199. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the author(s) and do not necessarily reflect the views of the funding agencies.

SIGMOD 2003, June 9-12, 2003, San Diego, CA.

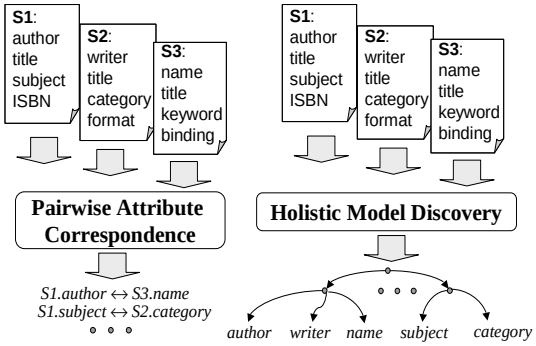


Figure 1: Two different matching approaches.

esis modeling: We first specify a parameterized structure of the hypothetical hidden models. Such models should capture the *target questions* of schema matching that we want to address– e.g., the model in Figure 1 targets at synonym discovery. (2) *Hypothesis generation*: We then generate all "consistent" models that instantiate the observed schemas with non-zero probabilities. (3) *Hypothesis selection*: Finally, we select models of sufficient statistical consistency with the observed schemas. Such an underlying model is likely the one that unifies the input schemas, and answers our target questions. We stress that the discovery of the underlying model can directly address the target questions (e.g., attribute correspondence). Also, the model is itself useful as a mediated schema (e.g., as a query front-end).

Further, we specialize the MGS framework for synonym discovery– a canonical problem in schema matching– across query interfaces of deep Web sources. We view each query interface as a set of attributes (for querying), or a schema, and our goal is to find the synonym attributes. In this Algorithm MGS_{sd}, we design a simple model to capture the target question of synonym attributes (i.e., multiple attributes are of the same "concept"). Given a set of schemas, MGS_{sd} generates hypothetical models, under which it is "possible" (with non-zero probabilities) to "observe" these schemas. Finally, MGS_{sd} adopts χ^2 hypothesis testing to select candidate models that are consistent with the observed schemas at a sufficient significance level.

We performed case studies for over 200 real sources in 4 domains: books (e.g., amazon.com), movies (e.g., blockbuster-.com), music records (e.g., towerrecords.com), and automobiles (e.g., cars.com). Our goals are two-fold: (1) Verify the phenomenons (of proliferating sources and converging vocabularies) that support our motivating hypotheses (Section 3). (2) Validate the performance of $\mathsf{MGS}_{sd}$with two suites of metrics: *model accuracy* and *target accuracy*. In either case, we observed good performance (Section 6).

In our development, we also observed several interesting issues. Can we deal with more expressive models? How can our framework benefit from existing techniques (as [16] surveys)? Does a hidden model always exist for a collection of schemas? We discuss these open issues in Section 7.

In summary, our approach takes a new view to cope with schema matching: (1) We study and report two distinguishing characteristics of databases on the deep Web, an important frontier for information integration. (2) We present a new paradigm, hidden model discovery, for large-scale schema matching, realized by a general statistical framework MGS. (3) We develop Algorithm MGS_{sd} specifically for synonym discovery across Web query interfaces.

We discuss related work in Section 2. Section 3 explores our observations of sources on the deep Web. Section 4 presents the general MGS framework for statistical schema matching, which Section 5 specializes to MGS_{sd} for synonym discovery for Web interfaces. Section 6 reports our case studies and evaluation. Section 7 discusses some open issues.

2. RELATED WORK

Schema matching (which this paper focuses on) is one critical step for schema integration [1, 17]. We relate our work to existing works in four aspects: the paradigms, the techniques, the input data, and the focuses.

First, *paradigms*: Traditionally, schema matching relies on matchings between pairwise attributes before integrating multiple schemas. For instance, traditional binary or n-ary [14] schema integration methodologies (as [1] surveys) exploit pairwise-attribute correspondence assertions (mostly manually given) for merging two or some n sources. The more recent development of general *model management* essentially abstracts schema matching as pairwise similarity mappings between two input sources [16]. In contrast, we propose a new paradigm, statistical schema matching, to holistically match many sources at the same time by discovering their underlying hidden model. Our work was motivated by integrating the deep Web, where the challenge of large scale matching is pressing. Our framework leverages such scale to enable statistical matching.

The closest idea is probably the recent REVERE proposal [9], which suggests to use a separately-built schema corpus as a "knowledge-base" for assisting matching of unseen sources. While sharing the same insights of statistics analysis over corpora, our approach differs in that it leverages input schemas themselves as the corpus and assumes a generative model to unify the corpus.

Second, *techniques*: Based on our observation of the deep Web sources (Section 3), we develop a statistical framework, which contrasts with existing techniques such as machine learning [8], constraint-based [11], and hybrid approaches [13]. The survey [16] presents a taxonomy of these approaches.

Third, *input data*: The previous works assume their input as either relational or structured schemas. Those schemas are designed internally for developers. As a consequence, the attributes of the schemas may be named in a highly inconsistent manner, imposing many difficulties in schema matching. In contrast, our work (Section 5) focuses on matching query interfaces of deep Web sources. These interfaces are designed for end users and are likely more meaningful and consistent. Thus, we observed this distinguishing characteristic of "converging vocabulary" in our deep Web studies (Section 3), which motivated our statistical approach.

Fourth, *focuses*: The previous works focus on different aspects of schema matching. To begin with, for name matching, some focus on simple 1:1 correspondence, while others complex m:n. Further, some works also consider structure matching. This paper targets at synonym discovery to support simple attribute correspondence, which seems sufficient for integrating Web interfaces. However, we believe our general framework can deal with a wider class of schema matching problems (Section 4).

The concept of generative models, as we intend to hypothesize and discover, has been applied in many different contexts. In particular, information retrieval [15] as well as model-based clustering [7] both assume an underlying generative model for a collection of objects. Similar to us, their

domain	*sources*	*all attributes*	*non-rare*
books	55	47	12
movies	52	44	12
music records	49	35	11
automobiles	55	37	11

Figure 2: Statistics of sources studied.

tasks also involve estimating the parameters to construct the right model. However, our search space is significantly larger because we need to select a unified model among multiple model candidates.

3. MOTIVATION: THE DEEP WEB

In the last couple of years, the Web has been rapidly "deepened" with the prevalence of databases online: A significant amount of information is now hidden on the "deep" Web, behind the query forms of *searchable* databases. Such information cannot be accessed directly through static URL links; they are only available as responses to dynamic queries submitted through the query interface of a database.

With massive sources, the deep Web is clearly an important frontier for data integration. For any such attempts (e.g., mediating a specific domain, say, automobiles), it is essential to integrate these query interfaces, since data must be retrieved with queries. This paper focuses on matching the schema aspect of query interfaces by discovering synonym attributes across different sources.

This "wild" frontier of the deep Web is characterized by its unprecedented scale. As a *challenge*: we often need to match large numbers of sources. As an *opportunity*: ample sources are usually available to form a useful "context" of matching. Intuitively, by holistically unifying many sources in the same domain, our statistical approach intends to leverage the opportunity while addressing the challenge.

3.1 Deep Web Observations

To understand their characteristics, we performed informal study of sources on the deep Web. From Web directories, we drew sources in each of the four domains: books, music records, movies, and automobiles. In particular, we collected all of invisibleweb.com's sources (in these 4 domains) and most of yahoo.com's without any bias, until reaching about 50 sources in each domain, as Figure 2 summarizes.

On one hand, we observe *proliferating sources*: As the Web scales, many data sources exist to provide structured information in the same domains, as Figure 2 shows. While many Web directories such as invisibleweb.com already list impressive numbers of online sources by manual compilation, there are certainly much more sources out there: By using overlap analysis, a July 2000 survey [2] estimated 96,000 "search cites" and 550 billion hidden pages in the deep Web. Our survey [4] in December 2002 found 127,000 deep Web sources by exploiting the random IP-sampling approach. As the Web continues to expand, it will house virtually unlimited numbers of sources in interesting domains.

On the other hand, we also observe *converging vocabularies*: The aggregate schema vocabulary of sources in the same domain tends to converge at relatively small size. Figure 2 summarizes (in the middle column) the sizes of the entire vocabularies of all attributes used in any sources, which are about 40 for each domain. Figure 3(a) further analyzes the growth of vocabularies as sources increase in numbers. The curves clearly indicate the convergence of vocabularies. For instance, for book domain, 92% (43/47) attributes are observed at 25^{th} sources, and 98% (46/47) at 35^{th}. Since the vocabulary growth rates (i.e., the slopes of these curves) decrease rapidly, as sources proliferate, their vocabularies will tend to stabilize. Note that the sources are sorted in the same order as they were collected without any bias.

In fact, the vocabularies will converge more rapidly, if we exclude "rare" attributes. To quantify, let the frequency of an attribute be the number of sources in which it occurs. Figure 3(b) orders these frequencies for the book domain over their ranks, with attributes detailed in Figure 8. It is interesting but perhaps not surprising to observe that the distribution obeys the Zipf's law: The frequencies are inversely proportional to their ranks. Many low-ranked attributes thus rarely occur; Figure 3(b) shows only the top 12 attributes (which account for 78% or 230/294 of all the attribute occurrences); most others occur only once. In practice, these rare attributes are likely unimportant in matching since their rareness indicates that very few other sources will find them useful. With such rare attributes (say, below 10% frequencies) excluded, the "useful" vocabularies are much smaller: about 11 attributes per domain (Figure 2).

Note that, while vocabularies tend to converge, schema heterogeneity still persists. That is, although Web query interfaces tend to share attributes, they are not universally unified– thus creating the real challenge of schema matching. In particular, among the top "popular" attributes for books in Figure 3(b)– how many different attributes are "synonyms" for the same concepts? We found 5 ({author, last name, first name}, {subject, category}) out of 12, or a significant 42%. We observed similar levels of heterogeneity in other domains as well (see Figure 8).

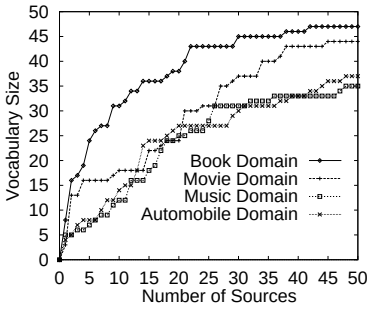
3.2 Toward Hidden Model Discovery

These observations lead us to hypothesize the existence of a hidden schema model that probabilistically generates, from a finite vocabulary, the schemas we observed. Intuitively, such a model gives the "structure" of the vocabulary to constrain how instances can be generated. We believe this hypothesis reasonable, since it naturally explains our observations in Section 3.1.

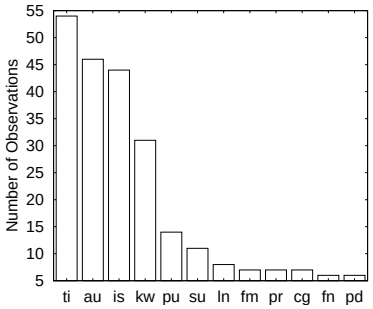
Example 1 (Hidden Model): Referring to Figure 1 (right), the example model structures vocabulary {author, subject, name, $\cdots$} as $\mathcal{M}$ = {(author, writer, name), (subject, category), $\cdots$}. Under $\mathcal{M}$, some schemas can be generated (and thus observed), while others cannot. In particular, schema I_1 = {author, name, title, $\cdots$} is unlikely, since it contains redundant attributes (i.e., author and name) for the same concept, but I_2 = {title, author, subject, $\cdots$} is possible. ■

The hypothesis sheds new light on a different way for coping with schema matching: If a hidden model does exist, its *discovery* would reveal the vocabulary structure, which will in principle answer "any" schema matching questions. (As an analogy, an English dictionary can semantically relate all English words, subsuming the need for their pairwise correspondence.) As Figure 1 contrasts, such model-level unification of all attributes in the same domain will subsume their pairwise correspondence (as used in traditional schema matching). We thus propose a new paradigm: statistical schema matching by hidden model discovery.

4. STATISTICAL SCHEMA MATCHING: THE MGS FRAMEWORK



(a) Vocabulary growth over proliferating sources.



(b) Frequencies over ranks of attributes.

Figure 3: Analyzing schema vocabularies of deep Web sources.

As just motivated, we view schema matching as a quest for an underlying model generating the input schemas. That is, our probabilistic approach seeks to treat the schemas as being generated by a random process following a specific distribution. Our goal is thus, given the input schemas as "observations," to reconstruct the hidden generative distribution. To emphasize the statistic nature, we refer to this general paradigm as *statistical schema matching*.

We believe the statistical approach has several advantages over traditional schema matching: First, *scalability*: By unifying large number of input schemas holistically rather than matching attributes pairwise, it addresses the scale of matching required in the new frontier of networked databases, such as our motivating goal of the deep Web.

Second, *solvability*: In fact, the large scale can itself be a crucial leverage to make schema matching more solvable– in particular, it enables a principled and effective statistical approach, by taking advantages of sufficient samples. Intuitively, we are building upon the "peer context" among schemas. Our approach showed good accuracy with relatively small number of sources (Section 6). Being statistics-based, our approach will benefit from the scale: the accuracy will "scale" with the number of sources.

Third, *generality*: We believe such hidden model discovery can generally deal with many different aspects of schema matching, and present an abstract framework that can be specialized. Except synonym discovery (which we will focus on), there are certainly other important questions related to schema matching, such as: *What concepts are popular?* (Such popularity can guide the design of a "mediator view" over sources, by exporting commonly supported concepts). *How are attributes frequently associated?* (An interactive mediator query interface may, following author, prompt users with title, or following make with model.)

To realize this statistical approach, we propose a general framework, MGS, consisting of hypothesis modeling, generation, and selection. We believe the MGS framework is important in its own rights: In principle, by application-specific hypothesis modeling, MGS can be applied to find models addressing different "*target questions*." We next present the abstract MGS framework by explaining its three essential steps below. Section 5 will develop MGS_{sd} as a concrete specialization for finding synonyms.

1.*Hypotheses Modeling*: To guide the seeking of a hypothetical model, or a *hypothesis*, we start by defining the general structure of such models. Such modeling should essentially capture specific target questions. For instance, if finding synonyms is the target, a model should explicitly express the grouping of "synonyms." Such modeling will also specify a generative behavior of how schemas can be generated. Such behavior is mainly *probabilistic* (e.g., attributes will be drawn randomly by their "popularity"), although it can also partially be *deterministic* (e.g., no synonyms can be selected together). Effectively, the model forms a statistical distribution, which generates a particular schema with some *instantiation probability*.

2.*Hypotheses Generation*: We then enumerate concrete hypotheses (in the specified abstract model) that are consistent with the observed schemas (with non-zero probabilities). Note that, even with a parameterized structure, there will be a large space of candidate hypotheses to search, for a vocabulary of reasonable size. This generation step helps to focus the search to only those promising hypotheses that are likely to generate the observed schemas.

3.*Hypotheses Selection*: Finally, we select hypotheses that are consistent with the observed schemas with sufficient statistical significance. There are various statistical devices for such hypothesis testing [3]. For instance, we use χ^2 testing in our MGS_{sd} algorithm (Section 5).

In summary, we propose MGS as a general framework for the hidden model discovery problem: Given a set of schemas $\mathcal{I}$ as observations, hypothesize and select the schema models with sufficient statistical consistency as the generative distributions of $\mathcal{I}$. We next specialize the abstract framework for synonym discovery.

5. SYNONYM ATTRIBUTES DISCOVERY

Finding corresponding attributes is a central problem for schema matching; in this paper, we pursue this problem as *synonym discovery*. The challenge is to find, typically without semantics understanding, the synonyms among the input attributes. That is, across different schemas, some attributes (e.g., author and name, or subject and category) are *synonyms* for the same *concepts* (e.g., for the "author" and "subject" concepts respectively). As Section 3 motivated, we focus on matching query interfaces for sources in the same domain on the deep Web. Thus, given such schemas, our goal is to discover all the synonym attributes.

Guided by the general MGS framework, we develop Algorithm MGS_{sd} (Figure 6), specifically for synonym attributes discovery as the target question. MGS_{sd} first defines the hypothetical model structure for capturing synonym attributes (Section 5.1), generates the model candidates with non-zero probabilities (Section 5.2), and selects the sufficiently consistent ones (Section 5.3). Beyond these essential steps, we develop techniques for coping with several real-world issues that complicate our statistical approach (Section 5.4). Finally, we put all the components together to present the complete algorithm (Section 5.5).

5.1 Hypothesis Modeling

Following MGS, we first define the structure of the underlying model. Specifically, we aim at answering the target question of synonym attributes for Web interfaces. (Incidentally, our model can also capture the target question of concept popularity.) We view a query interface as a "flat" schema, or a set of attributes; e.g., amazon.com has a schema {title, author, $\cdots$}. This simple view is sufficient for our purpose of synonym discovery. In particular, we do not concern complex matching (e.g., author as last and first name), which itself is another interesting target question.

To reasonably define a simple model, we make several assumptions of how our schemas are generated. (Imagine a human Web developer generates such Web interfaces to bring databases online.) First, *concept mutual-independence*: A query interface contains several different concepts (e.g., "author" or "subject"). We assume that, in generating a schema (which may not contain all concepts), different concepts are selected independently.

Second, *synonym mutual-exclusion*: When multiple synonyms exist for a concept (e.g., author and name), we assume that, in generating a schema, no two synonyms will both be selected. Such duplicated selections will create redundancy and perhaps confusion; our case studies (of real sources; Section 3.1) in fact have found no such schemas. As Section 5.2 will discuss, this mutual exclusion enables significant pruning of the hypothetical model space.

Third, *non-overlapping concepts*: We assume that different concepts do not overlap in semantics, i.e., no distinct concepts will share attributes. This assumption holds in most cases, when synonyms in the same concept are fully *equivalent*: e.g., concepts {author, name} and {subject, category} do not overlap. Thus this assumption says that all concepts will form an *equivalence partition* of the vocabulary set. However, as our case studies observed (Section 6), sometimes an attribute can be a *non-equivalent* synonym to others, and thus participate in distinct concepts– e.g., concepts {author, last name} and {author, first name}, where author corresponds to last name and first name in different "senses." This assumption excludes such cases: Instead of complicating simple synonym equivalence, such cases can be more systematically treated, by first grouping attributes [last name, first name] (such grouping can itself be another target question; see Section 4) and then finding equivalent synonym {author, [last name, first name]} (see Section 6).

5.1.1 Model Structure

Based on our assumptions, we define a simple model for capturing synonym attributes. Essentially, a model describes how to generate schemas from a vocabulary of attributes. Figure 4 visualizes $\mathcal{M}_B$ (an example for book sources) as a two-level tree, for vocabulary $\mathcal{V}_B$ = {author, title, ISBN, subject, category}. To express synonyms, our model partitions all attributes into concepts, or equivalent classes (by the non-overlapping concepts assumption): e.g., C_1: {author}, $\cdots$, C_4: {subject, category} in $\mathcal{M}_B$. The model will generate schemas by, first, independently selecting each concept C_i with probability α_i (by concept mutual-independence). For any selected concept, the model will then choose exactly one of its member attributes A_j with probability β_j (by synonym mutual-exclusion). The model thus generates a schema with all the chosen attributes.

Definition 1: A *schema model* $\mathcal{M}$ is a 4-tuple $(\mathcal{V}, \mathcal{C}, P_c, P_a)$:

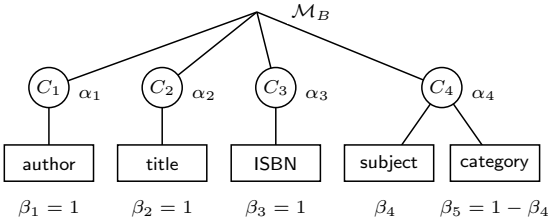


Figure 4: An example of schema model $\mathcal{M}_B$.

The *vocabulary* $\mathcal{V}$ is a set of attributes $\{A_1, \cdots, A_n\}$. The *concept partition* $\mathcal{C}$ is a set of *concepts* $\{C_1, \cdots, C_m\}$ that partition $\mathcal{V}$ (i.e., $\mathcal{V} = \bigcup C_i$ and $C_i \cap C_k = \emptyset$). P_c is the *concept probability function*, which determines the probability α_i for including concept C_i in schema generation. P_a is the *attribute probability function*, which determines the probability β_j for selecting attribute A_j, once its concept is included. For every concept C_i: $\sum_{A_j \in C_i} \beta_j = 1$. ■

Notationally, we will write a model by parenthesizing attributes in concepts with probability annotations, e.g.: (Figure 4) $\mathcal{M}_B$ = {(author: β_1): α_1, (title: β_2): α_2, (ISBN: β_3): α_3, (subject: β_4, category: β_5): α_4}. When probabilities are not critical in the context, we will simply write $\mathcal{M}_B$ = {(author), (title), (ISBN), (subject, category)}).

5.1.2 Schema Generation and Observations

We now discuss how a model $\mathcal{M}$ will generate schemas. By Definition 1, $\mathcal{M}$ will simply decide, for each concept C_i, if C_i is included, and if so, select one attribute A_j to represent C_i. This process will generate a schema as a set of attributes.

Example 2: For $\mathcal{M}_B$ in Figure 4: Possible schemas (with non-zero probabilities) from $\mathcal{M}_B$ include: I_1 = {author, title, subject, ISBN} and I_2 = {title, category, ISBN}. ■

Note that a model $\mathcal{M}$, by definition, represents a generative distribution, giving probabilities for any schema that can be generated. We now formalize such probabilities. First, to generate a schema, $\mathcal{M}$ selects concepts to include: By Definition 1, a concept C_i will appear with probability $Pr(C_i|\mathcal{M}) = \alpha_i$ or otherwise $Pr(\neg C_i|\mathcal{M}) = 1 - \alpha_i$.

Next, we consider the probability of picking some attribute: By Definition 1, the probability of selecting an individual attribute A_j in schema generation from $\mathcal{M}$ is:

$$Pr(A_j|\mathcal{M}) = \begin{cases} \alpha_i \times \beta_j, & \exists i : A_j \in C_i \\ 0, & otherwise \end{cases}$$

How about selecting a set of attributes A_1, A_2,.., A_m from $\mathcal{M}$ in any schema? Definition 1 implies this probability as below, where the first condition represents synonym mutual-exclusion and the other concept mutual-independence.

$$Pr(A_1, A_2, .., A_m|\mathcal{M}) = \begin{cases} 0, \exists j \neq k, \exists i : A_j \in C_i \wedge A_k \in C_i \\ \prod Pr(A_j|\mathcal{M}), otherwise \end{cases}$$

Putting together, we can derive the probability that $\mathcal{M}$ will generate some schema I, denoted by $Pr(I|\mathcal{M})$. Definition 2 below formalizes this *instantiation probability*. Specifically, $Pr(I|\mathcal{M})$ is the probability of used attributes times the probability of unselected concepts.

Definition 2: For model $\mathcal{M} = (\mathcal{V}, \mathcal{C}, P_c, P_a)$, the *instantiation probability* of a schema $I = \{A_1,...,A_m\}$ is $Pr(I|\mathcal{M}) = Pr(A_1, A_2, .., A_m|\mathcal{M}) \times \prod_{\forall A_j, A_j \notin C_i} Pr(\neg C_i|\mathcal{M})$. We say I can be *instantiated* from model $\mathcal{M}$ if $Pr(I|\mathcal{M}) > 0$. ■

Example 3: Continuing Example 2: we have $Pr(I_1|\mathcal{M}_B) = \alpha_1 \times \alpha_2 \times \alpha_3 \times \alpha_4 \times \beta_4$, $Pr(I_2|\mathcal{M}_B) = (1-\alpha_1) \times \alpha_2 \times \alpha_3 \times \alpha_4 \times \beta_5$, where $(1-\alpha_1)$ is the probability that the concept C_1 is not used. However, for $I_3 = \{$author, ISBN, subject, category$\}$, we have $Pr(I_3|\mathcal{M}_B) = 0$, since subject and category both belong to C_4. Thus I_1 and I_2 can be instantiated from $\mathcal{M}_B$, but I_3 cannot. ∎

Our approach seeks to discover the hidden model from many schemas observed (as input). Therefore, we will take a set of schemas $\mathcal{I}$ (e.g., the Web sources summarized in Figure 2), our input, as *schema observations*. To emphasize that in our input we may observe the same schema several times, we write $\mathcal{I}$ as a set of 2-tuple $\langle I_i, B_i \rangle$. Each $\langle I_i, B_i \rangle$ denotes the number of occurrences B_i for each schema I_i.

To discover the hidden model, it is essential to answer: Given model $\mathcal{M}$, how likely will $\mathcal{M}$ generate the schemas in $\mathcal{I}$? (Or, how likely can we observe $\mathcal{I}$, if $\mathcal{M}$ is the hidden model?) It follows Definition 2 that this probability is $Pr(\mathcal{I}|\mathcal{M}) = \prod Pr(I_i|\mathcal{M})^{B_i}$. Note that, if $Pr(\mathcal{I}|\mathcal{M}) = 0$, it is impossible to observe $\mathcal{I}$ under $\mathcal{M}$. Therefore, we say model $\mathcal{M}$ is *consistent* with observations $\mathcal{I}$, if $Pr(\mathcal{I}|\mathcal{M}) > 0$. Thus, the hypothesis generation finds these "consistent models" as candidate hidden models (Section 5.2).

Example 4: Continuing Example 3: We may have observations $\mathcal{I} = \{\langle I_1, 3\rangle, \langle I_2, 5\rangle\}$, i.e., I_1 3 times and I_2 5 times. Thus, $Pr(\mathcal{I}|\mathcal{M}_B) = Pr(I_1|\mathcal{M}_B)^3 \times Pr(I_2|\mathcal{M}_B)^5$. Note $\mathcal{M}_B$ is consistent with $\mathcal{I}$, since $Pr(I_1|\mathcal{M}_B)$ and $Pr(I_2|\mathcal{M}_B)$ are both non-zero (Example 3). ∎

5.2 Hypothesis Generation

Guided by the second step of the MGS framework, we now generate candidate models that are likely to be sufficiently consistent (which Section 5.3 will determine) with the input observations $\mathcal{I}$. It is clear that any candidate $\mathcal{M}$ has to be at least consistent with $\mathcal{I}$, i.e., $Pr(\mathcal{I}|\mathcal{M}) > 0$, so that $\mathcal{I}$ is at least possible under $\mathcal{M}$ (Section 5.1). This section focuses on constructing such models.

Intuitively, we want to reconstruct $\mathcal{M}$ from our given observations $\mathcal{I}$. Using a statistical approach, we assume the observations are *unbiased* and *sufficient*. First, by the *unbiased* assumption, we will observe (or collect) a schema I with a frequency in $\mathcal{I}$ proportional to how likely I will be generated under $\mathcal{M}$, i.e., $Pr(I|\mathcal{M})$. (Say, we will not collect *only* schemas that contain author – that would be biased.) Second, by the *sufficient* assumption, our observations will be large enough, so that every possible schema is represented in $\mathcal{I}$. We use these assumptions to estimate the probability parameters (P_a and P_c) of a candidate model. In practice, the sufficient assumption is likely not to be satisfied; we discuss techniques for dealing with "the real world" in Section 5.4.

Our goal in hypothesis generation is, given $\mathcal{I}$, to construct models $\mathcal{M} = (\mathcal{V}, \mathcal{C}, P_c, P_a)$ so that $Pr(\mathcal{I}|\mathcal{M}) > 0$. To begin with, we determine $\mathcal{V}$: By our above assumptions, $\mathcal{V} = \bigcup I_i$, since every possible schema occurs in $\mathcal{I}$, and so does every attribute in $\mathcal{V}$. On the other hand, even if the observations are not perfect, for our purpose of matching, we do not care any "phantom" attributes that have not occurred in any input source. Thus, our model will capture only attributes that are used by at least one schema (in $\mathcal{I}$).

Next, having determined $\mathcal{V}$, we complete the model ($\mathcal{V}$, $\mathcal{C}$, P_c, P_a) by constructing first the concept partition $\mathcal{C}$ (Section 5.2.1), and then the probabilities P_c, P_a (Section 5.2.2).

5.2.1 Building Concept Partitions

Given the vocabulary set $\mathcal{V}$, we first construct a concept partition $\mathcal{C}$ for a candidate model. By Definition 1, $\mathcal{C}$ is a partition of $\mathcal{V}$. It is clear that, given $\mathcal{V}$, there can easily be a large number of possible partitions. The number of partitions for an n-set is called a Bell number $B(n)$, which has an exponential generating function and satisfies recursive relation $B(n+1) = \sum_{k=0}^{n} B(k)\binom{n}{k}$. A vocabulary with, say, 12 attributes will thus have 4213597 possible concept partitions (and as many possible models).

To cope with the large space, it is thus critical to focus on only concept partitions that can lead to consistent models ($\mathcal{M}$, such that $Pr(\mathcal{I}|\mathcal{M}) > 0$). These consistent models form the *hypothesis space* with respect to $\mathcal{I}$. Our case studies show that the "consistent" condition can prune the search space to a very small number of models. For instance, in the book domain, we only have 20 models left in the hypothesis space with 12 attributes. To construct the hypothesis space, a naive approach that constructs and test every hypothesis will not work, due to the large number of possible concept partitions (as just discussed).

However, not all concept partitions are useful for constructing a consistent model: It is important to note that, not every model (with arbitrary concept partitions) can generate a schema observed. In particular, as Example 3 showed, I_3 cannot be observed under $\mathcal{M}_B$, or $Pr(I_3|\mathcal{M}_B) = 0$, since subject and category are both synonyms in C_4 (Figure 4). Thus, if I_3 is in $\mathcal{I}$ as part of our input schema, we will not consider $\mathcal{M}_B$, since it will be inconsistent with $\mathcal{I}$, i.e., $Pr(\mathcal{I}|\mathcal{M}_B) = 0$ as $Pr(I_3|\mathcal{M}_B) = 0$.

We can easily generalize this idea to focus on models that will not "contradict" with any observed schema I. In such models, none of the concepts will contain attributes A_j and A_k that are used by I. (In Example 3, $\mathcal{M}_B$ is not good for I_3, since $\mathcal{M}_B$ contains C_4 with attributes subject and category both from I_3.) That is, we will construct consistent models by using only *consistent concepts*, which do not contain any *co-occurring attributes* from any schema in $\mathcal{I}$. Property 1 formalizes this idea.

Property 1: Given observations $\mathcal{I}$ with vocabulary $\mathcal{V}$, let $\mathcal{C} = \{C_1, \cdots, C_m\}$ be a concept partition for vocabulary $\mathcal{V}$. Any model $\mathcal{M}$ constructed from $\mathcal{C}$ will be inconsistent with $\mathcal{I}$, or $Pr(\mathcal{I}|\mathcal{M}) = 0$, if for some attributes A_j and A_k, both of the following hold:

1. $\exists$ schema $I \in \mathcal{I}$, such that $A_j \in I$ and $A_k \in I$.
2. $\exists$ concept $C_i \in \mathcal{C}$, such that $A_j \in C_i$ and $A_k \in C_i$. ∎

Based on Property 1, we use a two-step process to build the hypothesis space (of consistent models) using consistent concepts as building blocks. (For instance, $\mathcal{M}_B$ in Figure 4 is built upon concepts $C_1, \cdots, C_4$.) Step 1, ConsistentConceptsConstruction, will first find all consistent concept, and Step 2, buildHypothesisSpace, will then build consistent models accordingly. These two procedures are used to build the initial hypothesis space in Algorithm MGS_{sd}.

In Step 1, we can translate the problem of finding consistent concepts into finding all cliques in an attribute "co-occurrence graph" [6]. Specifically, we construct a *concept network* from our observations $\mathcal{I}$: In this graph, a node represents an attribute, and an edge exists between attributes A_j and A_k if and only if they do not co-occur in any schema I in $\mathcal{I}$. Thus, non-cooccurring attributes will be connected

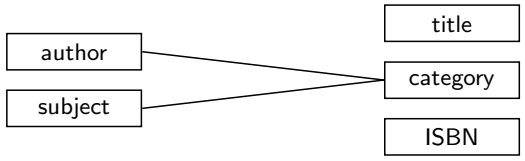


Figure 5: An example concept network.

with an edge– Precisely such attributes will form consistent concepts. However, a concept can be of any number of attributes. Therefore, we look for cliques for any size in the graph to construct consistent concepts.

Example 5: Consider observations $\mathcal{I}$ in Example 4. From $\mathcal{I}$, we can derive its concept network in Figure 5. In particular, author and title do not have an edge because they co-occur in I_1. Author and category have an edge since they do not co-occur in any schema.

Further, what can be consistent concepts? There are 7 cliques in Figure 5: {author}, {title}, {subject}, {category}, {ISBN}, {author, category}, and {subject, category}. Any clique represents a cluster of non-cooccurring attributes, and therefore is a consistent concept (by Property 1). In particular, some of these concepts, such as {author} and {subject, category}, are part of $\mathcal{M}_B$, which is consistent with $\mathcal{I}$ (as Example 4 explained). ∎

In Step 2, we use the consistent concepts just obtained as the building blocks for constructing consistent models. Since all the concepts in a model partition its vocabulary set $\mathcal{V}$, this step is essentially a classic set cover problem [6], with the covering subsets being non-overlapping. That is, given some subsets (the consistent concepts) of set $\mathcal{V}$, we want to select some non-overlapping subsets to cover $\mathcal{V}$. Below we illustrate the result of constructing all the consistent models as the hypothesis space, which concludes our hypothesis generation step in this section.

Example 6: Given the consistent concepts in Example 5, we can construct a consistent model $\mathcal{M}_1 = \{$(author), (title), (ISBN), (subject), (category)$\}$, since the five concepts partition the vocabulary. We can find all the other consistent models: $\mathcal{M}_2 = \{$(author), (title), (ISBN), (subject, category)$\}$ and $\mathcal{M}_3 = \{$(author, category), (title), (ISBN), (subject)$\}$. The hypothesis space is therefore $\{\mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3\}$. ∎

5.2.2 Building Probability Functions

We have generated all the consistent models, which form the hypothesis space. However, these models are still incomplete: As a 4-tuple $(\mathcal{V}, \mathcal{C}, P_c, P_a)$, $\mathcal{M}$ has yet to determine the probability functions P_c and P_a, although $\mathcal{V}$ and $\mathcal{C}$ are specified. Recall our ultimate goal is to discover those hidden models that are sufficiently consistent with input $\mathcal{I}$. So far, for each consistent model $\mathcal{M}$, by building upon only consistent concepts, we guarantee that $Pr(\mathcal{I}|\mathcal{M})$ is not *necessarily* zero. Therefore, there exist P_c and P_a assignments for $\mathcal{M}$, such that $Pr(\mathcal{I}|\mathcal{M}) > 0$.

To complete each of these consistent models, we still need to specify P_c and P_a– clearly these probabilities should further maximize $Pr(\mathcal{I}|\mathcal{M})$. The reason is that with the assumptions of unbiased and sufficient input data, the values of P_c and P_a must be the ones that make the model the most consistent with the data. The "consistency" is reflected as the instantiation probability. So the most consistent model is corresponding to the model with the highest probability. Thus, we have an optimization problem to find

$$\max_{P_c, P_a} Pr(\mathcal{I}|\mathcal{M}(\mathcal{V}, \mathcal{C}, P_c, P_a)), \quad (1)$$

which is essentially the *maximum likelihood estimation* problem, for given $\mathcal{V}$ and $\mathcal{C}$.

Example 7: Continue Example 6, where we showed $\mathcal{M}_2$ as one of the consistent models. To completely specify $\mathcal{M}_2$, we need to determine P_c and P_a to maximize $Pr(\mathcal{I}|\mathcal{M}_2)$ (for $\mathcal{I}$ given in Example 4).

As Example 4 derives (note $\mathcal{M}_2$ and $\mathcal{M}_B$ are the same model): $Pr(\mathcal{I}|\mathcal{M}_2) = Pr(I_1|\mathcal{M}_2)^3 \times Pr(I_2|\mathcal{M}_2)^5 = \alpha_1^3 \times (1-\alpha_1)^5 \times \alpha_2^8 \times \alpha_3^8 \times \alpha_4^8 \times \beta_4^3 \times \beta_5^5$. We apply maximum likelihood estimation to select those α's and β's that can maximize $Pr(\mathcal{I}|\mathcal{M}_2)$. The result is $\alpha_1 = 0.375, \alpha_2 = 1, \alpha_3 = 1, \alpha_4 = 1, \beta_4 = 0.375$, and $\beta_5 = 0.625$. ∎

In maximum likelihood estimation of functions P_c and P_a, we are effectively estimating parameters α_i and β_j (Definition 1). Since concepts are independently selected (the concept mutual independence assumption of Section 5.1), each α_i can be estimated independently. We can also derive the solution for β_j based on [3], since β_j in a concept C_i form a multinomial distribution. Therefore, for any schema model, Equation 1 has the closed-form solutions:

$$\alpha_i^* = \frac{\sum_{A_j \in C_i} O_j}{|\mathcal{I}|}, \ \beta_j^* = \frac{O_j}{\sum_{A_j \in C_i} O_j}$$

where O_j is the *frequency* of attribute A_j in observations $\mathcal{I}$ (i.e., the number of schemas that contain A_j), and $|\mathcal{I}|$ is the total number of schemas in $\mathcal{I}$.

5.3 Hypothesis Selection

Guided by the third step of the MGS framework, we need to select sufficiently consistent hypotheses. After hypothesis generation, a hypothesis is a determined model (distribution) $\mathcal{M} = (\mathcal{V}, \mathcal{C}, P_c, P_a)$. We propose to apply χ^2 hypothesis testing to quantify how consistent the schema model is with the data. Below we briefly introduce χ^2 testing [3].

Suppose we have n independent observations (schemas) and in each observation, precisely one of r events (schemas with non-zero probability), $I_1, .., I_r$ must happen, and their respective probabilities are $p_1, .., p_r$, with $\sum_{j=1}^r p_j = 1$. Suppose that $p_{10}, .., p_{r0}$ are the respective instantiation probabilities of the observed $I_1, .., I_r$ with respect to the tested model $\mathcal{M}$, with $\sum_{j=1}^r p_{j0} = 1$. We want to test the hypothesis $p_1 = p_{10}, .., p_r = p_{r0}$ by considering the statistic

$$D^2 = \sum_{j=1}^r \frac{(B_j - np_{j0})^2}{np_{j0}}$$

where n is essentially $|\mathcal{I}|$. It can be shown that D^2 has asymptotically a χ^2 distribution with $r-1$ degrees of freedom. Again a test of the null hypothesis $H : p_1 = p_{10}, .., p_r = p_{r0}$ at the $100a\%$ significance level is obtained by choosing a number b such that $Pr\{D^2 > b\} = a$, where D^2 has the χ^2 distribution with $r-1$ degrees, and rejecting the hypothesis if a value of D^2 greater than b is actually observed.

Example 8: Assume we have observations $\mathcal{I} = \{\langle I_1, 6\rangle, \langle I_2, 3\rangle, \langle I_3, 1\rangle\}$, with $I_1 = \{$author, subject$\}$, $I_2 = \{$author, category$\}$, and $I_3 = \{$subject$\}$. Our goal is to select the schema model at the significance level 0.05. The hypothesis generation step will output two hypotheses(models):

$\mathcal{M}_1 = \{$(author:1):0.6,(subject:0.7,category:0.3):1$\}$ and

$\mathcal{M}_2 = \{$(author:1):0.6,(subject:1):0.7,(category:1):0.3$\}$.

We first consider $\mathcal{M}_1$. Four schemas can be instantiated from $\mathcal{M}_1$: {author}, {category}, {author, subject}, and {author, category} with instantiation probabilities 0.28, 0.12, 0.42, and 0.18 respectively. Thus, the computation of D^2 is: $D^2(\mathcal{M}_1) = \frac{(1-10*0.28)^2}{10*0.28} + \frac{(0-10*0.12)^2}{10*0.12} + \frac{(6-10*0.42)^2}{10*0.42} + \frac{(3-10*0.18)^2}{10*0.18} \doteq 3.93$ with freedom degree 3. The χ^2 distribution table shows $Pr(D^2 > 7.815) = 0.05$ at that freedom degree. Since $3.93 < 7.815$, we accept this hypothesis and consider it as a sufficiently consistent schema model.

$\mathcal{M}_2$ is processed in the same way. Eight schemas can be instantiated from $\mathcal{M}_2$: {}, {author}, {subject}, {category}, {author, subject}, {author, category}, {subject, category}, and {author, subject, category} with probabilities 0.084, 0.126, 0.196, 0.036, 0.294, 0.054, 0.084, and 0.126 respectively. Then we have $D^2(\mathcal{M}_2) \doteq 20.20$ with freedom degree 7. The χ^2 distribution table shows $Pr(D^2 > 14.067) = 0.05$. Since $20.20 > 14.067$, we should not select $\mathcal{M}_2$. Therefore, hypothesis selection will select $\mathcal{M}_1$ as the schema model. ■

5.4 Dealing With the Real World

We presented the overall process of Algorithm MGS$_{sd}$, guided by the general principles of the MGS framework. Further, there are often "real-world" issues on data observations that can compromise a statistical approach like ours. We find that, specifically for schema matching, the key challenge is the extremely "unbalanced" attribute distribution– We observed a Zipf-like distribution (Figure 3b) of attributes in our analysis of deep Web sources (Section 3).

Challenges arise on the either end of this Zipf distribution: On one hand, the *head-ranked* attributes (e.g., ti and au in Figure 3b) are extremely frequent, occurring in almost every schema: Their occurrences tend to dominate any models and thus render these models indiscriminate under hypothesis testing (as Section 5.3 developed). Section 5.4.3 addresses dominating attributes with incremental consensus projection to isolate their effects.

On the other hand, the *tail-ranked* attributes (e.g., those not shown in Figure 3b) are extremely rare, often occurring only once in some schema. Their occurrences in observations (while rare) tend to "confuse" our statistical approach that asserts sufficient samples. In principle, a rare attribute A can appear in many concepts (by combining with other attributes in schema generation). Also, as A being rare, these "A-schemas" are unlikely to be observed in $\mathcal{I}$ if it is not arbitrarily large– Thus A will compromise a statistical approach for the lack of schemas. Section 5.4.1 addresses rare attributes with attribute selection.

Together, this *head-often, tail-rare* attribute distribution will imply similar non-uniformness of schemas. Thus, some schemas (with rare attributes) will be extremely rare too. Our hypothesis testing essentially relies on estimating schema frequencies B_j (Section 5.3). A rare schema I occurring only once in $\mathcal{I}$ tends to result in an *overestimated* frequency, or $\mathcal{I}$ needs to be arbitrarily large to justify I's only occurrence being sufficiently rare. Section 5.4.2 addresses rare schemas by "smoothing."

5.4.1 Attribute Selection

Rare attributes can confuse a statistical approach, with their lack of complete schemas in our observations $\mathcal{I}$. Such rare attributes will require virtually arbitrarily large $\mathcal{I}$ to give them sufficient context. That is, for these rare attributes, $\mathcal{I}$ is unlikely to be sufficient to statistically "explain" their properties– Thus, our sufficient assumption (Section 5.2) is unlikely to hold for such attributes. To draw valid statistical results, our approach is to systematically remove rare attributes– they are effectively "noises" in our setting.

Fortunately, these rare attributes may indeed be unimportant in schema matching. As Section 3.1 explained, with Zipf distribution, most rare attributes occur in only one source. Thus, few other sources will find these attributes useful in query mediation or data exchange. (A mediator will not be likely to support such attributes; they are neither "mediatable" nor "exchangeable.") We believe it is naturally justified to remove rare noises in matching.

We believe systematic *attribute selection* will be crucial for finding attribute subsets, for which robust statistical results can be achieved. We use a frequency-based pruning to select only frequent attributes into vocabulary $\mathcal{V}$ (Section 5.2), as a procedure ATTRIBUTESELECTION in Algorithm MGS$_{sd}$ (Figure 6). Specifically, we select an attribute A_j if its observation frequency $O_j \geq f$, where f is a given threshold set as 10% in our experiments. While this empirical value works well (Section 6), further investigation is clearly necessary to automate threshold selection.

5.4.2 Rare Schema Smoothing

Our observations $\mathcal{I}$ may contain infrequent schemas I that are presumably rare, as explained earlier. In particular, the χ^2 testing (Section 5.3) evaluates the difference between the estimated probabilities $Pr(I|\mathcal{M})$ and the observed frequencies B_j. For infrequent schemas, such difference will significantly distort the closeness of D^2 to the χ^2 distribution, which may influence the result of hypothesis selection.

Example 9: Suppose our observations $\mathcal{I} = \{\langle I_1, 45\rangle, \langle I_2, 5\rangle, \langle I_3, 2\rangle, \langle I_4, 1\rangle\}$, with I_1 = {author}, I_2 = {last name}, I_3 = {author, price}, and I_4 = {price}. The hypothesis generation will find three hypotheses:

$\mathcal{M}_1$={(author:.9,last name:.1):.98,(price:1):.06}

$\mathcal{M}_2$={(author:1):.89,(last name:.62,price:.38):.15}

$\mathcal{M}_3$={(author:1):.89,(last name:1):.09,(price:1):.06}.

The probabilities of I_4 in $\mathcal{M}_1$, $\mathcal{M}_2$, and $\mathcal{M}_3$ are .0012, .0064, and .0058 respectively, which indicates I_4 a rare schema. The χ^2 testing will in fact reject all three models (at the significance level 0.05).

Note that even the correct model $\mathcal{M}_1$ does not pass the test, simply because the early observation of the rare schema I_4 results in an unreliable estimation of its probability. Thus the rare schema disturbs the result. ■

We cope with this problem by *rare schema smoothing*: Instead of regarding each possible schema I_j as an individual event (Section 5.3), we will aggregate infrequent schemas into a conceptual event I_{rare}, whose probability is the sum of the probabilities of its members. Such aggregation will smooth the overestimation in frequency counting, thus giving more reliable probability indication [12]. We will then take χ^2 testing on those frequent events plus I_{rare}.

The key issue is then how to determine whether a schema I_j is rare. Our basis is its frequency in observations $\mathcal{I}$ (with size $|\mathcal{I}|$), since the real probability is hidden to be discovered. We apply two criteria: 1) If not observed in $\mathcal{I}$, I_j is rare. 2) If observed, I_j is rare if $Pr(I_j|\mathcal{M}) \times |\mathcal{I}| < T_{smooth}$, where T_{smooth} is a threshold (dynamically determined).

We further develop adaptive thresholding of T_{smooth} in smoothing, as a procedure DYNAMICSELECTION in Algorithm MGS$_{sd}$ (Figure 6): During hypothesis selection (Section 5.3),

we test the hypotheses with increasing thresholds until reaching at least one qualified hypothesis. (Implicitly, we are applying our motivating assertion that there must exist a correct hidden model.) Otherwise, it will output all the hypotheses, since they are not distinguishable (and at least one must be correct). Empirically, we start the adaptive thresholding at $T_{smooth} = 0.2$ with a step size 0.1, and stop at 1.0, which works well (Section 6).

5.4.3 Consensus Projection

Straightforward testing cannot always distinguish models that share a dominating "consensus" (which makes other differences insignificant). As explained earlier, the head-ranked attributes often dominate the testing and thus all these models may agree on the "structure" of these attributes– Such *consensus* can be recognized (for early conclusion) and projected (for isolating dominating attributes). Note that we assume a consensus must be correct, based on our motivating assertion that there exists at least a correct model.

Example 10: Suppose our observations $\mathcal{I} = \{\langle I_1, 45\rangle, \langle I_2, 6\rangle, \langle I_3, 2\rangle, \langle I_4, 4\rangle\}$ with $I_1 = \{$title$\}$, $I_2 = \{$title, subject$\}$, $I_3 = \{$title, subject, price$\}$, and $I_4 = \{$title, category$\}$. Hypothesis generation will output three hypotheses:

$\mathcal{M}_1$={(title:1):1,(subject:.67,category:.33):.21,(price:1):.035}
$\mathcal{M}_2$={(title:1):1,(subject:1):.14,(category:.67,price:.33):.11}
$\mathcal{M}_3$={(title:1):1,(subject:1):.14,(category:1):.07,(price:1):.035}.

The χ^2 hypothesis testing will reject all the three models at the significance level 0.05. In fact, their D^2 values are not distinguishable– due to the highly frequent attribute title, which dominates the χ^2 testing. However, it is clear that they all share a "consensus" on title. ■

We thus propose *consensus projection* for recognizing and extracting consensuses (or shared concepts across models), so that hypothesis testing will better focus on models' distinctions. Note that, the soundness of such projection (of consensus concepts) follows our concept mutual independence assumption (Section 5.1).

Specifically, consensus projection will extract the consensus from all the models in the hypothesis space. Also it will extract the consensus attributes from the observed schemas and aggregate the projected schemas that become identical. The projection and aggregation will result in a new set of input schemas, which are used for the re-estimation of the parameters of the projected models. Such projection can be repeated, since more consensuses will gradually emerge as the algorithm progresses. We can then discover the final models incrementally by projecting consensuses in progressive iterations. We thus structure Algorithm MGS$_{sd}$ as an iterative framework, as Section 5.5 will discuss.

Example 11: Continuing Example 10: We recognize concept (title) as the consensus. We thus perform consensus projection to extract (title) from all hypotheses and attribute title from all schemas in $\mathcal{I}$.

So we have $\mathcal{H}^* = \pi_{\{\text{subject,category,price}\}}(\mathcal{H})$, with
$\mathcal{M}_1^*$={(subject:.67,category:.33):1,(price:1):.17}
$\mathcal{M}_2^*$={(subject:1):.67,(category:.67,price:.33):.5}
$\mathcal{M}_3^*$={(subject:1):.67,(category:1):.33,(price:1):.17}
and $\mathcal{I}^* = \pi_{\{\text{subject,category,price}\}}(\mathcal{I}) = \{\langle I_2^*, 6\rangle, \langle I_3^*, 2\rangle, \langle I_4^*, 4\rangle\}$ with $I_2^* = \{$subject$\}$, $I_3^* = \{$subject, price$\}$, and $I_4^* = \{$category$\}$. I_1^* is empty after projection and thus removed. The new parameters of $\mathcal{M}^*$ are estimated from $\mathcal{I}^*$ with maximum likelihood estimation. The χ^2 testing will select $\mathcal{M}_1^*$ (and reject others) at the significance level 0.05. ■

5.5 Putting It All Together: Algorithm MGS$_{sd}$

For solving the target question of synonym attributes, Algorithm MGS$_{sd}$ (Figure 6) consists of two phases: building initial hypothesis space and iteratively discovering the hidden models. The first phase selects the attributes as the vocabulary (Section 5.4.1) and builds the hypothesis space (Section 5.2.1). The iterative process is based on consensus projection (Section 5.4.3): In each iteration, it projects the consensus, re-estimates the parameters (Section 5.2.2), and tests the hypotheses (Section 5.3) with the smoothing technique (Section 5.4.2).

Example 12: Consider the book domain sources listed in Figure 8, the iterative process is illustrated in Figure 7. In the first iteration, the consensus consists of concepts (ti), (is), (kw), (pr), (fm), and (pd). The DYNAMICSELECTION function will select four hypotheses as *selectedH* with T_{smooth} as 0.5, which are listed in the third column of the 1st iteration of Figure 7. In the second iteration, the consensus consists of concept (pu). The DYNAMICSELECTION function will select two hypotheses among the four in the 1st iteration. In the third iteration, the consensus is (su, cg) and DYNAMICSELECTION cannot find any passing hypothesis with all the T_{smooth}'s. Therefore, the algorithm will stop and output two discovered schema models: $\mathcal{M}_1$ = {(ti), (is), (kw), (pr), (fm), (pd), (pu), (su, cg), (au, ln), (fn)} and $\mathcal{M}_2$ = {(ti), (is), (kw), (pr), (fm), (pd), (pu), (su, cg), (au, fn), (ln)}, where the parameters α's and β's are omitted. ■

The time complexity of MGS$_{sd}$ is exponential with respect to the number of attributes. For instance, the complexity of CONSISTENTCONCEPTSCONSTRUCTION is exponential since the clique problem is NP-complete. Similarly, the steps of BUILDHYPOTHESISSPACE and DYNAMICSELECTION are both exponential. Since schema matching is typically done "offline," such computation time may still be tolerable in most situations. For instance, in our experimental setting (Section 6), the running time is typically within one minute. Further, our observation in Section 3 indicates that in practice the computation is likely to scale to many sources: Even with more sources, their aggregate vocabulary tends to converge– The growth of attributes and thus the corresponding computation cost are likely to stop at some point. Nevertheless, it is certainly a real issue to explore more efficient algorithms, as Section 7 discusses.

6. CASE STUDIES

To evaluate the MGS$_{sd}$ framework, we test it with four domains of sources on the deep Web. We design two suites of metrics to quantify the accuracy of both the model itself and its ability to answer the target questions. The experimental results show remarkable accuracy for both metrics.

6.1 Experiment Setup

We collect over 200 sources over four domains as stated in Section 3.1. For each source, we manually extracted attributes from its query interface and did some straightforward preprocessing to merge attributes of slight textual variations (e.g., author's name and author). Thus, we focus on discovering synonym attributes and consider such attribute extraction and preprocessing as independent tasks. In particular, attribute extraction (e.g., extracting "author's name" from "please input author's name") can be automated with noun-phrase extraction tools, such as LinkIT

kth	*consensus*	*hypotheses pass kth iteration*	T_{smooth}
1st	(ti),(is),(kw), (pr),(fm),(pd)	{(au:0.85,ln:0.15):0.98,(pu:1):0.25,(su:1):0.2,(cg:1):0.13,(fn:1):0.11}	0.5
		{(au:0.85,ln:0.15):0.98,(pu:1):0.25,(su:0.61,cg:0.39):0.33,(fn:1):0.11}	
		{(au:0.88,fn:0.12):0.95,(pu:1):0.25,(su:1):0.2,(cg:1):0.13,(ln:1):0.15}	
		{(au:0.88,fn:0.12):0.95,(pu:1):0.25,(su:0.61,cg:0.39):0.33,(ln:1):0.15}	
2nd	(pu)	{(au:0.85,ln:0.15):0.98,(su:0.61,cg:0.39):0.33,(fn:1):0.11}	0.6
		{(au:0.88,fn:0.12):0.95,(su:0.61,cg:0.39):0.33,(ln:1):0.15}	
3rd	(su,cg)	{(au:0.85,ln:0.15):1,(fn:1):0.11}	1.0
		{(au:0.88,fn:0.12):0.96,(ln:1):0.15}	
4th	$\emptyset$		

Figure 7: Process of discovering schema model for the book domain.

domain	*vocabulary (abbreviation)*
books	title(ti),author(au),ISBN(is),keyword(kw),publisher(pu),subject(su),last name(ln), format(fm),category(cg),price(pr),first name(fn),publication date(pd)
movies	title(ti),director(dr),actor(ac),genre(gn),format(fm),category(cg), keyword(kw),rating(rt),price(pr),studio(sd),star(st),artist(at)
music records	artist(at),song(sg),album(ab),title(ti),label(lb),format(fm), genre(gn),soundtrack(sr),catalog #(ct),keyword(kw),band(bn)
automobiles	make(mk),model(md),price(pr),year(yr),type(tp),zip code(zc), mileage(ml),style(sy),color(cl),state(st),category(cg)

Figure 8: Vocabularies of the four domains.

Require: SchemaSet $\mathcal{I}$, SignificanceLevel a
1: **/* Initial Hypothesis Generation: */**
2: $\mathcal{V}$ = ATTRIBUTESELECTION($\bigcup I_i$)
3: C = CONSISTENTCONCEPTSCONSTRUCTION($\mathcal{I}$)
4: $\mathcal{H}$ = BUILDHYPOTHESISSPACE(C)
5: **/* Iterative Framework: */**
6: **while** true **do**
7: *conAttrs* = attributes in the consensus of $\mathcal{H}$
8: **if** *conAttrs* = $\emptyset$ or $\mathcal{V}$ = *conAttrs* **then**
9: **output** the initial models of $\mathcal{H}$
10: **else**
11: */* consensus projection */*
12: $\mathcal{V} = \mathcal{V} - conAttrs$; $\mathcal{I}^* = \pi_{\mathcal{V}}(\mathcal{I})$; $\mathcal{H}^* = \pi_{\mathcal{V}}(\mathcal{H})$
13: */* maximum likelihood estimation */*
14: **for** each $\mathcal{M}$ in $\mathcal{H}^*$ **do**
15: estimate parameters α, β of $\mathcal{M}$ using $\mathcal{I}^*$
16: **end for**
17: */* hypothesis selection: χ^2 testing+smoothing */*
18: *selectedH* = DYNAMICSELECTION($\mathcal{H}^*$)
19: */* new hypothesis space for next iteration*/*
20: $\mathcal{H}$ = *selectedH*
21: **end if**
22: **end while**

Figure 6: Algorithm MGS_{sd}.

(http://www.columbia.edu/cu/cria/SigTops/). Moreover, our preprocessing simply matched entities with obvious textual similarity (e.g., book title and title), which could be done with more sophisticated techniques such as [5].

In the experiments, we select the attributes using the approach proposed in Section 5.4.1 with threshold $f = 10\%$. The attributes passing that threshold are listed in Figure 8. Also, in the experiments we assume 0.05 as the significance level of χ^2 hypothesis testing. In practice, the threshold and significance level can be specified by users.

6.2 Metrics

We propose two suites of metrics for different purposes. The first suite is generic since it measures how the hypothesized schema model is close to the correct schema model written by human experts. The second suite of metrics is specific in the sense that it measures how good the hypothesized schema model can answer the target questions.

First, we introduce the notion of *correct schema model*. A correct schema model $\mathcal{M}_c$ is a schema model where attributes are correctly partitioned into concepts. Since it is difficult and unreliable (even for human experts) to specify the ideal probability parameters, we assign them using maximum likelihood estimation, which is consistent with the "unbias" and "sufficient" assumptions in Section 5.2.

The purpose of the first suite of metrics is to compare two models (or distributions). We view each distribution as a set of schemas (instantiated from that distribution), associated with a probability (or member frequency). Thus, we adopt precision and recall to measure this "member frequency". We define $Ins(\mathcal{M})$ as the set of all schemas that can be instantiated from $\mathcal{M}$. Precision is designed to measure the portion of the hypothesized set that is correct. In our case, the correct part is the intersection of $Ins(\mathcal{M}_h)$ and $Ins(\mathcal{M}_c)$, denoted by S. So the *model precision* is:

$$P_M(\mathcal{M}_h, \mathcal{M}_c) = \frac{\sum_{I \in S} Pr(I|\mathcal{M}_h)}{\sum_{I \in Ins(\mathcal{M}_h)} Pr(I|\mathcal{M}_h)} = \sum_{I \in S} Pr(I|\mathcal{M}_h),$$

where $\sum_{I \in Ins(\mathcal{M})} Pr(I|\mathcal{M}) = 1$ for any model $\mathcal{M}$. Similarly, *model recall* measures the portion of $\mathcal{M}_c$ that is contained in $\mathcal{M}_h$, which is $R_M(\mathcal{M}_h, \mathcal{M}_c) = \sum_{I \in S} Pr(I|\mathcal{M}_c)$.

Example 13: Consider Example 8, we can see that the correct schema model is actually $\mathcal{M}_1$ and thus both model precision and recall of $\mathcal{M}_1$ are 1.0. Now consider $\mathcal{M}_2$, although it is rejected, we still can measure it as an example. Example 8 has shown the schemas and instantiation probabilities of $Ins(\mathcal{M}_2)$ and $Ins(\mathcal{M}_c)$. So S contains four schemas: {subject}, {category}, {author, subject} and {author, category}. Then we can compute the model precision and recall as $P_M(\mathcal{M}_2, \mathcal{M}_c) = 0.196 + 0.036 + 0.294 + 0.054 = 0.58$ and $R_M(\mathcal{M}_2, \mathcal{M}_c) = 0.28 + 0.12 + 0.42 + 0.18 = 1$. ■

The second suite of metrics measures how the model is correct in answering the target questions. In our case, the target question is to ask for the synonyms of attributes. Specifically, we imagine there is a "random querier" who will ask for the synonyms of each attribute according to the probability of that attribute. The model will answer each question by returning the set of synonyms of the queried attribute in that model. We define $Syn(A_j|\mathcal{M})$ as the set of synonyms of attribute A_j in model $\mathcal{M}$. To compare two synonym sets, precision and recall are again applied. Given

domain	*output models*	P_M	R_M	P_T	R_T
movies	$\mathcal{M}_{movie1}$ = {(ti),(dr),(fm),(rt),(pr),(sd),(kw),(ac,st),(gn,cg),(at)}	0.94	1	1	0.88
	$\mathcal{M}_{movie2}$ = {(ti),(dr),(fm),(rt),(pr),(sd),(kw),(ac,st,at),(gn),(cg)}	0.96	1	1	0.88
	$\mathcal{M}_{movie3}$ = {(ti),(dr),(fm),(rt),(pr),(sd),(kw),(ac,st,at),(gn,cg)}	1	1	1	1
music records	$\mathcal{M}_{music1}$ = {(sg),(lb),(fm),(at,bn),(ab,ti),(gn),(sr),(kw),(ct)}	1	1	1	1
	$\mathcal{M}_{music2}$ = {(sg),(lb),(fm),(at,bn),(ab,ti),(gn),(sr),(kw,ct)}	1	0.99	0.94	1
	$\mathcal{M}_{music3}$ = {(sg),(lb),(fm),(at,bn),(ab,ti),(gn),(sr,kw),(ct)}	1	0.99	0.94	1
	$\mathcal{M}_{music4}$ = {(sg),(lb),(fm),(at,bn),(ab,ti),(gn,sr),(kw),(ct)}	1	0.98	0.93	1
	$\mathcal{M}_{music5}$ = {(sg),(lb),(fm),(at,bn),(ab,ti),(gn,sr),(kw,ct)}	1	0.97	0.86	1
automobiles	$\mathcal{M}_{auto}$ = {(mk),(md),(pr),(yr),(sy,tp,cg),(zc,cl),(st,ml)}	1	0.94	0.84	1

Figure 9: Experimental results for movies, music records and automobiles.

the correct model $\mathcal{M}_c$ and a hypothesized model $\mathcal{M}_h$, the precision and recall of the synonym sets of attribute A_j are:
$P_{A_j}(\mathcal{M}_h, \mathcal{M}_c) = \frac{|Syn(A_j|\mathcal{M}_c) \cap Syn(A_j|\mathcal{M}_h)|}{|Syn(A_j|\mathcal{M}_h)|}$ and
$R_{A_j}(\mathcal{M}_h, \mathcal{M}_c) = \frac{|Syn(A_j|\mathcal{M}_c) \cap Syn(A_j|\mathcal{M}_h)|}{|Syn(A_j|\mathcal{M}_c)|}$.

For this "random querier," more frequently observed attributes have higher probabilities to be asked. Thus we compute the weighted average of all the P_{A_j}'s and R_{A_j}'s as the *target precision* and *target recall*. The weight is assigned as a normalized probability of the attributes. That is, for attribute A_j, the weight $w_j = \frac{Pr(A_j|\mathcal{M})}{\sum_{A_j} Pr(A_j|\mathcal{M})} = \frac{\alpha_i \times \beta_j}{\sum_{A_j} \alpha_i \times \beta_j} = \frac{O_j}{\sum O_k}$ ($\alpha_i \times \beta_j = \frac{O_j}{|\mathcal{I}|}$ according to the formulae in Section 5.2.2). Therefore, *target precision* and *target recall* of $\mathcal{M}_h$ with respect to $\mathcal{M}_c$ are defined as:

$$P_T(\mathcal{M}_h, \mathcal{M}_c) = \sum_{A_j \in \mathcal{V}_h} \frac{O_j}{\sum O_k} P_{A_j}(\mathcal{M}_h, \mathcal{M}_c)$$
$$R_T(\mathcal{M}_h, \mathcal{M}_c) = \sum_{A_j \in \mathcal{V}_c} \frac{O_j}{\sum O_k} R_{A_j}(\mathcal{M}_h, \mathcal{M}_c),$$

where $\mathcal{V}_h$ and $\mathcal{V}_c$ are the vocabulary sets of $\mathcal{M}_h$ and $\mathcal{M}_c$.

Example 14: Still consider Example 8, the target precision and recall of $\mathcal{M}_1$ are both 1.0 since $\mathcal{M}_1$ is the correct schema model. For $\mathcal{M}_2$, we have $P_{\text{author}}(\mathcal{M}_2, \mathcal{M}_c) = 1$ and $R_{\text{author}}(\mathcal{M}_2, \mathcal{M}_c) = 1$ since author is correctly partitioned in $\mathcal{M}_2$. However, for subject, we have Syn(subject $|\mathcal{M}_c$) = {category} and Syn(subject $|\mathcal{M}_2$) = $\emptyset$. Therefore $P_{\text{subject}}(\mathcal{M}_2, \mathcal{M}_1) = 1$ and $R_{\text{subject}}(\mathcal{M}_2, \mathcal{M}_1) = 0$. We do the same measurement on category and then compute the weighted average. The occurrences of author, subject, and category are 9, 7, and 3 respectively. Thus, the results are $P_T(\mathcal{M}_2, \mathcal{M}_c) = \frac{9}{19} \times 1 + \frac{7}{19} \times 1 + \frac{3}{19} \times 1 = 1$ and $R_T(\mathcal{M}_2, \mathcal{M}_c) = \frac{9}{19} \times 1 + \frac{7}{19} \times 0 + \frac{3}{19} \times 0 = 0.47$. ■

6.3 Experimental Results

We report and discuss the experimental results for the book domain. For other domains, we only show the input and output. Figure 8 lists all the selected attributes. The result shows two sufficiently consistent models: $\mathcal{M}_{book1}$ = {(ti:1):.98, (is:1):.8, (kw:1):.56, (pr:1):.13, (fm:1):.13, (pd:1):.1, (pu:1):.25, (su:.61, cg:.39):.33, (au:.85, ln:.15):.98, (fn:1):.11} and $\mathcal{M}_{book2}$ = {(ti:1):.98, (is:1) :.8, (kw:1):.56, (pr:1):.13, (fm:1):.13, (pd:1):.1, (pu:1):.25, (su:.61, cg:.39):.33, (au:.88, fn:.12):.95, (ln:1):.15}.

The result successfully identifies the matchings (au, ln), (au, fn) and (su, cg). Without attribute grouping techniques (Section 5.1) to merge last name and first name, human experts can only consider that $\mathcal{M}_{book1}$ and $\mathcal{M}_{book2}$ both are correct schema models and thus give 1.0 precision and 1.0 recall in both model and target metrics. As stated in Section 5.1, attribute grouping is a different target question. Assume another specialized framework MGS_{ag} have done this task, then the result will be $\mathcal{M}_{book}$ = {(ti:1):.98, (is:1):.8, (kw:1):.56, (pr:1):.13, (fm:1):.13, (pd:1):.1, (pu:1):.25, (su:.61, cg:.39):.33, (au:.85, [ln,fn]:.15):.98)}, which is perfectly accurate in the sense of "equivalent synonym." In addition, the parameters in the results can be used to answer the question of concept popularity (Section 5.1), which indicates that this model is not limited to synonym discovery.

For the other three domains: movies, music records, and automobiles, their output is summarized in Figure 9. The results show that our approach can identify most concepts correctly. In movies and music records, the correct schema model is returned in our output models, which are $\mathcal{M}_{movie3}$ and $\mathcal{M}_{music1}$ respectively. However, for automobiles, we did not get the correct model. The incorrect matchings are due to the small number of observations we have. If we observe more sources, we should be able to observe some co-occurrences to remove false synonyms. For example, in the automobile domain, the incorrect matchings (zc, cl) and (st, ml) are because we did not observe the co-occurrences of zip code and color, state and mileage. With larger observation size, we believe the result will be better.

The measurement results in Figure 9 show that we do need two suites of metrics because they evaluate different aspects. For instance, the model recall of $\mathcal{M}_{movie1} = 1$ means $\mathcal{M}_{movie1}$ can generate all correct instances, while the target precision of $\mathcal{M}_{movie1} = 1$ denotes the synonyms answered by $\mathcal{M}_{movie1}$ are all correct ones.

Finally, although in principle the time complexity of MGS_{sd} is exponential, in our experiments, the overall execution time is within one minute (on a Pentium-III 700GHz with 128MB memory). Therefore, we believe that in practice the computation cost is likely to be acceptable for schema-matching as an off-line process.

7. DISCUSSION

In our study for statistical schema matching, we also observed some open issues that warrant further research. First, as inherent in statistical methods, our approach handles only frequent attributes with sufficient observations. (However, a more principled attribute selection may help to include rare attributes in popular concepts; see later.) Is it still useful then? Section 3 observed that these frequent attributes, while small in number, dominate 80% occurrences (because of the Zipf-like distribution). Thus, in the sense of the classic 80-20 rule, our technique is appealing in coping with at least the "vital few," although it leaves out the "trivial many" (that are likely unimportant). Further, as Section 4 mentioned, the "solvability" will scale; i.e., more attributes can be handled with more sources. Finally, we stress that, for large-scale integration of sources for numerous domains on the Web, automatic techniques like ours (even only for frequent attributes in each domain) will be imperative, as manual matching will not scale.

Second, can this approach deal with more expressive models for harder target questions, e.g., homonyms, hypernyms, or complex m:n matchings? We are currently studying hypernyms and complex matchings, for which we found the framework promising. However, there is a fundamental limitation: Our approach, relying on only attribute *syntactic* names (but not *semantics* in data contents), cannot distinguish homonyms. For sources created in the *same* domains, homonyms may be unlikely; we indeed found none in our study of Web query interfaces. For other scenarios when matching different domains, homonyms can be more significant (e.g., title in books and jobs). We may thus incorporate other techniques that consider more semantics (see below).

Third, can we reduce the exponential complexity of Algorithm MGS_{sd}? While such computation may be tolerable as schema matching is typically off-line and input attributes will likely converge (Section 5.5), cost reduction is clearly necessary. We believe that, within the MGS framework, cost reduction is possible by interleaving the model generation and selection phases to greedily construct, rather than exhaustively search, the best models. (Analogously, cost-based query optimization adopts, say, dynamic programming to speedup search in the plan space.)

Fourth, how can our framework integrate various techniques [16] for schema matching? We feel the statistical approach can benefit from complementary techniques that explore more "semantics" (e.g., data values for distinguishing homonyms). We believe the framework can integrate other techniques in a principled way, by "importing" their results as an *a priori* probabilistic structure for vocabulary $\mathcal{V}$ (currently structureless). Thus, say, a linguistic approach may indicate that author and writer are likely synonyms, with a score 0.7. Such vocabulary structure will then bias the statistical framework toward better results.

Fifth, does a hidden model always exist for a collection of schemas? Our approach hypothesizes such models. For sources of the same domains, this hypothesis seems empirically appealing. (Our recent study [10] clusters schemas into domains by such statistical models.) Can we relax to more liberal notion of "domains"? For instance, some sources only partially overlap in structure (e.g., automobiles and car rental) – Can they be coped with by a statistical approach?

Sixth, there are several "knobs" in MGS_{sd} that currently rely on thresholds. In particular, we use frequency thresholding (by 10%) for attribute selection, which might be too crude: It can prune rare attributes that are actually in frequent concepts (e.g., for books, binding and media type are pruned, while their synonym format is among the top concepts). Clearly, a more principled treatment will help in distinguishing true "outlier" attributes.

Finally, can we leverage massive sources to enhance the statistical approach? In principle, our approach relies on having sufficient observations. In practice, we have developed a suite of techniques (Section 5.4) for alleviating insufficiency. Think the other way: Can we leverage the virtually unlimited supply of sources on the Internet to help establish the required statistical context for schema matching? For instance, we may (automatically) collect many book sources, just to help in integrating similar ones.

8. CONCLUSION

This paper explores statistical schema matching, by hypothesizing and discovering hidden models that unify input schemas. Our experience indicates high promise for moving the traditional pairwise-attribute correspondence toward a new paradigm of holistic matching of massive sources. This approach is well suited for the new frontier of large-scale networked databases, such as our focus of the deep Web. We propose a general statistical framework MGS, and further specialize it to develop Algorithm MGS_{sd} for finding synonym attributes. Our extensive case studies motivated our approach as well as validated its effectiveness. We discussed several open issues, and we are eager to see how the statistical paradigm and framework can be generally applied in other schema matching scenarios.

Acknowledgments We are grateful to the Illinois Statistics Office at University of Illinois at Urbana-Champaign, for their discussions on developing the statistical techniques.

9. REFERENCES

[1] C. Batini, M. Lenzerini, and S. B. Navathe. A comparative analysis of methodologies for database schema integration. *ACM Computing Surveys*, 18(4):323–364, 1986.

[2] M. K. Bergman. The deep web: Surfacing hidden value. Technical report, BrightPlanet LLC, Dec. 2000.

[3] P. Bickel and K. Doksum. *Mathematical Statistics: Basic Ideas and Selected Topics*. Prentice Hall, 2001.

[4] K. C.-C. Chang, B. He, C. Li, and Z. Zhang. Structured databases on the web: Observations and implications. Report UIUCDCS-R-2003-2321, Dept. of Computer Science, UIUC, Feb. 2003.

[5] W. W. Cohen. Integration of heterogeneous databases without common domains using queries based on textual similarity. In *SIGMOD* 1998.

[6] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms (Section Edition)*. MIT Press, 2001.

[7] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum-likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society Series B*, 39:1–38, 1977.

[8] A. Doan, P. Domingos, and A. Y. Halevy. Reconciling schemas of disparate data sources: A machine-learning approach. In *SIGMOD* 2001.

[9] A. Halevy, O. Etzioni, A. Doan, Z. Ives, J. Madhavan, L. McDowell, and I. Tatarinov. Crossing the structure chasm. *Conf. on Innovative Database Research*, 2003.

[10] B. He, T. Tao, C. Li, and K. C.-C. Chang. Clustering structured web sources: A schema-based, model-differentiation approach. Report UIUCDCS-R-2003-2322, Dept. of Computer Science, UIUC, Feb. 2003.

[11] J. Larson, S. Navathe, and R. Elmasri. A theory of attributed equivalence in databases with application to schema integration. *IEEE Trans. on Software Engr.*, 16(4):449–463, 1989.

[12] C. J. Lloyd. *Statistical Analysis of Categorical Data*. Wiley, 1999.

[13] J. Madhavan, P. A. Bernstein, and E. Rahm. Generic schema matching with cupid. In *VLDB* 2001.

[14] S. Navathe and S. Gadgil. A methodology for view integration in logical data base design. In *VLDB* 1982.

[15] J. Ponte and W. Croft. A language modelling approach to information retrieval. In *SIGIR* 1998.

[16] E. Rahm and P. A. Bernstein. A survey of approaches to automatic schema matching. *VLDB Journal*, 10(4):334–350, 2001.

[17] L. Seligman, A. Rosenthal, P. Lehner, and A. Smith. Data integration: Where does the time go? *Bulletin of the Tech. Committee on Data Engr.*, 25(3), 2002.